An Analytical Model for Electricity-Price-Aware Resource Allocation in Virtualized Data Centers

Giuseppe Faraci, Giovanni Schembra
DIEEI – University of Catania, Catania, Italy
(giuseppe.faraci, schembra)@dieei.unict.it

***Abstract*—Costs associated with the power consumption and cooling requirements of servers in a data center are significant. For this reason, a lot of research efforts in the area of power management have been devoted toward greening data centers and clouds in the last years. This work starts from the observation that the main target of a DC manager is not only to save energy for the environment, but also reduce operational costs. To this purpose, the paper focuses on a single server management, and proposes a joint power management and resource allocation strategy that modulates the number of active VMs on a given server according to both the current workload and the current value of the virtual machine power cost, in order to minimize the total server management cost, while matching the SLA with the customers. An analytical model is proposed to support design of the resource allocation controller parameters.**

***Index Terms*—Cloud Computing; Resource Allocation; QoS; Pricing; Markov Modeling.**

I. Introduction

In the last decade cloud computing has received an enormous success, becoming one of the most important paradigms for computing and service. A common form to host cloud computing is with Data Centers (DC). The main problem of managing a DC is that it consumes huge amounts of energy, contributing to high operational costs and carbon footprints to the environment. Recent studies [1] indicate that the costs associated with the power consumption and cooling requirements of servers over their lifetime are significant. For example, according to Amazon.com's estimates [2], energy-related costs amount to 42% of the total, including both direct power consumption (19%) and the cooling infrastructure (23%).

As a result, a lot of research efforts in both academia and industry in the area of power management have been devoted toward greening DCs and clouds. One of the most common approach exploit virtualization technology for server consolidation and dynamic load distribution and/or load balancing. According to this approach, multiple OS environments are able to coexist on the same physical computer in different Virtual Machines (VM), that can be migrated to facilitate server-consolidation in such a way that idle servers could be turned off (or hibernated) to save energy.

A lot of literature has regarded power reduction in this context, working at three different levels: the server level, the data center level, and the inter-data center level. Most of the focus has been devoted to the second (e.g. through data center right sizing [3]) and third levels (e.g. balancing workload across centers [4]), while at the server level the approach has been mainly limited to power speed scaling [5], by using techniques such as DVFS and low-power P-states [6].

Our work falls at the server level, but is focused on the management of VMs within a given server. This role is played by the DC control architecture [7]. The paper starts from the observation that the main target of a DC manager is not only to save energy for the environment, but also reduce operational costs, so maximizing its revenue while satisfying the Service Level Agreement (SLA) with the DC customers. To this purpose let us notice that, besides workload aspects, other factors, like for example time-variant price of the electricity, have to be considered. More specifically, the whole cost to provide a given service by running a VM on a server of a DC depends at least on the following factors: 1) the power usage of the specific VM [8]; 2) the tariff currently applied by the energy provider; 3) the amount of available energy from low-cost renewable energy sources, if any; 4) the energy dissipated by the air conditioners to cool the server room. For these reasons, in the following we will indicate the whole cost, expressed in price units (PUs), to maintain active a VM in a given time unit as VMPC (VM power cost). As discussed so far, it changes in time according to the above factors.

On the basis of the above observations, the paper focuses on a single server management, and proposes a joint power management and resource allocation strategy that modulates the number of active VMs on a given server according to both the current workload and the current value of the VMPC, in order to minimize the total server management cost, while matching the SLA with the customers. This is pursued by leveraging on the possibility of delaying some service in order to take advantage of the periods when the VMPC is lower. To this purpose the number of active VMs is reduced during periods when the VMPC is high, while is increased in the other periods to serve job requests that have been accumulated in a buffer to wait VM availability. Hence our proposed solution is relevant to many current and future cloud computing scenarios, e.g. search, data

This work has been partially supported by the INPUT project funded by the European Commission under the Horizon 2020 Programme (Call H2020-ICT-2014-1, Grant no. 644672), and by the "PON "Ricerca & Competitività" 2007-2013" within the project "PON04a2_E – SINERGREEN – RES NOVAE – Smart Energy Master per il governo energetico del territorio"..

analytics, social networking, etc. because is very effective for delay tolerant workloads [9], such as massively parallel and data intensive MapReduce jobs.

The paper target is twofold: 1) it proposes an electricity-price-aware Resource Allocation Controller (RAC) that implements an automated policy to decide at runtime when turning on and off VMs on a server of a DC; 2) it proposes an analytical model that supports design of the RAC, minimizing costs due to both the electrical energy consumption and the possible usage of external backup resources in the case of temporary lack of available local resources.

The paper is structured as follows. Section II describes the system we consider in the rest of the paper. Section III introduces the analytical model of the job queueing system and the time-variant set of active VMs. Section IV analytically derives the main performance parameters. Section V illustrates the proposed price-aware strategy, stressing the fact that the analytical model is so general that it can be applied to capture any RAC policy based on VM management. Section VI applies the proposed RAC to a case study and numerically evaluates performance. Finally, Section VII draws some conclusions.

II. System Description

In this paper we consider a DC that receives VM requests from customers in the form of jobs. Each server in the DC hosts VMs of the same type, characterized by a specific configuration of CPU, memory, and storage. The DC operates in a time-slotted fashion.

Specifically we focalize on a generic server that is devoted to manage a given VM configuration. Accordingly, let N_S be the maximum number of VMs that the considered server can host, and μ be the job processing rate for a VM, representing the mean number of jobs that a VM can process in one slot.

Taking into account, as discussed in the previous section, that a running VM causes an amount of electricity power consumption, and that electricity power has a time variant cost, let us indicate the whole time-variant VM power cost (VMPC), expressed in Price Units (PUs), with the process $P(n)$. The goal of the Resource Allocation Controller (RAC) is to orchestrate the on and off switching operations of the VMs according to the current load, the run-time value of $P(n)$ and the required level of quality of service (QoS) defined in the SLA with the customers, expressed as the mean response time, i.e. the time needed to complete the service, starting from the instant when the request arrives to the DC.

Let us now describe the server behavior model we consider in this paper. Arriving jobs are queued when no VM is available in the server. Let *K* be the maximum number of jobs that can be accommodated in the queue. When some active VMs conclude their current job, the RAC extracts new jobs from the queue according to a first-in-first-out (FIFO) policy and runs them on the free VMs. Let us define the RAC decision interval, Δ, as the period between two consecutive events when the RAC monitors the system and decides if turning off or on some of the available VMs.

At each decision interval, the RAC decides the number of VMs that have to work in the next period Δ according to a given decision function $\Gamma_{s_Q}^{(RAC)}(v)$ that depends on the current number s_Q of jobs that are present in the server, i.e. both running on the processor or waiting in the queue. The RAC uses the function $\Gamma_{s_Q}^{(RAC)}(v)$ as follows: at each decision time, if s_Q jobs are present in the server, in the next decision interval a number of v VMs have to be active with a probability $\Gamma_{s_Q}^{(RAC)}(v)$. In Section V we will define a particular function $\Gamma_{s_Q}^{(RAC)}(v)$, but any decision function can be used without requiring any modification of the proposed model.

In order to avoid sudden changes in the number of active VMs, the new number of active VMs, $\hat{v}_{next}$, is decided also using the current number of VMs, v_{curr} , applying the following exponentially weighted moving average (EWMA) filter:

$$\hat{v}_{next} = \gamma \cdot v_{curr} + (1-\gamma)\cdot v \qquad (1)$$

where v is the number of VMs obtained by generating an integer random variable using $\Gamma_{s_Q}^{(RAC)}(v)$, and γ is a parameter belonging to the interval $[0,1]$ that allows the RAC to decide the weight to be given to the past history. Since the value of $\hat{v}_{next}$ calculated as in (1) may not be an integer, the RAC will decide to use either $v_{next} = \lfloor \hat{v}_{next} \rfloor$ or $v_{next} = \lfloor \hat{v}_{next} \rfloor + 1$ with probabilities that are proportional to the distance from $\hat{v}_{next}$, that is:

$$v_{next} = \begin{cases} \lfloor \hat{v}_{next} \rfloor & \text{with prob } p_{\lfloor\ \rfloor} \\ \lfloor \hat{v}_{next} \rfloor + 1 & \text{with prob}: p_{\lfloor\ \rfloor+1} \end{cases} \qquad (2)$$

where:

$$\begin{aligned} p_{\lfloor\ \rfloor} &= \lfloor \hat{v}_{next} \rfloor + 1 - \hat{v}_{next} \\ p_{\lfloor\ \rfloor+1} &= \hat{v}_{next} - \lfloor \hat{v}_{next} \rfloor \end{aligned} \qquad (3)$$

III. System Model

In this section we will model the server behavior when the RAC applies the decision function $\Gamma_{s_Q}^{(RAC)}(v)$. Let us stress that the model is very general and can be easily extended to any RAC policy that considers the same target.

Let us define a discrete-time Markov model, using the slot time as equal to the RAC decision interval, Δ, as defined in the previous section. The model has to capture the time-variant behavior of both the job arrival and the VMPC processes. To this end we will model the above two processes by using a switched batch Bernoulli process (SBBP) [10], the most general Markov modulated process in the discrete-time domain.

An SBBP, $Y(n)$, is a process that is modulated by an underlying Markov chain (uMc). In this way the process $Y(n)$ behaves following a different probability density function (pdf) according to the state of the uMc. Therefore an SBBP $Y(n)$ is fully described by the set $\aleph^{(Y)}$ of values that it can assume, the state space $\Im^{(Y)}$ of its uMc, and the matrix set $\left(P^{(Y)}, B^{(Y)}\right)$, where $P^{(Y)}$ is the transition probability matrix of the uMc, while $B^{(Y)}$ is the matrix whose rows contain the pdfs associated to the chain states. If we indicate the state of the uMc in the generic slot *n* as $S^{(Y)}(n)$, the generic elements of the above matrices are defined as follows:

$$\begin{aligned} P^{(Y)}_{[s'_Y,s''_Y]} &= \text{Prob}\left\{S^{(Y)}(n+1) = s''_Y \middle| S^{(Y)}(n) = s'_Y\right\} \\ B^{(Y)}_{[s'_Y,r]} &= \text{Prob}\left\{Y(n) = r \middle| S^{(Y)}(n) = s'_Y\right\} \\ &\forall s'_Y, s''_Y \in \Im^{(Y)} \text{ and } \forall r \in \aleph^{(Y)} \end{aligned} \tag{4}$$

where $P^{(Y)}_{[s'_Y,s''_Y]}$ represents the probability of transition from s'_Y to the state s''_Y, while $B^{(Y)}_{[s'_Y,r]}$ is the probability that $Y(n)$ assumes the value r when the uMc state is s'_Y.

So we will characterize the *job arrival process* $A(n)$ with the SBBP model described by the matrix set $\left(Q^{(A)}, B^{(A)}\right)$, the state space of the uMc $\Im^{(A)}$, and the set of possible values the process $A(n)$ can assume, $\aleph^{(A)}$. Likewise, $\left(Q^{(P)}, B^{(P)}\right)$, $\Im^{(P)}$, and $\aleph^{(P)}$ will characterize the *VMPC process* $P(n)$.

Now let us define the state of the job queueing system of the server at the generic slot *n* with the 4-dimensional Markov process $S^{(\Sigma)}(n) = \left(S^{(S)}(n), S^{(Q)}(n), S^{(A)}(n), S^{(P)}(n)\right)$, where:

- $S^{(S)}(n) \in [0, N_S]$ represents the number of active VMs;
- $S^{(Q)}(n) \in [0, K + N_S]$ represents the population in the server at the slot *n*, that is the total number of jobs waiting in the queue or running on the active VMs;
- $S^{(A)}(n) \in \Im^{(A)}$ represents the state of the uMc of the job arrival process $A(n)$;
- $S^{(P)}(n) \in \Im^{(P)}$ represents the state of the uMc of the VMPC process $P(n)$.

Let us now consider two generic states: $s_{\Sigma 1} = \left(s_{S1}, s_{Q1}, s_{A1}, s_{P1}\right)$ in the slot *n*, and $s_{\Sigma 2} = \left(s_{S2}, s_{Q2}, s_{A2}, s_{P2}\right)$ in the slot $n+1$. We assume the following event sequence during each slot:

1. the VMPC and the arrival processes change their values at the beginning of the slot;
2. some new jobs enter the system to be served; if the active VMs are not sufficient, some of them are enqueued;
3. the jobs that have finished their work leave the server, so freeing some VMs;
4. the server population variable is updated according to the number of arrived and served jobs;
5. at the end of the slot, the RAC decides how many VMs have to be active in the next slot according to the current values of population and the current VMPC;
6. Finally, the system state variables are observed and the Markov processes updated.

Now we can define the generic element of the state transition probability matrix as follows:

$$\begin{aligned} P^{(\Sigma)}_{[s_{\Sigma 1},s_{\Sigma 2}]} &= \text{Prob}\left\{S^{(\Sigma)}(n+1) = s_{\Sigma 2} \middle| S^{(\Sigma)}(n) = s_{\Sigma 1}\right\} = \\ &= P^{(S)}_{[s_{S1},s_{S2}]}\left(s_{Q2}, s_{A2}, s_{P2}\right) \cdot P^{(Q)}_{[s_{Q1},s_{Q2}]}\left(s_{S1}, s_{A2}\right) \cdot P^{(A)}_{[s_{A1},s_{A2}]} \cdot P^{(P)}_{[s_{P1},s_{P2}]} \end{aligned} \tag{5}$$

where:

- $P^{(S)}\left(s_{Q2}, s_{A2}, s_{P2}\right)$ is the transition probability matrix of the number of active VMs. In order to derive its generic element, let us consider that the RAC decides the number $\hat{\nu}_{next}$ of active VMs, first drawing a value ν by using the decision function $\Gamma^{(RAO)}_{s_Q}(\nu)$, then weighting the past history with the parameter γ according to (1), and finally, as in (2), rounding the result to one of the closest integer values with the probabilities in (3). So, since the value ν is drawn with probability $\Gamma^{(RAO)}_{s_Q}(\nu)$, and the term $\gamma \cdot s_{S1} + (1-\gamma) \cdot \nu$ is then rounded to the closer integer values with probabilities $p_{\lfloor\ \rfloor}$ and $p_{\lfloor\ \rfloor+1}$, we have:

$$P^{(S)}_{[s_{S1},s_{S2}]}\left(s_{Q2}, s_{A2}, s_{P2}\right) = \sum_{\nu=0}^{N_S} \left\{\varsigma\left(s_{S1}, s_{S2}, \nu\right) \cdot \Gamma^{(RAC)}_{s_{Q2}}(\nu)\right\} \tag{6}$$

where

$$\begin{aligned} &\varsigma\left(s_{S1}, s_{S2}, \nu\right) = \\ &= \begin{cases} p_{\lfloor\ \rfloor} & \text{if } \nu : \lfloor \gamma \cdot s_{S1} + (1-\gamma) \cdot \nu \rfloor = s_{S2} \\ p_{\lfloor\ \rfloor+1} & \text{if } \nu : \lfloor \gamma \cdot s_{S1} + (1-\gamma) \cdot \nu \rfloor + 1 = s_{S2} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{7}$$

- $P^{(Q)}\left(s_{S1}, s_{A2}\right)$ is the transition probability matrix of the server population, that depends on both the number of active VMs and the state of the arrival process. If we indicate the number of jobs that leave the server with $\sigma \in [0, s_{S1}]$, and the number of job arrivals as ρ, with $\rho \in \aleph^{(A)}$, the transition probability for the server population state from s_{Q1} to s_{Q2} can be derived as follows:

$$\begin{aligned} &P^{(Q)}_{[s_{Q1},s_{Q2}]}\left(s_{S1}, s_{A2}\right) = \\ &= \sum_{\rho \in \aleph^{(A)}} \sum_{\sigma=0}^{s_{S2}} B^{(A)}_{[s_{A2},\rho]} \cdot p^{(SERV)}_{s_{S1}}\left(\sigma, s_{Q1}\right) \cdot I_{s_{Q1},s_{Q2}}(\rho, \sigma) \end{aligned} \tag{8}$$

where:

  - $I_{s_{Q1},s_{Q2}}(\rho, \sigma)$ is the Boolean function indicating that the server population state s_{Q2} is reachable from s_{Q1} when ρ arrivals occurred in the generic slot, and σ jobs have been served in the same slot. According to the Lindley equation, we have:

$$\begin{aligned} &I_{s_{Q1},s_{Q2}}(\rho, \sigma) = \\ &= \begin{cases} 1 & \text{if } \min\left[\max\left(s_{Q1} + \rho, K + s_{S1}\right) - \sigma, 0\right] = s_{Q2} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{9}$$

  - $B^{(A)}_{[s_{A2},\rho]}$ is the probability that ρ arrivals occur when the underlying Markov chain of the arrival process is in the state s_{A2};
  - $p^{(SERV)}_{s_{S1}}\left(\sigma, s_{Q1}\right)$ is the probability that σ jobs are served in one slot when s_{S1} VMs are active and s_{Q1} jobs are in the server.

In order to calculate $p_{s_{S1}}^{(SERV)}(\sigma, s_{Q1})$, first let us observe that the number of working VMs in a generic slot is given by the minimum between the number of active VMs, s_{S1}, and the whole number of jobs in the server, s_{Q1}. Therefore, the expected number of jobs served by the whole server in the current slot is:

$$\omega = \mu \cdot \min(s_{Q1}, s_{S1}) \qquad (10)$$

where μ is the mean number of jobs that a VM can serve in one slot. Since ω may not be an integer, similarly to what has been done so far, we consider that the server serves either $\lfloor \omega \rfloor$ or $\lfloor \omega \rfloor + 1$ jobs with probabilities $\psi_{\lfloor\ \rfloor}$ and $\psi_{\lfloor\ \rfloor+1}$, respectively. Therefore, we have:

$$p_{s_{S1}}^{(SERV)}(\sigma, s_{Q1}) = \begin{cases} \psi_{\lfloor\ \rfloor} & \text{if } \lfloor \omega \rfloor = \sigma \\ \psi_{\lfloor\ \rfloor+1} & \text{if } \lfloor \omega \rfloor + 1 = \sigma \\ 0 & \text{otherwise} \end{cases} \qquad (11)$$

- $P^{(A)}$ and $P^{(P)}$ are the transition probability matrices of the underlying Markov chains of the arrival and the VMPC processes, as defined so far; they are model inputs.

Now, from the matrix $P^{(\Sigma)}$ defined in (5) we can derive the system steady-state probability array $\underline{\pi}^{(\Sigma)}$ by solving the linear equation system $\underline{\pi}^{(\Sigma)} P^{(\Sigma)} = \underline{\pi}^{(\Sigma)}$.

IV. Performance Parameter Derivation

In this section we derive the main QoS system parameters by using the model proposed in the previous section.

The mean response time, defined as the mean value of the total time spent by the generic job in the server, can be derived by applying the Little theorem:

$$E\{T_\Sigma\} = \left[\sum_{s_S=0}^{N_S} \sum_{s_Q=0}^{K+s_S} \sum_{s_P \in \Im^{(P)}} \sum_{s_A \in \Im^{(A)}} s_Q \cdot \pi^{(\Sigma)}_{[s_S, s_Q, s_P, s_A]} \right] \Big/ \bar{J} \qquad (12)$$

where the numerator is the mean number of jobs that are present in the whole server, while the denominator is the mean job arrival rate, that can be easily derived from the SBBP model of the arrival process as follows:

$$\bar{J} = \sum_{r \in \aleph^{(A)}} \sum_{s_A \in \Im^{(A)}} r \cdot B^{(A)}_{[s_A, r]} \cdot \pi^{(A)}_{[s_A]} \qquad (13)$$

Another important parameter that characterizes the system behavior is the mean number of active VMs. It can be calculated as follows:

$$E\{VM\} = \sum_{s_S=0}^{N_S} \sum_{s_Q=0}^{K+s_S} \sum_{s_P \in \Im^{(P)}} \sum_{s_A \in \Im^{(A)}} s_S \cdot \pi^{(\Sigma)}_{[s_S, s_Q, s_P, s_A]} \qquad (14)$$

Now, let us calculate the rejection probability of a job, defined as the probability that a job cannot be enqueued due to buffer overflow, and so is redirected to a backup resource to be served with a higher cost:

$$P_{rej} = \bar{L} / \bar{J} \qquad (15)$$

where $\bar{L}$ is the mean value of rejected jobs per slot. It can be obtained taking into account that, if s_Q jobs are in the server and the server is working with s_S VMs, the system can accommodate $[(K + s_S) - s_Q]$ jobs only: Therefore, if r new jobs arrive to the server, $\ell(r, s_Q, s_S) = r - [(K + s_S) - s_Q]$ are rejected. So, we have:

$$\bar{L} = \sum_{s_S=0}^{N_S} \sum_{s_Q=0}^{K+s_S} \sum_{\substack{s_P \in \Im^{(P)} \\ s_A \in \Im^{(A)}}} \sum_{r=K+s_S-s_Q+1}^{r_{MAX}^{(A)}} \ell(r, s_Q, s_S) \cdot B^{(A)}_{[s_A, r]} \cdot \pi^{(\Sigma)}_{[s_S, s_Q, s_P, s_A]} \qquad (16)$$

Another important QoS parameter is the mean cost due to the electrical energy consumed by the active VMs. It can be calculated as follows:

$$E\{C_{\text{Prim}}\} = \sum_{s_S=0}^{N_S} \sum_{s_Q=0}^{K+s_S} \sum_{s_P \in \Im^{(P)}} \sum_{s_A \in \Im^{(A)}} \sum_{r_P \in \aleph^{(P)}} r_P \cdot s_S \cdot B^{(P)}_{[s_P, r_P]} \cdot \pi^{(\Sigma)}_{[s_Q, s_S, s_P, s_A]} \qquad (17)$$

where r_P is the generic value assumed by the VMPC process (so $(r_P \cdot s_S)$ is the generic cost to maintain s_S VMs active), and $B^{(P)}_{[s_P, r_P]}$ represents the probability that the VMPC value is r_P when the state of the underlying Markov chain of the price process is s_P.

Finally, let us calculate the total management cost, taking into account that all the jobs rejected by the considered server are forwarded to a more expensive backup server (for example belonging to a remote public cloud). If we indicate the price applied by the backup server to serve a job as $\wp_{BKP}$, we have:

$$E\{C_{\text{TOT}}\} = E\{C_{\text{Prim}}\} + E\{C_{\text{BKP}}\} \qquad (18)$$

where $E\{C_{\text{BKP}}\}$ is the mean cost of the backup resources:

$$E\{C_{\text{BKP}}\} = \wp_{BKP} \cdot P_{Loss} \cdot \bar{J} \qquad (19)$$

V. Price-Aware Decision Policy

In this section we will propose a decision policy $\Gamma_{s_Q}^{(RAC)}(\nu) = f_{s_Q}^{(A,\alpha,\beta)}(\nu)$, where $f_{s_Q}^{(A,\alpha,\beta)}(\nu)$ is a function that depends on a set of parameters A, α and β, with $A \in [0,1]$, $\alpha \in [0, N_S]$, and $\beta \in [0,1]$, that will be determined in the folowing. It has the target of deciding the number of VMs in such a way that queue overflow and VMs underutilization are minimized, so minimizing the total server management cost.

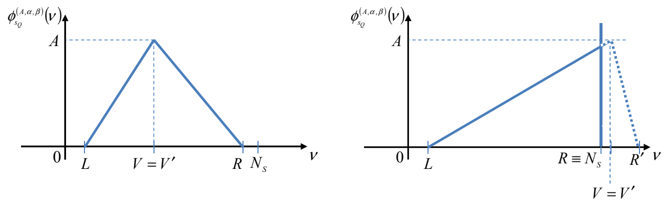
The set of parameters A, α and β is varied in time according to the VMPC with the aim of using more VMs when the VMPC is lower. The function $f_{s_Q}^{(A,\alpha,\beta)}(\nu)$ has to have the following properties:

- the sum of its elements has to be equal to 1 for each number of jobs present in the server, that is:

$$\sum_{\nu=0}^{N_S} f_{s_Q}^{(A,\alpha,\beta)}(\nu) = 1 \quad \forall s_Q \in [0, K + N_S] \qquad (20)$$

- it has to present a maximum for the number of active VMs that is considered the most appropriate. It has to depend on the current value of jobs in the server, s_Q.

According to the above considerations, we derive this function with a triangular shape. As we will see, for each value of s_Q it is exhaustively determined for each set of parameters A, α and


$\phi_{s_Q}^{(A,\alpha,\beta)}(v)$
A
0
L
$V=V'$
R
N_S
v
$\phi_{s_Q}^{(A,\alpha,\beta)}(v)$
A
0
L
$R \equiv N_S$
R'
v
$V=V'$

(a) $V' < N_S$ (b) $V' \geq N_S$

Fig. 1: Triangular shaped decision function

TABLE I. SYSTEM CONFIGURATION PARAMETERS

	$\overline{T}_{req} = 2$ slot				$\overline{T}_{req} =$ 6 slot			
Periods	A	α	β	γ	A	α	β	γ
Low price	0.204	1.969	0.650	0.195	0.127	0.959	0.934	0.182
High price	0.488	0.788	0.563		0.400	0.215	0.853	

β. The first step is to calculate the temporary function $\phi_{s_Q}^{(A,\alpha,\beta)}(v)$ depicted in Fig. 1 as follows:

- the abscissa of the maximum value, representing the most appropriate number of active VMs, is indicated as V, and is calculated proportionally to the value of s_Q by using α as the proportionality constant. In order to avoid that V is greater than the maximum number of available VMs, we derive:

$$V = \min\{V', N_S\}\ ,\ \text{ where } V' = \text{round}\{\alpha \cdot s_Q\} \tag{21}$$

- the ordinate of the maximum value is given by A;
- the abscissa of the left vertex of the triangle, here indicated as L, is determined as a fraction of V', by using the parameter β :

$$L = \min\{\text{round}\{\beta \cdot V'\}, V-1\} \tag{22}$$

- the abscissa of the right vertex of the triangle, here indicated as R, is determined such that the condition in (20) is respected, i.e.:

$$R = \min\{R', N_S\}\ ,\quad \text{where } R' = \lceil 2/A + L \rceil \tag{23}$$

Finally, in order to satisfy (20), we calculate $f_{s_Q}^{(A,\alpha,\beta)}(v)$ from $\phi_{s_Q}^{(A,\alpha,\beta)}(v)$ by normalizing it with its sum.

Let us note that, as it is easy to argue, the values of the parameters A, α and β for each value of VMPC, as well as the value of γ, strongly influence the server performance. For this reason in Section VI we will set an optimization problem that allows us to calculate the best set of parameters for each VMPC that minimizes the server management cost while respecting the SLA with the customers.

VI. NUMERICAL RESULTS

In this section we derive some numerical results by applying the proposed analytical model to a case study. We focus on a specific server of a DC that runs VMs to provide the same service to all the jobs loading the server. Let the mean service duration of a job be equal to one slot, and the maximum number of VMs that can run on that server simultaneously equal to $N_S = 30$. Moreover, we assume that the maximum queue length of jobs waiting to be served by the considered server is $K = 60$.

As regards the time variability of the VMPC, we assume that it can have two values, i.e. 1 PU, and 3 PUs, characterizing low-price and high-price periods, both with a mean duration of 6 slots. Therefore, assuming that an active VM causes a power consumption of 23.3 W, the SBBP modeling the VMPC is given by: $\Im^{(P)} = \{low\, price, high\, price\}$, $\aleph^{(P)} = \{23.3, 69.9\}$PU, and

$$Q^{(P)} = \begin{bmatrix} 5/6 & 1/6 \\ 1/6 & 5/6 \end{bmatrix}, \qquad B^{(P)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{24}$$

Let $\wp_{BKP} = 100$ PUs be the cost to run a VM on an external backup server.

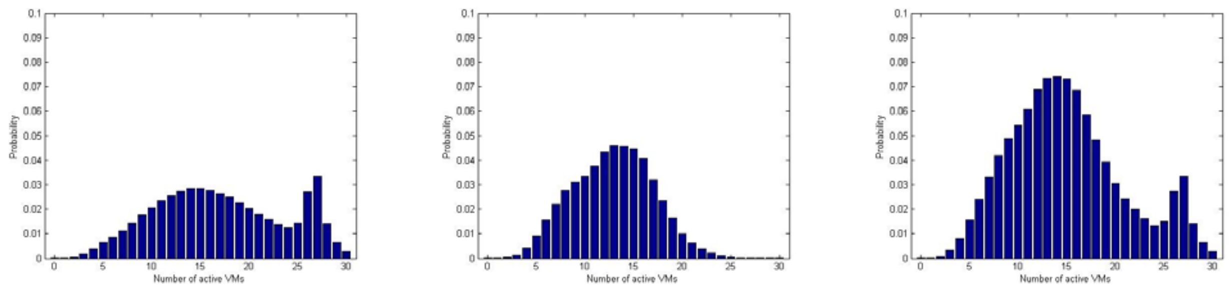
As far as the job arrival process is concerned, as in [9] we consider a Poisson process truncated in the range [0,30], i.e. $\aleph^{(A)} = [0,30]$, with a mean value of 15 arrivals per slot. Finally, we assume that the SLA with the customers is on the mean response time, $\overline{T}_{req}$. In order to analyze the impact of the delay tolerance expressed by customers by the SLA on the management cost, we compare the management cost for the two cases of the SLA parameter $\overline{T}_{req}$ equal to 2 and 6 slots.

In order to choose the best set of parameters A, α, and β that characterize the decision function $\Gamma_{s_Q}^{(RAC)}(v)$ for each of the two VMPC periods (low and high price), and the parameter γ that characterizes the EWMA filter, we have used a genetic algorithm, obtaining the parameter set listed in Table I.

Figs. 2 and 3 show the pdf of the number of active VMs for the two considered target values of $\overline{T}_{req}$, also analyzing the periods of low and high VMPC, separately. As expected, we can notice how, thanks to the application of the proposed price-aware resource allocation policy, the VMs are more likely active when the VMPC is low. The peaks that are present on the right of Figs. 2a and 3a, and consequently in Figs. 2c and 3c, are due to the exigency of using a higher number of active VMs immediately after the VMPC changes from high to low until the short-term steady-state is not reached. Moreover, comparing plots in Fig. 2 with plots in Fig. 3 we observe that a looser bound for the mean response time (i.e. $\overline{T}_{req} = 6$) allows the use of a less number of active VMs during high-price periods (see Figs. 3b vs. 2b), so a management cost reduction.

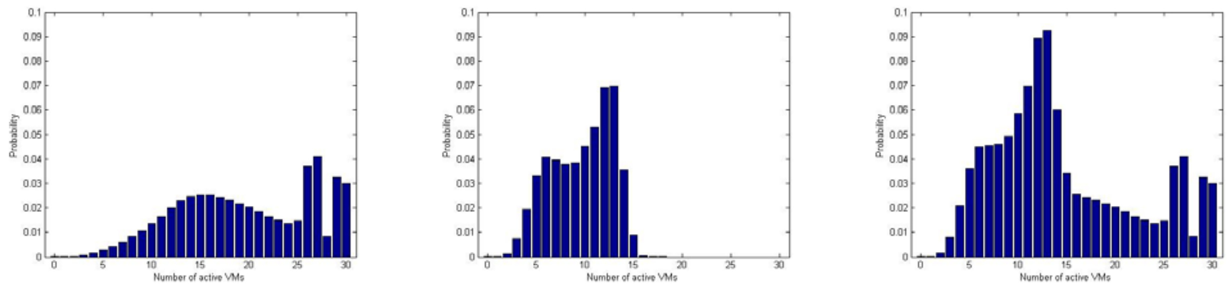
Finally, in order to analyze the gain obtained with the price-aware policy on the management cost, in Table II we have showed the mean number of VMs and the management cost indices comparing the case the RAC uses the proposed allocation strategy with the following two cases: 1) *VMPC unaware policy*, when the allocation strategy does not take care of the VMPC variability; 2) *VM always on*, when all the VMs are maintained active, independently of the system population.

First we can observe that, when the proposed strategy is applied, the mean number of active VMs, $E\{VM\}$, is higher when the SLA is more stringent. Instead, for the VMPC unaware



(a) Low-price periods (b) High-price periods (c) Infinite horizon

Fig. 2: Pdf of the number of active VMs for a required mean response time of *2 slot*



(a) Low-price periods (b) High-price periods (c) Infinite horizon

Fig. 3: Pdf of the number of active VMs for a required mean response time of *6 slot*

	Proposed		**VMPC unaware**		**VM always on**	
	$\overline{T}_{req}=2$	$\overline{T}_{req}=6$	$\overline{T}_{req}=2$	$\overline{T}_{req}=6$	$\overline{T}_{req}=2$	$\overline{T}_{req}=6$
$E\{VM\}$	15.03	14.62	15.00	15.000	30.00	
$E\{C_{\text{TOT}}\}$	651.06	600.94	698.99	698.99	1398.00	
$E\{C_{\text{Prim}}\}$	651.06	567.98	698.99	698.99	1398.00	
$E\{C_{\text{bck}}\}$	0.00	32.97	0.00	0.00	0.00	

strategy the mean number of active VMs is equal to the mean load, i.e. 15 arrivals/slot. This means that the queue works with a unitary utilization coefficient, due to the fact that the decision probability function is able to avoid both losses and VM inactivity periods. The last column of Table II represents the worst case in terms of energy consumption, i.e. when all the 30 VMs are maintained on for any state of the server population. The second row of the same table demonstrates the strength of the proposed strategy, saying that we achieve a power saving gain greater than 53% for $\overline{T}_{req}=2$, reaching 57% for $\overline{T}_{req}=6$, as compared with the "VM always on" strategy. Moreover, analyzing the impact of the price awareness, we can notice that the proposed policy has a power saving gain in the two SLA cases of about 7% and 14% over the VMPC unaware policy. The last two rows present the composition of the whole management cost, and explain the way in which the proposed strategy achieves the maximum gain. It is evident that the gain is realized by introducing some risk of not finding available primary resources, but this is strongly compensated by the obtained high power saving.

VII. Conclusions

In this paper we propose an analytical model to support the design of price-aware management policies of servers in virtualized data centers. A management policy is then proposed and numerically analyzed with the proposed model in a case study, demonstrating that it achieves great power saving in respect to a similar price-unaware policy and the worst-case policy when all the VMs are maintained on, whatever the server load. It has been also evaluated how the power saving gain increases for delay-tolerant jobs, that is jobs that have no stringent requirements on the mean response time.

References

[1] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Computer Comm. Review*, vol. 39, no. 1, Jan. 2009.

[2] Hamilton, J., "Cooperative Expendable Micro-Slice Servers (CEMS): Low Cost, Low Power Servers for Internet-Scale Services," in *Proc. CIDR 2009*, Asilomar, CA, USA, Jan. 2009.

[3] M. Li net *al.*, "Dynamic right-sizing for power-proportional data centers," in Proc. Of *IEEE Infocom* 2011, Shangai, China, 2011.

[4] Z. Liu, A. Wierman, S. Low, and L. Andrew, "Greening geographical load balancing," in *ACM SIGMETRICS*, 2011.

[5] A. Wierman, L. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in Proc. of *IEEE Infocom* 2009, Rio de Janeiro, Brazil, 2009.

[6] R. Bruschi, P. Lago, A. Lombardo, G. Schembra, "Modeling Power Management in Networked Devices," Computer Communications, Elsevier, Vol. 50, no. 1, September 2014.

[7] R. Urgaonkar, U. Kozat , K. Igarashi and M. Neely , "Dynamic Resource Allocation and Power Management in Virtualized Data Centers," Proc. IEEE/IFIP NOMS, pp.479 -486 2010.

[8] R. Shea, *et al.*, "Power Consumption of Virtual Machines with Network Transactions: Measurement and Improvement," in Proc. of IEEE *Infocom* 2014, Toronto, Canada, Apr. 2014.

[9] Y. Yao *et al.* "Data Centers Power Reduction: A Two Time Scale Approach for Delay Tolerant Workloads,", IEEE Infocom, 2012.

[10] A. La Corte *et al.,* "An Analytical Paradigm to Calculate Multiplexer Performance in an ATM Multimedia Environment," Computer Networks, Vol. 29, No. 16, Dec. 1997.