Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 337 (2018) 5–22

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Power Management Policy for Heterogeneous Data Center Based on Histogram and Discrete-Time MDP

Marziyeh Bayati[2]

LACL
Université Paris-Est Créteil (UPEC)
France

Abstract

This work presents a stochastic model for Dynamic Power Management (DPM) that is based on switching-on/off machines in a data center of heterogeneous servers. The aim of a DPM is to ensure both a reasonable energy consumption and an acceptable Quality of Services (QoS). In this paper, arrival jobs and service rates are specified with histograms which are discrete distributions obtained from real traces, empirical data, or incoming traffic measurements. A data center is modeled by a queue, then we formulate the optimization problem by a discrete time Markov Decision Process (MDP) to find the optimal policy. We prove that the optimal policy is not hysteretic. Our approach was applied and tested for several system parameters over real Google traffic traces, when we performed a comparison between homogeneous and heterogeneous data centers.

Keywords: Markov Decision Process, Stochastic Modeling, Energy-Aware optimization, Heterogeneous Data Centers, Digital Pollution, Queuing System.

1 Introduction

Several studies as in [23,10,17,5] show a significant augmentation of energy consumption and digital pollution caused by recent expansion of *Clouds* and *Data Centers*. More than 1.3% of the global energy consumption is due to the electricity used by data centers. Additionally, one data center server can produce more than 10 kilogram of CO_2 per day, rates that are increasing, revealed by surveys conducted in [23,40], saying a lot about the increasing evolution of data centers. Thus, needs for energy saving is emerging to consider a power management strategy face to the energetic problems and digital pollution issues. Data centers are designed to support the expected peak traffic load, however the global load is about 60% of the

[1] Special thanks to N. Pekergin.

[2] Email:mbayati@hotmail.fr

https://doi.org/10.1016/j.entcs.2018.03.031

peak load [10]. In fact, an important number of servers are not under load and still consume about 65%-70% of the maximal energy consumption [17,5].

1.1 Related works

Studies like [26,3] show that much of the energy consumed in the data center is mainly due to the electricity used to run the servers and to cool them (70% of the total cost of the data center). Thus, the main factor of this energy consumption is related to the number of operational servers. An important effort was focused on servers and their cooling. Efforts have been made to build low-energy-consumption processors and better components [18], more efficient cooling systems [37], optimized kernels [20], and more efficient energy network [10]. Against this background, complementary saving energy approach is to consider a power policy to manage the switching-on/off of servers in a data center to ensure both a better quality of services offered by these data centers and reasonable energy consumption.

Two requirements are in conflict:

(i) Maintaining a high Quality of the Service (QoS).

(ii) Consuming less energy.

For the first requirement, we need to turn-on a large number of servers which consumes more energy and leads to less waiting time and decreases the rate of losing jobs but needs a high energy consumption. For the second requirement, we need to turn-on a small number of servers which leads to less energy consumption, but causes more waiting time and increases the rate of losing jobs. Thus, the goal is to design better power management algorithms which take into account these two constraints to minimize waiting time, loss rate and energy consumption.

According to research published by Dell and Principled Technologies, a single server consumes around something between 384 and 455 Watts. Other works evaluate that the power consumption of each server ranges between 238 and 376 Watts [34]. Rajesh et al. [40] estimate the cost of one kWh of energy to 0.0897\$. These values may vary depending on where the data center is located and how electricity is generated. Using that baseline, one server costs around 300\$ per year to run. Otherwise, every job may generates a profit, and the average profit per job can be computed as a ratio of the total profit over the number of served jobs. For instance, 10^6 requests (page views) may bring 1000\$ of revenue. Thus, it can be said that each job brings 10^{-3}\$ on average. Work in [12] suggests that each successfully processed job [3] generates a profit around 6.2×10^{-6}\$. In this case, a lost job costs 6.2×10^{-6}\$.

The heterogeneity of a data center servers appears from the fact that machines are gradually upgraded, provisioned, and replaced during the lifetime of a data center [4,24]. Several studies [32,36] show that in average a data center may host from three to five machine generations with different configurations per generation in terms of speeds and sizes of: memory, processor, storage, and network. Indeed,

[3] Each job is a request for a web page view.

it is natural to have several hardware machine configurations inside a data center. Thus, ignoring this heterogeneity may reduce significantly the data center efficiency. Our analysis of the Google trace servers [47,41] shows a non negligible level of server heterogeneity. Mainly, we observed three levels of heterogeneity that are based on two criteria: CPU-speed and memory-size of each machine. Those criteria are good indicators for QoS and also energy consumption. Figure 1 shows that a very important number of servers run with a medium CPU-speed and a medium amount of memory.

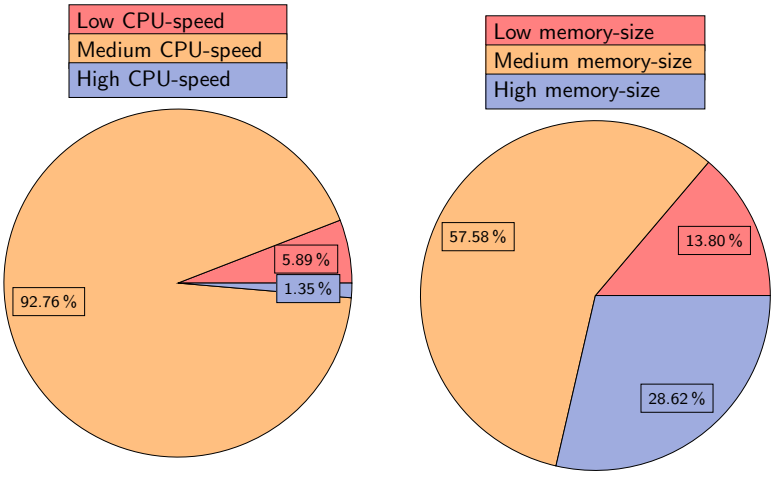


Fig. 1. Servers heterogeneity: Google trace servers repartition according to CPU-speed and memory-size.

In order to save energy, works as in [14,15,16,44,48,42,1,35,33] consider models under specific management mechanisms, that leads to suboptimal strategies like threshold policy. They model homogeneous data center servers as a theoretical queuing model to evaluate the trade-off between waiting time and energy consumption where jobs arrive according to an identically independently distributed (i.i.d) Poisson process then served according to an exponentially distributed service time. Analysis based on analytic equation and experimental simulation show that energy consumption is significantly reduced (from 20% to 40%) while still keeping a reasonable waiting time. With other authors, we do a similar work [7] where arrival jobs are modeled as histograms. On the other hand there are other works focused on computing the optimal policy [28,49,31,27,9,15]. To achieve this aim, the authors formalize the energy saving models by *Markov Decision Process* (MDP).

MDP is a formalism allowing to model a decision system. Basically, it is a Markov chain augmented by a set of decision actions. Applying an action moves the system from a state to another state and gives rise to some monetary cost. Thus, the probability that the system moves into its new state is influenced by the chosen action. In the context of energy saving, the monetary cost of the consumed energy depends on the number of operational servers, and the QoS monetary cost depends on the number of waiting and/or rejected jobs. Additionally, from every state of the system we have to consider all different ways of turning-off/turning-on each server to find the optimal strategy which is the best sequence of actions to

minimize the expected total monetary cost accumulated during a finite period of time.

MDP models can be considered in continuous or discrete time context. In the case of discrete time MDP, the value iteration algorithm, that implements the Bellman's backup equations [8,39,11], computes the optimal policy. However, continuous time MDP are considered when the arrival jobs are modeled by a continuous time distribution obtained by fitting the empirical data or by asserting some assumptions that leads usually to a Poisson distribution [49,9,31]. By using the *uniformization* conversion process [2], the continuous time MDP is translated into a discrete time framework in which the value iteration algorithm can then be used to compute the optimal policy.

It is known that the size of an MDP is important, and the computation of the optimal policy can be hard even impossible for big models. Indeed, it is crucial to analyze the structural properties of the optimal policy to speed-up the computation. Several works as in [46,29,43,19,38] investigate the structure of optimal policy and give conditions to check double-threshold structure.

An MDP model for heterogeneous systems were considered in several works [28,36]. In particular [30,13,22] used continuous time MDP's for a heterogeneous data center where jobs arrive in a Poisson fashion and join a single queue served by several non identical servers. The authors study structural properties of the optimal policy as the threshold structure property.

In our work, real incoming traffic traces are sampled then used directly to build empirical discrete distributions (histograms) to model job arrivals and service rates. In fact, we can accommodate less regular processes than the Poisson process considered in several works like in [35,42]. Thus, assumptions like Poisson arrivals, exponential services, and the infinite buffer capacity are not required for our approach analysis and its optimization procedure. We already used histograms in [6,7], and similar works as in [45] used them for network traffic. The data center is modeled by a discrete time queue including a set of heterogeneous servers with different level of energy consumption and different service capacity. Notice that both arrival jobs and capacity processing of servers distributions are obtained from real traces, empirical data, or incoming traffic measurements. The buffer size is finite. We set the length of the time slot to the length of the sampling time used to sample the traffic traces to obtain the histogram. The problem of energy-performance optimization is formulated by a discrete time MDP without passing by uniformization process. Then we analyze the energy consumption and its evolution.

1.2 Outline of the paper

The rest of the paper is organized as following. Section 2 shows how the arrival jobs can be modeled by discrete distribution obtained from real Google traffic traces [47,41]. We also suggest a method to build a discrete identically independently distributions. Section 3 models the system by simple queue. Then, Section 4 formulates the optimization problem as discrete time MDP. After that, in Section 5 we prove some structural properties related to the optimal policy. Finally, Section 6

analyzes a real case study.

2 Arrivals Description

In this section we show how the arrival jobs are modeled by histograms.

Let A be a discrete random variables taking values in $\mathbb{N}$. The couple $\mathcal{H}_A = (S_A, P_A)$ denotes the histogram of A where $P_A : \mathbb{N} \to [0,1]$ is the probability mass function of A and $S_A = \{i \in \mathbb{N} : \ P_A(i) > 0\}$ is the support of P_A. Thus, the number of jobs arriving to the data center during a slot is modeled by a histogram $\mathcal{H}_A$ where $P_A(i)$ gives the probability to have i arrival jobs per a slot.

Example 2.1 Assume that, per slot, we have a probability of 0.14 to receive 3 arrival jobs, 0.19 to receive 7 arrival jobs, and 0.67 to receive 11 arrival jobs. In this case, arrivals are modeled by histogram $\mathcal{H}_A = (S_A, P_A)$ where $S_A = \{3, 7, 11\}$, $P_A(3) = 0.17$, $P_A(7) = 0.19$, and $P_A(11) = 0.67$.

To model job arrivals, we uses real traffic traces based on the open *cluster-data-2011-2* trace [47,41]. As in [7], we focus on the part that contains the job events corresponding to the requests destined to a specific Google data center for the whole month of May 2011. The job events are organized as a table of eight attributes. Column *timestamps* refers to the arrival job instant expressed in μ-sec.

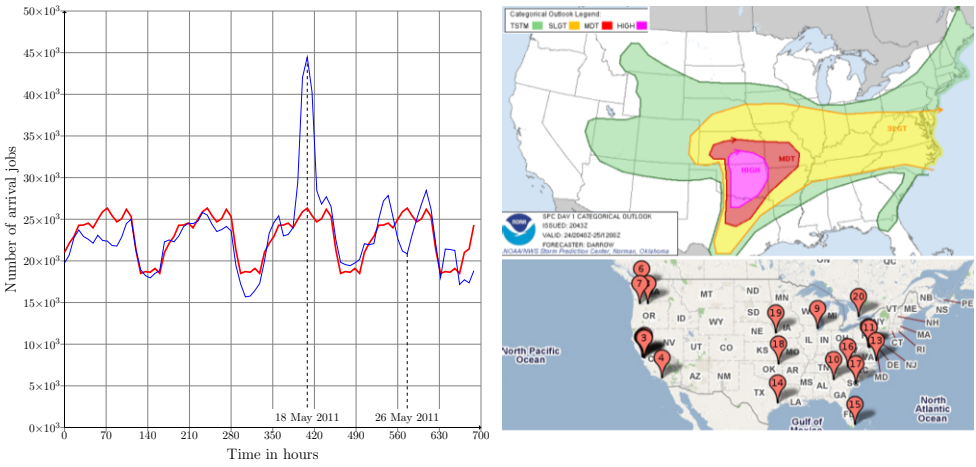


Fig. 2. At left: Evolution of the number of arrival jobs during a months for the Google trace (blue curve is the real traffic, the red curve is a periodic interpolation of the real traffic). Map at top right: Tornado outbreak sequence of May 21–26, 2011 – Day 4 of outbreak which included a 45% tornado area, above minimum high risk threshold. Map at bottom right: location of Google's data centers.

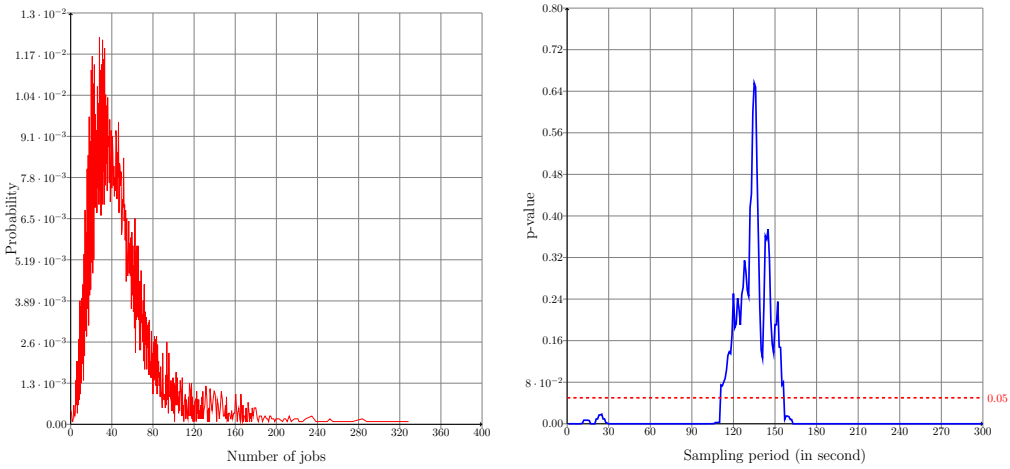


Fig. 3. Left: Google trace arrival jobs distribution. Right: i.i.d-ness test. The turning point test for the sampled data with a sampling period between 115 and 155 second has a $p-value$ bigger than > 0.05, thus at confidence level 95% we accept the null hypothesis, namely, the arrival jobs are i.i.d.

Figure 2 shows a summery of our time series analysis on the Google trace arrival jobs. The left graph shows the evolution of the number of arrival jobs during a months. Blue curve is the real traffic, the red curve is a periodic interpolation of the real traffic. We observe two aberrant points: the first one is a heavy load on Google's servers on May 18, 2011 (probably for DSK case judgment). The second one is a significant drop in load on May 26, 2011, which is probably related to meteorological conditions on the US southeast region. The two maps at right show Tornado impact and location of Google's computing centers.

This traffic trace must be sampled with a sampling period for which the arrival jobs can be considered as i.i.d. In fact, we sample the data with several sampling periods between one and three hundred second, then we applied the turning point test [21] for each sampling periods in order to test the i.i.d-ness of the sampled data. We found that sampling period between 115 and 155 second has a $p-value$ bigger than 0.05, thus at confidence level 0.95 we accept the null hypothesis, namely, the sampled data is i.i.d. Thus, we consider frames of 136 second to sample the trace and construct one empirical distribution. The support of the obtained histogram is formed of around 200 bins, and the average of arrival jobs is around 50 jobs per slot: $\mathbb{E}(\mathcal{H}_A) = 50$.

3 Model Description

In this work we consider discrete time queue model. Our queuing model is a batch arrival queue with a finite capacity buffer b. We model arrival jobs and service rates by discrete distributions (histograms).

A data center is composed of heterogeneous servers grouped essentially into several levels of energy consumption. To simplify we consider the set of the following levels $G = \{high, med, low\}$ (med for medium). The number of operational servers of type g is denoted by m_g and max_g is the maximal number of servers of level g.

During one time unit, the number of jobs that can be served by a server of type g is modeled by a histogram $\mathcal{H}_{D_g}$ where $P_{D_g}(d)$ gives the probability to process d jobs. Thus, the total number of operational servers is $\sum_{g \in G} m_g$ and $max = \sum_{g \in G} max_g$ is the maximal number of operational servers.

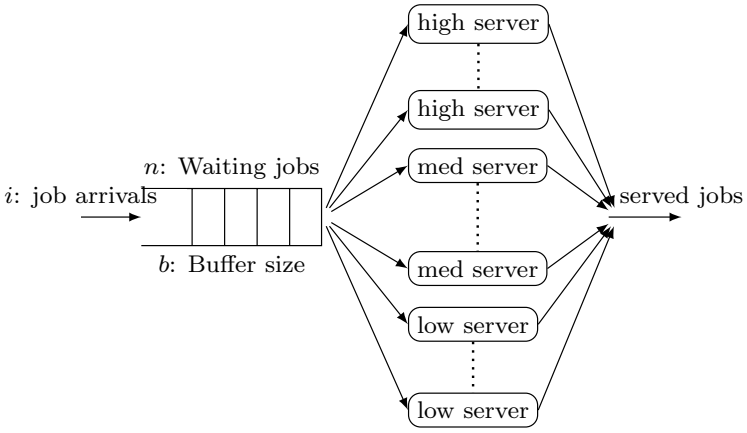


Fig. 4. Illustration of the queuing model.

The number of waiting jobs in the buffer is denoted by n. The number of rejected (lost) jobs is denoted by l. We assume that initially the number of operational servers, the number of waiting jobs, and the number of rejected jobs are 0. The number of waiting jobs n can be computed by induction where the exact sequence of events during a slot have to be described. First, the jobs are added to the buffer then they are executed by the servers. The admission is performed per job, according to the Tail Drop policy: a job is accepted if there is a place in the buffer, otherwise it is rejected. The job is assigned, at first, to a high free running server, if no high server available, the job is attributed to a medium free running server, otherwise it will be given to a low free running server. The following equations give the number of waiting jobs in the buffer and the lost jobs. If i jobs arrive, we have:

$$\begin{cases} n \leftarrow \min\{b, \max\{0, n + i - \sum_{g \in G} m_g \times d_g\}\} \\ l \leftarrow \max\{0, n + i - \sum_{g \in G} m_g \times d_g - b\} \end{cases} \tag{1}$$

It is assumed that the input arrivals are i.i.d. and under these assumptions, the model of the queue is a time-homogeneous Discrete Time Markov Chains. As we are dealing with real traffic trace, to ensure the i.i.d-ness assumption, we have to find the sampled period for which the sampled trace can be assumed as i.i.d. In fact we use a statistical hypothesis testing called *the turning point test* [21] to test the i.i.d-ness of the sampled trace, then we consider only a sampled period for which the data is i.i.d (see Section 2).

The problem we have to consider is to find a trade-off between the performance

(i.e. waiting and loss jobs) and the energy consumption (i.e. number of operational servers). However, as the number of servers changes with time, the system becomes more complex to analyze. The number of servers may vary according to the traffic and performance indexes. More precisely, depending on n, l, and a particular cost function some decisions are taken to change m_{high}, m_{med}, and m_{low}.

3.1 Energy and Performance metric

The energy consumption takes into account the number of operational servers. A server may consume a very low amount of energy when it is turned-off. But consumes some units of energy per slot when it is operational and it costs some monetary units (€, \$, £). Even we consider that a server switches-on immediately, a server may consume an additional amount of energy. The total consumed energy during a specific period is the sum of all consumed energy. QoS takes into account the number of waiting and lost jobs. Each waiting/rejected job costs some monetary units.

Cost	**Meaning**
$c_{M_g} \in \mathbb{R}^+$	energy cost for running one operational server of type g
$c_{On_g} \in \mathbb{R}^+$	energy cost needed to switch-on a server of type g
$c_N \in \mathbb{R}^+$	waiting cost for one job per unit of time
$c_L \in \mathbb{R}^+$	rejection cost of one lost job

3.2 Objective function

For a given period of time, a dynamic control policy for energy-aware consists in doing each slot an action (turn-on or turn-off a specific number of servers) in order to adapt the number of operational servers to incoming job changes. The optimal strategy consists in finding the best sequence of actions to minimize the overall monetary cost (energetic cost plus performance cost). The cost generated for each slot can be presented as:

$$n \times c_N + l \times c_L + \sum_{g \in G} (m'_g \times c_{M_g} + \max\{0, m'_g - m_g)\} \times c_{On_g}) \tag{2}$$

where n is the number of waiting jobs, l is the number of lost jobs, m_g is the number of operational servers of level g before doing an action, and m'_g is the number of operational servers after doing the action.

4 Markov Decision Process

In order to find the optimal strategy and then analyze the performance and the energy consumption of the data center under this optimal strategy, we will use the concept of *Markov Decision Process* to formulate our optimization problem. Notice that (n, l) and $\{m_g : g \in G\}$ are mutually dependent. If we decrease m_g, we save more energy, but both n and l increase, so we will have an undesirable diminution of QoS and vice versa.

Let $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{C})$ be an MDP where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the set of actions, $\mathcal{P}$ is the transition probability, and $\mathcal{C}$ the immediate cost of each action. Let $\mathcal{H}_A = (S_A, P_A)$ be the histogram used to model the arrival jobs. The state of the system is defined by the couple $((m_{high}, m_{med}, m_{low}), (n, l))$, indeed the state space $\mathcal{S}$ is defined as:

$$\mathcal{S} = \{((m_g : \ g \in G), (n, l)) \mid m_g \in [0..max_g],\ n \in [0..b] \text{ and } l \in [0..\max(S_A)]\} \quad (3)$$

The action space is defined as:

$$\mathcal{A} = \{\alpha_{(x,y,z)} \mid x \in [0..max_{high}],\ y \in [0..max_{med}],\ z \in [0..max_{low}]\} \quad (4)$$

where action $\alpha_{(x,y,z)}$ consist of keeping exactly: x high, y medium, and z low operational servers, during the current slot. Indeed, we have a probability of $\mathcal{P}_{ss'}^{(\alpha_{(x,y,z)})}$ to move from state $s = (m_{high}, m_{med}, m_{low}, (n, l))$ to state $s' = (x, y, z, (n', l'))$ under action $\alpha_{(x,y,z)}$. This probability is defined as:

$$\mathcal{P}_{ss'}^{\alpha_{(x,y,z)}} = \sum_{\left\{\begin{array}{l}\text{for each } i \in S_A \text{ and each } d_g \in S_{D_g} \text{ satisfying:} \\ n' = \min\{b, \max\{0, n + i - x \times d_{high} - y \times d_{med} - z \times d_{low}\}\} \\ l' = \max\{0, n + a - x \times d_{high} - y \times d_{med} - z \times d_{low} - b\}\end{array}\right.} P_A(i) \times \prod_{g \in G} P_{D_g}(d_g) \quad (5)$$

Where n' is the new number of waiting jobs, l' is the new number of lost jobs, and x, y, z is the number of operational servers for each level of G after applying action $\alpha_{(x,y,z)}$. In this case, applying action $\alpha_{(x,y,z)}$ induces immediately a cost $\mathcal{C}_s^{\alpha_{(x,y,z)}}$ defined as:

$$\begin{aligned}\mathcal{C}_s^{\alpha_{(x,y,z)}} = n \times c_N + l \times c_L \\ +x \times c_{M_{high}} \quad + \max\{0, x - m_{high}\} \times c_{On_{high}} \\ +y \times c_{M_{med}} \quad + \max\{0, y - m_{med}\} \times c_{On_{med}} \\ +z \times c_{M_{low}} \quad + \max\{0, z - m_{low}\} \times c_{On_{low}}\end{aligned} \quad (6)$$

The immediate cost $\mathcal{C}_s^{\alpha_{(x,y,z)}}$ includes two parts. The QoS part which includes cost due to waiting and rejected jobs. Power part composed of energy consumed for running operational servers and energy used to power-up additional servers. Table 1 resumes parameters used in our model and our MDP formulation.

Every state of the MDP includes three elements:

(i) the number of operational servers for every level which is between 0 and max_g,

(ii) the number of waiting jobs in the buffer which is bounded by b, and

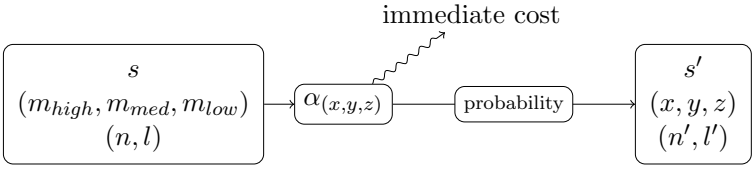
(iii) the number of rejected jobs which can be at most equals to the maximum number of arrival jobs given by $\max(S_A)$.

So, $|\mathcal{S}|$ is bounded by $(b+1) \times (\max(S_A)+1) \times \prod_{g \in G}(max_g + 1)$. In fact the number

of state of the MDP is in $\mathcal{O}\left(b \times \max(S_A) \times \prod_{g \in G} max_g\right)$. Additionally, from each state of the MDP we have at most $\prod_{g \in G}(max_g + 1)$ action, and each action leads to a number of transition equals to $|S_A|$ (one transition for each bin in the support of the arrival distribution). Then, our MDP transitions number is in $\mathcal{O}\left(b \times |S_A| \times \max(S_A) \times \prod_{g \in G} max_g^2\right)$.

Table 1
Model and MDP Parameters.

Parameters	Description
h	duration of analysis
b	buffer size
max_g	total number of servers of type g
$\mathcal{H}_{D_g}$	histogram modeling the processing capacity of a server of type g
$\mathcal{H}_A$	histogram of job arrivals
m_g	number of operational servers of level g
n	number of waiting jobs
l	number of rejected jobs
$\mathcal{S}$	set of all possible states
$\mathcal{A}$	set of all possible actions
s	$s = ((m_g : \ g \in G), (n, l))$ system state
s_0	$s_0 = ((0,0,0),(0,0))$ starting state
$\alpha_{(x,y,z)}$	action to keep exactly: x high operational servers y medium operational servers z low operational servers
$\mathcal{P}_{ss'}^{\alpha_{(x,y,z)}}$	probability of transition from s to s' under action $\alpha_{(x,y,z)}$
$\mathcal{C}_s^{\alpha_{(x,y,z)}}$	immediate cost from s under action $\alpha_{(x,y,z)}$



Example 4.1 To illustrate our formalization let's show a little part [4] of an MDP modeling a very simple data center of for servers (one high server, two medium level servers, and one low server) with a buffer size equals to 4. Thus $max = 4$ and $b = 4$.

[4] The number of states of this MDP is around 700.

Additionally, to keep the example simple, we set service rate of each machine type as a histogram with only one bin: $P_{D_{high}}(3) = P_{D_{med}}(2) = P_{D_{low}}(1) = 1$. Job arrivals are modeled by histogram of Example 2.1. For instance, state $s_{82} = ((1,0,1),(3,1))$ means that the system is running by only one high server and one low server, when three jobs wait in the buffer and one job where rejected. Action $\alpha_{(0,2,1)}$ switches-off the high server and switches-on the two low servers. This action induces an immediate cost of 1.45 and move the system to state s_{380} with probability 0.14, to state s_{502} with probability 0.67, and to state s_{121} with probability 0.19.

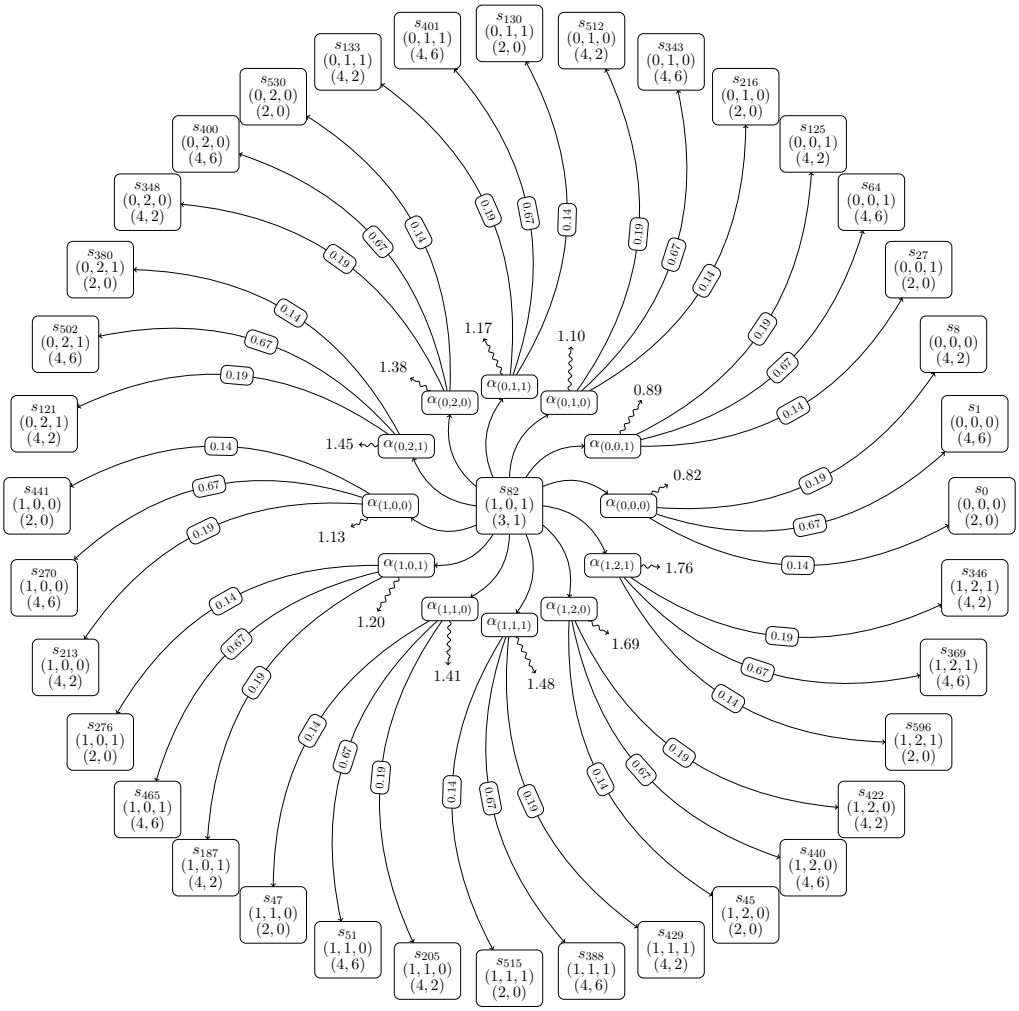


Fig. 5. MDP example.

5 Optimal strategy

Each unit of time, every action consists in turning-on a specific number of servers and turning-off the rest of the servers. The optimal strategy is the best sequence of actions to be done in order to minimize the overall cost during a finite period of time

called *horizon* and noted h. More generally, the value function $V : \mathcal{S} \times [0..h] \to \mathbb{R}^+$ has as objective minimizing the expected sum of costs over time:

$$V(s,t) = \min_{\pi} \mathbb{E}\left[\sum_{k=0}^{t} \mathcal{C}_{s_k}^{\pi(s_k,k)}\right] \tag{7}$$

The value function can be seen as a Bellman equation [8,39,11]:

$$V(s,t) = \min_{\alpha_{(x,y,z)}} \left\{ \mathcal{C}_s^{\alpha_{(x,y,z)}} + \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^{\alpha_{(x,y,z)}} V(s', t-1) \right\} \tag{8}$$

Where $\alpha_{(x,y,z)}$ is the action taken by the system, and $\mathcal{P}_{ss'}^{\alpha_{(x,y,z)}}$ is the transition probability from state s to state s'. In this case the optimal policy for each state s is:

$$\pi^*(s,t) = \operatorname*{argmin}_{\alpha_{(x,y,z)} \in \mathcal{A}} \left\{ \mathcal{C}_s^{\alpha_{(x,y,z)}} + \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^{\alpha_{(x,y,z)}} V(s', t-1) \right\} \tag{9}$$

As the value iteration algorithm is an efficient dynamic programming, implementation for solving Bellman equation, we used it to compute the optimal control policy of our MDP.

As shown previously, the size of the MDP is important, and the computation of the optimal policy can be hard, and even impossible for a big data center. In fact, it is essential to analyze the structural properties of the optimal policy to make the computation efficient. In particular the double-threshold structure was proved in the case of a homogeneous data center in several models like in [49]. In the following we will be interested in some properties around the optimal policy, and we show in particular that the property of the double-threshold structure does not hold for our heterogeneous data center model.

Definition 5.1 [[46,19]] A policy π is called to be **hysteretic** for slot t if $\exists m \in [0..max]$, $\exists \alpha_{(x,y,z)} \in \mathcal{A}$ such as:

$$\pi(((m_{high}, m_{med}, m_{low}), (n,l)), t) = \alpha_{(x,y,z)} \implies \pi(((x,y,z),(n,l)),t) = \alpha_{(x,y,z)}.$$

In addition if π can be formulated as a double-threshold structure then we said that π is **monotone**. Furthermore if the thresholds of the double-threshold structure are ordered then π is called to be **isotone** [46,19].

Theorem 5.2 *The optimal policy (9) is not hysteretic.*

Proof. Let $m = (m_{high}, m_{med}, m_{low})$ and $j = (x,y,z)$. Let's define function $f(m,j) : \left(\prod_{g \in G} [0..max_g]\right)^2 \to \mathbb{R}^+$ as the cost for switching the number of high operational servers from m_{high} to x, medium operational servers from m_{med} to y, and low operational servers from m_{low} to z. And function $w((m,(n,l)),t) : \mathcal{S} \times [0..h] \to \mathbb{R}^+$ presents the expected and the possible future cost starting with n waiting jobs in

the buffer, losing l jobs, serving with $(m_{high}, m_{med}, m_{low})$ operational servers during one slot, and then following an optimal policy. So, our optimal policy (9) can be formulated as:

$$\pi^*(s,t) = \underset{\alpha_{(x,y,z)} \in \mathcal{A}}{\operatorname{argmin}} \{f(m,j) + w((j,(n,l)),t)\} \tag{10}$$

Where:

$$\begin{cases} w((j,(n,l)),t) = x \times c_{M_{high}} + y \times c_{M_{med}} + z \times c_{M_{low}} + n \times c_N + l \times c_L \\ \qquad\qquad + \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^{\alpha_{(x,y,z)}} V(s', t-1) \\ f(m,j) \qquad = \max\{0, x - m_{high}\} \times c_{On_{high}} \\ \qquad\qquad + \max\{0, y - m_{med}\} \times c_{On_{med}} \\ \qquad\qquad + \max\{0, z - m_{low}\} \times c_{On_{low}} \end{cases} \tag{11}$$

According to Theorem 1 of [19], the following condition is necessary for the policy to be hysteretic:

$$\forall (m,j) : \sum_{g \in G} m_g = x + y + z \implies f(m,j) = 0 \tag{12}$$

which means that moving between two states with the same total number of operational servers does not increase or decrease the switching energy cost. Condition 12 does not hold for our MDP model. We have just to choose for a counterexample $m = (x,y,z)$, $j = (x+1, y-1, z)$ and a strictly positive value for every c_{On_g}. □

Corollary 5.3 *The optimal policy (9) is neither isotone nor monotone.*

Proof. From Definition of monotony and isotony given in [43], hysteresis is a necessary condition for isotony and monotony. However form Theorems 5.2, we deduce that the optimal policy (9) is neither isotone nor monotone. □

6 Case study

This case study uses real traffic traces based on the open *cluster-data-2011-2* trace [47,41]. As in [7], we model arrival jobs, and additionally service rates, by discrete distributions build from the job/machine events corresponding to the requests destined to a specific Google data center for the whole month of May 2011. This traffic trace is sampled with a sampling period that ensure the i.i.d-ness assumption. Thus, we consider frames of 136 second to sample the trace and construct empirical distributions.

In order to specify, solve, then analyze the energy consumption of our data centers, we use an efficient model checker called PRISM [25] which is a probabilistic software that can be used for the specification of MDP models. In PRISM we need

to write a specification for our MDP model which can be written easily if the system parameters were small, but when we consider real systems with big parameters it will be very hard. As an example the PRISM specification associated with a data center with 10 heterogeneous servers is a file of several thousands of lines, all lines are different from each other. Trying to write our specification without automated generation is time consuming, gives rise to making easily mistakes, forgetting some cases, losing time for updating or maintaining. Indeed, as the specification of our heterogeneous model is huge and not obvious to write (see Section 4), we have coded a tool that helps us to generate automatically the PRISM specification file.

Table 2 resumes values of the parameters of our system. Notice that the order of magnitude of costs is more important than the effective value of each cost. For instance costs of high machines are more important than other machine costs. Figure 6 shows the minimal expected overall cost during one day ($h = 1$ day) when varying the buffer size b and the unitary waiting/losing job cost and keeping the rest of parameters as in Table 2.

Table 2
Parameters of the numerical analysis.

b	max	$\mathbb{E}(\mathcal{H}_{D_{high}})$	$\mathbb{E}(\mathcal{H}_{D_{med}})$	$\mathbb{E}(\mathcal{H}_{D_{low}})$	$c_{M_{high}}$	$c_{M_{med}}$	$c_{M_{low}}$	$c_{On_{high}}$	$c_{On_{med}}$	$c_{On_{low}}$	c_N	c_L
50	10	10	5	2	6	4	2	3	2	1	1	2

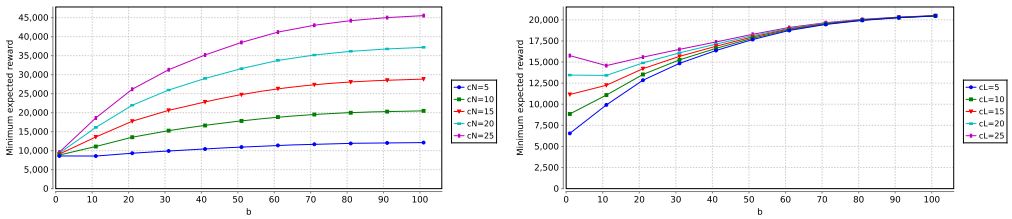


Fig. 6. At left: varying the unitary waiting job cost c_N where keeping other unitary costs constant. We observe that the more c_N is big the more the total cost is important. At right: varying the unitary losing job cost c_L where keeping other unitary costs constant. For big values of b, unitary cost for loosing jobs c_L does not affect the total cost of the system.

Table 3 resumes the obtained results for the following configurations:

(i) a data center with ten heterogeneous servers,

(ii) a data center with ten identical high servers,

(iii) a data center with ten identical medium servers,

(iv) a data center with ten identical low servers.

Table 3
Result of the numerical analysis.

	(i) Heterogeneous servers	(ii) Homogeneous high servers	(iii) Homogeneous medium servers	(iv) Homogeneous low servers
$\mathbf{max_{high}}$	2	10	0	0
$\mathbf{max_{med}}$	5	0	10	0
$\mathbf{max_{low}}$	3	0	0	10
Prism lines	7×10^4	7.5×10^2	7.5×10^2	7.5×10^2
States	1.6×10^4	1.6×10^3	1.6×10^3	1.6×10^3
Transitions	88×10^6	0.81×10^6	0.87×10^6	0.89×10^6
Memory	2 GB	18.8 MB	20.4 MB	20.7 MB
Time	1.6 hour	1 minute	2 minute	2 minute
Expected minimal cost	50342	52251	54175	56381

Results show that heterogeneous model saves more energy (12% better then homogeneous model). However, the size of the three homogeneous models is relatively small compared to the size of the heterogeneous one which is huge. The size of heterogeneous model is due to the large number of different action combinations.

Figure 7 shows the result of experiments in which we are analyzing the total cost over one day when varying the buffer size b for heterogeneous/homogeneous data centers. We observe that the total cost increase when b is less than some value ($b_0 \simeq 40$). However, when b is bigger than b_0 the total cost seams to be convergent for any configuration. We can explain this behavior as following: for a small size of the buffer, the number of waiting jobs is low. This leads to a small number of served jobs. In fact the system switches-on a less number of servers. So the energetic and also the waiting jobs are low. When the buffer size is bigger, the number of waiting jobs is more important. Which leads to a bigger number of served jobs. In fact the system switches-on more servers. So the energetic and also the waiting jobs increase. As the arrival jobs are bounded, when the buffer size is bigger than some value, the number of waiting jobs and the number of served job become stable which leads to a stationary number of running servers and necessarily to a stationary total cost.

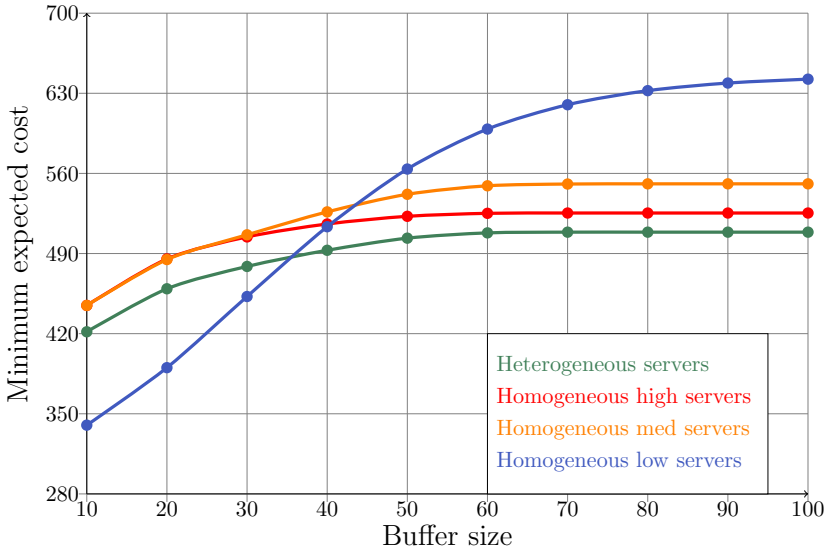


Fig. 7. Total cost when varying buffer size b for heterogeneous and homogeneous data centers.

We can also observe that:

- globally heterogeneous model leads to a better cost minimization: this can be explained by the fact that heterogeneous model presents a huge number of ways to combine actions, which leads to a better control policy that may save more energy.
- homogeneous model with low servers induces an important cost: this can be explained by the fact that low servers have a low service capacity ($\mathbb{E}(\mathcal{H}_{D_{low}}) = 2$) which leads to an important number of waiting jobs.

7 Conclusion

This paper presents a discrete-time Markov decision process model for optimal management of a data center with heterogeneous servers, with the objective of minimizing energy consumption and Quality of Service (QoS) costs. Additionally, some theoretical results are presented, as well as a numerical study on the performance of the optimal policy.

In this work, job arrivals and service rates are modeled by a general discrete probability distributions, that can be estimated from real data as histograms. The optimal control policy is computed by the value iteration algorithm and used to define the Dynamic Power Management that ensures the trade-off between performance and energy consumption. We proved that neither hysteretic nor monotony property holds for optimal heterogeneous policy. Consequently, the optimal policy cannot be designed as a simple double-threshold structure.

Theoretical and experimental results show that the size of heterogeneous model is bigger than the size of homogeneous model. However, from experimental results, it seems that increasing server heterogeneity leads to more potential energy savings.

As future work, we consider to approximate the heterogeneous model by a homogeneous one, and apply some metaheuristic methods to approximate as better as possible the optimal policy when avoiding state space explosion.

References

[1] K. Aidarov, Paul D. Ezhilchelvan, and Isi Mitrani. Energy-aware management of customer streams. *Electr. Notes Theor. Comput. Sci.*, 296:199–210, 2013.

[2] Oguzhan Alagoz and Mehmet US Ayvaci. Uniformization in markov decision processes. *Wiley Encyclopedia of Operations Research and Management Science*.

[3] Jayant Baliga, Robert WA Ayre, Kerry Hinton, and Rodney S Tucker. Green cloud computing: Balancing energy in processing, storage, and transport. *Proceedings of the IEEE*, 99(1):149–167, 2011.

[4] Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle. The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis lectures on computer architecture*, 8(3):1–154, 2013.

[5] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12), 2007.

[6] Marziyeh Bayati. Managing energy consumption and quality of service in data centers. In *Proceedings of the 5th International Conference on Smart Cities and Green ICT Systems*, pages 293–301, 2016.

[7] Marziyeh Bayati, Mohamed Dahmoune, Jean-Michel Fourneau, Nihal Pekergin, and Dimitrios Vekris. A tool based on traffic traces and stochastic monotonicity to analyze data centers and their energy consumption. *EAI Endorsed Trans. Energy Web*, 3(10):e3, 2016.

[8] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[9] Luca Benini, Alessandro Bogliolo, Giuseppe A Paleologo, and Giovanni De Micheli. Policy optimization for dynamic power management. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(6):813–833, 1999.

[10] Theophilus Benson, Aditya Akella, and David A Maltz. Network traffic characteristics of data centers in the wild. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 267–280. ACM, 2010.

[11] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[12] Dmytro Dyachuk and Michele Mazzucco. On allocation policies for power and performance. In *2010 11th IEEE/ACM International Conference on Grid Computing*, pages 313–320. IEEE, 2010.

[13] Dmitry Efrosinin. *Controlled queueing systems with heterogeneous servers*. PhD thesis, Univ. Trier, FB 4, Informatik, 2004.

[14] Anshul Gandhi, Sherwin Doroudi, Mor Harchol-Balter, and Alan Scheller-Wolf. Exact analysis of the m/m/k/setup class of markov chains via recursive renewal reward. In *ACM SIGMETRICS Performance Evaluation Review*, volume 41, pages 153–166. ACM, 2013.

[15] Anshul Gandhi, Varun Gupta, Mor Harchol-Balter, and Michael A Kozuch. Optimality analysis of energy-performance trade-off for server farm management. *Performance Evaluation*, 67(11):1155–1171, 2010.

[16] Anshul Gandhi, Mor Harchol-Balter, and Ivo Adan. Server farms with setup costs. *Performance Evaluation*, 67(11):1123–1138, 2010.

[17] Albert G. Greenberg, James R. Hamilton, David A. Maltz, and Parveen Patel. The cost of a cloud: research problems in data center networks. *Computer Communication Review*, 39(1):68–73, 2009.

[18] Dirk Grunwald, Charles B. Morrey, III, Philip Levis, Michael Neufeld, and Keith I. Farkas. Policies for dynamic clock scheduling. In *Proceedings of the 4th Conference on Symposium on Operating System Design & Implementation - Volume 4*, OSDI'00, pages 6–6, Berkeley, CA, USA, 2000. USENIX Association.

[19] Sabine K Hipp and Ulrich D Holzbaur. Decision processes with monotone hysteretic policies. *Operations Research*, 36(4):585–588, 1988.

[20] Yichao Jin, Yonggang Wen, and Qinghua Chen. Energy efficiency and server virtualization in data centers: An empirical investigation. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 133–138. IEEE, 2012.

[21] M.G. Kendall. *Time-series*. Griffin, 1973.

[22] Jung Hyun Kim, Hyun-Soo Ahn, and Rhonda Righter. Managing queues with heterogeneous servers. *Journal of Applied probability*, 48(02):435–452, 2011.

[23] Jonathan Koomey. Growth in data center electricity use 2005 to 2010. *A report by Analytical Press, completed at the request of The New York Times*, page 9, 2011.

[24] Christos Kozyrakis, Aman Kansal, Sriram Sankar, and Kushagra Vaid. Server engineering insights for large-scale online services. *IEEE micro*, 30(4):8–19, 2010.

[25] Marta Kwiatkowska, Gethin Norman, and David Parker. Prism: Probabilistic symbolic model checker. In *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 200–204. Springer, 2002.

[26] Young Choon Lee and Albert Y Zomaya. Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, 60(2):268–280, 2012.

[27] B. Leng, B. Krishnamachari, X. Guo, and Z. Niu. Optimal operation of a green server with bursty traffic. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2016.

[28] Keqin Li. Optimal power allocation among multiple heterogeneous servers in a data center. *Sustainable Computing: Informatics and Systems*, 2(1):13–22, 2012.

[29] FV Lu and Richard F Serfozo. M/m/1 queueing decision processes with monotone hysteretic optimal policies. *Operations Research*, 32(5):1116–1132, 1984.

[30] Hsing Paul Luh and Ioannis Viniotis. Threshold control policies for heterogeneous server systems. *Mathematical Methods of Operations Research*, 55(1):121–142, 2002.

[31] Vincent J Maccio and Douglas G Down. On optimal control for energy-aware queueing systems. In *Teletraffic Congress (ITC 27), 2015 27th International*, pages 98–106. IEEE, 2015.

[32] Jason Mars, Lingjia Tang, and Robert Hundt. Heterogeneity in "homogeneous" warehouse-scale computers: A performance opportunity. *IEEE Computer Architecture Letters*, 10(2):29–32, 2011.

[33] Michele Mazzucco and Dmytro Dyachuk. Optimizing cloud providers revenues via energy efficient server allocation. *Sustainable Computing: Informatics and Systems*, 2(1):1–12, 2012.

[34] Michele Mazzucco, Dmytro Dyachuk, and Marios Dikaiakos. Profit-aware server allocation for green internet services. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2010 IEEE International Symposium on*, pages 277–284. IEEE, 2010.

[35] Isi Mitrani. Managing performance and power consumption in a server farm. *Annals OR*, 202(1):121–134, 2013.

[36] Ripal Nathuji, Canturk Isci, and Eugene Gorbatov. Exploiting platform heterogeneity for power efficient data centers. In *Autonomic Computing, 2007. ICAC'07. Fourth International Conference on*, pages 5–5. IEEE, 2007.

[37] C. D. Patel, C. E. Bash, R. Sharma, and M. Beitelmal. Smart cooling of data centers. In *Proceedings of IPACK*, 2003.

[38] Hans-Joachim Plum. Optimal monotone hysteretic markov policies in an m/m/1 queueing model with switching costs and finite time horizon. *Mathematical Methods of Operations Research*, 35(5):377–399, 1991.

[39] Martin L Puterman. *Markov Decision Processes. J.* Wiley and Sons, 1994.

[40] Chheda Rajesh, Shookowsky Dan, Stefanovich Steve, and Toscano Joe. Profiling energy usage for efficient consumption. *The Architecture Journal*, 18:24, 2008.

[41] Charles Reiss, John Wilkes, and Joseph L. Hellerstein. Google cluster-usage traces: format + schema. Technical report, Google Inc., Mountain View, CA, USA, November 2011. Revised 2012.03.20.

[42] C. Schwartz, R. Pries, and P. Tran-Gia. A queuing analysis of an energy-saving mechanism in data centers. In *Information Networking (ICOIN), 2012 International Conference on*, pages 70–75, Feb 2012.

[43] Richard F Serfozo. Technical note—an equivalence between continuous and discrete time markov decision processes. *Operations Research*, 27(3):616–620, 1979.

[44] Joris Slegers, Nigel Thomas, and Isi Mitrani. Dynamic server allocation for power and performance. In *SPEC International Performance Evaluation Workshop*, pages 247–261. Springer, 2008.

[45] Jean-Sébastien Tancrez, Pierre Semal, and Philippe Chevalier. Histogram based bounds and approximations for production lines. *European Journal of Operational Research*, 197(3):1133–1141, 2009.

[46] Donald M Topkis. Minimizing a submodular function on a lattice. *Operations research*, 26(2):305–321, 1978.

[47] John Wilkes. More Google cluster data. Google research blog, November 2011. Posted at `http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html`.

[48] Xiuli Xu and Naishuo Tian. The m/m/c queue with (e, d) setup time. *Journal of Systems Science and Complexity*, 21(3):446–455, 2008.

[49] Zexi Yang, Meng-Hsi Chen, Zhisheng Niu, and Dawei Huang. An optimal hysteretic control policy for energy saving in cloud computing. In *Global Telecommunications Conference*, pages 1–5. IEEE, 2011.