Energy-Aware Capacity Scaling in Virtualized Environments with Performance Guarantees[*]

J. Anselmi[a] and I.M. Verloop[a,b]

[a]BCAM - Basque Center for Applied Mathematics,
Bizkaia Technology Park, Derio, 48170, Spain
[b]Université de Toulouse, IRIT-CNRS,
2 rue C. Camichel, 310171 Toulouse Cedex 7, France

Abstract

We investigate the trade-off between performance and power consumption in servers hosting virtual machines running IT services. The performance behavior of such servers is modeled through Generalized Processor Sharing (GPS) queues enhanced with a green speed-scaling mechanism that controls the processing capacity to use depending on the number of active virtual machines. When the number of virtual machines grows large, we show that the stochastic evolution of our model converges to a system of ordinary differential equations for which we derive a closed-form formula for its unique stationary point. This point is a function of the capacity and the shares that characterize the GPS mechanism. It allows us to show that speed-scaling mechanisms can provide large reduction in power consumption having only small performance degradation in terms of the delays experienced in the virtual machines. In addition, we derive the optimal choice for the shares of the GPS discipline, which turns out to be non-trivial. Finally, we show how our asymptotic analysis can be applied to the dimensioning and service partitioning in data-centers. Experimental results show that our asymptotic formulas are accurate even when the number of virtual machines is small.

1 Introduction

In the context of Information Technology (IT), virtualization has emerged as a practical way to implement service differentiation and to facilitate the deployment of services among servers. A service can be thought of as a web application or even as an entire network infrastructure dedicated to an IT business provider. Virtualization technologies are very common in the management of IT systems because of their isolation properties that enhance flexibility, reliability, security and utilization of resources. Matter of fact, they are the enabling tool for the development of cloud computing infrastructures and of consolidation projects, where the goal is to reduce the management costs of data-centers while satisfying performance and availability constraints.

A virtualization technology allows multiple operating systems or virtual machines (VMs) to be running simultaneously on one server, and each of these VMs may be running one or more IT services. A virtualization monitor, or hypervisor, is a software that allows the creation of these VMs and determines the resource-sharing mechanism. The latter establishes how the VMs access to common server resources such as cpus, disks or bandwidth. Typically, hypervisors implement some form of weighted fair scheduling [28], where such weights are usually called *shares*. For instance, the commercial VMware ESX Server supports a proportional-share allocation mechanism [1]. The shares ensure isolation among competing VMs by guaranteeing a minimum fraction of the overall processing capacity to each active VM, so that a VM is protected against unexpected workload peaks in another VM. So far, the qualitative behavior of the mean response time in each VM, which is an important performance metric, is not well-understood as a function of the shares.

[*]Research partially supported by grant MTM2010-17405 (Ministerio de Ciencia e Innovación, Spain) and grant PI2010-2 (Department of Education and Research, Basque Government).

The generalized processor sharing (GPS) queueing model [22] emerged in the literature as a robust approximation to capture the behavior of virtualized environments; e.g., [9, 6, 1]. Under GPS, each VM is associated with a positive number (corresponding to the share) guaranteeing each VM with a minimum fraction of the overall capacity. The surplus capacity from non-active VMs is reallocated proportionally over all active VMs. The performance analysis of GPS queues attracted the attention of several researchers during the last years, but accurate and efficient analyses only exist for systems with a very limited number of VMs; we refer to [25] for a complete overview. However, nowadays multi-core blade servers can host even hundreds of VMs, which resulted in the development of approximation results. Matter of fact, large-scale consolidation projects either focus on server utilizations only [24], providing rule-of-thumb guarantees for response times, or rely on rough bounds of response times in terms of some product-form queueing network [21, 6], which are computationally tractable but can be arbitrarily inaccurate when the system is sufficiently loaded because their stability condition changes [31]. We observe that server utilizations do not depend on the specific resource allocation algorithm implemented by the virtualization software because it is commonly work-conserving. On the other hand, response times do depend on it.

An important problem in the management of IT enterprises is power consumption [2] that, in the last decade, triggered substantial research aimed at finding energy-aware designs at all levels from chips to data-centers. For instance, at the data-center level, consolidation projects reduce energy usage by reducing the number of servers to use [24, 6, 13], while, at chipset level, speed-scaling designs reduce energy usage by varying the frequency speed of the cpu depending on the overall load [30, 7, 3]. Several approaches can thus be undertaken to find a greener system but in presence of virtualized services and large networks, it is not clear to what extent this is permitted because of the lack of accurate systematic frameworks able to trade-off between power consumption and performance.

In this paper, we provide a systematic optimization framework to trade-off between energy consumption and performance in virtualized systems hosting several VMs. We achieve this goal by studying the GPS queueing model enhanced with a robust speed-scaling mechanism that controls the processing capacity of each cpu core depending on the number of active VMs. Motivated by the fact that real servers host a large and growing number of VMs, we analyze our model in the limiting regime where the number of VMs grows proportionally with the number of cpu cores. In the limit, we show that the stochastic evolution of our model converges to a system of (deterministic) ordinary differential equations having a unique stationary point, which we use to approximate the long-term performance and power consumption in closed-form. The speed of convergence is $O(1/\sqrt{N})$, where N is the total number of VMs. Numerically, we observe that our asymptotic formulas are accurate even when the number of VMs is small, e.g., the average percentage relative error on queue lengths (or response time) is $\approx 6\%$ with only 10 VMs.

Using the popular function $s^\alpha + c$, $\alpha > 1$, $c \geq 0$, for the power consumption of one cpu core working at speed s (see [30, 7, 29, 3]), we then characterize the exact energy benefit and performance loss of our model with respect to the standard GPS queue without speed-scaling. In the limiting regime, we obtain the following results:

i) A considerable reduction in power consumption can be obtained with a controllable performance degradation. Furthermore, we establish when using all the available cpu cores with our speed-scaling function outperforms the policy of turning cores on/off depending on the load (core-parking).

ii) We derive the shares in GPS that minimize the mean stationary response time. The optimal share of each VM is given by its load and, surprisingly, it does not depend explicitly on the service rates, as is usual the case in stochastic scheduling; e.g., the classic $c\mu$-rule.

iii) The simplicity of our asymptotic formulae allows for the development of other convex optimization frameworks to trade-off between performance and power consumption at other levels of IT management; e.g., at data-center level. Examples are discussed in the context of optimal routing in parallel servers and data-center consolidation.

The paper is organized as follows. Section 2 presents the model under investigation and Section 3 introduces the proposed speed-scaling mechanism. In Section 4 we analyze the system behavior in a limiting regime where the number of VMs grows large, and the unique stationary point is obtained. In Section 5 the trade-off between power reduction and performance guarantees is discussed. Section 6 describes how our asymptotic analysis applies to the performance and power consumption optimization in data-centers, with particular emphasis on the selection of the optimal shares. Experimental results are shown in Section 7 and, finally, Section 8 draws the conclusions of our work outlining future research.

2 GPS queueing model

We model a server hosting several virtual machines (VMs) each providing an IT service. Each VM is associated to a class that describes the service it implements. There are a total of R classes, and $N_r > 0$ is the number of VMs providing service of class r (referred to as a class-r VM), $r = 1, \dots, R$. Let $N \overset{\text{def}}{=} \sum_r N_r$ denote the total number of VMs, $\beta_r \overset{\text{def}}{=} N_r/N$, and hence $\sum_r \beta_r = 1$.

Each VM of class r accommodates an infinite stream of incoming jobs having mean arrival rate λ_r and having mean size (amount of work) μ_r^{-1}. We assume that the arrival process to each VM is Poisson and that the job sizes are i.i.d. and exponentially distributed (as outlined in the Conclusions, our analysis could extend to phase-type distributed service requirements and/or inter-arrival times). We define the load corresponding to one class-r VM by $\rho_r \overset{\text{def}}{=} \frac{\lambda_r}{\mu_r}$ and we define $\rho \overset{\text{def}}{=} \sum_r \beta_r \rho_r$, so that $N\rho$ represents the total load offered to the system.

Let $M_{ri}^{(N)}(t)$ be the proportion of class-r VMs having i jobs at time t when the total number of VMs is N, hence $0 \leq M_{ri}^{(N)}(t) \leq \beta_r$. We refer to $M^{(N)}(t) \overset{\text{def}}{=} (M_{ri}^{(N)}(t))$, $r = 1, \dots, R$, $i \in \mathbb{N}$, as the *state* of the system at time t. By definition, $\sum_{i \geq 0} M_{ri}^{(N)}(t) = \beta_r$.

We model the sharing dynamics of the processing capacity in a server hosting multiple VMs with a GPS queueing model [22] in its packetized version as in [9, 6, 1]. Under GPS, the fraction of available processing capacity dedicated to an active class-r VM in state $M^{(N)}$ is

$$\frac{\phi_r}{\sum_{s=1}^{R} \phi_s N(\beta_s - M_{s0}^{(N)})}, \tag{1}$$

i.e., each VM is associated a positive number ϕ_r, $r = 1, \dots, R$, (called share), and the available capacity is scaled proportionally to the shares and the set of active VMs.[1] Hence, each VM of class r is *guaranteed* to receive the fraction $\frac{\phi_r}{N \sum_{s=1}^{R} \beta_s \phi_s}$ of the available processing capacity. Finally, jobs belonging to the same VM are handled in any work-conserving manner. Upon completion of service, a job leaves the system returning to its issuer.

In the traditionally studied GPS queue, the processing capacity of the system is fixed, say equal to aN. In this case, the system is stable (i.e., positive recurrent) if and only if $\rho < a$. In order to reduce power consumption, we propose and analyze in this paper a dynamic speed-scaling mechanism that determines the capacity that is used at each moment in time. The specific dynamic speed-scaling control that we consider in the paper is defined in Section 3.

As performance measure, we are interested in the response time (or sojourn time) of jobs being served by a class-r VM, $S_r^{(N)}(t)$, $r = 1, \dots, R$. We denote by

$$Q_r^{(N)}(t) \overset{\text{def}}{=} \frac{1}{\beta_r} \sum_i i M_{ri}^{(N)}(t) \tag{2}$$

the number of jobs, or queue length, at time t in a generic class-r VM (by symmetry, the queue length of class-r VMs have the same distribution). Therefore, $N_r \mathbb{E}(Q_r^{(N)}(t))$ is the mean number of jobs in all class-r VMs, and $\sum_r N_r \mathbb{E}(Q_r^{(N)}(t))$ is the mean total number of jobs in the system. The mean response time (or sojourn time) at time t of each class-r VM follows by Little's law [19]:

$$\mathbb{E}(S_r^{(N)}(t)) = \frac{1}{\lambda_r}\, \mathbb{E}(Q_r^{(N)}(t)). \tag{3}$$

Another crucial measure is the power consumption, which is a function of the speed the system works at. This will be developed in more detail in the next section. A table summarizing our notation can be found in Appendix A for quick reference.

[1] We assume that the load of each VM can be always spread over the whole fraction of capacity (1). This assumption is valid if each VM is configured to access to all the available cpu cores and if the system is not in light-load conditions. The latter is typically true, since consolidation projects ensure that servers are utilized at ≈ 70% of their maximum capabilities.

3 Energy-aware capacity scaling

The processing unit of a server hosting multiple VMs is composed of all the cpu cores of the machine. We assume we have aN, $a > 0$, cores all having an identical maximum processing speed or capacity equal to 1. Hence, the processing unit has a total maximum capacity equal to aN.

Motivated by the fact that even a single server hosts a large and growing number of services (even hundreds) and that many VMs are just replicas of other machines implementing the same service (increasing reliability), we are interested in the behavior of the system when the number of VMs (N) grows large, while keeping the proportion of services from the different classes, i.e., the vector $(\beta_1, \dots, \beta_R)$, fixed. Note that if instead of a capacity aN we would take sub or super linear growths for the processing capacity, this would imply the instability of the per-VM number of jobs or the system overprovisioning, respectively, since the total load offered to the system is ρN.

3.1 Power consumption and performance

The power consumption of the server depends on the speed at which the cores work. We denote by $f(s) : [0, 1] \to \mathbb{R}^+$ the power consumed by one core working at speed s. We assume that $f(s)$ is a continuous and increasing function. In case of fixed speeds, i.e., each core works at speed 1 whenever at least one VM is active, the expected power consumption of the system at time t is given by

$$\tilde{W}^{(N)}(t) = aN(f(1)\ \mathbb{P}(\sum_r M_{r0}^{(N)}(t) < 1) + f(0)\ \mathbb{P}(\sum_r M_{r0}^{(N)}(t) = 1)), \tag{4}$$

where the first (second) term represents the power consumption when the system is active (idle). Since for the system with fixed speed $\lim_{t\to\infty} \mathbb{P}(\sum_r M_{r0}^{(N)}(t) < 1) = \frac{\rho}{a}$, one has that in steady state the expected power consumption equals

$$N(f(1)\rho + f(0)(a - \rho)). \tag{5}$$

We note that the power consumption grows linearly in N and in the server *utilization*, which is in agreement with the observations in [23]. In fact, we recall that ρ/a is the long-term proportion of time where the server is busy for the system with fixed capacity aN. Within the foregoing assumptions, it represents the stationary server utilization by means of the utilization's law [19].

If at each moment in time all the aN cores work at their maximum speed one, the mean stationary total number of jobs is known to be finite if and only if $\rho < a$, and independent of the actual value of N. Hence, as the number of VMs (N) grows large, the mean number of jobs in each VM, and hence the mean response time, will converge to zero. For instance, assume $R = 1$. In this case, the mean (stationary) total number of jobs in the system is known to be $\frac{\rho}{a-\rho}$ (since the total arrival rate is λN and the available capacity is aN). By symmetry, $\frac{\rho}{N(a-\rho)}$ jobs are present in mean in each of the N VMs, which approaches zero as N grows large. By Little's law [19], the delay experienced by a generic job is therefore $\frac{\rho}{\lambda N(a-\rho)}$.

As N grows large, the argument above shows that the system yields *overprovisioning of processing capacity*, because the mean response time for a job in any VM will converge to 0 as N grows large. This observation gives rise to the following question: *Can we find a greener usage of the overall processing capacity in order to reduce the power consumption, at the cost of letting the mean response time in each VM become larger but still controllable?*

3.2 Speed scaling mechanism

Motivated by previous question, we propose the following speed-scaling mechanism, which we will analyze in the remainder of this paper. Under our speed-scaling rule, the overall available capacity in state $M^{(N)}(t)$ is

$$aN(1 - \sum_r M_{r0}^{(N)}(t)), \tag{6}$$

instead of aN. In other words, the overall capacity is varied dynamically depending on the number of VMs that are active. We will interpret the speed-scaling as follows: each of the aN (identical) cores works at processing speed $(1 - \sum_r M_{r0}^{(N)}(t))$. Hence, the principle behind the speed-scaling (6) is simple and robust:

the larger the number of active VMs, the larger the speed of each core. If all VMs are active, then the full capacity aN is used.

From a practical standpoint, the capacity scaling (6) can be easily implemented by the virtualization software that at each time knows the number of active VMs. From an analytical standpoint, (6) could be further complicated by introducing dependency on the terms $M_{ri}^{(N)}(t)$, for $i > 0$. However, since the systems under investigation have many VMs, a control that depends heavily on the state of each VM would induce a non-negligible overhead in practice.

Coherently with (4), the expected power consumption of the system at time t with the new speed-scaling mechanism (6) is

$$W^{(N)}(t) = aN\, \mathbb{E}(f(1 - \sum_r M_{r0}^{(N)}(t))), \tag{7}$$

since at time t each of the aN cores is working at speed $1 - \sum_r M_{r0}^{(N)}(t)$.

Remark 1. *In the following, all quantities referring to scaling* (6) *(respectively, scaling Na) are denoted without (with) a tilde.*

The potential reduction in the power consumption of the proposed speed-scaling (6) comes at the cost of some performance loss, as stated in the following proposition:

Proposition 1. *For all $r, N, t > 0$, if $\tilde{Q}_r^{(N)}(0) \leq_{st} Q_r^{(N)}(0)$, then $\tilde{Q}_r^{(N)}(t) \leq_{st} Q_r^{(N)}(t)$, where $\leq_{st}$ is the usual stochastic order.*

Proof. The proof follows by a simple coupling argument between the sample-paths of the stochastic processes $\tilde{Q}_r^{(N)}(t)$ and $Q_r^{(N)}(t)$. □

In the following we will analyze the behavior of the system under the proposed speed-scaling mechanism, see Section 4, with as main goal to compare $\tilde{W}^{(N)}(t)$ and $\tilde{Q}^{(N)}(t)$ with $W^{(N)}(t)$ and $Q^{(N)}(t)$ in order to quantify the reduction in power consumption and the performance loss of the proposed speed-scaling mechanism. The latter will be discussed in Section 5.

4 Large-scale system

In this section, we analyze the GPS system when the proposed speed-scaling mechanism (see (6)) is applied. Let $e_{r,i}$ be the $R \times \mathbb{N}$-matrix with all zeros except for component (r, i) which is one. The process $M^{(N)}(t)$ is a continuous-time Markov chain having the following transition rates: For the arrivals of jobs to class-r VMs, we have

$$M^{(N)} \;\to\; M^{(N)} + \frac{1}{N}(e_{r,i+1} - e_{r,i}) \;\text{ at rate }\; \lambda_r M_{ri}^{(N)} N,\; i = 0, 1, \ldots, \tag{8}$$

because there are $M_{ri}^{(N)} N$ class-r VMs having i jobs in the queue and in each VM a new job arrives at rate λ_r. For the departures of jobs from class-r VMs, we have

$$M^{(N)} \;\to\; M^{(N)} - \frac{1}{N}(e_{r,i+1} - e_{r,i}) \;\text{ at rate }\; \mu_r \frac{\phi_r}{\sum_s \phi_s(\beta_s - M_{s0}^{(N)})} a \left(1 - \sum_s M_{s0}^{(N)}\right) M_{ri+1}^{(N)} N, \tag{9}$$

$i = 0, 1, \ldots$, because there are $M_{ri+1}^{(N)} N$ class-r VMs having $i + 1$ jobs in the queue receiving a fraction of $\phi_r / \sum_s \phi_s N(\beta_s - M_{s0}^{(N)})$ of the total available capacity $aN(1 - \sum_s M_{s0}^{(N)})$. Hence, the process $M^{(N)}(t)$ is a *density-dependent population process*, as defined in [12].

In the case of equal weights, $\phi_r = C$, for all r, the system can be analyzed in closed form. In fact, the GPS system reduces to a system of N independent M/M/1 queues, of which N_r queues have arrival rate λ_r and service rate $\mu_r a$. This follows since the fraction of capacity dedicated to a class-r VM is equal to $1/N(1 - \sum_s M_{s0}^{(N)})$ (see (1)) and the capacity available is $aN(1 - \sum_s M_{s0}^{(N)})$ (see (6)), hence each class-r VM receives capacity a at any moment in time. Therefore, the mean stationary number of jobs in each VM is given by $\mathbb{E}(Q_r^{(N)}(\infty)) = \frac{\rho_r}{a-\rho_r}$ and the *total* number of jobs grows linearly in N, i.e., $\sum_r \mathbb{E}(Q_r^{(N)}(\infty)) = N \sum_r \beta_r \frac{\rho_r}{a-\rho_r}$. Furthermore, since the N queues behave independently, and a class-r VM is active with probability ρ_r/a, the fraction of active VMs is $\frac{1}{N}\sum_{r=1}^{R} \frac{N_r \rho_r}{a} = \rho/a$. Hence, from Equation (7), we get that the mean power consumption is given by $W^{(N)}(\infty) = aNf(\frac{\rho}{a})$, which grows linearly in N. In addition, we note that if the function $f(\cdot)$ is convex, then the mean power consumption is indeed reduced under our proposed speed-scaling mechanism, see Equation (5).

For unequal shares exact analysis seems to be difficult. Motivated by the above case of equal shares, we expect that the total number of jobs will grow linearly in N. In order to investigate this, we study the stochastic process $M^{(N)}(t)$ as the number of VMs N grows large. Since $M^{(N)}(t)$ is a density-dependent population process, we expect the evolution of $M^{(N)}(t)$ to converge to a deterministic limit $m(t)$, called the *mean-field limit.* (Convergence to $m(t)$ will be discussed in Section 4.1.) The dynamics of $m(t)$ is described by the expected *drift* of the system [12]. In our case, the mean-field limit $m(t)$ is the solution of a system of ODEs described by the transition rates given in (8) and (9):

$$\begin{aligned} \dot{m}_{r0} &= -\lambda_r m_{r0} + \mu_r \tfrac{\phi_r}{\sum_s \phi_s(\beta_s - m_{s0})} a\left(1 - \textstyle\sum_s m_{s0}\right) m_{r1}, \\ \dot{m}_{ri} &= \lambda_r (m_{r,i-1} - m_{ri}) - \mu_r \tfrac{\phi_r}{\sum_s \phi_s(\beta_s - m_{s0})} a\left(1 - \textstyle\sum_s m_{s0}\right)(m_{ri} - m_{r,i+1}), \quad i = 1, 2, \ldots, \end{aligned} \tag{10}$$

$m_{ri} \geq 0$, and $\sum_i m_{ri} = \beta_r$, for all $r = 1, \ldots, R$.

The above system of ODEs has a unique stationary point as described in the following theorem.

Theorem 1. *If*

$$\rho < a \cdot \frac{\sum_s \beta_s \rho_s / \phi_s}{\max(\rho_r/\phi_r)}, \tag{11}$$

then (10) *has a unique stationary point* $\bar{m}$ *that is given by, for all* $r = 1, \ldots, R$,

$$\begin{aligned} \bar{m}_{r0} &= \beta_r \left(1 - \frac{\rho}{a} \frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}}\right) \\ \bar{m}_{ri} &= \bar{m}_{r0} \left(\frac{\rho}{a} \frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}}\right)^i, \quad i = 0, 1, \ldots \end{aligned} \tag{12}$$

Proof. Setting the ODEs (10) equal to zero and summing the i^{th} and $i+1^{th}$ equations, we derive that

$$m_{ri} = m_{r0} \left(\frac{\rho_r}{\phi_r} \frac{\sum_s \phi_s(\beta_s - m_{s0})}{a(1 - \sum_s m_{s0})}\right)^i, \quad i = 1, 2, \ldots \tag{13}$$

Substituting (13) in the normalizing condition $\sum_{i=0}^{\infty} m_{ri} = \beta_r$, we have that m_{r0} must be the solution of

$$m_{r0} = \beta_r - \beta_r \frac{\rho_r}{\phi_r} \frac{\sum_s \phi_s(\beta_s - m_{s0})}{a(1 - \sum_s m_{s0})}, \ \forall r, \tag{14}$$

for $m_{r0} \neq \beta_r$ (note that the point $m_{r0} = \beta_r$, $\forall r$, is not a stationary point for (10)). The system of equations (14) can be written as

$$\begin{cases} \dfrac{\phi_1}{\beta_1 \rho_1}(\beta_1 - m_{10}) = \dfrac{\phi_s}{\beta_s \rho_s}(\beta_s - m_{s0}), & \forall s > 1, \\ \dfrac{\sum_s \phi_s(\beta_s - m_{s0})}{a(1 - \sum_s m_{s0})} = \dfrac{\phi_1}{\beta_1 \rho_1}(\beta_1 - m_{10}) \end{cases} \tag{15}$$

From the first equation in (15), we have $(\beta_s - m_{s0}) = \frac{\beta_s \rho_s}{\phi_s} \frac{\phi_1}{\beta_1 \rho_1}(\beta_1 - m_{10})$, which we can substitute in the right-hand term of (14). Simplifying the common terms $\beta_1 - m_{10}$, which cannot be zero, we obtain m_{r0} as in (12). □

We will refer to condition (11) as the *asymptotic stability condition* of the system, because it guarantees that $\bar{m}_{r0} > 0$, for all r, i.e., there is a positive fraction of class-r VMs empty, for all classes r. We expect that as $N \to \infty$ the stability condition of the system with N VMs will in fact converge to the condition as given in (11). We observe that the asymptotic stability condition depends on the shares ϕ_r. This does not come as a surprise since these weights influence the fraction of VMs that are active which determines the capacity at which the system works. In particular, we observe that when the shares are $\phi_r = C\rho_r$, for all r, the asymptotic stability condition is maximized, and is equal to $\rho < a$. This coincides with the stability condition of the standard GPS queue without speed scaling. We also see that if we let the relative share of class 1 go to zero, i.e., $\phi_1/\phi_r \to 0$, $r \neq 1$, the asymptotic stability condition equals $\rho < a\beta_1$. This can be understood as follows: class-1 jobs will get only served when no jobs are present in any of the other classes. The capacity at which the system works on class 1 is equal to $aN(1 - \sum_{r\neq 1} \beta_r) = aN\beta_1$ (since it is only served when $M_{r0}^{(N)} = \beta_r$, for all $r \neq 1$). Apparently, as N grows large, the number of active VMs in the other classes is of an order smaller than N, so that the total available capacity for the whole system equals $aN\beta_1 + o(N)$. Since the total load offered to the system equals ρN, as $N \to \infty$ the stability condition converges to $\rho < a\beta_1$.

4.1 Convergence to mean-field limit

We now discuss the relation between the mean-field limit $m(t)$ and the density-dependent population process $M^{(N)}(t)$ as N grows large. In order to avoid technicalities, we assume (only in this section) that each VM has a finite buffer of size B. The convergence in the case of infinite buffer will be verified numerically in Section 7.

In the case of finite buffers, the fixed point $\bar{m}$ slightly changes (remaining unique) and, by normalizing terms to make sure that $\sum_{i=0}^{B} \bar{m}_{ri} = \beta_r$, it is given by

$$\begin{aligned} \bar{m}_{r0} &= \beta_r \left(1 - \frac{\rho}{a}\frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}}\right)\left(1 - (\frac{\rho}{a}\frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}})^{B+1}\right)^{-1}, \\ \bar{m}_{ri} &= \bar{m}_{r0}\left(\frac{\rho}{a}\frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}}\right)^{i}, \quad i = 0, 1, \ldots, B. \end{aligned} \tag{16}$$

The next two theorems show properties of the convergence of the process $M^{(N)}(t)$. The proofs are in Appendix B, and follow by verifying the hypotheses of Theorems 2.1 and 2.3 of [12, Chapter 11]. Let $E \overset{\text{def}}{=} \{m \in \mathbb{R}^{R(B+1)} : m_{ri} \geq 0, \sum_{i=0}^{B} m_{ri} = \beta_r\}$ denote the state space of the process $m(t)$.

Theorem 2. *Assume each VM has a finite buffer of size B. If $\lim_{N\to\infty} M^{(N)}(0) = m_0 \in E$, then for any $t > 0$, we have*

$$\lim_{N\to\infty} \sup_{0\leq s\leq t} |M^{(N)}(s) - m(s)| = 0, \ \textit{almost surely,} \tag{17}$$

where $m(t)$ is the unique solution of (10) *(truncated to $i = B$) with initial condition $m(0) = m_0$.*

Let us rewrite the system of ODEs (10) as $\dot{m} = F(m)$. Let $V^{(N)}(t) \overset{\text{def}}{=} \sqrt{N}(M^{(N)}(t) - m(t))$ and $V(t) \overset{\text{def}}{=} V(0) + U(t) + \int_0^t \partial F(m(s))V(s)ds$, where $U(t)$ is a time-inhomogeneous Brownian motion and ∂F denotes the Jacobian matrix of F. The following theorem shows that the speed of convergence of $M^{(N)}(t)$ to $m(t)$ is $O(1/\sqrt{N})$.

Theorem 3. *Assume each VM has a finite buffer of size B. If $V^{(N)}(0) \xrightarrow{d} V(0)$, then $V^{(N)}(t) \xrightarrow{d} V(t)$, where $\xrightarrow{d}$ denotes convergence in distribution.*

Theorems 2 and 3 show the (speed of) convergence of $M^{(N)}(t)$ to $m(t)$, as $N \to \infty$. Unfortunately, they do not imply that $m(t)$ will eventually converge to $\bar{m}$, as $t \to \infty$. In fact, such convergence may be prevented by limit cycles or chaotic behavior of the ODEs (10). In order to prevent such strange behavior, one needs to show that $\bar{m}$ is a global attractor, i.e., all the trajectories of $m(t)$ converge to $\bar{m}$ (see Corollary 5 of [15]). In the case that all shares are equal, $\bar{m}$ is indeed a global attractor because the set of ODEs (10) can be interpreted as the Kolmogorov equations of R independent M/M/1 queues. However, in general, proving that $\bar{m}$ is a global attractor is a difficult task because the stochastic process $M^{(N)}(t)$ is not monotone, and classic arguments, as used for example in the proof of [14, Theorem 4.5], cannot be applied. In Section 7, we give numerical evidence that $\bar{m}$ is a global attractor in the case of unequal shares.

4.2 Stationary behavior

The following proposition describes the steady-state behavior of the density-dependent population process, assuming it exists.

Theorem 4. *Assume each VM has a finite buffer and Equation (11) is satisfied. If the point $\bar{m}$, the equilibrium point as obtained in Theorem 1, is a global attractor of $m(t)$, then the steady-state distribution of $M^{(N)}(t)$ concentrates around $\bar{m}$, as N grows large.*

Proof. From Theorem 1 we obtain that $\bar{m}$ is the unique fixed point of the ODE. If $\bar{m}$ is a global attractor, then by [15, Corollary 5] we obtain the result. □

In what follows, we make the hypothesis that the point $\bar{m}$ is a global attractor. (As mentioned above, we believe this to be true.) Motivated by the above proposition, we approximate the steady-state behavior of the system with N VMs by the equilibrium point $\bar{m}$.

We now calculate the stationary queue length of each VM and the stationary power consumption in the mean-field limit, which will serve as approximations for the steady-state queue lengths and power consumption in the original system with N VMs. Assuming symmetry in the queue lengths of VMs of the same class, the stationary queue length of a class-r VM in the mean-field limit, defined by q_r, is given by

$$q_r = \tfrac{1}{\beta_r} \textstyle\sum_{i>0} i\bar{m}_{ri} = \frac{\rho \frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}}}{\left(a - \rho \frac{\frac{\rho_r}{\phi_r}}{\sum_s \beta_s \frac{\rho_s}{\phi_s}}\right)}, \tag{18}$$

where we substituted $\bar{m}$ in (2). The mean response time follows by Little's law (see (3)). Similarly, for the scaled power consumption, i.e., $\frac{W^{(N)}(t)}{N}$, we have that its stationary point in the mean-field limit is given by

$$w = af(1 - \textstyle\sum_r \bar{m}_{r0}) = af(\tfrac{\rho}{a}), \tag{19}$$

where we substituted $\bar{m}$ in (7) and used the continuity of the function $f(\cdot)$.

We note that Equations (18) and (19) coincide with the formulas as obtained in the beginning of Section 4 for the case of equal shares.

5 Reduction in power consumption

In this section, we evaluate the power-consumption reduction and performance loss of our speed-scaling mechanism (6) by comparing it to the standard GPS system without speed-scaling. Theoretical research on speed-scaling designs states that the power consumed by a core working at speed s can be approximated by (see, e.g., [30, 7, 29, 3])

$$f(s) = Cs^\alpha + c, \tag{20}$$

where $\alpha > 1$ (typically between 2 and 3) [17] and $c \geq 0$. We assume that the set of values that s can take is continuous. This is an approximation, since in practice this set is finite.

In what follows we take for $f(\cdot)$ the polynomial form as given in (20), and set w.l.o.g. $C = 1$. Similar analysis can be performed analogously with other power-consumption functions, but we leave this issue as future research.

In the standard GPS system, using (5) we find that the stationary power consumption (4) scaled with N is $\tilde{w} = \rho + ca$. On the other hand, for a GPS system with our speed-scaling (6), the power consumption in the mean-field limit is given by

$$w = a\left(\frac{\rho}{a}\right)^\alpha + ca \text{ if (11) is satisfied.} \tag{21}$$

This expression is obtained from (19). In case $c \approx 0$ and (11) is satisfied, the power consumption is decreasing in a. This may appear counter-intuitive. However, as a increases, the mean processing speed of each core decreases proportionally (speed $1-\sum_r M_{r0}^{(N)}(t)$ converges to $\frac{\rho}{a}$). Consequently, the power consumption of each of the aN cores decreases with $O(1/a^\alpha)$, and we can save energy because $\alpha > 1$.

Our proposed speed-scaling mechanism provides a factor

$$\frac{w}{\tilde{w}} = \frac{(\frac{\rho}{a})^\alpha + c}{\frac{\rho}{a} + c}$$

of reduction in the power consumption compared to the standard GPS system (if the stability condition (11) is satisfied). This improvement can be significant. For instance, assume that the server load or utilization [19] of the system is $\rho/a = 0.7$, which is a common operating point of data-center servers, and let $\alpha = 2.5$ and $c = 0.2$. Then, the asymptotic power consumption is reduced by a factor of ≈ 0.6.

We note that the stationary queue length in a VM for the standard GPS system converges to 0 as $N \to \infty$, see Section 3, while under our proposed power-scaling the queue length of a VM is estimated by the controlled value (18), which is strictly positive. Hence, a performance loss occurs. However, it is important to note that q_r is decreasing in a. Hence, as a grows large, that is, we have a very large number of cores, the per-VM queue length, q_r, approaches zero under the power-scaling (6) (if $a \to \infty$ then $\bar{m}_{0r} \approx \beta_r$, and hence $q_r \approx 0,\ \forall r$).

On the other hand, one can calculate the optimal number of cores a needed in the system in order to minimize power consumption while keeping the system stable. Under our speed-scaling mechanism, the latter is given by

$$a^* = \rho \max\left\{\frac{\max_r \rho_r/\phi_r}{\sum_s \beta_s \rho_s/\phi_s}, \left(\frac{\alpha-1}{c}\right)^{1/\alpha}\right\}.$$

The minimum scaled power consumption can be obtained by substituting a^* in Equation (21) and is denoted by w^*. The optimal choice of cores for the standard GPS system is $\tilde{a}^* = \rho$, so that the scaled minimum power consumption equals $\tilde{w}^* = \rho + c\rho$.

When $c \geq 0$ is small enough we can make the following observation:

Observation 1. *When c is small enough, the optimal number of cores under the speed-scaling mechanism is $a^* = \rho(\frac{\alpha-1}{c})^{1/\alpha}$ and $w^* = \alpha\rho(\frac{c}{\alpha-1})^{1-1/\alpha}$. This provides a factor $\frac{w^*}{\tilde{w}^*} = \frac{\alpha}{1+c}\left(\frac{c}{\alpha-1}\right)^{1-1/\alpha} < 1$ of reduction in the minimum power consumption compared with the standard GPS system. In addition, if c is small enough (and hence the number of optimal cores is large enough), all performance guarantees on the per-queue delays are met under the speed-scaling mechanism with the optimal number of cores.*

5.1 Comparison with core parking

To reduce power consumption dynamically, a technique currently used in Hyper-V and Windows Server 2008 R2 is "core parking", where cpu cores are turned on/off depending on the server load. We observe that the analysis in Section 4 can be used as well for a specific implementation of core-parking. In fact, the processing speed in state $M^{(N)}$, given by (6), could be interpreted as having $aN(1-\sum_r M_{r0}^{(N)}(t))$ cores turned on and working at maximum speed one, while all the other cores are switched off. The queue length in each VM under this core parking strategy is therefore given by q_r, see Equation (18), i.e., it gives the same performance as our proposed speed-scaling mechanism (6). Assuming that the cost of turning on/off cores and the setup times are negligible, and that cores that are turned off do not consume power, the expected power consumption of the system at time t under this core-parking policy is

$$W_{CP}^{(N)}(t) = aN\mathbb{E}(1-\sum_r M_{r0}^{(N)}(t))f(1). \tag{22}$$

Taking $f(\cdot)$ equal to (20) and using Theorem 2, the scaled stationary power consumption consumed in the mean-field limit is $w_{CP} = \rho + c\rho$, when (11) is satisfied. Recall that $\tilde{w} = \rho + ca$. Hence, we conclude that core-parking has less power consumption compared to the standard GPS system (i.e., no speed-scaling), since $\rho < a$. However, in its turn, our proposed speed-scaling rule where a^*N cores are working at load-dependent speeds outperform the core parking strategy in case c is small enough, and reduces the power consumption by a factor $\frac{\alpha}{1+c}\left(\frac{c}{\alpha-1}\right)^{1-1/\alpha} < 1$.

6 Optimization Frameworks

In this section we use the large-scale analysis of Section 4 to discuss several optimization frameworks: In Section 6.1 we find the optimal shares of the GPS discipline. In Sections 6.2 and 6.3 we consider the trade-off between performance and power consumption at other levels of IT enterprise management; e.g., at data-center level.

6.1 Selection of optimal shares

From Equation (19) we observe that the asymptotic scaled power consumption w does not depend on the shares ϕ_r. It is therefore natural to ask which values should the shares have in order to optimize the performance (for example to minimize response times). The selection of optimal shares attracted the attention of several researchers, but so far analysis focused on a limited number of VMs; e.g., [20, 26, 11, 18]. Using the expressions obtained from the large-scale analysis, we are able to derive the optimal weights for the system as the number of VMs grows to infinite ($N \to \infty$).

We aim at finding the best share vector $\phi = (\phi_1, \dots, \phi_R)$ that minimizes the response time subject to service level agreement (SLA) constraints. In the following, we assume $a = 1$ for simplicity. By Little's law, $\frac{1}{\lambda}\sum_r \beta_r q_r(\phi)$ represents the overall response time, where we write $q_r(\phi)$ instead of q_r (as given in (18)) to emphasize the dependence on the weight vector ϕ. Hence, we aim at solving the following optimization problem:

$$\begin{array}{rl} \min_\phi & \frac{1}{\lambda}\sum\limits_r \beta_r q_r(\phi) \\ \text{s.t.} & q_r(\phi) \le \lambda_r \overline{S}_r,\ \forall r \\ & \rho\frac{\rho_r}{\phi_r} \le \sum_s \beta_s \frac{\rho_s}{\phi_s} - \epsilon,\ \forall r \\ & \phi_r \ge 0,\ \forall r. \end{array} \tag{23}$$

Here $\overline{S}_r$ is the bound on the maximum mean response time for class-r VMs, $\lambda = \sum_r \beta_r \lambda_r$, and the constraints $\rho\frac{\rho_r}{\phi_r} \le \sum_s \beta_s \frac{\rho_s}{\phi_s} - \epsilon$, $r = 1, \dots, R$, ensure the existence of the stationary point (i.e., the asymptotic stability condition (11) holds), with $\epsilon > 0$ an arbitrarily small constant. Numerical experiments with Ipopt [27] reveal that the optimal shares can be computed efficiently. Problem (23) can be further enriched with reliability features when mixes $(\beta_1, \dots, \beta_R)$ become part of the optimization process.

In case the first family of constraints in (23) is removed (or equivalently $\overline{S}_r$ is very large), the optimization problem can be solved in closed form.

Theorem 5. *Assume $\overline{S}_r = \infty$. The optimal choice for the shares ϕ_r in* (23) *are $\phi_r = \rho_r$, $r = 1, \dots, R$.*

Proof. With the change of variable $\overline{\phi}_r = \phi_r^{-1}$, introducing $y_r = (\sum_s \beta_s \rho_s \overline{\phi}_s - \rho\rho_r\overline{\phi}_r)^{-1}$, and using expression (18), formulation (23) can be rewritten as

$$\begin{array}{rl} z^* \stackrel{\text{def}}{=} \min_{\overline{\phi}} & \frac{1}{\lambda}\sum\limits_r \beta_r \rho\rho_r \overline{\phi}_r y_r \\ \text{s.t.} & \rho\rho_r\overline{\phi}_r \le \sum_s \beta_s\rho_s\overline{\phi}_s - \epsilon,\ \forall r \\ & \frac{1}{y_r} = \sum_s \beta_s\rho_s\overline{\phi}_s - \rho\rho_r\overline{\phi}_r,\ \forall r \\ & \overline{\phi}_r \ge 0,\ \forall r. \end{array} \tag{24}$$

Define the new optimization problem

$$\begin{array}{rl} z^+ \stackrel{\text{def}}{=} \min_{\overline{\phi},y} & \frac{1}{\lambda}\sum\limits_r \beta_r \rho\rho_r \overline{\phi}_r y_r \\ \text{s.t.} & \rho\rho_r\overline{\phi}_r \le \sum_s \beta_s\rho_s\overline{\phi}_s - \epsilon,\ \forall r \\ & \frac{1}{y_r} \le \sum_s \beta_s\rho_s\overline{\phi}_s - \rho\rho_r\overline{\phi}_r,\ \forall r \\ & \overline{\phi}_r \ge 0,\ \forall r, \end{array} \tag{25}$$

in decision variables $y = (y_r)$ and $\overline{\phi} = (\overline{\phi}_r)$. Formulation (25) is convex [8] (while (24) is not). In addition, $z^+ \le z^*$ because the set of constraints in (25) is weaker than the one of (24).

The KKT conditions of (25), which are necessary and sufficient for a point to be an optimum (because of convexity), are

$$\tfrac{1}{\lambda}\beta_r\rho\rho_r y_r + B_r\rho\rho_r - \beta_r\rho_r \textstyle\sum_s B_s + C_r\rho\rho_r - \beta_r\rho_r \textstyle\sum_s C_s - D_r = 0,\ \forall r \tag{26}$$

$$\tfrac{1}{\lambda}\beta_r\rho\rho_r\overline{\phi}_r - C_r\tfrac{1}{y_r^2} = 0,\ \forall r, \tag{27}$$

where B_r, C_r, D_r are Lagrange multipliers, plus the complementarity slackness and feasibility equations. We now check whether the point $\overline{\phi}_r = 1/\rho_r, y_r = (1-\rho)^{-1},\ \forall r$, satisfies the KKT conditions above. In this point, $\overline{\phi}_r > 0$, hence, by the complementary slackness conditions $D_r\overline{\phi}_r = 0$, we must have $D_r = 0$, for all r. In addition, the first family of constraints in (25) in this specific point becomes $\rho \leq 1-\epsilon$, which does not hold strictly because we have assumed $\rho < 1$. Hence, by the complementary slackness condition we obtain $B_r = 0, \forall r$. Now, noting that $y_r > 0$, (26) and (27) simplify into

$$\tfrac{1}{\lambda}\beta_r\rho\rho_r\frac{1}{1-\rho} + C_r\rho\rho_r - \beta_r\rho_r \textstyle\sum_s C_s = 0,\ \forall r, \tag{28}$$

$$C_r = \tfrac{\beta_r}{\lambda}\tfrac{\rho}{(1-\rho)^2},\ \forall r. \tag{29}$$

Substituting (29) into (28), we obtain an identity, i.e., the KKT conditions are satisfied, and hence the point $\overline{\phi}_r = 1/\rho_r, y_r = (1-\rho)^{-1},\ \forall r$, minimizes (25). Since this is also a feasible point for (24) and $z^+ \leq z^*$, we conclude that this point minimizes (24) as well. □

We note that for the standard GPS system, i.e., fixed capacity, the optimal choice for the shares has been studied in [26] for the two-VM case ($N = 2$). It was shown that the shares that minimize the mean overall response time are trivial, i.e., under the optimal shares the GPS policy behaves as a priority queue where priority is given to the VM having largest service rate μ_r. This might be unwanted since in a priority queue the important isolation property among competing classes of the GPS discipline is lost. Interestingly, for our proposed power scaling (6), the optimal shares are non-trivial, hence retaining some isolation property. In fact, the optimal shares $\phi_r = \rho_r$ are such that the asymptotic stability condition (11) is maximum, i.e., it coincides with $\rho < a = 1$. We finally note that the optimal weights do not depend explicitly on the service requirement parameters μ_r, as is usual the case in stochastic scheduling, e.g., the $c\mu$-rule.

6.2 Optimal routing

In several contexts, e.g, web-server farms, jobs arrive to a central dispatcher that is in charge of routing them to a set of parallel servers hosting VMs. The problem of finding the routing strategy that minimizes the mean response time of a parallel system of resources is a well-known problem in queueing theory; see, e.g., [10, 5]. In the following, we provide a systematic framework to optimize performance and power consumption in parallel servers enhanced with our speed-scaling mechanism.

Assume that class-r jobs arrive to the central dispatcher following a Poisson arrival process with intensity $\bar{\lambda}_r N$. Let K be the number of parallel servers. In what follows we will use the superscript $^{(k)}$ when referring to parameters and functions corresponding to server k. Server k hosts $\beta_r^{(k)}N$ VMs dedicated to class-r services, has capacity $a^{(k)}N$ and applies GPS with weights $\phi_r^{(k)}$ using energy-efficient speed scaling as proposed in (6). Assume that the routing policy is probabilistic (see [16]), i.e., with probability $p_r^{(k)}$ a job requiring service of class r is forwarded to server k. We further assume that once a class-r job is sent to server k, it arrives in a uniformly chosen class-r VM of server k. Note that the arrival rate to a class-r VM in server k is given by $\lambda_r^{(k)} \overset{\text{def}}{=} \bar{\lambda}_r p_r^{(k)}/\beta_r^{(k)}$, hence $\rho_r^{(k)} = \lambda_r^{(k)}/\mu_r$ and $\rho^{(k)} = \sum_r \beta_r^{(k)}\rho_r^{(k)}$. For a given routing policy $p = (p_1^{(1)}, \ldots, p_1^{(K)}, \ldots, p_R^{(1)}, \ldots, p_R^{(K)})$, the queue length in a class-r VM of server k is given by $q_r^{(k)}(p)$ as defined in Equation (18) and the power consumption equals $w^{(k)}(p) = a^{(k)}\left(\frac{\rho^{(k)}}{a^{(k)}}\right)^{\alpha} + ca^{(k)}$, see (19).

We aim at finding a routing probability vector p that minimizes a weighted sum of the response time and total power consumption, i.e.,

$$
\begin{array}{ll}
\min_p & \frac{c_1}{\lambda}\sum_{k,r}\beta_r^{(k)} q_r^{(k)}(p) + c_2\sum_k w^{(k)}(p) \\
\text{s.t.} & \rho^{(k)}\frac{\rho_r^{(k)}}{\phi_r^{(k)}} \le a^{(k)}\sum_s \beta_s^{(k)}\frac{\rho_s^{(k)}}{\phi_s^{(k)}} - \epsilon,\ \forall r,k \\
& \sum_k p_r^{(k)} = 1, \forall r \\
& p_r^{(k)} \ge 0, \forall r,k
\end{array}
\tag{30}
$$

with $c_1, c_2 \ge 0$ some constants, $\lambda = \sum_r \bar{\lambda}_r$. The first condition represents the asymptotic stability condition, with $\epsilon > 0$ arbitrarily small. Given that problem (30) is convex, optimization solvers can solve it efficiently (in polynomial time) in an exact manner [8].

The optimization problem simplifies in case the shares are also part of the optimal routing problem. For a given load balancing vector p, we know that the shares that minimize the objective function of (30) are such that $\phi_r^{(k)} = \rho_r^{(k)}$, for all r, see Theorem 5. Hence, under the assumption that each server will set his shares according to the traffic load it receives, the load-balancing problem simplifies to

$$
\begin{array}{ll}
\min_{(\rho_r^{(k)})} & \frac{c_1}{\lambda}\sum_k \frac{\rho^{(k)}}{a^{(k)} - \rho^{(k)}} + c_2\sum_k a^{(k)}\left(\frac{\rho^{(k)}}{a^{(k)}}\right)^\alpha \\
\text{s.t.} & \rho^{(k)} = \sum_r \beta_r^{(k)}\rho_r^{(k)},\ \forall r,k \\
& \rho^{(k)} \le a^{(k)} - \epsilon,\ \forall r,k \\
& \sum_k \beta_r^{(k)}\rho_r^{(k)} = \bar{\lambda}_r/\mu_r, \forall r \\
& \rho_r^{(k)} \ge 0, \forall r,k.
\end{array}
$$

Note that we rewrote the optimization problem in terms of $(\rho_r^{(k)})$ instead of p. One observes that the above load-balancing problem is equivalent to an optimal routing problem in parallel multi-class processor-sharing queues. Again, being the resulting problem convex, it can be solved efficiently.

6.3 Data-center consolidation

Formally stated, the data-center consolidation problem reads as follow: Given a data-center, a set of servers and a set of services (commonly web-applications), the problem is to deploy services to servers to minimize the total number of servers to use (and thus energy and space) while ensuring performance and reliability constraints; we point the reader to [24] for further details. As outlined in the introduction, consolidation projects commonly use virtualization technologies but existing work either limits the focus on server utilizations only or relies on rough bounds for response time. Since the mean-field formula (18) is exact as the number of VMs per server grows, we believe that existing mathematical formulations, e.g., [21, 6, 24], can be adapted to achieve a better exploitation of resources. For example, one could achieve more flexibility in the dimensioning and service partitioning of data-centers if in the optimization problem (30) one would also set as decision variables: i) the shares $\phi_r^{(k)}$, ii) the $\beta_r^{(k)}$'s (the fraction of VMs dedicated to the various classes of services), iii) the $a^{(k)}$'s (the capacity of the servers), or iv) whether or not a server is chosen to be used [24] (thus introducing a binary variable). We leave as future research the development and the analysis of the resulting optimization problems.

7 Experimental Results

The goal of this section is to show numerically that i) $M^{(N)}(\infty)$ converges to $\bar{m}$ as $N \to \infty$, and that ii) our mean-field analysis is accurate even when N is small.

7.1 Stationarity

As discussed in Section 4.1, a sufficient condition for $M^{(N)}(\infty) \to \bar{m}$ to hold is that $\bar{m}$ is a global attractor of the system of ODEs (10), see Theorem 4. In the case of equal shares this condition is indeed satisfied, see

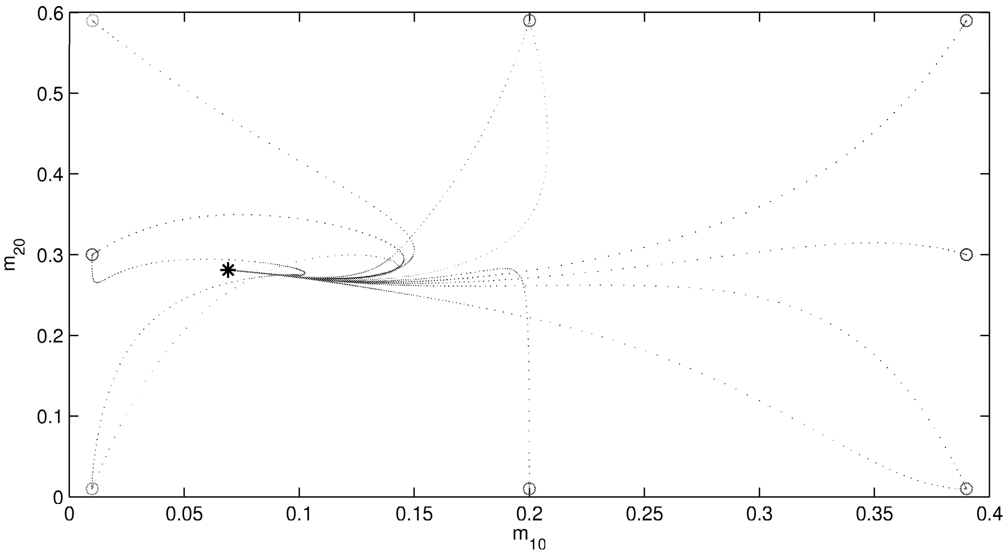

Figure 1: Evolution in time of components $\dot{m}_{10}$ and $\dot{m}_{20}$ of the system of ODEs (10). The starting conditions are denoted with a circle.

Section 4.1. The goal of this section is to simulate the system of ODEs to give numerical evidence that $\bar{m}$ is a global attractor for unequal shares as well.

Figure 1 shows the evolution in the phase diagram of $\dot{m}_{10}$ and $\dot{m}_{20}$ starting from 12 different initial conditions (denoted with a circle) of a model with the following parameters: $R = 2$, $(\lambda_1, \lambda_2) = (0.5, 1.5)$, $(\mu_1, \mu_2) = (1, 2)$, $(\phi_1, \phi_2) = (0.3, 0.7)$, $(\beta_1, \beta_2) = (0.4, 0.6)$, and $a = 1$. Within these parameters, the load is close to the asymptotic stability condition as given in (11) ($\rho = 0.65 < 0.78$). In the figure, we see that all the trajectories converge to our fixed point $\bar{m}$ (denoted with an asterisk). The trajectories have been computed using the `ode45` function of Matlab (note that they refer to a system with several dimensions, which is the reason why they seem to overlap in the figure). We have repeated this procedure over 1,000 randomly generated models from 100 random initial conditions by varying $R = 2, \ldots, 5$ and $N = 2, \ldots, 10$. In all cases, we observed that our fixed point behaves as a global attractor.

7.2 Accuracy

In this section we test the accuracy of approximating the stationary behavior of the system with the fixed point analysis of the mean-field limit. By Theorem 4 we expect the approximation to be accurate as the number of VMs grows large, $N \to \infty$. By numerical tests we observe that the error of our analysis is already small for a small number of VMs.

In order to test the accuracy we measure the percentage relative errors, which we define as

$$Err_Q(N, R) \overset{\text{def}}{=} \tfrac{1}{R} \textstyle\sum_{r=1}^{R} \frac{|Q_{r,sim} - q_r|}{Q_{r,sim}} \cdot 100\%, \;\; Err_W(N, R) \overset{\text{def}}{=} \frac{|W_{sim} - Nw|}{W_{sim}} \cdot 100\%. \tag{31}$$

where q_r and w are as given in (18) and (19), and $Q_{r,sim} = Q_{r,sim}(N, R)$ and $W_{sim} = W_{sim}(N, R)$ are the average stationary number of jobs in a class-r VM and power consumption (computed via simulation), respectively, in the GPS system with N VMs hosted on a server with capacity N that uses the speed-scaling as given in (6). Since the considered queueing process is a multi-dimensional birth-and-death Markov chain with N dimensions, the application of standard solution techniques such as solving the global balance equations are computationally intractable even when $N = 5$. We therefore use simulation in order to solve the model up to $N = 10$ VMs, provided that the system is not heavily loaded.

In Table 1, we give $Err_Q(N, 2)$ and $Err_W(N, 2)$ by varying N and keeping fixed $\frac{\rho}{\rho_{\max}} = 0.7$, where $\rho_{\max}$ is the right-hand term of (11) (the maximum stability condition). Therefore, quantity $\frac{\rho}{\rho_{\max}}$ is how much the system is stressed with respect to its maximum capability. A typical server utilization is $\approx 70\%$. Each number in the table represents the error (31) averaged over 200 experiments. For each experiment, we have randomly generated $N_r \in \{1, \ldots, N\}$, $\phi_r \in (0, 1)$, $\beta_r \in (0, 1)$, $\lambda_r \in [0.1, 100]$, and $\mu_r \in [0.1, 100]$, according to uniform distributions such that $\frac{\rho}{\rho_{\max}} = 0.7$. Parameter a was fixed to 1, and we chose the power consumption

	$N=4$	$N=6$	$N=8$	$N=10$
$Err_Q(N,2)$	17.08%	10.79%	8.41%	6.23%
$Err_W(N,2)$	14.03%	9.23%	7.85%	6.46%

Table 1: Errors (31) by increasing the number of VMs N and keeping fixed $\frac{\rho}{\rho_{\max}} = 0.7$.

function $f(s) = s^2$. We see that already for $N = 10$, the errors of our asymptotic formulas are small. For higher values of N, we expect an improved accuracy by means of Theorem 4. Since data-center servers host even hundreds of VMs, we conclude that our formulas can be used to predict power consumption and performance behavior of virtualized environments.

8 Conclusions

We have proposed a systematic framework to trade-off between performance and energy consumption in virtualized environments composed of servers with several virtual machines and modeled by GPS queues. Asymptotically, the evolution of our stochastic model is captured by a set of (deterministic) ODEs having a unique stationary point that we use to approximate and optimize long-term performance and power consumption in closed form. Our results indicate that a simple and robust speed-scaling mechanism can significantly reduce power consumption with a small controllable performance degradation. Furthermore, our analysis lets us answer important open problems such as the selection of the optimal share for each VM, which turns out to be given by its load, and the optimal routing in parallel servers, which is equivalent to the optimal routing in M/M/1 processor-sharing queues.

We leave as future work the application of our results to data-center consolidation problems (see as well Section 6.3) and to game-theoretic analysis of virtualized environments. With respect to the latter point, cloud-computing providers use virtualization to sell whole network infrastructures to IT business providers, who typically set access prices to clients maximizing their own profits (as in, e.g., [4]). The competition that emerges in this game affects the overall performance. In order to analyze the latter, explicit formulas for the delays should be obtained.

Finally, our mathematical analysis assumes that job inter-arrival times and sizes are exponentially distributed. It is worth noting that a similar analysis can be performed under the assumption that inter-arrival times and job sizes are phase-type distributed. In this case, the drift equations (10) must keep track of which stage of the phase-type distribution jobs are.

Acknowledgments

The authors are very grateful to D. Ardagna, U. Ayesta, T. Bonald, P. Cremonesi, N. Gast, and A. Sansottera for several discussions and comments that significantly increased the quality of this paper.

References

[1] ESX server performance and resource management for CPU-intensive workloads. *VMware white paper, 2005.*

[2] U.S. environmental protection agency. EPA report on server and data center energy efficiency. 2007.

[3] L. L. Andrew, M. Lin, and A. Wierman. Optimality, fairness, and robustness in speed scaling designs. In *Proceedings of SIGMETRICS*, pages 37–48, New York, NY, USA, 2010. ACM.

[4] J. Anselmi, U. Ayesta, and A. Wierman. Competition yields efficiency in load balancing games. In *Performance Evaluation*, To appear.

[5] J. Anselmi and B. Gaujal. The price of forgetting in parallel and non-observable queues. In *Performance Evaluation*, To appear.

[6] D. Ardagna, B. Panicucci, M. Trubian, and L. Zhang. Energy-aware autonomic resource allocation in multi-tier virtualized environments. *IEEE Transactions on Services Computing*, 99, 2010.

[7] N. Bansal, T. Kimbrel, and K. Pruhs. Speed scaling to manage energy and temperature. *J. ACM*, 54:3:1–3:39, 2007.

[8] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, March 2004.

[9] A. Chandra, W. Gong, and P. Shenoy. Dynamic resource allocation for shared data centers using online measurements. In *Proceedings of ACM SIGMETRICS*, pages 300–301, NY, USA, 2003. ACM.

[10] M. B. Combé and O. J. Boxma. Optimization of static traffic allocation policies. *Theor. Comput. Sci.*, 125(1):17–43, 1994.

[11] A. Elwalid and D. Mitra. Design of generalized processor sharing schedulers which statistically multiplex heterogeneous QoS classes. In *INFOCOM*, pages 1220–1230, 1999.

[12] S. Ethier and T. Kurtz. *Markov Processes: Characterization and Convergence.* Wiley, 1986.

[13] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. A. Kozuch. Optimality analysis of energy-performance trade-off for server farm management. *Perform. Eval.*, 67:1155–1171, November 2010.

[14] A. Ganesh, S. Lilienthal, D. Manjunath, A. Proutiere, and F. Simatos. Load balancing via random local search in closed and open systems. In *Proc. of SIGMETRICS*, pages 287–298, NY, USA, 2010. ACM.

[15] N. Gast and B. Gaujal. A mean field model of work stealing in large-scale systems. In *Proceedings of SIGMETRICS*, pages 13–24, NY, USA, 2010. ACM.

[16] X. Guo, Y. Lu, and M. S. Squillante. Optimal probabilistic routing in distributed parallel queues. *SIGMETRICS Perf. Eval. Review*, 32(2):53–54, 2004.

[17] S. Kaxiras and M. Martonosi. *Computer Architecture Techniques for Power-Efficiency.* Morgan and Claypool, 2008.

[18] K. Kumaran, G. Margrave, D. Mitra, and K. Stanley. Novel techniques for the design and control of generalized processor sharing schedulers for multiple QoS classes. In *INFOCOM*, pages 932–941, 2000.

[19] E. D. Lazowska, J. Zahorjan, G. Graham, and K. C. Sevcik. *Quantitative system performance: computer system analysis using queueing network models.* Prentice-Hall, Upper Saddle River, NJ, US, 1984.

[20] P. Lieshout, M. Mandjes, and S. C. Borst. GPS scheduling: selection of optimal weights and comparison with strict priorities. In *Proceedings of SIGMETRICS*, pages 75–86, NY, USA, 2006. ACM.

[21] Z. Liu, M. S. Squillante, and J. L. Wolf. On maximizing service-level-agreement profits. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 213–223, New York, NY, USA, 2001. ACM.

[22] A. K. Parekh and R. G. Gallagher. A generalized processor sharing approach to flow control in integrated services networks: the multiple node case. *IEEE/ACM Trans. Netw.*, 2(2):137–150, 1994.

[23] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu. No “power” struggles: coordinated multi-level power management for the data center. In *Proc. of 13th Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 48–59, NY, USA, 2008. ACM.

[24] B. Speitkamp and M. Bichler. A mathematical programming approach for server consolidation problems in virtualized data centers. *IEEE Trans. Serv. Comput.*, 3:266–278, October 2010.

[25] M. van Uitert. *Generalized Processor Sharing Queues.* PhD dissertation, 2003.

[26] I. M. Verloop, U. Ayesta, and S. C. Borst. Monotonicity properties for multi-class queueing systems. *Discrete Events Dynamic Systems*, 20:473–509, 2010.

[27] A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[28] Z. Wang, X. Zhu, P. Padala, and S. Singhal. Capacity and performance overhead in dynamic resource allocation to virtual containers. In *Integrated Network Management*, pages 149–158, 2007.

[29] A. Wierman, L. L. H. Andrew, and A. Tang. Power-aware speed scaling in processor sharing systems. In *INFOCOM*, pages 2007–2015, 2009.

[30] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced CPU energy. In *Proc. of the 36th FOCS '95*, pages 374–382, Washington, DC, USA, 1995. IEEE Computer Society.

[31] Z.-L. Zhang, D. F. Towsley, and J. F. Kurose. Statistical analysis of generalized processor sharing scheduling discipline. *IEEE Journal on Selected Areas in Communications*, 13(6):1071–1080, 1995.

Appendix

A Notation

See Table 2 for a summary of the notation used in the paper.

R	number of classes
N	number of Virtual Machines (VMs)
N_r	number of class-r VMs
β_r	$= N_r/N$
λ_r	mean arrival rate of jobs to a class-r VM
μ_r^{-1}	mean size of jobs to a class-r VM
ρ_r	$= \lambda_r/\mu_r$
ρ	$= \sum_r \beta_r \rho_r$
ϕ_r	> 0, share of one class-r VM
Na	number of cpu cores with maximum capacity 1
$M_{ri}^{(N)}(t)$	proportion of class-r VMs having i jobs at time t
$M^{(N)}(t)$	$= (M_{ri}^{(N)}(t), \forall r = 1, \dots, R, \forall i = 0, 1, \dots)$
$\tilde{Q}_r^{(N)}(t)$	number of jobs in one class-r VM at time t with power-scaling Na
$Q_r^{(N)}(t)$	number of jobs in one class-r VM at time t with power-scaling (6)
$\tilde{W}_r^{(N)}(t)$	mean power consumption at time t with power-scaling Na
$W_r^{(N)}(t)$	mean power consumption at time t with power-scaling (6)
$\tilde{S}_r^{(N)}(t)$	response time of jobs in one class-r VM at time t with power-scaling Na
$S_r^{(N)}(t)$	response time of jobs in one class-r VM at time t with power-scaling (6)
$f(s)$	power consumed by one cpu core when working at speed s
α	a positive number
$m(t)$	mean-field limit of the process $M^{(N)}(t)$
q_r	stationary number of jobs in one class-r VM with power scaling (6) in the mean-field limit
$\tilde{w}$	stationary power consumption with power-scaling Na in the mean-field limit
w	stationary power consumption with power-scaling (6) in the mean-field limit

Table 2: Summary of the notation used in the paper.

B Proof of Theorems 2 and 3

As discussed in Section 4, the stochastic process $M^{(N)}(t)$ belongs to the family of *density-dependent population processes* [12], i.e., the transition rate from $M^{(N)}$ to $M^{(N)} + \ell/N$, $\ell \subset \mathbb{N}^{C+1}$, has the form $Nb_\ell(M^{(N)})$ where $b_\ell(M^{(N)})$ does not depend on N (in the notation used in Chapter 11 of [12], b_ℓ is called β_ℓ). Let us rewrite the system of ODEs (10) as $\dot{m} = F(m)$. Note that $F(m) = \sum_\ell \ell b_\ell(m)$.

Using Theorems 2.1 and 2.3 in Chapter 11 of [12], Theorems 2 and 3 follow directly if we prove that

i) $\sum_\ell |\ell| \sup_{m\in K} b_\ell(m) < \infty$ for each compact $K \subset E$,

ii) $\sum_\ell |\ell|^2 \sup_{m\in K} b_\ell(m) < \infty$ for each compact $K \subset E$,

iii) F is Lipschitz on each compact $K \subset E$.

Conditions i) and ii) follow immediately, since there are only a finite number of possible values for l with strictly positive transition rates $b_\ell(\cdot)$, and the latter being uniformly bounded. We now prove condition iii). For the (r,i) component of F, we have

$$\begin{aligned}\left|\frac{\partial F_{r,i}}{\partial m_{s,0}}\right| &\le \left| -\lambda_r \mathbf{1}_{\{r=s\}} + \mu_r\phi_r \frac{-\sum_k \phi_k(\beta_k - m_{k0}) + \phi_s(1-\sum_k m_{k0})}{(\sum_k \phi_k(\beta_k - m_{k0}))^2}(m_{r,i+1} - m_{ri}\mathbf{1}_{\{i>0\}})\right| \\ &\le \lambda_r + \mu_r\phi_r\left(\frac{m_{r,i+1} + m_{ri}\mathbf{1}_{\{i>0\}}}{\sum_k \phi_k(\beta_k - m_{k0})} + \frac{\phi_s(1-\sum_k m_{k0})}{(\sum_k \phi_k(\beta_k - m_{k0}))^2}(m_{r,i+1} + m_{ri}\mathbf{1}_{\{i>0\}})\right).\end{aligned} \tag{32}$$

Now, for each i

$$\frac{m_{r,i}}{\sum_k \phi_k(\beta_k - m_{k0})} \le \frac{\beta_r - m_{r0}}{\sum_k \phi_k(\beta_k - m_{k0})} \le \frac{1}{\phi_r}, \tag{33}$$

and

$$\frac{\phi_s(1-\sum_k m_{k0})}{(\sum_k \phi_k(\beta_k - m_{k0}))^2} m_{ri} \le \frac{\phi_s(1-\sum_k m_{k0})^2}{(\sum_k \phi_k(\beta_k - m_{k0}))^2} \le \frac{\phi_s}{(\min_k \phi_k)^2}, \tag{34}$$

hence the expression in (32) is upper bounded. Using a similar argument one obtains that $\left|\frac{\partial F_{r,i}}{\partial m_{s,j}}\right|$, $j > 0$, is upper bounded as well, and hence F is Lipschitz.