Managing Energy Consumption and Quality of Service in Data Centers

Marziyeh Bayati
Département d'Informatique, LACL, Université Paris-Est Créteil (UPEC), 61 avenue du Général de Gaulle, Créteil, France

Keywords: Data Center, Discrete Stochastic Process, Energy consumption, Energy Saving, Numerical Analysis, Quality of Service, Queues, Service Performance, Simulation.

Abstract: The main goal of this paper is to manage the switching on/off of servers in a data center during time to adapt the system with incoming traffic changes to ensure a good performance and a reasonable energy consumption. In this work, the system is modeled by a queue then, an optimization algorithm is designed to manage energy consumption and quality of service in the data center. For several systems, the algorithm is tested by numerical analysis under various types of job arrivals: arrivals with constant rate, arrivals defined by an constant discrete distribution, arrivals specified by a variable discrete distribution over time, and arrivals modeled by discrete distributions obtained from real traffic traces. The optimization algorithm that we suggest, adapts and adjusts dynamically the number of operational servers according to: traffic variation, workload, cost of keeping a job in the buffer, cost of losing a job, and energetic cost for serving a job.

1 INTRODUCTION

The increasing development of *Data Centers* is causing problems in energy consumption. More than 1.3% of the global energy consumption is due to the electricity used by data centers, a rate that is increasing, revealed by a survey conducted in (Koomey, 2011), which says a lot about the increasing evolution of data centers. Therefore, to ensure both a good performance of services offered by these data centers and reasonable energy consumption, a detailed analysis of the behavior of these systems is essential for designing efficient optimization algorithms to reduce the energy consumption.

Two requirements are in conflict: (i) Switching on a maximum number of servers leads to less waiting time and decreases the loss of jobs but requires a high energy consumption. (ii) Switching on a minimum number of servers leads to less energy consumption, but causes more waiting time and increases the loss of jobs.

The goal is to design better managing algorithms which take into account these two constraints to minimize: waiting time, loss rate and energy consumption.

Studies like in (Berl et al., 2010; Baliga et al., 2011; Lee and Zomaya, 2012) 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 total cost of the data center). Thus the main factor of this energy consumption is related to the number of operational servers. Many efforts have focused on servers and their cooling. Works have been done to build better components, low-energy-consumption processors, more efficient energy circuits, more efficient cooling systems (Grunwald et al., 2000), and optimized kernels (Patel et al., 2003).

(Aidarov et al., 2013) analyze an energy optimization strategy for a data center where: (i) job arrivals rate, (ii) service price, (iii) promised Quality of Service (QoS), (iv) penalty paid by the supplier if the QoS provided is less than promised, are fixed. Their objective is to maximize revenues from the service provider. The strategy should find a balance between minimizing the penalties, minimizing the cost of energy, and maximizing the number of served jobs. They used queues as model and tested their strategy by simulations.

(Mazzucco and Mitrani, 2012) provide a real test of an energy optimization strategy. The strategy is to switch on a number of servers : the more waiting jobs are observed, the more servers are switched on, and vice versa. Starting and stopping servers are progressive. This strategy has been tested on the platform Cloud Amazon EC2 with a cluster of servers. They have compared their strategy with two other strategies: (i) keeping all servers switched on, (ii) starting and stopping servers periodically.

(Mitrani, 2013) studies the problem of managing

Bayati, M.
Managing Energy Consumption and Quality of Service in Data Centers.
In *Proceedings of the 5th International Conference on Smart Cities and Green ICT Systems (SMARTGREENS 2016)*, pages 293-301
ISBN: 978-989-758-184-7

a data center to keep a low energy consumption. This problem is modeled by a queue in which jobs can leave the system if the waiting time is too long. The proposed strategy is to consider reserve server groups that are gradually switched on when the number of jobs in the buffer exceeds a certain threshold. Similarly these server groups are gradually switched off when the number of jobs in the buffer decreases and exceeds another threshold. The thresholds are analytically evaluated using an objective function that takes into account the parameters of the systems.

(Chase et al., 2001) work on an approach based on the management of servers based on the supplied performance. The system continuously monitors its workload and provides resource allocations by estimating their effects on the performance of the service. They describe a greedy resource allocation algorithm to balance supply and demand. Their experimental results show that the consumption of energy is reduced by 30%.

(Schwartz et al., 2012) present a theoretical queuing model to find a compromise between jobs waiting time and energy consumption where a group of servers is active all the time and the remaining servers are activated on request.

In (Bayati et al., 2015) we present, with other coauthors, a tool to study the trade-off between energy consumption and performance evaluation. The tool uses real traffic traces and stochastic monotonicity property to insure fast computation. Given a set of parameters that are fixed by the modeler, the tool determines the best threshold based policy.

Our approach differs from these methods by several points. First it is numerical rather than analytical or simulation based. Thus, this paper considers less regular processes than the for example Poisson process considered by Mitrani. Note that the Markovian assumptions (Poisson arrivals, exponential services and switching times) and the infinite buffer capacity are not mandatory for this analysis. However, here, the arrival process is assumed to be stationary for short periods of time and change between periods. This allows us to represent for instance hourly or daily variations of the job arrivals. Real traffic traces are used to build discrete distributions for the job arrival.

In this paper a data center is modeled by a discrete time queue with a finite buffer capacity and with a time slot equals to the sampling period used to sample the traffic traces. The job arrivals are specified by a discrete distribution. The system is analyzed for a finite time period (let's say a day or a week). This time period is divided into sub-intervals where the batch of arrivals are supposed constant

We design an optimization algorithm in order to manage energy consumption and QoS in the data center. The cost of the consumed energy depends on the number of operational servers. The QoS cost depends on the number of waiting time (which depends on the number of jobs in the buffer) and the losing rate (which depends on the number of lost jobs). Every slot, our algorithm minimizes an objective function that combines the cost of energy and the cost of QoS, in order to increase or decrease the number of operational servers according to traffic variation.

As the model is solved numerically, it is much faster and more accurate than simulation.

The rest of this paper is organized as follows. At first Section 2 models the system by simple queues. Then, Section 3 develops an optimization algorithms. And finally, Sections 4, 5 and 6 test and analyses numerically some systems.

Subsequently we will try to interpret the tests and explain the behavior of the system. We will carry out several tests for various types of arrivals: (i) arrivals with constant rate, (ii) arrivals defined by an constant discrete distribution, (iii) arrivals specified by a variable discrete distribution over time, (iv) and arrivals modeled by discrete distribution obtained from real traffic traces.

2 QUEUE MODEL DEPICTION

A discrete time model is considered. The number of job arrivals is given by a discrete random variable. For any distribution X, $X[i]$ is the probability of item i. The following operators defined over distributions are needed to compute the system evolution:

- δ_v is the Dirac function with $v \in \mathbb{N}$, defined as:

$$\delta_v[i] = 1 \text{ if } i = v \text{ and } \delta_v[i] = 0 \text{ otherwise} \quad (1)$$

- $Z = X \otimes Y$ is the convolution of distributions. It is defined by:

$$Z[i] = \sum_{j=0}^{i} X[j] \times Y[i-j] \quad (2)$$

- $Y = SUB_v(X)$ is the distribution X translated by constant v. This function corresponds to a subtraction on the underlying random variable. It is defined by:

$$Y[i] = X[i-v] \text{ if } i > v > 0 \text{ and } Y[0] = \sum_{i=0}^{v} X[i] \quad (3)$$

- $Y = MIN_b(X)$ is the distribution X bounded by constant b. This function corresponds to a minimum on the underlying random variable. It is defined by:

$$Y[i] = X[i] \text{ if } i < b \text{ and } Y[b] = \sum_{i=b}^{\infty} X[i] \quad (4)$$

Let DC be a data center composed of Max identical servers working under the FIFO[1] discipline. DC receives jobs requesting the proposed service. The number of jobs served by one server in one slot is assumed to be constant and denoted by S. The job arrivals are sampled with a time interval equal to the slot duration. Thus, the queuing model is a batch arrival queue with constant services and finite capacity buffer B (buffer size). The number of jobs arriving to

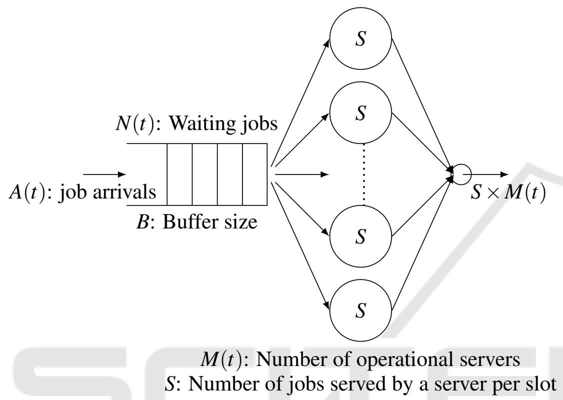

Figure 1: Illustration of the queuing model.

the data center during the t^{th} slot is denoted by $A(t)$ which is a discrete distribution. $N(t)$ denotes the distribution of the number of waiting jobs in the buffer (buffer length). The number of operational servers during the slot t is denoted by $M(t)$. Thus, $M(0)$ is the initial number of operational servers and Max is the maximal number of servers which can be operational. The distribution $N(t)$ can be computed by induction on t using the previous operators. As a discrete-time model is considered, 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 following equations give the distributions of the number of waiting jobs in the buffer and the lost jobs. For $t \geq 1$, we have:

$$N(t) = MIN_B(SUB_{S\times M(t)}(N(t-1) \otimes A(t))) \quad (5)$$

From now it is assumed that $N(0) = \delta_0$. The distribution of the number of the lost jobs during slot t is:

$$R(t) = SUB_{(S\times M(t)+B)}(N(t-1) \otimes A(t)) \quad (6)$$

[1]First In First Out.

Thus $A(t)$, $N(t)$ and $R(t)$ are distributions, and $M(t)$ is an integer value. It is assumed that the input arrivals are independent of the current queue state and the past of the arrival process. Under these assumptions, the model of the queue is a time-inhomogeneous Discrete Time Markov Chains. The problem we have to deal with is related to the nature of the arrival process. Typically, the job arrivals cannot be assumed to be stationary. The data center adapts to the fluctuation of the process by changing the number of servers associated with the queue, such a policy leads to a trade-off between the performance (i.e. waiting and loss probabilities) 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, distributions $N(t)$ and $R(t)$ are considered and then some decisions are taken according to a particular cost function. Let t be the current slot, the expectations $\mathbb{E}(N(t))$ and $\mathbb{E}(R(t))$ are considered.

Other parameters may be considered:

- the latency to switch on or off a server, which is a discrete non negative integer values, assumed to be zero in this study.
- the number of servers g that are switched on or off. It is a discrete positive integer value. During this analyses we assume that $g = 1$.

The energy consumption takes into account the number of operational servers. Each server consumes some units of energy per slot when a server is operational and it costs c_M monetary unit. The total energy used is the sum of units of energy consumed among the sample path.

QoS takes into account the number of waiting and lost jobs. Each waiting job costs c_N monetary unit per slot. A loss job cost c_R monetary unit:

Table 1: Considered costs.

Cost	Meaning
c_M	energetic cost for running one operational serer during one slot
c_N	waiting cost for one job over one slot
c_R	rejection cost for one losing one job

3 OPTIMIZATION ALGORITHM

In any optimization problem we have to *minimize a cost* or *a maximize gain*. In our problem we may both:

- Minimize the number of operational servers $M(t)$ to minimize the cost paid for electricity.

- Minimize the number of waiting and lost jobs, $N(t)$ and $R(t)$, to minimize the waiting time and the loss rate which increase the QoS.

This section explains our approach to optimize energy and QoS. Based on some elements of the system like the number of job arrivals, the number of waiting jobs, the number of rejected jobs and the number of servers, the method consists in turning on or off, each slot, an optimal number of servers to minimize energy consumption and maximize the QoS. So we will need to evaluate every slot the number of waiting and rejected jobs. Notice that parameters $M(t)$ and $N(t)$ & $R(t)$ are dependent and connected, gain on energy consumption leads to a degradation on the QoS and vice versa. Let's define the following *objective cost function*:

$$C(t) = c_M \times M(t) + c_N \times \mathbb{E}(N(t)) + c_R \times \mathbb{E}(R(t)) \quad (7)$$

Knowing the parameters of the system at slot $(t-1)$, our strategy consists to determine the best number of servers to be switched on, in slot t, in order to minimize the cost. To do so, every slot t: for each possible value of $M(t) \in \{0,1,\ldots,Max\}$, we compute $C(t)$ and then we returns the value of $M(t)$ that minimizes the cost $C(t)$.

Each slot, our algorithm evaluates all possible costs for switching on any possible number of servers, then it returns the number of servers for which the value of the cost is minimal.

Table 2: Example of optimization. Suppose $Max = 10$, $B = 15$, $S = 3$, $c_M = 11$, $c_N = 5$, $c_R = 0$ and for $t = 1$ we have: $\mathbb{E}(N(t-1)) = 5$ and $\mathbb{E}(A(t)) = 7$. In this case our algorithm chooses $M(t) = 4$ this value leads to the minimal cost.

$M(t)$	0	1	2	3	4	5	6	7	8	9	10
$C(t)$	60	56	52	48	44	55	66	77	88	99	110

Algorithm 1: Optimization for slot t.

```
Data: Max, S, B, N_{t-1}, A_t, c_M, c_N, c_R
Result: M_t

cost_min ← ∞
servers_min ← 0
for M ← 0 to Max (by a step of g) do
    N ← MIN_B(SUB_{S×M}(N_{t-1} ⊗ A_t))
    R ← SUB_{(S×M+B)}(N_{t-1} ⊗ A_t)
    cost← c_M × M + c_N × E(N) + c_R × E(R)
    if cost < cost_min then
        cost_min ← cost
        servers_min ← M
    end
end
M_t ← servers_min
```

As in (Bayati et al., 2015), taking into account the monotonicity of the system and using coupling detection algorithm (Sericola, 1999), numerical analysis becomes faster by avoiding unnecessary computations.

In the next sections, we will use this optimization algorithm to test, simulate, analyze and compare the evolution of cost (energy consumption and QoS) of a computer center for different types of job arrivals. We will consider mainly three types of arrivals: (i) Section 4 which analyzes constant arrival rate. (ii) Section 5 which considers arrivals modeled by an constant distribution. (iii) Section 6 which is addressed to arrivals modeled by a pairwise constant distribution.

4 CONSTANT ARRIVAL RATE

Let's first study the case where arrivals are modeled by a constant rate of job arrivals:

$$a \in \mathbb{R} : \forall t \ : \ A(t) = \delta_a \quad (8)$$

Given the variety of parameters that must be taken into account to test and analyze our optimization, we define the workload of the system as follows:

$$\rho = \frac{\mathbb{E}(A)}{Max \times S} = \frac{a}{Max \times S} \quad (9)$$

Table 3: Workload of system according to ρ.

ρ	0.2	0.5	0.7	0.9	1.2
	relaxed	moderate	comfortable	high	excessive

4.1 Optimization and Workload

Tests show that, our algorithm turns on, eventually, a number of servers proportional to the system workload (see Figure 2). Although our algorithm calcu-

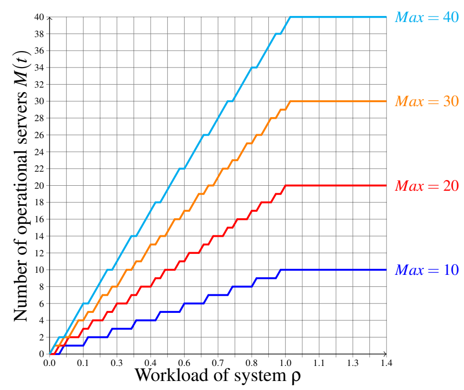

Figure 2: Relationship between the number of operational servers and the system workload.

lates from the beginning the best number of servers to be switched on, we observed that this number is kept the same during the whole observation period: $\forall t \;\; M(t) = \lceil \rho \times Max \rceil$. This formula is only valid if the workload of the system ρ is smaller than 1. Otherwise, if the workload is greater than 1, the best number of servers to be turned on will exceed the number of servers available, so the system turns on all available servers to be as close as possible to the optimal number of servers. Finally, the number of servers to be switched on by our algorithm in the case of a constant arrival rate is given by:

$$\forall t \;:\; M(t) = \min(\lceil \rho \times Max \rceil, Max) \qquad (10)$$

Note that the more the system has heavy workload, the more the number of operational servers is bigger. Note that the results of this subsection are true for costs c_M, c_N and c_R of the same order of magnitude. In the next subsection, we will show impact of varying the order of magnitude between these costs.

4.2 Cost Order Magnitude

For the rest, notice that the value $\frac{c_M}{S}$ gives the energy cost paid in a slot for the treatment of one single job by a server. The various tests we have done show that the behavior of our strategy against energy & QoS optimization depends on the values $\frac{c_M}{S}$, c_N, and c_R. We distinguish mainly three types of behavior:

Table 4: Impact of costs on the number of operational servers.

Condition	# operational servers	Behavior
$\frac{c_M}{S} >> c_N, c_R$	0	Lazy
$\frac{c_M}{S} << c_N, c_R$	Max	Fully active
$\frac{c_M}{S} \approx c_N, c_R$	$\min(\lceil \rho \times Max \rceil, Max)$	Proportional

- If cost $\frac{c_M}{S}$ is higher than c_N and c_R, the system prefers to turn off all servers because the cost of switching on a server to serve S jobs is more expensive than the cost of rejecting and/or keeping S jobs in the buffer. We say that the system is *lazy*.
- If cost $\frac{c_M}{S}$ is smaller than c_N and c_R, then the system prefers to turn on all the servers because the total cost of waiting and rejection is much more important than the cost of the switching on more servers. We say that the system is *fully active*.
- If costs $\frac{c_M}{s}$, c_N & c_R are close to each other, the system immediately turns on a number of servers proportional to the workload: $\min(\lceil \rho \times Max \rceil, Max)$.

To better clarify the results reported in the table 4 a closer analysis of the relationship between costs, system workload and the optimal number of servers is discussed in the following.

Theorem 1. *Assume that the buffer size is infinite. For any slot t:* $M(t) = \begin{cases} Max & \text{if } \frac{c_M}{S} < c_N \\ 0 & \text{otherwise.} \end{cases}$

Proof. Assume that B is ∞. This implies that all jobs will be accepted and no job will be rejected, which means that the number of loss jobs $R(t)$ is always zero: $(B \text{ is } \infty) \implies (\forall\, t : R(t) = 0)$. The total cost $C(t)$ will be:

$$\begin{aligned}
&= c_M \times M(t) + c_N \times \mathbb{E}(N) + c_R \times \mathbb{E}(R) \\
&= c_M \times M(t) + c_N \times \mathbb{E}(N) + c_R \times 0 \\
&= c_M \times M(t) + c_N \times \mathbb{E}(N) \\
&= c_M \times M(t) + c_N \times \mathbb{E}(MIN_B(SUB_{S\times M(t)}(N(t-1) \otimes A(t)))) \\
&= c_M \times M(t) + c_N \times \mathbb{E}(MIN_\infty(SUB_{S\times M(t)}(N(t-1) \otimes A(t)))) \\
&= c_M \times M(t) + c_N \times \mathbb{E}(SUB_{S\times M(t)}(N(t-1) \otimes A(t))).
\end{aligned}$$

As $A(t)$ is modeled in this section by a constant rate, we have: $\forall t \;:\; A(t) = \delta_a$, we deduce that:

$$C(t) = c_M \times M(t) + c_N \times \max\{0, N(t-1) + a - S \times M(t)\}.$$

We have two cases:

1. if $(N(t-1) + a - S \times M(t)) \leq 0$ then $C(t) = c_M \times M(t)$.
   In this case we must always choose $M(t) = 0$ to ensure a minimum total cost.
2. if $(N(t-1) + a - S \times M(t)) > 0$ then
   $$\begin{aligned} C(t) &= c_M \times M(t) + c_N \times (N(t-1) + a - S \times M(t)) \\ &= M(t) \times (c_M - S \times c_N) + c_N \times (N(t-1) + a). \end{aligned}$$
   It is clear that the right term $c_N \times (N(t-1) + a)$ is always a positive value. Thus minimizing the total cost requires the minimization of the left term $M(t) \times (c_M - S \times c_N)$. Thus, we are mainly dealing with two sub-cases:
   (a) If $\frac{c_M}{S} > c_N$ then $M(t) \times (c_M - S \times c_N)$ is negative, and choosing a maximum value of $M(t) = Max$ assures a minimum total cost.
   (b) If $\frac{c_M}{S} < c_N$ then $M(t) \times (c_M - S \times c_N)$ is positive, and choosing a zero value of $M(t) = 0$ provides a minimum total cost. □

5 CONSTANT ARRIVAL DISTRIBUTION

In this section we will study the case where arrivals are modeled by a single constant distribution that does not change over time: $\forall t \;:\; A(t) = D$.

As in the previous section we have implemented and tested our method based on the objective function to test and analyze the behavior of the system. We will consider three types of job arrival distributions with the same expectations (to keep nearly the same workload) but with different variances:

- *Uniform arrivals*: same probability everywhere in the distribution.
- *Middle arrivals*: high probabilities in the middle of the distribution and low probability at the extremities (leads to low variance).
- *Extremities arrivals*: low probability in the middle of the distribution and high probabilities at the extremities (leads to high variance).

Table 5 illustrates the three types of the considered distributions.

Table 5: Example of the considered distributions.

Arrivals Type	Distribution	$\mathbb{E}(A)$	$\mathbb{V}ar(A)$
Uniform	i: [0, 2, 4, 6, 8]; $A[i]$: [0.20, 0.20, 0.20, 0.20, 0.20]	4	8
High in middle	i: [0, 2, 4, 6, 8]; $A[i]$: [0.05, 0.20, 0.50, 0.20, 0.05]	4	2.4
High at extremities	i: [0, 2, 4, 6, 8]; $A[i]$: [0.40, 0.08, 0.04, 0.08, 0.40]	4	13.44

5.1 Workload Variation

In this subsection we will use the three types of arrivals we have previously introduced to analyze numerically the system whose parameters are described in Table 6. Figure 3 shows the evolution, over time, of the number of operational servers according to several workload values of ρ. Clearly, on the curves of this graph, we observe that the number of operational servers tends progressively to a value proportional to the system workload. The more the system has heavy workload, the more the needed operational servers is bigger. If the system is overwork-loaded the system eventually turn on all available servers. Thus, in general, the ultimate number of operational servers for a long term converges to: $\min\{Max, \lceil \rho \times Max \rceil\}$.

Table 6: Settings of the first numerical analysis.

Parameters	Value	Unit	Description
Max	150	servers	total number of servers
S	1	jobs/server	processing capacity of a server
B	1000	jobs	buffer size
$\mathbb{E}(A)$	20-150	jobs	average job arrivals
c_M	10	€	cost of energy needed by a server
c_N	12	€	cost of waiting a job
c_R	11	€	cost of rejecting a job

Furthermore, Figure 3 shows that for the same workload value ρ, the ultimate number of operational servers is reached quickly for middle arrivals (red curve), slowly for uniform arrivals (curve blue), and very slowly for extremities arrivals (orange curve).

Note that the results of this subsection were obtained for close costs (same order of magnitude). In the following subsection, we will conduct tests by varying costs c_M, c_N and c_R.

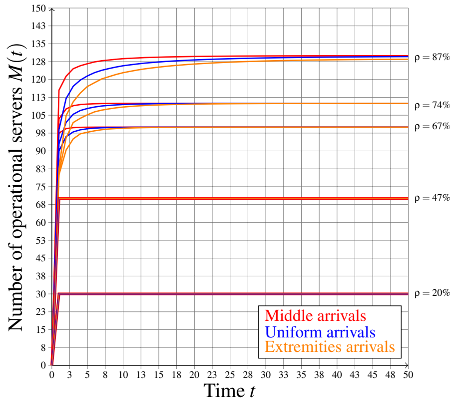

Figure 3: Evolution, over time, of the number of operational servers depending on workload values ρ according to several type of arrivals.

5.2 Cost Variation

In this subsection we will consider an analysis with a fixed job arrivals distribution $A(t)$ with a fixed expectation and a fixed variance, where varying the costs c_M, c_N & c_R. We analyze numerically the system whose parameters are described in Table 7.

Table 7: Settings of the second numerical analysis.

Parameters	Value	Unit	Description
Max	300	servers	total number of servers
S	1	jobs/server	processing capacity of a server
B	1000-9000	jobs	buffer size
$\mathbb{E}(A)$	155	jobs	average job arrivals
$\mathbb{V}ar(A)$	149.92^2	jobs	variance in job arrivals
ρ	52%	-	workload

5.2.1 Loss Cost

In this numerical analysis we have set a small value for the cost of energy consumption $\frac{c_M}{S}$ and even smaller value for the cost of waiting c_N but a high value for the cost of rejection c_R: $c_R > \frac{c_M}{S} > c_N$.

We observe that at the beginning and during a certain period, the system keeps all the servers switched off and holds jobs on waiting, because the cost of turning on a server is higher than keeping a job waiting. Thus the system prefers to put jobs on buffer. That said, after a certain latency period the buffer B becomes full and the system begins to reject jobs, and as the loss cost is too high compared to other costs, the system starts to turn on servers in order to reduce loss rate. Figure 4 illustrates this reaction latency phenomenon. Curves show that the larger the buffer B is, the longer latency period is. Therefore, the latency period before the response of the system can be approximated by: $\frac{B}{\mathbb{E}(A)}$. Now, fixing B the size of the buffer and varying c_R (while keeping $c_R > \frac{c_M}{S} > c_N$).

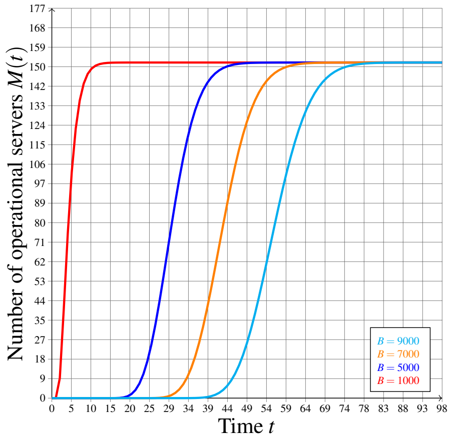

Figure 4: Latency phenomenon.

We observe that the system period latency is the same (because the value of B was fixed) for the different values of c_R. However, the higher the cost c_R is, the higher the system turns on servers. It is a completely natural reaction because the system tries to minimize the total cost, and as the loss cost c_R is highest, the system will turn on more servers to serve more jobs and emptying further the queue which allows a low loss rate. Figure 5 illustrates this phenomenon.

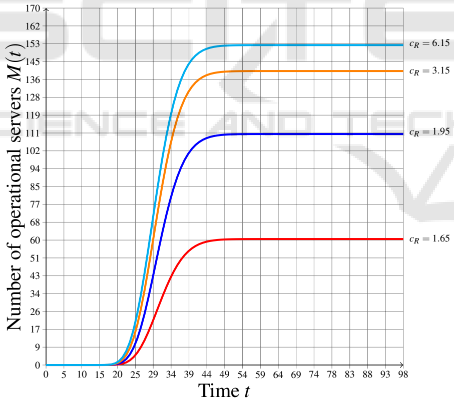

Figure 5: Evolution of the number of operational servers for different values of loss cost c_R.

5.2.2 Waiting Cost

In this subsection we have set a small value for the cost of energy consumption $\frac{c_M}{S}$ and a high value for the cost of waiting $c_N > \frac{c_M}{S}$ with any value for the loss cost. Figure 6 illustrates the result of this configuration. We observed that from the beginning, the system switch on a significant number of servers because the cost of energy is lower than the one of waiting. Thus the system prefers to turn on more servers to serve more jobs and avoid long waiting time. Eventually, we note that the loss cost is not involved in this configuration, because the system is fairly active and loss rate is negligible.

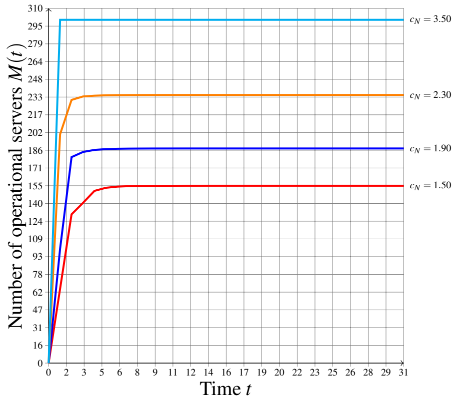

Figure 6: Evolution of the number of operational servers for different values of waiting cost c_N.

6 VARIABLE ARRIVAL DISTRIBUTION

In this section we will generalize our study extending it for arrivals modeled by a distribution that changes over time: $\exists t_1\ \exists t_2\ :\ A(t_1) \neq A(t_2)$.

6.1 Hourly Arrival Variation

In a real data center the arrivals jobs vary over the day. For example high rate arrivals between 8 a.m. and 4 p.m., low arrivals between 4 p.m. and midnight, and medium arrivals between midnight and 8 a.m. Figure 7 shows the results of analyzing numerically the system whose parameters are described in Table 9.

We observe that the system turns on a number of servers at the beginning of the day to treat arriving jobs. Then, it gradually increases the number of operational servers to treat the high arrival rate between

Table 8: Example of hourly variation of arrivals.

Arrivals rate	**Period**	$A(t)$	$\mathbb{E}(A)$
Medium arrivals	0 a.m.-8 a.m.	i [0 , 100 , 200 , 300 , 400 , 500] $A[i]$ [0.48 , 0.24 , 0.12 , 0.08 , 0.04 , 0.04]	100 jobs
High arrivals	8 a.m.-4 p.m.	i [0 , 100 , 200 , 300 , 400 , 500] $A[i]$ [0.17 , 0.04 , 0.13 , 0.17 , 0.22 , 0.27]	300 job
Low arrivals	4 p.m.-0 a.m.	i [0 , 100 , 200 , 300 , 400 , 500] $A[i]$ [0.73 , 0.15 , 0.07 , 0.03 , 0.01 , 0.01]	50 jobs

Table 9: Settings of the third numerical analysis.

Parameters	**Value**	**Unit**	**Description**
Max	400	servers	total number of servers
S	1	jobs/server	processing capacity of a server
B	3000	jobs	buffer size
c_M	7	€	cost of energy needed by a server
c_N	9	€	cost of waiting a job
c_R	8	€	cost of rejecting a job

8 a.m. and 4 p.m. Then from 4 p.m. it begins to turn off the servers and keeping only reduced number of operational servers to serve the low arrival rate until midnight. We clearly note the dynamic adaptation of the energy optimization system to the traffic variation.

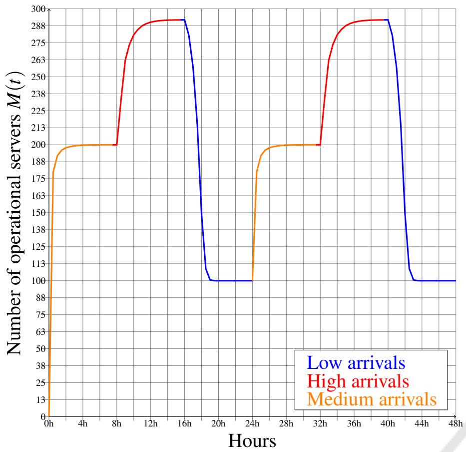

Figure 7: Evolution of the number of operational servers during two days.

6.2 Daily Arrival Variation

This section uses real traffic traces to model arrivals. We use the open *clusterdata-2011-2* trace (Wilkes, 2011; Reiss et al., 2011), and 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; we only use the column *timestamps* that refer to the arrival times of jobs expressed in μ-sec. This traffic trace is sampled with a sampling period equal to the slot duration. We consider frames of one minute to sample the trace and construct seven empirical distributions corresponding to arrivals during each day of the week. Such an assumption is consistent with the week evolution of job arrivals observed by long traces.

High arrivals rate is observed on Thursday, low arrivals rate on Saturday, Sunday and Monday, and medium arrivals rate during the rest of the week (see Table 10). These distributions have different statistical properties reflecting the fluctuation of traffic over the week (see Figure 8). For instance, we observe an average of 39 (resp. 58) jobs per minute during Sunday (resp. Thursday) with a standard deviation of 22 (resp. 38)(see Figure 9). Figure 10 shows the results of analyzing numerically the system whose parameters are described in Table 11.

Table 10: Example of daily variation of arrivals obtained from Google traces.

Day of week	**Arrivals rate**	$\mathbb{E}(A)$	$\sigma(A)$
Monday	Low	43 jobs	21
Tuesday	Medium	51 jobs	25
Wednesday	Medium	49 jobs	23
Thursday	High	58 jobs	38
Friday	Medium	53 jobs	25
Saturday	Low	41 jobs	24
Sunday	Low	39 jobs	22

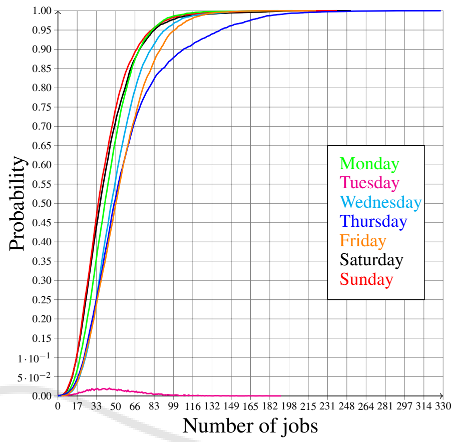

Figure 8: Cumulative distribution of $A(t)$ for days of the week.

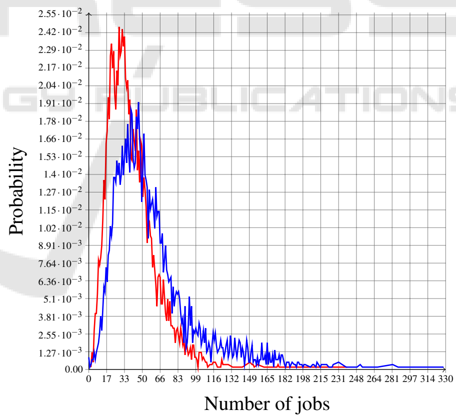

Figure 9: Distribution of $A(t)$ for Sunday (red curve) and Thursday (blue curve)

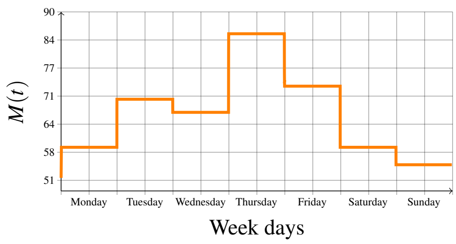

Figure 10: Evolution, over a week, of the number of operational servers. Our algorithm adapts the number of operational server according to the traffic variation.

Table 11: Settings of the last numerical analysis.

Parameters	Value	Unit	Description
Max	100	servers	total number of servers
S	1	jobs/server	processing capacity of a server
B	300	jobs	buffer size
c_M	7	€	cost of energy needed by a server
c_N	23	€	cost of waiting a job
c_R	29	€	cost of rejecting a job

7 CONCLUSION

In this paper we presented an optimization stochastic algorithm in order to manage energy consumption and QoS in a data center modeled by discrete time queue.

Every slot, the algorithm minimizes an objective function that combines the cost of energy and the cost of QoS, in order to change the number of operational servers according to traffic variation.

We show the ability of our algorithm to adapt dynamically to arrivals changes. Test were showed through various numerical analysis for several types of arrivals: (i) arrivals with constant rate, (ii) arrivals defined by an constant discrete distribution, (iii) arrivals specified by a variable discrete distribution over time, (iv) and arrivals modeled by discrete distribution obtained from Google real traffic traces. The system starts turning on servers progressively when high arrivals rate is detected. And turn off gradually the servers when arrivals rate becomes low.

Doing a closer analysis of the relationship between costs, workload and optimal number of operational servers is considered for future work to determine more accurate link between these parameters. We also intend to extend this study for the case in which, the number of served jobs in a slot by a server is defined by a distribution, the latency to switch on or off a server is not zero, and the servers are not identical in performance and energy consumption.

ACKNOWLEDGEMENTS

Special thanks to Vekris, D. and Dahmoune, M..

REFERENCES

Aidarov, K., Ezhilchelvan, P. D., and Mitrani, I. (2013). Energy-aware management of customer streams. *Electr. Notes Theor. Comput. Sci.*, 296:199–210.

Baliga, J., Ayre, R. W., Hinton, K., and Tucker, R. S. (2011). Green cloud computing: Balancing energy in processing, storage, and transport. *Proceedings of the IEEE*, 99(1):149–167.

Bayati, M., Dahmoune, M., Fourneau, J., Pekergin, N., and Vekris, D. (2015). A tool based on traffic traces and stochastic monotonicity to analyze data centers and their energy consumption. In *Valuetools '15: 9th international conference on Performance evaluation methodologies and tools*, page to appear. Acm.

Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M. Q., and Pentikousis, K. (2010). Energy-efficient cloud computing. *The computer journal*, 53(7):1045–1051.

Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., and Doyle, R. P. (2001). Managing energy and server resources in hosting centers. In *ACM SIGOPS Operating Systems Review*, volume 35, pages 103–116. ACM.

Grunwald, D., Morrey, III, C. B., Levis, P., Neufeld, M., and Farkas, K. I. (2000). 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. USENIX Association.

Koomey, J. (2011). 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.

Lee, Y. C. and Zomaya, A. Y. (2012). Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, 60(2):268–280.

Mazzucco, M. and Mitrani, I. (2012). Empirical evaluation of power saving policies for data centers. *SIGMETRICS Performance Evaluation Review*, 40(3):18–22.

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

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

Reiss, C., Wilkes, J., and Hellerstein, J. L. (2011). Google cluster-usage traces: format + schema. Technical report, Google Inc., Mountain View, CA, USA. Revised 2012.03.20.

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

Sericola, B. (1999). Availability analysis of repairable computer systems and stationarity detection. *IEEE Trans. Computers*, 48(11):1166–1172.

Wilkes, J. (2011). More Google cluster data. Google research blog. Posted at http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html.