Mathematical methods for analyzing performance and energy consumption in the cloud

M. Kandi *, F. Aït-Salaht[†] [‡], H. Castel-Taleb *, and E. Hyon [‡] [§]
* SAMOVAR, CNRS, Telecom SudParis, Université Paris-Saclay
9, rue Charles Fourier, 91011 Evry Cedex, France
Email: medmehdikandi@gmail.com, hind.castel@telecom-sudparis.eu
[†] Crest-Ensai, Rennes, France
Email: farah.ait-salaht@ensai.fr
[‡] Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 Paris UMR 7606,
4 place Jussieu 75005 Paris
[§] Université Paris Nanterre
Email: emmanuel.hyon@lip6.fr

***Abstract*—We propose in this paper to evaluate using mathematical methods the performance and the energy consumption of cloud system. We consider for the analysis an hysteresis queueing system, which is characterized by forward and backward thresholds for activation and deactivation of block of servers representing a set of VMs (Virtual Machines). The system is represented by a complex Markov Chain which is difficult to analyze when the size of the system is huge. For this case, we propose three different mathematical methods for computing the steady-state probability distribution: the SCA (Stochastic Complement Analysis) method in order to aggregate the state space, Level Dependent Quasi Birth and Death (LDQBD) method, and the balance equations that allows to derive exact formulas for the steady-state probability distribution. We compute both performance and energy consumption measures and we define an overall cost taking into account both aspects. Then these three methods are compared from their computation time. Moreover, we analyze the impact of some parameters as the thresholds, and the arrival rate on the behavior of the system.**

I. INTRODUCTION

Recently, Cloud computing has changed the way people do computing and manage information, since a pool of virtualized, dynamically-scalable computing functions and services are made accessible over the internet to remote users in an on-demand fashion. With virtualization, the cloud providers can adapt the virtual machines according to the demand, and service providers can provide resources in a cost-effective manner by consolidating VMs onto fewer physical resources when system load is low, and quickly scale up workloads to more physical resources when system load is high.

Finding the policy that tailors resources to demand is a crucial point in such systems. Multi server queuing models [1] or server farms models [2], [3] have been proposed since they are well suited to represent the behavior of a data center and its dynamicity as well as to compute its performance metrics. In such models the servers can be activated and deactivated either server by server or by blocks of servers according to the intensity of user demand. A special case of model is the multi-server threshold based-queueing system with hysteresis policy [4], in which activations and deactivations are governed by sequences of forward and reverse different thresholds where each time the workload (or the number of customers in the system) reaches a forward or reverse thresholds only one server (here one VM) is activated or deactivated. We propose in this paper to extend the current state of art and to couple the advantages of the activation by block and the advantages of hysteresis policy by considering a multi-server system with hysteresis in which activation/deactivation are made by block (ie. activate or deactivate at same time a set of VMs). This to our knowledge has never been considered and studied previously in the literature. Moreover, in this paper, one objective is to consider both performance and energy consumption in order to propose a tradeoff between them.

The assessment of both performance and energy consumption of cloud system driven by hysteresis policies requires the computation of the expected measures. But we face up a computational complexity problem since cloud systems are often defined on very large state spaces which makes their exact analysis very cumbersome or even impossible. Under some considerations and assumptions, this problem has been already studied in the literature and different resolution methods have been presented to compute efficiently (in exact and less complex manner) the performance measures of the system. Among the most significant works, we can mention the work of Lui and Golubchik [5] which solves the model, using the concept of stochastic complementation. This method is based on partitioning the state space in disjoint sets in order to aggregate the Markov chain. The main advantage of this method is to obtain exact performance results, with reduced execution times. In [6], Le Ny et al. propose to compute the steady-state probabilities of a heterogeneous multi-server threshold queue with hysteresis by using a closed-form solution. Finally in [7], the authors analyze the system through stochastic bounding theory and define accurate bounds on performance measures. However, in these works, we emphasize that they considered only the case where one VM is activated (resp. deactivated) according to the demand and the threshold sequences.

In this work we propose to investigate a general case, and suppose that the activation and the deactivation can be done by blocks. We establish and present in this paper some proved analysis and resolution methods. First method consists to adapt and extend the SCA (Stochastic Complement Analysis) method, proposed by Lui et al. [5]. Based on aggregating state space, we propose to couple this method with a numerical resolution technique (such as GTH method [8]) and relieve all the constraints imposed by Lui et al. in their previous work, on the sequences of thresholds. The aim of this approach is to improve the efficiency. The second investigated method is an analytical approach based on adapting the balance equations method of [6] and get exact formulas for the steady-state probability distribution. We also extend the method by relaxing the former assumptions on the thresholds that [6] made. Moreover, for this method, we give an algorithmic solution to compute the steady state probability vector. At last, a numerical approach based on Level Dependent Quasi Birth and Death (LDQBD) method is presented in details. This last one is the closest of usual resolution methods and allows to know the improvement brought by the two previous ones. We obtain numerical results even for Markov chains with a large state space, and we establish an overall cost taking into account both performance and energy consumption. Moreover, as we consider in this model more general assumptions for the thresholds, then we can see in details the impact of their values on performance and energy consumption.

The paper is organized as follows: next (section II), we describe the cloud system and present the considered queueing model for the analysis. In section III, we detail the different methods to solve the model and compute the steady state probability vector. While IV presents the formulation used to express the expected costs in terms of performances and energetic consumption for the model, section V presents numerical results of performance and energy consumption measures. Finally, achieved results are discussed in the conclusion and comments about further research issues are given.

II. Cloud system description

We analyse a cloud system composed by a set of Virtual Machines (VMs). We assume that the job requests arrive at the system following a Poisson process with rate λ, and are enqueued in a finite queue with capacity B. An arriving request can be rejected if it finds the buffer full. We model this system using a multi-server queue, with C homogeneous servers representing the VMs. The service time of each VM is exponential with mean rate μ. In order to represent the dynamicity of resource provisioning, the VMs are activated and deactivated according to the system occupancy. Actually, the buffer management is governed by thresholds vectors corresponding to the number of customer waiting in the system, which controls the operation of activating and deactivating the VMs. We suppose the case where the VMs are activated or deactivated by block, which means that several VMs can be simultaneously activated or deactivated.

We define K functioning levels, where each level corresponds to a given number of active servers. The number of active servers at level k is fixed and denoted by S_k, where $S_1 \leq S_2 \leq \ldots \leq S_K = C$. We suppose that $S_1 \geq 1$, so we have at least one active server by assumption. The transition from functioning level k to level $k+1$ allows to allocate (turn on) one or more additional servers, going from S_k to S_{k+1} active servers, while the transition from level k to level $k-1$ allows to remove (turn off) one or more active servers, going from S_k to S_{k-1} active servers. Depending on the system occupancy, we transit from the level k to level $k+1$ when the occupancy in the system exceeds a threshold F_k, and from level k to level $k-1$ when the occupancy in the system falls below a threshold R_{k-1}. The model is then characterized by activation thresholds $F = (F_1, F_2, \ldots, F_{K-1})$ (called also forward thresholds), and deactivation thresholds $R = (R_1, R_2, \ldots, R_{K-1})$ (called also reverse thresholds). These thresholds are fixed and can not be modified during the system works. We furthermore assume that $F_1 < F_2 < \ldots < F_{K-1}$, that $R_1 < R_2 < \ldots < R_{K-1}$ and that $R_k < F_k, \forall k, 1 \leq k \leq K-1$. We suppose here that server deactivations occur at the end of the service, and when multiple servers are deactivated at the same times, all the customers who have not completed their service return to the queue.

The underlying model is described by the Continuous-Time Markov Chains (CTMCs), denoted $\{X(t)\}_{t \geq 0}$. A state is represented by a couple (m, k) such that m is the number of customers in the system and k is the functioning level. The state space is denoted by A and is given as follows:

$$\begin{aligned} A = \{(m,\, k) \,|\, & 0 \leq m \leq F_1, \text{ if } k = 1; \\ & R_{k-1} + 1 \leq m \leq F_k, \text{ s.t. } 1 < k < K; \\ & R_{K-1} + 1 \leq m \leq B, \text{ if } k = K; \} \end{aligned}$$

Recalling that S_k represents the number of active servers at level k, the transitions between states follows:

$$\begin{aligned} (m,\, k) \;\; \rightarrow \;\; & (\min\{B, m+1\},\, k) \text{ with rate } \lambda\,, \text{ if } m < F_k; \\ \rightarrow \;\; & (\min\{B, m+1\},\, \min\{K,\, k+1\}) \\ & \text{with rate } \lambda, \text{ if } m = F_k; \\ \rightarrow \;\; & (\max\{0, m-1\},\, k), \text{with rate } \min\{S_k,\, m\} \cdot \mu, \\ & \text{if } m > R_{k-1} + 1; \\ \rightarrow \;\; & (\max\{0, m-1\},\, \max\{1, k-1\}) \\ & \text{with rate } \min\{S_k,\, m\} \cdot \mu, \\ & \text{if } m = R_{k-1} + 1\,. \end{aligned}$$

An example of the transitions is given Figure 1.

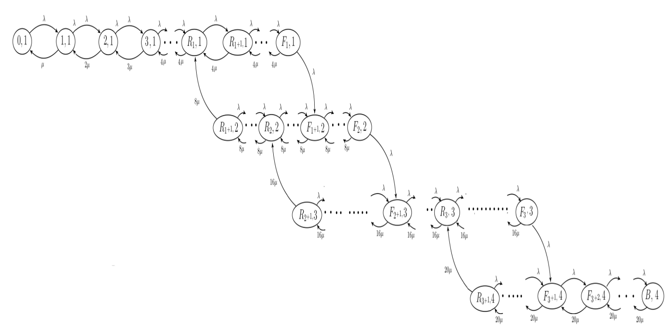

Fig. 1. Transition structure for $K = 4$, $S_1 = 4$, $S_2 = 8$, $S_3 = 16$, $S_4 = 20$, $R_1 + 1 \geq 8$, $R_2 + 1 \geq 16$ and $R_3 + 1 \geq 20$.

III. Resolution approaches

In order to compute the performance measures of the presented model, we expose hereafter some techniques to solve the CTMC and compute the steady state probability vector. Three resolution methods, either numerical or analytical,have been developed for the model and a comparison in terms of execution times have been also done for these three proved method (see Section V).

A. Stochastic Complement Analysis (SCA)

To solve the $\{X(t)\}_{t\geq 0}$ Markov chain, the first approach consists to aggregate the underlying chain and use a numerical method to compute the steady state distribution. This approach has been proposed by Lui et al. [5] and works as follows. First, we aggregate the state space of the underlying Markov chain by partitioning the set A into disjoint subsets. The number of derived subsets depends on the number of functioning level. From each subset, we define a corresponding Markov chain. These derived Markov chains are defined on reduced state spaces which makes their analysis less complex. The resolution of each Markov chain defines a conditional steady state probabilities. By applying the state aggregation technique, each subset is now represented by a single state, and an aggregated process is defined. A resolution of this aggregated process is performed, i.e., the probabilities of the system being in any given set are computed. We note that the compute of the steady state probabilities of the derived Markov chains is performed using a numerical rsolution method (see [8]). Lastly, a disaggregation technique is applied to compute the individual steady state probabilities for the original Markov process.

In the following, we present an important theorem stated by Lui et al. in their article [5].

Theorem 1: Given an irreducible Markov process with state space A, let us partition this state space into two disjoint set A_1 and A_2. Then, the transition rate matrix (denoted by Q) is given as follows:

$$Q = \begin{pmatrix} Q_{A_1A_1} & Q_{A_1A_2} \\ Q_{A_2A_1} & Q_{A_2A_2} \end{pmatrix},$$

where $Q_{i,j}$ is the transition rate sub-matrix corresponding to transitions from partition i to partition j.

Based on this theorem and under some restrictions, Lui et al. have investigated a simple version of multi-server threshold-based queueing system with hysteresis. The author have considered the case where we activate (resp. deactivate) only one server, and under constraint $R_k < F_{k-1}\ \forall k$, they derived a closed-form solution for the steady state probability vector. Here, without any restrictions, we propose to give a general formulation for the multi-server threshold-based queueing system with hysteresis. Indeed in this paper, we propose to extend the work of Lui et al. considering, on the one hand, that the activation and the deactivation are made by block and on the other hand, by not specifying any constraints on the sequences of forward and reverse threshold (i.e., consider both cases $R_k < F_{k-1}$ and $R_k \geq F_{k-1}$ for all k). We describe our approach below.

Given the $\{X_t\}_{t\geq 0}$ Markov chain with state space A, we first partition the state space A into K distinct sets denoted A_k, $1 \leq k \leq K$, where:

$$A_k = \{(i,\,j)|(i,\,j) \in A \text{ and } j = k\};\ \forall\, k = 1...K \quad (1)$$

The set A_k contains the states belonging the level k.

Let $\{X_k(t)\}_{t\geq 0}$ be a Markov chain defined on state space $A_k, \forall k = 1...K$. We denote by π_k the steady state probabilities of X_k. The states transition in $\{X_k(t)\}_{t\geq 0}$ are identical to those appearing at level k in the original process $\{X(t)\}_{t\geq 0}$, except some additional modifications, specific to each chain, that are set out below:

For $k = 1$, we add a transition from state $(F_1, 1)$ to state $(R_1, 1)$ with a rate λ.

For $k \in \{2, \ldots, K-1\}$, we add a transition from state $(F_k,\,k)$ to state $(R_k,\,k)$, with a rate λ, and we add transition from $(R_{k-1}+1,\,k)$ to state $(F_{k-1}+1,,)$, with a rate $(\min\{R_{k-1}+1, S_k\}\cdot\mu)$.

For $m = K$, we add a transition from state $(R_{K-1}+1,\,K)$ to state $(F_{K-1}+1,\,K)$, with a rate $\min\{R_{K-1}+1, S_K\}\cdot\mu$.

For each CTMC $\{X_k(t)\}_{t\geq 0}$, a conditional state probability vector π_k is computed using a numerical resolution technique [8] (GTH or Power method for instance). We notice that the aggregated process that brings all aggregate states is a simple birth and death process, with:

- λ_k=$\lambda \cdot \pi_i(F_k), \quad \forall k = 1\cdots K-1$
- μ_k=$\mu\cdot\min(R_{k-1}$+$1, S_k)\cdot\pi_k(R_{k-1}$+$1), \ \forall k = 2\cdots K.$

We denote by $\overline{\pi}$ the steady state probabilities of this aggregated process.

Finally, the steady state probability vector π of $\{X(t)\}_{t\geq 0}$ is can be thus expressed as follows: for each individual state $(i,j) \in A$, we have: $\pi(i,\,j) = \pi_j(i)\,\overline{\pi}(j)$ where $(i,\,j) \in A_j$.

B. Level Dependant Quasi Birth and Death Process

The particular form of the generator of $\{X(t)\}_{t\geq 0}$ suggest us to use the Quasi Birth and Death (QBD) processes in order to benefits of the numerous existing numerical methods to solve them [9]. For short, a QBD process is a stochastic process in which the state space is two dimensionals and can be decomposed in disjoint sets such that transition may only occur inside a set or occur towards only two other sets. This results in a generator with a tridiagonal form (as the birth and death process) in which the terms on the diagonals are matrices. When the matrices are identical for each level it is said *level independant* but when the matrices are different the QBD is said *level dependant* (LDQBD).

Let us define $Q_{k,k'}(i,j)$ that denotes the i-th line and j-th column element of matrix $Q_{k,k'}$. We have

Proposition 1: The Markov Chain $\{X_t\}_{t\geq 0}$ is a Level Dependant QBD with K levels, corresponding to the functioning levels. Its generator Q can be decomposed in :

$$Q = \begin{pmatrix} Q_{1,1} & Q_{12,} & & & & \\ Q_{2,1} & Q_{2,2} & Q_{2,3} & & & \\ & Q_{3,2} & Q_{3,3} & Q_{3,4} & & \\ & & \ddots & \ddots & \ddots & \\ & & & Q_{K-1,K-2} & Q_{K-1,K-1} & Q_{K-1,K} \\ & & & & Q_{K,K-1} & Q_{K,K} \end{pmatrix}.$$

For all k, the inner matrices $Q_{k,k-1}$, $Q_{k,k}$ and $Q_{k,k+1}$ are respectively of dimension $d_k \times d_{k-1}$, $d_k \times d_k$ and $d_k \times d_{k+1}$, letting $d_k = F_k - R_{k-1}$, $R_0 = -1$ and $F_K = B$.

For $k = 1$ we have:

$$Q_{1,1}(i,j) = \begin{cases} \lambda & \text{if } j = i+1 \\ \mu \min\{S_1, i\} & \text{if } j = i-1 \\ -\lambda & \text{if } i = 1 \text{ and } j = 1 \\ -(\lambda + \mu \min\{S_1, i\}) & \text{if } i = j \text{ and } i \neq 1 \\ 0 & \text{otherwise} \end{cases},$$

and

$$Q_{1,2}(i,j) = \begin{cases} \lambda & \text{if } i = d_1 \text{ and } j = F_1 - R_1 + 1 \\ 0 & \text{otherwise} \end{cases}.$$

For $k \in 2, ...K-1$, we get:

$$Q_{k,k-1}(i,j) = \begin{cases} \mu \min\{S_k, R_{k-1}+1\} & \text{if } i=1 \text{ and } j=R_{k-1}-R_{k-2} \\ 0 & \text{otherwise} \end{cases},$$

also

$$Q_{k,k}(i,j) = \begin{cases} \lambda & \text{if } j = i+1 \\ \mu \min\{S_k, R_{k-1}+i\} & \text{if } j = i-1 \\ -(\lambda + \mu \min\{S_k, R_{k-1}+i\}) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases},$$

and

$$Q_{k,k+1}(i,j) = \begin{cases} \lambda & \text{if } i = d_k \text{ and } j = F_k - R_k + 1 \\ 0 & \text{otherwise} \end{cases}.$$

Finally for $k = K$, it follows

$$Q_{K,K-1}(i,j) = \begin{cases} \mu \min\{S_K, R_{K-1}+1\} & \text{if } i=1 \text{ and } j=R_{K-1}-R_{K-2} \\ 0 & \text{otherwise} \end{cases},$$

and

$$Q_{K,K}(i,j) = \begin{cases} \lambda & \text{if } j = i+1 \\ \mu \min\{S_K, R_{K-1}+j\} & \text{if } j = i-1 \\ -(\lambda + \mu \min\{S_K, R_{K-1}+j\}) & \text{if } i = j \text{ and } j \neq d_K \\ -\mu \min\{S_K, R_{K-1}+j\} & \text{if } i = j = d_K \\ 0 & \text{otherwise} \end{cases}.$$

Proof: Let i, j be the coordinate on line i and column j of the generator Q. It records the transition from the ith state to the jth state. Let us describe now which is the ith state. Recall that, by definition, the number of customers on a given level k can vary from $R_{k-1}+1$ to F_k. Let k be the number such that $\sum_{l=1}^{k-1} d_l < i$ and such that $\sum_{l=1}^{k} d_l \geq i$ then the functionning level of the ith state is k and the number of customers is $m = R_{k-1} + \left(i - \sum_{l=1}^{k-1} d_l\right)$.

We detail now the possible transitions from state (m,k). We first study the transition inside a level. We can jump in $\min\{B, m+1\}$ with rate λ and then $j = i+1$. We can also jump in state $\max\{0, m-1\}$ with rate $\min\{S_k, m\}$ and then $j = i-1$. At last, it can be noticed that, by construction of the matrices, the coordinate (i,i) of the generator Q is the coordinate (i', i') of the matrix $Q_{k,k}$ with $i' = m - R_{k-1}$.

For transitions to the upper level. If $m = F_k$ then we can only jump to $(\min\{F_k+1, B\}, \max\{K, k+1\})$ with rate λ and in this case $j = \sum_{l=1}^{k} d_l + F_k - R_k + 1$. Furthermore, the coordinate $(i, i + F_k - R_k + 1)$ with $i = \sum_{l=1}^{k} d_l$ of the generator Q is the coordinate $(d_k, F_k - R_k + 1)$ of the matrix $Q_{k,k+1}$.

For transitions to the lower level. If $m = R_{k-1}+1$ then we can only jump to $(\max\{m-1, 0\}, \max\{1, k-1\})$ with rate $\mu \min\{S_k, m\}$ and in this case $j = \sum_{l=1}^{k-2} d_l + R_{k-1} - R_{k-2}$. Furthermore, the coordinate $((\sum_{l=1}^{k-1} d_l)+1, \sum_{l=1}^{k-2} d_l + R_{k-1} - R_{k-2})$ of the generator Q is the coordinate $(1, R_{k-1} - R_{k-2})$ of the matrix $Q_{k,k-1}$.

There is no other possible transitions, the terms at the bounds of the state space should be adapted and the terms on the diagonal follow. Thus the generator has a tridiagonal form. and for a given k the matrices $Q_{k,k-1}$ (resp $Q_{k,k}$, $Q_{k,k+1}$) records the events associated with a decrease of the level (resp staying in same level and an increase of the level). ■

Numerically solving QBD is an hard computationnal task requiring to solve matrix equations and is often based on matrix geometric methods [9], [10] or kernel methods [11]. This is even more the case for LDQBD. Here among the numerical existing methods to solve them, this one proposed in [12] is used since it is shown that this method is efficient and numerically stable.

C. Closed form solution using balance equations

We give a closed form for the steady state probability using balance equations and cuts on the state space. As in [6], we compute the probabilities level by level, where one level corresponds to a given number of active servers. The relevance of our work is that we can take more general cases than [6] for the thresholds, namely $R_k \leq F_{k-1}$, and $R_k > F_{k-1}$ for each level $2 \leq k \leq K$.

The probabilities are computed level by level, from level 1 to level K. For states of level 1, the steady state probabilities are expressed in terms of $\pi(0,1)$. For a level $k \in \{2 \dots K\}$, the steady-state probability of the first state of the level: $\pi(R_{k-1}+1, k)$ is expressed in terms of the last state of the precedent level $\pi(F_{k-1}+1, k-1)$ which has been already computed. After that, the other probabilities of the level k are computed in terms of $\pi(R_{k-1}+1, k)$. So, it results that all the probabilities are computed in terms of $\pi(0,1)$. At the end, from the normalizing condition, $\pi(0,1)$ can be derived. This method to compute π is detailled by Algorithm 1.

We suppose that for each level $2 \leq k \leq K$, $R_{k-1}+1 \geq S_k$, so the service rate for each level is $\min(R_{k-1}+1, S_k) = S_k\mu$. The level one is a particular case, as the service rate depends on the number of customers in the system: so for a state $(m,1)$, where $1 \leq m < S_1$, the service rate is $m\mu$ and if $m \geq S_1$ it is $S_1\mu$. From now on, for any $k \in \{1 \dots K\}$, we denote by $\mu_k = \mu S_k$ and by $\rho_k = \frac{\lambda}{\mu_k}$. We consider also that $\rho = \frac{\lambda}{\mu}$.

1) Analysis of level 1: The following lemma gives the steady-state probabilities for level 1.

Lemma 1: We have three cases:

Algorithm 1 Compute π

```
{probabilities of level 1}
for 1 ≤ m ≤ F_1 do
  if m < S_1 then
    compute π(m,1) from Equation (2)
  else if S_1 < m ≤ R_1 then
    compute π(m,1) from Equation (3)
  else
    compute π(m,1) from Equation (4)
  end if
end for
{probabilities of level k }
for 2 ≤ k ≤ K−1 do
  compute π(R_{k−1}+1, k) from Equation (10)
  compute π(R_k+1, k+1) from Equation (28)
  if R_k ≤ F_{k−1} then
    for R_{k−1}+2 ≤ m ≤ F_k do
      if R_{k−1}+2 ≤ m ≤ R_k then
        compute π(m,k) from Equation (29)
      else if R_k+1 ≤ m ≤ F_{k−1}+1 then
        compute π(m,k) from Equation (30)
      else
        compute π(m,k) from Equation (31)
      end if
    end for
  else
    for R_{k−1}+2 ≤ m ≤ F_k do
      if R_{k−1}+2 ≤ m ≤ F_{k−1}+1 then
        Compute π(m,k) from Equation (11)
      else if F_{k−1}+2 ≤ m ≤ R_k then
        Compute π(m,k) from Equation (12)
      else
        Compute π(m,k) from Equation (13)
      end if
    end for
  end if
end for
{probabilities of level K }
compute π(R_{K−1}+1, K) from Equation (10)
for R_{k−1}+2 ≤ m ≤ B do
  if R_{K−1}+2 ≤ m ≤ F_{K−1}+1 then
    Compute π(m,k) from Equation (32)
  else
    Compute π(m,k) from Equation (33)
  end if
end for
```

- if $0 \leq m \leq S_1$:

$$\pi(m,1) = \frac{\rho^m}{m!}\pi(0,1)\,, \tag{2}$$

- if $S_1 < m \leq R_1$:

$$\pi(m,1) = \rho_1^{m-S_1}\frac{\rho^{S_1}}{S_1!}\pi(0,1)\,, \tag{3}$$

- if $R_1 + 1 \leq m \leq F_1$:

$$\pi(m,1) = \frac{\rho^{S_1}}{S_1!}(\rho_1^{m-S_1} - \frac{\rho_1^{F_1-S_1+1}(1-\rho_1^{m-R_1})}{1-\rho_1^{F_1-R_1+1}})\pi(0,1). \tag{4}$$

Proof: In the level one, it is logical to suppose that $R_1 \geq S_1$. We propose to make cuts in the Markov chain diagram around sets $\{(0,1),\ldots,(m,1)\}$. Same kinds of partitions appear in [3]. For $0 \leq m < R_1$, we have the following evolution equation: $\mu(m+1)\pi(m+1,1) = \lambda\pi(m,1)$. We then deduce and prove, Equation (2) when $0 \leq m \leq S_1$ and Equation (3) when $S_1 < m \leq R_1$.

For $R_1 \leq m \leq F_1 - 1$, making cuts around the sets $\{(0,1),\ldots,(R_1,1),\ldots,(m,1)\}$ allows us to derive the following balance equations: $\mu_1\pi(m+1,1) + \mu_2\pi(R_1+1,2) = \lambda\pi(m,1)$. So, if $R_1 + 1 \leq m \leq F_1$, we obtain:

$$\pi(m,1) = \rho_1^{m-R_1}\pi(R_1,1) - \pi(R_1+1,2)\frac{\rho_1}{\rho_2}\sum_{k=0}^{m-R_1-1}\rho_1^k \tag{5}$$

and we deduce that:

$$\pi(m,1) = \rho_1^{m-R_1}\pi(R_1,1) - \pi(R_1+1,2)\frac{\rho_1}{\rho_2}\frac{1-\rho_1^{m-R_1}}{1-\rho_1}\,. \tag{6}$$

Replacing m by F_1 in Equation (6) yields to:

$$\pi(F_1,1) = \rho_1^{F_1-R_1}\pi(R_1,1) - \pi(R_1+1,2)\frac{\rho_1}{\rho_2}\frac{1-\rho_1^{F_1-R_1}}{1-\rho_1}\,. \tag{7}$$

If we make a cut between states of level 1 and states of other levels, then we obtain the following evolution equation: $\lambda\pi(F_1,1) = \mu_2\pi(R_1+1,2)$. Thus we deduce that: $\pi(F_1,1) = \frac{1}{\rho_2}\pi(R_1+1,2)$. From Eq. (3), with $m = R_1$ it follows:

$$\pi(R_1,1) = \rho_1^{R_1-S_1}\frac{\rho^{S_1}}{S_1!}\pi(0,1)\,. \tag{8}$$

Thus it comes:

$$\pi(R_1+1,2) = \frac{\rho_2\rho_1^{F_1-S_1}(1-\rho_1)}{1-\rho_1^{F_1-R_1+1}}\frac{\rho^{S_1}}{S_1!}\pi(0,1)\,. \tag{9}$$

Then using equations (8) and (9) in Equation (6), we prove Equation (4) for $R_1 + 1 \leq m \leq F_1$. ∎

2) Analysis of level k: We consider now a level k such that $2 \leq k \leq K-1$. If we consider the cut of the state space between states of level $k-1$ and states of level k, we have the following evolution equation: $\pi(F_{k-1},k-1)\lambda = \pi(R_{k-1}+1,k)\mu_k$, which is equivalent to:

$$\pi(R_{k-1}+1,k) = \rho_k\pi(F_{k-1},k-1)\,. \tag{10}$$

Now, we compute the probabilities of each level, by expressing each of them with repsect to $\pi(R_{k-1}+1,k)$. According to the threshold values, two cases should be considered: either $R_k \leq F_{k-1}$ or $R_k > F_{k-1}$. Since the case $R_k \leq F_{k-1}$, has been considered in [6], we only recall main equations. We present now the case $R_k > F_{k-1}$ in details.

a) Case 1: $R_k > F_{k-1}$: We have

Lemma 2: For any level $k \in \{2\ldots K\}$ three cases occur:

- if $R_{k-1} + 2 \leq m \leq F_{k-1} + 1$:

$$\pi(m,k) = \frac{1-\rho^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k)\,, \tag{11}$$

- if $F_{k-1} + 2 \leq m \leq R_k$:

$$\pi(m,k) = \frac{\rho_k^{m-F_{k-1}-1} - \rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k)\,, \tag{12}$$

• if $R_k + 1 \leq m \leq F_k$:

$$\pi(m,k) = \frac{\rho_k^{m-F_{k-1}-1} - \rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) - \frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\,. \quad (13)$$

Proof: If we consider cuts on the state space around the sets $\{(R_{k-1}+1,k),\ldots,(m,k)\}$ for $R_{k-1}+1 \leq m \leq F_{k-1}$, then we obtain the following equation: $\lambda\pi(m,k) + \mu_k\pi(R_{k-1}+1,k) = \mu_k\pi(m+1,k)$. So, if $R_{k-1}+2 \leq m \leq F_{k-1}+1$, it comes:

$$\pi(m,k) = \rho_k\pi(m-1,k) + \pi(R_{k-1}+1,k)\,, \quad (14)$$

and we prove Equation (11) with induction on Equation (14).

If we consider cuts on the state space around sets $\{(R_{k-1}+1,k),\ldots,(F_{k-1}+1,k),\ldots,(m,k)\}$, for $F_{k-1}+1 \leq m \leq R_k - 1$, then we obtain the following balance equation: $\lambda\pi(m,k) + \mu_k\pi(R_{k-1}+1,k) = \mu_k\pi(m+1,k)$. So, for $F_{k-1}+2 \leq m \leq R_k$, we get :

$$\pi(m,k) = \rho_k\pi(m-1,k) - \rho_k\pi(F_{k-1},k-1) + \pi(R_{k-1}+1,k)\,. \quad (15)$$

Its follows from the preceeding equation:

$$\pi(m,k) = \rho_k^{m-F_{k-1}-1}\pi(F_{k-1}+1,k) + \sum_{i=0}^{m-F_{k-1}-2}\rho_k^i\pi(R_{k-1}+1,k) - \sum_{i=1}^{m-F_{k-1}-1}\rho_k^i\pi(F_{k-1},k-1)\,. \quad (16)$$

Letting $m = F_{k-1}+1$ in Equation (11) yields to:

$$\pi(F_{k-1}+1,k) = \frac{1-\rho^{F_{k-1}+1-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k)\,. \quad (17)$$

Thus from equations (17), (10), and (16), we obtain:

$$\pi(m,k) = \frac{1-\rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) - \sum_{i=1}^{m-F_{k-1}-1}\rho_k^i\frac{1}{\rho_k}\pi(R_{k-1}+1,k)\,. \quad (18)$$

Hence Equation (12) is proved for $F_{k-1}+2 \leq m \leq R_k$.

For $R_k \leq m \leq F_k - 1$, if we make cuts on sets of states $\{(R_{k-1}+1,k),\ldots,(R_k,k),\ldots,(m,k)\}$, then we obtain the following balance equations:

$$\pi(m,k)\lambda + \pi(R_{k-1}+1,k)\mu_k = \mu_k\pi(m+1,k) + \mu_{k+1}\pi(R_{k-1}+1,k) - \mu_{k+1}\pi(R_k+1,k+1) + \lambda\pi(F_{k-1},k-1) \quad (19)$$

For $R_k \leq m \leq F_k - 1$, from Equation (19), we have:

$$\pi(m+1,k) = \rho_k\pi(m,k) + \pi(R_{k-1}+1,k) - \frac{\rho_k}{\rho_{k+1}}\pi(R_k+1,k+1) - \rho_k\pi(F_{k-1},k-1)\,, \quad (20)$$

and from Equation (10):

$$\pi(F_{k-1},k-1) = \frac{1}{\rho_k}\pi(R_{k-1}+1,k)\,. \quad (21)$$

Using Eq. (21) in Equation (20), we obtain:

$$\pi(m+1,k) = \rho_k\pi(m,k) - \frac{\rho_k}{\rho_{k+1}}\pi(R_k+1,k+1)\,, \quad (22)$$

which by induction gives the following equation for $R_k+1 \leq m \leq F_k$:

$$\pi(m,k) = \rho_k^{m-R_k}\pi(R_k,k) - \frac{\rho_k}{\rho_{k+1}}\sum_{i=0}^{m-R_k-1}\rho_k^i\pi(R_k+1,k+1)\,. \quad (23)$$

Letting $m = R_k$, in Equation (12) yields to

$$\pi(R_k,k) = \frac{\rho_k^{R_k-F_{k-1}-1} - \rho_k^{R_k-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \quad (24)$$

and using Eq. (24) in Eq. (23), we get for $R_k+1 \leq m \leq F_k$:

$$\pi(m,k) = \rho_k^{m-R_k}\frac{\rho_k^{R_k-F_{k-1}-1} - \rho_k^{R_k-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) - \frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\,. \quad (25)$$

Thus, for $R_k+1 \leq m \leq F_k$, from Eq. (25) we prove Eq. (13). ■

It remains us to compute $\pi(R_k+1,k+1)$. From Equation (10), we have:

$$\pi(F_k,k) = \frac{1}{\rho_{k+1}}\pi(R_k+1,k+1)\,, \quad (26)$$

and using Equation (25) with $m = F_k$, it follows:

$$\pi(F_k,k) = \frac{\rho_k^{F_k-F_{k-1}-1} - \rho_k^{F_k-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) - \frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{F_k-R_k}}{1-\rho_k}\pi(R_k+1,k+1) \quad (27)$$

Hence, using equations (26) and (27), we derive:

$$\pi(R_{k+1},k+1) = \rho_{k+1}\frac{\rho_k^{F_k-F_{k-1}-1} - \rho_k^{F_k-R_{k-1}}}{1-\rho_k^{F_k-R_k+1}}\pi(R_{k-1}+1,k) \quad (28)$$

b) Case 2: $R_k \leq F_{k-1}$: We only recall the main equations, the details and the proofs are given in [6].

• if $R_{k-1}+2 \leq m \leq R_k$:

$$\pi(m,k) = \frac{1-\rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k)\,, \quad (29)$$

• if $R_k+1 \leq m \leq F_{k-1}+1$,

$$\pi(m,k) = \frac{1-\rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) - \frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\,, \quad (30)$$

• if $F_{k-1}+2 \leq m \leq F_k$

$$\pi(m,k) = \rho_k^{m-F_{k-1}-1}\frac{1-\rho_k^{F_{k-1}-R_{k-1}+1}}{1-\rho_k}\pi(R_{k-1}+1,k) - \frac{\rho_k}{\rho_{k+1}}\frac{1-\rho^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\,. \quad (31)$$

3) For the level K: The steady-state probabilities are given in the following lemma, and are proven in [6]:

Lemma 3 ([6]): We have two cases:

• if $R_{K-1}+2 \leq m \leq F_{K-1}+1$:

$$\pi(m,K) = \frac{1-\rho_k^{m-R_{K-1}}}{1-\rho_K}\pi(R_{K-1}+1,K)\,, \quad (32)$$

• if $F_{K-1}+2 \leq m \leq B$:

$$\pi(m,K) = \rho_K^{m-F_{K-1}-1}\frac{1-\rho_k^{F_{K-1}+1-R_{K-1}}}{1-\rho_K}\pi(R_{K-1}+1,K)\,. \quad (33)$$

IV. Performance measures and energy cost parameters

We propose now to calculate the expected cost in terms of performances and energetic consumption for the model presented in this paper. Once the steady state vector is calculated, we get various performance and energy consumption measures. Indeed the costs is expressed as an expected Markov reward function $\mathcal{R}$, where $\mathcal{R} = \sum_{m,k}\pi(m,\,k)\,r(m,\,k)$ and $r(m,\,k)$ the reward of state $(m,\,k)$, the metrics of interest are described hereafter. First, we give the performance measures.

The *mean number of customers* in the system is denoted by $\overline{N_C}$ and is equal to: $\overline{N_C} = \sum_{(m,k)\in A}\pi(m,k)\cdot m$.

The *mean number of losses* due to full queue by time unit is denoted by $\overline{N_R}$ and is equal to: $\overline{N_R} = \lambda\cdot\pi(B,K)\,.$

We denote by $\overline{R}$ the *mean response time* which is

$$\overline{R} = \frac{\overline{N_C}}{\lambda\cdot(1-\pi(B,K))}.$$

As the energy consumption depends on mean number of active servers as well as mean number of their activation/deactivation, then we define energetic consumption measures.

The *mean number of active servers* in the system is denoted by $\overline{N_S}$ and is equal to: $\overline{N_S} = \sum_{(m,k)\in A}S_k\cdot\pi(m,k)\,.$

The *mean number of activations triggered* by time unit, is denoted by $\overline{N_A}$ and is given by:

$$\overline{N_A} = \lambda\sum_{(m,k)\in A}(S_{k+1}-S_k)\cdot\mathbb{1}_{\{m=F_k;1\leq k\leq K-1\}}\cdot\pi(m,k).$$

The *mean number of deactivations triggered* by time unit is denoted by $\overline{N_D}$ and is given by:

$$\overline{N_D} = \sum_{(m,k)\in A}\min\{S_k,m\}\cdot\mu\cdot(S_k-S_{k-1})\cdot \mathbb{1}_{\{m=R_{k-1}+1\,;\,1\leq k\leq K-1\}}\cdot\pi(m,k)$$

In order to consider both performance and energy consumption, then we define the overall expected cost by time unit for the underlying model as follows:

$$\overline{C} = C_H\cdot\overline{N_C}+C_S\cdot\overline{N_S}+C_A\cdot\overline{N_A}+C_D\cdot\overline{N_D}+C_R\cdot\overline{N_R},$$

where, C_H is the per capita cost of holding one customer in the system within one time unit, C_S is the per capita cost of using one working server within one time unit, C_A is the activating cost (cost of switching one server from deactivating mode to activating mode), C_D is the deactivating cost and C_R is the cost of losses jobs due to full queue.

V. Numerical results

In this section, we implemented the resolution approaches presented in this paper (SCA + numerical resolution method, LDQBD and balance equations) and we performed a set of experiments. First, we present in part V-A a comparison in terms of execution time for developed resolution approaches. We notice that for our evaluations, we opted to use the GTH method [8] as numerical resolution techniques, because it is the most frequently used method. However, in the part V-A, we present also the case where we combine SCA with "Power method" (which is also a method often used to compute steady state probability vectors). Then, we give in part V-B some case studies in which we illustrate the evolution of performance and cost measures (presented in section IV) according to arrival rate and thresholds values. The observations that we extract from these case studies can be useful to design an optimization method to find thresholds values that minimise the overall cost.

A. Comparison of the resolution approaches in terms of execution time

Table I shows the execution time of each resolution approach for different values of: the number of levels (K) and the capacity of the system (B). We consider a model in which only one VM is activated/deactivated when we switch from one level to another ($S_1 = 1$, $S_{k+1} = S_k + 1\ \forall k < K$, so $C = S_K = K$) and we assume that $\mu = 1$. In the first line we have the smallest instance (K=5, B=300 - Markov Chain with 521 states), and in the last line we have the largest one (K=1000, B=75000 - Markov Chain with 134921 states). All tests were implemented and performed on a machine with "Intel i7" and 8GB of RAM. Results of table I show that the three approaches (column 2, 3 and 4) improve significantly the execution time compared to the power method (column 1). The closed form approach based on balance equations is the fastest one for all instances (at least about 100 times faster than conventional methods). Indeed, the computation in this approach is made using formulas that contain basic operators. The SCA+GTH approach takes a lot of time for large chains. The reason is the complexity of GTH method used to resolve the generated sub chains. The approach based on the LDQBD structure uses matrix inversion. For $K = 1000$ and $B = 75000$, the program returned an error "*out of memory*" because of the huge size of the matrix. We can conclude that the balance equations approach is the most appropriate for cloud systems with a very large number of VMs (thousands).

TABLE I. COMPARISON OF RESOLUTION APPROACHES IN TERMS OF EXECUTION TIME

-	Power Method	SCA + GTH	LDQBD	Balance equations
K = 5 B = 300 (521 states)	0.496 sec	0.246 sec	0.017 sec	0.006 sec
K = 10 B = 750 (1271 states)	6.863 sec	1.678 sec	0.041 sec	0.011 sec
K = 50 B = 3750 (6671 states)	651.848 sec	89.010 sec	0.496 sec	0.095 sec
K = 100 B = 7500 (13421 states)	+60 min	679.342 sec	2.775 sec	0.282 sec
K = 500 B = 37500 (67421 states)	+60 min	+60 min	304.437 sec	7.408 sec
K = 1000 B = 75000 (134921 states)	+60 min	+60 min	"Out of memory" (inversion of a very large matrix)	34.401 sec

B. Performance and energy consumption measures - Case Studies

In this section we consider a threshold-based queuing system for the Cloud and we present some experiments results. The goal is to observe the evolution of performance measures and the overall cost (that we defined in section IV) according to arrival rate and thresholds values. In order to have a clear interpretation of results, we assume in all experiments that $\mu = 1$. It could be pointed out that the notation used to write thresholds in figures is $F = [a : b : c]$ (with $a = F_1$, $c = F_{K-1}$ and $\forall k\ b = F_{k+1} - F_k$).

Experiment 1: we analyze the behavior of the average response time (figure 2) when we let vary the arrival rate λ. We assume that the number of levels $(K) = 5$, the system capacity $(B) = 300$, $S_1 = 3$ and $S_{k+1} = S_k + 3\ \forall k$ (i.e. we activate three VMs when we switch from level k to $k+1$). We performed experiments for different configurations of thresholds F and R (a curve for each configuration). In figure 2, the average response time increases with fluctuation according to λ. It becomes constant when λ goes beyond 15. The reason in that above 15 we have $\frac{\lambda}{C \cdot \mu} \geq 1$, $(C = S_5)$, which means that there are more customers arriving than the ability of the system to meet their needs so the system is saturated (i.e. the buffer is is generally full). If we compare the curves of figure 2, we notice that more the threshold values are significant then more the average response time is considerable. Indeed, when we choose larger thresholds, the VMs are activated following a larger number of customers in the system, which results in less performance. We also notice that when λ is not high the choice of the thresholds configuration has an impact (the difference between the curves is not significant), then when λ grows the configuration has less impact on the values of performance results.

Experiment 2: we analyse the behavior of the average number of servers (VMs) activations per time unit according to arrival rate (figures 3 and 4). We assume that the number of levels $(K) = 7$, the system capacity $(B) = 250$, $S_1 = 3$ and $S_{k+1} = S_k + 3\ \forall k$ (i.e. we activate three VMs when we switch from level k to $k+1$). We let vary the arrival rate λ in X-axis. We performed tests for different values for thresholds F and R. We assume in figure 3 a total overlap between the thresholds

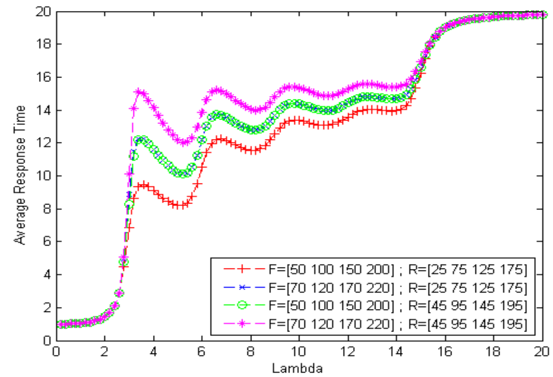

Fig. 2. Average response time in the system versus arrival rate (λ): $\mu = 1$, $K = 5$ and $B = 300$

of activation F and deactivation R, i.e. $R_1 < F_1 < R_2 < F_2 < R_3 < F_3 < R_4 < F_4 < R_5 < F_5 < R_6 < F_6$ and in figure 4 no overlap between the thresholds of activation F and deactivation R, i.e. $R_1 < R_2 < R_3 < R_4 < R_5 < R_6 < F_1 < F_2 < R_3 < F_4 < R_5 < F_6$. Results (figure 3 and 4) show that thresholds values and the difference between activation and deactivation thresholds influence the shape of the curve of activation rate. We notice that more activation thresholds (F) are far from deactivation thresholds (R) then less there are activations. Indeed, when thresholds are far from each other, we minimize oscillations between levels, which shows that it is interesting to use a hysteresis models for Cloud resources scaling.

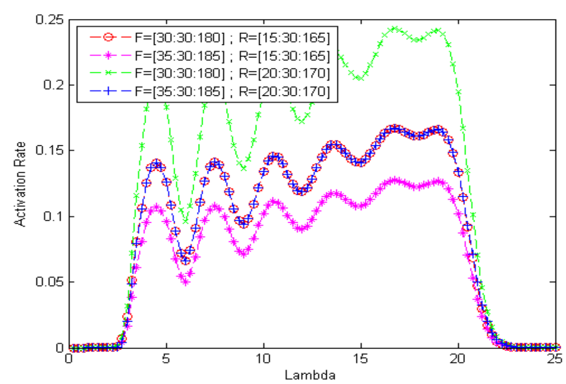

Fig. 3. Activation rate versus arrival rate (λ): $\mu = 1$, $K = 7$ and $B = 250$ (total overlap between activation and deactivation the thresholds)

Experiment 3: we analyse the overall cost (Figure 5). we assume that the arrival rate $(\lambda) = 4$, the number of levels $(K) = 7$ and system capacity $(B) = 500$, $S_1 = 3$ and $S_{k+1} = S_k + 3$ $\forall k$ (i.e. we activate three VMs when we switch from level k to $k+1$). We set $R = [45\ 95\ 145\ 195\ 245\ 295]$ and we vary in X-axis values of F. The initial value is $F_{init} = [50\ 100\ 150\ 200\ 250\ 300]$ (it corresponds to 0 in the x-axis) then we increase the values of F. For example, 10 in the X-axis corresponds to $F = F_{init} + 10 = [50\ 100\ 150\ 200\ 250\ 300] + 10 = [60\ 110\ 160\ 210\ 260\ 310]$. We measure the overall cost for different values of C_A (activationCost in Figure 5). We notice that when F increases the overall cost decreases in a

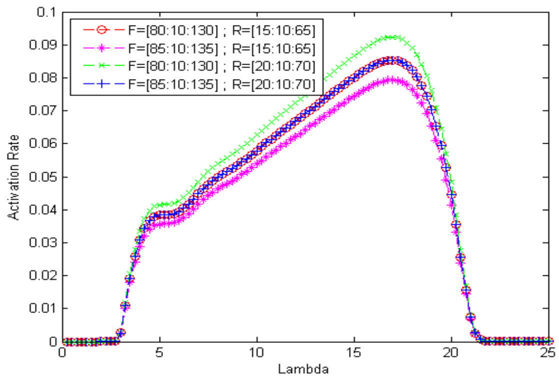

Fig. 4. Activation rate versus arrival rate (λ) : $\mu = 1$, $K = 7$ and $B = 250$ (no overlap between activation and deactivation thresholds)

first phase then increases after. The reason is that the formula of the overall cost includes both parameters that increases (for example: the average number of clients) and parameters that decreases (for example: the activation rate) according to F. The thresholds configuration that ensures the minimal cost depends on C_A (activationCost).

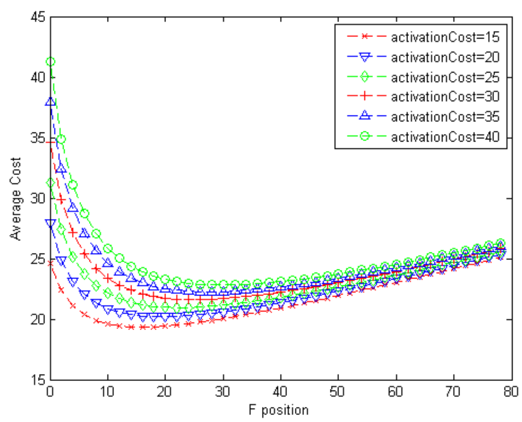

Fig. 5. Global cost versus F values: $\lambda = 4$, $\mu = 1$, $K = 7$, $B = 500$, $R = [45\ 95\ 145\ 195\ 245\ 295]$ and $F_{init} = [50\ 100\ 150\ 200\ 250\ 300]$

Experiment 4: we compare a model (**conf 1**) in which only one VM is activated/deactivated when we switch from one level to another ($K = 32$, $S_1 = 1$, $S_{k+1} = S_k{+}1\ \forall k < K$, so $S_K = S_{32} = 32$) and two models (**conf2**, **conf3**) in which many VMs are activated/deactivated when we switch from one level to another ($K = 6$, $S_1 = 1$, $S_{k+1} = 2 * S_k\ \forall k < K$, so $S_K = S_6 = 32$). The overall number of servers (VMs) for the three models is $C = 32$ but we have less thresholds in **conf2** and **conf3**. In the experiment, we assume that the system capacity (B) = 400 and we vary the arrivals rate (λ). Thresholds of **conf2** (respectively **conf3**) were chosen so that the associated model is an upper bound (respectively lower bound) for the performances of the model **conf1**. Values of F thresholds are illustrated in Figure 6 and we have $R_k = F_k - 5$ $\forall k$ in all models. Figure 7 illustrates the experiment results (the average number of clients of each model for different arrival rates (λ)). The result of **conf2** (resp. **conf3**) is always greater (resp. smaller) to the one-by-one model (**conf1**). **conf2** and **conf3** have less thresholds than **conf1**, so faster to analyse. This idea is useful to analyse large one-by-one models in which we can analyse bounding and smaller models to find a lower and upper bounds for performance rather than analysing the large (so complex) original model.

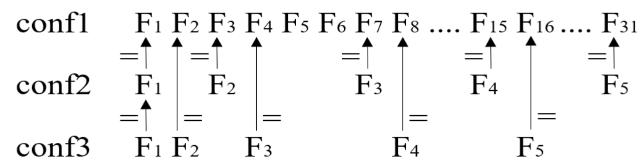

Fig. 6. The values of F for conf1, conf2 (upper bound) and conf3 (lower bound)

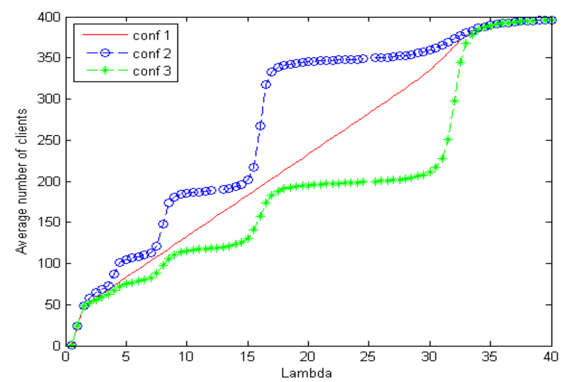

Fig. 7. Average response time in the system versus arrival rate (λ): $\mu = 1$, $B = 400$, 32 servers - Comparison of the three models -

VI. Conclusion

We develop numerical and analytical methods for the analysis of a hysteresis queueing system modelling a cloud system with activation/deactivation of block of VMs. We give numerical values even in the case of large Markov chains, for both performance and energy consumption measures, and we analyze the impact of the thresholds. One another important contribution of this paper is to suppose fewer constraints on the thresholds in order to analyze the impact on mean number of activations/deactivations. We define a global cost for performance and energy consumption in order to propose a trade off between performance and energy consumption. As a future, we propose to develop optimization algorithms in order to obtain the thresholds with minimize the overall cost.

References

[1] J. R. Artalejo, A. Economou, and M. J. Lopez-Herrero, “Analysis of a multiserver queue with setup times,” *Queueing Systems*, vol. 51, no. 1-2, pp. 53–76, 2005.

[2] A. Gandhi, M. Harchol-Balter, and I. Adan, “Server farms with setup costs,” *Performance Evaluation*, vol. 67, no. 11, pp. 1123–1138, 2010.

[3] I. Mitrani, “Managing performance and power consumption in a server farm,” *Annals of Operations Research*, vol. 202, pp. 121–134, 2013.

[4] M. Kitaev and R. Serfozo, “M/M/1 queues with switching costs and hysteretic optimal control,” *Operations Research*, pp. 310–312, 1999.

[5] J. C. Lui and L. Golubchik, “Stochastic complement analysis of multi-server threshold queues with hysteresis,” *Performance Evaluation*, vol. 35, no. 1, pp. 19–48, 1999.

[6] L.-M. Le Ny and B. Tuffin, "A simple analysis of heterogeneous multi-server threshold queues with hysteresis," in *Applied Telecommunication Symposium (ATS)*, 2002.

[7] F. Aït-Salaht and H. Castel-Taleb, "Bounding aggregations on phase-type arrivals for performance analysis of clouds," in *24th IEEE MASCOTS*, 2016, pp. 319–324.

[8] W. Stewart, *Introduction to the numerical Solution of Markov Chains*. Princeton University Press, 1995.

[9] M. F. Neuts, *Matrix-geometric solutions in stochastic models: an algorithmic approach*. John Hopkins University Press, 1981.

[10] G. Latouche and V. Ramaswami, "A logarithmic reduction algoritm for quasi-birth-death processes," *Journal of Applied Probability*, vol. 30, pp. 650–674, 1993.

[11] B. Gaujal, E. Hyon, and A. Jean-Marie, "Optimal routing in two parallel queues with exponential service times," *Discrete Event Dynamic Systems*, vol. 16, no. 1, pp. 71–107, 2006.

[12] H. Baumann and W. Sandmann, "Numerical solution of level dependent quasi-birth-and-death processes," *Procedia Computer Science*, vol. 1, no. 1, pp. 1561–1569, 2010.