A Resource-Sharing Game with Applications to Cloud-Computing

Paper # 19, ... pages

Abstract

Motivated by cloud-based computing resources operating with relative priorities, we investigate the strategic interaction between a fixed number of users sharing the capacity of a processor. Each user chooses a payment, which corresponds to his priority level, and submits jobs of variable sizes according to a stochastic process. These jobs have to be completed before some user-specific deadline. They are executed on the processor and receive a share of the capacity that is proportional to the priority level. The users' goal is to choose priority levels so as to minimize their own payment, while guaranteeing that their jobs meet their deadlines. We show that the equilibria in this game always coincides with the social welfare of the system. We fully characterize the solution of the game for two classes of users and exponential service times. For an arbitrary number of classes and general service times, we develop an approximation based on heavy-traffic. We characterize the solution of the game under the heavy-traffic assumption and we numerically investigate the accuracy of the approximation. Our results show that the approximate solution captures accurately the structure of the equilibrium in the original game.

---

1. Introduction

We are interested in the equilibria that arises in queueing games where a common resource is shared among multiple concurrent users. The study of strategic behavior in queueing systems has a long history and there is by now a broad literature, *cf.* [1] and [2] for monographs. A particular problem who has received a lot of attention deals with the strategic behavior of users in parallel servers, see for example [3, 4, 5]. In recent years, motivated by the rise of paid resource sharing systems like in cloud computing, researchers have started to investigate pricing schemes, where capacity of the server is shared simultaneously by all jobs present in the system, see for example [6] or [7]. For the case in which the the underlying queueing model has no priorities we refer to [8] and [9].

We analyse the equilibria in a more complex scenario where users may arrive at random and leave the system after getting service, and when the capacity allocated to each user is a function of the prices. More precisely, we assume that the capacity is shared according to the Discriminatory Processor Sharing (DPS) discipline, as they do in [10] and [11]. Introduced by [12], the DPS model is a multi-class generalization of the *egalitarian* Processor Sharing (PS) queue that captures the essential features of a system that implements service differentiation (see [13] for a survey). As we will see later in Section 2 the analysis of this model is extremely challenging, and as a consequence results are scarce.

Our model captures the fundamental properties of the Infrastructure-as-a-Service (IaaS) cloud-computing platforms that are based on priority level differentiation. For instance, in Amazon EC2 cloud the users can bid for unused capacity using the so-called Spot Instances [14]. Amazon fixes the Spot Price which depends on the capacity demand of the users and the available resources. On the other hand, most file hosting web providers dispose different premium subscriptions which let the users increase the speed of download/upload. We observe that in both instances a higher payment leads to a higher speed of service and that our model also satisfies this property.

In this model, each user submits jobs of variable sizes according to a stochastic process and chooses a priority level so that the processor capacity is shared in proportion to the priority level of all jobs being executed. The higher the priority level chosen by the user, the higher the cost he will have to pay for the

Contributions	Original Game		Heavy-Traffic Game	
	Number of Classes	Service Times	Number of Classes	Service Times
Feasibility	Arbitrary	Exponential	Arbitrary	General
Existence of NE	Arbitrary	General	Arbitrary	General
Uniqueness of NE	2	General	Arbitrary	General
Price of Anarchy	2	General	Arbitrary	General
NE Characterization	2	Exponential	Arbitrary	General

Table 1: Summary of the main contributions of the article.

execution of his job on the processor. All jobs in the system receive simultaneous service with a share of the capacity, that is proportional to the priority level of the user. The users' goal is to choose priority levels so as to minimize their own payment, while guaranteeing that the probability of their jobs not meeting their deadlines is below some threshold.

A central difficulty in the analysis of the equilibria of this game comes from the absence of a closed-form expression for the mean processing times of the jobs in the system. For example, the mean unconditional sojourn time in a DPS queue is only known in the case of two classes with exponentially distributed service requirements. It is thus not surprising that results on strategic behavior under this type of systems are scarce. For example, in [10] the authors consider two types of applications in a DPS queue that compete to be served and they analyse how optimal prices can be found. A more recent work is [11], where the authors define a game for the DPS queue where each user seeks to minimize the sum of the expected processing cost and payment. Given the difficulty in analysing the model, the authors propose a heavy-traffic approximation of the problem. Even though in our work also consider the heavy-traffic approximation, we analyse a different game where each user minimizes its payment ensuring its jobs to be served before a certain deadline.

The main contributions of the article are summarized in table 1. We give the necessary and sufficient conditions for the existence of the equilibrium of the game for exponential service times and arbitrary number of classes. For general service times and two classes of users, we show that the equilibrium is unique and that the Price of Anarchy is one. When the number of classes is two and exponential service times, we characterize the unique equilibrium of the game. For the rest of the cases, given the difficulty of this model, we use results from [15] of the heavy-traffic theory to obtain tractable expressions for the mean response time in the system. Even though of approximate nature, we believe that the heavy-traffic approach allows to derive interesting insights into the performance of the system. Using the heavy-traffic approximation, we characterize the sufficient and necessary conditions for the game to have a Nash equilibrium, and then show that this equilibrium is unique and fully characterize it. Interestingly, we show that classes can be ordered in a decreasing order with respect to the ratio between the mean size requirement and their constraints on the response time and that in equilibrium, the prices that users pay decrease as this ratio decreases. Furthermore, we prove that the Price of Anarchy of the heavy-traffic game is always one. We then explain how the heavy-traffic solution can be used to obtain an approximate solution to the original problem. The numerical experiments illustrate that when the various classes have a similar ratio between the mean size and response time constraint, then the heavy-traffic approximation predicts satisfactorily the outcome (both in terms of equilibrium weights and performance) of the original game. However, an interesting situation arises when the disparity of the users increases or when the original game becomes infeasible far from the saturation point. In this case, the error in predicting the equilibrium weights can be very significant, but in spite of this, the heavy-traffic approximation captures very accurately the structure of the equilibrium.

We consider that our paper will represent a step further in the difficult area of analysis of time-sharing systems. Indeed, in our model we use the pricing technique to guarantee certain performance requirements of the users. Further, we discuss the applicability of our results for the case of interactive and CPU-intensive jobs executed in a virtualized environment. We believe that the presented results give significant insights of the pricing techniques of the users of cloud-based architectures that operate with relative priorities.

The rest of the paper is organized as follows. In Section 2 we describe the model. We present the game with constraints on the mean response time in Section 3. In Section 4 we analyse the game for the heavy-traffic regime and in Section 5 we study the game for an arbitrary load of the system. In Section 6.3 we present a distributed algorithm that converges to the equilibrium of the game. We discuss the accuracy of our approximation using the numerical experiments of Section 6. Finally, in Section 7 we summarize the main conclusions of this paper.

2. Game description

Consider a game in which a single server of unit capacity is shared among R classes (or users). Let $\mathcal{C} = \{1, 2, \ldots, R\}$ be the set of classes. We assume that the arrival process of jobs of each class i is Poisson with rate λ_i and that the service requirements of jobs are i.i.d. and have an arbitrary distribution with mean $\mathbb{E}(B_i)$ and second moment $\mathbb{E}\left(B_i^2\right)$. For the case of exponential service time distributions, we will use the notation $\mathbb{E}(B_i) = \mu_i^{-1}$ and $\mathbb{E}\left(B_i^2\right) = 2/\mu_i^2$. We define the total incoming traffic of the system by $\lambda = \sum_{i=1}^R \lambda_i$. Let $\rho_i = \lambda_i \mathbb{E}(B_i)$ be the load of class i and the total load of the system be $\rho = \sum_{i=1}^R \rho_i$.

The processing capacity of the server is shared amongst jobs according to the DPS discipline, that is, all jobs present in the system are served simultaneously at rates controlled by a vector of weights $\boldsymbol{g} = (g_1, \ldots, g_R)$. If there are N_i jobs of class i present in the system, then class-i jobs are served at rate

$$r_i(N_1, \ldots, N_R) = \frac{g_i}{\sum_{j=1}^R g_j N_j}. \tag{1}$$

When all the weights are equal, DPS is equivalent to the ordinary PS discipline. By changing the weights, one can effectively control the instantaneous service rates of different job classes. For example, by setting the weight of a class close to infinity, one can give preemptive priority to this class. The possibility of providing different service rates to users of various classes makes DPS an appropriate model to study the performance of heterogeneous time-sharing systems. We note that a direct consequence of (1) is that the service rate every class gets for a vector $\theta\,\boldsymbol{g}$ is independent of the common factor $\theta > 0$.

We describe our game formulation in Subsection 2.2. Prior to that, we briefly mention the main results on DPS that we need in this paper.

2.1. Main results on DPS

We denote by $T_i(\boldsymbol{g}; \rho)$ the random variable corresponding to the response time of a class-i job in a DPS queue for the vector of weights $\boldsymbol{g} = (g_1, \ldots, g_R)$ when the load in the system is $\rho < 1$. The mean response time is denoted by $\overline{T}_i(\boldsymbol{g}; \rho) = \mathbb{E}(T_i(\boldsymbol{g}; \rho))$.

In a seminal paper, Fayolle et al. proved that for exponential service time distributions, the mean response time is the solution of a system of equations. For completeness we state their result:

Proposition 1 ([16]). *In the case of exponentially distributed required service times, the unconditional average response times satisfy the following linear system of equations:*

$$\overline{T}_k(\boldsymbol{g}; \rho) \left(1 - \sum_{j=1}^R \frac{\lambda_j g_j}{\mu_j g_j + \mu_k g_k} \right) - \sum_{j=1}^R \frac{\lambda_j g_j \overline{T}_j(\boldsymbol{g}; \rho)}{\mu_j g_j + \mu_k g_k} = \frac{1}{\mu_k}, \tag{2}$$

with $k = 1, \ldots, R$.

A solution to this system of equations is only known for the case $R = 2$. In this case the solution is:

$$\overline{T}_1(\boldsymbol{g}; \rho) = \frac{1}{\mu_1(1-\rho)} \left(1 + \frac{\mu_1 \rho_2 (g_2 - g_1)}{\mu_1 g_1 (1-\rho_1) + \mu_2 g_2 (1-\rho_2)} \right), \tag{3}$$

and

$$\overline{T}_2(\boldsymbol{g};\rho) = \frac{1}{\mu_2(1-\rho)}\left(1 + \frac{\mu_2\rho_1(g_1-g_2)}{\mu_1 g_1(1-\rho_1)+\mu_2 g_2(1-\rho_2)}\right). \tag{4}$$

For general service time distributions the results are scarce. In [16] the authors showed that the derivative of the mean conditional (on the service requirement) response time of the various classes satisfies a system of integro-differential equations. Unfortunately a closed-form solution of this system of equations has been obtained only in the case of exponential distributions. To the best of our knowledge, there is no known tractable results on the distribution of the response time $T_i(\boldsymbol{g};\rho)$.

To overcome this difficulty, in our approach we will approximate $T_i(\boldsymbol{g};\rho)$ using a heavy-traffic characterization. It turns out that the scaled response time $(1-\rho)T_i(\boldsymbol{g};\rho)$ has a proper distribution as $\rho \to 1$. The DPS queue in heavy-traffic was first considered in [15] (see also [17] and [18]). The result we require reads:

Proposition 2 ([15]). *When scaled with $1-\rho$, the response time of class-i jobs has a proper distribution as $\rho \to 1$.*

$$(1-\rho)\; T_i(\boldsymbol{g};\rho) \xrightarrow{d} T_i(\boldsymbol{g};1) = X \cdot \frac{\mathbb{E}(B_i)}{g_i}, \quad i \in \mathcal{C}, \tag{5}$$

where $\xrightarrow{d}$ denotes convergence in distribution and X is an exponentially distributed random variable with mean

$$E(X) = \frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)\frac{1}{g_k}}. \tag{6}$$

Proposition 2 implies that for sufficiently high load, the response time distribution in a DPS queue can be approximated by an exponential random variable, that is,

$$T_i(\boldsymbol{g};\rho) \approxeq \frac{T_i(\boldsymbol{g};1)}{1-\rho} \stackrel{d}{=} \frac{\mathbb{E}(B_i)}{g_i(1-\rho)}X, \tag{7}$$

and for the mean response time we obtain that

$$\overline{T}_i(\boldsymbol{g};\rho) \approx \frac{\mathbb{E}(B_i)}{g_i(1-\rho)}\frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)\frac{1}{g_k}}. \tag{8}$$

In the above derivation, we have ignored a technical subtlety. Indeed, in order for (8) to be valid, one needs to establish that the heavy-traffic limit and expectation can be interchanged, namely, $\lim_{\rho\to 1}\overline{T}_i(\boldsymbol{g};\rho) = \mathbb{E}(\lim_{\rho\to 1} T_i(\boldsymbol{g};\rho))$. In [18] the authors performed numerical experiments to validate the validity of this interchange. In the rest of the paper we will assume that the interchange is valid.

In the case of identical weights g_i, the DPS queue is equivalent to the well-known *egalitarian* PS, which has been thoroughly studied, see for example [19] or [20]. For PS, it holds that $\overline{T}_i(\boldsymbol{g};\rho) = \mathbb{E}(B_i)/(1-\rho)$. From (6) and (5) we get that $\overline{T}_i(\boldsymbol{g};1) = \mathbb{E}(B_i)$, and it follows that the approximation $\overline{T}_i(\boldsymbol{g};\rho) = \frac{\overline{T}_i(\boldsymbol{g};1)}{1-\rho}$ is exact.

We can now introduce the game formulation we study in the paper.

2.2. Game formulation

We assume that the service provider (or the server) proposes to each class $i \in \mathcal{C}$ the choice of its weight g_i in exchange of a payment per-unit-of-work proportional to the chosen weight. The quality-of-service metric of class i is the probability of its jobs missing a given deadline d_i. Class i then wants to ensure that this probability is below a certain threshold $\alpha_i \in (0,1)$ while paying as little as possible for this service. Formally, class-i solves the problem

$$\begin{aligned} \min_{g_i \geq \epsilon} \quad & \rho_i g_i \\ \text{subject to} \quad & \mathbb{P}\left(T_i(\boldsymbol{g};\rho) > d_i\right) \leq \alpha_i. \end{aligned} \tag{OPT-P}$$

The quantity ϵ is the minimum price a class has to pay in order to get access to the service.

As explained in Subsection 2.1 the probability of jobs missing a deadline in a DPS queue has no easy-to-compute closed-form expression. One could then consider a game in which the constraints are based on the mean response time of tasks. The optimization problem above then gets modified as follows

$$\begin{aligned} \min_{g_i \geq \epsilon} \quad & \rho_i g_i \\ \text{subject to} \quad & \overline{T}_i(\boldsymbol{g}; \rho) \leq c_i, \end{aligned} \tag{OPT-M}$$

for $i \in \mathcal{C}$.

The modified game (OPT-M) is not completely unrelated to the original game (OPT-P) as we shall argue next. Assuming the load is high enough, we invoke the heavy-traffic approximation so:

$$\mathbb{P}\left(T_i(\boldsymbol{g}; \rho) > d_i\right) \quad = \quad \mathbb{P}\left(T_i(\boldsymbol{g}; 1) > (1-\rho) d_i\right) = e^{-\frac{(1-\rho)d_i}{\overline{T}_i(\boldsymbol{g};1)}},$$

implying that

$$\mathbb{P}\left(T_i(\boldsymbol{g}; \rho) > d_i\right) \leq \alpha_i \iff -\frac{(1-\rho)\, d_i}{\overline{T}_i(\boldsymbol{g}; 1)} \leq \log \alpha_i.$$

Since $\alpha_i \in (0,1)$, we have $\log \alpha_i < 0$ and, hence, we obtain the following equivalent constraint $\overline{T}_i(\boldsymbol{g}; 1) \leq \tilde{c}_i = -\frac{(1-\rho)d_i}{\log \alpha_i}$.

Thus, we propose to use the heavy-traffic result given in Proposition 2 as an approximation to (OPT-P) and (OPT-M). We obtain the problem

$$\begin{aligned} \min_{g_i \geq \epsilon} \quad & \rho_i g_i \\ \text{subject to} \quad & \overline{T}_i(\boldsymbol{g}; 1) \leq \tilde{c}_i. \end{aligned} \tag{OPT-HT}$$

In the case $\tilde{c}_i = -\frac{(1-\rho)d_i}{\log \alpha_i}$ we will be approximating (OPT-P), and if $\tilde{c}_i = (1-\rho)c_i$ we will be approximating (OPT-M).

This approximation has the advantage that one can compute the scaled mean response time of a DPS queue in heavy-traffic $\overline{T}_i(\boldsymbol{g}; 1)$ for any service time distribution with finite second moment.

Our hope is that the solution of the game (OPT-HT) will give useful insights into the equilibrium properties of (OPT-P) and (OPT-M). We emphasize that the benefit of the heavy-traffic approximation is that the mean response time formulae have a nice closed-form expressions even for general service time distributions whereas (OPT-M) has a simple structure only in case of exponentially distributed service times, while (OPT-P) does not appear to be tractable even for that case. In Section 6 we investigate the accuracy of the approximation, and show that it always gives us the structure of the equilibrium and our approach is accurate when the users have similar mean size and mean service time characteristics.

Before going further, we give some definitions.

Definition 1 (Achievability). *A vector* $\mathbf{t}$ *of mean response times is said to be achievable if there exists a vector of weights* $\boldsymbol{g} > 0$ *for which the vector of mean response times is* $\mathbf{t}$*, i.e.,* $t_i = \overline{T}_i(\boldsymbol{g}; \rho)$ *for all* $i \in \mathcal{C}$*.*

Definition 2 (Deadline feasibility). *Let* $\mathcal{T} = \{\mathbf{t} : \mathbf{t}$ *is achievable*$\}$*. A vector of deadlines* $\boldsymbol{c} \in \mathbb{R}_+^R$ *is feasible if and only if*

$$\exists \mathbf{t} \in \mathcal{T} : \mathbf{t} \preceq \boldsymbol{c}, \tag{9}$$

where $\preceq$ *is the componentwise order.*

In the following, we say that a game is feasible if its vector of deadlines is feasible. We will also use the notion of a feasible weight vector, as defined below.

Definition 3 (Weight feasibility). *A vector of weights $\boldsymbol{g} \in \mathbb{R}_+^R$ is feasible if and only if $\overline{T}_i(\boldsymbol{g}, \rho) \leq c_i$ for all $i \in \mathcal{C}$.*

Finally, the last definition we will need is the following.

Definition 4. *A class i will be considered* fair *if $\mathbb{E}(B_i)/c_i \leq (1-\rho)$.*

In other words, a class i is fair if the response time it would obtain under PS, $\mathbb{E}(B_i)/(1-\rho)$, would satisfy its own constraint on the mean performance c_i. As we will see in Section 5, a sufficient condition for the PS weights to be the equilibrium is that all classes be fair.

Without loss of generality, we assume that the classes are ordered in decreasing order of $\mathbb{E}(B_k)/c_k$, i.e., if $i < j$, then $\mathbb{E}(B_i)/c_i \geq \mathbb{E}(B_j)/c_j$. We observe that the ratio $\mathbb{E}(B_k)/c_k$ is the maximum throughput of a class-i job with a service requirement equal to the mean. In the case of exponential service time distribution, it becomes $c_1\mu_1 \leq c_2\mu_2 \leq \cdots \leq c_R\mu_R$.

3. Solution of (OPT-M)

This section is devoted to the analysis of the game (OPT-M). We assume throughout this section that service times are exponentially distributed. Nevertheless, some of our results hold for arbitrary service time distributions, and this shall be indicated explicitly. We first establish in Section 3.1 a necessary and sufficient condition for the game (OPT-M) to be feasible. Assuming that the game is feasible, we then prove in Section 3.2 that there exist at least one Nash equilibrium, that is a point where no user has an incentive to unilaterally deviate and change his weight. We then study the uniqueness of the Nash equilibrium in Section 3.3. We provide an explicit characterization of the Nash equilibrium for the two-player game in Section 3.4. Finally, we address the question of the inefficiency of the Nash equilibrium from a user's perspective in Section 3.5.

3.1. Feasibility of the Game

For fixed traffic conditions, the game is feasible if the vector $\boldsymbol{c}$ of deadlines is such that there is an achievable vector $\mathbf{t}$ of performances such that $t_i \leq c_i$ for all $i \in \mathcal{C}$. For exponential service times, the set of achievable vectors for the DPS queue was characterized by Mitrani and Fayolle [21]. In order to present their result, we first need to introduce some notations. Let $\mathcal{R} = \mathcal{P}(\mathcal{C}) \setminus \emptyset$, where $\mathcal{P}(\mathcal{C})$ is the power set of $\mathcal{C}$, be the set of all subsets of $\mathcal{C}$ except the empty set. We define $\bar{\rho}_r = \sum_{i \in r} \rho_i$, and

$$W_r = \frac{1}{1-\bar{\rho}_r} \sum_{i \in r} \frac{\rho_i}{\mu_i}, \tag{10}$$

for all $r \in \mathcal{R}$. With these notations, the result of Mitrani and Fayolle reads as follows. A vector $\mathbf{t}$ of performances is achievable if and only if

$$\sum_{i \in \mathcal{C}} \rho_i t_i = W_{\mathcal{C}}, \tag{11}$$

$$\sum_{i \in r} \rho_i t_i \geq W_r, \ \forall r \in \mathcal{R} \setminus \{\mathcal{C}\}. \tag{12}$$

The following result gives a necessary and sufficient condition for a vector $\boldsymbol{c}$ to be feasible.

Theorem 1. *Assuming exponential service times, a vector* $\boldsymbol{c}$ *is feasible if and only if*

$$\sum_{i \in r} \rho_i c_i \geq W_r, \ \forall r \in \mathcal{R}. \tag{13}$$

Proof. See Appendix A.1. ■

3.2. Existence of Nash Equilibrium

Assuming that the game is feasible, a vector of weights $\boldsymbol{g}^{NE} = (g_1^{NE}, \ldots, g_R^{NE})$ is a Nash equilibrium (NE) for the game (OPT-M) if each class is paying the least possible amount while ensuring that its mean response time does not exceed its deadline. Thus, we can say that a vector of weights $\boldsymbol{g}^{NE}$ is a Nash equilibrium if for all $i = 1, \ldots, R$

$$g_i^{NE} = \text{argmin} \left\{ g_i \geq \epsilon \ : \ \overline{T}_i(g_i, \boldsymbol{g}_{-i}^{NE}; \rho) \leq c_i \right\},$$

where $\boldsymbol{g}_{-i}^{NE} = \left(g_1^{NE}, \ldots, g_{i-1}^{NE}, g_{i+1}^{NE}, \ldots, g_R^{NE}\right)$. In [22] the authors showed that $\overline{T}_i(\boldsymbol{g}; \rho)$ is decreasing with g_i and increasing in g_j for $j \neq i$. It then follows that, for a given i,

$$g_i^{NE} > \epsilon \quad \Rightarrow \overline{T}_i(\boldsymbol{g}^{NE}; \rho) = c_i, \tag{14}$$

$$g_i^{NE} = \epsilon \quad \Rightarrow \overline{T}_i(\boldsymbol{g}^{NE}; \rho) \leq c_i. \tag{15}$$

Since $\overline{T}_i(\boldsymbol{g}; \rho)$ is decreasing in g_i, a class which is paying more than ϵ is necessarily satisfying its constraint with equality. Otherwise, if it were to be satisfying the constraint with strict inequality, then it would pay less and still satisfy its deadline. On the other hand, a class which is paying the least possible price could be satisfying its deadline with strict inequality.

We notice that the dynamics of best-response are given by increasing the weight of class i when $\overline{T}_i(\boldsymbol{g}; \rho) > c_i$ and decreasing the weight of class i when $\overline{T}_i(\boldsymbol{g}; \rho) < c_i$ and $g_i > \epsilon$. Assume that we start the best-response dynamics from a feasible point $\boldsymbol{g}$. If all constraints $\overline{T}_i(\boldsymbol{g}; \rho) \leq c_i$ are satisfied as equality constraints (implying that the deadline vector $\boldsymbol{c}$ is achievable), then $\boldsymbol{g}$ is clearly a Nash equilibrium since no class can unilaterally decrease its weight and still satisfy its constraint. If on the contrary there is a nonempty subset $\mathcal{A} \subset \mathcal{C}$ such that $\overline{T}_i(\boldsymbol{g}; \rho) < c_i$ for all classes $i \in \mathcal{A}$, then we have either $g_i = \epsilon$ for all $i \in \mathcal{A}$ or there are some classes $i \in \mathcal{A}$ such that $g_i > \epsilon$. In the former case, $\boldsymbol{g}$ is again an equilibrium since clearly no class can decrease its weight. In the latter case, the best-response for each class $i \in \mathcal{A}$ such that $g_i > \epsilon$ is to decrease its weight. Moreover, after each best-response, the current vector of weights remains feasible because by decreasing its weight a class can only improve the mean response times of the other classes. Thus, in that case the best-response dynamics generate a sequence of feasible weight vectors which is strictly decreasing in the lexycographic order. Since feasible weight vectors belong to the set $[\epsilon, \infty)^R$ which is closed on the left, we can conclude that the dynamics of best-response converge to a Nash Equilibrium.

Proposition 3. *The best-response dynamics converge to a Nash Equilibrium when started from a feasible point.*

From Proposition 3, we immediately obtain the following corollary.

Corollary 1. *If the game is feasible, there exists a Nash equilibrium.*

We emphasize that Corollary 3 hold whatever the service time distributions of the users.

3.3. Uniqueness of the Nash Equilibrium

If the game (OPT-M) is feasible, there exists at least one Nash equilibrium. We now investigate the question of the number of equilibria. We first consider the case when the deadline vector $\boldsymbol{c}$ is achievable and obtain the following negative result.

Proposition 4. *If the deadline vector* $\mathbf{c}$ *is achievable, then there is an infinite number of equilibria.*

Proof. If $\boldsymbol{c}$ is achievable, there exists a weight vector $\boldsymbol{g}$ such that $\overline{T}_i(\boldsymbol{g};\rho) = c_i$ for all $i \in \mathcal{C}$. This weight vector is an equilibrium since no class can decrease its weight and still satisfy its constraint. To conclude the proof, it is enough to observe that the weight vector $\theta\,\boldsymbol{g}$ is such that $\overline{T}_i(\theta\,\boldsymbol{g};\rho) = c_i$ for all $i \in \mathcal{C}$ and is thus an equilibrium for any value of $\theta \geq \min\left(\frac{\epsilon}{g_1}, \ldots, \frac{\epsilon}{g_R}\right)$. ■

We now consider the case where $\boldsymbol{c}$ is not achievable. Proposition 5 below shows that in that case any two-player game has a unique Nash equilibrium.

Proposition 5. *For a two-player game such that the deadline vector* $\mathbf{c}$ *is not achievable, there is a unique Nash equilibrium.*

Proof. See Appendix A.2. ■

3.4. Characterization of the Equilibrium

Explicit expressions of the mean response times in a DPS queue are known only in the case of two classes and exponential service times (see (3) and (4)). This restricts the set of cases in which an explicit solution to the game can be computed.

Proposition 6. *For the two-player game with exponential service times, if the game is feasible and* $\mathbf{c}$ *is not achievable, then the unique equilibrium* $\mathbf{g^{NE}}$ *is such that:*

- *if class 1 is fair, i.e.* $(1-\rho)^{-1} \leq c_1\mu_1$*, then* $\mathbf{g^{NE}} = (\epsilon, \epsilon)$*,*
- *otherwise,* $\mathbf{g^{NE}} = (g_1^{NE}, \epsilon)$*, where* $g_1^{NE} = \epsilon\ \frac{-\mu_1\rho_2 + \mu_2(1-\rho_2)[\mu_1 c_1(1-\rho)-1]}{-\mu_1\rho_2 - \mu_1(1-\rho_1)[\mu_1 c_1(1-\rho)-1]}$.

Proof. See appendix A.3. ■

We explain briefly the solution of the problem. Assuming feasibility, at least class 2 is fair. If class 1 is also fair, then $(g_1, g_2) = (\epsilon, \epsilon)$ is the equilibrium; however, if the mean response time of class 1 for PS weights exceeds its deadline c_1, the class 1 must pay $g_1 > \epsilon$ per unit-of-work to ensure that its time constraint is satisfied.

We also show that the dynamics of the best-response converge to the Nash Equilibrium.

Proposition 7. *If the game is feasible, the best-response dynamics converge to the Nash Equilibrium for exponential service times and two classes of users.*

Proof. See appendix A.4. ■

3.5. Price of Anarchy

We define the social welfare (or social optimum) of the system as the strategy of the users such that the total payment is minimum. It is the vector of weights that solves the following minimization problem:

$$\begin{aligned} \min_{(g_1,\dots,g_R)} \quad & \sum_{i=1}^{R} \rho_i g_i \\ \text{subject to} \quad & \overline{T_i}(\boldsymbol{g};\rho) \le c_i, \text{for all } i = 1,\dots,R, \\ \text{and} \quad & g_i \ge \epsilon, \text{for all } i = 1,\dots,R. \end{aligned} \tag{SOC-OPT}$$

The main difference with respect to the game is that in the latter each user minimizes its own payment while in the social optimum the users coordinate to choose the weights that minimize the total payment. By its very definition, the total payment at the social optimum cannot be larger than that at a Nash equilibrium.

The sub-optimality of the game (OPT-M) can be measured using the notions of Price of Stability (PoS) and Price of Anarchy (PoS) which are defined as:

$$PoS = \min_{\boldsymbol{g}\in\mathcal{G}_M} \frac{\sum_{i=1}^{R} \rho_i g_i}{\sum_{i=1}^{R} \rho_i g_i^{SOC}}, \tag{16}$$

$$PoA = \max_{\boldsymbol{g}\in\mathcal{G}_M} \frac{\sum_{i=1}^{R} \rho_i g_i}{\sum_{i=1}^{R} \rho_i g_i^{SOC}}, \tag{17}$$

where $\mathcal{G}_M$ denotes the set of Nash equilibria of (OPT-M) and $\boldsymbol{g}^{SOC}$ is any vector of weights that is socially optimal. From these definitions, it follows that $PoA \ge PoS \ge 1$, and $PoA = PoS$ in particular when the Nash equilibrium is unique.

Let $\boldsymbol{g}^{SOC}$ be a social optimum. If it would exist i such that $g_i^{SOC} > \epsilon$ and $\overline{T}_i(\boldsymbol{g}^{SOC};\rho) < c_i$, then it would be possible to decrease g_i^{SOC} while still satisfying the constraint $\overline{T}_i(\boldsymbol{g}^{SOC};\rho) \le c_i$, implying that $\boldsymbol{g}^{SOC}$ would not be the solution of (SOC-OPT). We thus conclude that any social optimum is a vector of weights $\boldsymbol{g}^{SOC}$ such that each component verifies one of the following equations:

$$\text{if } g_i^{SOC} > \epsilon, \quad \Rightarrow \overline{T}_i(\boldsymbol{g}^{SOC};\rho) = c_i, \tag{18}$$

$$\text{if } g_i^{SOC} = \epsilon, \quad \Rightarrow \overline{T}_i(\boldsymbol{g}^{SOC};\rho) \le c_i. \tag{19}$$

Equations (18) and (19) give the necessary conditions for a vector to be the social optimum. They are, in fact, the same as (14) and (15) which are the necessary and sufficient conditions for a vector to be a Nash equilibrium. It then follows that

Proposition 8. *A social optimum is also a a Nash Equilibrium.*

An immediate consequence of this result is that the PoS is 1 for the DPS game. Moreover, from Proposition 5 it follows that:

Corollary 2. *For two classes of users and general service times, $PoA = 1$.*

4. Solution of (OPT-HT)

In this section we investigate the solution of the the game (OPT-HT). Even though some of the results follow using the same argument that in Section 3, we emphasize that the results of this section hold for general service times and an arbitrary number of players. In Section 4.1 we give a necessary and sufficient condition for the feasibility of the game (OPT-HT). Assuming this condition hold, we focus on the existence of a Nash equilibrium in Section 4.2. We then consider the uniqueness of the equilibrium and explicitly characterize it when it is unique in Section 4.3. Finally, we study the inefficiency of the equilibrium using the concept of Price of Anarchy in Section 4.4.

4.1. Feasibility of the Game

Before presenting our results on the feasibility of the game (OPT-HT), let us first characterize the achievability in heavy-traffic. A vector of performance is achievable in heavy-traffic if there exists a vector of weights $\boldsymbol{g} > 0$ for which $\overline{T}_i(\boldsymbol{g};1) = t_i$, for all $i \in \mathcal{C}$, where $\overline{T}_i(\boldsymbol{g};1)$ is the mean response time in heavy-traffic of a class-i job which is given by

$$\overline{T}_i(\boldsymbol{g};1) = \frac{\mathbb{E}(B_i)}{g_i} \frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)\frac{1}{g_k}}. \tag{20}$$

The following proposition characterizes the achievability of a vector of mean response times:

Proposition 9. *A vector of performances $(t_1, \ldots, t_R)$ is achievable in heavy-traffic if and only if*

$$\sum_{k=1}^{R} \lambda_k \frac{\mathbb{E}(B_k^2)}{\mathbb{E}(B_k)} t_k = \sum_{j=1}^{R} \lambda_j \mathbb{E}\left(B_j^2\right). \tag{21}$$

Proof. See appendix B.1 ■

Remark 1. *In the case $\rho < 1$, in [21] the authors characterized the achievable region of DPS under exponential service requirements. It can be easily checked, than the vector space obtained in [21] converges to the hyperplane* (21) *as the load increases to* 1*.*

We now give a sufficient and necessary condition for the game (OPT-HT) to be feasible.

Proposition 10. *The game* (OPT-HT) *is feasible if and only if*

$$\sum_i \lambda_i \mathbb{E}\left(B_i^2\right)\left(\frac{\tilde{c}_i}{\mathbb{E}(B_i)} - 1\right) \geq 0.$$

Proof. See appendix B.2. ■

Interestingly, we observe that a sufficient condition for the game to be feasible is that in heavy-traffic all classes be *fair*. Note that $\overline{T}_i(\boldsymbol{g}^{PS};1) = \mathbb{E}(B_i)$, thus from Proposition 10 if $\overline{T}_i(\boldsymbol{g}^{PS};1) \leq \tilde{c}_i$, $\forall i$, then the game is feasible.

4.2. Existence of the Nash Equilibrium

A vector of weights $\boldsymbol{g}^{NE}$ is a Nash equilibrium for(OPT-HT) if for all $i \in \mathcal{C}$

$$g_i^{NE} = \text{argmin}\left\{g_i \geq \epsilon \ : \ \overline{T}_i(g_i, \boldsymbol{g}_{-i}^{NE};1) \leq \tilde{c}_i\right\},$$

where $\boldsymbol{g}_{-i}^{NE} = \left(g_1^{NE}, \ldots, g_{i-1}^{NE}, g_{i+1}^{NE}, \ldots, g_R^{NE}\right)$. We observe from (20) that the mean response time in heavy-traffic of a class-i job is decreasing with g_i and increasing with g_j, for all $j \neq i$. Using the same reasoning as in Section 3.2 we conclude that each component of the equilibrium of this game satisfies (14) or (15). With exactly the same arguments as in Section 3.2 for the game (OPT-M), we can also prove that the best-response dynamics converge to a Nash equilibrium for the game (OPT-HT). We thus conclude to the existence of an equilibrium for this game.

Corollary 3. *If the game is feasible, there exists a Nash equilibrium for* (OPT-HT)*.*

4.3. Characterization of the Nash Equilibrium and Uniqueness

In this section, we assume that the game (OPT-HT) is feasible and we study the equilibrium of this game. We recall that it is assumed that the classes are ordered in decreasing order of $\frac{\mathbb{E}(B_i)}{c_i}$. We first focus on the case where $\boldsymbol{c}$ is achievable in heavy-traffic.

Proposition 11. *If* $\mathbf{c}$ *is achievable in heavy-traffic, there are an infinite number of equilibria.*

Proof. Similar to the proof of Proposition 4. ■

We now characterize the Nash equilibrium of (OPT-HT) if the vector of deadlines $\boldsymbol{c}$ is not achievable in heavy-traffic. The following theorem states our main result.

Theorem 2. *If the game is feasible and* $\mathbf{c}$ *is not achievable in heavy-traffic, the Nash equiblibrium is*

$$\begin{aligned} g_i^{NE} &= \epsilon \frac{\tilde{t_m}/\mathbb{E}(B_m)}{\tilde{c}_i/\mathbb{E}(B_i)}, \textit{ for all } i < m \\ g_i^{NE} &= \epsilon, \textit{ for all } i \geq m, \end{aligned}$$

where $m \in \mathcal{C}$ *is the minimum value such that there exists a value* $\tilde{t_m} \leq \tilde{c}_m$ *verifying*

$$\frac{\tilde{t_m}}{\mathbb{E}(B_m)} = \frac{\sum_{k=1}^{R} \lambda_k \mathbb{E}\left(B_k^2\right) - \sum_{k=1}^{m-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c}_k}{\sum_{k=m}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)}. \tag{22}$$

Proof. See appendix B.3. ■

In the particular case where all classes are fair, we notice that $m = 1$ and thus the solution is very simple:

Corollary 4. *Under the assumptions of Theorem 2, and if in addition all users are fair in heavy-traffic, i.e.,* $\overline{T}_i(\boldsymbol{g}^{PS}; 1) \leq \tilde{c}_i \ \forall i \in \mathcal{C}$*, then the equilibrium is* $\mathbf{g^{NE}} = (\epsilon, \ldots, \epsilon)$*.*

We now present the uniqueness of the equilibrium in the following theorem:

Theorem 3. *Under the assumptions of Theorem 2, the heavy-traffic equilibrium is unique.*

Proof. See appendix B.4. ■

The following corollary shows that the price paid by classes at the Nash equilibrium decreases as the ratio $\mathbb{E}(B_k)/\tilde{c}_k$ decreases

Corollary 5. *Under the assumptions of Theorem 2, let* $\mathbf{g^{NE}} = (g_1^{NE}, \ldots, g_R^{NE})$ *be the vector of weights at equilibrium. We have*

$$g_1^{NE} \geq g_2^{NE} \geq \cdots \geq g_{R-1}^{NE} \geq \epsilon$$

Proof. It follows from Theorem 2 and our assumption on the ordering of the classes. ■

It is interesting to observe that the ordering of classes at equilibrium do not depend on the arrival or second moment of the distributions. Instead, the key parameter is the ratio $\mathbb{E}(B_k)/\tilde{c}_k$, which can be interpreted as the throughput of a class k. Thus, classes will deviate from the minimum weight in decreasing order with respect to the throughput they expect to obtain from the system.

4.4. Price of Anarchy

We can also define the social optimum of the system for (OPT-HT):

$$\begin{aligned} \min_{(g_1,\dots,g_R)} \quad & \sum_{i=1}^{R} \rho_i g_i \\ \text{subject to} \quad & \overline{T_i}(\boldsymbol{g};1) \leq \tilde{c}_i, \text{for all } i=1,\dots,R, \\ \text{and} \quad & g_i \geq \epsilon, \text{for all } i=1,\dots,R. \end{aligned} \tag{SOC-OPT}$$

Assuming the game is feasible, the Price of Anarchy is defined as the ratio between the maximum payment of the users in the equilibria and the payment of the users in the social optimum. We know that there is an infinite number of equilibria if the vector of deadlines $\boldsymbol{c}$ is achievable. We now present the value of the Price of Anarchy in that case.

Proposition 12. *If $\boldsymbol{c}$ is achievable in heavy-traffic, $PoA = \infty$.*

Proof. The result follows from the lack of upper-bound on the vector of weights. ■

On the other hand, if $\boldsymbol{c}$ is not achievable in heavy-traffic, we have shown in Theorem 3 that the equilibrium is unique. In this case, we define the PoA as follows

$$PoA = \frac{\sum_{i=1}^{R} \rho_i g_i^{NE}}{\sum_{i=1}^{R} \rho_i g_i^{SOC}}, \tag{23}$$

where $\boldsymbol{g}^{NE}$ is the unique Nash equilibrium of (OPT-HT), while $\boldsymbol{g}^{SOC}$ is any optimal solution of (SOC-OPT). Using the same arguments that in Section 3.5 for the game (OPT-M), we can prove that any social optimum is also a Nash equilibrium for (OPT-HT). An immediate consequence is that the PoA is 1 for the DPS game in heavy-traffic, whatever the number of classes.

Corollary 6. *Under the assumptions of Theorem 2, $PoA = 1$ for the game* (OPT-HT).

5. Approximating (OPT-M)

In this section we explain how the results of Section 4 can be used to obtain insights into the solution of games (OPT-P) and (OPT-M). As explained in Section 2.2, provided that ρ is sufficiently large for the approximation $\overline{T}_i(\boldsymbol{g};\rho) = \frac{\overline{T}_i(\boldsymbol{g};1)}{1-\rho}$ to be valid, the results established for game (OPT-HT) can be applied to approximate the solution of (OPT-P) by setting $\tilde{c}_i = -(1-\rho)d_i/\log\alpha_i$ and the solution of (OPT-M) by setting $\tilde{c}_i = (1-\rho)c_i$. We will focus on the case (OPT-M). This choice allows to evaluate numerically the accuracy of the approximation using the formulas presented in Section 2.1.

5.1. Feasibility when $\rho < 1$

We focus on the existence of the solution of (OPT-M) when $\rho < 1$. Assuming exponential service times, the necessary and sufficient conditions of the feasibility of the game are given in theorem 1. When the service times are generally distributed, we use the heavy-traffic result to claim when the approximated solution of the game exists.

It follows directly from Proposition 10 that a necessary and sufficient condition for the (approximate) feasibility of (OPT-M) is

$$\sum_i \lambda_i \mathbb{E}\left(B_i^2\right)\left(\frac{c_i}{\mathbb{E}(B_i)/(1-\rho)} - 1\right) \geq 0. \tag{24}$$

This implies that if all users are fair, then the game is feasible.

5.2. The Nash Equilibrium for $\rho < 1$

Extending Theorem 2 to the case $\rho < 1$ with $\tilde{c}_i = c_i(1-\rho)$, we obtain that the Nash-Equilibrium of (OPT-M) can be approximated by

$$g_i^{NE} = \epsilon \frac{t_m/\mathbb{E}(B_m)}{c_i/\mathbb{E}(B_i)}, \text{ for all } i < m$$

$$g_i^{NE} = \epsilon, \text{ for all } i \geq m,$$

where $m = 1, \ldots, R$ is the minimum value such that there exists a value $t_m \leq c_m$ verifying

$$\frac{t_m}{\mathbb{E}(B_m)} = \frac{\sum_{k=1}^{R} \frac{\lambda_k \mathbb{E}\left(B_k^2\right)}{(1-\rho)} - \sum_{k=1}^{m-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \, c_k}{\sum_{k=m}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)}. \tag{25}$$

Note that if class 1 is fair, then all users are fair. In this case, the right-hand side of (25) is upper-bounded by $(1-\rho)^{-1}$, implying that $c_1 \geq \frac{\mathbb{E}(B_1)}{1-\rho} \geq t_1$, so that $m = 1$. Thus, if class 1 is fair, the approximate equilibrium corresponds to the PS solution $g_i^{NE} = \epsilon$ for all i, which is clearly the exact equilibrium.

It is interesting to compare the above approximate characterization of the Nash equilibrium with the exact result given in Proposition 6 in the case of two users and exponential service time distributions. As discussed above, if class 1 is fair, then the approximate and exact equilibria coincide and correspond to the PS queue. Otherwise, the equilibrium in both instances have the same form, i.e., $\mathbf{g^{NE}} = (g_1^{NE}, \epsilon)$, with $g_1^{NE} > \epsilon$.

Observe that $t_m/\mathbb{E}(B_m)$ increases with ρ. Indeed, when all the users are paying the minimum price ϵ, the approximation is exact and if there is at least one class that is paying more than ϵ, the prices that are larger than ϵ are approximated using the heavy-traffic result. Hence, we define ρ_E and ρ_F as the threshold values such that:

- if $\rho \leq \rho_E$, then the approximation is exact,
- if $\rho_E < \rho \leq \rho_F$, the heavy-traffic equilibrium approximates the solution of the original problem,
- if $\rho > \rho_F$, the game is not feasible.

5.3. Characterization of ρ_E

As discussed above, if class 1 is fair, that is if $\frac{\mathbb{E}(B_1)}{c_1} \leq 1 - \rho$, then all users are paying the minimum price at the equilibrium. As a consequence, the minimum value ρ_E such that at least one user pays more than ϵ is obtained when $\frac{\mathbb{E}(B_1)}{c_1} = 1 - \rho_E$, that is for

$$\rho_E = 1 - \frac{\mathbb{E}(B_1)}{c_1}. \tag{26}$$

We emphasize that, since we have not used heavy-traffic results to characterize ρ_E, the above expression of ρ_E is the exact threshold where class 1 starts paying more than ϵ. We also note from (26) that if the throughput $\mathbb{E}(B_1)/c_1$ of class 1 is close to 0, then ρ_E is close to 1, implying that the PS solution $(\epsilon, \ldots, \epsilon)$ corresponds to the equilibrium for a large range of utilization rates.

5.4. Characterization of ρ_F

We present the value of the maximum load in the system that makes the game not feasible. For exponential service times, we use the result of theorem 1 to state that ρ_F is the minimum value of the system load such that $\boldsymbol{c} \notin \mathcal{U}$, which is equivalent to $\exists r \in \mathcal{R}$ s.t. $\sum_{i \in r} \rho_i c_i < W_r$.

For general service times, we obtain an approximate value for ρ_F using the heavy-traffic characterization. From (24) it follows that

$$\rho_F = \frac{\sum_{i=1}^{R} \lambda_i \mathbb{E}\left(B_i^2\right)\left(\frac{c_i}{\mathbb{E}(B_i)} - 1\right)}{\sum_{i=1}^{R} \lambda_i \frac{\mathbb{E}\left(B_i^2\right)}{\mathbb{E}(B_i)} c_i}. \tag{27}$$

We emphasize that the value of ρ_F for exponential service times is exact and the approximation given for general service times is only valid if the real value is sufficiently close to 1 for the approximation $\overline{T}_i(\boldsymbol{g};\rho) = \frac{\overline{T}_i(\boldsymbol{g};1)}{1-\rho}$ to be accurate.

5.5. Identical throughput expectations

A particular case of interest is obtained when all classes have the same throughput expectations. In this case, we can characterize exactly the value of ρ_F.

Proposition 13. *If $\mathbb{E}(B_i)/c_i = k < 1$ for all i, then the unique equilibrium of the game is the PS solution $(\epsilon, \ldots, \epsilon)$ for $\rho \leq 1-k$, and the game is not feasible for $\rho > 1-k$.*

Proof. If all users had the same weights (so the equilibrium were PS), we would have that $\mathbb{E}(B_i)/c_i = 1-\rho$, for all i. Since $\mathbb{E}(B_i)/c_i = k < 1$, we conclude that if $\rho \leq 1-k$ then $(\epsilon, \ldots, \epsilon)$ is the unique equilibrium. When $\rho = 1-k$ we have $c_i = \mathbb{E}(B_i)/(1-\rho)$, $\forall i$, that is, c_i, $\forall i$, is equal to the sojourn time in a PS queue. This means that the vector $(c_1, \ldots, c_R)$ lies in the achievable region of the system, and as soon as ρ increases further the game becomes infeasible. ■

We thus have $\rho_E = \rho_F = 1-k$. From (27) we also conclude that in this case the approximation of ρ_F gives the exact value $1-k$.

6. Numerical Experiments

In this section, we illustrate the most important characteristics of the resource sharing model with relative priorities we have analysed. We first explain how the obtained result can be applied to a virtualized environment executing CPU-intensive and interactive jobs. We then present the results of numerical experiments in order to compare the equilibrium of the game (OPT-M) (which we call the original problem) with that of the heavy-traffic approximation (OPT-HT).

Without loss of generality, the minimum weight ϵ is set to 1 in all the experiments.

6.1. Application to Virtualization with two types of application

We show how the result of proposition 6 can be applied to a cloud computing architecture that executes two types of applications. In particular, we study the equilibrium of the game for the different experiments presented in [23], where interactive and CPU-intensive jobs are executed in a real virtualized platform.

We denote by class 1 the interactive jobs and class 2 type to the CPU-intensive jobs and we consider that the performance requirement of the interactive jobs is very tight while the requirement of the CPU-intensive jobs is always satisfied. Thus, for this instance, the interactive application may pay for obtaining the required

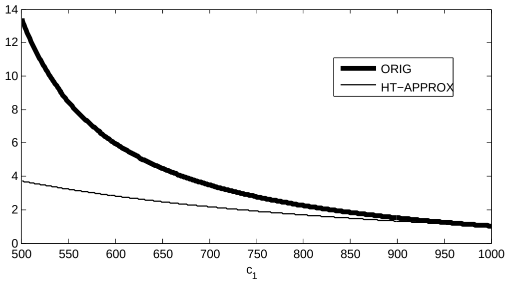

Figure 1: The evolution of g_1/g_2 when c_1 changes from 500 to 1000 seconds for the original problem and the heavy-traffic approximation.

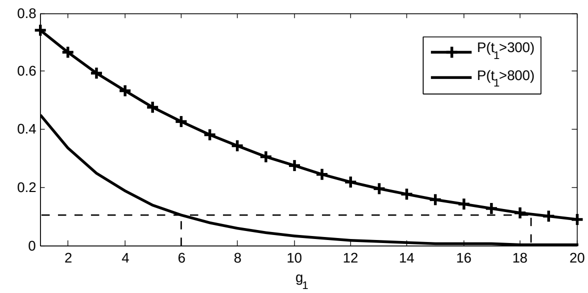

Figure 2: The probability of the response time of interactive jobs to be larger that 300 and 800 seconds with respect to the priority level of the interactive applications (g_1). Parameter: $g_2 = 1$.

priority level in order to get service before its deadline and the CPU-intensive application pays the minimum possible since its performance requirement will be always satisfied.

We analyse the equilibrium of the game for different values of the system load. We consider that the values of the deadlines of both applications are $c_1 = 290$ seconds and $c_2 = 10^5$ seconds. Recall that the mean response time of the real executions of the CPU-intensive and the interactive applications considered in [23] is 198 seconds and 201 seconds, respectively. We present the equilibrium when the load of both applications is 0.1, i.e., $\rho_1 = \rho_2 = 0.1$. For this case, both classes are fair and thus both classes pay ϵ. This means that no class need to increase its priority level to be served before its deadline. On the other hand, if $\rho_1 = 0.3$ and $\rho_2 = 0.2$, the CPU-intensive jobs pay also the minimum possible but the interactive application increases its priority level to be satisfied its performance requirement. The payment in the equilibrium of the interactive jobs is 85.6788 times the minimum price in this case.

We observe that the rational behaviour of interactive and CPU-intensive jobs is to start paying the minimum possible and one of the applications increases its weight in case its performance requirement is not satisfied. Thus, the Selfish Priority Adaptation Algorithm starting from the point (ϵ, ϵ) since it reflects the rational behaviour of the evolution of the payments of interactive and CPU-intensive jobs in a virtualized environment. In figure 8 we consider the case $\rho_1 = 0.3$ and $\rho_2 = 0.2$ while the rest of the parameters are the same as the previous paragraph and we plot the evolution of the weights and the mean response time under this algorithm. We observe that the CPU-intensive always satisfies its performance requirement. However, the interactive application need to increase its weight and then it decreases its weight until the algorithm converges to the equilibrium. Besides, we show that if we start from the point (ϵ, ϵ) the algorithm converges in 150 iterations to the equilibrium point $(85.6788\ \epsilon, \epsilon)$.

We also investigate the influence in the priority level achieved in the equilibrium if the deadline of the interactive application changes. We focus on the case $\rho_1 = 0.5$ and $\rho_2 = 0.3$ with the values of the parameters of the previous paragraph and we present in figure 1 the evolution of the relatives priority g_1/g_2 when c_1 changes from 500 to 1000 seconds for the equilibrium of the original game and the approximated equilibrium. We observe that the approximation is accurate for a large value of c_1 and the heavy-traffic equilibrium approximates worst the exact equilibrium when the deadline c_1 is 500 seconds.

In figure 2, we depict the constraint of (OPT-P) of the interactive application for $d_1 = 300$ and $d_1 = 800$ seconds when the priority level of the interactive jobs changes from 1 to 20. We are interested in the minimum weight that ensures that this probability is less than the 0.1, i.e., the equilibrium of the game for $\alpha_1 = 0.1$. We present that for the case $d_1 = 800$ the minimum weight that verifies this condition is $g_1^{800} = 6$ and for $d_1 = 300$ the minimum weight is $g_1^{300} \approx 18$. We observe that the obtained priorities verify that $g_1^{300} \approx 3\ g_1^{800}$.

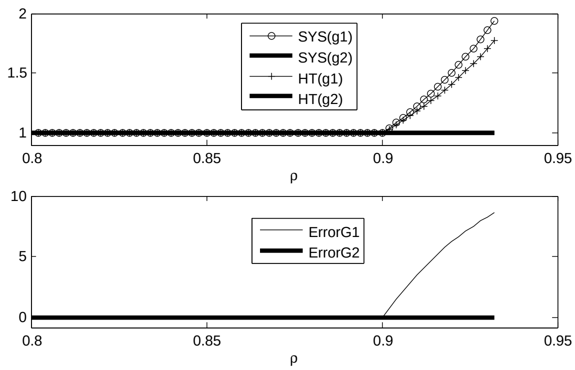

Figure 3: Comparison of equilibrium weights (above) and the corresponding percentage relative error (below) as a function of the total system load. $R = 2$ and exponential service time distribution.

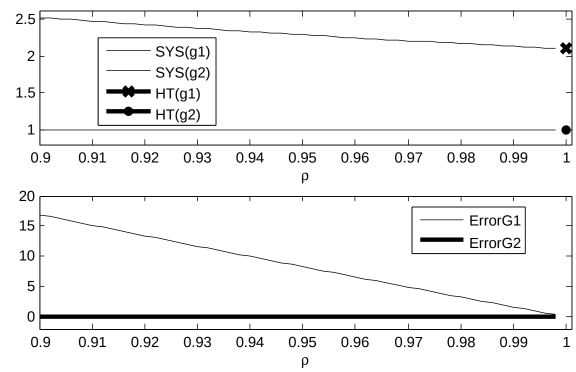

Figure 4: Comparision of equilibrium weights (above) and the corresponding percentage relative error (below) as a function of the total load, and the deadlines of the two classes are scaled by $(1-\rho)^{-1}$. $R = 2$ and exponential service time distribution.

6.2. Validation of the Approximation

We analyse numerically the accuracy of the heavy-traffic approximation. Our main observation from the experiments that we conducted is that while in certain cases the error in weights can be substantial, the proposed heavy-traffic approximation is good at predicting the set of classes that pay a higher than minimum price at the equilibrium, and the mean response times of the classes paying the minimum price.

6.2.1. Exponential service time distribution

First, we present the results for exponentially distributed service times. In the first set of experiments, there are two players with deadlines $c_1 = 5$ and $c_2 = 6$, and the mean service times $\mu_1 = 2$ and $\mu_2 = 3$. Note that $c_1\mu_1 = 10 < c_2\mu_2 = 18$. We now vary the total system load starting from 0.8 until the system becomes unfeasible while maintaining $\rho_1 = 0.3\rho$ and $\rho_2 = 0.7\rho$. For each value of load, the equilibrium is computed using the best-response algorithm. In order to compute the best-response of a class for the original problem, the mean response time is computed from the system of equations presented in Proposition 1.

In the bottom subfigure of Figure 3, we plot the equilibrium weights for both the original problem and the HT approximation as a function of the total system load. The percentage relative error[1] between the two is shown in the top subfigure of the same figure.

Both problems become unfeasible for $\rho > 0.93$, so the data is restricted to $\rho \leq 0.93$. When the load of the system is between 0.9 and 0.93 we observe in Figure 3 (below) that the equilibrium of the heavy-traffic result approximates very well the equilibrium of the original problem. In particular, the heavy-traffic approximation follows the same increasing trend of the equilibrium weight of class 1 as that of the original problem. The error of class 1 users is small, while there is no error for the users of class 2. We see in Figure 3 (above) that the maximum percentage relative error is 9% and it is achieved when $\rho = 0.93$.

In the second set of experiments, we scale the deadlines by $(1-\rho)^{-1}$, that is, the deadline of user i, $c_i = \frac{\tilde{c}_i}{(1-\rho)}$ for some fixed $\tilde{c}_i$. This reflects that class i jobs is aware that the performance worsens as ρ increases, and is willing to adjust its deadline correspondingly. When the deadlines are scaled with $(1-\rho)^{-1}$, the constraint on the mean response time of player i for the original problem becomes $\overline{T}_i(\boldsymbol{g};\rho) \leq \frac{\tilde{c}_i}{1-\rho}$, and

[1]The percentage relative error for class i is given by $\left|\frac{g_i^{SYS}-g_i^{HT}}{g_i^{SYS}}\right| \times 100$, where g_i^{SYS} (resp., g_i^{HT}) is its equilibrium weight for the original problem (resp. HT approximation).

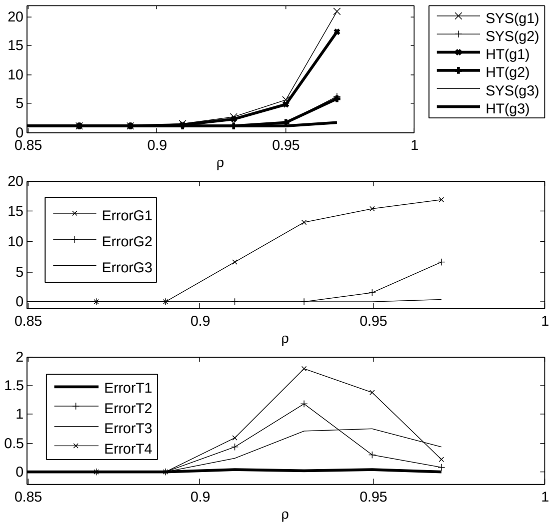

Figure 5: Comparison of equilibrium weights (above), the percentage relative error of the weights (middle) and the percentage relative error of the time (bellow) as a function of the total system load. $R = 4$ and exponential service time distribution. $\mathbf{c} = [10, 15, 25, 45]$, $\boldsymbol{\mu} = [1, 2, 4, 9]$.

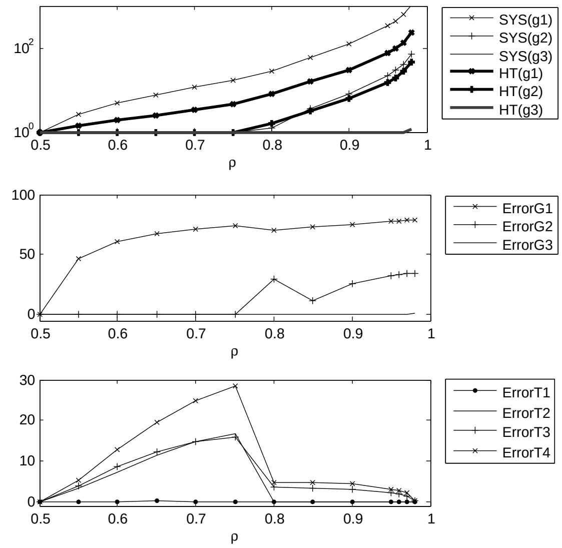

Figure 6: Comparison of equilibrium weights (above), the percentage relative error of the weights (middle) and the percentage relative error of the time (bellow) as a function of the total system load. $R = 4$ and exponential service time distribution. $\mathbf{c} = [5/3, 5/4, 10, 100]$, $\boldsymbol{\mu} = [1, 2, 8, 12]$.

that for the heavy-traffic approximation becomes $\overline{T}_i(\boldsymbol{g}; 1) \leq \tilde{c}_i$. Note that the latter constraint does not change with ρ.

We set the parameters to : $\mu_1 = 2$ and $\mu_2 = 3$, $\rho_1 = 0.3\rho$, and $\rho_2 = 0.7\rho$, with the scaled deadlines being $\tilde{c}_1 = 0.3$ and $\tilde{c}_2 = 0.7$. In Figure 4, we present the accuracy of the heavy-traffic approximation as $\rho \to 1$. As expected, the error in the weight of class 1 reduces as the load tends to 1.

In the next set of experiments, we look at a four-player game with parameters $\mathbf{c} = [10, 15, 25, 45]$ and $\boldsymbol{\mu} = [1, 2, 4, 9]$. The loads of individual classes are in the proportion $[1/3, 1/6, 1/4, 1/4]$, that is $\rho_1 = \rho/3$, $\rho_2 = \rho/6$, and so on. In Figure 5, the equilibrium weights are plotted in the top subfigure, the corresponding error is plotted in the middle subfigure, and in the bottom subfigure we plot the error in the equilibrium mean response times of the classes. We did not plot the weights and the error for class 4 because its weight is always 1 in both the systems.

The trend in the four-player plots is similar to that of the two-player example in which the deadlines are not scaled. Until $\rho < 1 - \frac{1}{c_1\mu_1} = 0.9$, PS is a feasible solution and all the classes pay the minimum price. As the load increases further and moves closer to the ρ_{max} of the original system, one or more classes start to pay, and the error in the equilibrium starts to increase. The main observation here is that, even though the maximum error in the weights is around 19%, the maximum error in the mean response times is less than 2%.

It is rather surprising that mean response times in the heavy-traffic approximation are so close to that in the original game. As another example in support of this observation, we set the parameters to : $\mathbf{c} = [5/3, 5/4, 10, 100]$ and $\boldsymbol{\mu} = [1, 2, 8, 12]$, with the proportion of loads being the same as before. The main difference with the previous example is that there is much more heterogeneity in the deadlines and the $c\mu$ of the classes. The first two classes have a much smaller deadline and the range of values of $c_i\mu_i$ is now much larger as well compared to the previous example. The plots are shown In Figure 6. Note that the scale is logarithmic for the vertical axis in the top subfigure, and that in the top and middle subfigure, the data is plotted only for the first two classes because the other two classes are always paying the minimum price.

The error in the weight of class 1 is close to 60% at $\rho = 0.5$ and increases to almost 100% at $\rho = 0.9$. The error for class 2 is similarly large for loads close to $\rho = 0.9$ which means that the prediction is poor. For example, for $\rho = 0.9$ the weights are : $g_1^{SYS} = 7287.8$, $g_1^{HT} = 45.2$, $g_2^{SYS} = 2120.6$, and $g_2^{HT} = 30.14$. That is, the heavy-traffic approximation predicts a weight of 45.2 for class 1 whereas the weight in the original system is 7287.8. There is a similar disparity in the weight of class 2. A similar observation on the negative impact of heterogeneity on the error was also made in [11]. On the other hand and in spite of the large disparity in the weights, the maximum error in the mean response times is negligible. For classes 1 and 2 it is not surprising that the error is small because their mean response times are equal to their constraint since they are paying more than the minimum price. For classes 3 and 4, the mean response times are strictly smaller than their constraint, and their values in the original system and as predicted by the heavy-traffic approximation are : $T_3^{SYS} = 9$, $T_3^{HT} = 9.41$, $T_4^{SYS} = 6.7$, and $T_4^{HT} = 6.28$, which are reasonably close.

6.2.2. Hyper-exponential service requirements

Finally, in this subsection, compare the approximation for a two-player game with hyper-exponentially distributed service times.

While there is no explicit expression for mean response time in DPS with service time distributions other than the exponential distribution, for the hyper-exponential distribution, a simple trick can be used to compute the mean response times using those of the exponential distribution. For example, consider a two-class DPS queue with hyper-exponential distribution of two phases each. The service rates of the phases are (μ_1, μ_2) for class 1 and (μ_3, μ_4) for class 2. and the arrival rates to these phases are (λ_1, λ_2) for class 1 and (λ_3, λ_4) for class 2. In order to compute the mean response time in this queue when the weights are $\mathbf{g} = (g_1, g_2)$, one first computes the mean response time in a four-class DPS queue with exponential distribution and weights $\mathbf{g} = (g_1, g_1, g_2, g_2)$. The arrival rate of class i in this queue is λ_i, and the rates of the exponential distribution of class i is taken to be μ_i. The mean respone time of class i in the DPS queue with hyper-exponential distribution is then:

$$\overline{T}_1^{HEXP}(\boldsymbol{g};\rho) = \frac{\lambda_1}{\lambda_1+\lambda_2}\overline{T}_1(\boldsymbol{g};\rho) + \frac{\lambda_2}{\lambda_1+\lambda_2}\overline{T}_2(\boldsymbol{g};\rho),$$
$$\overline{T}_2^{HEXP}(\boldsymbol{g};\rho) = \frac{\lambda_3}{\lambda_3+\lambda_4}\overline{T}_3(\boldsymbol{g};\rho) + \frac{\lambda_2}{\lambda_3+\lambda_4}\overline{T}_4(\boldsymbol{g};\rho).$$

Using the above trick, the equilibrium weights were computed for the two-player DPS game with parameters: $\mu_1 = 1$, $\mu_2 = 3$, $\mu_3 = 5$, $\mu_4 = 7$, and deadlines $c_1 = 5$ and $c_2 = 7$. The fraction of the load of class 1 was $(\rho_1, \rho_2) = (\frac{\rho}{6}, \frac{\rho}{3})$, and for class 2 it was $(\rho_3, \rho_4) = (\frac{\rho}{4}, \frac{\rho}{4})$.

In Figure 7 we depict variation of the weights and the relative error when the total load of the system changes. Finally, we observe that the error on the equilibrium is similar to that of the exponentially distributed service times.

6.3. Selfish Priority Adaptation

In section, we consider that users can update their priorities in a selfish fashion. We aim to analyse how the priorities of the users evolve for a given initial point if we let the users change their weights in a selfish way. We design an algorithm based on priority adaptation that reflects the selfish behaviour of the users and we show that from any vector of priorities this algorithm converges to the nash equilibrium of the game.

We first observe that the selfish behaviour of a user-i consists on increasing g_i in case of $\overline{T}_i(\boldsymbol{g};\rho) > c_i$ and decreasing g_i if $\overline{T}_i(\boldsymbol{g};\rho) < c_i$ and $g_i > \epsilon$. Taking into account this property, we present an algorithm that simulates the behaviour of selfish users that want to minimize their weights guaranteeing their performance requirements.

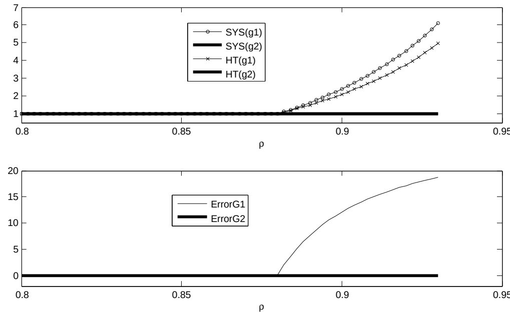

Figure 7: Comparison of equilibrium weights (above) and the corresponding percentage relative error (below) as a function of the total system load. $R = 2$ and hyper-exponential service time requirements.

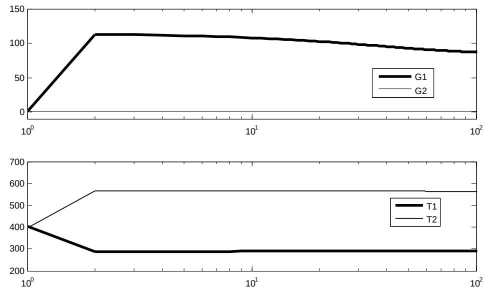

Figure 8: The evolution of the weights and the mean response time under the Selfish Priority Adaptation Algorithm when interactive and CPU-intensive applications start paying (ϵ, ϵ). X-axis in logarithmic scale.

The dynamics of the selfish adaptation Algorithm are very simple. The users are able to compute the mean response time of the jobs finished in Δt amount of time. According to this estimation, the users modify each weight according to the following rule:

$$g_i(t + \Delta t) = \max\Big(\epsilon, g_i(t) + (\hat{T}_i(\boldsymbol{g}; \rho) - c_i)\Delta t\Big)$$

where $\hat{T}_i(\boldsymbol{g}; \rho)$ is the estimated mean response time of jobs in the previous Δt period.

We observe that the modification of the classes depends on the difference between the estimated mean response time $\hat{T}_i(\boldsymbol{g}; \rho)$ and c_i as follows: if $\hat{T}_i(\boldsymbol{g}; \rho) > c_i$, then user-$i$ increases its weight which causes that $\hat{T}_i(\boldsymbol{g}; \rho)$ diminishes; if $\hat{T}_i(\boldsymbol{g}; \rho) < c_i$, then the priority of user-i decreases and thus the mean response time of class-i jobs increases. We impose the minimum weight to be ϵ using the max function.

We now explain the main differences of the proposed algorithm with respect to the best-response. When a player does best-response, it solves the optimization problem (OPT-M) and, as a result its weight will be ϵ or the weight that satisfies its performance constraint as an equality. On the contrary, the modification of the weights of Selfish Priority Adaptation Algorithm yields to the maximum between ϵ and $g_i(t)+(\hat{T}_i(\boldsymbol{g}; \rho)-c_i)\Delta t$, which does not always coincide with the weight obtained with the best-response algorithm.

We illustrate in figure 9 the evolution of the weights and the mean response time when behaviour of the users follows the Selfish Priority Adaptation Algorithm. We consider exponential service times and three classes of users with the following parameters: the load of each class is $(\rho_1, \rho_2, \rho_3) = (0.1, 0.5, 0.2)$, the mean job sizes are given by $(\mu_1, \mu_2, \mu_3) = (1, 2, 3)$ and the deadlines are $(c_1, c_2, c_3) = (2, 2.5, 100)$. We fix the value of ϵ to 1 and we want to see the differences of the algorithm for several starting points.

We observe in figure 9 that for the three different starting points the algorithm converge in less that 200 iterations. Besides, in all the cases, it converges to the a vector of weights that satisfies the equilibrium conditions (14)-(15). Thus, we can claim that the Selfish Priority Adaptation Algorithm converges to the equilibrium of the game.

7. Conclusions

We presented a priced model that studies the strategic behaviour of users that share the capacity of a processor with relative priorities. Each user chooses a price which corresponds to priority level and receive a share of the capacity that increases with its payment. The objective of a users is to choose its priority level so as to minimize its own payment, while guaranteeing that its jobs are served before their deadline. We have formulated this problem as a game with the latter Quality of Service constraint.

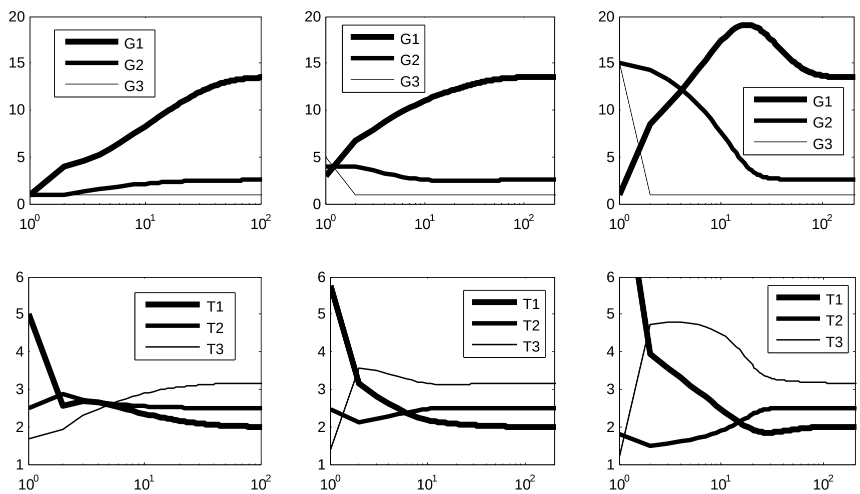

Figure 9: The evolution of the weights (up row) and the mean response times (down row) with the Selfish Priority Adaptation Algorithm for three different starting points: $\boldsymbol{g} = (1, 1, 1)$ (left column), $\boldsymbol{g} = (3, 4, 5)$ (middle column) and $\boldsymbol{g} = (1, 15, 15)$ (right column). X-axis in logarithmic scale.

We considered that the capacity among the different users is shared according to the DPS discipline. Since the DPS queue is very complicated model to analyse, the defined problem is not solvable in general. Nevertheless, we fully characterized the existence and uniqueness of the solution of this game when the number of users is two and the service time distribution is exponential. Besides, we defined a game in the heavy-traffic regime which we solved for the general instance. Interestingly, we observed that the ordering of the weights in the equilibrium depend only on throughput expectation of the classes, i.e., on the ratio between the mean job size and the deadline. We have also shown that the payment of the users decreases with the throughput expectation. We used the result obtained in heavy-traffic to give an approximation of the original problem. For exponential service time requirements, we fully characterize the necessary and sufficient conditions of the existence of the solution of the original game. We also showed that the approximated solution of the game coincides with the solution of the global optimization problem that minimizes the payment of all the users.

We have performed several numerical experiments that show the most important characteristics of the proposed approximation. On the one hand, we observed that the approximation is accurate when the throughput expectation of all the users is similar. On the other hand, if the heterogeneity of the throughput expectation of the classes increases, we conclude that the accuracy of the approximation can diminish. However, we derive that the heavy-traffic approximation gives us a negligible error in the mean response time prediction which let us capture the correct structure of the equilibrium.

Our work analyses the difficult problem of pricing in time-sharing systems using a heavy-traffic approximation. Indeed, our solution is based on a novel idea for time-sharing systems with a certain Quality of Service guaranties. As a future work we Despite its limitations, we believe that the presented methodology will let future researchers study more complicated models based on time-sharing systems.

8. Acknowledgementes

This work has been partially supported by grant ANR-11-INFR-001. PANACEA AS WELL????

References

[1] R. Hassin, M. Haviv, To Queue or not to Queue: Equilibrium Behavior in Queueing Systems, Kluwer Academic Publishers, Boston etc., 2003.

[2] N. Nisan, T. Roughgarden, E. Tardos, V. Vazirani, Algorithmic Game Theory, Cambridge University Press, New York, NY, USA, 2007.

[3] M. Haviv, T. Roughgarden, The price of anarchy in an exponential multi-server, Operations Research Letters 35 (2007) 421–426.

[4] H. Chen, J. Marden, A. Wierman, The effect of local scheduling in load balancing designs, in: Proceedings of IEEE Infocom, 2009.

[5] U. Ayesta, O. Brun, B. J. Prabhu, Price of anarchy in non-cooperative load balancing, in: Proceedings of IEEE Infocom, 2010.

[6] M. Haviv, J. van der Wal, Equilibrium strategies for processor sharing and random queues with relative priorities, Probability in the Engineering and Informational Sciences 11 (1997) 403–412.

[7] M. Vojnovic, T. Nguyen, The weighted proportional sharing mechanisms, in: ACM SIGMETRICS, 2011.

[8] S. Birmiwal, R. R. Mazumdar, S. Sundaram, Predictable revenue under processor sharing, in: CISS, 2012, pp. 1–6.

[9] Y. Yang, R. Ma, J. Lui, Price differentiation and control in the kelly mechanism, Performance Evaluation 70 (10) (2013) 792 – 805.

[10] Y. Hayel, B. Tuffin, Pricing for heterogeneous services at a discriminatory processor sharing queue, in: Proceedings of Networking, 2005.

[11] Y. Wu, L. Bui, R. Johari, Heavy traffic approximation of equilibria in resource sharing games, IEEE Journal on Selected Areas in Communications 30 (11) (2012) 2200–2209.

[12] L. Kleinrock, Time-shared systems: A theoretical treatment, Journal of the ACM 14 (2) (1967) 242–261.

[13] S. Aalto, U. Ayesta, S. Borst, V. Misra, R. N. nez Queija, Beyond Processor Sharing, ACM SIGMETRICS Performance Evaluation Review 34 (2007) 36–43.

[14] [link].
URL `aws.amazon.com/ec2/`

[15] S. Grishechkin, On a relationship between processor sharing queues and crump-mode-jagers branching processes, Adv. Appl. Prob. 24 (3) (1992) 653–698.

[16] G. Fayolle, I. Mitrani, R. Iasnogorodski, Sharing a processor among many job classes, Journal of the ACM 27 (3) (1980) 519–532.

[17] K. Rege, B. Sengupta, Queue length distribution for the discriminatory processor sharing queue, Operations Research 44 (4) (1996) 653–657.

[18] I. Verloop, U. Ayesta, R. Núñez-Queija, Heavy-traffic analysis of a multiple-phase network with discriminatory processor sharing, Operations Research 59 (3) (2011) 648–660.

[19] F. Kelly, Stochastic Networks and Reversibility, Wiley, Chichester, 1979.

[20] J. Cohen, The multiple phase service network with generalized processor sharing, Acta Informatica 12 (1979) 245–284.

[21] E. Gelenbe, I. Mitrani, Analysis and Synthesis of Computer Systems, London: Academic Press, 1980.

[22] K. Avrachenkov, U. Ayesta, P. Brown, R. Núñez-Queija, Discriminatory processor sharing revisited, in: Proceedings of IEEE INFOCOM, 2005.

[23] H. B. Cheikh, J. Doncel, O. Brun, B. Prabhu, Predicting response times of applications in virtualized environments, in: IEEE NCCA, 2014.

A. Proofs of the Section 3

A.1. Proof of Theorem 1

As in Definition 1, we let $\mathcal{T}$ be the set of achievable vectors. Define the set

$$\mathcal{U} = \left\{ \boldsymbol{c} \in \mathbb{R}_+^K : \sum_{i \in r} \rho_i c_i \geq W_r, \ \forall r \in \mathcal{R} \right\}.$$

Before giving a formal proof of Theorem 1, we briefly explain the main arguments behind the proof. It is easy to see from (11)-(12) and (9) that if $\boldsymbol{c}$ is a feasible vector, then $\boldsymbol{c} \in \mathcal{U}$. However, the converse is less clear. In order to show that each element $\boldsymbol{c}$ of $\mathcal{U}$ is a feasible vector, the idea is to construct from $\boldsymbol{c}$ a vector $\mathbf{t} \in \mathcal{U}$ such that $\mathbf{t} \preceq \boldsymbol{c}$ and for which $W_{\mathcal{C}} \leq \sum_{i \in \mathcal{C}} \rho_i t_i$ holds as an equality (whereas the inequality can be strict for $\boldsymbol{c}$). This vector $\mathbf{t}$ is obtained as the limit of a strictly decreasing sequence $\left\{\boldsymbol{c}^{(n)}\right\}_{n \geq 0}$ which, starting from $\boldsymbol{c}^{(0)} = \boldsymbol{c}$, converges in a finite number of steps. The key argument to generate this sequence is that, unless $\boldsymbol{c}^{(n)} \in \mathcal{T}$, there always exists at least one component of $\boldsymbol{c}^{(n)}$ that appears only in inequalities. By decreasing this component, we can obtain a vector $\boldsymbol{c}^{(n+1)} \in \mathcal{U}$ such that $\boldsymbol{c}^{(n+1)} \prec \boldsymbol{c}^{(n)}$ and $0 \leq \sum_{i \in \mathcal{C}} \rho_i c_i^{(n+1)} - W_{\mathcal{C}} < \sum_{i \in \mathcal{C}} \rho_i c_i^{(n)} - W_{\mathcal{C}}$, which implies the convergence to an achievable vector. We shall first prove that if $\boldsymbol{c}$ is not an achievable vector, then there is at least one of its components which is involved only in inequalities. Our first step in this direction is stated in Lemma 1.

Lemma 1. *If $r \subseteq s$ then $W_r \leq W_s$.*

Proof. From (10),

$$W_s = \frac{1}{1-\bar{\rho}_s} \sum_{i \in s} \frac{\rho_i}{\mu_i} \ \geq \frac{1}{1-\bar{\rho}_s} \sum_{i \in r} \frac{\rho_i}{\mu_i} \geq \frac{1-\bar{\rho}_r}{1-\bar{\rho}_s} W_r \ \geq W_r.$$

■

For $\boldsymbol{c} \in \mathcal{U}$, let us define the sets

$$\mathcal{S}^{=} = \left\{ r : \sum_{i \in r} \rho_i c_i = W_r \right\}$$
$$\mathcal{S}^{>} = \left\{ r : \sum_{i \in r} \rho_i c_i > W_r \right\}.$$

We have omitted the dependence of the sets on $\boldsymbol{c}$. The second result we need is the following.

Lemma 2. *If $r_1, r_2 \in \mathcal{S}^{=}$, then $r_1 \cup r_2 \in \mathcal{S}^{=}$.*

Proof. Let $s = r_1 \cup r_2$ and $v = r_1 \cap r_2$. In order to prove the desired result, we shall show that if $r_1, r_2 \in \mathcal{S}^=$ then $W_s \geq \sum_{i \in s} \rho_i c_i$. Since $\boldsymbol{c} \in \mathcal{U}$, we know that $W_s \leq \sum_{i \in s} \rho_i c_i$. Therefore, the only possible outcome is $W_s = \sum_{i \in s} \rho_i c_i$. From (10),

$$
\begin{aligned}
W_s &= \frac{1}{1-\bar{\rho}_s} \sum_{i \in s} \frac{\rho_i}{\mu_i} \\
&= \frac{1}{1-\bar{\rho}_s} \left(\sum_{i \in r_1} \frac{\rho_i}{\mu_i} + \sum_{i \in r_2} \frac{\rho_i}{\mu_i} - \sum_{i \in v} \frac{\rho_i}{\mu_i} \right) \\
&= \frac{1}{1-\bar{\rho}_s} \left((1-\bar{\rho}_{r_1}) W_{r_1} + (1-\bar{\rho}_{r_2}) W_{r_2} - (1-\bar{\rho}_v) W_v \right) \\
&= W_{r_1} + W_{r_2} + \frac{1}{1-\bar{\rho}_s} \left((\bar{\rho}_s - \bar{\rho}_{r_1}) W_{r_1} + (\bar{\rho}_s - \bar{\rho}_{r_2}) W_{r_2} - (1-\bar{\rho}_v) W_v \right) \\
&= \sum_{i \in r_1} \rho_i c_i + \sum_{i \in r_2} \rho_i c_i + \frac{1}{1-\bar{\rho}_s} \left((\bar{\rho}_s - \bar{\rho}_{r_1}) W_{r_1} + (\bar{\rho}_s - \bar{\rho}_{r_2}) W_{r_2} - (1-\bar{\rho}_v) W_v \right) \\
&= \sum_{i \in s} \rho_i c_i + \sum_{i \in v} \rho_i c_i + \frac{1}{1-\bar{\rho}_s} \left((\bar{\rho}_s - \bar{\rho}_{r_1}) W_{r_1} + (\bar{\rho}_s - \bar{\rho}_{r_2}) W_{r_2} - (1-\bar{\rho}_v) W_v \right) \\
&\geq \sum_{i \in s} \rho_i c_i + W_v + \frac{1}{1-\bar{\rho}_s} \left((\bar{\rho}_s - \bar{\rho}_{r_1}) W_{r_1} + (\bar{\rho}_s - \bar{\rho}_{r_2}) W_{r_2} - (1-\bar{\rho}_v) W_v \right) \\
&\geq \sum_{i \in s} \rho_i c_i + \frac{1}{1-\bar{\rho}_s} \left((\bar{\rho}_s - \bar{\rho}_{r_1}) W_{r_1} + (\bar{\rho}_s - \bar{\rho}_{r_2}) W_{r_2} - (\bar{\rho}_s - \bar{\rho}_v) W_v \right)
\end{aligned}
$$

In order to complete the proof it is sufficient to show that the second term on the RHS is non-negative, which will then imply that $W_s \geq \sum_{i \in s} \rho_i c_i$. Since $v = r_1 \cap r_2$, from Lemma 1, it follows that $W_{r_1} \geq W_v$ and $W_{r_2} \geq W_v$. Thus,

$$
(\bar{\rho}_s - \bar{\rho}_{r_1}) W_{r_1} + (\bar{\rho}_s - \bar{\rho}_{r_2}) W_{r_2} \geq (\bar{\rho}_s - \bar{\rho}_{r_1} + \bar{\rho}_s - \bar{\rho}_{r_2}) W_v = (\bar{\rho}_s - \bar{\rho}_v) W_v,
$$

where the last inequality follows from the fact that $\bar{\rho}_{r_1} + \bar{\rho}_{r_2} = \bar{\rho}_s + \bar{\rho}_v$. ■

Corollary 7. *The set $\mathcal{S}^=$ is closed under finite unions.*

We are now in position to prove Theorem 1.

Proof of Theorem 1. If $\boldsymbol{c}$ is feasible then it is easy to see that $\boldsymbol{c} \in \mathcal{U}$. We now prove that if $\boldsymbol{c} \in \mathcal{U}$, then $\boldsymbol{c}$ is feasible. Towards this end, for every $\boldsymbol{c}$, we shall construct a finite sequence of vectors $\boldsymbol{c} = \boldsymbol{c}^{(0)} \succ \boldsymbol{c}^{(1)} \succ \ldots \succ \boldsymbol{c}^{(n)}$, with $n \leq R$, $\boldsymbol{c}^{(i)} \in \mathcal{U}, \forall i$ and $\boldsymbol{c}^{(n)} \in \mathcal{T}$. Also, n will depend upon $\boldsymbol{c}$. The vector $\boldsymbol{c}^{(n)}$ is then an achievable vector which makes $\boldsymbol{c}$ feasible.

Consider the vector $\boldsymbol{c}^{(n)}$ obtained at step n. Define the corresponding sets $\mathcal{S}_n^=$ and $\mathcal{S}_n^>$ which contain the indices of the equalities and the strict inequalities that define $\boldsymbol{c}^{(n)}$. Also, define $\mathcal{E}_n = \bigcup_{r \in \mathcal{S}_n^=} r$,

the set of classes that appear in at least one equality. We shall show that the sequence of $\mathcal{E}_n$ associated to the componentwise decreasing vectors will eventually contain $\mathcal{C}$, and this will happen in a finite number of steps.

If $\mathcal{E}_n = \mathcal{C}$, it follows from Corollary 7 that $\mathcal{C} \in \mathcal{S}_n^=$, and that $\boldsymbol{c}^{(n)}$ is achievable. Otherwise, take some $i \in \mathcal{C} \setminus \mathcal{E}_n$, that is, a class which appears only in inequalities.

Define

$$
\begin{aligned}
c_i^{(n+1)} &= \max_{s \,:\, i \in s} \frac{W_s - \sum_{j \in s, j \neq i} \rho_j c_j^{(n)}}{\rho_i} \\
c_j^{(n+1)} &= c_j^{(n)}, \ \forall j \neq i.
\end{aligned}
$$

Note that $c_i^{(n+1)} \geq W_{\{i\}}/\rho_i > 0$, and that $c_i^{(n)} > c_i^{(n+1)}$. Therefore $\boldsymbol{c}^{(n)} \succ \boldsymbol{c}^{(n+1)}$.

With this definition class i will appear in at least one equality, and this class will be added to $\mathcal{E}_n$. Therefore, $\mathcal{E}_n \subset \mathcal{E}_{n+1}$, and $\mathcal{S}_n^{=} \subset \mathcal{S}_{n+1}^{=}$. Since there are R classes, after at most R steps all the classes will appear in at least one equality, that is, there is an $n \leq R$ such that $\mathcal{E}_n = \mathcal{C}$. From Corollary 7, it follows that $\mathcal{C} \in \mathcal{S}_n^{=}$, and $\boldsymbol{c}^{(n)}$ is an achievable vector such that $\boldsymbol{c}^{(n)} \preceq \boldsymbol{c}$. ∎

A.2. Proof of Proposition 5

Proof of Proposition 5. Assume that there exist two equilibria $\boldsymbol{g}$ and $\boldsymbol{h} \neq \boldsymbol{g}$.

If $h_1 = g_1$, then we can assume without loss of generality that $h_2 < g_2$. This implies that $g_2 > \epsilon$, and thus, according to (14), that $\overline{T}_2(\boldsymbol{g};\rho) = c_2$. Since $\overline{T}_2(\boldsymbol{g};\rho)$ is strictly decreasing in g_2, it yields $\overline{T}_2((h_1,h_2);\rho) = \overline{T}_2((g_1,h_2);\rho) > c_2$. Hence, $\boldsymbol{h}$ is not a feasible point for class 2 and thus cannot be an equilibrium. This is a contradiction, and therefore we cannot have two different equilibria $\boldsymbol{g}$ and $\boldsymbol{h}$ such that $h_1 = g_1$.

Assume therefore that $h_1 < g_1$. This implies that $g_1 > \epsilon$, and thus, from (14), that $\overline{T}_1(\boldsymbol{g};\rho) = c_1$. Since $\overline{T}_1(\boldsymbol{g};\rho)$ is strictly decreasing in g_1, $h_1 < g_1$ implies that $\overline{T}_1(\boldsymbol{g};\rho) = c_1 < \overline{T}_1((h_1,g_2);\rho)$. However, for $\boldsymbol{h}$ to be an equilibrium, we need to have $\overline{T}_1((h_1,h_2);\rho) \leq c_1 < \overline{T}_1((h_1,g_2);\rho)$. Since $\overline{T}_1(\boldsymbol{g};\rho)$ is increasing in g_2, it yields $h_2 < g_2$, which in turn implies that $g_2 > \epsilon$. The equilibrium $\boldsymbol{g}$ is therefore such that $g_1 > \epsilon$ and $g_2 > \epsilon$. However, since we have assumed that $\boldsymbol{c}$ is not achievable, we know that there exists $i \in \{1,2\}$ such that $\overline{T}_i(\boldsymbol{g}^{NE};\rho) < c_i$. According to (14), this implies that $g_i = \epsilon$. This is a contradiction. We thus conclude that we cannot have two different equilibria. ∎

A.3. Proof of proposition 6

Proof of proposition 6. According to the order of the classes, if class 1 is fair, then $c_2\,\mu_2 \geq c_1\,\mu_1 \geq (1-\rho)^{-1}$. Therefore the Processor Sharing weights satisfy both time constraints. The point $\boldsymbol{g}^{NE} = (\epsilon,\epsilon)$ is clearly the unique Nash equilibrium since both classes have the minimum weight possible and the time constraints are satisfied.

If class 1 is not fair, i.e., $c_1\,\mu_1 < (1-\rho)^{-1}$, then the feasibility of the game implies that $(1-\rho)^{-1} \leq c_2\,\mu_2$. In this case, the equilibrium is achieved in $\boldsymbol{g} = (g_1,\epsilon)$, where g_1 is such that $\overline{T}_1(\boldsymbol{g};\rho) = c_1$ and $\overline{T}_2(\boldsymbol{g};\rho) \leq c_2$. Indeed g_1 is the minimum weight satisfying class-1 time constraint and ϵ is the minimum weight possible for class 2 whose time constraint is satisfied.

From (3), it results that

$$\overline{T}_1(\boldsymbol{g};\rho) = c_1 \iff \frac{g_2}{g_1} = \frac{-\mu_1\rho_2 - \mu_1(1-\rho_1)\,[\mu_1 c_1(1-\rho)-1]}{-\mu_1\rho_2 + \mu_2(1-\rho_2)\,[\mu_1 c_1(1-\rho)-1]},$$

which yields the desired result since $g_2 = \epsilon$. ∎

A.4. Proof of Proposition 7

Proof of Proposition 7. We first note from (11) and (13) that for any weight vector $\boldsymbol{g}$ it holds that

$$\rho_1\overline{T}_1(\boldsymbol{g};\rho) + \rho_2\overline{T}_2(\boldsymbol{g};\rho) \leq \rho_1 c_1 + \rho_2 c_2. \tag{A.1}$$

Let $\boldsymbol{g}^0 = (g_1^0, g_2^0)$ be the starting point of the Best-Response algorithm. If this point satisfies that $\overline{T}_i(\boldsymbol{g}^0;\rho) \leq c_i$ for $i = 1,2$, then the convergence to the equilibrium follows from Proposition 3. Otherwise, (A.1) implies that we have either $\overline{T}_1(\boldsymbol{g}^0;\rho) > c_1$ or $\overline{T}_2(\boldsymbol{g}^0;\rho) > c_2$, but not both.

Assume that $\overline{T}_1(\boldsymbol{g}^0;\rho) > c_1$. Then, the best response of class 1 is to increase its weight to a value g_1^1 such that at point $\boldsymbol{g}^1 = (g_1^1, g_2^0)$ its constraint $\overline{T}_1(\boldsymbol{g}^1;\rho) \leq c_1$ is satisfied as an equality. At this point, we have from (A.1) that $\rho_1\overline{T}_1(\boldsymbol{g}^1;\rho) + \rho_2\overline{T}_2(\boldsymbol{g}^1;\rho) = \rho_1 c_1 + \rho_2\overline{T}_2(\boldsymbol{g}^1;\rho) \leq \rho_1 c_1 + \rho_2 c_2$ and thus that $\overline{T}_2(\boldsymbol{g}^1;\rho) \leq c_2$. We conclude that the weight vector $\boldsymbol{g}^1$ is feasible. Hence, using Proposition 3, we can claim that the best-response algorithm converges to the equilibrium. ■

B. Proofs of Section 4

B.1. Proof of Proposition 9

Proof of Proposition 9. It can be easily proven that if a vector of performance $\mathbf{t}$ is achievable in heavy-traffic then it satisfies (21). For the other implication, we show that a vector $\mathbf{t} \in \mathbb{R}_+^R$ satisfying (21) is achievable in heavy-traffic, i.e., there exists a vector of weights $\boldsymbol{g}$ such that $\overline{T}_i(\boldsymbol{g};1) = t_i$ for all $i \in \mathcal{C}$. Let $\boldsymbol{g}$ be a weight vector such that $\frac{g_i}{g_j} = \frac{t_j/\mathbb{E}(B_j)}{t_i/\mathbb{E}(B_i)}$ for all $i \neq j$. With (20), we have

$$\overline{T}_i(\boldsymbol{g};1) = \mathbb{E}(B_i)\frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)\frac{g_i}{g_k}} = \mathbb{E}(B_i)\frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)\frac{t_k/\mathbb{E}(B_k)}{t_i/\mathbb{E}(B_i)}} = t_i\frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} t_k} = t_i,$$

for all $i \in \mathcal{C}$, where the last inequality follows from (21). We thus conclude that the vector $\mathbf{t}$ is achievable. ■

B.2. Proof of Proposition 10

Proof of Proposition 10. If the problem is feasible in heavy-traffic there exists an achievable vector in heavy-traffic $\mathbf{t} = (t_1,\dots,t_R)$ such that $t_i \leq \tilde{c}_i$, for all i. Then, since $t_i \leq \tilde{c}_i$ for all i, it follows from Proposition 9 that $\sum_i \lambda_i \frac{\mathbb{E}\left(B_i^2\right)}{\mathbb{E}(B_i)}\tilde{c}_i \geq \sum_k \lambda_k \mathbb{E}\left(B_k^2\right)$.

We now focus on the other implication of the proposition. Given a vector of deadlines $\tilde{\boldsymbol{c}} = (\tilde{c}_1,\dots,\tilde{c}_R)$ such that $\sum_k \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)}\tilde{c}_k \geq \sum_k \lambda_k \mathbb{E}\left(B_k^2\right)$, we show that there exists a vector of performances $\mathbf{t}$ achievable in heavy-traffic. Let $\mathbf{t} = (t_1,\dots,t_R)$ be such that

$$t_i = \tilde{c}_i \frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)}\tilde{c}_k},$$

for all i. We observe that t_i is positive for all i and from $\sum_k \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)}\tilde{c}_k \geq \sum_k \lambda_k \mathbb{E}\left(B_k^2\right)$ we derive that $t_i \leq \tilde{c}_i$ for all i. Moreover

$$\sum_k \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} t_k = \sum_k \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c}_i \frac{\sum_i \lambda_i \mathbb{E}\left(B_i^2\right)}{\sum_i \lambda_i \frac{\mathbb{E}\left(B_i^2\right)}{\mathbb{E}(B_i)}\tilde{c}_i} = \sum_i \lambda_i \mathbb{E}\left(B_i^2\right),$$

and we thus conclude with Proposition 9 that the vector $\mathbf{t}$ is achievable. ■

B.3. Proof of Theorem 2

Let us first introduce some results that will be used to prove Theorem 2. Let $\mathbf{g}^m$ be a vector of the form

$$\mathbf{g}^m = (g_1^m, g_2^m, \dots, g_{m-1}^m, \epsilon, \dots, \epsilon), \tag{B.1}$$

where $g_i^m > \epsilon$, if $i < m$. We now show the following property of the vector $\mathbf{g}^m$.

Lemma 3. *If $\overline{T}_m(\boldsymbol{g}^m;1) \leq \tilde{c_m}$, then, for all $j > m$, $\overline{T}_j(\boldsymbol{g}^m;1) \leq \tilde{c}_j$.*

Proof. From (20) and $\overline{T}_m(\boldsymbol{g}^m;1) \leq \tilde{c_m}$ we obtain

$$\frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)/g_k} \leq \tilde{c_m} g_m^m/\mathbb{E}(B_m) = \tilde{c_m}\ \epsilon/\mathbb{E}(B_m)$$

Since for all $j > m$ we have that $\tilde{c_m}/\mathbb{E}(B_m) \leq \tilde{c_j}/\mathbb{E}(B_j)$ and since $g_j = \epsilon$ for all $j \geq m$, we have for all $j > m$

$$\frac{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_k \lambda_k \mathbb{E}\left(B_k^2\right)/g_k} \leq \tilde{c_m}\ \epsilon/\mathbb{E}(B_m) \leq \tilde{c_j}\ \epsilon/\mathbb{E}(B_j) \iff \overline{T}_j(\boldsymbol{g};1) \leq \tilde{c_j}.$$

■

We are now in position to proof the result of theorem 2.

Proof of theorem 2.
Let m be the minimum value such that $\overline{T}_m(\boldsymbol{g}^m;1) \leq \tilde{c_m}$, where $\boldsymbol{g}^m$ is as defined in (B.1). According to Lemma 3, we have that $\overline{T}_k(\boldsymbol{g}^m;1) \leq \tilde{c}_k$, for $k \geq m$. On the other hand, we choose g_k such that $\overline{T}_k(\boldsymbol{g}^m;1) = \tilde{c}_k$ for all $k < m$. It then results that $\boldsymbol{g}^m$ is the equilibrium since in case any of the first $m-1$ coordinates of $\boldsymbol{g}^m$ diminishes its weight its time constraint is not satisfied and the rest of the coordinates of $\mathbf{g}^m$ are ϵ.

We now characterize the first $m-1$ components of the equilibrium. From (20), it follow that $\frac{g_i^m}{g_j^m} = \frac{\tilde{t_j}/\mathbb{E}(B_j)}{\tilde{t_i}/\mathbb{E}(B_i)}$ for all $i \neq j$, where $\overline{T}_i(\boldsymbol{g}^m;1) = \tilde{t_i}$. Since $\tilde{t_i} = \tilde{c_i}$ for all $i < m$, we can state that for all $i < m$

$$\frac{g_i^m}{g_m^m} = \frac{\tilde{t_m}/\mathbb{E}(B_m)}{\tilde{c_i}/\mathbb{E}(B_i)} \iff g_i^m = \epsilon\ \frac{\tilde{t_m}/\mathbb{E}(B_m)}{\tilde{c_i}/\mathbb{E}(B_i)}.$$

Finally, we prove that $\overline{T}_m(\boldsymbol{g}^m;1) = \tilde{t_m} \leq \tilde{c_m}$ is equivalent to (22). Hence, using (20), we obtain

$$\tilde{c_m} \geq \tilde{t_m} = \mathbb{E}(B_m)\frac{\sum_{k=1}^R \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_{k=1}^R \lambda_k \mathbb{E}\left(B_k^2\right)\frac{g_m^m}{g_k^m}} = \mathbb{E}(B_m)\frac{\sum_{k=1}^R \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_{k=1}^{m-1} \lambda_k \mathbb{E}\left(B_k^2\right)\frac{\tilde{c_k}/\mathbb{E}(B_k)}{\tilde{t_m}/\mathbb{E}(B_m)} + \sum_{k=m}^R \lambda_k \mathbb{E}\left(B_k^2\right)}$$

Since $\tilde{t_m} \leq \tilde{c_m}$ holds since the problem is feasible, rearranging both sides of the equation we derive the expression (22)

$$\frac{\tilde{t_m}}{\mathbb{E}(B_m)} = \frac{\sum_{k=1}^R \lambda_k \mathbb{E}\left(B_k^2\right) - \sum_{k=1}^{m-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} c_k}{\sum_{k=m}^R \lambda_k \mathbb{E}\left(B_k^2\right)}.$$

■

B.4. Proof of theorem 3

We show the uniqueness of the equilibrium proving that if the equilibrium is $\mathbf{g^m}$, then $\mathbf{g^{m+i}}$ is not the equilibrium, for $i = 1, \ldots, R-m$. We thus consider that there exists a value m satisfying

$$\frac{\tilde{c_m}}{\mathbb{E}(B_m)} \geq \frac{t_m}{\mathbb{E}(B_m)} = \frac{\sum_{k=1}^R \lambda_k \mathbb{E}\left(B_k^2\right) - \sum_{k=1}^{m-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c_k}}{\sum_{k=m}^R \lambda_k \mathbb{E}\left(B_k^2\right)}$$

which is equivalent to

$$\frac{t_m}{\mathbb{E}(B_m)} \sum_{k=m}^{R} \lambda_k \mathbb{E}(B_k^2) = \sum_{k=1}^{R} \lambda_k \mathbb{E}\left(B_k^2\right) - \sum_{k=1}^{m-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c_k} \tag{B.2}$$

We will see that for any $i = 1, \ldots, R-m$, $\mathbf{g^{m+i}}$ that satisfies (22) is not the equilibrium. To do so, we show that there is no vector $\mathbf{g^{m+i}}$ with weights as defined in theorem 2 that verifies

We suppose that there exist a value $i = 1, \ldots, R-m$ such that

$$\frac{\tilde{c_{m+i}}}{\mathbb{E}(B_{m+i})} \geq \frac{\overline{t_{m+i}}}{\mathbb{E}(B_{m+i})} = \frac{\sum_{k=1}^{R} \lambda_k \mathbb{E}\left(B_k^2\right) - \sum_{k=1}^{m+i-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c_k}}{\sum_{k=m+i}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)} \tag{B.3}$$

is verified.

It thus follows that

$$\frac{\tilde{c_{m+i}}}{\mathbb{E}(B_{m+i})} \geq \frac{\overline{t_{m+i}}}{\mathbb{E}(B_{m+i})} = \frac{\sum_{k=1}^{R} \lambda_k \mathbb{E}\left(B_k^2\right) - \sum_{k=1}^{m-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c_k} - \sum_{k=m}^{m+i-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c_k}}{\sum_{k=m+i}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)}$$

Taking into account the equality of (B.2) and that $\frac{\tilde{c_m}}{\mathbb{E}(B_m)} \leq \frac{\tilde{c_k}}{\mathbb{E}(B_k)}$ for all $k > m$, we derive

$$\frac{\overline{t_{m+i}}}{\mathbb{E}(B_{m+i})} = \frac{\frac{t_m}{\mathbb{E}(B_m)} \sum_{k=m}^{R} \lambda_k \mathbb{E}(B_k^2) - \sum_{k=m}^{m+i-1} \lambda_k \frac{\mathbb{E}\left(B_k^2\right)}{\mathbb{E}(B_k)} \tilde{c_k}}{\sum_{k=m+i}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)} \leq \frac{\frac{t_m}{\mathbb{E}(B_m)} \sum_{k=m}^{R} \lambda_k \mathbb{E}(B_k^2) - \frac{\tilde{c_m}}{\mathbb{E}(B_k)} \sum_{k=m}^{m+i-1} \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_{k=m+i}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)}$$

We now use that $t_m \leq \tilde{c_m}$ since $\mathbf{g^m}$ is the equilibrium to state that

$$\frac{\overline{t_{m+i}}}{\mathbb{E}(B_{m+i})} \leq \frac{\frac{t_m}{\mathbb{E}(B_m)} \sum_{k=m}^{R} \lambda_k \mathbb{E}(B_k^2) - \frac{\tilde{c}_m}{\mathbb{E}(B_k)} \sum_{k=m}^{m+i-1} \lambda_k \mathbb{E}\left(B_k^2\right)}{\sum_{k=m+i}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)} = \frac{\frac{\tilde{c_m}}{\mathbb{E}(B_m)} \sum_{k=m+i}^{R} \lambda_k \mathbb{E}(B_k^2)}{\sum_{k=m+i}^{R} \lambda_k \mathbb{E}\left(B_k^2\right)} = \frac{\tilde{c}_m}{\mathbb{E}(B_m)}$$

From the relation $\frac{g_k^{m+i}}{g_j^{m+i}} = \frac{\tilde{t_j}/\mathbb{E}(B_j)}{\tilde{t_k}/\mathbb{E}(B_k)}$, the above inequality implies that the component m of the equilibrium $\mathbf{g^{m+i}}$ is less than the $m+i$-th component, i.e., $g_{m+i}^{m+i} \leq g_m^{m+i}$, which is in contradiction the result of corollary 5.