Noname manuscript No.
(will be inserted by the editor)

Identification of Critical Factors for the Availability of a Disaster Tolerant Cloud Computing Systems

Bruno Silva · Rubens Matos · Eduardo Tavares · Paulo Maciel · Armin Zimmermann

the date of receipt and acceptance should be inserted later

Abstract Due to the dependence on Internet-based services, many efforts have been conceived to mitigate the impact of disasters on service provision. In this context, cloud computing has become an interesting alternative for implementing disaster tolerant services due to its resource on-demand and pay-as-you-go models. This paper proposes a sensitivity analysis approach to assess the parameters that most impact the availability of cloud data centers, taking into account disaster occurrence, hardware and software failures, as well as recovery actions. The analysis adopts continuous-time Markov chains, and the results indicate that disaster issues should not be neglected. Hardware failure rate and time for migration of VMs are the critical factors pointed out for the system modeled in our analysis. Moreover, the location where data centers are placed has a significant impact on system availability, due to time for migrating VMs from a backup server.

Keywords Sensitivity analysis · cloud computing · continuous time Markov chains.

1 Introduction

Internet-based services have become critical to several businesses in which many aspects of our lives depend on (e.g., online banking, collaborative work, video conferencing). Business continuity is a remarkable property and a real

Bruno Silva and Rubens Matos and Eduardo Tavares and Paulo Maciel
Federal University of Pernambuco
Tel.: +558121268430
E-mail: {bs, rsmj, eagt, prmm}@cin.ufpe.br

Armin Zimmermann
Technische Universität Ilmenau
Tel.: +493677694420
E-mail: Armin.Zimmermann@TU-Ilmenau.De

concern for many companies, since service disruption may cause huge revenue and market share losses. Thus, many organizations have relied on disaster tolerant services to avert disasters from generating service outages [1]. Disasters contemplate both natural and manmade events. Hurricanes, flooding, and earthquakes are examples of natural causes of disasters. Manmade disasters might occur due to an intentionally set fire, or even an unintentional event, such as a car accident that leaded to a power cut affecting Amazon's data center [2].

Over the last years, cloud computing has turned into an interesting alternative for implementing disaster tolerant services due to its resource on-demand and pay-as-you-go models [3]. More specifically, additional resources, such as virtual machines (VMs), are only allocated when a disaster takes place, and the automated virtual platform also performs a transparent recovery and minimizes the time to restore the service.

Availability is a prominent indicator to assess cloud provider's quality-of-service (QoS). This metric takes into account the effects of failure/recovery behavior of data center systems. For prominent cloud providers, the quality level is regulated by adopting a Service Level Agreement (SLA) which specifies, for instance, the maximum downtime per year. Penalties may be applied if the defined quality level is not satisfied. Thus, to meet SLA requirements, cloud providers need to evaluate the availability level of their environment, considering also the possibility of disasters.

A disaster recovery plan requires the utilization of different data centers located far enough apart to mitigate the effects of unforeseen citywide disasters (e.g., earthquakes) [4]. If multiple data centers are located in different geographical locations (considering disaster independent places), the availability level of the whole system will improve. On the other hand, VM migration time increases due to distance between data centers.

Consequently, dependability evaluation [5] is of utmost importance to assess availability in distributed cloud systems with disaster recovery mechanisms. Dependability takes into account techniques and modeling approaches, which evaluate the impact of failures on service provision. In this context, sensitivity analysis [6,7] is a prominent technique that allows the identification of parameters that most influence availability related measures.

Sensitivity analysis is often adopted to evaluate how "sensitive" a metric is to variations in the value of a parameter (parametric) or to changes in the model (structural). Traditionally, parametric sensitivity analysis is performed by a discrete variation of input parameters over their value ranges, and graphing the effects on output measures. Another technique for performing parametric sensitivity analysis is the differential sensitivity analysis. This approach calculates the partial derivatives of the measure of interest with respect to each input parameter. The main advantages of differential sensitivity analysis is the reduced computation time when compared to other methods (e.g., discrete method) [8].

This work presents an approach based on differential parametric sensitivity analysis to identify the parameters that most impact the availability of

cloud data centers. The analysis is based on continuous time Markov chains (CTMC) for assessing system availability considering hardware and software failures, disaster occurrence, recovery, and VM migration. Using the proposed approach, cloud providers can determine the parameters that deserve more attention in order to meet SLA requirements. We also evaluate the distance between data centers and backup server, which indicates a significant influence on system availability.

The paper is organized as follows. Section 2 highlights the related works, and Section 3 presents important concepts. Section 4 describes the cloud computing system adopted in this work. Section 5 presents the case study, which includes the analytical model and results. Finally, Section 6 concludes this paper and introduces future works.

2 Related Work

Over the last years, some authors have been devoting efforts to study dependability issues on cloud computing systems. Longo et al. [9] proposed an approach for availability analysis of cloud computing systems based on Petri nets and Markov chains. The authors also developed closed-form equations and demonstrated that their approach can scale for large systems. In [10], a performability analysis for cloud systems is presented. The authors quantify the effects of variations in workload, failure rate and system capacity on service quality. In [11], the authors investigate the software aging effects on Eucalyptus framework [12], and they also propose a strategy to mitigate such issues during system execution.

Bradford et al [13] describe a system design approach for supporting transparent migration of VMs adopting local storage for their persistent state. The approach is transparent to the migrated VM, and it does not interrupt open network connections during VM migration. In [14], the authors present a case study that quantifies the effect of VM live migrations in the performance of an Internet application. Such study helps data center designers to plan environments in which SLAs determine a desired level for the specified metrics, such as service availability and responsiveness. Dantas et al. [15] present a study on warm-standby mechanisms in Eucalyptus-based private clouds. Their results demonstrate that replacing machines by more reliable counterparts would not produce significant improvements in system availability, whereas some techniques of fault-tolerance can indeed increase dependability levels.

In [16], the authors present an sensitivity analysis for a variant of Hadoop Distributed File System (HDFS), which contemplates energy saving techniques. The proposed system divides the cluster data in Hot and Cold Zones. In this approach, data that present long periods (i.e., several days) of idleness are allocated in the Cold Zone. That analysis also shows the energy-saving behavior considering the variation of file system parameters.

The work presented in [17] adopts a two-level hierarchical modeling approach for virtualized systems which uses fault trees in the upper level, and

CTMC in the lower level. The support for sensitivity analysis in these analytical models is important for detecting bottlenecks in system availability. In [18], the authors show four different sensitivity analysis techniques to determine the parameters that cause the greatest impact on the availability of a mobile cloud system. The authors use a combined evaluation of results from different analysis techniques to deal with the evaluation of the system. Their results show that the availability can be effectively improved by changing a reduced set of parameters.

Unlike previous works, this paper proposes a differential parametric sensitivity analysis for evaluating cloud computing systems deployed into geographically distributed data centers, considering VM migration and disaster occurrence. The proposed approach also adopts continuous time Markov chains (CTMC) [5] to evaluate availability and to identify parameters that are critical for such a metric.

3 Background

This section presents fundamental concepts for providing a better understanding of this paper.

3.1 Dependability

Dependability is the capacity of a system to offer a service in a reliable way [5]. An important concept is system failure, which happens when the system stops providing the respective service. A fault concerns the failure of a system component (or subsystem), which may cause other faults or system failure. Dependability comprises several metrics, and, in this work, the metrics of interest are:

- Reliability: probability of a system executing its functions without failures for a specified period of time [5]: $R(t) = P\{T \geq t\}$, in which T is the random variable representing the time to failure of the system (or a single component, depending on the target analysis).
- Mean Time to Failure: $MTTF = \int_0^\infty R(t)dt$;
- Mean Time to Repair: $MTTR = \int_0^\infty 1 - F_m(t)$. $F_m(t)$ is the cumulative distribution function representing the probability that a repair will occur within time t;
- Availability: probability of a system being in a working condition. It considers the alternation of operational and non-operational states [5]. Steady-state availability (A) is commonly adopted, and it is represented by $A = MTTF/(MTTF + MTTR)$.

State-based analytical models are very prominent in estimating dependability metrics, since they allow the representation of complex interactions between components, such as dynamic redundancy mechanisms. In general, they

model a system behavior by its states and event occurrences [5]. However, state-based models may suffer from state space explosion. Continuous-time Markov chains (CTMC) as well as stochastic Petri nets (SPN) are prominent state-based models [5].

CTMC is graphically represented by a labelled transition system (e.g., Figure 1), in which edges are annotated with rates. Q denotes its infinitesimal generator matrix, in which q_{ij} is the rate from state i to state j; $q_{ii} = -\sum_{j=1,j\neq i}^{n} qij$; and n is the amount of states. Steady-state probabilities (π) are calculated using a system of linear equations

$$\begin{aligned} \pi Q &= 0, \\ \sum_{i=1}^{n} \pi(i) &= 1, \end{aligned} \tag{1}$$

in which $\pi(i)$ represents the probability of state i.

3.2 Sensitivity Analysis of Analytical Models

The creation of analytical models enables the evaluation of many system attributes, such as those related to performance, dependability, and energy consumption. One important activity of analytical modeling is the assessment that some parameters may have on a metric of interest, usually referred to as sensitivity analysis [6,7]. The main aim is to predict the effect on outputs with respect to variations in input parameters [6].

When dealing with analytic models such as Markov models, stochastic Petri nets, and queueing networks, parametric sensitivity analysis is particularly important to find performance, reliability, and availability bottlenecks, or in guiding an optimization process [19,8]. The parametric sensitivity analysis may be done by repeatedly varying one parameter at a time, while keeping the others fixed. When applying this method, a sensitivity ranking is obtained by noting the corresponding changes in the measure of interest. The slopes of lines in a scatter plot are commonly used to determine the difference of influence from one parameter to another.

Numerical sensitivity indices can provide a view that is more accurate than a graphical analysis of the variation of one parameter at a time. Differential sensitivity analysis, also called direct method, enable the identification of a measure's sensitivity index for each parameter of the model. Birnbaum's Component Importance [20] – also known as Reliability Importance – is a well known sensitivity index which is based on the idea of differential analysis, but it is specific for reliability evaluation. In the context of availability assessment, Barabady and Kumar [21] propose an availability importance measure which is also based on differentiation. In a general sense, differential sensitivity analysis is performed by computing the partial derivatives of the measure of interest with respect to each input parameter. Subsequently, the sensitivity of

a given measure Y, which depends on a specific parameter λ, is computed as in Equation 2, or 3 for a scaled sensitivity.

$$S_\lambda(Y) = \frac{\partial Y}{\partial \lambda}, \tag{2}$$

$$SS_\lambda(Y) = \frac{\lambda}{Y}\frac{\partial Y}{\partial \lambda}. \tag{3}$$

A scaled (also called relative) sensitivity index may be employed to counterbalance the effect caused by large differences between absolute parameter values. The property just mentioned may be explained by the indirect relation between scaled sensitivity functions and logarithms, mentioned in [6] and shown in Equation 4.

$$\frac{\partial\, ln\, Y}{\partial\, ln\, \lambda} = \frac{\frac{\partial\, Y}{Y}}{\frac{\partial\, \lambda}{\lambda}} = S_\lambda(Y)\frac{\lambda}{Y} = SS_\lambda(Y) \tag{4}$$

There is a case in which the logarithm may be applied only in the parameters, because the measure Y might already be a logarithmic measure. In such a case, the scaled sensitivity will be described by Equation 5. In this kind of scaling, the sensitivity is multiplied only by the parameter, instead of the ratio between parameter and measure of interest. $\widetilde{SS_\lambda}(Y)$ is also referred to as a semi-relative sensitivity function.

$$\widetilde{SS_\lambda}(Y) = \frac{\partial\, Y}{\partial\, ln\lambda} = \frac{\partial\, Y}{\partial\, \lambda/\lambda} = \lambda S_\lambda(Y) \tag{5}$$

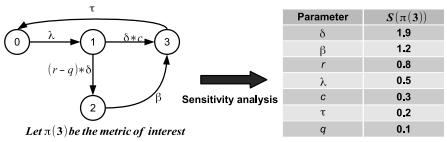

Fig. 1: Representation of sensitivity ranking for a CTMC model

Many papers have already described how to apply parametric sensitivity analysis through partial derivatives in a variety of analytic models, including CTMC [19] [8] [22], Markov reward models [23], generalized stochastic Petri nets [24], and queueing networks [25]. Those approaches are helpful for obtaining the sensitivity ranking in cases when there is no direct closed-form equations for the measure of interest. Figure 1 depicts a sensitivity ranking for a specific CTMC measure. The parameters which have the highest impact on the measure of interest are located in the top positions of the ranking, since the sensitivity indices are in decreasing order. The ranking provides a simple and accurate view of the importance of each parameter, enabling objective comparisons and decision making.

3.3 Disaster Recovery and Tolerance

Disaster recovery is the practice of making a system capable of surviving unexpected or extraordinary failures [26]. The models and analyses discussed in this paper are closely related to the development of a disaster recovery plan (DRP). A DRP is an information system-focused plan designed to restore operation of the target system, application, or computer facility infrastructure at an alternate site after an emergency [27]. Note that a DRP shall be activated only after major system disruptions with long-term effects, because it usually involves operations that might be too costly for occasions of brief service interruptions.

A typical disaster recovery service works by replicating application state (e.g., database contents) between two data centers; if the primary data center undergoes a disaster, then the backup site can take over, activating a new copy of the application using the most recently replicated data [3]. In order to be effective, a disaster recovery solution must address some key requirements, related to variables such as financial costs of system downtime and affordable data loss. Among those requirements are: recovery time objective (RTO), recovery point objective (RPO), performance, consistency, and geographic separation [3] [26]. RTO refers the amount of time between an outage and the restoration of the system capabilities following an unplanned event or disaster. RPO is the maximum acceptable level of data loss following an unplanned event or disaster. Since disasters may be citywide (e.g., hurricanes) or even reach multiple countries simultaneously (e.g., the 2004 South Asian tsunami), the geographic separation is of utmost importance. Therefore, ideal disaster recovery mechanisms should employ continuous synchronous replication of data between geographically separated sites, through dedicated high-bandwidth connections [26].

Systems administrators must deal with the trade-off between cost, speed, and effectiveness of recovery, since solutions which provide zero data loss, instantaneous recovery time, full performance and consistency are not always feasible. One approach discussed in recent years (to enable business continuity at low cost) is the employment of cloud computing platforms to provide disaster recovery solutions [28] [3], benefiting from the pay-as-you-go model. Due to data privacy concerns, some companies and organizations might decide to build and use their own private clouds or virtualized data centers instead of third-party services such as public clouds, despite the possible higher costs.

4 System Architecture of Reliable Distributed Data Centers

This section presents an overview of the cloud computing system considered in this work, which contemplates a set of components, distributed over distinct data centers (Figure 2). The Infrastructure-as-a-Service (IaaS) model is adopted, considering delivery of computing resources on-demand as virtual machines (VM).

The system is composed of d data centers, each with two sets of machines, namely, hot and warm pools. The hot pool is composed of n physical machines (PM), which are active and running virtual machines (VM). The warm pool consists of m PMs that are active, but without running VMs. Thus, the number of PMs in a data center is $t = m + n$.

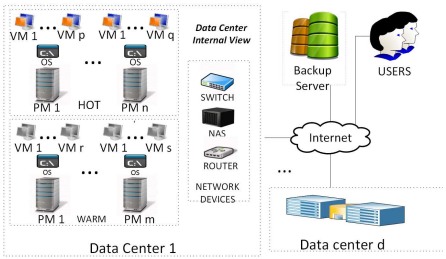

Fig. 2: Distributed Cloud System Example

Depending on the capacity of each PM, it is possible to run multiple VMs in the same host. In this study, we assume all physical machines are identical, in the sense that they adopt the same services, hardware, and software components. PMs may share a common network attached storage (NAS) or a storage area network (SAN) to provide distributed storage and to allow the migration of a virtual machine from one server to another in the same data center [29]. In case of failure, a VM must be instantiated in another physical machine of the same data center. If there is no available PM in the current data center, the VM image is moved to another data center.

Furthermore, a Backup Server (BS) is assumed to provide backup of VM data. This component receives a copy of each VM image during data center operation. Hence, whenever a disaster makes one data center unavailable, BS sends VM copies to an operational data center. In this work, the number of running VMs (w) is compared with a threshold (k) to evaluate the availability of cloud computing system. Hence, if $w \geq k$ the system is assumed operational.

4.1 High Level IaaS Model

This section presents a high level model to represent the proposed architecture [30]. A geographically distributed IaaS system corresponds to the tuple $G = (F_{lt}, T_{di}, T_{re}, MTT, T_{fhw}, T_{rhw})$ in which:

- F_{lt} is a finite set of facilities, including data centers and backup servers, such that $F_{lt} = D \cup BS$. D is a finite set of data centers and BS represents the set of backup servers;
- $T_{di} : F_{lt} \rightarrow f_{di}$ denotes the disaster occurrence function. For each facility $d_c \in F_{lt}$, a probability distribution function (PDF) f_{di} is associated. The function f_{di} provides the probability of a disaster for each instant t;

- $T_{re} : F_{lt} \rightarrow f_{re}$ represents the disaster recovery function. Similarly to the previous function, it associates a PDF (f_{re}) with each facility $d_c \in F_{lt}$. For each time t a probability of disaster recovery is provided;
- $MTT : F_{lt} \times F_{lt} \rightarrow f_{MTT}$ denotes the VM transmission function. The function relates a pair of facilities $(d_{c1}, d_{c2}) \in F_{lt} \times F_{lt}$ to a PDF f_{MTT}. The resulted function f_{MTT} provides the probability of finishing the data transmission between d_{c1} and d_{c2} at time t;
- $T_{fhw} : F_{lt} \rightarrow f_{fhw}$ represents the function for the failure due to hardware or software error. Each facility $d_c \in F_{lt}$ presents a probability distribution function f_{fhw} for representing the probability of failure occurrence at time t;
- $T_{rhw} : F_{lt} \rightarrow f_{rhw}$ means the function for repair hardware or software components (from non-disaster failures). It is analogous to T_{fhw} but f_{rhw} is represents the repair probability at time t.

A data center $dc \in D$ corresponds to the ordered pair (P_d, C_d), where P_d represents a represents a physical machine finite set. C_d represents the finite set of basic components of network infrastructure.

A physical machine $p \in P_d$ corresponds to the tuple (V_p, S_p, os, hw, m) where:

- V_p represents a virtual machine finite set assigned to the physical machine at cloud system start up;
- $S_p : V_p \rightarrow f_p$ provides the virtual machine set up time probability distribution function;
- $os \in O_p$ corresponds to the physical machines's software component;
- $hw \in H_p$ represents the hardware of the physical machine;
- $m \in \mathbb{N}$ denotes the maximum number of VMs that the physical machine can execute;

O_p and H_p are finite sets of software and hardware components related to physical machines. C ($C = C_d \cup O_p \cup H_p \cup V_p$) corresponds to a finite set of all data center's basic components. $T_{fr} : C \rightarrow f_{fr}$ represents the failure probability distribution function associated with a component $c \in C$, and $T_{rp} : C \rightarrow f_{rp}$ represents the repair PDF associated with a component $c \in C$.

4.2 VM Transmission (MTT) estimation

To estimate the Mean Time to VM Transmission (MTT), we consider the approach presented in [31] that provides an equation to assess the network throughput based on the distance between the communication nodes. The VM transmission rate is obtained as follows:

$$Rate < (MSS/RTT) \times (1/\sqrt{p}) \tag{6}$$

where *Rate* (kbps) is the TCP transfer rate, *MSS* (bytes) is the maximum segment size per package, *RTT* (ms) is the round trip time, and p is the packet loss ratio. The following equation is adopted to estimate the RTT:

$$RTT = \frac{Dist}{\alpha \times 100} \tag{7}$$

in which *Dist* means the distance in kilometers. The equation associates a constant α with network directness, in which values close to one mean the path between the hosts follows a direct path. Values much smaller than one mean the path is very indirect.

5 Case study: A disaster recovery architecture

This section presents our analytical model for a cloud data center concerning disaster issues, and a sensitivity analysis of such model to assess the impact of each parameter on system availability. Mercury tool [32] was adopted for CTMC modeling and sensitivity analysis.

5.1 System description

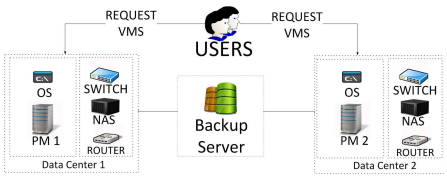

Fig. 3: Cloud System Architecture

Figure 3 depicts the infrastructure adopted ($F_{lt} = D \cup BS$) as our case study, which contemplates two data centers ($D = \{D1, D2\}$) and a backup server ($BS = \{B\}$). Each data center is composed by one physical machine (PM) ($D1 = (P_{D1} = \{Pmc_1\}, C_{D1})$,$D2 = (P_{D2} = \{Pmc_2\}, C_{D2})$ that contemplates hardware and basic software (C_{D1} and C_{D2}) for running at most 2 virtual machines ($m = 2$). Backup server B is responsible for periodically saving the states of all VMs. Thus, whenever a data center fails, a VM is transmitted to the other data center (assuming it is operational) using the snapshot available in the backup server [33].

The system has 2 VMs and both must be operating. In other words, if a single VM fails, the system stops providing the service. Data center 1 is the primary center, and data center 2 is considered the spare component. The following assumptions were considered for this particular study:

- The backup server is not affected by disasters;
- VM migration is only possible if the backup server is operational;
- During VM migration, other components do not fail;

- Whenever the backup server is on a failure state, only VMs can fail;
- VM migration only occurs when the data center executing both VMs stops. If a single VM fails, it is recovered on the same data center;
- VM migration time also includes the time for initialization of both VMs in the new data center;
- Data center 1 has greater priority over Data center 2 considering recovery;
- Disaster recovery is complete, in the sense that it also considers hardware and VM recovery;
- Two simultaneous disasters are not possible. In other words, if a disaster is affecting one data center, the other data center cannot fail due to a disaster. However, the latter can fail due to hardware or software issues;
- A data center recovering from a hardware failure has priority over the recovery of a data center impacted by disasters.

Such assumptions exclude some rare simultaneous events and were carefully evaluated to not cause significant changes on results, while avoiding the state space explosion.

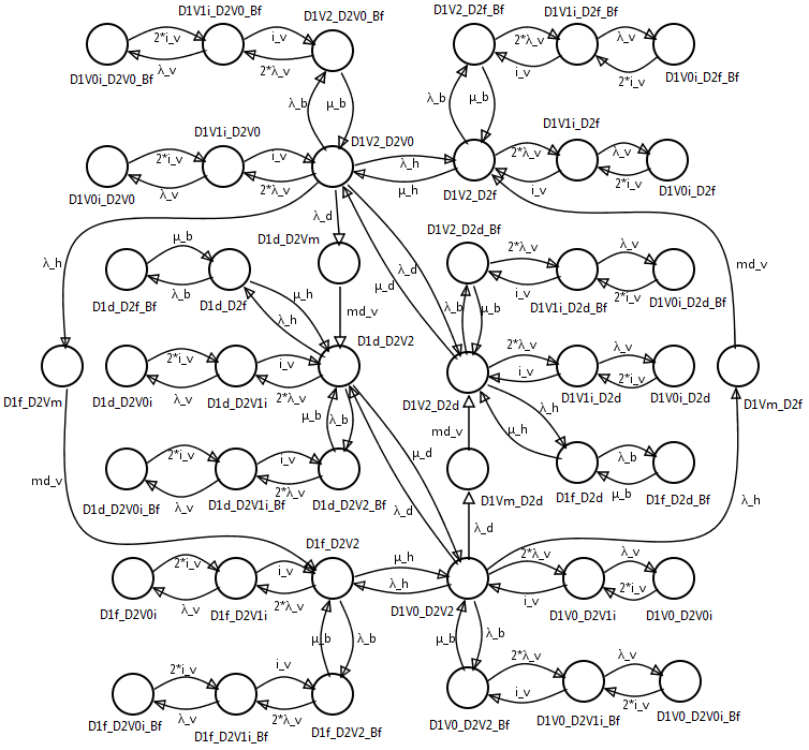

Fig. 4: CTMC Model

5.2 Analytical model

Figure 4 depicts the conceived CTMC model. As previously mentioned, the proposed model takes into account a set of assumptions for representing the data center behavior (see Section Section 5.1). The rates and notation are described in Table 1 and 2, respectively. The rates have been obtained from [17], [34], [35], [36] and the VM migration time was estimated using the approach described in Section 4.2.

As an example of state notation, state D1V1i_D2d_Bf represents the following situation: (i) data center 1 is operational with 1 running VM and 1 VM being instantiated; (ii) data center 2 is failed due to a disaster; and (iii) the backup server is not operational. The metric of interest is steady-state availability (A), in which the operational condition is two running VMs. Therefore, the system is available in the following states: UP={D1V2_D2V0, D1V2_D2V0_Bf, D1d_D2V2_Bf, D1V2_D2f, D1V2_D2d, D1d_D2V2, D1V2_D2f_Bf, D1f_D2V2, D1V2_D2d_Bf, D1V0_D2V2, D1f_D2V2_Bf , D1V0_D2V2_Bf}.

More specifically, $A = \sum_{s \in UP} \pi(s)$, in which $\pi(s)$ denotes the steady-state probability of state s. From the system of linear equations of a CTMC (see Section 3.1), a closed-form equation can be obtained for A, but this work omits it due to the excessive length of such expression. The evaluation process was conducted by using the sensitivity analysis feature of Mercury tool [32]. This approach calculates the partial derivatives of the measure of interest with respect to each input parameter (Section 3.2).

Table 1: Rates for CTMC model

Parameter	Description	Value (h)
$1/i\_v$	Mean time to instantiate a VM (S_p)	0.008
$1/\lambda\_v$	Mean time to VM failure	2880.000
$1/\lambda\_b$	Mean time to failure of backup server (T_{fhw})	5000.000
$1/\mu\_b$	Mean time to repair of backup server (T_{rhw})	0.500
$1/\lambda\_h$	Mean time to failure of a data center due to hardware or software (T_{fhw})	800.000
$1/\mu\_h$	Mean time to repair a data center due to hardware or software (T_{rhw})	0.500
$1/\lambda\_d$	Mean time to disaster occurrence for a data center (MTT)	87600.000
$1/\mu\_d$	Mean time to repair a data center due to disaster (T_{re})	4320.000
$1/md\_v$	Mean time to migrate and instantiate all VMs ($2 \times MTT + 2 \times S_p$)	4.083

5.3 Sensitivity Analysis Results

Table 1 presents the parameters ψ adopted in this work (see Figure 4). Equation (8) defines the scaled sensitivity of system availability.

Table 2: State Notation

Notation	Description
DxVn	Data center x is operational, and n VMs are running ($n \in \{0, 2\}$)
DxVmi	m VMs are running on data center x ($0 \leq m \leq 1$), and VMs are being instantiated
Dxd	Data center x is not operational due to a disaster
Dxf	Data center x is not operational due to hardware or software failure
Bf	Backup server is failed.

Table 3: Sensitivity ranking based on partial derivatives for steady-state availability

Parameter	$\lvert SS(A)\rvert$
λ_h	5.481×10^{-3}
md_v	4.557×10^{-3}
μ_h	9.653×10^{-4}
λ_d	6.364×10^{-4}
μ_d	5.951×10^{-4}
i_v	1.151×10^{-4}
λ_v	5.754×10^{-6}
λ_b	4.557×10^{-7}
μ_b	4.557×10^{-7}

$$SS_{\psi}(A) = \frac{\psi}{A}\frac{\partial A}{\partial \psi} = \frac{\psi}{(\sum_{s \in UP} \pi(s))} \times \frac{\partial(\sum_{s \in UP} \pi(s))}{\partial \psi}. \tag{8}$$

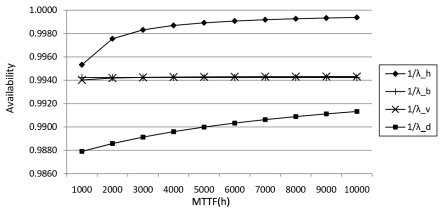

Fig. 5: Availability Results - Varying Component's MTTF

Table 3 depicts the sensitivity ranking, which indicates that hardware failure (λ_h) has the largest impact on system availability. The time for migration of VMs (md_v) is the second most important parameter.

It is important to stress that the steady-state availability in the baseline scenario – i.e., assigning all parameters to values of Table 1 – is 0.99423684, equivalent to an annual downtime of 50.48 hours. The scaled sensitivity indices for λ_h and md_v are both within the order of 10^{-3}. Therefore, these parameters may change the system availability in the third decimal place, raising it to about 0.999, or bringing it down to 0.99, depending whether a decrease or increase is applied on them. Table 3 also shows that both disaster-related

parameters ($lambda\_d$ and mu_d) affect significantly the availability, but the failure rate (λ_d) has a greater impact than the respective recovery ((μ_d).

The results can be verified using the plots depicted in Figure 5, 6 and 7. The plots are separated because failure and recovery parameters have different magnitudes (i.e., recovery actions takes less time than failures). The recovery is also split into two plots, as only parameters i_v and μ_b are in the order of minutes. Nevertheless, in all plots, we kept the parameters fixed using the values in Table 1; only one parameter is varied at a time; then system availability is calculated. Whilst a larger time for a failure event positively contributes to an improved availability, a larger time for a recovery action decreases availability.

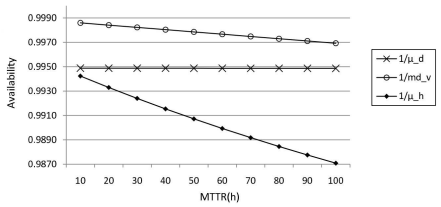

Fig. 6: Availability Results - Varying μ_d, md_v and μ_h

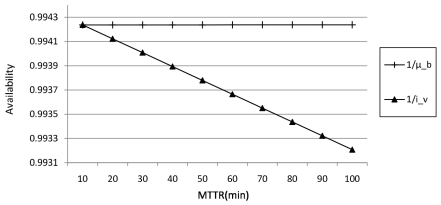

Fig. 7: Availability Results - Varying μ_b and i_v

Notice that lines for the hardware and disaster failures ($1/\lambda\_h$ and $1/\lambda\_d$) in Figure 5 have the largest slopes, confirming their importance as also shown in Table 3. On the other hand, failures of VMs and the backup server yield a negligible impact on the availability when compared to the other failure-related parameters. Such an issue is also exposed by the low sensitivity indices of λ_v and λ_b, indicating effect only on the seventh decimal place of the steady-state availability measure.

Figure 6 confirms that varying time to hardware repair ($1/mu\_h$), time for migration of VMs ($1/md\_v$), and time for disaster recovery ($1/\mu\_d$) cause changes to the availability that are more significant than those produced by the other recovery parameters: VMs instantiation time, and time to repair the backup server.

We can also see the impact of each parameter on the availability through the sensitivity ranking for the metric number of nines. This metric is calcu-

Table 4: Sensitivity ranking based on partial derivatives for the number of nines

Parameter	$\lvert\widehat{SS}(Nines)\rvert$
λ_h	4.186×10^{-1}
md_v	3.481×10^{-1}
μ_h	7.373×10^{-2}
λ_d	4.861×10^{-2}
μ_d	4.546×10^{-2}
λ_v	4.395×10^{-4}
i_v	4.395×10^{-4}
λ_b	3.481×10^{-5}
μ_b	3.481×10^{-5}

lated using $-log_{10}(1-A)$ (in which A refers to availability). For instance, the baseline availability 0.99434554 can be presented as $-log_{10}(1-0.99423684) = 2.23933932$ nines.

Equation (9) is used to compute the semi-relative sensitivity index [6] of this metric with respect to each parameter ψ of the model. We do not use the same scaling approach as done for the steady-state availability because the metric number of nines is logarithmic, and as seen in Section 3.2, a semi-relative sensitivity function better suits this kind of measure.

$$\widehat{S}_{\psi}(Nines) = \psi \times \frac{\partial(-log_{10}(1-A))}{\partial\psi} = \psi \times \frac{S_{\psi}(A)}{log(10) - log(10) \times A} \tag{9}$$

The ranking of Table 4 presents the same order of parameters as shown in the ranking for the sensitivity of steady-state availability. This is expected because the number of nines is only a different view for the availability. The sensitivity indices for λ_h and md_v are 4.186×10^{-1} and 3.481×10^{-1}, respectively, indicating that the system might go from the current value of 2.24761 nines to about three nines – what would be a significant improvement – by gradual adjusts on these parameters.

These results can also be verified using the plots depicted in Figures 8, 9 and 10, which confirm what was already seen in Table 3.

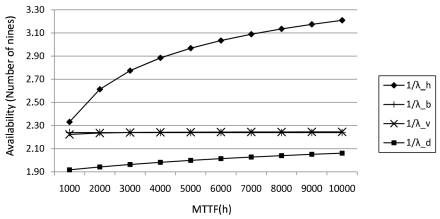

Fig. 8: Availability as Number of Nines - Varying Component's MTTF

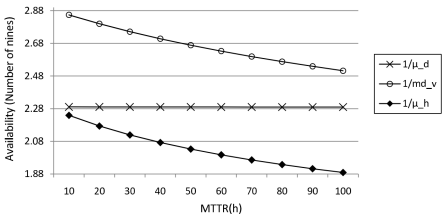

Fig. 9: Availability as Numberf of Nines - Varying μ_d, md_v and μ_h

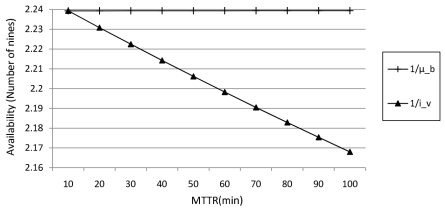

Fig. 10: Availability as Number of Nines - Varying μ_b and i_v

From the results, one important conclusion is disaster issues should not be neglected when designing data centers as well as the location where such infrastructures are placed. For instance, the VM migration (md_v) is the most impacting recovery parameter, and it is directly related to the distance between data centers (which is analyzed further in Section 5.4). Additionally, the time between disasters (λ_d) is the second most important among the failure parameters. The careful choice of locations rarely affected by natural disasters is one of the few actions that can effectively change λ_d. The usage of highly reliable hardware and preventive maintenance policies may be instead affordable decision,s which affect hardware failure rate (λ_h) and, therefore, they will have a significant impact on overall system availability.

5.4 Data Center and Backup Server Locations

Sensitivity analysis indicates md_v as an important parameter, as the sensitive index for md_v points out an impact on the third decimal place of system availability (A). Since system downtime (DT) is calculated as $DT = [(1 - A) \times period]$, a smaller VM migration time can reduce downtime as much as $86400 \times 0.001 = 8.64$ hours in a year.

Thus, the definition of data centers and backup server locations is a prominent design decision. In this section, we consider an experiment, in which Data center 1 ($D1$) and Data center 2 ($D2$) are equidistant to a backup server (B) located in one of major Brazilian cities: Sao Paulo or Rio de Janeiro (Rio). Such distances are adopted to estimate md_v, considering the migration of 2 VMs. The approach described in Section 4.2 is considered, assuming the following parameters: (i) $\alpha = 0.35$; (ii) 4 GB for a VM image; and (iii) $p = 0.01$.

Table 5: Backup server and data center location

Cities	Distance (km)	$1/md\_v$ (h)
Rio-Brasilia	932	4.435
Rio-Recife	1,877	8.847
Rio-New York	7,765	36.341
Rio-Calcutta	15,088	70.535
Rio-Tokyo	18,580	86.841
Sao Paulo-Rio	353	1.731
Sao Paulo-Brasilia	873	4.159
Sao Paulo-Recife	2,128	10.019
Sao Paulo-New York	7,694	36.009
Sao Paulo-Calcutta	15,440	72.179
Sao Paulo-Tokyo	18,546	86.683

Additionally, Table 5 depicts the pair of cities, which defines the distances between a data center and a backup server, as well as the estimated values for md_v (in hours).

We have adopted the values presented in Table 1, in the sense that all parameter values are kept constant, except md_v. Next, system availability has been estimated, and Figure 11 presents the results. There is no major difference whether backup server is located in Rio or Sao Paulo, for instance, since they are near cities. However, when the distance between the backup server and data center increases, availability is affected. Assuming the data center is located in Rio and backup server is in Sao Paulo (or vice-versa), the system availability has the highest value. If the data center is located in other Brazilian cities, such as Brasilia and Recife, the impact on availability is already noticeable. The impact is even higher as the distance increase to the other parts of the same continent (e.g., New York). When extremely distant places (e.g., Calcuta and Tokyo) are taken into account, availability is significantly reduced – reaching about 0.9 (number of nines equal to 1) – due to the large time to migrate VMs. Nonetheless, if the assumption of countrywide disasters is not very unlikely, the first result would not be so high, and intermediate distances would be preferred. However, such an evaluation is out of scope of this work.

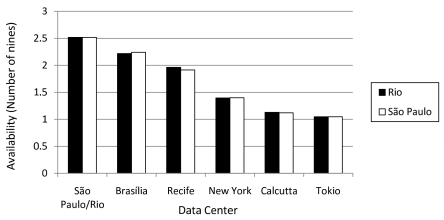

Fig. 11: Availability Results -Different Locations

6 Conclusion

This paper presented an approach based on sensitivity analysis and continuous time Markov chains to identify the parameters that most impact the availabil-

ity of cloud data centers. The analysis contemplated hardware and software failures, disaster occurrence, as well as VM migration. The sensitivity ranking indicates hardware failures have the most impact on the availability metric, but the time for VM migration is the second most important parameter. Besides, disaster occurrence and recovery are also on the top of the ranking.

The proposed approach demonstrates that researchers and practitioners have a prominent technique to assess different parameters that affect the operational state of a data center. The results should guide decision making to meet SLA requirements by tuning system parameters as well carefully choosing the place where the infrastructures should be located.

References

1. J. Sterbenz, et al, Evaluation of network resilience, survivability, and disruption tolerance: analysis, topology generation, simulation, and experimentation, Telecommunication Systems 52 (2) (2013) 705–736.
2. R. Miller, Car crash triggers amazon power outage, Data Center Knowledge, available on http://www.datacenterknowledge.com/archives/2010/05/13/car-crash-triggers-amazon-power-outage/ (May 2010).
3. T. Wood, E. Cecchet, K. K. Ramakrishnan, P. Shenoy, J. van der Merwe, A. Venkataramani, Disaster recovery as a cloud service: Economic benefits & deployment challenges, in: Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing, HotCloud'10, USENIX Association, Berkeley, CA, USA, 2010, pp. 8–8. URL http://dl.acm.org/citation.cfm?id=1863103.1863111
4. Hyper-V Live Migration over Distance. URL http://goo.gl/GzlkNk
5. P. Maciel, K. S. Trivedi, R. Matias, D. S. Kim, Performance and dependability in service computing: Concepts, techniques and research directions (2011). doi:doi:10.4018/978-1-60960-794-4.ch003.
6. P. M. Frank, Introduction to System Sensitivity Theory, Academic Press Inc, 1978.
7. D. M. Hamby, A review of techniques for parameter sensitivity analysis of environmental models, Environmental Monitoring and Assessment (1994) 135–154.
8. R. Matos, P. R. M. Maciel, F. Machida, D. S. Kim, K. S. Trivedi, Sensitivity analysis of server virtualized system availability, Reliability, IEEE Transactions on 61 (4) (2012) 994–1006.
9. F. Longo, R. Ghosh, V. Naik, K. Trivedi, A scalable availability model for infrastructure-as-a-service cloud, in: Dependable Systems Networks (DSN), 2011 IEEE/IFIP 41st International Conference on, 2011, pp. 335 –346. doi:10.1109/DSN.2011.5958247.
10. R. Ghosh, K. S. Trivedi, V. K. Naik, D. S. Kim, End-to-end performability analysis for infrastructure-as-a-service cloud: An interacting stochastic models approach, in: Proceedings of the 2010 IEEE 16th Pacific Rim International Symposium on Dependable Computing, PRDC '10, IEEE Computer Society, Washington, DC, USA, 2010, pp. 125–132. doi:10.1109/PRDC.2010.30. URL http://dx.doi.org/10.1109/PRDC.2010.30
11. J. Araujo, R. Matos, P. Maciel, R. Matias, I. Beicker, Experimental evaluation of software aging effects on the Eucalyptus cloud computing infrastructure, in: Proceedings of the Middleware 2011 Industry Track Workshop, Middleware '11, ACM, New York, NY, USA, 2011, pp. 4:1–4:7. doi:10.1145/2090181.2090185. URL http://doi.acm.org/10.1145/2090181.2090185
12. Open source private and hybrid clouds from Eucalyptus, http://www.eucalyptus.com.
13. R. Bradford, E. Kotsovinos, A. Feldmann, H. Schiöberg, Live wide-area migration of virtual machines including local persistent state, in: Proceedings of the 3rd international conference on Virtual execution environments, VEE '07, ACM, New York, NY, USA, 2007, pp. 169–179. doi:10.1145/1254810.1254834. URL http://doi.acm.org/10.1145/1254810.1254834
14. W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya, Cost of virtual machine live migration in clouds: A performance evaluation, in: Proceedings of the 1st International Conference on Cloud Computing, CloudCom '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 254–265. doi:10.1007/978-3-642-10665-1_23.

15. J. Dantas, R. Matos, J. Araujo, 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, 2012, pp. 1664 –1669. doi:10.1109/ICSMC.2012.6377976.
16. R. T. Kaushik, M. Bhandarkar, K. Nahrstedt, Evaluation and analysis of greenhdfs: A self-adaptive, energy-conserving variant of the hadoop distributed file system, in: Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, IEEE, 2010, pp. 274–287.
17. D. S. Kim, F. Machida, K. S. Trivedi, Availability modeling and analysis of a virtualized system, in: Proceedings of the 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing, PRDC '09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 365–371. doi:10.1109/PRDC.2009.64.
18. R. Matos, J. Araujo, D. Oliveira, P. Maciel, K. Trivedi, Sensitivity analysis of a hierarchical model of mobile cloud computing, Simulation Modelling Practice and Theorydoi:http://dx.doi.org/10.1016/j.simpat.2014.04.003.
19. J. T. Blake, A. L. Reibman, K. S. Trivedi, Sensitivity analysis of reliability and performability measures for multiprocessor systems, in: Proceedings of the 1988 ACM SIGMETRICS conference on Measurement and modeling of computer systems, ACM, New York, NY, USA, 1988, pp. 177–186. doi:http://doi.acm.org/10.1145/55595.55616.
20. Z. W. Birnbaum, On the importance of different components in a multicomponent system, Multivariate Analysis - II (1969) 581–592Academic Press.
21. J. Barabady, U. Kumar, Availability allocation through importance measures, International journal of quality & reliability management 24 (6) (2007) 643–657.
22. Y. Ou, J. B. Dugan, Approximate sensitivity analysis for acyclic markov reliability models (June 2003).
23. H. Abdallah, M. Hamza, On the sensitivity analysis of the expected accumulated reward, Performance Evaluation 47 (2) (2002) 163–179. doi:http://dx.doi.org/10.1016/S0166-5316(01)00063-3.
24. J. K. Muppala, K. S. Trivedi, GSPN models: sensitivity analysis and applications, in: ACM-SE 28: Proceedings of the 28th annual Southeast regional conference, ACM, New York, NY, USA, 1990, pp. 25–33. doi:http://doi.acm.org/10.1145/98949.98962.
25. B. Yin, G. Dai, Y. Li, H. Xi, Sensitivity analysis and estimates of the performance for m/g/1 queueing systems, Perform. Eval. 64 (4) (2007) 347–356. doi:http://dx.doi.org/10.1016/j.peva.2006.06.004.
26. G. Reese, Cloud application architectures, O'Reilly Media, Inc., 2009.
27. M. Swanson, P. Bowen, A. W. Phillips, D. Gallup, D. Lynes, Contingency planning guide for federal information systems, Tech. rep., available on http://www.nist.gov/customcf/get_pdf.cfm?pub_id=905266 (2010).
28. S. Rajagopalan, B. Cully, R. O'Connor, A. Warfield, Secondsite: disaster tolerance as a service, in: Proceedings of the 8th ACM SIGPLAN/SIGOPS conference on Virtual Execution Environments, VEE '12, ACM, New York, NY, USA, 2012, pp. 97–108.
29. C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, Live migration of virtual machines, in: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2, NSDI'05, USENIX Association, Berkeley, CA, USA, 2005, pp. 273–286.
URL http://dl.acm.org/citation.cfm?id=1251203.1251223
30. B. Silva, P. R. M. Maciel, A. Zimmermann, Geoclouds modcs: A performability evaluation tool for disaster tolerant iaas clouds, in: 8th annual ieee international systems conference, 2013.
31. W. M. Les Cottrell, C. Logg, Tutorial on internet monitoring and pinger at SLAC, Tech. rep. (1996).
URL http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html
32. B. Silva, G. Callou, E. Tavares, P. Maciel, J. Figueiredo, E. Sousa, C. Araujo, F. Magnani, F. Neves, Astro: An integrated environment for dependability and sustainability evaluation, Sustainable Computing: Informatics and Systemsdoi:10.1016/j.suscom.2012.10.004.
33. B. Silva, P. R. M. Maciel, A. Zimmermann, Dependability models for designing disaster tolerant cloud computing systems, in: The Third International Workshop on Dependability of Clouds, Data Centers and Virtual Machine Technology (DCDV), 2013.
34. Cisco systems: Switch dependability parameters (Oct. 2012).
URL http://tinyurl.com/cr9nssu
35. Cisco systems: Router dependability parameters (Oct. 2012).
URL http://tinyurl.com/d7kcnqo
36. Service level agreement - megapath business access and value added services (Oct. 2012).
URL http://tinyurl.com/cwdeebt