Mathematical methods for performance analysis of hysteresis queues

M. Kandi F. Aït Salaht

H. Castel-Taleb
INSTITUT TELECOM/Telecom SudParis
SAMOVAR, UMR 5157
9, rue Charles Fourier, 91011 Evry Cedex, France
Email: hind.castel@it-sudparis.eu

E. Hyon
Sorbonne Universités,
UPMC Univ Paris 06,
CNRS, LIP6 Paris UMR 7606,
4 place Jussieu 75005 Paris
Université Paris Nanterre

***Abstract*—Considering a cloud system, we propose in this paper to evaluate the performance of a data center using mathematical methods. Modeled as a hysteresis queueing system, a data center is characterized by a forward and backward threshold which allow to represent its dynamic behavior. The client requests (or jobs) are represented by a Poisson process which arrive into the buffers and are executed by Virtual Machines (VMs). According to the occupation of the queue and the thresholds, the VMs are activated and deactivated by block. The system is represented by a complex Markov chain which is difficult to analyze when the size of the system is huge. We propose to use in this case different mathematical methods in order to compute the steady-state probability distribution. We propose to apply the SCA (Stochastic Complement Analysis) method in order to aggregate the state space. Another method which we develop is based on the numerical analysis of the system, it is the QBD (Quasi Birth and Death method). We propose also to use the balance equations in order to derive exact formulas for the steady-state probability distribution. We present some numerical results for the performance measures in order to compare the methods from their accuracy and the computation times.**

I. Introduction

One of the most significant recent progresses in the field of information and communication technology is Cloud computing, which may change the way people do computing and manage information. In this environment, a pool of abstracted, virtualized, dynamically-scalable computing functions and services are made accessible over the internet to remote users in an on-demand fashion, without the need for infrastructure investments and maintenance.

Virtualization plays a key role in the success of cloud computing because it simplifies the delivery of the services by providing a platform for resources in a scalable manner. One physical host can have more than one VM (Virtual Machine: it is a software that can run its own operating system and applications just like an operating system on a physical computer). With this flexibility, the cloud providers can rent the virtual machines depending on the demand and can gain more profit out of a single physical machine. With virtualization, service providers can ensure isolation of multiple user workloads, 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. In [**?**], they study the right ratio of VM instances to physical processors that optimizes the workload's performance given a workload and a set of physical computing resources.

Performance evaluation of cloud centers is an important research task which becomes difficult because the dynamic nature of cloud environments and diversity of user requests. Then, it is not surprising that in the recent area of cloud computing, only a portion of research results has been devoted to performance evaluation. In [**?**], they develop an analytical model in order to evaluate the performance of cloud centers with a high degree of virtualization and Poisson batch arrivals. The model of the physical machine with m VMs is based on the $M^{[x]}/G/m/m+r$ queue. They derive exact formulas for performance measures as blocking probability and mean waiting time of tasks. In [**?**], they consider a cloud center with a number of physical machines that are allocated to users in the order of task arrivals. Physical Machines (PMs) are considered with a high degree of virtualization, and are categorized into three server pools: hot, warm, and cold. The authors implement the sub-models using interactive Continuous Time Markov Chain (CTMC). The sub-models are interactive such that the output of one sub-model is input to the other one.

In this paper, we propose to use a mathematical model in order to evaluate the performance of a cloud node, more precisely, a data center. We represent the system by a queueing model based on queue-dependent virtual machines in order to analyse quantitatively the dynamic behavior of the data center. The data center is represented by a set of PM (Physical Machines) hosting a set of VMs which are instanced according to user demand. In this paper, we represent the data center as a set VMs which could be very large, especially if the user demand is high. With this model, virtual machines are activated and deactivated according to the intensity of user demand. The queueing model is a multi-server with threshold queues and hysteresis [**?**]. We suppose that customer requests arrivals follow a bulk process. Each server represents a VM, and the multi-server queueing model with hysteresis is governed by a sequence of forward and reverse thresholds which are different. The forward (resp. the backward) thresholds represent the value of the number of customers from which an additional VM is activated (resp. deactivated). Obviously, the relevance of this model is to offer the flexibility of different thresholds for activating and removing VMs.

As the system is difficult to analyze exactly, especially when the number of VMs or the size of bulk arrivals is high, we propose to use stochastic comparisons in order to compute more easily, and so faster performance measure bounds. The bounding models are obtained by the simplification of the hysteresis model in order to compute easily the performance measures. We propose to simplify the batch arrival process by generating aggregated bounding processes. So the bounding systems are equivalent to the hysteresis system with aggregated bounding arrival process. We derive an upper bounding system (resp. a lower bounding system) from an upper bound batch arrival distribution (resp. a lower bound batch arrival distribution). We prove using stochastic comparisons that these processes provide really bounds for performance measures as blocking probabilities, expected buffer length and expected departure.

We give some numerical values according to different values of input parameters: arrival rate, size batches, and the number of VMs (called the degree of virtualization). The results show clearly the relevance of our approach to propose a tradeoff between computational complexity and accuracy of results. So it can efficiently solve the network dimensioning problem from QoS (Quality of Service) constraint requirements.

The paper is organized as follows: next, we describe the cloud system, and in section II, we present the queueing model for the analysis. In section III, we give some theoretical notions of the stochastic ordering theory and in section **??**, we give the bounding models and we prove using the stochastic comparisons that they represent really bounds. In the section **??**, we give numerical results of the performance measures. Finally, achieved results are discussed in the conclusion and comments about further research issues are given. The activation/deactivation one by one model has been extensively studied in the literature [**?**], [**?**], [**?**]. In [**?**], the authors use the concept of stochastic complementation to solve the system. They propose to partition 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 [**?**], 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. And In [**?**], the authors propose to analyze this system through stochastic bounding theory. Confronted with a computational complexity problem in the Cloud system (the cloud systems are often defined on very large state spaces which makes their exact analysis very cumbersome or even impossible), the authors derived bounding models and defined an accurate bounds on performance measures. They offer a trade-off between the accuracy of the results and the computation time.

In this paper, we propose to study the Activation/Deactivation model block by block, which to our knowledge has never been considered and studied previously in the literature. So, we try in the following, to investigate this model and present some analysis and resolution methods.

II. Cloud system description

We analyse a data center in 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 level, where each level corresponds to a certain number of active servers. The number of active servers at level i is denoted by S_i, where $S_1 \leq S_2 \leq ... \leq S_K$. We suppose that $S_1 \geq 1$, so we have at least one active server by assumption. The transition from functioning level i to level $i+1$ allows to allocate (turn on) one or more additional servers, going from S_i to S_{i+1} active servers, while the transition from level i to level $i-1$ allows to remove (turn off) one or more active servers, going from S_i to S_{i-1} active servers. Depending on the system occupancy, we transit from the level i to level $i+1$ when the workload in the system exceeds a threshold F_i, and from level i to level $i-1$ when the workload in the system falls below a threshold R_{i-1}. So, the model is characterized by activation thresholds $F = (F_1, F_2, ..., F_{K-1})$ (called also forward thresholds), and deactivation thresholds $R = (R_1, R_2, ..., R_{K-1})$ (called also reverse thresholds). These thresholds are fixed and can not be modified during the system works. We furthermore assume that $F_1 < F_2 < ... < F_{K-1}$, that $R_1 < R_2 < ... < R_{K-1}$ and that $R_i < F_i, \forall i, 1 \leq i \leq K-1$.

We assume 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 (x_1, x_2) such that x_1 is the number of customers in the system and x_2 is the functioning level. The state space is denoted by A and is given as follows:

$$\begin{aligned} A = \{(x_1, x_2) \,|\, & 0 \leq x_1 \leq F_1, \text{ if } x_2 = 1; \\ & R_{i-1} + 1 \leq x_1 \leq F_i, \text{ if } i-1 < x_2 \leq i, \\ & \qquad \forall i \text{ s.t. } 1 < i < K; \\ & R_{K-1} + 1 \leq x_1 \leq B, \text{ if } x_2 = K; \} \end{aligned}$$

Recalling that S_{x_2} represents the number of active servers at level x_2, the transitions between states follows:

$$\begin{array}{lll} (x_1,\, x_2) & \rightarrow & (\min\{B, x_1+1\},\, x_2) \\ & & \text{with rate } \lambda, \\ & & \text{if } x_1 \neq F_i \text{ or if } x_1 = F_i \text{ and } x_2 \neq i; \\ & \rightarrow & (\min\{B, x_1+1\},\, \min\{K,\, x_2+1\}) \\ & & \text{with rate } \lambda, \\ & & \text{if } x_1 = F_i \text{ and } x_2 = i; \\ & \rightarrow & (\max\{0, x_1-1\},\, x_2), \\ & & \text{with rate } \min\{S_{x_2},\, x_1\} \cdot \mu, \\ & & \text{if } x_1 \neq R_i+1 \text{ or if } x_1 = R_i+1 \text{ and } x_2 \neq i+1; \\ & \rightarrow & (\max\{0, x_1-1\},\, \max\{0, x_2-1\}) \\ & & \text{with rate } \min\{S_{x_2},\, x_1\} \cdot \mu, \\ & & \text{if } x_1 = R_i+1 \text{ and } x_2 = i+1\,. \end{array}$$

An example of the transitions is given Figure 1.

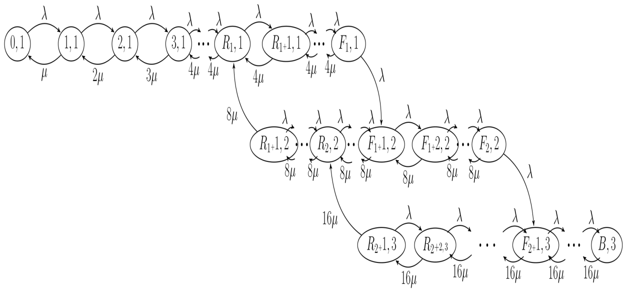

Fig. 1. Transition structure for $K = 3$, $S_1 = 4$, $S_2 = 8$; $S_3 = 16$, $R_1 + 1 \geq 8$ and $R_2 + 1 \geq 16$.

III. Resolution approaches

In order to compute the exact performance measures of the presented models, we expose hereafter some techniques to solve the CTMCs and compute the steady state probability vector. In this section, we present three resolution methods. A comparison in terms of execution times of these methods is presented in Section V.

A. Stochastic Complement Analysis (SCA)

To solve the $X(t)$ and $Y(t)$ Markov chains, 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. [**?**] and works as follows. First, we aggregate the state space of the underlying 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 a Markov chain can be obtained using any chosen solution technique, as described in [**?**]. Lastly, a disaggregation technique is applied to compute the individual steady state probabilities of the original Markov process.

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

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

$$Q = \begin{pmatrix} Q_{A_1A_1} & Q_{A_1A_2} \\ Q_{A_2A_1} & Q_{A_2A_2} \end{pmatrix} \quad (1)$$

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

Based on this theorem Lui et al. have investigated the "A/D One by One" model using some restrictions. Here, and without any restrictions, we propose to give first a general formulation for the "A/D One by One" model and then state some results for the "A/D by block" model.

Given the $\{X_t\}_{t\geq 0}$ Markov chain (resp. $\{Y_t\}_{t\geq 0}$ Markov chain) with state space A. We partition the state space A into K distinct sets denoted A_m, where:

$$A_m = \{(x_1, x_2)|(x_1, x_2) \in A \text{ and } x_2 = m\}; \forall m \in \{1, 2, ..., K\} \quad (2)$$

The set A_m contains the states belonging the level m.

Let $\{(X_m)_t\}_{t\geq 0}$ (resp. $\{Y_t\}_{t\geq 0}$) be a Markov chain defined on state space A_m, $\forall m \in \{1, 2, ..., K\}$. We denote by π_m the steady state probabilities of X_m (resp. Y_m) . The transitions in $\{(X_m)_t\}_{t\geq 0}$ (resp. $\{(Y_m)_t\}_{t\geq 0}$) are identical to those appear in the original process $\{X_t\}_{t\geq 0}$ (resp. $\{(X)_t\}_{t\geq 0}$) for the level m except some additional modifications. This modifications are set out below.

For $m = 1$:

- We add a transition from state $(F_1, 1)$ to state $(R_1, 1)$, with a rate λ.

For $m \in \{2, \ldots, K-1\}$:

- We add a transition from state (F_m, m) to state (R_m, m), with a rate λ.
- and a transition with a rate $(\min\{R_{m-1}+1, S_m\} * \mu)$ is added from $(R_{m-1}+1, m)$ to state $(F_{m-1}+1, m)$.

For $m = K$:

- We add a transition from state $(R_{K-1}+1,\, K)$ to state $(F_{K-1}+1,\, K)$, with a rate $\min\{R_{K-1}+1, S_K\} * \mu$.

The aggregated process that brings all aggregate states is a simple birth and death process, with:

- $\lambda_i = \lambda * \pi_i(F_i), \quad \forall i \in 1, ...K-1$ and
- $\mu_i = \min(R_{i-1}+1, S_i) * \mu * \pi_i(R_{i-1}+1), \ \forall i \in 2, ...K$.

We denote by π the steady state probabilities of this aggregated process.

At this point we have all the necessary information to compute the steady probabilities of $\{X_t\}_{t\geq 0}$ (resp. $\{Y_t\}_{t\geq 0}$).

Indeed, we determine: (1) for each level m ($m = 1 \dots K$), the conditional state probabilities of all states, and (2) the steady state probability of the aggregated process. Hence, the steady state probability of each individual state (i, j) in $\{X_t\}_{t\geq 0}$ (resp. $\{Y_t\}_{t\geq 0}$), can be expressed as

$$\Pi(i,\, j) = \pi_j(i)\, \pi(j) \;\text{ where }\; (i,\, j) \in A_j.$$

B. Closed-form solution: QBD

The particular form of the Markov Chain generator suggest us to use the Quasi Birth and Death (QBD) processes framework to benefits of the numerous numerical methods to solve them [**?**]. 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 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 $Q_{m,n}(i, j)$ that denotes the i-th line and j-th column element of matrix $Q_{m,n}$. We have

Proposition 1: The Markov Chain $\{X_t\}_{t\geq 0}$ defined section II is a level dependant QBD with K levels, corresponding to the functioning levels, with a generator Q given by:

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

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

For $i = 1$ we have:

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

and

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

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

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

also

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

and

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

Finally for $i = K$, it follows

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

and

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

Proof:

Supposons qu'on est dans un tat (i, j) quelconque, et vrifions que les tats qui peuvent łtre atteints par (i, j) sont ceux appartenant au młme niveau (i.e. le niveau j), le niveau suivant (i.e. le niveau $j + 1$) ou bien le niveau prcdent (i.e. le niveau $j - 1$). En se basant sur la description de la chane (section II), il y a deux types d'vnements possibles:

(1) une arrive: si $j < K$ et $i = F_j$ alors on passe au niveau suivant (i.e. le niveau $j + 1$), sinon on reste au młme niveau (i.e. le niveau j),

(2) un dpart: si $j > 1$ et $i = R_{j-1} + 1$ alors on passe au niveau prcdent (i.e. le niveau $j - 1$), sinon on reste au młme niveau(i.e. le niveau j).

Donc le gnrateur infinitsimal peut łtre reprsent par une matrice tridiagonale par blocs. Le contenu de chaque bloc est driv directement partir de la description des transitions. ■

Numerically solving QBD as well as level dependant QBD is often based on a matrix geometric methods [**?**], [**?**] or kernel methods [**?**]. This is a hard computationnal task requiring to solve matrices equation. Equally, for LDQBD it exists numerical methods to solve them. Here the method proposed in [**?**] is used since it is shown that this method is efficient and numerically stable.

C. Mathematical analysis using balance equations

We give a closed form for the steady state probability using balance equations, and cuts on the state space. As in [**?**], we compute the probabilities level by level, where one level corresponds to a certain number of active servers. The relevance of our work is that we take more general cases for the thresholds for each level $2 \leq k \leq K : R_k \leq F_{k-1}$, and $R_k > F_{k-1}$. 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 it is $S_1\mu$, if $m \geq S_1$.

In the sequel, we denote by $\mu_k = S_k\mu$, for $k = 1, \dots, K$, and by $\rho_k = \frac{\lambda}{\mu_k}$, for $k = 1, \dots K$. We consider also that $\rho = \frac{\lambda}{\mu}$. In the sequel, we give the probabilities 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 $2 \leq k \leq K$, 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. Next, we give in details the steady-state probabilities level by level.

1) Analysis of level 1: 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 $\{(0,1), \dots, (m,1)\}$. If $0 \leq m < S_1$, we have :

$$\mu(m+1)\pi(m+1,1) = \lambda\pi(m,1) \tag{3}$$

We deduce that if $0 \leq m \leq S_1$:

$$\pi(m,1) = \frac{\rho^m}{m!}\pi(0,1) \tag{4}$$

and if $S_1 < m \leq R_1$,

$$\pi(m,1) = \rho_1^{m-S_1}\frac{\rho^{S_1}}{S_1!}\pi(0,1) \tag{5}$$

Making cuts around the sets $\{(0,1), \dots, (R_1,1), \dots, (m,1)\}$, for $R_1 \leq m \leq F_1 - 1$, allows to derive the following balance equations:

$$\mu_1\pi(m+1,1) + \mu_2\pi(R_1+1,2) = \lambda\pi(m,1) \tag{6}$$

For equation 6, we have if $R_1 + 1 \leq m \leq F_1$:

$$\pi(m,1) = \rho_1^{m-R_1}\pi(R_1,1) - \pi(R_1+1,2)\frac{\rho_1}{\rho_2}\sum_{k=0}^{m-R_1-1}\rho_1^k \tag{7}$$

And we deduce that:

$$\pi(m,1) = \rho_1^{m-R_1}\pi(R_1,1) - \pi(R_1+1,2)\frac{\rho_1}{\rho_2}\frac{1-\rho_1^{m-R_1}}{1-\rho_1} \tag{8}$$

From equation 8, we deduce for $m = F_1$:

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

If we make a cut between states of level 1 and states of other levels, then we obtain the following evolution equation: $\lambda\pi(F_1,1) = \mu_2\pi(R_1+1,2)$. We deduce that:

$$\pi(F_1,1) = \frac{1}{\rho_2}\pi(R_1+1,2).$$

As from Equation (5), for $m = R_1$, we have that:

$$\pi(R_1,1) = \rho_1^{R_1-S_1}\frac{\rho^{S_1}}{S_1!}\pi(0,1) \tag{10}$$

So, we deduce that:

$$\pi(R_1+1,2) = \frac{\rho_2\rho_1^{F_1-S_1}(1-\rho_1)}{1-\rho_1^{F_1-R_1+1}}\frac{\rho^{S_1}}{S_1!}\pi(0,1) \tag{11}$$

Then using equations 10, and equation 11, in equation 8, we deduce that for $R_1 + 1 \leq m \leq F_1$:

$$\pi(m,1) = \frac{\rho^{S_1}}{S_1!}(\rho_1^{m-S_1} - \frac{\rho_1^{F_1-S_1+1}(1-\rho_1^{m-R_1})}{1-\rho_1^{F_1-R_1+1}})\pi(0,1) \tag{12}$$

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:

$$\pi(R_{k-1}+1,k) = \rho_k\pi(F_{k-1},k-1) \tag{13}$$

Now, we compute the probabilities of each level, by expressed each of them in terms of $\pi(R_{k-1}+1,k)$. According to the threshold values, we have two cases to consider: $R_k \leq F_{k-1}$ or $R_k > F_{k-1}$. Note that the case of $R_k \leq F_{k-1}$, has been considered in [**?**], so we give just the main equations. We present in details the case $R_k > F_{k-1}$.

- **If $\mathbf{R_k > F_{k-1}}$**

If we consider cuts on the state space around the sets $\{(R_{k-1}+1,k), \dots, (m,k)\}$ for $R_{k-1}+1 \leq m \leq F_{k-1}$, then we obtain the following equations:

$$\lambda\pi(m,k) + \mu_k\pi(R_{k-1}+1,k) = \mu_k\pi(m+1,k) \tag{14}$$

So we deduce the following equation if $R_{k-1}+2 \leq m \leq F_{k-1}+1$:

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

And we obtain from equation 15 by induction that for $R_{k-1}+2 \leq m \leq F_{k-1}+1$:

$$\pi(m,k) = \frac{1-\rho^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \tag{16}$$

If we consider cuts on the state space around sets $\{(R_{k-1}+1,k), \dots, (F_{k-1}+1,k), \dots, (m,k)\}$, for $F_{k-1}+1 \leq m \leq R_k - 1$, then we obtain the following balance equation :

$$\begin{aligned}\lambda\pi(m,k) + \mu_k\pi(R_{k-1}+1,k) &= \mu_k\pi(m+1,k) \\ &\quad +\lambda\pi(F_{k-1},k-1)\end{aligned} \tag{17}$$

From Equation (17), we get for $F_{k-1}+2 \leq m \leq R_k$:

$$\begin{aligned}\pi(m,k) &= \rho_k\pi(m-1,k) - \rho_k\pi(F_{k-1},k-1) \\ &\quad +\pi(R_{k-1}+1,k)\end{aligned} \tag{18}$$

So, we deduce from (18) that:

$$\begin{aligned}\pi(m,k) &= \rho_k^{m-F_{k-1}-1}\pi(F_{k-1}+1,k) \\ &+ \sum_{i=0}^{m-F_{k-1}-2} \rho_k^i \pi(R_{k-1}+1,k) \\ &- \sum_{i=1}^{m-F_{k-1}-1} \rho_k^i \pi(F_{k-1},k-1)\end{aligned} \tag{19}$$

From Equation (16), for $m = F_{k-1}+1$, we obtain:

$$\pi(F_{k-1}+1,k) = \frac{1-\rho^{F_{k-1}+1-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \tag{20}$$

So from equations (20), (13), and (19) we obtain:

$$\begin{aligned}\pi(m,k) &= \frac{1-\rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \\ &- \sum_{i=1}^{m-F_{k-1}-1} \rho_k^i \frac{1}{\rho_k}\pi(R_{k-1}+1,k)\end{aligned} \tag{21}$$

So we obtain for $F_{k-1}+2 \leq m \leq R_k$:

$$\pi(m,k) = \frac{\rho_k^{m-F_{k-1}-1} - \rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \tag{22}$$

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

$$\begin{aligned}\pi(m,k)\lambda + \pi(R_{k-1}+1,k)\mu_k &= \mu_k\pi(m+1,k) \\ &+\mu_{k+1}\pi(R_{k-1}+1,k) \\ &-\mu_{k+1}\pi(R_k+1,k+1) \\ &+\lambda\pi(F_{k-1},k-1)\end{aligned} \tag{23}$$

From equation 23, we have for $R_k \leq m \leq F_k - 1$

$$\begin{aligned}\pi(m+1,k) &= \rho_k\pi(m,k) + \pi(R_{k-1}+1,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\pi(R_k+1,k+1) \\ &-\rho_k\pi(F_{k-1},k-1)\end{aligned} \tag{24}$$

From equation 13, we have:

$$\pi(F_{k-1},k-1) = \frac{1}{\rho_k}\pi(R_{k-1}+1,k) \tag{25}$$

Using 25 in equation 24, we obtain:

$$\pi(m+1,k) = \rho_k\pi(m,k) - \frac{\rho_k}{\rho_{k+1}}\pi(R_k+1,k+1) \tag{26}$$

By induction, we derive the following equation for $R_k + 1 \leq m \leq F_k$:

$$\begin{aligned}\pi(m,k) &= \rho_k^{m-R_k}\pi(R_k,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\sum_{i=0}^{m-R_k-1}\rho_k^i\pi(R_k+1,k+1)\end{aligned} \tag{27}$$

From equation 22, for $m = R_k$, we obtain:

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

So using equation 28 in equation 27, we obtain for $R_k+1 \leq m \leq F_k$:

$$\begin{aligned}\pi(m,k) &= \rho_k^{m-R_k}\frac{\rho_k^{R_k-F_{k-1}-1} - \rho_k^{R_k-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\end{aligned} \tag{29}$$

So from Equation (29), we derive $R_k + 1 \leq m \leq F_k$:

$$\begin{aligned}\pi(m,k) &= \frac{\rho_k^{m-F_{k-1}-1} - \rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\end{aligned} \tag{30}$$

As we need to compute $\pi(R_k+1,k+1)$, then from Equation (13), we have:

$$\pi(F_k,k) = \frac{1}{\rho_{k+1}}\pi(R_k+1,k+1) \tag{31}$$

And using Equation (29) for $m = F_k$, we obtain:

$$\begin{aligned}\pi(F_k,k) &= \frac{\rho_k^{F_k-F_{k-1}-1} - \rho_k^{F_k-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{F_k-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\end{aligned} \tag{32}$$

So using equations 31 and 32, we derive:

$$\begin{aligned}&\pi(R_{k+1},k+1) = \\ &\rho_{k+1}\frac{\rho_k^{F_k-F_{k-1}-1} - \rho_k^{F_k-R_{k-1}}}{1-\rho_k^{F_k-R_k+1}}\pi(R_{k-1}+1,k)\end{aligned} \tag{33}$$

In the case of $R_k \leq F_{k-1}$, the main equations are given in [**?**]:

If $R_{k-1}+2 \leq m \leq R_k$,

$$\pi(m,k) = \frac{1-\rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \tag{34}$$

If $R_k + 1 \leq m \leq F_{k-1}+1$,

$$\begin{aligned}\pi(m,k) &= \frac{1-\rho_k^{m-R_{k-1}}}{1-\rho_k}\pi(R_{k-1}+1,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\frac{1-\rho_k^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\end{aligned} \tag{35}$$

If $F_{k-1}+2 \leq m \leq F_k$

$$\begin{aligned}\pi(m,k) &= \rho_k^{m-F_{k-1}-1}\frac{1-\rho_k^{F_{k-1}-R_{k-1}+1}}{1-\rho_k}\pi(R_{k-1}+1,k) \\ &-\frac{\rho_k}{\rho_{k+1}}\frac{1-\rho^{m-R_k}}{1-\rho_k}\pi(R_k+1,k+1)\end{aligned} \tag{36}$$

3) For the level K: If $R_{K-1}+2 \leq m \leq F_{K-1}+1$, we have the following equations:

$$\pi(m,K) = \frac{1-\rho_k^{m-R_{K-1}}}{1-\rho_K}\pi(R_{K-1}+1,K) \tag{37}$$

and for $F_{K-1}+2 \leq m \leq B$, we have: $\pi(m,K) = \rho_K^{m-F_{K-1}-1}\pi(F_{K-1}+1,K)$. Which is equivalent to:

$$\pi(m,K) = \rho_K^{m-F_{K-1}-1}\frac{1-\rho_k^{F_{K-1}+1-R_{K-1}}}{1-\rho_K}\pi(R_{K-1}+1,K) \tag{38}$$

Algorithm 1 Compute π

```
{probabilities of level 1}
for $1 \leq m \leq F_1$ do
  if $m < S_1$ then
    compute $\pi(m,1)$ from equation 4
  else if $S_1 < m \leq R_1$ then
    compute $\pi(m,1)$ from equation 5
  else
    compute $\pi(m,1)$ from equation 12
  end if
end for
{probabilities of level k }
for $2 \leq k \leq K-1$ do
  compute $\pi(R_{k-1}+1,k)$ from equation 13
  compute $\pi(R_k+1,k+1)$ from equation 33
  if $R_k \leq F_{k-1}$ then
    for $R_{k-1}+2 \leq m \leq F_k$ do
      if $R_{k-1}+2 \leq m \leq R_k$ then
        compute $\Pi(m,k)$ from equation 34
      else if $R_k+1 \leq m \leq F_{k-1}+1$ then
        compute $\Pi(m,k)$ from equation 35
      else
        compute $\Pi(m,k)$ from equation 36
      end if
    end for
  else
    for $R_{k-1}+2 \leq m \leq F_k$ do
      if $R_{k-1}+2 \leq m \leq F_{k-1}+1$ then
        Compute $\Pi(m,k)$ from equation 16
      else if $F_{k-1}+2 \leq m \leq R_k$ then
        Compute $\Pi(m,k)$ from equation 22
      else
        Compute $\Pi(m,k)$ from equation 30
      end if
    end for
  end if
end for
{probabilities of level K }
compute $\pi(R_{K-1}+1,K)$ from equation 13
for $R_{k-1}+2 \leq m \leq B$ do
  if $R_{K-1}+2 \leq m \leq F_{K-1}+1$ then
    Compute $\Pi(m,k)$ from equation 37
  else
    Compute $\Pi(m,k)$ from equation 38
  end if
end for
```

IV. PERFORMANCE MEASURES AND ENERGY COST PARAMETERS

We propose in this section to calculate the cost generated by the models presented in this paper in terms of performances and energetic consumption. To compute this cost, we need to define, in advance, some performance measures.

From the steady state vector of the model, we can compute various performance measures; more specifically, we can compute any performance measures which can be expressed in the form of a Markov reward function $\mathcal{R}$, where $\mathcal{R} = \sum_{i,j}\pi(i,j)\,R(i,j)$ and $R(i,j)$ is the reward for state (i,j). We present in the following, the metrics we considered.

■ Mean number of customer in the system for both "A/D one by one" and "A/B by block" models.

$$\overline{N_C} = \sum_{(x_1,x_2)\in A} x_1 * \pi(x_1,x_2) \tag{39}$$

■ Mean number of active servers in the system.

– For the "A/D one by one" model, we have:

$$\overline{N_S} = \sum_{(x_1,x_2)\in A} x_2 * \pi(x_1,x_2) \tag{40}$$

– For the "A/D by block" model we have the following formula:

$$\overline{N_S} = \sum_{(x_1,x_2)\in A} S_{x_2} * \pi(x_1,x_2) \tag{41}$$

■ Mean number of activations triggered by time unit.

– In the case of the "A/D one by one" model, we can express it as follows:

$$\overline{N_A} = \sum_{(x_1,x_2)\in A} \lambda * 1_{\{x_1=F_i \text{ et } x_2=i\ ;\ 1\leq i\leq K-1\}} * \pi(x_1,x_2) \tag{42}$$

– In the case of "A/D by block" model, we have the following formula:

$$\overline{N_A} = \sum_{(x_1,x_2)\in A} \lambda * (S_{x_2+1} - S_{x_2}) * 1_{\{x_1=F_i \text{ and } x_2=i\ ;\ 1\leq i\leq K-1\}} * \pi(x_1,x_2) \tag{43}$$

■ Mean number of deactivation triggered by time unit. For the two models, we have respectively the following expressions.

– For the "A/D one by one" model, we have:

$$\overline{N_D} = \sum_{(x_1,x_2)\in A} (x_2\mu) \times 1_{\{x_1=R_i+1 \text{ and } x_2=i+1\ ;\ 1\leq i\leq K-1\}} \times \pi(x_1,x_2) \tag{44}$$

– For the "A/D by block" model, we have:

$$\overline{N_D} = \sum_{(x_1,x_2)\in A} (\min\{S_{x_2},x_1\} * \mu) * (S_{x_2} - S_{x_2-1}) \times 1_{\{x_1=R_i+1 \text{ and } x_2=i+1\ ;\ 1\leq i\leq K-1\}} \times \pi(x_1,x_2) \tag{45}$$

■ Mean number of losses due to full queue (in both models):

$$\overline{N_R} = \lambda * \pi(B,K) \tag{46}$$

■ Mean response time.

$$\overline{R} = \frac{\overline{N_C}}{\lambda * (1 - \pi(B,K))} \tag{47}$$

The overall cost per time unit for the underlying model is given by:

$$\overline{C} = C_H*\overline{N_C}+C_S*\overline{N_S}+C_A*\overline{N_A}+C_D*\overline{N_D}+C_R*\overline{N_R} \tag{48}$$

where, C_H is the cost of holding within one time unit, C_S is the cost of one working server within one time unit, C_A is the activating cost (cots 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

We implemented the mathematical methods : (1) aggregation, (2) LDQBD, (3) balance equations, and we performed a set of experiments. In this section, we first present a comparison between the three methods in terms of resolution time. Then we give some numerical examples that aim to show the evolution of performance measures and cost according to arrival rate and thresholds values. All tests were implemented and performed on a machine with "Intel i7" CPU and 8GB of RAM.

A. Comparison between mathematical methods

Table I shows the execution time of each mathematical method for different values of : K (the number of levels) and B (the capacity of the system). In the first line we have the smallest instance (K=5, B=300, markov chain with 521 stats), and in the last line we have the largest one (K=1000, B=75000, markov chain with 134921 stats). We observe that the method using the balance equations is the fastest among the three for all instances. Indeed, the calculation is made using formulas that contain basic operations. The aggregation method takes a lot of time for large chains. The reason is the complexity of the GTH approach that we use to resolve the generated sub chains. The method 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.

TABLE I. Comparison between mathematical methods in terms of execution time

-	Agregation + GTH	LDQBD	Balance equations

K = 5
B = 300
(521 stats)
	0.246 sec	0.017 sec	0.006 sec

K = 10
B = 750
(1271 stats)
	1.678 sec	0.041 sec	0.011 sec

K = 50
B = 3750
(6671 stats)
	89.010 sec	0.496 sec	0.095 sec

K = 100
B = 7500
(13421 stats)
	679.342 sec	2.775 sec	0.282 sec

K = 500
B = 37500
(67421 stats)
	+30 min	304.437 sec	7.408 sec

K = 1000
B = 75000
(134921 stats)
	+30 min	"Out of memory"
		(inversion of
		a very large matrix)
			34.401 sec

B. Numerical Examples : Evaluation of performance and cost

In this section we consider a threshold-based queuing system with hysteresis and we present some numerical examples that aim observe the evolution of performance measures and cost.

We first analyze the behavior of the average number of clients in the system (figure 2) and the average response time (figure 3). In these figures : the service rate $(\mu) = 1$, the number of servers $(K) = 5$, and system capacity $(B) = 400$. In X-axis, we vary the arrival rate (λ). Y-axis in figure 2 (resp. figure 3) represents the average number of clients in the system (resp. the average response time). We performed tests for different values for thresholds F and R (which is represented by the four curves). In figure 2, the average number of clients increases when the arrival rate (λ) is bigger. The system is saturated when λ reaches 5. The reason is that above $\lambda = 5$, we have $\frac{\lambda}{K*\mu} >= 1$. If we compare the curves of figure 2 (resp. 3), we notice that more the threshold values are significant then more the average number of clients (resp. average response time) is considerable. i.e. when we choose larger thresholds, the servers are activated following a larger number of customers in the system, which results in less performance.

We next analyze the behavior the average number of servers activations per time unit (figures 4, 5 and 6). In these figures : service rate $(\mu) = 1$, the number of servers $(K) = 7$, and system capacity $(B) = 250$. In X-axis, we vary the arrival rate (λ).Y-axis represents activation rate (i.e. the average number of servers activations per time unit). We performed tests for different values for thresholds F and R. The notation used to write thresholds in the figure is $F = [a : b : c]$ (with $a = F_1$, $c = F_{K-1}$ and $\forall i \; b = F_{i+1} - F_i$). In figure 4, we have a total overlap between the thresholds of activation F and deactivation R (i.e. $R_1 < F_1 < R_2 < F_2 < R_3 < F_3 < R_4 < F_4 < R_5 < F_5 < R_6 < F_6$). In figure 5 we have no overlap between activation and deactivation thresholds (i.e. $R_1 < R_2 < R_3 < R_4 < R_5 < R_6 < F_1 < F_2 < F_3 < F_4 < F_5 < F_6$). And finally for Figure 6, we have a partial overlap between the activation and deactivation thresholds (i.e. $R_1 < R_2 < R_3 < F_1 < F_2 < F_3 < R_4 < R_5 < R_6 < F_4 < F_5 < F_6$). Results show that thresholds values and the difference between activation thresholds and deactivation thresholds influence the shape of the curve of activation rate. We notice that more activation thresholds (F) are far from deactivation thresholds

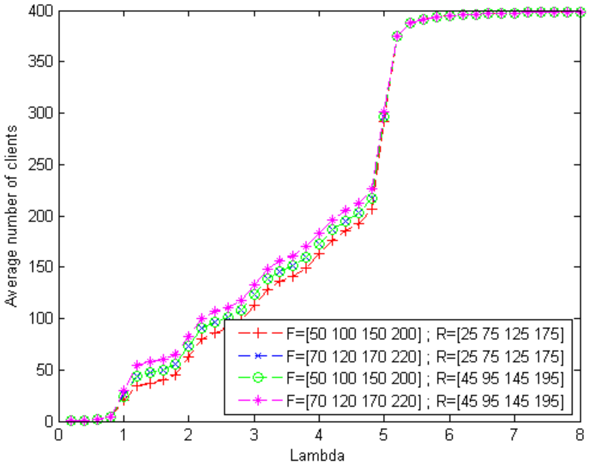

Fig. 2. Average number of clients in the system versus arrival rate (λ) : $\mu = 1$, $K = 5$ and $B = 400$

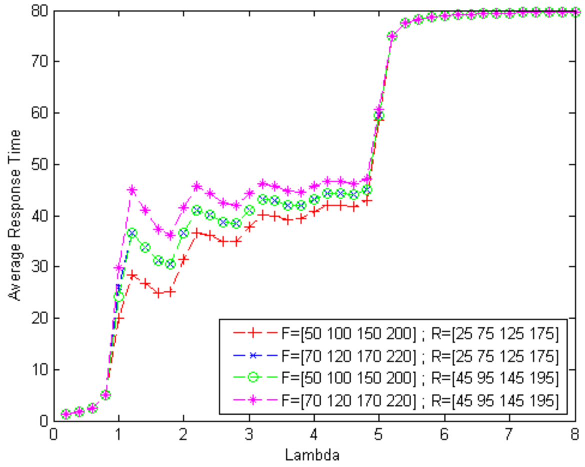

Fig. 3. Average response time in the system versus arrival rate (λ) : $\mu = 1$, $K = 5$ and $B = 400$

(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.

Finally we analyze the global cost (figure 7). In this figure : service rate (μ) = 1, arrival rate (λ) = 4, number of servers (K) = 7 and system capacity (B) = 500. 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 global cost for different values of C_A (activationCost in the figure). We notice that when F increases the global cost decreases in a first phase then increases after. The reason is that the formula of the global cost includes both the average number of clients that increases and the activation rate that decreases. The thresholds configuration that ensures the minimal cost depends on C_A (activationCost).

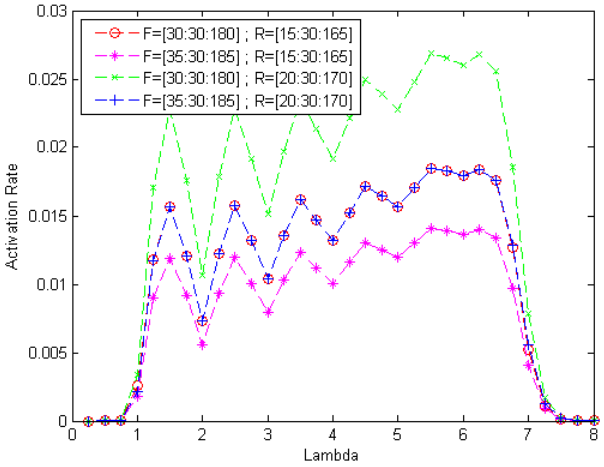

Fig. 4. Activation rate versus arrival rate (λ) : $\mu = 1$, $K = 7$ and $B = 250$ (total overlap between activation and deactivation the thresholds)

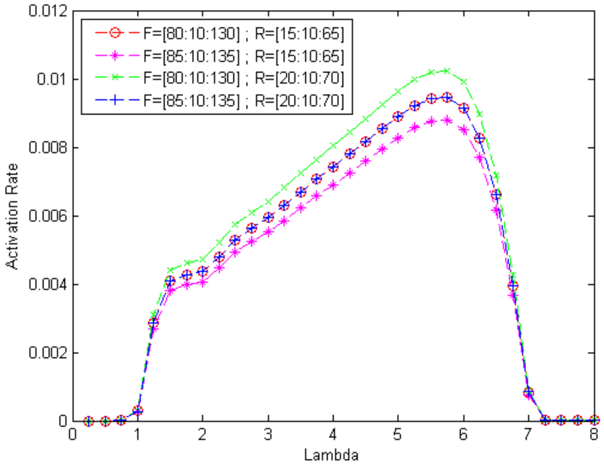

Fig. 5. Activation rate versus arrival rate (λ) : $\mu = 1$, $K = 7$ and $B = 250$ (no overlap between activation and deactivation thresholds)

VI. Conclusion

Acknowledgment

The authors would like to thank...

References

[1] 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.

[2] L.-M. Le Ny and B. Tuffin, *A simple analysis of heterogeneous multi-server threshold queues with hysteresis*. IRISA, 2000.

[3] F. Ait-Salaht and H. Castel-Taleb, "The threshold based queueing system with hysteresis for performance analysis of clouds," in *Computer, Information and Telecommunication Systems (CITS), 2015 International Conference on*. IEEE, 2015, pp. 1–5.

[4] W. Stewart, *Introduction to the numerical Solution of Markov Chains*. New Jersey: Princeton University Press, 1995.

[5] M. F. Neuts, *Matrix-geometric solutions in stochastic models: an algorithmic approach*. Courier Corporation, 1981.

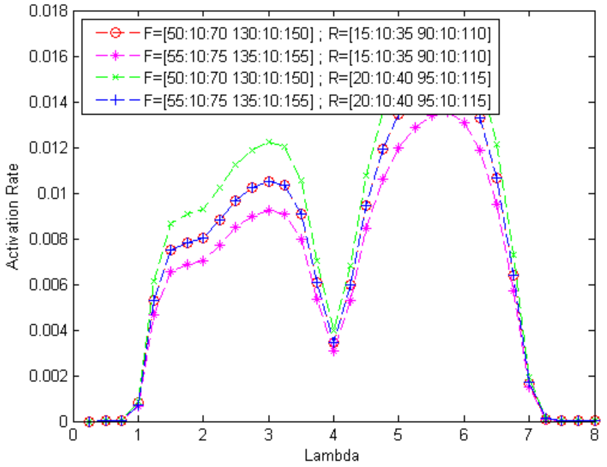

Fig. 6. Activation rate versus arrival rate (λ) : $\mu = 1$, $K = 7$ and $B = 250$ (partial overlap between the activation and deactivation thresholds)

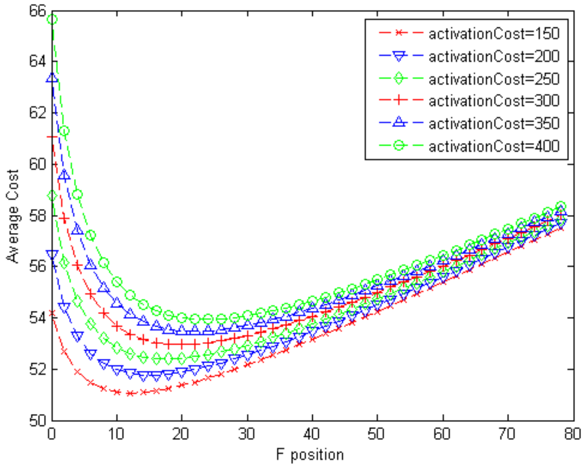

Fig. 7. Global cost versus F values : $\lambda = 4$, $\mu = 1$, $K = 7$, $B = 500$, $R = [45\ 95\ 145\ 195\ 245\ 295]$ and $F_{init} = [50\ 100\ 150\ 200\ 250\ 300]$

[6] 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.

[7] 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.