

Model-based Sensitivity Analysis of IaaS Cloud Availability

Bo Liu^{a,b}, Xiaolin Chang^a, Zhen Han^a, Kishor Trivedi^c, Ricardo J. Rodríguez^d

^aBeijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, P. R. China

^bBeijing Institute of Information Application Technology, P. R. China

^cDepartment of Electrical and Computer Engineering, Duke University, USA

^dCentro Universitario de la Defensa, Academia General Militar, Zaragoza, Spain

Abstract— The increasing shift of various critical services towards Infrastructure-as-a-Service (IaaS) cloud data centers (CDCs) creates a need for analyzing CDCs’ availability, which is affected by various factors including repair policy and system parameters. This paper aims to apply analytical modeling and sensitivity analysis techniques to investigate the impact of these factors on the availability of a large-scale IaaS CDC, which (1) consists of active and two kinds of standby physical machines (PMs), (2) allows PM moving among active and two kinds of standby PM pools, and (3) allows active and two kinds of standby PMs to have different mean repair times. Two repair policies are considered: (P1) all pools share a repair station and (P2) each pool uses its own repair station. We develop monolithic availability models for each repair policy by using Stochastic Reward Nets and also develop the corresponding scalable two-level models in order to overcome the monolithic model’s limitations, caused by the large-scale feature of a CDC and the complicated interactions among CDC components. We also explore how to apply differential sensitivity analysis technique to conduct parametric sensitivity analysis in the case of interacting sub-models. Numerical results of monolithic models and simulation results are used to verify the approximate accuracy of interacting sub-models, which are further applied to examine the sensitivity of the large-scale CDC availability with respect to repair policy and system parameters.

Index Terms—Availability; Sensitivity analysis; Markov chain; IaaS; Cloud computing; Stochastic Reward Nets

1 INTRODUCTION

The past few years have witnessed fundamental changes caused by cloud computing to business computing models. Infrastructure as a Service (IaaS) is one of the basic cloud services. This cloud service is provisioned to customers in the form of virtual machines (VMs), which are deployed on physical machines (PMs). Each VM has specific characteristics in terms of number of CPU cores, amount of memory and amount of storage. It was reported that global spending on IaaS cloud services is expected to reach 56 Billion USD by 2020 [1]. The ever-increasing demands for IaaS cloud services have created the need for cloud service providers (CSPs) to analyze cloud infrastructure availability in order to maintain high cloud service availability [2] while reducing various costs. Service availability is commonly specified via Service Level Agreements (SLAs) [3]-[5]. Any availability violation may cause the loss of revenue. In addition, some common IaaS cloud management tools, such as OpenStack [6], have allowed configuring standby PMs for high availability. However, there is no suggestion about how to configure.

System availability is affected by various factors, such as system parameters and repair policy. The latter one determines how quickly PMs get repaired upon their failure. Repair policy analysis is significant to the CDC design with respect to CDC availability. State-space models are popular and found effective for system availability analysis [7]. They also allow the derivation of sensitivity functions of the measures of interest with respect to various system parameters, which are assigned in a continuous domain. These functions could be applied to assess the impact of each of these parameters on

system quality of service (QoS) and then to identify the QoS bottlenecks for systems of interest.

This paper aims to explore analytical modeling and sensitivity analysis techniques to improve the availability of a large-scale IaaS cloud data center (CDC). Following Ghosh *et al.* [12], we assume that there are three PM pools, namely hot (running PMs), warm (turned on, but not ready PMs) and cold (turned off PMs). Thus, there are two kinds of standby PMs, warm-standby and cold-standby. A small provisioning delay is needed for deploying default VM images on hot PMs. Additional provisioning time (to make the PM ready) is required for the VM deployment on a warm PM. Further delay is added when PMs in the cold pool are used, since they need to be turned on before being used. PMs can move among pools due to failure/repair events. PM repair times of different PM pools may be different. The main reason of considering the CDC with three PM pools in this paper is that the modeling approach of this scenario could be applied directly to scenarios with arbitrary number of pools. Note that although IaaS CSPs in production CDCs have offered standby PMs for disaster recovery [8], there is no published information about the number of PM pools.

Large scale is a feature of CDCs, leading to the well-known largeness problem [8] associated with a monolithic or one-level Markov chain for the availability analysis of an IaaS CDC. Moreover, complex interactions among CDC components and different failure/repair behaviors further exacerbate the largeness problem. Our numerical results show that the

monolithic model cannot be solved when each pool size is larger than six. Even Stochastic Petri Nets (SPN) [10], which could automate the generation of Markov models, still faces the issues of generating, storing and solving large models. A scalable model could be obtained by resorting to a two-level hierarchical model.

In this paper, we consider two kinds of repair policies: 1) each pool has its own *Independent Repair Station (IRS)*, and 2) all pools Share a *Repair Station* (denoted as *SRS*). The repair rates of different PM pools are different. We develop monolithic availability models for each repair policy by using Stochastic Reward Nets (SRNs) [10] and also develop the corresponding scalable two-level models in order to overcome the monolithic model's limitations, caused by the large-scale feature of a CDC and the complicated interactions among CDC components. Experiments under various settings are carried out to verify the approximate accuracy of hierarchical models by comparison with numerical results of monolithic models and simulation results. Further, we apply the proposed hierarchical models to investigate the sensitivity of the large-scale CDC availability with respect to repair policy and system parameters.

The major contributions are summarized as follows:

- 1). For each repair policy, we develop a monolithic model and scalable interacting sub-models. Note that the work in [12] is close to ours but there is difference not only in the modeled system (namely, the repair policy) but also in the model design (namely, the modeling of PM failure). Section 2 details the difference. As mentioned in [12], a small change to the modeled system will make the modeling more complex. We detail the design and verify the approximate accuracy.
- 2). Differential analysis method [26] is explored to conduct parametric sensitivity analysis of system availability based on interacting SRN sub-models. To our best knowledge, it is the first time to investigate the ability of differential analysis method in parametric sensitivity analysis [11] of interacting-type hierarchical models. In Section 6, we show that without calculating the exact derivatives of the overall system availability with respect to each system input parameter, we could identify parameters which have greatest impact on system QoS. It is done by applying differential sensitivity analysis method to each sub-model.

The rest of the paper is organized as follows. Section 2 presents related work and background knowledge of sensitivity analysis. Section 3 introduces system architecture considered in this paper. Section 4 and 5 describe models for each repair policy. Numerical results and discussions are presented in Section 6. Finally, Section 7 concludes this paper and discusses future work.

2 RELATED WORK

This section first presents related work on modeling cloud data centers. Then methods for analyzing the sensitivity of system metrics with respect to parameters are presented.

2.1 Model-based for CDC Availability

Monolithic analytical models for investigating the availability of virtualized systems with detailed failures have been proposed [14], [15]. These models focused on the evaluation of VM and assumed all PMs are homogeneous. Our work in this paper analyzed PM availability and distinguished the difference among PMs in terms of repair rate and state. In [16], SPNs and reliability block diagrams were used for quantification of sustainability impact, cost and dependability of data center cooling infrastructures, but focusing only on the cooling system. A monolithic Markov model for a large-scale system may face largeness and stiffness problems. These modeling papers ignored scalability issues in availability evaluation and hence were not suitable for large-scale CDCs.

The authors in [18] explored the PM availability analysis in the same situation as our paper. They proposed a monolithic model without thinking the scalability issue, which is our main focus in this paper. Decomposition is an effective approach to overcome these problems [25]. The authors in [20] ignored the moving delay and constructed scalable interacting SRN sub-models in order to reduce the analysis complexity of a monolithic model. Dependencies among these SRN sub-models were resolved using fixed-point iteration technique [24]. They, in [12], further extended their modeling to the scenario with the PM moving delay, which made the availability model more complex to be analyzed. In [12], they assumed that (Ri) each pool has its own repair station, (Rii) PM repair time for each pool must be same, and (Riii) a failed hot PM can be repaired by a warm/cold repair facility. There are two major differences between [12] and our work as follows:

Difference 1. The first is the repair policy. The assumptions about repairing PMs in [12] simplified the construction of the interacting SRN sub-models. By simplified, we mean that our sub-models and sub-model interaction are more complicated than in [12]. In a realistic system, the repair time of a failed warm PM may be different from that of a failed hot PM. In our paper, we first relax the second assumption (namely, Rii in the previous paragraph) by equipping each pool with an independent repair station and then each failed PM of different pools has different repair times, denoted as *IRS* repair policy in the rest of paper. We then consider the system which has only one repair pool but the failed PMs of different pools has different repair times, namely, *SRS* repair policy.

Difference 2. The interacting sub-models proposed in [12] could not capture the PM failure process in both hot and warm sub-models in an approximately accurate way in some situations. We analyze the reason in Section 4 and make experimental evaluation in Section 6.

Failure characteristics of CDC servers were studied in [19], where hardware failure/repair rates of PMs in cloud systems were empirically obtained. The results are applicable for setting the failure/repair rates of the models we propose in this paper. A variety of pure performance models for cloud ser-

VICES were proposed in the last few years. See [13] and references therein. These models are complementary to our models to capture IaaS cloud service behaviors. In the following we focus on the literature on cloud availability analysis. In [17], cloud service availability was evaluated from a user-centric point of view, unlike our work that considers a cloud service provider's point of view.

2.2 Sensitivity analysis

Sensitivity analysis allows the exposure of system QoS bottleneck as well as providing guidelines for the system optimization. It could be divided into nonparametric and parametric sensitivity analysis [27]. The first kind studies output variations caused by modifications in the structure of a model (e.g., addition or removal of a given component in a model). The second studies the output variations due to a change in system parameter values. There are several approaches for performing sensitivity analysis [26]. The following presents three approaches to be used in this paper:

(i) Vary one parameter at a time within the considered range while keeping the others constant and observe system measures of interest with respect to the varying parameter. In order to determine the parameters that cause the greatest impact on the system QoS, simulations or numerical analysis for all parameters in their defined ranges must be done.

(ii) Differential sensitivity analysis (also called directed method). It computes the sensitivity of a given measure Y , which depends on a specific parameter θ , as $S_{\theta}(Y) = \frac{\partial Y}{\partial \theta}$,

or $SS_{\theta}(Y) = \frac{\partial Y}{\partial \theta} \cdot \frac{\theta}{Y}$ for a scaled sensitivity. The sign of SS_{θ}

denotes whether an increase of θ causes a corresponding increase or instead a decrease of the measure Y . Its absolute value indicates the magnitude of the variations of Y for small variations of θ . This method is only suitable for continuous parameters.

(iii) Sensitivity index. This technique is designed for integer-valued parameters which are not properly evaluated by the differential sensitivity analysis approach. The sensitivity

formula is $S_{\theta}(Y) = 1 - \frac{\min\{Y(\theta)\}}{\max\{Y(\theta)\}}$, where $\theta \in [\theta_1, \theta_n]$,

$\min\{Y(\theta)\} = \min\{Y(\theta_1), Y(\theta_2) \dots Y(\theta_n)\}$ and

$\max\{Y(\theta)\} = \max\{Y(\theta_1), Y(\theta_2) \dots Y(\theta_n)\}$.

Sensitivity analysis has been conducted in cloud systems. In [27], the last two methods mentioned above were used for sensitivity analysis of the availability of a virtualized system, which was modeled as a continuous-time Markov chain (CTMC). The authors in [28] studied a hierarchical model, which consisted of several independent sub-models, each of which was modeled as a CTMC. Thus, the overall system measure is the multiplication of the measure of each sub-model. Then the sensitivity of the overall system availability with respect to a system continuous parameter could be obtained accordingly by calculating the overall availability sensitivity with respect to each component and the component availability sensitivity with respect to this parameter. But in

our hierarchical models, there exist complex interactions among sub-models. It is hard, if not impossible, to compute the derivative of the whole system measure with respect to any system parameter. In Section 6, we show that although $S_{\theta}(Y)$ of each parameter could not be calculated, we could identify parameters which impact system most significant by applying differential sensitivity analysis method to each sub-model and then ignoring some parameters with less impact on system QoS.

3 SYSTEM DESCRIPTION

In this paper, we assume that there are three PM pools (namely hot, warm and cold) in a CDC. It is known that there exist several types of failures in a cloud system such as software failures, hardware failures and network failures [12]. This paper considers the overall effect of these possible failures with an aggregated mean time to failure (MTTF) [22], [23]. Failure detection is assumed to be an instantaneous event. PMs in the same pool have independent and identical distributed TTFs. TTFs of hot, warm and cold PM pools are exponentially distributed. As in [12], mean TTF rates are assumed as $\lambda_h > \lambda_w > \lambda_c$ in this paper. Three possible reasons for such assumption are as follows. It is known that software execution could speed up hardware component failure, such as fan and hard disk. In addition, software aging is unavoidable and then a computer is forced to shut down if there is no active action to take. The third is that a computer could generate corrupted files, which could damage the computer hardware on the long term.

Upon failure of a hot PM, this failed PM is moved from the hot PM pool to the pre-determined repair station for repair. Meanwhile, a PM available in the warm pool is moved to the hot pool. When the warm pool is empty but there exists a PM available in the cold pool, moving this PM to the hot pool is performed. Similarly, when a warm PM fails, it is moved from the warm pool for repair and a PM is moved from the cold pool to perform the role of this warm PM. For each pool, if there is a PM moving from other pools in order to play the role of a failed PM, this moving PM will return to its original pool after the failed PM completes its repair. Time to move a PM from one pool to another follows an exponential distribution. PM repair activities are work conserving and repaired PMs are as good as new. We consider two kinds of repair policies as follows:

1). *Independent repair station (IRS)*. Each pool has its own repair station. There is at least one repair facility in each station. Each facility repairs a failed PM independently. A PM in a pool could be repaired only by a repair facility of this pool's repair station. If the number of PMs in a pool to be repaired is larger than the number of the corresponding repair facilities/servers, failed PMs are placed in the corresponding waiting queue. Hot, warm and cold PM mean repair times are exponentially distributed.

2). *Sharing repair station (SRS)*. The hot, warm and cold pools share a single repair station. Failed hot PMs have the repair priority over the failed PMs of the other pools, while failed warm PMs have priority over cold failed

PMs. The priority is non-preemptive. Similar to previous policy, PM repair time is exponentially distributed.

TABLE 1 summarizes definitions of system input parameters to be used in the following sections. $n_h, n_w, n_c, n_{rh}, n_{rw}$ and n_{rc} are design parameters, but MTTF, MTTR and MTM values could be experimentally measured. Note that we try to use notations similar to those used in [12] in order to highlight the difference of our models from those in [12] and then indicate the challenges of modeling in this paper.

TABLE 1 System Input Parameter Definition

Symbol	Description
n_h, n_w, n_c	Denote the initial numbers of non-failed PMs in the hot, warm and cold PM pools, respectively.
$1/\lambda_h, 1/\lambda_w, 1/\lambda_c$	Denote mean time to fail (MTTF) for hot, warm and cold PMs, respectively.
$1/\mu_h, 1/\mu_w, 1/\mu_c$	Denote mean time to repair (MTTR) for hot, warm and cold PMs, respectively.
n_{rh}, n_{rw}, n_{rc}	Denote repair facility number for hot, warm and cold repair stations, respectively. Used in IRS policy
n_r	Denote repair facility number, used for SRS policy
$1/\gamma_{wh}, 1/\gamma_{hw}, 1/\gamma_{hc}, 1/\gamma_{ch}, 1/\gamma_{cw}, 1/\gamma_{wc}$	Denote mean time to move (MTM) between two pools, e.g., γ_{wh} is the moving rate from warm pool to hot pool.

4 SYSTEM MODELS UNDER SRS POLICY

This section first presents monolithic SRN model under **SRS** repair policy. Then the corresponding scalable interacting SRN sub-models are given.

4.1 Monolithic SRN model

Figure 1 shows the monolithic SRN model for the availability analysis of IaaS cloud under **SRS** repair policy. The numbers of tokens in places P_h, P_w and P_c represent the number of non-failed PMs in hot, warm and cold pools respectively. The firing of each of transitions T_{bwhf}, T_{bchf} and T_{hf} represents the failure event of a hot PM. That is, there are three cases that will occur when a hot PM fails:

Case F1) A non-failed warm PM is available for moving to the hot pool, represented by firing T_{bwhf} ;

Case F2) The warm pool is empty but a non-failed cold PM is available, denoted by firing T_{bchf} ;

Case F3) Both the warm and cold pools are empty and then the failed hot PM cannot be substituted by any other pool PM, represented by firing T_{hf} .

Guard functions $[g_{1m}]$ and $[g_{2m}]$ are used to achieve the mutual exclusion of these three cases. All guard functions of the **SRS** monolithic model are described in TABLE A.1 of Appendix A. The overall hot PM failure rate is equal to λ_h multiplied by the number of non-failed hot PMs. Since only one case (case F1-F3) occurs at any time instant of a hot failed PM, the firing rate of each of T_{hf}, T_{bchf} and T_{bwhf} is $\#P_h \cdot \lambda_h$. Here, $\#P_h$ represents the number of tokens in place P_h . TABLE A.2 of Appendix A summarizes rates of transitions modeling PM failure in monolithic SRN model under **SRS** Policy.

We now detail the analysis for **Case F1: Warm PM moving occurs if a hot PM fails**. When transition T_{bwhf} fires, one token is taken from places P_w and P_h each, and one token is put in place P_{whm} , modeling a warm PM borrowed for moving to the hot pool. Each token in place P_{whm} represents a PM waiting for its moving completion. When a moving is completed, the firing of transition T_{whm} is triggered. Then one token is removed from place P_{whm} and one token is deposited to places P_h, P_{bwh} and P_{Bh} each. Rate of transition T_{whm} is determined by the number of tokens in place P_{whm} , allowing all moving PMs to move in parallel. Place P_{bwh} keeps track of the number of failed hot PMs, each of which is replaced by a warm PM. It also represents the number of warm PMs, having moved to the hot pool. Similar explanations can be applied to **Case F2: Cold PM moving occurrence in the case of hot PM failure**. All moving rates are given in TABLE A.3 of Appendix A.

Now we explain **Case F3: Neither warm nor cold PM is available for substitution of a failed PM**. In this case, transition T_{hf} fires. A token is removed from place P_{hf} modeling the reduction in the number of available PMs in the hot pool by one and a token is deposited in place P_{hf} representing the failed PM to be repaired and given back to the hot pool.

Failure events of PMs in the warm and cold pools are modeled in a similar way. Note that guard function $[g_{3m}]$ is defined to achieve mutual exclusion of transitions T_{bcwf} and T_{wcf} , each modeling a warm PM failure event. Note also that place P_{Bh}, P_{Bw} and P_{Bc} keep track of the number of failed PMs in the hot, warm and cold pool, respectively.

Output parameters of warm SRN sub-model. Warm SRN sub-model aims to compute p_w , $E[P_w]$, $E[P_{bwh}]$, $E[P_{wvf}]$ and $E[P_{cwm}]$. All these computed values are used as input parameters to cold pool SRN sub-model, while the first three p_w , $E[P_w]$ and $E[P_{bwh}]$ are used as input parameters to the hot pool SRN sub-model.

Output parameters of cold SRN sub-model. From cold pool sub-model, we compute p_c , $E[P_{bch}]$ and $E[P_{bcw}]$. $E[P_{bcw}]$ denotes the mean number of cold PMs moving to warm pool.

TABLE 4 Guard Functions for Interacting SRN Sub-Models under **SRS** Policy

Guard function	Value
$[g_{dh1}]$	if $((1-p_w)(1-p_c) > 0 \parallel \#P_{chm} \geq E[P_c])$ then 1 else 0
$[g_{dh2}]$	if $((1-p_w)p_c > 0 \parallel \#P_{whm} \geq E[P_w]) \&\&$ $\#P_{chm} < E[P_c]$ then 1 else 0
$[g_{dh3}]$	if $p_w > 0 \&\& \#P_{whm} < E[P_w]$ then 1 else 0
$[g_{dw1}]$	if $E[P_h] > 0$ then 1 else 0
$[g_{dw2}]$	if $((1-p_c) > 0 \parallel \#P_{cwm} \geq E[P_c])$ then 1 else 0
$[g_{dw3}]$	if $p_c > 0 \&\& \#P_{cwm} < E[P_c]$ then 1 else 0
$[g_{dw4}]$	if $E[P_{hf}] + E[P_{bch}] + \#P_{bwh} < n_r$ then 1 else 0
$[g_{dc1}]$	if $E[P_h] \cdot ((1-p_w) > 0 \parallel E[P_{whm}] \geq E[P_w])$ then 1 else 0
$[g_{dc2}]$	if $E[P_w] > 0$ then 1 else 0
$[g_{dc3}]$	if $E[P_{hf}] + E[P_{bwh}] + \#P_{bch} < n_r$ then 1 else 0
$[g_{dc4}]$	if $E[P_{hf}] + E[P_{wvf}] + E[P_{bwh}] + \#P_{bcw} + \#P_{bch} < n_r$ then 1 else 0

Failure behaviors. When the authors in [12] and [20] designed the rates of transition T_{hf} , T_{bchf} and T_{bwhf} in hot SRN sub-model, they modeled the impact of behaviors of warm and cold SRN sub-models. But they ignored that the tokens of P_{bchf} and P_{bwhf} should be less than the available PM number in the warm and cold pools, respectively. We propose that the PM failing transitions in hot SRN sub-model should be designed from the following three cases:

(Case FR1) $\#P_{whm} < E[P_w]$ and $\#P_{chm} < E[P_c]$. λ_h is divided to transition T_{hf} , T_{bchf} and T_{bwhf} according to p_w and p_c . Note that the authors in [12] only took into account this case.

(Case FR 2) $\#P_{whm} \geq E[P_w]$ and $\#P_{chm} < E[P_c]$. A failed hot PM will be replaced by a cool PM or be sent to be repaired. Namely, λ_h is divided to transition T_{hf} and T_{bchf} according to

p_w and p_c .

(Case FR 3) $\#P_{whm} \geq E[P_w]$ and $\#P_{chm} \geq E[P_c]$. The failed hot PM will be sent to be repaired. Thus, λ_h is only for transition T_{hf} .

Therefore, the rate of T_{bwhf} is $\#P_h \cdot \lambda_h \cdot p_w$ only when $p_w > 0$ and $\#P_{whm} < E[P_w]$. $T_{bchf} = \#P_h \cdot \lambda_h \cdot (1-p_w) \cdot p_c$ in the case of $(1-p_w)p_c > 0$ and $\#P_{chm} < E[P_c]$; when $p_w > 0$ and $\#P_{chm} < E[P_c]$, $T_{bchf} = \#P_h \cdot \lambda_h \cdot p_c$. For T_{hf} , its value is $\#P_h \cdot \lambda_h \cdot (1-p_w) \cdot (1-p_c)$ when $(1-p_w)(1-p_c) > 0$; its value is $\#P_h \cdot \lambda_h \cdot (1-p_w)$ when $\#P_{chm} \geq E[P_c]$. Following the above approach, we could compute the failure rates of transitions modeling PM failures in warm and cold sub-models. TABLE 4 defines guard functions to be used in hot, warm and cold pool sub-models. TABLE 5 details the rates of transitions modeling PM failures.

Repair behaviors. According to the **SRS** repairing policy in Section 4.1, only $\#P_{hf}$ failed hot PMs are sent back to the hot pool after repairing. The left are sent to warm and cold pools, which are modeled by transitions T_{whr} and T_{chr} in warm and cold sub-models, respectively. The total number of failed hot PMs seen in the hot pool is $(\#P_{hf} + E[P_{bwh}] + E[P_{bch}])$. The total number of failed hot PMs seen in warm and cold SRN sub-models is respectively $(E[P_{hf}] + P_{bwh} + E[P_{bch}])$ and $(E[P_{hf}] + E[P_{bwh}] + P_{bch})$. Equation (1) describes the rate of transition T_{hr} , in which only the number of failed hot PMs is considered in order to capture the highest repairing priority of the failed hot PMs. The rates of transition T_{whr} and T_{chr} in warm and cool pools are similarly defined.

$$\begin{cases} \#P_{hf} \cdot \mu_h, & \text{if } (\#P_{hf} + E[P_{bch}] + E[P_{bwh}]) > n_r \\ \frac{\#P_{hf} \cdot \mu_h \cdot n_r}{(\#P_{hf} + E[P_{bch}] + E[P_{bwh}])}, & \text{otherwise} \end{cases} \quad (1)$$

Following the above approach, we could define the rates of repairing warm failed PMs (namely, rates of T_{wr} and T_{cwr}) and the rates of repairing cold failed PM (namely, rate of T_{cr}). TABLE 6 describes the rates of transitions modeling PM repair.

4.2.2 SRN sub-model interactions

As in [12], the cyclic dependencies among the sub-models are solved by using fixed point iteration [24]. For each variable in Figure 2 (d), we could find a function of variables ($E[P_h]$, $E[P_{hf}]$, $E[P_w]$, $E[P_{wvf}]$, p_w , $E[P_{bwh}]$, $E[P_{bch}]$, $E[P_{bcw}]$, p_c , $E[P_c]$, $E[P_{whm}]$) to generate it. We use Ψ to denote these variables. Thus, the fixed point equation for solving the dependencies can be expressed as follows:

$$\mathbf{x} = \mathbf{H}(\mathbf{x}) \quad (2)$$

where $x=(\Psi)$ and H is a vector function over set $C \subset \mathbb{N} \rightarrow \mathbb{N}$. We can show the solution to Equation (2) exists by using the method similar to [12].

TABLE 5 Rates of Transitions Modeling PM Failure in Interacting SRN Sub-Models under *SRS* Policy

Transition	Firing rate
T_{hf} (hot)	if $\#P_{whm} < E(P_w) \&\& \#P_{chm} < E(P_c)$ then $\#P_h \cdot \lambda_h \cdot (1-p_w) \cdot (1-p_c)$ else if $\#P_{whm} \geq E(P_w) \&\& \#P_{chm} < E(P_c)$ then $\#P_h \cdot \lambda_h \cdot (1-p_c)$ else $\#P_h \cdot \lambda_h$
T_{bchf} (hot)	if $\#P_{whm} < E(P_w)$ then $\#P_h \cdot \lambda_h \cdot (1-p_w) \cdot p_c$ else $\#P_h \cdot \lambda_h \cdot p_c$
T_{bchf} (cold)	if $(E[P_{whm}] < E[P_w])$ then $E[P_h] \cdot \lambda_h \cdot (1-p_w)$ else $E[P_h] \cdot \lambda_h$
T_{bwhf} (hot)	$\#P_h \cdot \lambda_h \cdot p_w$
T_{bwhf} (warm)	$E[P_h] \cdot \lambda_h$
T_{wvf} (warm)	if $\#P_{cwm} < E(P_c)$ then $E[P_h] \cdot \lambda_h$ else $\#P_w \cdot \lambda_w$
T_{bcwf} (warm)	$\#P_w \cdot \lambda_w \cdot p_c$
T_{bcwf} (cold)	$E[P_w] \cdot \lambda_w$
T_{cf} (cold)	$\#P_c \cdot \lambda_c$

4.3 Performance measures

The measures considered in this paper are computed by using Markov reward approach [8]. Namely, a reward rate function is assigned at the SRN level and the expected reward rate at steady state is computed as the desired measures [8]. The measures considered in this paper include:

(i) Mean number of PMs in each pool. The mean number of non-failed PMs in the hot, warm and cold pool is given by the mean number of tokens in the corresponding place P_h , P_w and P_c (denoted as $E[P_h]$, $E[P_w]$ and $E[P_c]$, respectively).

(ii) Downtime. This is an availability measure of IaaS CDC infrastructure. Using redundant components is an effective method to improve system availability. A common form

of redundancy is a k -out-of- n system in which at least k components operate for the system success. We consider the IaaS CDC infrastructure to be available if the total number of non-failed hot PMs is greater than or equal to k , where $k \leq n_h$.

TABLE 6 Rates of Transitions Modeling the Repair of Failed PMs in Interacting SRN Sub-Models under *SRS* Policy

Transition	Firing rate
T_{hr} (hot)	if $E[P_{bch}] + E[P_{bwh}] + \#P_{hf} > n_r$ then $\frac{\#P_{hf} \cdot \mu_h \cdot n_r}{\#P_{hf} + E[P_{bch}] + E[P_{bwh}]}$ else $\#P_{hf} \cdot \mu_h$
T_{whr} (warm)	if $E[P_{hf}] + E[P_{bch}] + \#P_{bwh} \leq n_r$ then $\#P_{bwh} \cdot \mu_h$ else $\frac{\#P_{bwh} \cdot \mu_h \cdot n_r}{E[P_{hf}] + E[P_{bch}] + \#P_{bwh}}$
T_{chr} (cold)	if $E[P_{hf}] + E[P_{bwh}] + \#P_{bch} \leq n_r$ then $\#P_{bch} \cdot \mu_h$ else $\frac{\#P_{bch} \cdot \mu_h \cdot n_r}{E[P_{hf}] + E[P_{bwh}] + \#P_{bch}}$
T_{wr} (warm)	if $E[P_{bcw}] + E[P_{hf}] + E[P_{bch}] + \#P_{bwh} + \#P_{wf} > n_r$ then $\frac{\mu_w \cdot (n_r - E[P_{hf}] - E[P_{bch}] - \#P_{bwh}) \cdot \#P_{wf}}{\#P_{wf} + E[P_{bcw}]}$ else $\#P_{wf} \cdot \mu_w$
T_{cwr} (cold)	if $E[P_{hf}] + E[P_{wf}] + E[P_{bwh}] + \#P_{bcw} + \#P_{bch} \leq n_r$ then $\#P_{bcw} \cdot \mu_w$ else $\frac{\mu_w \cdot \max\{0, (n_r - E[P_{hf}] - E[P_{bwh}] - \#P_{bch})\} \cdot \#P_{bcw}}{E[P_{wf}] + \#P_{bcw}}$
T_{cr} (cold)	if $E[P_{bwh}] + E[P_{hf}] + E[P_{wf}] + \#P_{bcw} + \#P_{cf} + \#P_{bch} > n_r$ Then $\mu_c \cdot (n_r - E[P_{hf}] - E[P_{wf}] - E[P_{bwh}] - \#P_{bch} - \#P_{bcw})$ else $\#P_{cf} \cdot \mu_c$

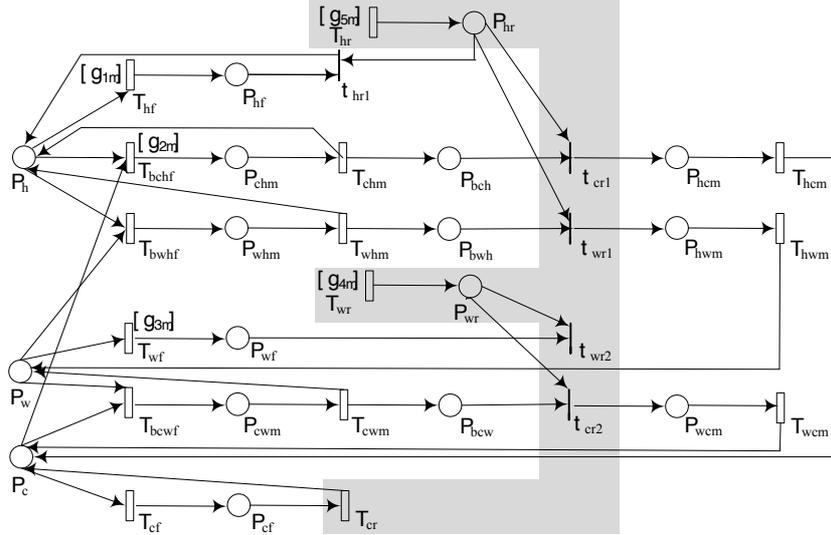


Figure 3 Monolithic SRN model under *IRS* policy

5. SYSTEM MODELS UNDER *IRS* POLICY

Figure 3 shows a monolithic SRN model for the availability analysis of IaaS cloud under *IRS* repair policy. The shaded parts denotes those differences from the model in [12]. Input parameters for this model are the same as those for the monolithic model under *SRS* repair policy, except that n_{rh} , n_{rw} and n_{rc} are used instead of n_r . Besides the guard functions defined in TABLE 4, this model requires two additional guard functions, defined in TABLE 7. TABLE 8 describes the rates of transitions regarding repair. The failure-related and moving-related transition rates depend on the number of tokens in corresponding places, similar to those in the monolithic model of *SRS* policy.

TABLE 7 Guard Functions for Monolithic SRN models under *IRS* Policy

Guard function	Value
$[g_{4m}]$	if $\#P_{wf} > 0 \parallel \#P_{bcw} > 0$ then 1 else 0
$[g_{5m}]$	if $\#P_{hf} > 0 \parallel \#P_{bch} > 0 \parallel \#P_{bwh} > 0$ then 1 else 0

TABLE 8 Rates of Transitions Modeling the Repair of Failed PMs in Monolithic SRN Model under *IRS* Policy

Transition	Firing rate
T_{hr}	if $\#P_{bwh} + \#P_{bch} + \#P_{hf} < n_{rh}$ then $(\#P_{bwh} + \#P_{bch} + \#P_{hf}) \cdot \mu_h$ else $n_{rh} \cdot \mu_h$
T_{wr}	if $\#P_{wfm} + \#P_{bcw} \leq n_{rw}$ then $(\#P_{wfm} + \#P_{bcw}) \cdot \mu_w$ else $n_{rw} \cdot \mu_w$
T_{cr}	if $\#P_{cf} \leq n_{rc}$ then $\#P_{cf} \cdot \mu_c$ else $n_{rc} \cdot \mu_c$

TABLE 9 Rates of Transitions Modeling the Repair of Failed PMs in Interacting SRN Sub-Models under *IRS* Policy

Transition	Firing rate
T_{hr} (hot)	if $E[P_{bwh}] + E[P_{bch}] + \#P_{hf} \geq n_{rh}$ then $\frac{\#P_{hf} \cdot \mu_h \cdot n_{rh}}{\#P_{hf} + E[P_{bch}] + E[P_{bwh}]}$ else $\#P_{hf} \cdot \mu_h$
T_{wr} (warm)	if $E[P_{bcw}] + \#P_{wfm} > n_{rw}$ then $\frac{\mu_w \cdot n_{rw} \cdot \#P_{wfm}}{\#P_{wfm} + E[P_{bcw}]}$ else $\#P_{wfm} \cdot \mu_w$
T_{whr} (warm)	If $E[P_{hf}] + E[P_{bch}] + \#P_{bwh} < n_{rh}$ then $\#P_{bwh} \cdot \mu_h$ else $\frac{\#P_{bwh} \cdot \mu_h \cdot n_{rh}}{E[P_{hf}] + E[P_{bch}] + \#P_{bwh}}$
T_{chr} (cold)	if $E[P_{hf}] + E[P_{bwh}] + \#P_{bch} < n_{rh}$ then $\#P_{bch} \cdot \mu_h$ else $\frac{\#P_{bch} \cdot \mu_h \cdot n_{rh}}{E[P_{hf}] + E[P_{bwh}] + \#P_{bch}}$
T_{cwr} (cold)	if $E[P_{wfm}] + \#P_{bcw} < n_{rw}$ then $\#P_{bcw} \cdot \mu_w$ else $\frac{\mu_w \cdot n_{rw} \cdot \#P_{bcw}}{E[P_{wfm}] + \#P_{bcw}}$
T_{cr} (cold)	$\#P_{cf} \cdot \mu_c$

For interacting SRN sub-models, structures of hot, warm and cold pool SRN sub-models are the same as those of *SRS* policy except for guard functions and the repairing-related rates. The rates of transitions modeling the repair of failed PMs are defined in TABLE 9. Note that there is no guard function for T_{wr} , T_{chr} and T_{cwr} in the interacting SRN sub-models

of *IRS* policy. That is, except for $[g_{dv4}]$, $[g_{dc3}]$ and $[g_{dc4}]$, the remaining guard functions defined in TABLE 4 are used in the interacting SRN sub-models under *IRS* policy. Note that the interactions among SRN sub-models are also equivalent for both repair policies.

6. NUMERICAL RESULTS

This section aims to evaluate the capability of our models. Reference [12] is close to our work. There are two modeling difference between [12] and our models. The first is the repair policy. As mentioned above, [12] required MTTRs of different pools to be same, which is a special case of our *IRS* model. Thus, the following doesn't compare our modeling work with [12] from the aspect of repair policy. The second difference is the PM-failure rates in hot and warm sub-models. We evaluate the effectiveness of our failure-related rate design in 6.2.

In the following, we first describe basic configurations in Section 6.1. Then, the validation of *SRS* interacting SRN sub-models is presented in Section 6.2. The verification of *IRS* interact SRN sub-models is presented in Appendix B. Section 6.3 presents the sensitivity analysis of CDC steady-state availability (*SSA*) with respect to repair policy and input parameters of *SRS* interact SRN sub-models.

6.1 Experiment configurations

SNPN software package [30] is used to solve the SRN models to obtain analytic-numeric and simulative results as well as sensitivity $S_\theta(SSA)$. The fixed point iteration approach used for solving the interacting SRN sub-models is implemented in Python. Parameters are configured according to [12] and [31]. Without loss of generality, the mean time to moving (MTM) between pools are set to 30 minutes. The PM mean time to repair (MTTR) of hot, warm and cold pool are set to 2, 5 and 10 hours. In default, MTTFs of hot, warm and cold PMs are assumed to be 500, 1750 and 2500 hours respectively. Note that simulation results are obtained by means of simulation of the corresponding monolithic model, which directly solves the stochastic net instead of generating the underlying Markov model. The following simulation results are obtained with a 95% confidence interval.

6.2 Validation of interacting SRN sub-models under *SRS* policy

The advantage of interacting SRN sub-models over their corresponding monolithic SRN model was detailed in [12] from the aspects of execution time and number of model states. This section focuses on validating approximate accuracy of our interacting SRN sub-models. These results are obtained by varying n_h from 3 to 10000. Unless otherwise specified, the other system input parameters are set as follows. $n_w = n_h$ when $n_h \leq 100$ otherwise, $n_w = 100$. $n_c = n_h$ when $n_h \leq 10$; otherwise, $n_c = 25$. “-” represents no solutions.

Comparison between monolithic SRN model and interacting SRN sub-models. When pool size (initial number of PMs in a pool) is greater than 5, the *SRS* monolithic SRN

model becomes unsolvable. Thus we do experiments by varying n_h from 3 to 5. k denotes the number of available hot PMs below which the IaaS CDC is assumed unavailable. For each pool size, we vary the value of k and report the results. TABLE 10 present downtime values (in minutes per year), indicating the increasing downtime value with increasing value of k . Note that as long as the number of non-failed hot PMs is consistent between hierarchical and monolithic models at each time instant, downtime is consistent. That is, the consistent downtime is just sufficient condition for verifying the accuracy of the interacting sub-models. It is necessary to further examine the number of warm and cold non-failed PMs. TABLE 11 presents mean number of non-failed PMs, validating the approximate accuracy of interacting SRN sub-models under these configurations.

TABLE 10 Comparison of Downtime Values (In Minutes per Year) between Monolithic SRN Model and Interacting SRN Sub-Models under *SRS* Policy

Pool size	k	Monolithic model	Interacting sub-models
3	1	0.000524	0.000524
	2	1.572603	1.572603
	3	1573.6516	1573.6516
4	2	0.0020945	0.00209
	3	3.14311	3.1431116
	4	2097.154493	2097.15449
5	3	0.0052324	0.0052324
	4	5.23503	5.23503107
	5	2620.1343	2620.134359

TABLE 11 Comparison of Mean #PMs in Each Pool between Monolithic SRN Model and Interacting SRN Sub-Models under *SRS* Policy

Pool size	Monolithic model			Interacting sub-models		
	hot	warm	cold	hot	warm	cold
3	2.997	2.975100	2.9896	2.997	2.97517	2.98960
4	3.996	3.966896	3.9862	3.996	3.96689	3.98615
5	4.995	4.958600	4.9827	4.995	4.95860	4.98270

Comparison with simulation results. When pool size is increasing, monolithic model can't work. Thus, simulation results are used to verify interacting sub-models. TABLE 12 and Figure 4 show the results by varying hot pool size n_h from 5 to 10000. Unless otherwise specified, “SRS-simu” denotes simulation results and “SRS-INum” denotes numerical results of interacting sub-models in the following. *SSA* is calculated by setting $k = n_h$. The other parameter values remain unchanged as default. We observe that:

(1) The analytic-numerical results approximate to simulation results from the aspect of the available PM number in each pool, shown in TABLE 12.

(2) The hierarchical model's accuracy decreases from the aspect of steady-state availability with the increasing n_h , shown in Figure 4. Analytic-numerical results of interacting SRN sub-models agree with simulation results when $n_h < 250$.

When $n_h \geq 250$, there exists inconsistency. The main reason is

that when n_h and n_r increase, more hot PMs will fail in unit time and then standby PMs should be moved to hot pool. Later the failed hot PM which completes its repair will return to the pool where the corresponding substitution PM comes from. Such frequent moving leads to large variation in the number of tokens in places in each sub-model. Our interacting sub-models use mean values to reflect the interaction between sub-models and then the large variation is covered, leading to the inconsistency between numerical and simulation results. This suggests the importance of designing interacting variables in Figure 2(d).

TABLE 12 Comparison of Mean Number of Non-Failed PMs in Each Pool between Numeric Solution of Interaction SRN Sub-Models and Simulation Results of the System under **SRS** Policy

Pool size	Interacting sub-models			Simulation		
	hot	warm	cold	hot	warm	cold
5	4.9950	4.9586	4.9827	4.9959	4.9641	4.9870
6	5.9940	5.9503	5.9792	5.9950	5.9565	5.9823
7	6.9930	6.9420	6.9758	6.9931	6.9497	6.9797
8	7.9920	7.9330	7.9723	7.9929	7.9396	7.9729
9	8.9910	8.9255	8.9688	8.9933	8.9333	8.9701
10	9.9900	9.9372	9.9262	9.9953	9.9450	9.9240
50	49.950	49.686	24.730	49.951	49.669	24.693
100	99.900	99.372	24.561	99.891	99.247	24.479
250	249.75	98.428	24.549	249.65	96.952	24.147
500	499.50	96.935	24.568	499.31	94.883	24.338
750	749.25	95.476	24.575	748.80	93.002	24.491
1000	999.00	93.973	24.579	998.31	90.670	24.538
2500	2497.5	84.991	24.610	2496.3	80.691	24.594
5000	4995.0	70.010	24.661	4993.9	65.865	24.647
7500	7492.5	55.029	24.713	7491.1	51.616	24.718
10000	9990.0	40.048	24.764	9987.7	38.586	24.796

MTTM between hot and warm pools varies from 30 minutes to 3 minutes. The other parameters are set as default. Figure 5 shows mean number of non-failed hot PMs. “SRS-simu” denotes simulation results. “SRS-INum-New” denotes numerical results by applying our interacting sub-models with failure transition rates set as in TABLE 5. “SRS-INum-old” denotes numerical results by applying our interacting sub-models but failure-related rates set as in [12]. As mentioned above, the inconsistency of the non-failed hot PM number between numerical and simulation results indicates interacting sub-models’ inaccuracy. “SRS-INum-old” results are far away from simulation results. Thus, the interacting sub-models with rates defined in TABLE 5 can capture the system behaviors more accurately.

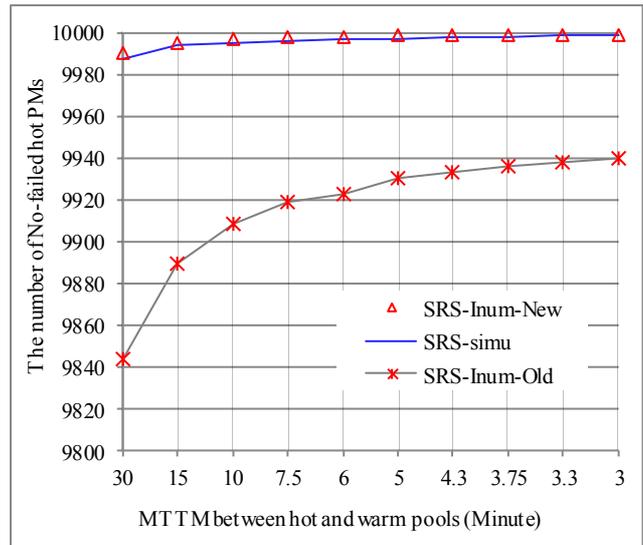


Figure 5 SSA comparison over moving rate under **SRS** policy

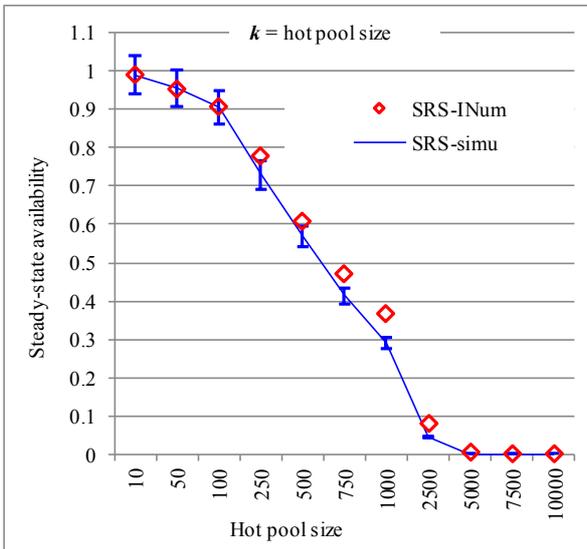


Figure 4 SSA over hot pool size under **SRS** policy

Comparison with [12] in terms of failure-related rates in each sub-model. We do experiments to show the problem of the failure-related rates in Fig.2 of [12]. n_h is set to 10000 and

6.3 Sensitivity analysis

This section applies interacting sub-models for sensitivity analysis. We first compare the effect of repair policy on system availability in Section 6.3.1. Section 6.3.2 focuses on **SRS** policy and applies the last two sensitivity analysis methods described in Section 2.2 to carry out sensitivity analysis with respect to system parameters. We compare **SRS** and **IRS** policies by assuming the cost of each repair facility is same and the cost of each PM in a pool is same. Then we could compare **SRS** and **IRS** by using only SSA or the number of non-failed PMs or the number of repair facilities.

6.3.1 The impact of repair policy on system availability based on numerical analysis of interacting sub-models

In some scenarios, **SRS** and **IRS** repair policies produce the same availability but **IRS** requires more standby PMs in order to maintain a certain level of availability. Namely, in these scenarios where there are not enough standby PMs, the system availability cannot be maintained to a certain level under **IRS** policy but **SRS** could. Thus, an IaaS CDC with **SRS** policy can maintain a level of availability with less cost than using **IRS** policy in terms of the number of standby non-failed PMs.

In the following two subsections, we compare two repair policies in terms of repair facility number and repair rate.

(A) Varying the number of repair facilities

Experiments are carried out by letting $n_h = 1000$, $n_w = n_c = 10$, $\mu_h = 2$ hours, $\mu_w = 50$ hours, $\mu_c = 100$ hours, $n_{rw} = n_{rc} = 1$. Other parameter values are set as default in Section 6.2. We do experiments by varying n_{rh} from 3 to 8. Note that $n_r = n_{rh} + n_{rw} + n_{rc}$. Figure 6 shows the results. “SRS-INum” denotes the numerical results under **SRS** policy. “IRS-Num” denote the numerical results under **IRS** policy. From Figure 6, we observe that 1) **SRS** policy can achieve higher availability than **IRS** policy under the same number of repair facilities; namely, **SRS** policy with less investment can achieve higher availability, compared to **IRS** policy; 2) increasing the number of repair facilities for hot pool can significantly improve system availability under **IRS** policy but this improvement stops after $n_r = 7$.

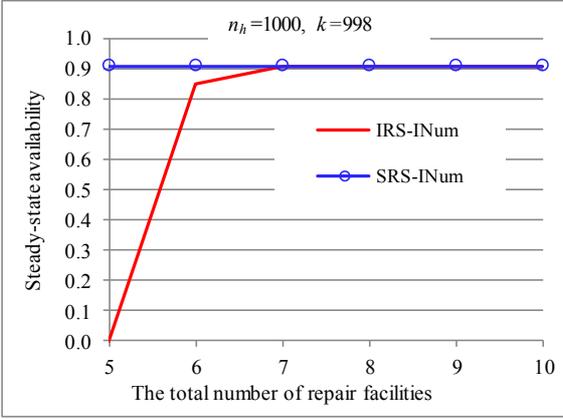


Figure 6 SSA over the total number of repair facilities for failed PMs

(B) Varying repair time

This subsection examines the effect of system repair capability (repair time of a failed hot PM) on system availability. We set $n_h = 1000$, $n_{rh} = 4$, $n_w = n_c = 10$, $n_{rw} = n_{rc} = 1$, $\mu_w = 50$ hours, $\mu_c = 100$ hours, $n_{rw} = n_{rc} = 1$. We do experiments by varying $1/\mu_h$ from 2 hours to 1 hour. The values of the other parameters remain unchanged as default in Section 6.2. Figure 7 shows SSA of interacting SRN sub-models under **IRS** and **SRS**. We observe that 1) **IRS** policy requires more powerful repair capability in order to achieve the same availability level as **SRS** when number of PMs are fixed; for example, the availability achieved under **IRS** at $\mu_h = 0.7$ is similar to under **SRS** at $\mu_h = 0.6$; 2) the increasing repair capability reduces the difference of SSA between two repair policies.

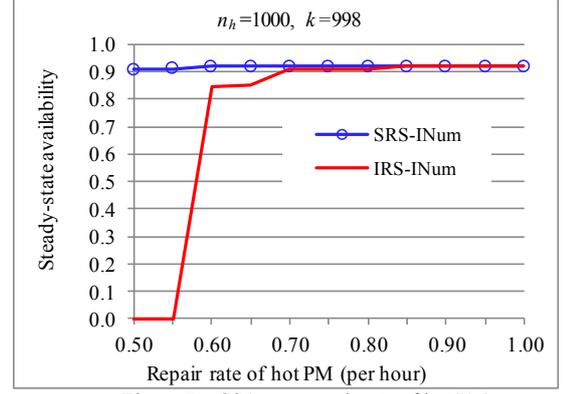


Figure 7 SSA over repair rate of hot PMs

6.3.2 The impact of input parameters on system availability under **SRS** policy

This subsection aims to investigate which system parameters are the most relevant to the improvement of system availability when **SRS** policy is applied. The last two approaches presented in Section 2.2 are used for sensitivity analysis. Let $n_h = 1000$. The values of the other parameters are set as in Section 6.2. We first use “sensitivity index” method to check the effect of n_r on availability by setting $n_r = 10$ and $n_r = 100$. The availability value variation is less, indicating that the SSA sensitivity to n_r is zero under this set of system parameters.

Now the direct method-based technique is used to analyze the first twelve parameters in the first column of TABLE 13. Note that we cannot calculate $S_\theta(SSA)$ due to the complicate interactions among of the interacting sub-models. But we could apply direct method to each sub-model and then rank the sub-model input parameters. By analyzing the sensitivity rank in each sub-model, we could get the parameters which produces the highest impact on system availability. The details are as follows.

Parametric sensitivity analysis is carried out to hot sub-model. The second column shows $S_\theta(SSA)$ of input parameters in the first column of TABLE 13. Note that all the values in TABLE 13 are obtained by using SPNP software package. “\” means that this parameter is not the input to the sub-model. Besides system parameters, $E[P_{bch}]$, $E[P_{bwh}]$, p_w and p_c are also input parameters to hot sub-model. But the second column does not show $S_{E[P_{bch}]}(SSA)$ and $S_{E[P_{bwh}]}(SSA)$ because $E[P_{bch}]$ and $E[P_{bwh}]$ are used to calculate T_{hr} in the hot sub-model and the computing formula of T_{hr} is complicated. See T_{hr} in TABLE 6. It is hard, if not impossible, to directly calculate $S_{E[P_{bch}]}(SSA)$ and $S_{E[P_{bwh}]}(SSA)$. From the numerical analysis results of the system parameter’ initial settings, we observe that $E[P_{bch}] + E[P_{bwh}] + \#P_{hf} > n_r$ seldom occurs under the system input parameter settings. That means both

$S_{E[P_{beh}]}(SSA)$ and $S_{E[P_{bwh}]}(SSA)$ are very small and then their effects on system availability could be ignored.

TABLE 13 Sensitivities with respect to Parameters for Each Pool

Parameter	Hot ($Y=SSA$)	Warm ($Y=p_w$)	Cold ($Y=p_c$)
λ_h	-4.9900E+02	4.2435E-21	0
λ_w	\	2.8473E-25	1.8556E-27
λ_c	\	\	1.8557E-27
μ_h	0.0000E+00	-1.6178E-15	0
μ_w	\	0.0000E+00	8.4722E-29
μ_c	\	\	-4.4067E-27
γ_{wh}	4.9900E-01	-9.5870E-22	\
γ_{ch}	0.0000E+00	0	0
γ_{cw}	\	-1.2332E-24	7.1499E-31
γ_{hw}	\	-9.5870E-22	\
γ_{hc}	\	\	0
γ_{wc}	\	\	-8.0158E-30
p_w	-9.98E-01	\	0
p_c	0.0000E+00	1.5997E-28	\

Although $S_{\theta}(SSA)$ of some parameters does not exist in the second column, such as λ_w and λ_c , these parameters may affect SSA. For example, λ_w and λ_c affect the system availability through p_w . Since $S_{p_w}(SSA)$ is -9.98E-01 and p_w is the output of warm sub-model, it is necessary to study warm sub-model. Direct method is applied to warm sub-model and the third column of TABLE 13 gives $S_{\theta}(p_w)$ of the related model input parameters. Similarly, we give the $S_{\theta}(p_c)$ of the related parameters in the fourth column because $S_{p_c}(p_w)$ exists in the third column. Here, its value is 1.5997E-28, very small. Note that the values of the 3rd column affect the system availability through p_w and the 4th column through $(p_w \cdot p_c)$. Since the values in the third and fourth columns are very small, they affect the system availability less, compared to the order of magnitude of γ_{wh} and λ_h in the 2nd column. That means that γ_{wh} and λ_h are the parameters most significantly affecting system availability. The third is μ_h . Since $\lambda_h=0.002$, $\mu_h=0.5$, $\gamma_{wh}=2$ and $SSA=0.9199$ under $k=998$, we could calculate $S_{\theta}(SSA)$ of γ_{wh} , λ_h and μ_h , shown in TABLE 14. The calculation details are as follows.

(1) $SS_{\lambda_h}(SSA) =$

$$\frac{\lambda_h}{SSA} \cdot (S_{\lambda_h}(SSA) + S_{p_w}(SSA) \cdot (S_{\lambda_h}(p_w) + S_{p_c}(p_w) \cdot S_{\lambda_h}(p_c)))$$

$$-1.08490533 = (-499 + 0.998 * ((-4.2435e-21) + (-1.5997e-28) * 0)) * 0.002 / 0.9199$$

(2) $SS_{\gamma_{wh}}(SSA) = \frac{\gamma_{wh}}{SSA} \cdot (S_{\gamma_{wh}}(SSA) + S_{p_w}(SSA) \cdot S_{\gamma_{wh}}(p_w))$

$$1.08490533 = (0.499 + 0.998 * (9.5870e-22)) * 2 / 0.9199$$

(3) $SS_{\mu_h}(SSA) =$

$$\frac{\mu_h}{SSA} \cdot (S_{\mu_h}(SSA) + S_{p_w}(SSA) \cdot (S_{\mu_h}(p_w) + S_{p_c}(p_w) \cdot S_{\mu_h}(p_c)))$$

$$8.7757E-16 = 0.5 / 0.9199 * (0 + 0.998 * (1.6178E-15))$$

TABLE 14 Scaled sensitivities for SSA

Parameter θ	$SS_{\theta}(SSA)$
λ_h	-1.0849038
γ_{wh}	1.0849038
μ_h	8.7757E-16

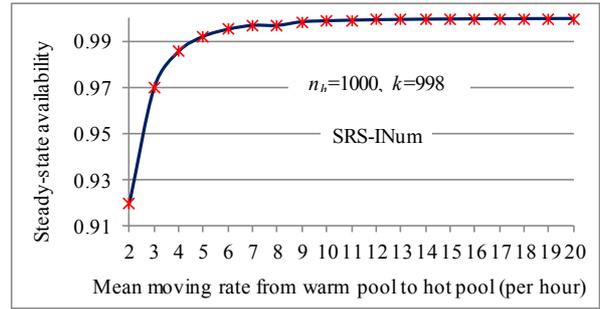


Figure 8 SSA over γ_{wh}

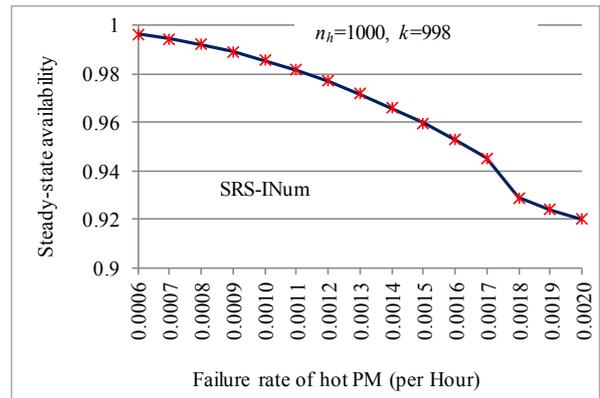


Figure 9 SSA over λ_h

Now we vary γ_{wh} and λ_h to investigate their effects on

system availability. First, $1/\gamma_{wh}$ varies from 0.05 hours to 2 hours but the other parameters are fixed. Figure 8 reveals the system steady-state availability increase quickly with the increasing γ_{wh} . Since it has a positive sensitivity value, system availability increases with the increased γ_{wh} . Figure 9 shows the impact of λ_h with a negative sensitivity value. Thus, reducing λ_h could improve SSA.

7. CONCLUSIONS AND FUTURE WORK

The primary focus of this paper is to explore stochastic modeling and sensitivity analysis techniques for analyzing the impact of repair policy and system parameters on the IaaS CDC availability. We examined two repair policies: (i) independent repair station (*IRS*), i.e., each pool having its own repair facilities; and (ii) sharing repair station (*SRS*), i.e., all PM pools share all repair facilities. Interacting SRN sub-models are developed for each policy. The corresponding monolithic SRN models are also developed, and their analytic numerical results and simulation results are used to verify the accuracy of the interacting SRN sub-models.

The numerical analysis results show that an IaaS CDC with *SRS* policy maintains the same level of availability with less cost than using *IRS* policy. As long as there are enough repair facilities and MTTR is small enough, there is no difference between these two repair policies in terms of system availability. Parametric sensitivity analysis based on the hierarchical models is conducted by applying differential analysis method to each sub-model.

We summarize future research as follows. This paper used an aggregate of failures occurring in a PM. We endeavor to extend the modeling approach proposed in this paper to capture influences caused by different types of failures. Various availability solutions for cloud were proposed [21]. Future work also includes using our proposed modeling approach to analyze these solutions and compare from a model-based analysis point of view. Large scale is a fact of life in CDCs. Thus, developing an effective approach for analyzing parametric sensitivity based on interacting-type hierarchical model and applying the hierarchical model to design the strategy for deploying standby PMs in an optimized way are our future research directions. Finally, in reality, a cloud repair system must include at least two types of repair facilities: automated (software-based) and manual in which human intervention is required. We plan to model that and then investigate how various parameters influence availability.

REFERENCES

- [1] <http://www.forbes.com/sites/louiscolombus/2015/09/27/roundup-of-cloud-computing-forecasts-and-market-estimates-q3-update-2015>.
- [2] Patricia Takako Endo, Moisés Rodrigues, Glauco Estacio Gonçalves, Judith Kelner, Djamel Fawzi Hadj Sadok, Calin Curescu: High availability in clouds: systematic review and research challenges. *J. Cloud Computing* 5: 16 (2016).
- [3] Vincent C. Emeakaroha, Marco Aurélio Stelmar Netto, Rodrigo N. Calheiros, Ivona Brandic, Rajkumar Buyya, César A. F. De Rose: Towards
- Autonomic Detection of SLA Violations in Cloud Infrastructures. *Future Generation Comp. Syst.* 28(7): 1017-1029 (2012).
- [4] Salman A. Baset: Cloud SLAs: Present and Future. *ACM SIGOPS Operating Systems Review.* 46(2): 57-66 (2012).
- [5] Google. *AppEngineServiceLevelAgreement*[Online]. (<https://developers.google.com/appengine/sla>); 2014.
- [6] <https://www.openstack.org/>
- [7] Kishor S. Trivedi, Ermeson C. Andrade, Fumio Machida: Combining Performance and Availability Analysis in Practice. *Advances in Computers* 84: 1-38 (2012).
- [8] <https://aws.amazon.com/disaster-recovery/>
- [9] K.S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications.* second ed., John Wiley & Sons, 2001.
- [10] Gianfranco Ciardo, Alex Blakemore, Philip F. Chimento Jr., Jogesh K. Muppala, Kishor S. Trivedi: Automated Generation and Analysis of MarkovReward Models Using Stochastic Reward Nets. *Mathematics and Its Applications: Linear Algebra, Markov Chains and Queueing Models*, vol. 48, pp. 145-191, Springer, 1993.
- [11] Hoon Choi, Varsha Mainkar, Kishor S. Trivedi: Sensitivity Analysis of Deterministic and Stochastic Petri Nets. *MASCOTS 1993*: 271-276.
- [12] Rahul Ghosh, Francesco Longo, Flavio Frattini, Stefano Russo, Kishor S. Trivedi: Scalable Analytics for IaaS Cloud Availability. *IEEE T. Cloud Computing* 2(1): 57-70 (2014).
- [13] Xiaolin Chang, Bin Wang, Jogesh K. Muppala, Jiqiang Liu: Modeling Active Virtual Machines on IaaS Clouds Using an M/G/m/m+K Queue. *IEEE Trans. Services Computing* 9(3): 408-420 (2016).
- [14] Xiaolin Chang, Tianju Wang, Ri, Zhenjiang Zhang: Modeling and Analysis of High Availability Techniques in a Virtualized System. *The Computer Journal*, 2017.
- [15] Tuan Anh Nguyen, Dong Seong Kim, Jong Sou Park: Availability Modeling and Analysis of A Data Center for Disaster Tolerance. *Future Generation Computer Systems*, 2016.
- [16] Gustavo Rau de Almeida Callou, Paulo Romero Martins Maciel, Eduardo Tavares, Erica Sousa, Bruno L. B. Silva, J. Figueiredo, Carlos Araújo, F. S. Magnani, F. Neves: Sustainability and Dependability Evaluation on Data Center Architectures. *SMC 2011*: 398-403.
- [17] Zi Hu, Liang Zhu, Calvin Ardi, Ethan Katz-Bassett, Harsha V. Madhyastha, John S. Heidemann, Minlan Yu: The Need for End-to-End Evaluation of Cloud Availability. *PAM 2014*: 119-130.
- [18] Hamzeh Khazaei, Jelena V. Mistic, Vojislav B. Mistic, Nasim Beigi Mohammadi: Availability Analysis of Cloud Computing Centers. *GLOBECOM 2012*: 1957-1962.
- [19] Kashi Venkatesh Vishwanath, Nachiappan Nagappan: Characterizing Cloud Computing Hardware Reliability. *SoCC 2010*: 193-204.
- [20] Francesco Longo, Rahul Ghosh, Vijay K. Naik, Kishor S. Trivedi: A Scalable Availability Model for Infrastructure-as-a-Service Cloud. *DSN 2011*: 335-346.
- [21] Mina Nabi, Maria Toeroe, Ferhat Khendek: Availability in the Cloud: State of the Art. *Journal of Network and Computer Applications*, Volume 59, 2016.
- [22] Mark Lanus, Liang Yin, Kishor S. Trivedi: Hierarchical Composition and Aggregation of State-based Availability and Performability Models. *IEEE Transactions on Reliability* 52(1): 44-52 (2003).
- [23] Kishor S. Trivedi, Ranjith Vasireddy, David Trindale, Swami Nathan, Rick Castro: Modeling High Availability. *PRDC 2006*: 154-164.
- [24] Lorrie A. Tomek, Kishor S. Trivedi: Fixed Point Iteration in Availability Modeling. *Fault-Tolerant Computing Systems 1991*: 229-240.
- [25] Gianfranco Ciardo, Kishor S. Trivedi: A Decomposition Approach for Stochastic Reward Net Models. *Perform. Eval.* 18(1): 37-59 (1993).

- [26] James T. Blake, Andrew L. Reibman, Kishor S. Trivedi: Sensitivity Analysis of Reliability and Performability Measures for Multiprocessor Systems. SIGMETRICS 1988: 177-186.
- [27] Rubens de S. Matos, Paulo Romero Martins Maciel, Fumio Machida, Dong Seong Kim, Kishor S. Trivedi: Sensitivity Analysis of Server Virtualized System Availability. IEEE Trans. Reliability 61(4): 994-1006 (2012).
- [28] Maria Clara Bezerra, Rosangela Melo, Jamilson Dantas, Paulo Romero Martins Maciel: Availability Evaluation of a VoD Streaming Cloud Service. SMC 2015: 765-770.
- [29] Rubens de S. Matos, Jean Araujo, Danilo Oliveira, Paulo Romero Martins Maciel, Kishor S. Trivedi: Sensitivity Analysis of a Hierarchical Model of Mobile Cloud Computing. Simulation Modelling Practice and Theory 50: 151-164 (2015).
- [30] Christophe Hirel, Bruno Tuffin, Kishor S. Trivedi: SPNP: Stochastic Petri Nets. Version 6.0. Computer Performance Evaluation / TOOLS 2000: 354-357.
- [31] Downtime statistics of current cloud solutions. 2013. <http://iwgcr.org/wp-content/uploads/2013/06/TWGCR-Paris.Ranking-003.2-en.pdf>.
- [32] Tuan Anh Nguyen, Dong Seong Kim, Jong Sou Park: Availability modeling and analysis of a data center for disaster tolerance. Future Generation Comp. Syst. 56: 27-50 (2016).

APPENDIX A. SYMBOL SETTINGS OF SRN MODELS UNDER *SRS* POLICY

The definitions of guard functions and transitions due to PM failure and moving in Figure 1 are same as in [12]. To make our paper complete, we describe these definitions in the following. TBLE A.1 shows the guard functions. TABLE A.2 and A.3 summarize rates of transitions modeling PM failures and PM moving, respectively. Note that the settings of PM moving rates in the interacting SRN sub-models are same as in TABLE A.3.

TABLE A.1 Guard Functions for Monolithic SRN Model under *SRS* Policy

Guard function	Value
$[g_{1m}]$	if $\#P_w=0$ and $\#P_c=0$ then 1 else 0
$[g_{2m}]$	if $\#P_w=0$ and $\#P_c>0$ then 1 else 0
$[g_{3m}]$	if $\#P_c=0$ then 1 else 0

TABLE A.2 Rates of Transitions Modeling PM Failures in Monolithic SRN Model under *SRS* Policy

Transition	Firing rate
T_{hf}	$\#P_h \cdot \lambda_h$
T_{bchf}	$\#P_h \cdot \lambda_h$
T_{bwhf}	$\#P_h \cdot \lambda_h$
T_{wvf}	$\#P_w \cdot \lambda_w$
T_{bcvf}	$\#P_w \cdot \lambda_w$
T_{cf}	$\#P_c \cdot \lambda_c$

TABLE A.3 Rates of Transitions Modeling PM Moving in Monolithic SRN

Transition	Firing rate
T_{whm}	$\#P_{whm} \cdot \gamma_{wh}$
T_{chm}	$\#P_{chm} \cdot \gamma_{ch}$
T_{cwm}	$\#P_{cwm} \cdot \gamma_{cw}$
T_{hcm}	$\#P_{hcm} \cdot \gamma_{hc}$
T_{hwm}	$\#P_{hwm} \cdot \gamma_{hw}$
T_{wcm}	$\#P_{wcm} \cdot \gamma_{wc}$

APPENDIX B. VERIFYING INTERACTING SRN SUB-MODELS UNDER *IRS* POLICY

This section aims to validate the approximate accuracy of interacting SRN sub-models under *IRS* policy. $n_{rh} = n_r \cdot 70\%$, $n_{rw} = n_r \cdot 20\%$ and $n_{rc} = n_r \cdot 10\%$. The settings of the other system input parameters and n_r are same as in the default settings of Section 6.2.

When pool size (initial number of PMs in a pool) is greater than 6, the monolithic SRN model under *IRS* policy becomes unsolvable. Thus we do experiment by varying n_i from 3 to 6. For each pool size, we vary the value of k and report the results. TABLE B.1 presents the downtime values (in minutes per year), indicating the increasing downtime value with increasing value of k . TABLE B.2 presents mean number of non-failed PMs, validating the approximate accuracy of interacting SRN sub-models under these configurations. When pool size is increasing, monolithic model can't work. Thus, simulation results are used to verify interacting sub-models. TABLE B.3 shows the results by varying hot pool size from 5 to 10000. The other parameter values remain unchanged as default. These results confirm the conclusion of Section 6.2.

TABLE B.1 Comparison of Downtime Values (In Minutes per Year) between Monolithic SRN Model and Interacting SRN Sub-Models under *IRS* Policy

Pool size	k	Monolithic model	Interacting sub-models
3	1	0.000524	0.000524
	2	1.572603	1.572603
	3	1573.6516	1573.6516
4	2	0.0020945	.00209
	3	3.14311	3.1431116
	4	2097.154493	2097.15449
5	3	0.0052324	0.0052324
	4	5.23503	5.23503107
	5	2620.1343	2620.134359
6	4	0.01045698	0.01045698
	5	7.847318	7.847318111
	6	3142.591768	3142.591768

TABLE B.2 Comparison of Mean #PMs in Each Pool between Monolithic SRN Model and Interacting SRN Sub-Models under *IRS* Policy

Pool	Monolithic model	Interacting sub-models
------	------------------	------------------------

size	hot	warm	cold	hot	warm	cold
3	2.997	2.97486	2.98958	2.997	2.97484	2.98958
4	3.996	3.96633	3.98609	3.996	3.96631	3.98609
5	4.995	4.95772	4.98259	4.995	4.95769	4.98259
6	5.994	5.94904	5.97909	5.994	5.94900	5.97909

Pool size	hot	warm	cold	hot	warm	cold
5	4.9950	4.9586	4.9827	4.9959	4.9641	4.9870
6	5.9940	5.9503	5.9792	5.9950	5.9565	5.9823
7	6.9930	6.9420	6.9758	6.9931	6.9497	6.9797
8	7.9920	7.9330	7.9723	7.9929	7.9396	7.9729
9	8.9910	8.9255	8.9688	8.9933	8.9333	8.9701
10	9.9900	9.9356	9.9240	9.9840	9.9360	9.9290
50	49.950	49.686	24.697	49.935	49.662	24.705
100	99.900	99.371	24.439	99.898	99.315	24.435
250	249.75	98.428	24.445	249.69	97.982	24.485
500	499.50	96.091	24.459	499.32	95.528	24.480
750	749.25	95.449	24.559	748.79	93.772	24.485
1000	999.00	93.800	24.565	998.37	91.452	24.491
2500	2497.5	84.896	24.610	2496.1	81.819	24.616
5000	4995.0	70.008	24.662	4993.7	67.178	24.495
7500	7492.5	55.029	24.713	7491.4	52.150	24.622
10000	9990.0	40.049	24.764	9989.0	41.226	24.524

TABLE B.3 Comparison of Mean Number of Non-Failed PMs in Each Pool between Numeric Solution of Interacting Sub-Models and Simulation Results under *IRS* Policy

	Interacting sub-models	Simulation
--	------------------------	------------