A Scalable Availability Model for Infrastructure-as-a-Service Cloud

Francesco Longo*, Rahul Ghosh‡, Vijay K. Naik§, and Kishor S. Trivedi‡
*Dipartimento di Matematica, Università degli Studi di Messina, Contrada di Dio, S.Agata, 98166 Messina, Italia
‡Dept. of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA
§IBM T. J. Watson Research Center, Hawthorne, NY 10532, USA
Email: flongo@unime.it,{rg51, kst}@ee.duke.edu, vkn@us.ibm.com

***Abstract*—High availability is one of the key characteristics of Infrastructure-as-a-Service (IaaS) cloud. In this paper, we show a scalable method for availability analysis of large scale IaaS cloud using analytic models. To reduce the complexity of analysis and the solution time, we use an interacting Markov chain based approach. The construction and the solution of the Markov chains is facilitated by the use of a high-level Petri net based paradigm known as stochastic reward net (SRN). Overall solution is composed by iteration over individual SRN sub-model solutions. Dependencies among the sub-models are resolved using fixed-point iteration, for which existence of a solution is proved. We compare the solution obtained from the interacting sub-models with a monolithic model and show that errors introduced by decomposition are insignificant. Additionally, we provide closed form solutions of the sub-models and show that our approach can handle very large size IaaS clouds.**

***Keywords*-Analytic model, availability analysis, cloud, fixed-point iteration, Markov models.**

I. Introduction

Cloud computing is a model of Internet-based computing. An IaaS cloud, such as Amazon EC2 [1] and IBM Smart Business Cloud [2], [3], delivers, on-demand, operating system (OS) instances provisioning computational resources in the form of virtual machines deployed in the cloud provider's data center. Requests submitted by the users are provisioned and served if the cloud has enough available capacity in terms of physical machines. Large cloud service providers such as IBM provide service level agreements (SLAs) regulating the availability of the cloud service. Before committing an SLA to the customers of a cloud, the service provider needs to carry out availability analysis of the infrastructure on which the cloud service is hosted. In this paper, we show how stochastic analytic models can be utilized for cloud service availability analysis. We first develop a one-level monolithic model. However, such monolithic models become intractable as the size of cloud increases. To overcome this difficulty, we use an interacting sub-models approach. Overall model solution is obtained by iteration over individual sub-model solutions. Comparison of the results with monolithic model shows that errors introduced by model decomposition are negligible. We also develop closed form solutions of the sub-models and show that our approach can scale for large size clouds. To the best of our knowledge, this is the first attempt to analyze availability of a cloud computing infrastructure by using stochastic analytic models. The presence of three pools of physical machines and the migration of them from one pool to another caused by failure events makes the model both novel and interesting. In order to automate the construction and solution of underlying Markov models, we use a variant of stochastic Petri net (SPN) called stochastic reward net (SRN). This paradigm is supported by two of our own software packages, SHARPE [4] and SPNP [5].

Rest of the paper is organized as follows. Section II describes cloud system model, assumptions and problem formulation. Section III, presents the monolithic SRN model. Interacting SRN sub-models are described in Section IV and their closed form solutions are presented in Section V. Fixed point iteration among the interacting sub-models and proof of existence of a solution is shown in Section VI. Results obtained from monolithic approach and interacting sub-models approach are compared in Section VII. Related research is highlighted in Section VIII. We conclude this work and discuss future avenues of research in Section IX. Appendix A provides backgrounds on SPNs and SRNs.

II. Problem definition

System model and assumptions. In IaaS cloud, when a request is processed, a pre-built image is used to create one or more Virtual Machine (VM) instances [6]. When the VM instances are deployed, they are provisioned with request specific CPU, RAM, and disk capacity. VMs are deployed on physical machines (PMs) each of which may be shared by multiple VMs. To reduce overall VM provisioning delays and operational costs, we assume that the PMs are grouped into three pools; hot (running), warm (turned on, but not ready) and cold (turned off). Maintaining the PMs in three pools (in general, multiple tiered pools) helps to minimize power and cooling costs without incurring high startup delays for all VMs. A pre-instantiated VM can be readily provisioned and brought to ready state on a running PM (hot PM) with minimum provisioning delay. Instantiating a VM from an image and deploying it on a warm PM needs additional provisioning time. PMs in the cold pool are turned-off when not in use and deploying a VM on such a PM adds to the startup delays. A performability model of

978-1-4244-9233-6/11/$26.00 ©2011 IEEE

this system was presented in [6] where it has been shown that the "bottleneck" model is the availability model. Hence, the objective of this paper is to develop a scalable availability model of cloud service with the following assumptions:

(1) Variety of failures/repairs can occur in a cloud environment such as failure/repair of hardware, software, hypervisor, VM, OS and applications. In this paper, we consider only the net effect of different failures and repairs of PMs in the hot, warm and cold pools. We do not consider software and OS failures in a VM. Typically, these failures are handled by restarting the VM. Although the cause of a PM failure can be because of variety of reasons, in our analysis we consider the net combined effect on the PM failure rate. In future, we plan to extend our availability model to capture detailed PM failure modes and recovery steps as in [7], [8].

(2) We assume that all times to failure are exponentially distributed. Equivalent mean time to failure (MTTF) of each hot PM is $1/\lambda_h$ and that of each warm PM is $1/\lambda_w$. Typically, $1/\lambda_w$ is higher than $1/\lambda_h$ by a factor of 2 to 4. Cold machines can fail with a very low failure rate λ_c with $\lambda_h >> \lambda_c$ and $\lambda_w >> \lambda_c$. We will remove the assumption of exponential distribution in future as in [9].

(3) All PMs in a pool are identical. Failure of a PM in one pool triggers migration of a PM (if available) from other pools to replace the failed one. When a hot PM fails, the failed PM needs to be repaired and at the same time the system tries to replace it by a warm PM, if available (i.e., in "UP" state). If no warm PM is available, replacement is attempted by migrating an available cold PM to the hot pool. When a warm PM fails, the failed PM undergoes repair and at the same time it is replaced by a PM from the cold pool (if there is at least one PM available in cold pool). We assume that the migration process is instantaneous.

(4) Each pool has its own repair facilities. Within a pool, maximum number of PMs that can be repaired in parallel is assumed to be n_r. Value of n_r is assumed to be greater than equal to 1 but less than the maximum number of PMs in the pool. When the number of PM failures are higher than n_r, failed PMs are put in a queue for repair. Across different pools, repairs can be done in parallel. We assume that time to repair is exponentially distributed with mean $1/\mu$. Once a failed PM is repaired, it is returned to the original pool where it belonged before failure. If a PM was borrowed from other pool to replace the failed PM; such borrowed PM is also returned to its original pool instantaneously.

Problem Formulation. Assume that n_h, n_w and n_c PMs are initially available in the hot, warm, and cold pools, respectively. Our definition of availability is that at least k PMs (with $1 \leq k \leq n_h + n_w + n_c$) should be available across all the pools combined in order for the system to be up. Under the failure, repair and migration of the PMs across different pools, we wish to compute the average number of PMs in each pool at steady state and the effects of downtime on the cloud service. Note that the migration of PMs from one pool to another induces a dependence between the three pools making the availability model both interesting and novel.

We start by developing a monolithic model using the high-level formalism of SRN for the automated generation and solution of the underlying Markov chain. The monolithic model is not scalable to the large size clouds that we wish to analyze. Hence, we propose an interacting SRN sub-models approach that is scalable. As an important side-benefit, the decomposition also enables us to obtain closed form solutions of sub-models. Three key comparisons are made between these two approaches: (1) errors introduced by interacting sub-models, (ii) maximum number of PMs that each approach can handle and (iii) solution time required for both the approaches. Through systematic analysis, we show that interacting SRN sub-models approach is highly scalable compared to single monolithic modeling approach. Closed form solutions of the sub-models are especially useful in providing a highly scalable and fast method for the availability analysis of large sized IaaS cloud.

III. Monolithic availability model

Monolithic SRN model for the availability analysis of IaaS cloud is shown in Fig. 1. Input parameters of monolithic

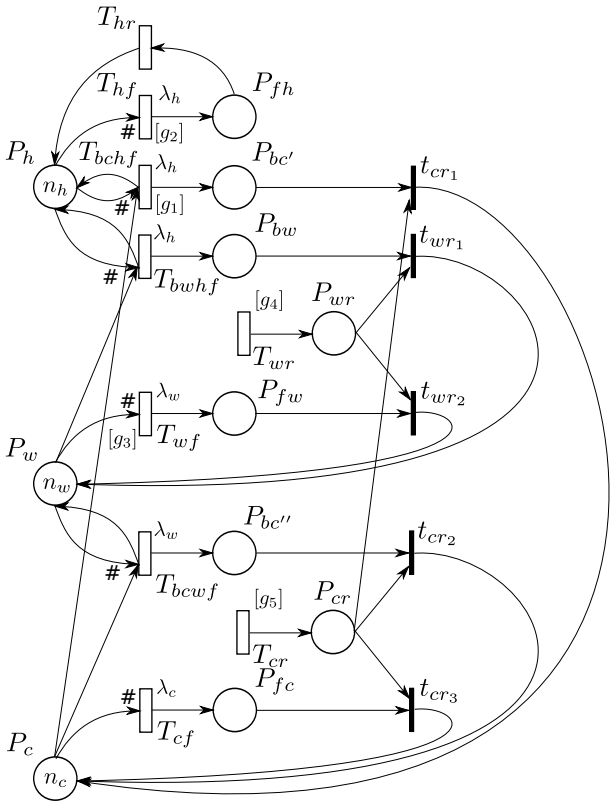

Figure 1. Monolithic SRN model for availability analysis of IaaS cloud.

model are: (1) initial number of PMs in each pool (n_h, n_w,

and n_c), (2) MTTFs of hot, warm, and cold PMs ($1/\lambda_h$, $1/\lambda_w$ and $1/\lambda_c$, respectively), (3) number of repair facilities for each pool (n_r), (4) MTTR of a PM ($1/\mu$). Among the input parameters, n_h, n_w, n_c, and n_r are design parameters, MTTF and MTTR values are measured. Five guard functions are defined on the model and they are described in Table I[1].

Table I
GUARD FUNCTIONS DEFINED ON MONOLITHIC SRN MODEL AND INTERACTING SRN SUB-MODELS.

Guard functions	Values
g_1	1 if $\#P_w = 0$ 0 otherwise
g_2	1 if $\#P_w = 0$ and $\#P_c = 0$ 0 otherwise
g_3	1 if $\#P_c = 0$ 0 otherwise
g_4	1 if $\#P_{fw} + \#P_{bw} > 0$ 0 otherwise
g_5	1 if $\#P_{fc} + \#P_{bc'} + \#P_{bc''} > 0$ 0 otherwise

Places P_h, P_w, and P_c represent the hot, warm, and cold pool, respectively. Number of tokens in these places indicates the number of "UP" PMs in the corresponding pool. Transitions T_{bwhf}, T_{bchf}, and T_{hf} represent the failure event of a hot PM. Since migration of a PM is attempted upon failure of a hot PM, three cases are possible: (1) T_{bwhf} fires if a warm PM is available for migration to the hot pool, (2) T_{bchf} fires if the warm pool is empty but a cold PM is available to be borrowed, and (3) T_{hf} fires if both the warm and the cold pool are empty so that no PM is available to substitute the failed hot PM. The guard functions g_1 and g_2 model the three mutually exclusive cases. Moreover, rates of these transitions are considered to be dependent on the number of tokens in place P_h so that the overall hot PM failure rate is equal to λ_h multiplied by the number of available hot PMs. These marking dependent firing rates are represented by the $\#$ symbol near the input arcs which connect the transitions T_{bwhf}, T_{bchf}, and T_{hf} to the place P_h.

Upon firing of transition T_{bwhf}, a token is removed from place P_w and the number of tokens in place P_h remains unchanged. At the same time, a token is deposited in place P_{bw}. This place keeps track of number of failed PMs that need to be repaired and given back to the warm pool at the end of the repair process. Similarly, upon firing of transition T_{bchf}, a token is removed from place P_c and the number of tokens in place P_h remains unchanged. Simultaneously, a token is deposited in place $P_{bc'}$ to take into account that a PM has to be repaired and given back to the cold pool. Upon firing of transition T_{hf}, following token exchanges happen: (i) removal of a token from place P_h to model the reduction in number of available PMs in the hot pool by one and (ii) deposition of a token in place P_{fh} to model that the failed PM has to be repaired and given back to the hot pool.

Failure-repair behavior of warm pool is modeled similarly. Transitions T_{bcwf} and T_{wf} model the failure event of a warm PM. Two cases are possible: (1) T_{bcwf} fires if a cold PM is available for migration to warm pool, and (2) T_{wf} fires if the cold pool is empty and no PM is available to substitute the failed warm PM. The guard function g_3 models the fact that the two cases are mutually exclusive. Rates of transitions T_{bcwf} and T_{wf} are considered to be dependent on the number of tokens in place P_w so that the overall warm PM failure rate is equal to λ_w multiplied by the number of available warm PMs. Firing of T_{bcwf} removes a token from place P_c and deposits a token to place $P_{bc''}$ representing the failed PM that needs to be repaired and given back to the cold pool. Upon firing of T_{wf}, a token is removed from place P_w and a token is deposited to place P_{fw} representing the failed PM that needs to be repaired and given back to the warm pool.

Transition T_{cf} fires when a cold PM fails. Rate of such transition is considered to be dependent on the number of tokens in place P_c so that the overall cold PM failure rate is equal to λ_c multiplied by the number of available cold PMs. Upon firing of T_{cf}, a token is removed from place P_c and deposited to place P_{fc}.

Transitions T_{hr}, T_{wr} and T_{cr} model the repair of the failed PMs. Rates of these transitions are marking dependent to take into account the presence of n_r repair facilities for each pool. In particular, the rates of the above mentioned transitions are reported in Table II. Guard functions g_4 and g_5 allow transitions T_{wr} and T_{cr} to be enabled only when at least one PM needs to be repaired. Immediate transitions t_{wr_1}, t_{wr_2}, t_{cr_1}, t_{cr_2}, and t_{cr_3} model the instantaneous migrations of repaired PMs to the original pool.

Model outputs. Outputs of the model are obtained using the Markov reward approach by assigning an appropriate reward rate to each marking of the SRN and then computing the expected reward rate both in transient and steady state as the desired measures [7]. Let r_i be the reward rate assigned to marking i of the SRN in Figure 1. If $\pi_i(t)$ denotes the probability for the SRN to be in marking i at time t then the expected reward rate at time t is given by $\sum_i \pi_i(t) r_i$. The expected steady state reward rate can be computed by taking into consideration the steady state probabilities π_i of the SRN as $\sum_i \pi_i r_i$. Our measures of interest are following.

(i) Mean number of PMs in each pool The mean number of PMs in the hot pool is given by the mean number of tokens in the corresponding place P_h ($E[\#P_h]$). Similarly, for the warm and the cold pool we consider the mean number of tokens in places P_w and P_c, respectively ($E[\#P_w]$ and $E[\#P_c]$). Reward assignment for this measures is shown in Table III.

(ii) Availability of cloud service (A) As mentioned above, we consider the cloud service to be available if the

[1]The notation $\#P$ indicates the number of tokens in place P.

Table II
RATES OF TRANSITIONS MODELING THE REPAIR OF FAILED PMS IN MONOLITHIC SRN MODEL AND INTERACTING SRN SUB-MODELS.

Transitions	Rates of transitions
T_{hr}	$\#P_{fh} \cdot \mu$ if $\#P_{fh} \leq n_r$ $n_r \cdot \mu$ otherwise
T_{wr}	$(\#P_{fw} + \#P_{bw}) \cdot \mu$ if $\#P_{fw} + \#P_{bw} \leq n_r$ $n_r \cdot \mu$ otherwise
T_{cr}	$(\#P_{fc} + \#P_{bc'} + \#P_{bc''}) \cdot \mu$ if $\#P_{fc} + \#P_{bc'} + \#P_{bc''} \leq n_r$ $n_r \cdot \mu$ otherwise

Table III
REWARD RATES TO COMPUTE DIFFERENT OUTPUT MEASURES FROM MONOLITHIC SRN MODEL AND INTERACTING SRN SUB-MODELS

Measures	**Reward rates**
Mean number of PMs in the hot pool ($E[\#P_h]$)	$\#P_h$
Mean number of PMs in the warm pool ($E[\#P_w]$)	$\#P_w$
Mean number of PMs in the cold pool ($E[\#P_c]$)	$\#P_c$
Availability of cloud service (A)	1 if $(\#P_h + \#P_w + \#P_c) \geq k$; 0 o/w
Availability of hot pool (A_{k_h})	1 if $\#P_h \geq k_h$; 0 o/w
Availability of warm pool (A_{k_w})	1 if $\#P_w \geq k_w$; 0 o/w
Availability of cold pool (A_{k_c})	1 if $\#P_c \geq k_c$; 0 o/w
Probability to have at least one PM in warm pool (p_w)	1 if $\#P_w \geq 1$; 0 o/w
Probability to have at least one PM in cold pool (p_c)	1 if $\#P_c \geq 1$; 0 o/w

total number of PMs across all hot, warm, and cold pool is greater than or equal to k (with $1 \leq k \leq n_h + h_w + n_c$). As a consequence, the reward assignment for this measure is the one shown in Table III.

IV. INTERACTING SRN SUB-MODELS

We decompose the monolithic model into three sub-models, each of which captures the failure and repair behavior of a single pool. Here, we describe how these three sub-models interact with each other and allow us to compute the same quantities that are computed from the monolithic model. The SRN sub-models for the hot, warm and cold pool are shown in Figures 2, 3 and 4, respectively. Observe that some of the transitions of the monolithic model are present in more than one sub-model. Hence, to obtain overall model solution, sub-models exchange some of the input parameters and output measures. Guard functions g_1, g_2, and g_3 are not present in the interacting sub-models approach while the rates of transitions T_{hr}, T_{wr}, T_{cr} are still marking dependent according to the functions described in Table II.

The structure of the hot pool sub-model in Figure 2 is obtained from the structure of the monolithic model by keeping the transitions that directly interact with place P_h and disregarding the others and the related places. Input parameters to this sub-model are: (i) initial number of PMs in hot pool ($n_{_h}$), (ii) hot PMs failure rate (λ_h), (iii) hot PMs repair rate (μ), (iv) number of repair facilities in the hot pool (n_r). Among these input parameters, $n_{_h}$ and n_r are design parameters, λ_h and μ are measured. Assume p_w and p_c are the probabilities to have at least one PM available in warm and cold pool, respectively as computed from the warm and cold pool sub-models discussed later. In the hot pool sub-model, the rate of transition T_{bwhf} is $\lambda_h \cdot p_w$. This is because,

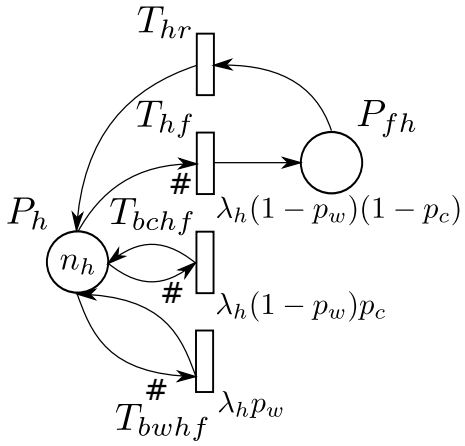

Figure 2. SRN sub-model for the availability analysis of the hot pool.

in the monolithic model, an arc is present from place P_w to such a transition. In the hot pool sub-model, place P_w is not present but still the impact of the behavior of the warm pool sub-model on the throughput of transition T_{bwhf} needs to be taken into account by scaling its rate with the quantity p_w. Similarly, the rate of transition T_{bchf} is $\lambda_h \cdot (1-p_w) \cdot p_c$ because it is necessary to take into account the presence of the arc from place P_c (by multiplying with p_c) and the guard function $[g_1]$ (by multiplying with $1-p_w$) as used in the monolithic model. Finally, the rate of transition T_{hf} is $\lambda_h \cdot (1-p_w) \cdot (1-p_c)$ because we need to take into account the presence of the guard function $[g_2]$.

From the hot pool sub-model we compute $E[\#P_h]$ that represents the mean number of tokens in place P_h, i.e., mean number of available PMs in the hot pool. It will be used in

the warm and cold pool SRN sub-models to approximate the rate of transitions T_{bwhf} and T_{bchf}. Moreover, from the hot pool sub-model we compute the probability (A_{k_h}) for the hot pool to be available, i.e., the probability for the number of tokens in place P_h to be greater or equal to k_h with $0 \leq k_h \leq n_h$. It will be used to compute the overall cloud service availability from the interacting sub-models. These output measures can be computed by assigning the reward rates reported in Table III.

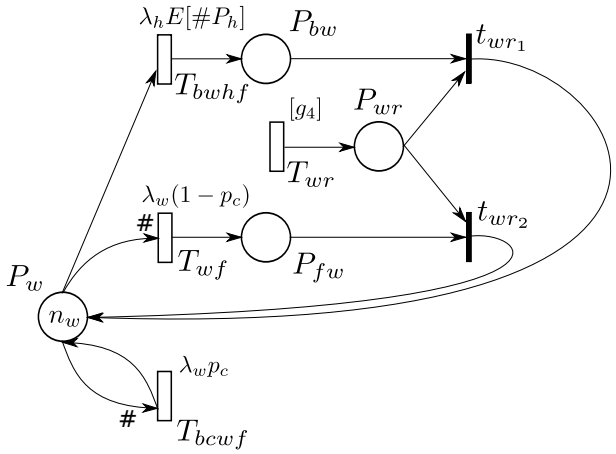

Figure 3. SRN sub-model for the availability analysis of the warm pool.

Similar to the hot pool sub-model, the structure of the warm pool sub-model in Figure 3 is obtained from the structure of the monolithic model by keeping the transitions that directly interact with place P_w and disregarding the others and the related places. Input parameters to this sub-model are: (i) initial number of PMs in warm pool (n_w), (ii) warm PMs failure rate (λ_w), (iii) warm PMs repair rate (μ) and (iv) number of repair facilities for the warm pool (n_r). Among these input parameters, n_w and n_r are design parameters, λ_w and μ are measured. Computation of rate of transitions T_{bcwf} and T_{wf} is similar to the computation of rate of transitions $T_{bwhf}, T_{bchf}, T_{hf}$ as described for hot pool sub-model. Probability p_c is obtained from cold pool sub-model. However, the rate of transition T_{bwhf} needs to be set so that the throughput of this transition and the throughput of the transition with the same name in the hot pool sub-model are equal. In fact, the two transitions are same in the monolithic model. In the hot pool sub-model, the expected throughput of transition T_{bwhf} is given by:

$$\begin{aligned} Th_h(T_{bwhf}) &= \sum_{i=0}^{n_h} i \cdot \lambda_h \cdot p_w \cdot p(\#P_h = i) \\ &= \lambda_h \cdot p_w \cdot E[\#P_h] \end{aligned} \quad (1)$$

where $p(\#P_h = i)$ is the probability that the number of tokens in place P_h is equal to i. Let $rate_w(T_{bwhf})$ be the rate of transition T_{bwhf} in warm pool sub-model. Then, its expected throughput in this sub-model is given by:

$$\begin{aligned} Th_w(T_{bwhf}) &= \sum_{i=0}^{n_w} rate_w(T_{bwhf}) \cdot p(\#P_w = i) \\ &= rate_w(T_{bwhf}) \cdot p(\#P_w > 0) \\ &= rate_w(T_{bwhf}) \cdot p_w \end{aligned} \quad (2)$$

Given that we want $Th_w(T_{bwhf}) = Th_h(T_{bwhf})$ we can compute what should be the value of $rate_w(T_{bwhf})$:

$$rate_w(T_{bwhf}) = \lambda_h \cdot E[\#P_h] \quad (3)$$

where $E[\#P_h]$ is obtained from hot pool sub-model.

Outputs of warm pool sub-model are: (i) probability (p_w) to have at least one token in place P_w, i.e., at least one PM is available in the warm pool, (ii) mean number of tokens ($E[\#P_w]$) in place P_w, i.e., mean number of available PMs in the warm pool, and probability (A_{k_w}) for the number of tokens in place P_w to be greater or equal to k_w (with $0 \leq k_w \leq n_w$), i.e., availability of the warm pool. Among these output measures p_w will be used as an input parameter to the hot pool SRN sub-model to approximate the rates of transitions T_{bwhf}, T_{bchf} and T_{hf}, $E[\#P_w]$ will be used as an input parameter to the cold pool SRN sub-model to approximate the rate of transition T_{bcwf}, and A_{k_w} will be used to compute the overall cloud service availability from the interacting sub-models. The reward rates assignment for such output measures are shown in Table III

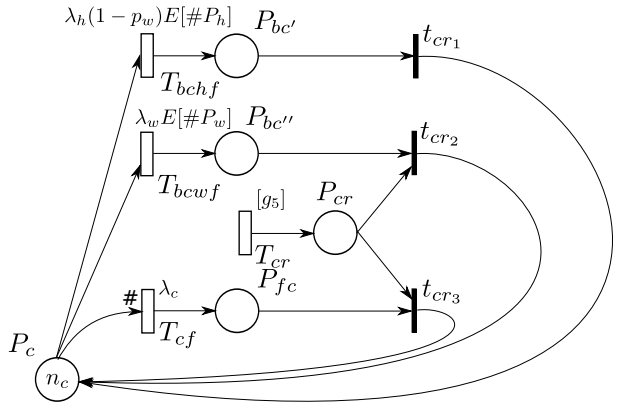

Figure 4. SRN sub-model for the availability analysis of the cold pool.

Also in the case of cold pool, the structure of the sub-model in Figure 4 can be obtained from the structure of the monolithic model by keeping the transitions that directly interact with place P_c and disregarding the others and the related places. Input parameters to the cold pool sub-model are: (i) initial number of PMs in cold pool (n_c), (ii) cold PMs failure rate (λ_c), (iii) cold PMs repair rate (μ) and (iv) number of repair facilities for the cold pool (n_r). Among these input parameters, n_c and n_r are design parameters, λ_c and μ are measured. Following similar arguments as in the

case of warm pool, we can compute the rate of transitions T_{bchf} and T_{bcwf} in the cold pool sub-model:

$$rate_c(T_{bchf}) = \lambda_h \cdot (1 - p_w) \cdot E[\#P_h] \quad (4)$$

and

$$rate_c(T_{bcwf}) = \lambda_w \cdot E[\#P_w] \quad (5)$$

where p_w and $E[\#P_w]$ are obtained from warm pool sub-model, and $E[\#P_h]$ is obtained from hot pool sub-model.

Output measures of cold pool sub-model are: (i) probability (p_c) to have at least one token in place P_c, i.e., at least one PM is available in the cold pool, (ii) mean number of tokens ($E[\#P_c]$) in place P_c, i.e., mean number of available PMs in the cold pool, and (iii) the probability (A_{k_c}) for the number of tokens in place P_c to be greater or equal to k_c (with $0 \leq k_c \leq n_c$), i.e., the cold pool availability. p_c will be used as an input parameter to the hot and warm SRN sub-models to approximate the rate of transitions T_{bchf}, T_{hf}, T_{bcwf} and T_{wf}. A_{k_c} will be used to compute the overall cloud service availability. Reward assignments for such output measures are reported in Table III.

All these sub-models and the interactions among them are shown as an import graph in Figure 5. We briefly describe the interactions among these models here. The hot pool sub-model computes the mean number of PMs in the hot pool ($E[\#P_h]$) that is needed as an input parameter to both the warm and cold pool sub-models. The warm pool sub-model compute the probability for the warm pool to have at least one available PM (p_w) and the mean number of PMs in the warm pool ($E[\#P_w]$). The former quantity is used both in the hot and cold pool sub-models while the latter is used in the cold pool sub-model. Finally, the output measure of cold pool sub-model (p_c, i.e., the probability for the cold pool to have at least one available PM) is used both in the hot and warm pool sub-models. Observe, the import graph shows cyclic dependencies among the sub-models. Such dependencies are resolved using fixed point iteration [10], [11].

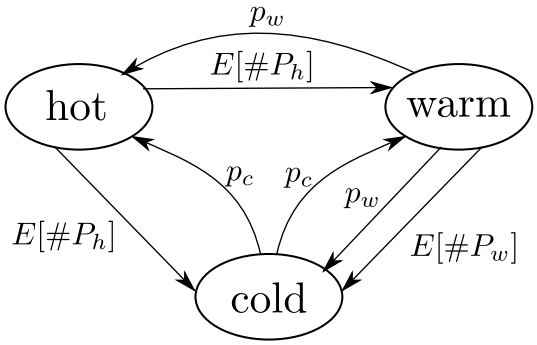

Figure 5. Interactions among the sub-models as an import graph.

Model outputs. Once the interacting sub-models have been solved, the same output measures of interest for the availability analysis of the cloud service that can be computed from the monolithic model can be obtained. In particular, the mean number of PMs in each pool ($E[\#P_h]$, $E[\#P_w]$, and $E[\#P_c]$) are immediately available from the hot, warm, and cold pool sub-models, respectively. The availability of cloud service for a given k can be computed by combining the availability of hot, warm, and cold pool as computed from hot, warm, and cold pool sub-models such that $k_h + k_w + k_c \geq k$.

V. Closed form solution of the sub-models

In this section, we show the closed form solution for the interacting SRN sub-models approach by using equivalent Markov chain models. The Markov chains are reported in the case of $n_r = 1$ to simplify calculations, but the closed form results can also be derived for the cases $n_r > 1$.

The SRN sub-model for the hot pool shown in Figure 2 is equivalent to the Markov chain model shown in Figure 6. In this Markov chain, state i represents the configuration of the hot pool in which i PMs are available. While solving this Markov chain for steady state probability of each state, we can ignore the self-loops [7]. Hence, the Markov chain depicted in Figure 6 is a simple birth-death process where birth rate for state i is $\lambda_h \cdot (1 - p_w) \cdot (1 - p_c) \cdot i$ and death rate for all states is μ. Let

$$\lambda'_h = \lambda_h \cdot (1 - p_w) \cdot (1 - p_c) \quad (6)$$

Let p_{h_i} be the steady state probability to be in state i for the Markov chain of Figure 6, i.e., the probability to have i PMs in the hot pool. Under such assumptions, p_{h_i} is given by:

$$p_{h_i} = \frac{\lambda_h'^{(n_h - i)}}{\mu^{(n_h - i)}} \cdot \frac{(n_h)!}{i!} \cdot p_{h_{n_h}} \text{ with } (0 \leq i \leq n_h - 1) \quad (7)$$

and $p_{h_{n_h}}$ is given by:

$$p_{h_{n_h}} = \frac{1}{\sum_{i=0}^{n_h} \frac{\lambda_h'^{(n_h - i)}}{\mu^{(n_h - i)}} \cdot \frac{(n_h)!}{i!}} \quad (8)$$

From the steady-state state probabilities, we can compute the mean number of PMs in the hot pool ($E[\#P_h]$) that needs to be exchanged with the other sub-models:

$$\begin{aligned} E[\#P_h] &= \sum_{i=0}^{n_h} i \cdot p_{h_i} \quad (9) \\ &= \sum_{i=0}^{n_h} i \cdot \frac{\lambda_h'^{(n_h - i)}}{\mu^{(n_h - i)}} \cdot \frac{(n_h)!}{i!} \cdot \frac{1}{\sum_{j=0}^{n_h} \frac{\lambda_h'^{(n_h - j)}}{\mu^{(n_h - j)}} \cdot \frac{(n_h)!}{j!}} \end{aligned}$$

and the availability of the hot pool:

$$A_{k_h} = \sum_{i=k_h}^{n_h} p_{h_i} \quad (10)$$

The SRN sub-model for the warm pool in Figure 3 is equivalent to the Markov chain model in Fig. 7 in the case

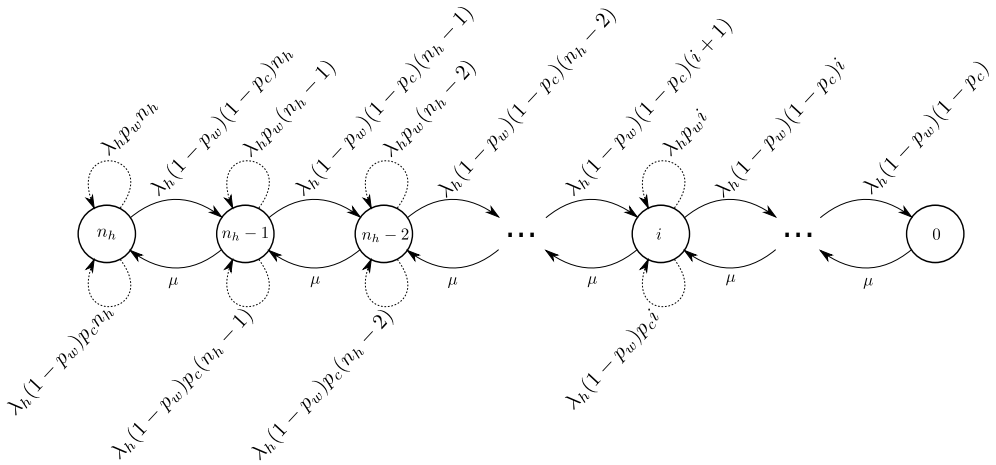

Figure 6. Markov chain equivalent to the hot pool SRN sub-model.

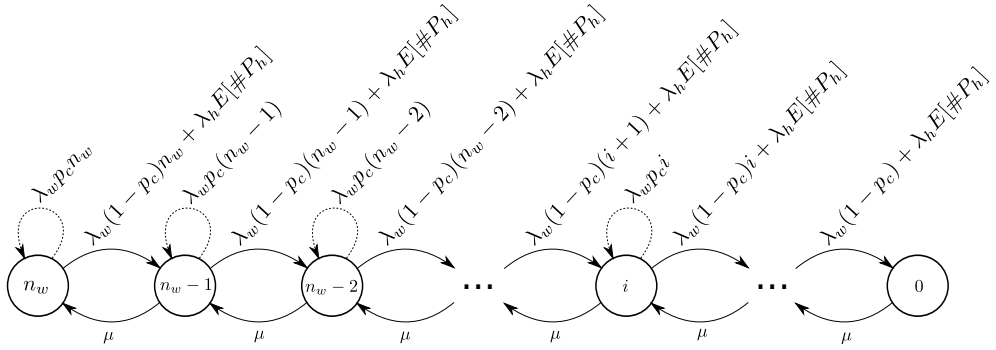

Figure 7. Markov chain equivalent to the warm pool SRN sub-model.

of $n_r = 1$. Also in this case, the Markov chain is a simple birth-death process where birth rate for state i is $\lambda_w \cdot (1 - p_c)i + \lambda_h \cdot E[\#P_h]$ and death rate for all states is μ. Let us define the quantities:

$$\lambda'_w = \lambda_w \cdot (1 - p_c) \tag{11}$$

and

$$\alpha = \lambda_h \cdot E[\#P_h] \tag{12}$$

Let, p_{w_i} be the steady state probability to be in state i for the Markov chain of Figure 7, i.e., the probability to have i PMs in the warm pool. p_{w_i} is given by:

$$p_{w_i} = \prod_{j=i}^{n_w-1} \frac{\lambda'_w \cdot (j+1) + \alpha}{\mu} \cdot p_{w_{n_w}} \text{ with } (0 \leq i \leq n_w - 1) \tag{13}$$

while $p_{w_{n_h}}$ is given by:

$$p_{w_{n_h}} = \frac{1}{1 + \sum_{i=0}^{n_w-1} \prod_{j=i}^{n_w-1} \frac{\lambda'_w \cdot (j+1) + \alpha}{\mu}} \tag{14}$$

From the steady-state probabilities, we can compute the values of p_w and $E[\#P_w]$ that need to be exchanged with the other sub-models. In particular

$$\begin{aligned} p_w &= 1 - p_{w_0} \\ &= 1 - \prod_{j=0}^{n_w-1} \frac{\lambda'_w \cdot (j+1) + \alpha}{\mu} \cdot F_w \end{aligned} \tag{15}$$

where $F_w = \frac{1}{1+\sum_{i=0}^{n_w-1} \prod_{k=i}^{n_w-1} \frac{\lambda'_w(k+1)+\alpha}{\mu}}$ and

$$\begin{aligned} E[\#P_w] &= \sum_{i=0}^{n_w} i \cdot p_{w_i} \\ &= \sum_{i=0}^{n_w} i \cdot \prod_{j=i}^{n_w-1} \frac{\lambda'_w (j+1) + \alpha}{\mu} \cdot G_w \end{aligned} \tag{16}$$

where $G_w = \frac{1}{1+\sum_{l=0}^{n_w-1} \prod_{k=l}^{n_w-1} \frac{\lambda'_w(k+1)+\alpha}{\mu}}$. Finally, it is possible to compute the availability of the warm pool:

$$A_{k_w} = \sum_{i=k_w}^{n_w} p_{w_i} \tag{17}$$

The SRN sub-model for the cold pool in Figure 4 is equivalent to the Markov chain model in Figure 8 in the case of $n_r = 1$. It is a simple birth-death process where birth rate

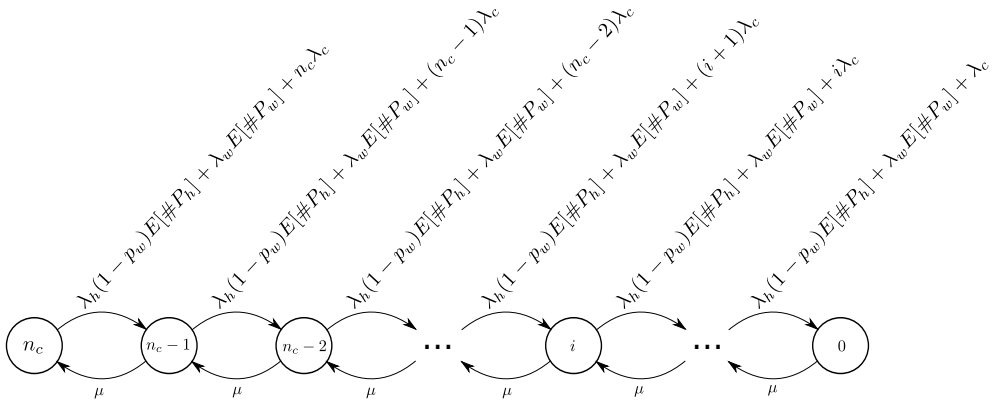

Figure 8. Markov chain equivalent to the cold pool SRN sub-model.

for state i is $\lambda_c i + \lambda_h \cdot (1-p_w) \cdot E[\#P_h] + \lambda_w \cdot E[\#P_w]$ and all death rates are μ. Define β:

$$\beta = \lambda_h \cdot (1-p_w) \cdot E[\#P_h] + \lambda_w \cdot E[\#P_w] \tag{18}$$

Let p_{c_i} be the steady state probability for Markov chain of Figure 8 to be in state i, i.e., the probability to have i PMs in the cold pool. p_{c_i} is given by:

$$p_{c_i} = \prod_{j=i}^{n_c-1} \frac{\lambda_c \cdot (j+1) + \beta}{\mu} \cdot p_{c_{n_c}} \text{ with } (0 \leq i \leq n_c - 1) \tag{19}$$

while $p_{c_{n_c}}$ is given by:

$$p_{c_{n_c}} = \frac{1}{1 + \sum_{i=0}^{n_c-1} \prod_{j=i}^{n_c-1} \frac{\lambda_c(j+1)+\beta}{\mu}} \tag{20}$$

From the steady-state probabilities we can compute p_c and $E[\#P_c]$ that need to be exchanged with the other sub-models, and A_c. In particular:

$$\begin{aligned} p_c &= 1 - p_{c_0} \\ &= 1 - \prod_{j=0}^{n_c-1} \frac{\lambda_c(j+1)+\beta}{\mu} \cdot F_c \end{aligned} \tag{21}$$

$$\tag{22}$$

where $F_c = \frac{1}{1+\sum_{i=0}^{n_c-1} \prod_{k=i}^{n_c-1} \frac{\lambda_c(k+1)+\beta}{\mu}}$ and

$$\begin{aligned} E[\#P_c] &= \sum_{i=0}^{n_c} i \cdot p_{c_i} = \\ &= \sum_{i=0}^{n_c} i \cdot \prod_{j=i}^{n_c-1} \frac{\lambda_c(j+1)+\beta}{\mu} \cdot G_c \end{aligned} \tag{23}$$

$$\tag{24}$$

where $G_c = \frac{1}{1+\sum_{l=0}^{n_c-1} \prod_{k=l}^{n_c-1} \frac{\lambda_c(k+1)+\beta}{\mu}}$. Finally, it is possible to compute the availability of the cold pool:

$$A_{k_c} = \sum_{i=k_c}^{n_c} p_{c_i} \tag{25}$$

VI. Fixed point iteration

To resolve the cyclic dependency among the interacting sub-models, we need to use a fixed point iteration approach. Fixed point iteration variables are reported in the import graph depicted in Figure 5. Our fixed point equation is given by:

$$\mathbf{x} = \mathbf{G}(\mathbf{x}) \tag{26}$$

where $\mathbf{x} = (p_w, p_c, E[\#P_h], E[\#P_w])$.

We show the proof of existence of a solution for the Equation (26). First, we simplify the fixed point equation and express it in a simpler way considering only few variables. We observe that: (i) $E[\#P_h]$ is function of p_w and p_c; (ii) p_w is function of $E[\#P_h]$, and p_c; (iii) $E[\#P_w]$ is function of $E[\#P_h]$, and p_c; (iv) p_c is function of $E[\#P_h]$, p_w and $E[\#P_w]$. From the above relations, it can be shown that all variables can be expressed as functions of p_w and p_c. Hence, the fixed point Equation (26) can be rewritten as:

$$\mathbf{y} = \mathbf{F}(\mathbf{y}) \tag{27}$$

where $\mathbf{y} = (p_w, p_c)$.

Proof of existence of a solution to Equation (27) implies the existence of a solution to Equation (26). We use the Brouwer's fixed point theorem [12]:

Let $\mathbf{F} : \mathcal{C} \subset \mathbb{R}^2 \to \mathbb{R}^2$ be continuous on the compact, convex set $\mathcal{C}$, and suppose that $\mathbf{F}(\mathcal{C}) \subseteq \mathcal{C}$. Then, $\mathbf{F}$ has a fixed point in $\mathcal{C}$.

In our case, given that p_w and p_c are probabilities, we can define $\mathcal{C} = \{\mathbf{y} = (p_w, p_c) : p_w \in [0,1], p_c \in [0,1]\}$. Set $\mathcal{C}$ is compact since both p_w and p_c belong to the closed interval $[0,1]$ and, according to the Heine-Borel theorem, if a subset of the Euclidean space $\mathbb{R}^n$ is closed and bounded then it is also compact.

The set $\mathcal{C}$ is convex if, given two elements $x \in \mathcal{C}$ and $y \in \mathcal{C}$, the element $tx+(1-t)y$, with $t \in [0,1]$ belongs to $\mathcal{C}$. Since, p_w and p_c are both probabilities, convexity of set $\mathcal{C}$ is

Table IV
COMPARISON OF NUMBER OF STATES AND NUMBER OF NON-ZERO ENTRIES.

#PMs in each pool in the beginning	#States in monolithic model	Maximum #states in interacting sub-models	#Non-zero entries in monolithic model	Maximum #non-zero entries in interacting sub-models
5	7056	56	44520	210
10	207636	286	1535490	1320
15	1775616	136	13948160	480
17	3508920	171	27976968	612
19	6468000	210	52189200	760
20	Memory overflow	231	Memory overflow	840
50	-	1326	-	5100
100	-	5151	-	20200
150	-	11476	-	45300
200	-	20301	-	80400

Table VII
COMPARISON OF AVERAGE NUMBER OF PMS IN EACH POOL.

#PMs in each pool in the beginning	**Avg. #PMs in pools for monolithic model**			**Avg. #PMs in pools for interacting sub-models**		
	hot	warm	cold	hot	warm	cold
5	4.99	4.98	4.99	5.00	4.98	4.99
10	10.00	9.96	9.98	10.00	9.96	9.98
15	14.99	14.95	14.97	15.00	14.95	14.97
17	16.99	16.94	16.97	17.00	16.94	16.97
19	18.99	18.93	18.97	19.00	18.93	18.97

Table V
COMPARISON OF SOLUTION TIMES IN SECONDS.

#PMs in each pool in the beginning	Monolithic model	Interacting sub-models
5	0.627	0.406
10	18.670	0.517
15	373.822	0.278
17	1004.494	0.279
19	2459.553	0.280
20	Memory overflow	0.281
50	-	0.296
100	-	0.377
150	-	0.564
200	-	0.948

Table VI
COMPARISON OF DOWNTIME VALUES IN MINUTES PER YEAR, WITH 10 PMS IN EACH POOL OR 30 PMS IN TOTAL IN THE BEGINNING. CLOUD IS AVAILABLE WHEN THERE ARE AT LEAST k "UP" PMS ACROSS ALL POOLS, WHERE $k \leq 30$. MAXIMUM NUMBER OF PMS THAT CAN BE REPAIRED IN PARALLEL IS DENOTED BY n_r.

Value of k	Value of n_r	Downtime (minutes per year)	
		Monolithic model	Interacting sub-models
30	1	23185.793	23178.956
	2	22904.919	22898.454
	3	22903.681	22897.219
29	1	792.475	798.651
	2	499.081	505.258
	3	497.787	503.964
28	1	24.722	25.336
	2	8.412	8.691
	3	7.118	7.396
27	1	0.740	0.778
	2	0.129	0.138
	3	0.081	0.087
26	1	0.022	0.024
	2	0.002	0.002
	3	0.0008	0.0009

trivial in our case. The vector function $\mathbf{F}$ is continuous over $\mathcal{C}$ if its component functions $f_1(p_w, p_c)$ and $f_2(p_w, p_c)$ are continuous over $\mathcal{C}$. The functions f_1 and f_2 are continuous over $\mathcal{C}$ if, for each point $\hat{\mathbf{y}} \in \mathcal{C}$, $lim_{\mathbf{y}\to\hat{\mathbf{y}}} f_i(\mathbf{y}) = f_i(\hat{\mathbf{y}})$ with $i = 1, 2$. From the definition of $\mathcal{C}$, we observe that $\mathcal{C}$ is a singleton set of only element $\mathbf{y} =< p_w, p_c >$. Hence, each term in the limit converges to its (finite) value at $\hat{\mathbf{y}}$. Therefore, the limit $lim_{\mathbf{y}\to\hat{\mathbf{y}}} f_i(\mathbf{y}) = f_i(\hat{\mathbf{y}})$ with $i = 1, 2$ holds in our case and $\mathbf{F}$ is continuous over $\mathcal{C}$. This proves the existence of a solution for the fixed point Equation $\mathbf{y} = \mathbf{F}(\mathbf{y})$. The existence of a fixed point for the Equation $\mathbf{y} = \mathbf{F}(\mathbf{y})$ can be proved also by referring to the Corollary to the Theorem 2 in [10]:

Consider a system modeled using a set of stochastic reward nets. Let some of the transition firing rates/probabilities of each net be functions of one or more parameters computed from the other nets. Then, if such parameters (the iteration variables) are expected reward rates and the CTMCs underlying the SRN models have exactly one closed communication class for all values of the parameters, a fixed point will exist for the correspondent equation.

In our case, all the parameters that are exchanged between the SRN models according to the import graph depicted

in Figure 5 can be computed as expected reward rates. Moreover, from Figure 6, 7 and 8, we can observe that all SRN models have exactly one closed communication class for all the values of the parameters. Clearly, Equation (26) has a fixed point solution. Uniqueness of the fixed point is yet to be proved.

VII. Numerical Results

We used Stochastic Petri Net Package (SPNP) [5] to solve the SRN models. In particular, the interacting SRN sub-models were solved by implementing a fixed point iteration approach using Python scripts. Closed form solution of each sub-model was implemented using C language. Our models were solved for a broad range of parameter space so that they can represent large variety of clouds. However, for space limitation, here we report only interesting results. We assume MTTF of hot PMs to be in the range of $1-6$ months, MTTF of warm PMs to be in the range of $3.5-12$ months and MTTF of cold PMs to be in the range of 7 months - 2 years. MTTR of a PM can vary depending on type of repair process: (i) software based completely automated repair ($1-30$ minutes), (ii) completely manual repair ($1-5$ days) and (iii) combination of manual and automated repair ($1-12$ hours). All models were solved using a desktop PC with Intel Core 2 Duo processor (E8400, 3.0 GHz) and 4 GB memory. In Table IV, we report the state space and storage requirements for both the monolithic model and interacting sub-models. Monolithic model runs into a memory overflow problem when the number of PMs in each pool increases beyond 19. We observe that the state space size of the monolithic model increases quickly and becomes too large to construct the reachability graph even for small number of PMs. However, with interacting sub-models approach, the state space increases at a slower rate as the number of PMs in the system is increased. Table IV also shows a comparison of non-zero entries. These entries are number of non-zero elements in the infinitesimal generator matrix of the underlying continuous time Markov chain. For the same number of PMs, number of non-zero entries in interacting sub-models is $3-4$ orders of magnitude smaller compared to the monolithic model. Observe that, for interacting sub-models, in both cases (i.e., number of states and number of non-zero entries), we report only the maximum value among the three sub-models. Since three sub-models are solved separately, we assumed that for a given execution only states and non-zero entries of only one sub-model are require to be stored in the memory. Reduction in state space and non-zero entries for interacting sub-models also leads to concomitant reduction in solution time needed. A comparison of solution times is shown in Table V. Solution time for monolithic model increases almost exponentially with the increase in model size. Solution time for interacting sub-models remains almost constant with the increase in model size.

In Table VI, we compare the downtime values as obtained from the monolithic model and interacting sub-models. We assume that cloud is available if there are at least k "UP" PMs across all pools. For the example scenario investigated, we vary the value of k, with 10 PMs in each pool and 30 PMs in total. When k is 30, any failure of PM results in unavailability of cloud service. For each value k, we also change the value of n_r which denotes maximum number of PMs that can be repaired in parallel. If n_r is 1, failed PMs are repaired serially, i.e., one after another. MTTFs of hot, warm and cold PMs were assumed to be 1000 hrs, 3500 hrs and 5000 hrs respectively. MTTR was assumed to be 3 hrs. Table VI shows that results obtained from the interacting sub-models are accurate. As expected, downtime values are higher with increasing in values of k. For each k, downtime reduces if we increase value of n_r. This gives rise to interesting optimization problems as discussed in Section IX.

In Table VII, we show the mean number of PMs in each pool. MTTF and MTTR values for this case were assumed to same as in the Table VI. Value of n_r was assumed to be 1 for this example scenario. Results obtained from interacting sub-models are in good agreement with the results obtained from monolithic model. In Table VIII, we show the effect of changing MTTF of PMs on downtime. We assume 10 PMs in each pool (i.e., 30 PMs in total) and maximum number of parallel repairs in each pool is 2. In this example scenario, we further assume that cloud service is available when at least 28 PMs across all pools are "UP". For each value MTTF of hot PM, MTTFs of warm and cold PM were assumed to be 3.5 times and 5 times greater than that of hot PM's MTTF.

Table VIII
Effect of varying MTTF of PMs with 10 PMs in each pool.

MTTF of hot PM (hours)	Downtime (minutes per year)	
	Monolithic model	Interacting sub-models
800	16.313	16.848
1000	8.412	8.691
1200	4.892	5.055
1400	3.091	3.195
1600	2.076	2.146

Results described so far were obtained by solving SRN models using SPNP. Next, using the closed form solutions for the interacting sub-models, we solve large scale models (order of thousands PMs in each pool). Table IX shows that solution time needed for solving large models increases very slowly with the model input size. Clearly, interacting sub-models approach facilitates availability analysis of large sized clouds with a reasonably small solution time.

VIII. Related Research

In [13], Vishwanath *et al.* investigated failure characteristics of servers in large cloud data centers. They tried to

Table IX
SOLUTION TIME REQUIRED FOR AVAILABILITY ANALYSIS OF LARGE SCALE CLOUD USING CLOSED-FORM.

Number of PMs in each pool	Solution time (sec)
500	0.251
1000	0.592
1500	0.911
2000	1.715
3000	2.483
4000	2.651

quantify the relationships between successive failures on same PM by analyzing experimental data and empirically compute reliability. Our work can be complementary to this work since we take into consideration multiple classes of PMs and consider their failure and repair. In [14], Yang *et al.* investigated the failure of workloads on cloud service performance. In [15], Bonvin *et al.* designed a reliable and cost-effective storage system that maintains high availability guarantees despite failures of servers while in [16], Joshi *et al.* discussed the key challenges in achieving high availability in large scale cloud services.

There are limited research efforts which investigated availability in large scale infrastructure. In [17], Tan *et al.* designed and implemented a prediction system to achieve robust hosting for production hosting infrastructure. Our modeling approach can be complementary to such experimental work. In a very recent paper [18], Javadi *et al.* show how statistical models can be useful to predict availability of an Internet distributed system. In [19], Uemura *et al.* used discrete time semi-Markov process to describe the stochastic behavior of a scalable intrusion tolerant system. In [20] Chen *et al.* used a deterministic and stochastic Petri net method to illustrate the performance of producer/consumer based application models in cloud context. In our previous work [21], we showed an SRN modeling approach for resiliency analysis of IaaS cloud.

IX. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we propose a novel fast and scalable approach for availability analysis of IaaS cloud system with multiple class of server pools. The novelty is in modeling the system as coupled interacting Markov chain based sub-models. We have developed closed-form solutions for the sub-models and the dependencies among the sub-models are resolved using fixed-point iteration, for which we prove existence of a solution. This approach reduces the complexity and solution time for analyzing IaaS clouds; e.g., we are able to analyze availability of IaaS Cloud systems with thousands of physical servers in order of seconds. We have also developed a monolithic model which we use for comparison. We show that: (i) availability results using both the sub-models approach and the monolithic model are closely comparable and (ii) the proposed approach can be used to analyze the availability of IaaS cloud for sizes that are beyond the practical limits of the monolithic model.

With the fast scalable approach for modeling availability, we are now able to extend the performability analysis we described in [6] to large size IaaS clouds. In a future publication, we plan to describe results from a coupled pure performance and availability models for systems with thousands of PMs taking into account workload arrival, admission control, queueing, resource provisioning decisions, VM provisioning, and run-time execution in addition to the failures. Table VI shows that downtime can be reduced by increasing the maximum number repairs that can be done in parallel. Since, there is a cost associated with each repair, an optimal number of repair facilities required to minimize the repair cost for acceptable value of downtime can be determined. Combining the performance model with availability model, the economics of failure-repair for a given utilization rate of the PMs can be determined. For different utilization and failure-repair rates, there are different break-even points between loss of revenue and repair costs, which can be determined and analyzed. Our availability models also allow us to perform trade-off analysis of longer MTTF vs faster MTTR on system availability, the effect of having multiple concurrent repair facilities (i.e., higher labor costs) vs. higher availability but expensive components, repairing failed components vs replacing components for the a given service availability and so on. Another interesting type of analysis possible with this work is analyzing trade-off between cost of availability SLAs vs operational costs including repair, replacement, and energy costs. These are important questions cloud architects and designers often face. It is possible to answer these type of questions using tools based on the modeling and analysis techniques we describe in this paper. We plan on developing such tools and will be describing our work in future publications.

REFERENCES

[1] "Amazon EC2," http://aws.amazon.com/ec2.

[2] "IBM Smart Business Cloud," http://www-935.ibm.com/services/us/igs/cloud-development/.

[3] "IBM Cloud Computing," http://www.ibm.com/ibm/cloud/.

[4] K. S. Trivedi and R. Sahner, "SHARPE at the age of twenty two," *ACM Sigmetrics Performance Evaluation Review*, vol. 36, no. 4, pp. 52–57, March 2009.

[5] C. Hirel, B. Tuffin, and K. S. Trivedi, "SPNP: stochastic petri nets. version 6," in *Lecture Notes in Computer Science*, 2000.

[6] R. Ghosh, K. S. Trivedi, V. K. Naik, and D. S. Kim, "Performability analysis for Infrastructure-as-a-Service cloud: An interacting stochastic models approach," in *PRDC*, 2010.

[7] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications, second edition*. Wiley, 2001.

[8] K. S. Trivedi, D. Wang, D. Hunt, A. Rindos, W. Smith, and B. Vashaw, "Availability modeling of sip protocol on ibm websphere," in *PRDC*, 2008.

[9] D. Wang, R. M. Fricks, and K. S. Trivedi, "Dealing with non-exponential distributions in dependability models," in *Symp. on Performance Evaluation - Stories and Perspectives*. G. Kotsis (ed.), Oesterreichchische Computer Gessellschaft, 2003, pp. 273–302.

[10] V. Mainkar and K. S. Trivedi, "Sufficient conditions for existence of a fixed point in stochastic reward net-based iterative models," *IEEE Transactions on Software Engineering*, vol. 22, pp. 640–653, 1996.

[11] L. Tomek and K. Trivedi, "Fixed-point iteration in availability modeling," in *M. Dal Cin, editor, Informatik-fachberichte, Vol. 91: Fehlertolerierende Rechensysteme*. Springer-Verlag, Berlin, 1991.

[12] J. M. Ortega and W. C. Rheinboldt, *Iteartive Solution of Nonlinear Equations in Several Variables*. Academic Press, NY, 1970.

[13] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in *ACM SoCC*, 2010.

[14] B. Yang, F. Tan, Y. S. Dai, and S. Guo, "Performance evaluation of cloud service considering fault recovery," in *CloudCom*, 2009.

[15] N. Bonvin, T. G. Papaioannou, and K. Aberer, "Dynamic cost efficient replications in data clouds," in *Workshop on automated control for datacenters and clouds*, 2009.

[16] K. Joshi *et al.*, "Dependability in the cloud: Challenges and opportunities," in *DSN*, 2009.

[17] Y. Tan, X. Gu, and H. Wang, "Adaptive system anomaly prediction for large-scale hosting infrastructures," in *PODC*, 2010.

[18] B. Javadi, D. Kondo, J. Vincent, and D. Anderson, "Discovering statistical models of availability in large distributed systems: An empirical study of seti@home," *IEEE TPDS (accepted)*, 2010.

[19] T. Uemura, T. Dohi, and N. Kaio, "Availability analysis of a scalable intrusion tolerant architecture with two detection modes," in *CloudCom*, 2009.

[20] H. Chen, C. Zhou, and N. Xiong, "Petri net modeling of the reconfigurable protocol stack for cloud computing based control systems," in *CloudCom*, 2010.

[21] R. Ghosh, F. Longo, V. K. Naik, and K. S. Trivedi, "Quantifying resiliency of iaas cloud," in *SRDS RACOS workshop*, 2010.

[22] M. A. Marsan, G. Balbo, and G. Conte, "A class of generalized stochastic petri nets for the performance evaluation of the multiprocessor systems," *ACM Transactions on Computer Systems*, 1984.

[23] G. Ciardo *et al.*, "Automated generation and analysis of Markov reward models using stochastic reward nets," in *Linear Algebra, Markov Chains and Queuing Models*. Springer, 1993.

APPENDIX A.
STOCHASTIC PETRI NETS AND REWARD NETS

This section presents an overview of Stochastic Petri Nets (SPNs) and Stochastic Reward Nets (SRNs). A Petri net (PN) can be formally defined as a 4-tuple: $PN = (P, T, A, M)$, where P is the finite set of *places* (represented by circles), T is the finite set of *transitions* (represented by bars), A is the set of *arcs* (connecting elements of P and T) and M is the set of markings each of which denotes the number of tokens in the places of the net. The initial marking is denoted by M_0.

In SPN, exponentially distributed firing times can be associated to the net transitions so that the stochastic process underlying a SPN is a continuous time Markov chain (CTMC). In generalized stochastic Petri nets (GSPN) [22], transitions are allowed to be either *timed* (exponentially distributed firing time, drawn as rectangular boxes) or *immediate* (zero firing time, represented by thin black bars). Immediate transitions always have priority over timed transitions and if both timed and immediate transitions are enabled in a marking then timed transitions are treated as if they were not enabled. If several immediate transitions compete for firing, a specified probability mass function is used to break the tie. A marking of a GSPN is called *vanishing* if at least one immediate transition is enabled in it. A marking is called *tangible* otherwise. GSPN also introduces the concept of *inhibitor arc* (represented by a small hollow circle at the end of the arc) which connects a place to a transition. A transition with an inhibitor arc can not fire if the input place of the inhibitor arc contains more tokens than the multiplicity of the arc.

SRNs [23] are extensions of GSPNs. In SRNs, every tangible marking can be associated with a reward rate thus facilitating the computation of a variety of performance measures. Key features of SRNs are:
(1) each transition may have an enabling function (also called a guard) so that a transition is enabled only if its marking-dependent enabling function is true;
(2) marking dependent arc multiplicities are allowed;
(3) marking dependent firing rates are allowed;
(4) transitions can be assigned different priorities;
(5) besides traditional output measures obtained from a GSPN, such as throughput of a transition and mean number of tokens in a place, more complex measures can be computed by using reward functions.