Estimating Capacity-Oriented Availability in Cloud Systems

Jamilson Dantas*, Rubens Matos†, Danilo Oliveira*, Jean Araujo‡, Carlos Melo*, and Paulo Maciel*
* Informatics Center, Federal University of Pernambuco, Recife, Brazil
† Federal Institute of Education, Science, and Technology of Sergipe, Lagarto, Brazil
‡ Academic Unit of Garanhuns, Federal Rural University of Pernambuco, Garanhuns, Brazil
{jrd, dmo4, casm3, prmm}@cin.ufpe.br*
rubens.junior@ifs.edu.br†, jean.teixeira@ufrpe.br‡

***Abstract*—Over the years, many companies have employed Cloud Computing systems as the better way as infrastructure to support their services. Keep a high availability and ensure the number of available resources based on failure events are a significant challenge to a cloud computing infrastructure planner. Due mainly to their dynamic and virtualized resources, estimate the capacity based only on the amount of available resources is not a simple task. The capacity-oriented availability has been receiving considerable attention by cloud infrastructures providers as an important activity to find the best way to increase and improve their infrastructure and services provisioning with the best cost-benefit possible. This paper presents a strategy to evaluate the capacity-oriented availability of virtual machines on a private cloud infrastructure. The proposed strategy aims to provide closed-form equations to estimate the COA metric. Models, such like continuous time Markov chains, have their evaluation availability and execution time results compared to the values obtained by the closed-form equations.**

Keywords*—Capacity-oriented availability, Closed-form equation, Cloud computing, Continuous Time Markov Chain.

I. Introduction

Over the years, many companies have employed cloud Computing systems as the better way as infrastructure to support their services. Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1].

Keep a high availability and ensure the number of available resources based on failure events are a significant challenge to cloud computing infrastructure planning. In this context, many techniques have been employed to measure system availability based on their capacity [2]. The metric that can be used to estimate this information is the capacity-oriented availability (COA) [2]. Sometimes the measure of this metric requires hard work in combination with many formalisms and techniques, such as hierarchical and compositional modeling approaches, Continuous Time Markov Chains (CTMC) [ref] and Stochastic Petri Nets (SPN) [ref] models are often used to deal with the complexity of cloud systems, especially when using many components and services [3].

Many works have been conducted to estimate the COA in cloud computing systems. ARCHANA *et al.* [4] discuss the different availability modeling approaches, the authors combines CTMC, Reliability Block Diagram (RBD) [ref] and SPN to measure the availability of the VAXcluster system. MELO *et al.* [5] evaluates the COA of nodes on a private cloud computing environment, by pointing out how many virtual machines will be available based on the amount of cloud resources lost due to failures and repair routines. Analytical, combinatorial and non-state-space models were proposed as means to find the COA values. KIM *et al.*, [6] develops hierarchical models to demonstrate the availability of two hosts system models: a non-virtualized and one virtualized system, Fault Trees (FT) [ref] were used at the upper level and homogeneous CTMC represents the sub-models in the lower level. Furthermore, most works available in the literature do not take into account a generalized form to compute the capacity-oriented availability of a large number of VMs'.

This paper presents a strategy to evaluate the capacity-oriented availability of virtual machines on a private cloud computing infrastructure. The proposed strategy aims to provide closed-form equations to estimate the COA metric. Models to estimate availability and system execution time are proposed and have their obtained results compared to the ones obtained by closed-form equations. Specifically, our main contributions are the following:

- a quantitative capacity-oriented availability evaluation of cloud computing infrastructure considering various scenarios, in different deployment configurations;
- a closed-form equation to estimate the capacity-oriented availability, in different deployment configurations, with amount of computer nodes and amount of VMs;
- development of a CTMC model for the capacity-oriented availability analysis to a specific infrastructure as a mean to validate the proposed closed-form equation.

The results obtained from the closed-form equation are validated with steady-state models comparison. However, the COA obtained results can be combined with a costs evaluation scenario to obtain the best cost-benefit aiming the higher capacity possible with reasonable expenses.

The remainder of the paper is organized as follows: Section II presents related work on system availability and COA in cloud computing environment. Section III presents a background of fundamentals concepts of dependability measure of systems analysis, as well as, brief comments about cloud computing. Section V describe the mathematical model, this model can be described to represent the easier way to compute the COA metric using a generalized closed-form equation. Section VII presents the case study and validation method proposed. Section VIII presents concluding remarks and future works.

II. Related Work

This Section presents a set of related works to availability and capacity evaluation. ARCHANA *et al.* [4] discuss the different availability modeling approaches, by combining CTMC, RBD and SPN they were able to measure the availability of the VAXcluster system. In [7] and [8] analytical models that represents a private cloud computing environment were proposed, aiming the steady-state availability evaluation. The proposed approach was applied to a video streaming service in order to demonstrate how feasible were their proposed models, by evaluating and pointing out the impact of redundancy techniques on the video stream service availability. To accomplish these tasks, closed-form equations and Markov chain models were also proposed.

KIM *et al.*, [6] developed hierarchical models to demonstrate the availability of two hosts system models: a non-virtualized and one virtualized system, FTs' were used to represent the high level model, while homogeneous CTMC are used to represent system's sub-models. CHUOB [9] *et al.* uses a CTMC model to evaluate the availability metric in a private cloud environment based in an e-government data center. In WEI *et al.* [10] RBD and SPN models are proposed for dependability evaluation of a virtual data center. The author uses a hierarchical technique based on heterogeneous models, showing the numerical results as a case study that contemplates the components of the virtual data center in failure and repair state.

In MELO *et al.* [5] the authors evaluated the COA metric and compute the steady-state availability, considering the amount of virtual machines that can be deployed at a cloud infrastructure managed by Eucalyptus platform, they also present the amount of resources that will be lost due to failures and repair routines. MATOS *et al.* [3] proposed a sensitivity analysis approach to find the parameters that deserve more attention when you want to improve the availability of virtualized server systems. The authors provide Markov Reward Models to measure the availability and suggest improvements based on the failure or repair parameters. In addition to availability, COA is also used. The authors also proposed a closed-form equation to measure the steady-state availability to a specific scenery. In DONG *et al.* [11] the authors proposed a stochastic reward net (SRN) model to analyze a virtualized system availability. The authors used a virtualized data center (VDC) with three hosts and a multiple numbers of VMs on each hosting server. The results investigate the COA regarding the number of available VMs.

All of the steady-state availability studies proposed hierarchical modeling to measure an availability scenery. Models such as CTMC, SPN or RBD are used to evaluate the availability metric, some papers like [3], [8] and [5] presents the closed-form equation to measure them. The problem related to these solutions is the state explosion with large infrastructures. Differently, from previous studies, this paper proposes a generic closed-form equation to estimating availability and capacity-oriented availability in private clouds. The generic equation provides the measurement evaluation to different possibilities and components variability in a private cloud environment. The proposed approach indicates that the use of a formalism such as CTMC model in a simple scenery makes a significant amount of states for dependability evaluation.

III. Background

A. Dependability measures to systems analysis

System dependability can be understood as the ability to deliver a specified functionality or service that can be justifiably trusted [12]. Dependability evaluation usually involves many attributes such as reliability and availability. Systems with stringent dependability requirements demand methods for detecting, correcting, avoiding and tolerate faults and failures. A failure in a large scale system can mean catastrophic losses. In the context of fault-tolerant systems, the availability is a measure of great interest. The availability metric can be understood as the probability that the system is found operational during a given period of time or has been restored after the occurrence of a failure event.

Availability can be expressed as a percentage (e.g., 99.9876%), or as Real ($\mathbb{R}$) number in the interval between $\{0 \text{ and } 1\}$ (e.g., 0.999876). The steady-state availability is the metric used when considering a long-term probability, instead of a defined time interval. It may be considered as the average availability of the system. The steady-state availability may be computed from the systems mean time between failures (MTBF) and mean time to repair (MTTR). Therefore, for a constant failure rate $\lambda = 1/MTBF$ and a constant mean repair rate $\mu = 1/MTTR$, [13], the steady-state availability is equal to:

$$A = \frac{MTBF}{MTBF + MTTR} = \frac{\frac{1}{\mu}}{\frac{1}{\lambda} + \frac{1}{\mu}} = \frac{\mu}{\lambda + \mu} \quad (1)$$

The downtime is another common dependability metric, and is defined as the total time of unavailability in a given period of time. For example, the annual downtime (in minutes) for a system is computed as expressed in Equation 2 (yearly downtime in minutes), where D is the downtime of the system and A is the availability of the system [14].

$$D = (1 - A) \times 8760h/year \times 60min/h \quad (2)$$

The mean available capacity is another important measure. It may be expressed by the capacity-oriented availability (COA) [2] [3]. This metric allows to estimate how much service the system is capable of delivering considering failure states. COA may be calculated with Markov Reward Models (MRMs) by assigning a reward to each model state corresponding to the system capacity in that condition. Another way to COA evaluation is to consider the steady-state probability in each state of the system and the amount of capacity for each state. In a private cloud system, COA may be associated with the number of virtual machines supporting by computing nodes, representing the amount of virtual machines on the system (see Equation 3).

$$COA = \frac{\sum \forall s \pi_s \times nr(s)}{TNR}, \tag{3}$$

where $nr(s)$ is the number of operational resources in state s, and TNR is the total number of resources of the infrastructure.

B. Models to dependability evaluation

Reliability block diagrams (RBDs), fault trees (FTs), stochastic Petri nets (SPNs) and Markov chains have been used to model and evaluate dependability attributes [15]. These models may be broadly classified into combinatorial and state-based models [16]. The state-based models may also be referred to as non-combinatorial, already combinatorial models can be identified as non-state based.

The CTMC is a model type based on states, that represents the system's behavior, and the occurrence of an event is given by the state transition. A transition can occur from any state to any other state and represent a simple or a compound event. The CTMCs are drawn as a directed graph and the transition label is the respective rate that makes the system transitions go from one state to another. This model allows the representation of more complex relationships between system components, such as dependencies involving sub-systems and resource constraints [15].

C. Cloud Computing and HPE Helion Eucalyptus

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. In [17] the authors states that cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services. In such a model, users access services based on their requirements without regard to where the services are hosted or how they are delivered [18].

HPE Helion Eucalyptus is a software that implements scalable IaaS-style private and hybrid clouds. Eucalyptus is an open source platform that allows you to build an Amazon Web Services (AWS)-compatible, on-premise cloud. It is designed to run on commodity hardware and provide an implementation of popular AWS-compatible services, such as EC2 (Elastic Compute Cloud) and Auto Scaling [19]. In general, Eucalyptus and other private cloud platforms use the virtualization capabilities (i.e., hypervisor) of the underlying computer system to enable flexible allocation of computing resources uncoupled from specific hardware [19][20].

The core Eucalyptus is composed by user-facing services (UI and API applications), cloud services (cloud interfaces), availability zones clusters (clusters interfaces) and Nodes [19].

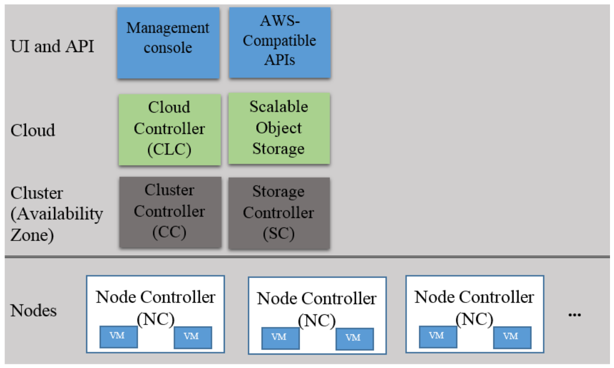

Fig. 1: Eucalyptus Architecture

Figure 1 depicts these main components, which are briefly explained as follows.

- *UI and API:* This component implements web services for the AWS-compatible service implementations and APIs and, the Management console. It represents an Easy-to-use, web-based interface that allows cloud users to provision and manages resources; it also gives cloud account administrators powerful tools to manage users, groups, and policies.
- *Cloud:* Composed by Cloud Controller hereafter ”CLC” and, Scalable Object Storage (SOS). The CLS is the front-end of the entire cloud infrastructure, exposing and managing the underlying virtualized resources (servers, network, and storage) via Amazon EC2 API [21] [19]. The (SOS) is a file-based data storage service, that is interface compatible with Amazons Simple Storage Service (S3) [22]. Eucalyptus cloud users can use SOS to stream data into and out of the cloud as well as from VMs that they have instantiated. In addition, SOS acts as a storage service for VM images.
- *Cluster (Availability Zone):* Composed by Cluster Controller (CC) and Sotorage Controller (SC). The CC usually executes on a cluster front-end machine [19], or on any machine that is able to communicate to both the nodes running Node Controllers and the machine running the CLC. The role of the CC may be summarized in three functions: determining which Node Controller will process the incoming requests for creating a VM instance (i.e., scheduling VM execution), controlling the instance virtual network overlay, and gathering/reporting information about the nodes which compose its cluster [19]. The SC provides persistent block storage for use by the virtual machine instances. It implements block-accessed network storage, similar to that provided by Amazon Elastic Block Store - EBS [23]. An elastic block storage is a Linux block device that can be remotely accessed. VM instances are not allowed to share the same EBS volume [19] [22].
- *Nodes:* Each physical node which is supposed to run VMs must have a Node Controller (NC). NCs control the execution, inspection, and termination of VM instances on the host where it runs through interaction

with the operating system running on the node and with the hypervisor. The NC fetches from SOS a copy of the VM image which will be instantiated [22] [19].

IV. Method Overview

This section introduces the set of activities to demonstrate how to create the closed-form equation for COA metric. Figure 2 presents seven activities until equation formularization. The activities are described as following: system comprehension; modeling infrastructure; generic equation; CTMC as VM infrastructure; RBD as physical infrastructure; extract closed-form equation and results.

It is important to stress that the main goal of the method overview is to demonstrate step by step how we get the closed-form equation. Initially, **system comprehension** represents the activity to understand how the cloud will be work. In this step, we define how many VMs' will be executed in the infrastructure and how many physical machines will be used.

The **generic equation** in the first step, is composed of two activities: **CTMC as VM infrastructure** and **RBD as physical infrastructure**. The choice of these formalisms gives us the possibility to extract the closed-form equation. It is easier to create a way to generalize the equation when we have the base of the mathematical formalism. The **CTMC as VM infrastructure** represents the logical sequence of failure and repair activities inside the virtual machines infrastructure. In another side, the physical machines are organized in parallel order, in this way, we chose the RBD formalism. This activity is represented by **RBD as physical infrastructure**. Next, **extract closed-form equation and generalization**, in this step we extract closed-form equation with the Mathematica tool [24] and, we tried to find a way of generalizing the equation for all kind of models.

Evaluating results activity represents the results obtained by steady-state model and closed-form equation. Next, **model validation** represents the validation method to validate the proposed mathematical form. In this step, we compare the results with steady-state model and closed-form equation. If the error was minimal we make draw final results, if it is not we have to **performing improvements** and return to make some improvements in the generic equation.

V. Closed-form Generalization Expression to Measure COA Metric

We generalize a closed form expression for the computer nodes that supporting by any number of virtual machines considering a cloud computing infrastructure.

A. Infrastructure Example

The system may be composed of two physical machines, PM1 and PM2, as is shown in Figure 3. Each physical machine support two virtual machines (VMs). The number of VMs in the system represents the system capacity.

The system has the capability of four VMs. We can consider the faults that can happen in each VM. Thus, the number of VMs that can be running in the system expressed by NVM is given by Equation 4.

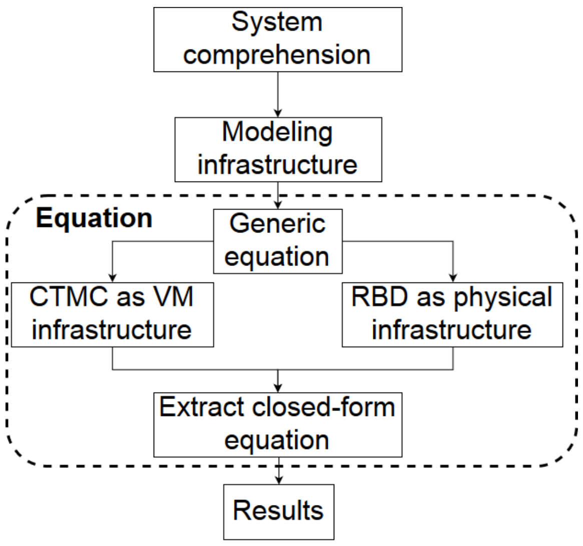
System comprehension
Modeling infrastructure
Equation
Generic equation
CTMC as VM infrastructure
RBD as physical infrastructure
Extract closed-form equation
Results

Fig. 2: Closed-form equation methodology

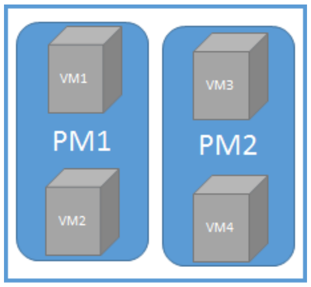

Fig. 3: Example of computer nodes

$$NVM = \sum_{i=0}^{n} [P_i \times i] \tag{4}$$

Thus, for the example shown in Figure 3, we can compute the number of running VMs in the system by Equation 5. However, the probability for each state must be known. We made a CTMC model to compute the probability for each state, the model is illustrated in Figure 4. In the model, the states 4, 3, 2, 1 and 0 represents the probability of amount of running VMs on the system, 4 VMs, 3 VMs, 2 VMs 1 VM and 0 VMs respectively.

$$NVM = [P_4 \times 4] + [P_3 \times 3] + [P_2 \times 2] + [P_1] \tag{5}$$

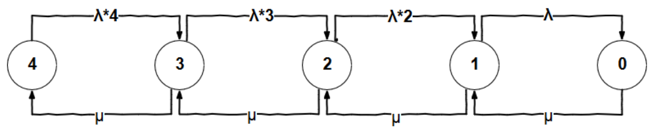

Fig. 4: CTMC model to represent virtual machines

The probability for each state may be given by Equations 6, 7, 8 and 9. The Equations denotes the probability of system having 4, 3, 2 or 1 running VMs.

$$P_4 = \frac{\mu^4}{24\lambda^4 + 24\lambda^3\mu + 12\lambda^2\mu^2 + 4\lambda\mu^3 + \mu^4} \tag{6}$$

$$P_3 = \frac{4\lambda\mu^3}{24\lambda^4 + 24\lambda^3\mu + 12\lambda^2\mu^2 + 4\lambda\mu^3 + \mu^4} \tag{7}$$

$$P_2 = \frac{12\lambda^2\mu^2}{24\lambda^4 + 24\lambda^3\mu + 12\lambda^2\mu^2 + 4\lambda\mu^3 + \mu^4} \tag{8}$$

$$P_1 = \frac{24\lambda^3\mu}{24\lambda^4 + 24\lambda^3\mu + 12\lambda^2\mu^2 + 4\lambda\mu^3 + \mu^4} \tag{9}$$

B. *Availability Model*

Considering the example described above and the cloud computing model shown in Section III, we conducted the study with compute nodes organized in parallel, so it is easier to made model using reliability block diagram - RBD (see Figure 5). In this context, the availability model is represented by the probability at least one functional block. Thus, the capacity of the system would fall by half in the case of the occurrence of the failure of one of the blocks.

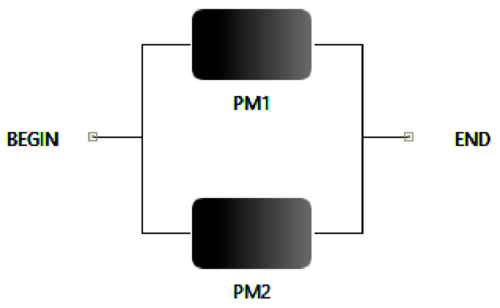

Fig. 5: RBD availability model

Considering the models presented previously, we can generate the closed-form expressions for the COA shown in the Equation 10.

$$\begin{aligned} COA = (((\frac{\mu_{pm}}{\mu_{pm} + \lambda_{pm}})^2) \times ([P_4 \times 4] + [P_3 \times 3] + \\ [P_2 \times 2] + [P_1])) + \\ ((\frac{2\lambda_{pm}\mu_{pm}}{(\lambda_{pm} + \mu_{pm})^2}) \times ([P_2 \times 2] + [P_1])). \end{aligned} \tag{10}$$

C. *Generalized closed-form Equation for COA measure*

Seeing the similarity of the probability of each state denoted in Equations 6, 7, 8 and 9 we can generalize the mathematical problem. Therefore, the COA for any configuration of cloud infrastructure can be found following a generic closed-form equation denoted by Equation 11.

$$\begin{aligned} COA = \frac{1}{n}\left(\left(\sum_{k=0}^{n}((A_s)^p\, VMs[k,n](n-k))\right) + \right. \\ \left.\left(\sum_{j=1}^{p-1}\sum_{k=0}^{NVj}(A_s(j,p) - A_s(j+1,p))(VMs[k,NVj])(NVj-k)\right)\right) \end{aligned} \tag{11}$$

Where,

$$VMs[K\_, n\_] := \left(\frac{\frac{n!}{(n-k)!}\mu^{(n-k)}\lambda_{vm}^{k})}{\sum_{i=0}^{n}\frac{n!}{i!}\lambda_{vm}^{n-i}\mu^{i}}\right) \tag{12}$$

- $VMs[K\_, n\_]$ is a function which defines the probability of state k from n VMs available (P_k);
- NVj represents the number of VMs (NV) per physical machine (j).

We can assume that the components of the cluster are independent and identical. Therefore, all the components have the same failure and repair distribution. In the case of components in a $k-out-of-n$, the availability of the system with such a configuration can be evaluated using the binomial distribution, or:

$$A_s(j,p) = \sum_{i=j}^{p}\binom{p}{i}A^i(1-A)^{p-1} \tag{13}$$

where,

- p is the total number of physical machine in parallel;
- j is the minimum number of units required for system success;
- A is the availability of each unit.

VI. Availability models and Cost Model

This section presents CTMC models and cost equation for representing a cloud scenario. The evaluated architecture was described in the section above.

A. *Models description*

The availability models may present by two representative models for the cloud infrastructure considering two physical machines and four virtual machines at the system. In this way, we are quantifying availability, downtime and COA metrics.

Figure 6 describe a CTMC model composed by two compute nodes and four VMs running. The CTMC comprises the twelve states, the description of each state are represented in Table I. The Shaded states UD0, UU0, DU0 and DD0 represents the unavailable states, in other words, the system does not have available VM.

The notation represents the system condition for each state. The computer node may be up (U) or down (D). If there are two computer nodes available the system may have four, three,

TABLE I: States descriptions of the CTMC model

States label	Description
UU4	Two compute nodes UPs, and four VMs UPs
UU3	Two compute nodes UPs, and three VMs UPs
UU2	Two compute nodes UPs, and two VMs UPs
UU1	Two compute nodes UPs, and one VM UP
UU0	Two compute nodes UPs, and zero VMs UPs
UD2	One compute node UP, and two VMs UPs
UD1	One compute node UP, and one VM UP
UD0	One compute node UP, and zero VMs UPs
DU2	One compute node UP, and two VMs UPs
DU1	One compute node UP, and one VM UP
DU0	One compute node UP, and zero VMs UPs
DD0	Not service

two, one or zero VMs available (UU4, UU3, UU2, UU1 or UU0 respectively). If there is only one computer node available the system may have two, one or zero VMs available (UD2, UD1, and UD0 or DU2, DU1 and DU0). The state DD0 represents the failure of all computers nodes, consequently does not have the possibility to instantiate VM in the system.

We consider the failure and repair activities of the system, as well as, failure and repair activities of VMs.

The compute node failure is an event that occurs when some component does not run to the correct form. The failure rates of the computer nodes are represented by λ_{PM} and λ_{PM2}, while μ_{PM} represents the repair rate of compute nodes. We assume that the VMs are identical. However, the failure and repair rates are expressed by λ_{VM} and μ_{VM} respectively.

The CTMC model described in Figure 6 represents all of the states of the system, starting from the assumption of we having two computer nodes we can minimize the model based on the premise that we represent the failure of computer nodes, not specifying which of the computer nodes fall. This equivalent model is described in Figure 7.

The CTMC model may be reduced from twelve to nine states, the description of each state are represented in Table II. The Shaded states 00, 10 and 20 represents the unavailable states.

TABLE II: States descriptions of the second CTMC model

States label	Description
24	Two compute nodes UPs, and four VMs UPs
23	Two compute nodes UPs, and three VMs UPs
22	Two compute nodes UPs, and two VMs UPs
21	Two compute nodes UPs, and one VM UP
20	Two compute nodes UPs, and zero VMs UPs
12	One compute node UP, and two VMs UPs
11	One compute node UP, and one VM UP
10	One compute node UP, and zero VMs UPs
00	Not service

The notation is folows the same idea of the first CTMC model. The system may have two (2), one (1) or zero (0) available computer nodes. If there are two computer nodes available the system may have four, three, two, one or zero VMs available (24, 23, 22, 21 or 20 respectively). If there is only one computer node available the system may have two, one or zero VMs available (12, 11, and 10). Thus, the first number represents the computer nodes available and the second number represents the amount of VMs available. The final state 00 represents the failure of all computers nodes, consequently, does not have the possibility to instantiate VMs in the system.

B. Cost model

We adopted for estimating the cost of equipment a closed-form equation. It is important to stress that we consider only the acquisition cost of each physical machine. Assuming NPM as the number of physical machines and CF as the effective cost of each physical machine we can compute the cost of the infrastructure CI as:

$$CI = NPM \times CF \tag{14}$$

VII. Case study

We conducted a case study to demonstrated the viability of the mathematical form. In the first time, we compare COA results with the state-state CTMC models and the closed-form equation to demonstrate the validation of the equation. In the second scenario, we demonstrated the viability of the closed-form equation with a wide range of equipment. We also compare relative COA with effective cost in each scenario.

A. Validation method

The validation method focus on availability and COA measures. The main goal of this study is to demonstrate the viability of the closed-form equation. In the first case, we made two models (described above). It is important to stress that the models were made based on the most simple scenario, in this way we can observe that the models present a significant number of states, which may represent an explosion of states in the case of a scenario with a larger number of computer nodes.

Table III describe the input parameters for CTMC models presented in the subsection *Models Descriptions* in Section VI. These values were obtained from [25] [8] [7].

TABLE III: Input parameters for CTMC models

Parameters	Description	Values (h^{-1})
$\lambda_{PM1} = \lambda_{PM2}$	Physical machine failure rate	1/8760
λ_{VM}	Virtual Machine failure rate	1/2880
μ_{PM} = 1/1	Physical machine repair rate	1
$\mu_{VM} = 1/1$	Virtual Machine repair rate	1

We computed the availability measures for the architecture described in Section V, using the mentioned input parameters. The results are shown in Table IV, including steady-state availability and COA. The results showed a comparison between two generic models and closed-form equation.

The results show the similarity of the closed-form equation and the steady-state models. The relative difference of the availability results at the equivalent model happens due to the abstraction of some states.

Another significant result is the execution time comparison between SPN model and closed-form equation. The execution time represents the time spent to calc the COA for each scenario. We chose two simple scenarios. The first one (S1) is composed of one physical machine and four VMs' per physical

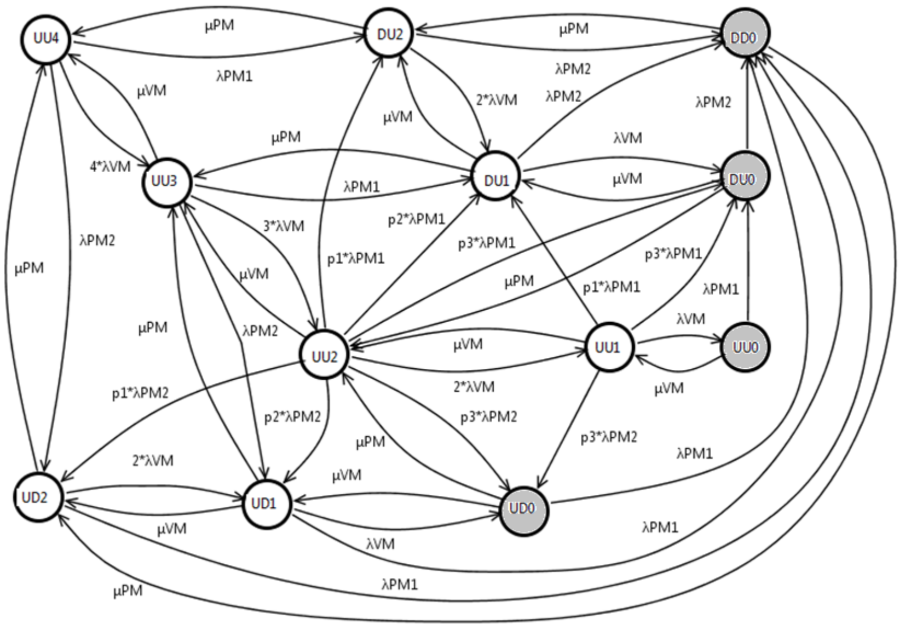

Fig. 6: Complete CTMC model

TABLE IV: Availability results

Model Description	Availability	COA
Complete CTMC model	0.999999987	0.9995384300
Equivalent CTMC model	0.999999974	0.9995385138
Closed-form Equation	0.999999987	0.9995384342

machine, totalizing four VMs' in the system. The second (S2) consists of two physical machines and four VMs' per physical machine, totalizing eight VMs' in the system.

Figure 8 shows the difference between SPN model and closed-form equation for the execution time. The two simple scenarios in SPN model represents the major discrepancies in compare to closed-form equation. It is important to stress that the another possible scenario with three physical machines and four VMs' per physical machine, totalizing twelve VMs' had an execution time result of the 76.418 seconds, whereas the same scenario has 0.03 seconds in the execution time in the closed-form equation. This result gives us almost 80% increase in execution time when we use SPN model. When we use more than four physical machines in SPN formalism, it can cause state explosion, so our approach is valid for the purpose.

B. Cost evaluation

Figure 9 denote the system COA based on some nodes in the system. It is important to stress that we consider the same kind and the same number of VMs per physical machine. In this way, we estimate five scenarios with 2, 10, 50, 100 and 200 nodes, each node with 2, 4 and 10 VMs per physical machine respectively. The decreasing behavior happens because there are the weigh between system availability and the number of VMs in the system.

The results shown in Figure 9 denote an important factor of the behavior of the system, in other words when we consider a large number of VMs in the same physical infrastructure we have worse results due to the probability of failure of more components (VMs per physical machine).

Based on this information we conducted a study to estimate the best choice of infrastructure considering the capacity and cost. Table V presents the evaluated scenarios. We divided into five scenarios. The adopted scenarios consist in the same configuration of the physical machine (processor model E3-1270 V5 four cores and 32 GB RAM), the unit cost of this physical machine is US$ 2,395.98.

TABLE V: Cost and COA results

Scenario	Number of Computers	Amount of VMs	Total Cost (US$)	COA
A1	1	4	2,395.98	0.999538
A2	5	20	11,979.90	0.999536
A3	10	40	23,959.80	0.999534
A4	50	200	119,799.00	0.999513
A5	100	400	239,598.00	0.999483

To determine the architectures with the best cost × COA relationship we must define what is the benefit we are seeking. In other words, we expect the ideal value between COA and cost. But they are different greatness, so wee need to normalize and put them in the same interval: 0 and 1. Equation 15 presents the normalization process.

$$N_{xy} = \frac{V_x - MinV_x}{MaxV_x - MinV_x} \qquad (15)$$

where x may represent the $cost$ and COA value, y represent the architectures: A1, A2, A3, A4 and A5. V_x represent the current value for the architectures, $MinV_x$ represents the minimum value for the architectures and $MaxV_x$ represents the maximum value for the architectures.

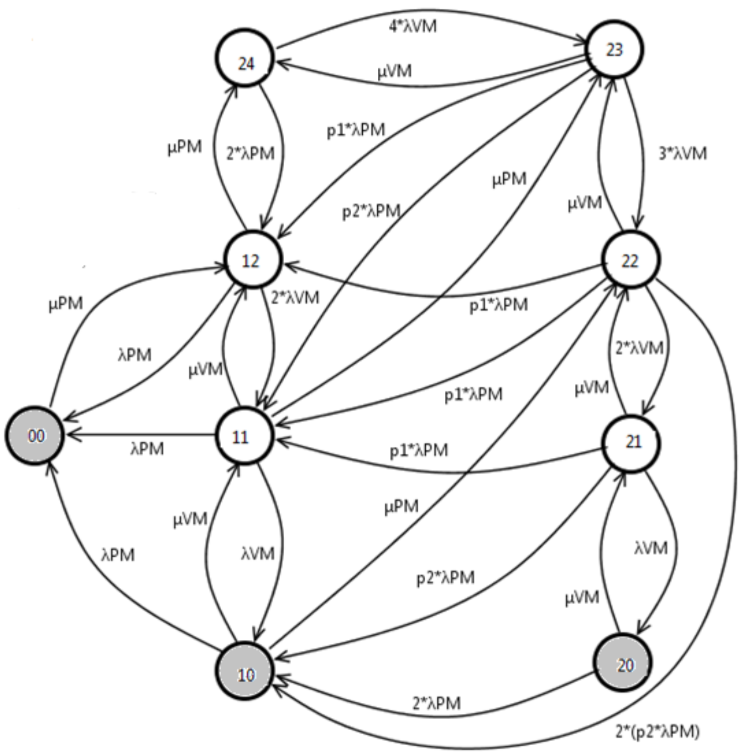

Fig. 7: Equivalent CTMC model

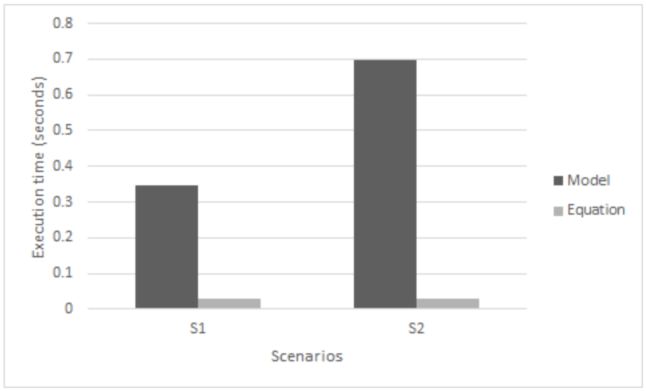

Fig. 8: Execution time results: A comparison between SPN model and closed-form equation.

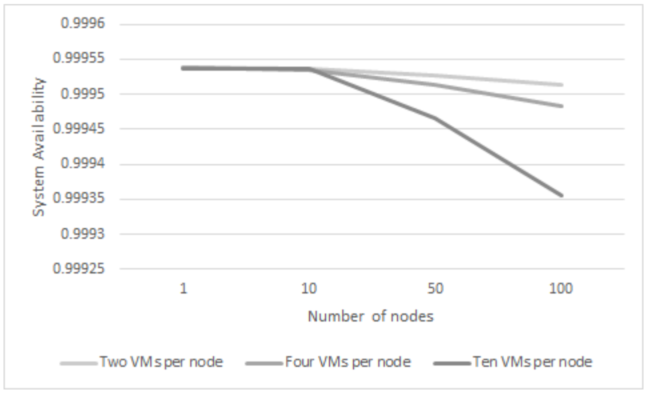

Fig. 9: Capacity-Oriented Availability

Now in possession of the normalized cost and COA values, we must relate them somehow, so we use the Euclidean distance [ref], the architecture with less distance from the origin tends to be the best one. Equation 16 shows how we perform the calculation of distances.

$$D_y = \sqrt{(N_{xy})^2 + (N_{xy})^2} \tag{16}$$

where N_{xy} may represents the Normalized Cost x for the architecture y and N_{xy} the Normalized COA x for the architecture y.

Figure 10 shows the relationship between cost and COA measures. The figure denotes the A4 architecture has the best relationship between cost and COA. This architecture doesn't have the higher COA value or the worst price, but thanks to the normalization and distance methods we are able to define it as the best architecture for our goal. The last place goes for architecture A1 and A5, with the worse values.

Another important result is to define the how long the company can have the return of the invested amount. It can be specified by the kind of VM is used in the infrastructure and the monthly cost of the equipment. In another word, we can define the kind of instance applied in all of our infrastructures as of the small VM type, in this way we used the VM with 1 CPU core and memory 1.7 GiB. This configuration gives us a value of US$ 206.00 for 1 Yr and US$ 618.00 for 3 Yr, by Amazon simple monthly calculator [26]. Figure

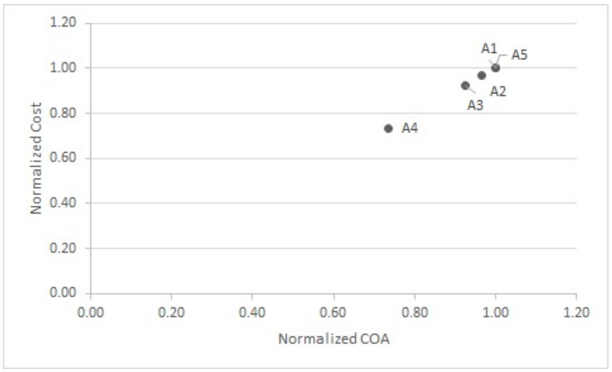

Fig. 10: Comparative Cost and COA

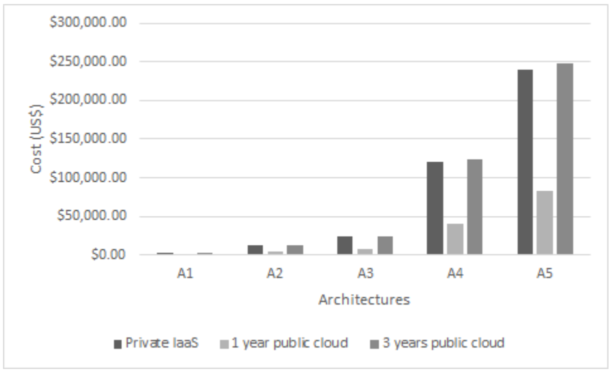

Fig. 11: Required cost of 1 Yr and 3 Yr: A comparison between a private IaaS and a public cloud

11 is a comparison of the private IaaS between one year and three years of public cloud, we use the same monthly cost of Amazon, in this way, the required time to the initial investment return gives us three years. It is important to stress that the initial cost adopted in this study is the cost of physical computer equipment.

VIII. Conclusion and Future works

This work proposed a closed-form equation to estimate the capacity-oriented availability (COA) in cloud systems. The formalisms such as Stochastic Petri Nets (SPN) and Continuous Time Markov Chain (CTMC) gives us the subsidies to evaluate the availability, performance and COA metrics. The cloud infrastructure used is based on Eucalyptus platform and the cloud model configured is the smaller as possible, one node and four VMs. The evaluation provides the possibilities to calculate the COA in large infrastructures. Whereas SPN and CTMC models can cause state explosion.

This work also provided a case study to demonstrate the planning for cloud infrastructure comparing the COA and Cost measures in order to illustrate the feasibility of the proposed equation.

As future work, we intend to evaluate the performance using a service type of infrastructure. We intend to use the COA, cost and performance measures to planning the best architecture combination to particular service. We also intend to use optimization model.

References

[1] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[2] D. Heimann, N. Mittal, and K. Trivedi, "Dependability modeling for computer systems," in *Reliability and Maintainability Symposium, 1991. Proceedings., Annual*. IEEE, 1991, pp. 120–128.

[3] R. Matos, P. R. M. Maciel, F. Machida, D. S. Kim, and K. S. Trivedi, "Sensitivity analysis of server virtualized system availability," *Reliability, IEEE Transactions on*, vol. 61, no. 4, pp. 994–1006, 2012.

[4] A. Sathaye, S. Ramani, and K. S. Trivedi, "Availability models in practice," in *Proc. of intl. workshop on fault-tolerant control and computing (FTCC-1)*, 2000.

[5] C. Melo, R. Matos, J. Dantas, and P. Maciel, "Capacity-oriented availability model for resources estimation on private cloud infrastructure," in *Dependable Computing (PRDC), 2017 IEEE 22nd Pacific Rim International Symposium on*. IEEE, 2017, pp. 255–260.

[6] D. S. Kim, F. Machida, and K. S. Trivedi, "Availability modeling and analysis of a virtualized system," in *Dependable Computing, 2009. PRDC '09. 15th IEEE Pacific Rim International Symposium on*, Nov 2009, pp. 365–371.

[7] R. M. De Melo, M. C. Bezerra, J. Dantas, R. Matos, I. J. De Melo Filho, and P. Maciel, "Redundant vod streaming service in a private cloud: availability modeling and sensitivity analysis," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[8] M. C. Bezerra, R. Melo, J. Dantas, P. Maciel, and F. Vieira, "Availability modeling and analysis of a vod service for eucalyptus platform," in *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3779–3784.

[9] S. Chuob, M. Pokharel, and J. S. Park, "Modeling and analysis of cloud computing availability based on eucalyptus platform for e-government data center," in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011 Fifth International Conference on*. IEEE, 2011, pp. 289–296.

[10] B. Wei, C. Lin, and X. Kong, "Dependability modeling and analysis for the virtual data center of cloud computing," in *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*. IEEE, 2011, pp. 784–789.

[11] D. S. Kim, J. B. Hong, T. A. Nguyen, F. Machida, J. S. Park, and K. S. Trivedi, "Availability modeling and analysis of a virtualized system using stochastic reward nets," in *Computer and Information Technology (CIT), 2016 IEEE International Conference on*. IEEE, 2016, pp. 210–218.

[12] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *Dependable and Secure Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 11–33, 2004.

[13] P. O'Connor and A. Kleyner, *Practical reliability engineering*. John Wiley & Sons, 2011.

[14] A. Sathaye, S. Ramani, and K. S. Trivedi, "Availability models in practice," in *Proc. of Intl. Workshop on Fault-Tolerant Control and Computing (FTCC-1)*, 2000.

[15] P. Maciel, K. S. Trivedi, R. Matias, and D. S. Kim, "Dependability modeling," in *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*. Hershey: IGI Global, 2011.

[16] M. Malhotra and K. S. Trivedi, "Power-hierarchy of dependability-model types," *Reliability, IEEE Transactions on*, vol. 43, no. 3, pp. 493–502, 1994.

[17] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[18] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation computer systems*, vol. 25, no. 6, pp. 599–616, 2009.

[19] H. P. Enterprise, "General purpose reference architecture: Hpe helion eucalyptus," online.

[20] ——. (2017) Eucalyptus: Eucalyptus 4.4.0 administration guide. Available in: https://docs.eucalyptus.com/eucalyptus/.

[21] J. Carolan, S. Gaede, J. Baty, G. Brunette, A. Licht, J. Remmell, L. Tucker, and J. Weise, "Introduction to cloud computing architecture," *White Paper, 1st edn. Sun Micro Systems Inc*, 2009.

[22] H. Packard, "General purpose reference architecture: Hpe helion eucalyptus," 2016, online:.

[23] Amazon. (2016) Amazon elastic block store (ebs). Amazon.com, Inc. Available in: http://aws.amazon.com/ebs.

[24] Wolfram. (2017) Wolfram mathematica. Wolfram.com,. Available in: https://www.wolfram.com/mathematica/.

[25] J. Dantas, R. Matos, J. Araujo, and P. Maciel, "An availability model for eucalyptus platform: An analysis of warm-standy replication mechanism," in *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1664–1669.

[26] Amazon. (2017) Simple monthly calculator. amazon.com,. Available in: https://calculator.s3.amazonaws.com/.