A Study of Live VM Migration for Server Consolidation in Data Centers

Alexandre Brandwjan[†], Thomas Begin[‡], Hind Castel-Taleb[§], Tulin Atmaca[§]
[†] Baskin School of Engineering, University of California, Santa Cruz, Santa Cruz, USA
[‡] Univ Lyon, UCB Lyon 1, ENS Lyon, Inria, CNRS, LIP UMR 5668, Lyon, France
[§] SAMOVAR, Télécom SudParis, CNRS, Université Paris-Saclay, Évry, France

***Abstract*—This paper presents a theoretical study of the combined effect of Virtual Machine (VM) migration and Physical Machine (PM) consolidation on energy consumption, end-user task performance and equipment wear. Our numerical results indicate that a moderately aggressive consolidation strategy seems to be a good compromise. They indicate also the potential importance of distributional assumptions for time between VM launch requests and for VM lifetime.**

***Index Terms*—Server consolidation, VM migration, Live migration, Energy performance trade-offs, Hierarchical model.**

I. Introduction

In the past several years, cloud computing with virtualized data centers has emerged as the dominant computing paradigm. With this technology, Virtual Machines (VMs) running on a set of Physical Machines (PMs or servers) can share the available physical resources. Benefits of this approach include improved overall resource utilization and lower energy requirements.

In a dynamic environment, VMs are launched (created) and terminated in response to customer requests. At VM launch, one of the objectives of the initial VM placement may be to balance (or to maximize) the PM utilization. With live VM migration, [1], it is possible to reposition a running VM on a different PM.

There are a number of potential motivations for live VM migration. Server consolidation to reduce energy consumption is one of them [2] and our paper focuses specifically on this important objective. It is intuitively clear that live VM migration may have a significant effect on power consumption and VM performance. In general, however, this effect doesn't seem easy to predict.

Cloud performance tends to be difficult to evaluate due to the scale of the system, its dynamic nature and interactions between numerous cloud components. In addition to performance, one needs to take into account energy usage [3] and potential equipement wear caused by repeatedly turning PMs on and off [4]. Our main goal in this paper is determine how aggressive PM consolidation should be when one takes into account the combined effects in terms of energy consumptions, performance and equipment wear.

A number of publications devoted to live VM migration and server consolidation have appeared in recent years, including several surveys (e.g., [5]). A large number of consolidation strategies have been proposed with a varied set of criteria. Since server consolidation can be viewed as an optimization problem, approaches used include exact or approximate methods (heuristics) (e.g., [6]), often stressing a specific aspect of resource consumption, e.g., CPU utilization [7] or energy footprint [6].

Existing studies of the performance aspects of live VM migration include experimental benchmarking evaluation (e.g., [8]), simulation studies (e.g., [9]) and analytical performance models (e.g., [10]–[12]). Several performance studies are devoted to the evaluation of the impact of implementation mechanisms for live migration (e.g., [13]). Other research looks at live migration in the context of load balancing (e.g., [10]) or energy-savings techniques to improve the trade-off between energy consumption and application performance (e.g., [14]).

In this paper we use a probabilistic model of VM live migration in the context of PM consolidation for energy savings. Our goal is not to model a specific cloud system. Rather, it is to assess how aggressive VM migration should be when taking into account its combined impact on energy consumption, end-user performance and equipment wear. Our study includes heterogeneous PMs and represents multiple classes of VMs through the use of distributions of VM memory sizes and lifetimes.

Our paper is organized as follows. In the next section we describe in detail the assumptions of our VM migration study. Section III presents the main results of our study. In Section IV we briefly describe the solution approach used to obtain our results. Section V concludes this paper.

II. Study Assumptions

Because VM lifetime and migration on one hand, and task executions on the other hand, generally occur at very different time scales, it is advantageous to consider these sets of events separately. Hence, we adopt a two-level view of the system. At the upper level, as shown in Figure 1, we represent the set of Physical Machines (PMs) and the Virtual Machines (VMs) competing for their use. Our lower level represents individual VMs and the end-user tasks executing on them. We use the term "tasks" to denote the end-user jobs executed by the VMs, the latter running on a set of PMs.

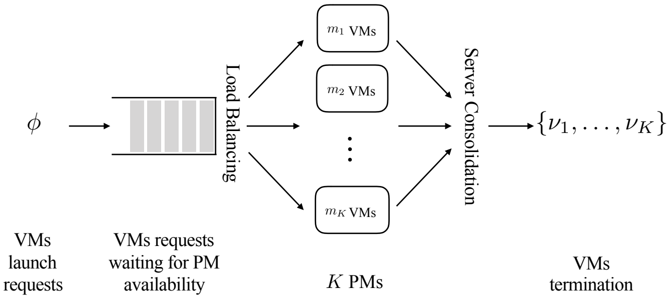


Fig. 1: Overview of the VM creation, destruction, and live migration process with K PMs.

A. VM lifetime and PM consolidation (Upper level)

We let K be the total number of available PMs. We denote by m_i the current number of VMs at PM number i $(i = 1, \ldots, K)$. This number includes VMs in "normal" operation and VMs being migrated to and from the given PM. Differences in the individual PMs may include their memory capacity, execution speed and energy consumption characteristics. Note that we assume that PM memory (and not the number of CPU cores) is the principal limiting resource (cf. [15]). We denote by S_i the available memory capacity of PM number i.

The time between consecutive VM launch requests has a distribution with mean $1/\phi$. Individual VMs may require different amounts of memory and may remain in the system for different time durations. This is represented in our study through a distribution of memory requirements, as well as a distribution of lifetime durations for VMs. We use distributions derived from published measurement data for the times between VM launch requests, memory requirement and VM lifetime [16]. Thus, requests for the launch of a new VM arrive to the system with rate ϕ. They are routed to a specific PM according to the current state of the system, including, of course, the amount of memory available on the PM, which must be sufficient to accommodate the new VM.

If, at some point in time, there are no VMs executing on a PM, the PM becomes inactive and is placed in a low power state. PM load balancing happens at the moment of VM launch when a new VM is initially assigned to a PM. If power savings is a driving consideration, the load balancing algorithm will attempt to assign the new VM to one of the currently active PMs. In this case, an inactive PM is activated only if no active PM has enough resources to accommodate the new VM. In all cases, if no PM has enough resources left, the VM launch request is queued. While the corresponding queue has a finite capacity, for moderate rates of VM launch requests it may be treated as unbounded. We assume that VM launch requests are considered in FCFS order.

As mentioned before, for the lifetime of a VM we use a distribution derived from published data [16]. We let ν_i be the rate with which a VM currently running on PM number i completes its lifetime and leaves the PM. Following such a departure, depending on VM launch request currently queued, the VMs remaining on PM i may be migrated to PM number j. For the migration to happen, the target PM number j must have the capacity to accommodate the remaining VMs from PM i and the migration must be deemed desirable under the migration policy in effect. Note that it may be also possible and advantageous to consolidate the VMs from another PM on PM number i following the departure of a VM from the latter. Clearly, during the migration of a VM there is some impact on task performance due to execution slow-down.

To be clear, in our study, we assume that server consolidation happens only following a departure of a VM from a PM. To control how aggressively the system attempts to consolidate, we use a "trigger" fraction f_t. PM consolidation is attempted after a VM ends its lifetime if available memory on the PM falls below $f_t S_i$, i.e., below the fraction f_t of the memory capacity of the PM on which the VM ended its lifetime. Thus, with f_t = 1, PM consolidation is considered following every VM departure, while with f_t = 0, no VM migration takes place. Additionally, we assume that the effects of migration collisions can be neglected [5]. We denote by $1/\gamma$ the mean time it takes to migrate a single VM.

The goal of this upper-level abstraction is to represent server consolidation and initial load balancing in the interactions between VMs and the PMs on which they execute. Note that the use of distributions for memory requirements and lifetime durations of the VMs represents a simple way to account for multiple classes of VMs in the system.

At this upper level we can estimate quantities related to resource utilization including the probability that at launch a VM is assigned to a given PM, denoted by q_i $(i = 1, \ldots, K)$, the mean number of VMs at a given PM, denoted by $\bar{m}_i$, the mean time between migrations out of an active PM, denoted by $1/\alpha_i$, as well as the probability that when a migration out of PM i does take place its target is PM number j, which we denote by p_{ij}. We can also assess quantities related to energy consumption for each PM including the number of migrations per time unit, $\theta_{\text{mig},i}$, the number of times the PM is switched on and off per time unit, $\theta_{\text{on},i}$ and $\theta_{\text{off},i}$, as well as the fraction of time the given PM is active denoted by $f_{\text{act},i}$, for $i = 1, \ldots, K$.

B. End user task performance (Lower level)

At the lower level, we have individual VMs and the end-user tasks executing on them. Figure 2 shows one such a VM. Tasks arrive at rate $\lambda(n)$ where n denotes the current number of tasks at the given VM. The maximum number of tasks that can be present in the system at any given time is limited to N. There are C virtual CPUs available to execute these tasks. With probability q_i a VM starts its life on PM number i $(i = 1, \ldots, K)$. A VM currently located on PM number i can be in its "normal" state or it can be in "migration" state, i.e., it can be undergoing migration out to another PM. The rate with which a VM in its "normal" state at PM i begins a migration to PM number j is given by $\alpha_i p_{ij}$. The quantities q_i, α_i and p_{ij} are known from the upper level.

The rate with which a migration from PM i to PM number j ends (i.e., the VM returns to its "normal" state) is given by γ (recall that $1/\gamma$ is the mean time to complete a VM

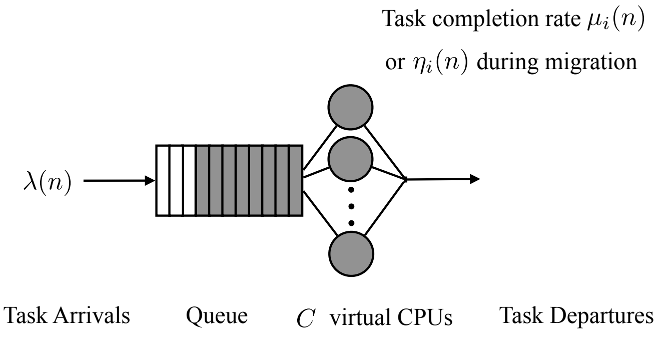

Task completion rate $\mu_i(n)$
or $\eta_i(n)$ during migration
$\lambda(n)$
Task Arrivals
Queue
C virtual CPUs
Task Departures

Fig. 2: End-user tasks at a VM.

migration). We let $\mu_i(n)$ be the rate of task completion for a VM in "normal" state at PM i with a current total of n tasks at the VM. Analogously, when the VM is undergoing migration out of PM number i, its rate of task completion is given by $\eta_i(n)$. Clearly, we will usually have $\eta_i(n) << \mu_i(n)$ (if task execution was to be totally stopped during migration, we would have $\eta_i(n) = 0$). Because of competition for PM resources by the VMs, the rates $\mu_i(n)$ and $\eta_i(n)$ may degrade with the mean number of VMs sharing the given PM, $\bar{m}_i$. The latter quantity is known from the upper level. For simplicity, at the lower level we assume that all times follow memoryless distributions.

The goal at this level is to represent the end-user task execution on a VM and the effect of VM migration on task performance. As a measure of task performance for VMs, we use the mean task response time, denoted by R, and the attained task throughput (the number of tasks processed per time unit), denoted by θ. In some applications, the mean execution time may be the proper measure of attained performance. We denote by X the mean execution time for VM. Note that with proper selection of the arrival rate function $\lambda(n)$ it is possible to represent batch workloads. Obviously, we need to know the task completion rates $\mu(n)$ and $\eta(n)$. These are parameters in our study.

C. Energy consumption model

Since our goal is to assess the combined impact of VM migration on energy consumption and end-user task performance, we need a model of energy consumption in our system. When a PM is fully booted or awoken from sleep or hibernation, we view its energy consumption per time unit as a sum of base energy consumed plus some amount of energy per VM running of the PM as suggested by Beloglazov and Buyya [17]. Thus, the energy consumed by PM number i per time unit in absence of migration, can be expressed as $P_{\text{base},i} f_{\text{act},i} + \bar{m}_i P_{\text{VM},i}$, where $P_{\text{base},i}$ is the average consumption of PM i with no VMs running on it (besides the OS) and $\bar{m}_i$ is the mean number of VMs running on the PM. Additionally, we assume that the migration of a VM requires on average E_{mig} units of energy. Also, we let $E_{\text{on},i}$ and $E_{\text{off},i}$ be the expected energy consumed to turn PM number i on and off, respectively. As mentioned before, we denote by $\theta_{\text{mig},i}$, $\theta_{\text{on},i}$ and $\theta_{\text{off},i}$ the corresponding rates of migration and on and off events. Thus, we can express the energy consumed per time unit by PM number i as

$$\begin{aligned} P_i = {} & P_{\text{base},i} f_{\text{act},i} + \bar{m}_i P_{\text{VM},i} + \theta_{\text{mig},i} E_{\text{mig}} \\ & + \theta_{\text{on},i} E_{\text{on},i} + \theta_{\text{off},i} E_{\text{off},i}, \quad \text{for} \quad i = 1, \ldots, K. \end{aligned} \qquad (1)$$

Note that formula (1) accounts for the energy use due to the activity of PMs and VM migrations [18] but does not include incompressible data center energy consumption (HVAC, lighting, etc). Note also, that in our accounting we assign the energy use due to a VM migration to the PM that hosted the VM being migrated. If desired, different or more elaborate energy usage models [19], [20] can be used.

D. Combined energy-performance-wear criterion

In general, performance and power consumption tend to be antagonistic. Hence, to assess the relative merits of different VM placement and migration policies, we need to look at their energy consumption and performance jointly. Since the absolute energy consumption per time unit P and the mean task response time R (or the mean execution time) have both different units and dynamic ranges, we create corresponding relative measures by scaling quantities to the range $[0, 1]$. For the energy consumption component we use $c_P = \frac{P - P_{\text{lo}}}{P_{\text{hi}} - P_{\text{lo}}}$ and for the mean response time component we use $c_R = \frac{R - R_{\text{lo}}}{R_{\text{hi}} - R_{\text{lo}}}$. P_{lo} corresponds to a value no higher than the lowest value for energy consumption that could be observed for a given PM configuration and workload range, and P_{hi} is no lower than the highest energy consumption under the same conditions. The quantities R_{lo} and R_{hi} are defined in an analogous way for the task response times. In situations where the task execution time is a more appropriate measure, we can redefine c_R as $c_R = \frac{X - X_{\text{lo}}}{X_{\text{hi}} - X_{\text{lo}}}$. X_{hi} and X_{lo} are defined analogously to R_{hi} and R_{lo}.

Besides energy consumption and performance, different PM selection and consolidation policies may have different impact on the reliability of the PMs. In particular, the frequency of switching the PMs on and off may impact their longevity [5]. Let $\omega = \sum_{i=1}^{K} (\theta_{\text{on},i} + \theta_{\text{off},i})$ be the total number of times the PMs are switched on and off per time unit. We use ω as a measure of PM wear and define the corresponding scaled relative measure $c_\omega = \frac{\omega - \omega_{\text{lo}}}{\omega_{\text{hi}} - \omega_{\text{lo}}}$. The quantities ω_{lo} and ω_{hi} are defined analogously to the corresponding quantities for energy consumption and performance so that c_ω remains in the range [0,1].

Our combined energy-performance-wear criterion is a simple weighted combination of the components defined above $C_1 = w_P c_P + w_R c_R + w_\omega c_\omega$ where w_P, w_R and w_ω are the weights assigned to power consumption, performance and equipment wear, respectively. With this criterion, it is easy to emphasize a specific component and the goal will be to seek policies that minimize the criterion chosen.

The next section presents the numerical results of our study, which assesses the combined impact of VM migration on energy consumption, end-user task performance and equipment wear.

Table I summarizes the notation used in this paper.

TABLE I: Principal notation used.

Upper level	
K	Number of available PMs
S_i	Allocatable memory capacity of PM i
ϕ	VM launch rate
$1/\nu_i$	Mean VM lifetime at PM speed of 1
ν_i	Lifetime rate of a VM running on PM number i
$1/\gamma$	Mean time to migrate a single VM
q_i	Probability that at launch a VM is assigned to PM number i
$\bar{m}_i$	Mean number of VMs at PM number i
$1/\alpha_i$	Mean time between migrations out of PM number i
p_{ij}	Probability that a migration out of PM i goes to PM number j
Lower level	
N	Maximum number of tasks in a VM
$\lambda(n)$	Tasks arrival rate at a VM with a current total of n tasks at the VM
C	Number of virtual CPUs available to a VM
$\mu_i(n)$	Rate of task completion for a VM in "normal" state at PM i with a current total of n tasks at the VM
$\eta_i(n)$	Rate of task completion for a VM in "migration" state at PM i with a current total of n tasks at the VM
Energy consumption parameters	
$P_{\text{base},i}$	Average energy consumption of an active PM i with no VMs running on it.
$f_{\text{act},i}$	Fraction of time PM number i is active
$P_{\text{VM},i}$	Additional energy consumption for each new VM on PM number i
$\theta_{\text{mig},i}$	Number of migrations of VMs out of PM number i per time unit
E_{mig}	Energy consumed to migrate a VM
$\theta_{\text{on},i}$	Number of times the PM number i is switched on per time unit
$E_{\text{on},i}$	Energy consumed to turn PM number i on
$\theta_{\text{off},i}$	Number of times the PM number i is switched off per time unit
$E_{\text{off},i}$	Energy consumed to turn PM number i off
P_i	Energy consumed per time unit by PM number i
Performance, energy and wear metrics	
R	Mean task response time
θ	Attained task throughput
X	Mean execution time
P	Energy consumed per time unit by all PMs
ω	Total number of times PM are switched on and off per time unit
c_P	Relative measure of energy
c_R	Relative measure of performance (mean relative response time or mean relative execution time)
c_ω	Relative measure of wear
w_P	Weight assigned to power consumption
w_R	Weight assigned to performance
w_ω	Weight assigned to wear
C_1	Combined energy-performance-wear criterion

TABLE II: PM attributes for Section III-A.

	Group 1	Group 2
Memory capacity (GB)	32	32
Speed factor	1.0	2.0
P_{base} (energy unit)	1.0	3.0
P_{VM} (energy unit)	0.15	0.45
E_{mig} (energy unit per hour)	$1/3 \times 10^{-3}$	$1/3 \times 10^{-3}$
E_{on} (energy unit per hour)	$5 * 10^{-3}$	$1.65 * 10^{-2}$
E_{off} (energy unit per hour)	$5 * 10^{-3}$	$1.65 * 10^{-2}$

III. Study Results

A. How aggressive should the PM consolidation be?

We are now ready to study the effects of live migration in a system with VMs running interactive type of workloads on a set of heterogeneous PMs. Specifically, in our case there are two groups of 4 PMs each ($K = 8$). The PMs in each group differ by their speed and power consumption. The machines in the first group are slower and also more energy thrifty. The machines in the second group run at twice the processor speed of the first group but use three times the energy per time unit. The relevant machine attributes for both groups are summarized in Table II. Note that the energy unit used in our study is the base power consumption of a slower PM (this might be, for instance, 175W as mentioned in [17]).

Based on published measurement values [16], we assume that VM memory requirements are uniformly distributed between 1GB and 8GB, the VM lifetime can be represented as a mixture of exponential and uniform distributions (e.g., [21]), and the time between arrivals of VM launch requests can be represented by a high-variability Weibull distribution. Relevant VM parameters are summarized in Table III.

The parameter values related to energy consumption have been inspired by values published in the literature [18], [22]. Clearly, these values may be expected to vary widely as technology and machine architecture change.

Since our study involves energy savings that can be expected from live VM migration, our initial VM placement strategy avoids waking dormant PMs if a VM launch request can be accommodated by one of active PMs. To represent VM performance degradation due to competition for PM resources, we use a factorf_{deg} which affects the rate of task execution as follows: $\mu_i(n) = \min(n, C)\mu_i\xi_i(\bar{m}_i)$ and $\eta_i(n) = \min(n, C)\eta_i\xi_i(\bar{m}_i)$ where $\xi_i(\bar{m}_i)$ is given by $\xi_i(\bar{m}_i) = 1 - f_{\text{deg}}\bar{m}_i/M_i$. M_i is the maximum number of VMs that can be accommodated on PM number i and C is the number of virtual CPUs.

As discussed in the assumptions of our study, PM consolidation can only happen following the end of life of a VM and we use use a "trigger" fraction f_t to control how aggressively the system attempts to consolidate. Recall that for $f_t = 1$, PM consolidation is considered as often as possible (following every VM departure), while with f_t= 0, there is

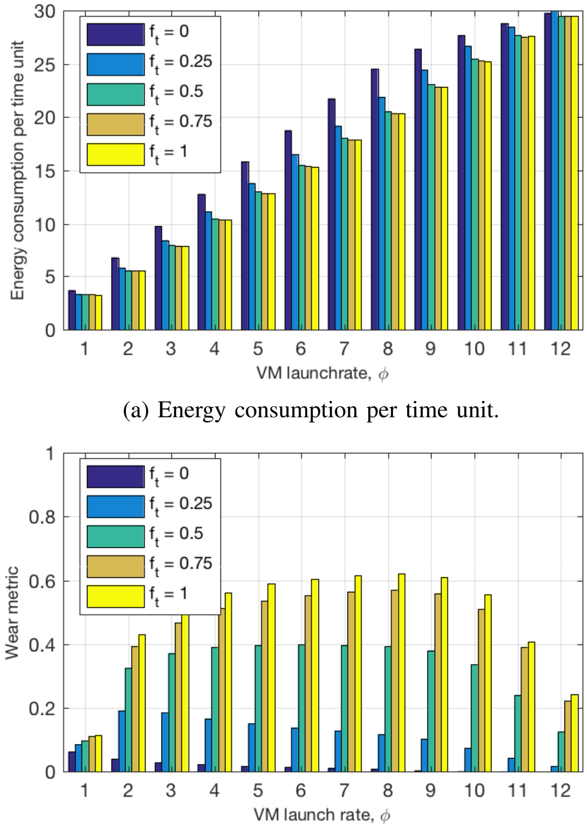
TABLE III: VM attributes for Section III-A.

VM memory requirements	
Uniform distribution between	1 and 8 GB
VM lifetime	
Mixture of exponential and uniform distrib.	
Probability of exponential branch	0.6
Exponential with mean value	1.0 hour
Probability of uniform branch	0.4
Uniform distribution between	2 and 24 hours
VM launch request inter-arrival times	
Weibull distribution	
Scale parameter	0.5
Shape parameter	0.5
Coeff. of variation	2.236
Task related parameters	
$1/\gamma$	6 sec
λ	0.3 sec
C	4
N	12 requests
μ (at processor speed factor of 1.0)	1 per sec
η (at processor speed factor of 1.0)	0.2 per sec

no VM migration. Figure 3a shows the energy consumption per time unit for a set of values of the trigger fraction f_t as a function of the rate of VM launch requests ϕ. We note that the impact of live migration on energy consumption tends to be the highest for moderate rates of VM launch. We also observe that, in our example, a moderately aggressive PM consolidation policy appears sufficient to reap much of the energy savings. Figure 3b shows the "raw" wear component, i.e., the total number of times the PMs are switched on and off per time unit, as a function of VM launch rate ϕ. The aggressiveness of the PM consolidation policy has a major effect on this component for a large range of VM launch rates. With the parameter values used in our study for the mean time to migrate a VM, there is only a limited impact on the mean task response time. Figure 4 illustrates the values of our combined energy-performance-wear criterion C_1 at a moderate VM launch rate of $\phi = 6$ with equal weights for each of the components in our criterion. We observe that, with the parameter values used, among the trigger fraction values considered, the value $f_t = 0.25$ (moderately aggressive PM consolidation) seems to offer a good compromise.

B. Are realistic distributions truly important?

As a last point in our study we look at how important proper workload characterization is at our higher-level abstraction of VM and PM interactions. We consider a system similar to the one described above except that we use three different distributions for the time between arrivals of VM launch requests. The distributions, referred to as Weibull 1, Weibull 2, and hyper-exponential are scaled so that in all cases they result in the same rate of arrivals of VM launch requests.



(a) Energy consumption per time unit.

(b) Wear metric (total number of times the PMs are switched on and off per time unit).

Fig. 3: Energy and wear metrics for varying migration agressiveness and VM launch rates.

TABLE IV: Distributions for the time between arrivals of VM launch requests for Section III-B.

	Weibull 1	Weibull 2	Hyper-exponential
Scale parameter	0.5	0.75	N/A
Shape parameter	0.5	0.665	N/A
Coeff. of variation	2.236	1.553	2.236

Table IV summarizes the parameters of each distribution. The distribution labelled Weibull 1 is the same distribution derived from measurements (cf. [16]) as used in Section III-A. Distributions Weibull 2 and hyper-exponential were designed for our needs. The choice of the hyper-exponential was motivated by the results published by [23].

We note that Weibull 1 and Weibull 2 have different coefficients of variation (and hence different second moments), while Weibull 1 and hyper-exponential have the same first two moments, i.e., their differences come from higher order moments. Here, we keep the trigger fraction at $f_t = 0.5$. Figures 5a, 5b and 5c show the expected task response time, the energy consumption per time unit and the "raw" equipment wear as a function of the rate of arrivals of VM launch requests, ϕ, respectively. Comparing the results for Weibull 1

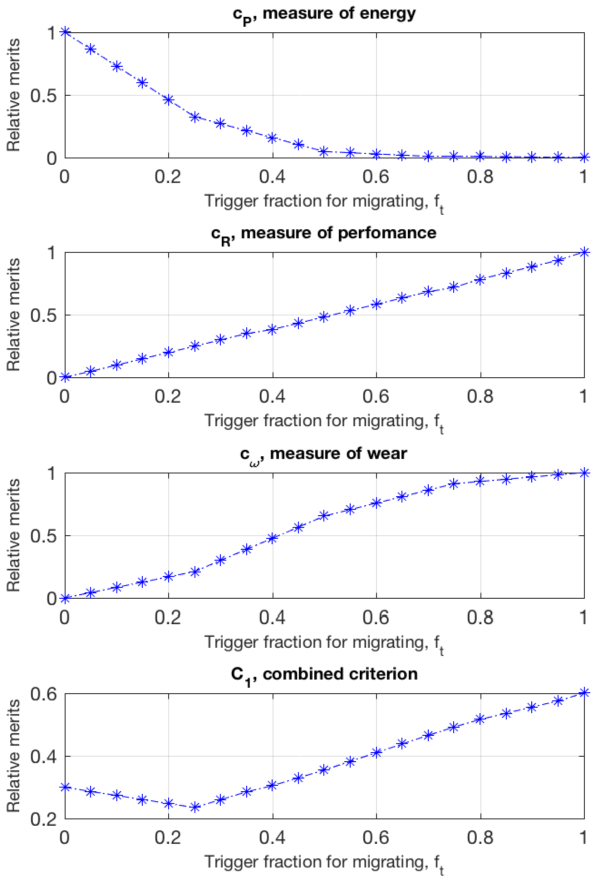
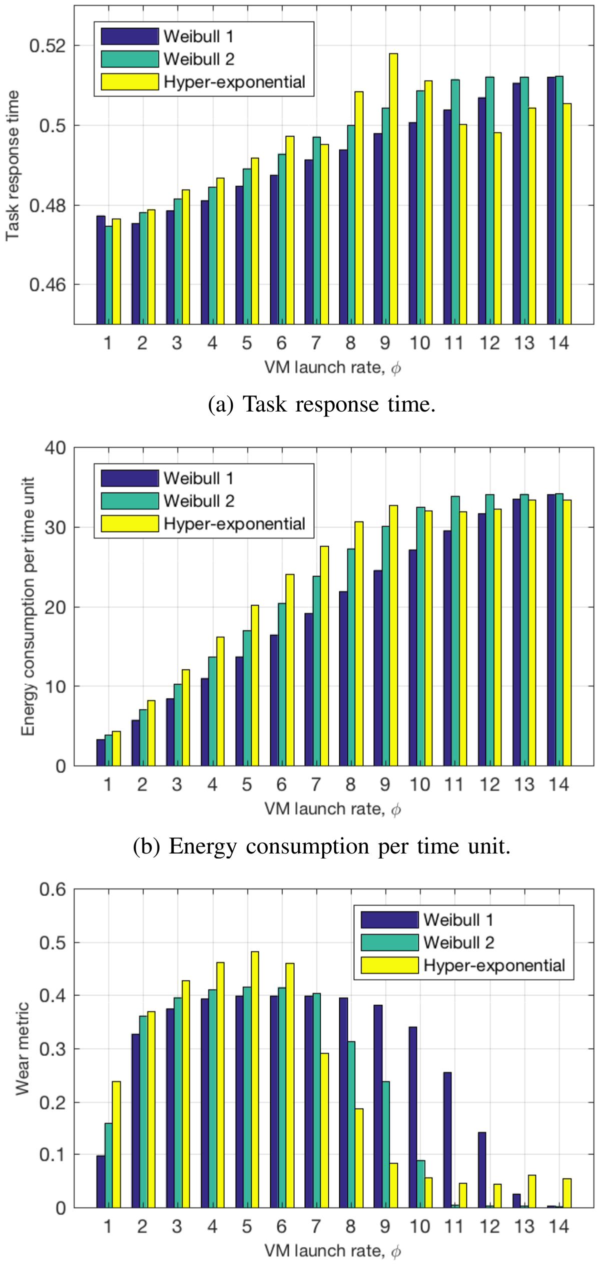


Fig. 4: Effect of migration agressiveness.

and Weibull 2, we notice the influence of the second moments of the distribution. Somewhat surprisingly, depending on the workload levels, higher order moments may be quite important (sometimes even more important than second moments). Additionally, we ran experiments, not shown in this paper, in which we used different VM lifetime distributions, all with the same mean. Here, too, the influence of the second and higher-order moments was non-negligible. This may not be surprising given that our higher level can be viewed as a multi-server queue and such queues are known to be sensitive to higher-order distributional properties. It clearly sounds a note of caution for the use of distributions not inspired by real-life measurements.

IV. Model solution

As mentioned in earlier sections, we adopt a two-level view of the system. Correspondingly, we have two models. At the higher level, a model of the VM life-time cycle. At the lower level, a model of the end-user task performance on a VM, taking into account live migration. We now briefly outline the solution approach used to solve these two models.



(a) Task response time.

(b) Energy consumption per time unit.

(c) Wear metric.

Fig. 5: Influence of the inter-arrivals distribution of VM requests.

A. *Upper-level model solution*

Our upper-level model represents the life cycle of VMs on the set of the PMs. It includes initial load balancing at VM launch as well as VM migrations for PM consolidation. To allow for easy study of a range of PM consolidation and initial VM placement algorithms, discrete-event simulation is by far the most appropriate solution method for this upper-level model.

Since this model does not include explicit representation of task execution, we need to simulate only a small number of events for each VM life cycle. This makes a simulation of even a large number of VM creations and departures quite nimble. In our simulation, written in C, we distinguish the following five events: arrival of a new VM launch request; initial placement of new VM on a PM; start of a VM lifetime; end of a VM lifetime; end of migration for a VM. Other than increasing the length of the simulation appropriately, there are few issues in scaling such a simulation to a large number of PMs.

The simulation produces statistical estimates for the following quantities:

- $\bar{m}_i$: mean number of VMs at PM i;
- q_i: probability that at launch a VM is assigned to PM i;
- $1/\alpha_i$: mean time between migrations out of PM i;
- p_{ij}: probability that a migration out of PM i is directed to PM j;
- $\theta_{\text{mig},i}$: number of migrations per time unit out of PM i;
- $\theta_{\text{on},i}$: number of times PM i is switched on per time unit;
- $\theta_{\text{off},i}$: number of times PM i is switched off per time unit;
- $f_{\text{act},i}$: fraction of time PM i is active.

We use the independent replications method with 7 independent replications each representing 9,000,000 VM life cycles. As a result, confidence intervals at 95% confidence level tend to be very narrow so that we feel one can safely use the middle point estimates.

B. Lower-level model solution

Our lower-level model represents the execution of tasks on a VM where the latter may be subject to migrations. With the distributional assumptions at this level, the state of a VM is defined by the couple (n, i) where n is the current number of tasks at this VM, i refers to the PM on which the VM resides. We use positive values of $i = 1, \ldots, K$ if the VM is in its "normal" execution mode, and negative values $(-i)$ if the VM is being migrated from PM number i to another PM. We denote by $p(n, i)$ the steady state probability, assuming it exists, that the VM is in state (n, i). For $n = 1, \ldots, N-1$ it is a straightforward matter to obtain the balance equations for our model

$$p(n,i)[\lambda(n) + \mu_i(n) + \alpha_i] = p(n-1,i)\lambda(n-1) + p(n+1,i)\mu_i(n+1) + \sum_{j \neq i} p(n,-j)\gamma p_{ji} \quad (2)$$

$$p(n,-i)[\lambda(n) + \eta_i(n) + \gamma] = p(n-1,-i)\lambda(n-1) + p(n+1,-i)\eta_i(n+1) + p(n,i)\alpha_i. \quad (3)$$

The equations for $n = N$ are similar except that there are no terms for $n+1$ and $\lambda(N)$ is replaced by 0.

To account for the finite duration of VM lifetime in our lower-level model, we assume that a VM can only end its life at a moment when there are no user tasks at the VM ($n = 0$) and the VM is in its normal state. To account for the initial VM placement of VMs on PMs, we assume that a new instance of a VM is created following the end of life of a VM and placed on PM number i with probability q_i. Denote by δ_i the rate with which a VM ends its life given that the current system state is $(0, i)$. Assuming the life of a VM ends only when the VM is in its normal state, we must have $\sum_{i=1}^{K} \delta_i p(0,i) = \sum_{i=1}^{K} \nu_i \sum_{n=0}^{N} p(n,i) / \sum_{j=1}^{K} \sum_{n=0}^{N} p(n,j)$. In order to satisfy this relationship, we let

$$\delta_i = \nu_i \sum_{n=0}^{N} p(n,i) / \left(p(0,i) \sum_{j=1}^{K} \sum_{n=0}^{N} p(n,j) \right) \text{for} \quad i = 1, \ldots, K. \quad (4)$$

With this in mind, we can obtain the following balance equations for $i = 1, \ldots, K$

$$p(0,i)[\lambda(0) + \alpha_i + \delta_i(1-q_i)] = p(1,i)\mu_i(1) + \sum_{j \neq i} p(0,-j)\gamma p_{ji} + \sum_{j \neq i} p(0,j)\delta_j q_i \quad (5)$$

$$p(0,-i)[\lambda(0) + \gamma] = p(1,-i)\eta_i + p(0,i)\alpha_i. \quad (6)$$

If solved directly, these balance equations are best solved using an iterative approach [24] where the computation of the values of δ_i using formula (4) is embedded into the iteration.

The above balance equations are well suited for interactive workloads. For a VM running predominantly batch workloads, a better approach is to study the performance with a constant number of jobs $N \geq C$. The goal is to obtain the mean execution time for a job and the job throughput.

Since with a constant number of jobs the system is never empty ($p(n,i) = p(n,-i) = 0$ for $n = 0, \ldots, N-1$), we now assume that a VM may end its life only at the completion of a job when running in normal mode. We denote by τ_i the probability that the VM finishes its life after completing a job in normal mode on PM i. Assuming that $\mu_i(N) \geq \nu_i$, we have $\tau_i = \nu_i / \mu_i(N)$. We then obtain the following balance equations for a VM with a batch workload for $i = 1, \ldots, K$.

$$p(N,i)[\mu_i(N)\tau_i(1-q_i) + \alpha_i] = \sum_{j \neq i} p(N,-j)\gamma p_{ji} + \sum_{j \neq i} p(N,j)\mu_j(N)\tau_j q_i \quad (7)$$

$$p(N,-i)\gamma = p(N,i)\alpha_i. \quad (8)$$

Formulas 7 and 8 together with the normalizing condition $\sum_{i=1}^{K}[p(N,i) + p(N,-i)] = 1$ can be solved using any of a number of methods suited for systems of linear equations.

C. Combined criterion components

Having obtained the steady-state probability distribution $p(n,i)$, we readily derive our performance indices. The overall attained throughput θ can be expressed as

$$\theta = \sum_{i=1}^{K} \sum_{n=0}^{N-1} [p(n,i) + p(n,-i)]\lambda(n). \quad (9)$$

The overall expected response time can then be obtained using Little's formula as

$$R = \frac{\sum_{i=1}^{K} \sum_{n=1}^{N} n[p(n,i) + p(n,-i)]}{\theta}. \qquad (10)$$

Similarly, the overall expected execution time can be obtained as

$$X = \frac{\sum_{i=1}^{K} \sum_{n=1}^{N} \min(n,C)[p(n,i) + p(n,-i)]}{\theta}. \qquad (11)$$

In summary, we solve first the simulation model. Several of the simulation results are then used as input parameters in our lower-level model. Finally, we use the results of both models to compute our combined energy-performance-wear criterion. In particular, we use simulation results to compute the power consumption and the equipment wear components (c_P and c_ω, respectively) of our combined criterion. The performance component (c_R) is obtained from the results of our lower-level model, i.e., from the expected response time R or the expected execution time, as the case may be. As a final point, note that one could use discrete-event simulation to solve our lower-level model. With this solution method we could readily relax the assumptions on memoryless distributions in the end-user task model.

V. Conclusions

In this paper we have presented a a study of the combined impact of server consolidation and live VM migration on energy consumption, end-user task performance and equipment wear. Our hierarchical view comprises at an upper level the VM life cycle on a set of PMs and at a lower level end-user tasks executing on a VM. In our opinion, for the purposes of the evaluation of live VM migration, the relative simplicity of our approach is an advantage compared with more elaborate cloud simulators such as CloudSim [25] and Simgrid [26]. In particular, the effort required to test a new allocation/migration policy, will be much smaller with our approach than with CloudSim or SimGrid

Our numerical studies address the question of how aggressive should the consolidation be. With the parameter values used in our study, a moderate level of VM live migration appears to be the best when taking into account in equal proportions energy saving, task performance and equipment wear. Note that we use s linear model of energy consumption (vs. the number of VMs on a PM). In essence, we assume that, on average, the processor utilization is a linear function of the number of VMs and so is the energy consumption. Polynomials of higher degree (e.g. cube) have been proposed by some authors [27] as fitting better observed energy characteristics. Intuitively, it would seem that such models would result in an even less aggressive PM consolidation (energy use increases more than linearly as the number of VMs on a PM increases). As a final point in our study, we show that it is important to use distributions inspired by measurements as distributional assumptions can matter. While we did use ”realistic” distributions of the time between VM launch requests and of VM lifetime duration, we used memoryless distributions for end-user tasks. We hope to address this shortcoming of our study in a future work.

References

[1] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live migration of virtual machines,” in *NSDI*, 2005.

[2] T. Mastelic and I. Brandic, “Recent trends in energy-efficient cloud computing,” *IEEE Cloud Computing*, 2015.

[3] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J.-M. Pierson, and A. V. Vasilakos, “Cloud computing: Survey on energy efficiency,” *ACM Computing Surveys*, 2014.

[4] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, “Managing server energy and operational costs in hosting centers,” in *ACM SIGMETRICS*, 2005.

[5] A. Varasteh and M. Goudarzi, “Server consolidation techniques in virtualized data centers: A survey,” *IEEE Systems Journal*, 2017.

[6] A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing,” *Future generation computer systems*, 2012.

[7] S. K. Garg, A. N. Toosi, S. K. Gopalaiyengar, and R. Buyya, “SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter,” *Journal of Network and Computer Applications*, 2014.

[8] N. Huber, M. von Quast, M. Hauck, and S. Kounev, “Evaluating and modeling virtualization performance overhead for cloud environments,” in *CLOSER*, 2011.

[9] Y. C. Lee and A. Y. Zomaya, “Energy efficient utilization of resources in cloud computing systems,” *The Journal of Supercomputing*, 2012.

[10] H. Khazaei, J. Misic, and V. B. Misic, “Performance of an IaaS cloud with live migration of virtual machines,” in *IEEE GLOBECOM*, 2013.

[11] F. Machida, D. S. Kim, and K. S. Trivedi, “Modeling and analysis of software rejuvenation in a server virtualized system with live VM migration,” *Performance Evaluation*, 2013.

[12] A. Aldhalaan and D. A. Menascé, “Analytic performance modeling and optimization of live VM migration,” in *EPEW*, 2013.

[13] H. Liu, H. Jin, C.-Z. Xu, and X. Liao, “Performance and energy modeling for live migration of virtual machines,” *Cluster computing*, 2013.

[14] C. Wang, B. Urgaonkar, Q. Wang, and G. Kesidis, “A hierarchical demand response framework for data center power cost optimization under real-world electricity pricing,” in *IEEE Mascots*, 2014.

[15] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S. K. Reinhardt, and T. F. Wenisch, “Disaggregated memory for expansion and sharing in blade servers,” *ACM SIGARCH computer architecture news*, vol. 37, no. 3, pp. 267–278, 2009.

[16] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, “Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms,” in *ACM SOSP*, 2017.

[17] A. Beloglazov and R. Buyya, “Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers.,” *MGC@ Middleware*, 2010.

[18] M. Callau-Zori, L. Samoila, A.-C. Orgerie, and G. Pierre, “An experiment-driven energy consumption model for virtual machine management systems,” *Sustainable Computing: Informatics and Systems*, 2018.

[19] B. Krishnan, H. Amur, A. Gavrilovska, and K. Schwan, “VM power metering: feasibility and challenges,” *ACM SIGMETRICS Performance Evaluation Review*, 2011.

[20] A. E. H. Bohra and V. Chaudhary, “VMeter: Power modelling for virtualized clouds,” in *IEEE IPDPSW*, 2010.

[21] H. Goudarzi and M. Pedram, “Geographical load balancing for online service applications in distributed datacenters,” in *IEEE CLOUD*, 2013.

[22] M. Kurpicz, A.-C. Orgerie, and A. Sobe, “How much does a VM cost? Energy-proportional accounting in VM-based environments,” in *IEEE PDP*, 2016.

[23] R. Wolski and J. Brevik, “Using parametric models to represent private cloud workloads,” *IEEE Transactions on Services Computing*, 2013.

[24] D. M. Young, *Iterative solution of large linear systems*. Elsevier, 2014.

[25] “CloudSim.” http://www.cloudbus.org/cloudsim/.

[26] “SimGrid.” https://simgrid.org/.

[27] X. Zhang, J.-J. Lu, X. Qin, and X.-N. Zhao, “A high-level energy consumption model for heterogeneous data centers,” *Simulation Modelling Practice and Theory*, vol. 39, pp. 41–55, 2013.