Availability Modeling and Evaluation of a Network Service Deployed via NFV

Mario Di Mauro[1], Maurizio Longo[1] Fabio Postiglione[1], and Marco Tambasco[2]

[1] University of Salerno, Department of Information and Electrical Engineering and Applied Mathematics (DIEM), Fisciano (SA), Italy,
{mdimauro,longo,fpostiglione}@unisa.it
[2] Research Consortium on Telecommunications (CoRiTeL), Fisciano (SA), Italy,
{marco.tambasco}@coritel.it

Abstract. The Network Function Virtualization (NFV) has been conceived as an enabler of novel network infrastructures and services that can be deployed by combining virtualized network elements. In particular, NFV is suited to boost the deployment flexibility of Service Function Chains (SFCs). In this paper, we address an availability evaluation of a chain of network nodes implementing a SFC managed by the Virtualized Infrastructure Manager (VIM), responsible for handling and controlling the system resources. A double-layer model is adopted, where Reliability Block Diagram describes the high-level dependencies among the architecture components, and Stochastic Reward Networks model the probabilistic behavior of each component. In particular, a steady-state availability analysis is carried out to characterize the minimal configuration of the overall system guaranteeing the so-called "five nines" requirement, along with a sensitivity analysis to evaluate the system robustness with respect to variations of some key parameters.

Keywords: Network Function Virtualization, Service Function Chaining, Stochastic Reward Nets, Reliability Block Diagram, Availability analysis.

1 Introduction

The increasing demand for a dynamic and elastic deployment of network and telecommunication services is becoming a critical issue for the operators. As enabler of fifth generation (5G) networks, the Network Function Virtualization (NFV) [1] has been conceived to boost the deployment of new services by adapting the classical virtualization concepts to the network environments. Therefore, classical network elements (e.g. routers, firewalls, load balancers) are deployed in a cloud computing infrastructure by means of virtual machines running on general purpose hardware. The resulting architecture consists in a set of Virtualized Network Functions (VNFs), realizing novel services. A composition of VNFs is often referred to as Service Function Chain (SFC), and is typically governed [2] by the Virtualized Infrastructure Manager (VIM), namely, the key

element of the whole NFV architecture. The result is a system, referred in this work to as Network Service (NS), composed by the VIM that plays a role of the manager furnishing vital services for VNFs (deployment, fault control, network and storage resources tuning etc.), and by the SFC that provides the service itself by means of its VNFs. This kind of deployment raises a critical availability issue, since the fault of an individual block could compromise the whole functionality. Thus, the purpose of this work is to characterize the NS availability aimed at finding the minimal configuration respecting the so-called "five nines" availability requirement (no more than 5 minutes and 26 seconds system downtime per year) as invoked in typical deployments, including cloud-based ones [3]. The availability analysis is performed by considering a hierarchical model that relies on two different formalisms: *i)* the Reliability Block Diagrams (RBDs), a combinatorial model used to characterize the high level dependencies among the elements, and *ii)* the Stochastic Reward Networks (SRNs), a state-space model exploited to typify the probabilistic behavior of the underlying system structure. The rest part of the paper is organized as follows. Section 2 introduces the theme by referring to some ongoing research. Section 3 offers a more detailed view about the NFV reference architecture. An availability model of the whole system along with the adopted formalisms is outlined in Section 4. Section 5 presents some experimental results whereas, conclusions and considerations about future works are drawn in the Section 6.

2 Related Research

The interest of scientific and technical communities in the reliability and availability issues related to the novel network architectures is growing recently. The European Telecommunications Standard Institute (ETSI) has provided some guidelines about the application of reliability concepts to new generation networks, and among which NFV-based ones [4]. Some research has been devoted to characterize the high availability of OpenStack platform, a cloud computing architecture implementing the Infrastructure-as-a-Service (IaaS) paradigm. In [5], a reliable restoration method of virtual resources is provided by exploiting Pacemaker, an OpenStack component able to detect physical server failures, and Libvirt, a set of software libraries in charge of managing virtual resources. An availability model of a IaaS based on a Continuous-Time Markov Chain (CTMC), has proposed in [6]. A similar CTMC-based approach has been used to evaluate the dependability of a virtualized server system in [7], and a cloud-based architecture in [8]. However some serious issues arise when Markov chains models are applied to complex systems, as the state space rapidly reaches intractable sizes. In order to counter this problem, we propose a model relying on the Stochastic Reward Nets, which admits more compact representations of large models, by identifying repetitive structures or model regularities. Inspired by some preliminary works concerning reliability and availability analyses of novel architectures ([9],[10],[11]), in this work the authors offer, as an original contribution, an SRN-based availability model of a Network Service in an NFV environment, that can

help to select the best Network Service configuration that fulfills the "five nines" condition with a minimum number of deployed components.

3 Service Chaining model in the NFV deployment

Classical deployment of network services is typically tightly coupled to the underlying physical topology, resulting in a lack of flexibility to re-arrange the existent components and obtain new functions. On the other hand, NFV allows the creation of an SFC by a chain of VNFs traversed in a predefined order, that in the NFV context, can be interpreted as a VNF Forwarding Graph (VNF-FG). A VNF can be regarded as independent (often geographically separated from other VNFs) element composed by the following three modules:

- *software*: a module implementing a service functionality;
- *hardware*: a module aggregating all physical components (CPU, RAM, Power Supply etc.);
- *hypervisor* (often referred to as Virtual Machine Monitor - VMM): a software layer acting as an interface between hardware and software modules.

On the other hand, the Virtualized Infrastructure Manager is a part of the MANO (MANagement and Orchestration domain) [1] that performs some critical operations as managing a set of resources (storage, networking etc.) to be deployed on demand, mapping the virtual resources on the physical ones and managing the chain of VNFs. In accordance to a common implementation based on OpenStack, we consider the VIM (in a virtualized implementation) as composed by five modules: hardware, hypervisor (similar to those in SFC), and other three elements:

- *database*: devoted to store the critical data of the whole system (inventory of hardware resources, register of available VNFs and others);
- *HAproxy*: the High Availability proxy acts as a load balancer distributing the load among the redundant VIM nodes;
- *functional blocks*: a collection of sub-elements accomplishing several functionalities, such as nova-keystone (providing authorization and authentication mechanisms), nova-scheduler (dispatcher of computational requests), rabbitmq (allowing the communication among internal components).

4 Availability Model of a Network Service

The RBD [12] of a Network Service is reported in Fig. 1, where it has been emphasized that an SFC is a chain of VNF nodes. Being a series configuration, the NS is available (working) when each subsystem (VIM, VNF_1,...,VNF_k) is available (working). On the other hand, the high availability requirements obtained by redundant parallel elements, will be addressed by SRN [13], [14], whose main features are presented in the following subsection.

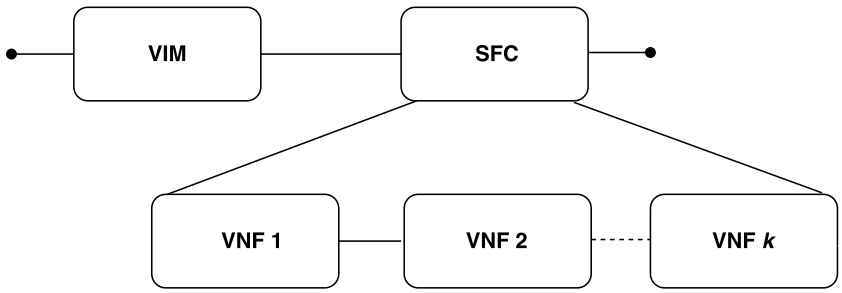

Fig. 1. RBD representation of the Network Service obtained by aggregating VIM and SFC. The latter is in turn composed by a sequence of VNFs.

4.1 SRN formalism

An SRN model is based on a Stochastic Petri Net and is represented by a bipartite directed graph, where a *place* (drawn as a circle) accounts for a specific condition (e.g. a system element up or down) and can contain one or more *tokens*, namely a parameter value associated to a condition. The distribution of tokens (at a time t) is referred to as *marking*, and can be described by a vector $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ where m_h is the number of tokens in the place h. A *transition* (drawn as a rectangle) accounts for a specific event (e.g. a system element that crashes), and lets a token to be transferred from a place to another one, resulting in a marking alteration. In particular, we discriminate between: *timed transitions* (drawn as unfilled rectangles), associated to the events whose times are assumed to be exponentially distributed random variables with a given rate parameter (*firing rate*), and *immediate transitions* (drawn as thin and filled rectangles), whose transition time is equal to zero. When the firing rate of timed transitions depends on tokens distribution in the SRN graph, we use the symbol (#) nearby the correspondent transition and we refer to as *marking dependent transitions*. A transition can be optionally controlled by a *guard function* (indicated by g) that assumes value 1 when transition has to be enabled. Besides, a transition can be inhibited by an *inhibitory arc* (depicted as a line with a blank unfilled circle nearby the correspondent transition). An SRN model is able to capture the dynamics of the system by analyzing the distribution of tokens as time elapses. In an SRN model, the *reward rate* is a non-negative random variable associated with certain conditions of the system, and its value is related to the particular measure (performance, dependability, availability etc.) we are interested to characterize [15]. For example, let $X(t)$ be the availability reward rate where $X(t) \in \{0, 1\}$ and $X(t) = 1$ when the system is working. The instantaneous availability $A(t)$=$P\{X(t) = 1\}$ of a system modeled by an SRN, can be computed by the expectation of $X(t)$ [13], viz.

$$A(t) = E(X(t)) = \sum_{i \in S} r_i p_i(t), \tag{1}$$

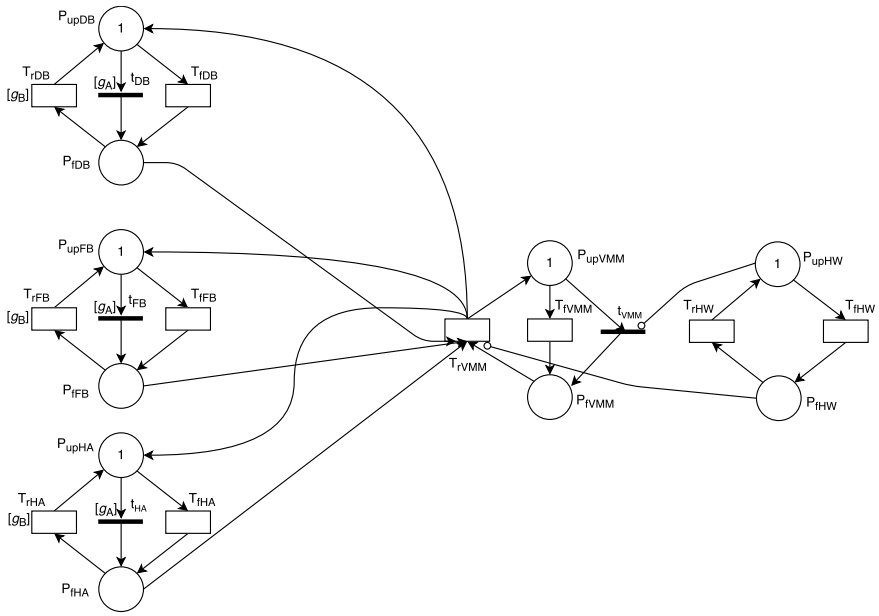

Fig. 2. SRN model of the VIM node. Such model follows the classical OpenStack deployment.

where S represents the set of possible markings, $r_i \in \{0, 1\}$ is the reward value associated to state i, i.e. $r_i = 1$ if system is working, and $p_i(t)$ the probability of system to be in state i.

For the sake of clarity, we analyze separately the SRN models of VIM node and SFC, respectively, and, finally, we combine the results by means of the RBD model in order to characterize the whole NS.

4.2 SRN model of VIM

The SRN model associated to the VIM node is reported in Fig. 2. We want to remark that the VIM subsystem is composed by N redundant VIM nodes. Places P_{upDB} [resp. P_{fDB}], P_{upFB} [P_{fFB}], P_{upHA} [P_{fHA}], P_{upVMM} [P_{fVMM}] and P_{upHW} [P_{fHW}] indicate the conditions where the database, the functional blocks, the HAproxy, the hypervisor and the hardware of the VIM are up [down], respectively. The numbers in the places (tokens) represent the corresponding initial conditions. The transitions T_{fDB} [T_{rDB}], T_{fFB} [T_{rFB}], T_{fHA} [T_{rHA}], T_{fVMM} [T_{rVMM}] and T_{fHW} [T_{rHW}] model the time to failure [repair] of database, functional blocks, HAproxy, hypervisor and hardware, respectively. Let us now briefly discuss the dynamics of such SRN. We start by considering a fully working system (namely all the elements are up and running). As an exemplary case, we focus on the sub-model of the HAproxy (similar considerations hold for database, functional blocks, hypervisor and hardware modules). When

Table 1. Guard Functions defined on the SRN model of VIM

Guard Function	Value
g_A	1 if # $P_{fVMM} = 1$, 0 otherwise
g_B	1 if # $P_{upVMM} = 1$, 0 otherwise

HAproxy fails, the transition T_{fHA} is *fired* and the token removed from P_{upHA} is deposited into P_{fHA}. In the considered model we also introduce some immediate transitions to cope with common cause failures: t_{DB}, t_{FB} and t_{HA} (associated to database, functional blocks and HAproxy, respectively) that are fired when the transition T_{fVMM} is fired, meaning that a hypervisor failure implies the three virtual modules failure as well. Similarly, the immediate transition t_{VMM} accounts for a hypervisor failure as a consequence of a hardware failure. The *inhibitory arc* between P_{upHW} and t_{VMM} compels the hypervisor failure in case of hardware failure. The *inhibitory arc* between P_{fHW} and T_{rVMM} forbids the hypervisor repair in case of hardware failure. Besides, in order to formally describe the dependencies among hypervisor and the three virtual modules (the case of hardware failure is included in the case of hypervisor failure) we introduce two guard functions g_A and g_B described in Table 1. The guard function g_A enables t_{DB}, t_{FB} and t_{HA} when hypervisor fails, namely, when a token is moved from P_{upVMM} to P_{fVMM}. The guard function g_B, instead, inhibits the repair of the three virtual modules in case of hypervisor failure.

4.3 SRN model of a single VNF

In this model, we adopt the realistic assumption that the number of deployed VNFs can vary dynamically with the time, in accordance to a pay-per-use cloud model. In an exemplary scenario, we consider L replicas sharing a (time-varying) load, and M (extra) replicas needed to satisfy a certain availability requirement. Thus, when the load increases or decreases (as per day/night variations), the number of replicas L changes, resulting in a possible variation of the number of M replicas in order to preserve the desired availability.

Figure 3 shows the model of a single VNF. The place P_{upVNF} indicates the VNF working condition, implying that the hardware, software and virtual resources are correctly working. The token value inside P_{upVNF} amounts to $L+M$, namely, the number of initial working VNFs replicas. Let us analyze directly the evolution of such model. First of all we take into account the Scale-Out (S-O) and Scale-In (S-I) operations corresponding to a provisioning phase (deploying replicas) and a de-provisioning phase (un-deploying replicas) respectively. Thus, when an S-O operation is requested, t_{so} is fired and a token is deposited in place P_p, modeling the condition of a replica requested but not working yet, until the token enters P_{upVNF} after T_p is fired. It is worth noting that the *inhibitory arc* from P_p to t_{so} models the impairment of multiple provisioning stages. On the contrary, when an S-I operation is requested, t_{si} is fired and the *inhibitory arc* from P_p to t_{so} preventing S-I operations during provisioning stages. In case of

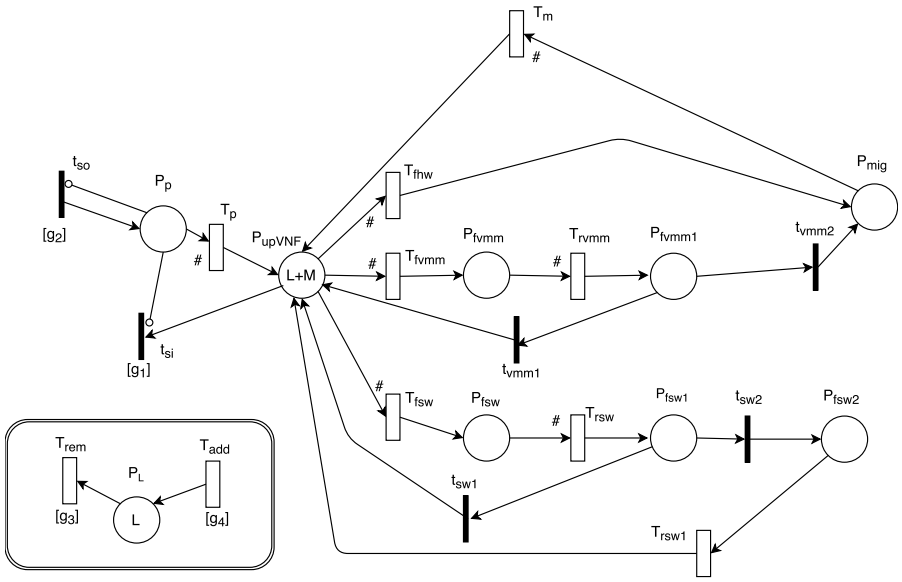

Fig. 3. SRN model of a single Virtualized Network Function.

an hardware failure, T_{fhw} is fired and a token passes from P_{upVNF} to P_{mig}, being the latter a place ruling a migration process (resources are transferred into another hardware platform with no state loss). Once T_m is fired, the migration is completed and the token can come back into P_{upVNF}. In case of a hypervisor failure, T_{fvmm} is fired and a token enters P_{fvmm}. The repair procedure is governed by T_{rvmm} that, when fired, lets the token move into P_{fvmm1}. In this place, two alternatives are admissible: *i)* the VMM repair procedure is successful (e.g. a simple *reboot* solves the problem) with a certain probability p_{vmm}, resulting in firing transition t_{vmm1} and the returning in the working place P_{upVNF}; *ii)* the vmm repair procedure is unsuccessful with probability $(1-p_{vmm})$, the transition t_{vmm2} is fired, the token reaches place P_{mig}, and the migration process described previously is activated. In case of software failure, T_{fsw} is fired and the token is deposited in P_{fsw}; the repair process is ruled by T_{rsw} and the token passes into P_{fsw1}. Similar to the previous case, two options are allowed: *i)* the software repair procedure is successful with a certain probability p_{sw}, thus t_{sw1} is fired and initial condition is gained; *ii)* the repair procedure is vain with probability $(1-p_{sw})$, hence t_{sw2} is fired and place P_{fsw2} (indicating a tough fault condition needing a repairman) is entered; once repair process is terminated, T_{rsw1} is fired and place P_{upVNF} is reached.

Finally, place P_L (see the boxed sub-model) takes into account the condition of L replicas that can be added (T_{add}) or removed (T_{rem}). We note that transitions T_p, T_{fhw}, T_{fvmm}, T_{fsw}, T_{rsw}, T_{rvmm} and T_m, depend only on the number of tokens in their originating places, and their overall firing rates are proportional to those numbers.

Table 2. Guard Functions defined on the SRN model of single VNF

Guard Function	Value
g_1	1 if $N_{tot} > L$ + M , 0 otherwise
g_2	1 if $N_{tot} < L$ + M , 0 otherwise
g_3	1 if $L < L_{max}$, 0 otherwise
g_4	1 if $L > L_{min}$, 0 otherwise

The guard functions present in this model are defined as follows. Said $\#P_k$ the number of tokens in the place k[1], we define N_{tot} the number of VNFs replicas in the SRN at time t as: $N_{tot} = \#P_p + \#P_{upVNF} + \#P_{fvmm} + \#P_{fsw} + \#P_{fsw2} + \#P_{mig}$, where we omit the time dependence [2]. Guard g_1 inhibits S-I operations when N_{tot} undergoes the sum $L + M$. Similarly, g_2 inhibits S-O operations when N_{tot} exceeds the sum $L + M$. Guard g_3 prevents that L exceeds the maximum number of admissible replicas (L_{max}) and g_4 prevents that L could be lower than the minimum number of admissible replicas (L_{min}), as summarized in Table 2.

4.4 Availability analysis and model combination

We now consider an NS, given by the combination of N VIM replicas and the $L + M$ replicas of the three VNFs arranged in a chain. As regards the VIM, let r_i be the reward value assigned to marking i and $p_i(t)$ the probability for SRN in Fig. 2 to be in marking i at time t; according (1) it is possible to express the instantaneous availability $A_{VIM}(t)$ as:

$$A_{VIM}(t) = \sum_{i \in I} r_i p_i(t), \tag{2}$$

where I is the set of *tangible markings* (markings where no immediate transitions are enabled). The instantaneous availability $A_{VIM}(t)$ in fact, is the probability that the VIM is available at time t and, being the markings mutually exclusive, can be expressed as the sum of the probabilities of all markings that at time t result in a working condition for the VIM subsystem.

The reward value r_i assumes value 1 for the markings identifying the VIM working conditions, namely all the operating conditions where at least two Database instances (as typical in many active-active configurations), one Functional Block and one HAproxy are active. For all the remaining states, r_i is set to 0. Thus, the reward value can be written as:

[1] In Petri Net jargon, there is a little abuse of (#) symbol that indicates either number of tokens, and marking-dependent transitions.

[2] Places P_{fvmm1} and P_{sw1} do not contribute to N_{tot} because the time spent in such places is zero.

$$r_i = \begin{cases} 1 & \text{if } \left(\sum_{k=1}^{N} \#P_{upDB}(k) \geq 2\right) \wedge \\ & \quad \left(\sum_{k=1}^{N} \#P_{upFB}(k) \geq 1\right) \wedge \\ & \quad \left(\sum_{k=1}^{N} \#P_{upHA}(k) \geq 1\right) \\ 0 & \text{otherwise,} \end{cases}$$

where k goes from 1 to the number of replicas N, and $\#P_{upDB}(k)$, $\#P_{upFB}(k)$ and $\#P_{upHA}(k)$, indicate the number of tokens in the "up" places of virtual modules, for the $k-th$ parallel element. The hardware and hypervisor "up" conditions do not appear, being included in expression when at least 1 virtual module is active. The VIM steady-state availability is given by (2) as $t \to \infty$,

$$A_{VIM} = \lim_{t \to +\infty} A_{VIM}(t) = \sum_{i \in I} r_i p_i, \tag{3}$$

where p_i is the steady-state probability of state i, i.e. $p_i = \lim_{t \to +\infty} p_i(t)$. A similar reasoning holds for the case of VNF. Let s_j be the reward value assigned to marking j and $q_j(t)$ the probability for SRN in Fig. 3 to be in marking j at time t. The expected reward value at time t for the VNF model corresponds to:

$$A_{VNF}(t) = \sum_{j \in J} s_j q_j(t), \tag{4}$$

where J identifies the set of *tangible markings* of the VNF model. In this case, the reward value s_j associated to the tangible marking j follows:

$$s_j = \begin{cases} 1 & \text{if } (\#P_{upVNF} \geq \#P_L) \\ 0 & \text{otherwise.} \end{cases}$$

The VNF working condition ($s_j = 1$) occurs when the number of total replicas (represented by the tokens in P_{upVNF}) is no less than the number of regular replicas (represented by the tokens in P_L). Thus, steady-state availability for a VNF is obtained by (4) as $t \to \infty$,

$$A_{VNF} = \lim_{t \to +\infty} A_{VNF}(t) = \sum_{j \in J} s_j q_j, \tag{5}$$

where q_j is the steady-state probability given by $q_j = \lim_{t \to +\infty} q_j(t)$.

Now, we are able to evaluate the steady-state availability of the whole Network Service, composed by the series of VIM and the 3 VNFs according to the RBD representation in Fig. 1.

The resulting Network Service availability A_{NS} can be expressed as the product of the availabilities associated with the VIM and SFC subsystems, namely

$$A_{NS} = A_{VIM} \prod_{m=1}^{3} A_{VNF}^{(m)}, \tag{6}$$

where A_{VIM} is derived by (3) and A_{VNF} is computed by (5), where m denotes the VNF m.

5 Experimental Results

An exemplary application of the proposed approach is now provided, where system parameters assume specific values suggested by the technical literature (e.g. [16]). The values of parameters associated to VIM and to VNF are reported in Tables 3 and 4, respectively. In order to respect the notation used in Figs. 2 and 3, we use uppercase subscripts for the VIM parameters and lower case subscripts for the VNF parameters. The present availability analysis has been carried out with the help of SHARPE [17], a tool that allows to analyze SRN availability models. According to elasticity concepts typical of cloud environments, we target two exemplary different operating conditions, namely c_1 and c_2 associated to a dynamically variable load managed by the system, that results in a different number of deployed replicas. We suppose that in condition c_1 (low load) VNFs share the load among two replicas ($L_{c1} = 2$), while in condition c_2 (high load) they share the load among three replicas ($L_{c2} = 3 \geq L_{c1}$).

Accordingly, we characterize the system in terms of the number of extra replicas M guaranteeing the "five nines" availability requirement, namely M_{c1} and $M_{c2} \geq M_{c1}$ in the two conditions of VNFs. We investigate also the influence of the number of VIM replicas N. We distinguish 6 settings $S_1, ..., S_6$ representing different configurations, as reported in table 5 with their corresponding availability value A_{NS}. Figure 4 reports the main results where, for visual comfort, we show the steady-state unavailability of the Network Service $(1 - A_{NS})$ in said exemplary settings. By comparing S_1, S_2, S_3 with S_4, S_5, S_6, it is evident that an increase from $N = 3$ to $N = 4$ improves the NS availability. For $N = 3$, instead of considering greater values of N for $M_{c1} = M_{c2} = 1$ (setting S_1), it is possible to achieve the "five nines" requirement by deploying $M_{c2} = 2$ replicas

Table 3. Input parameters for the SRN representing the VIM

Parameter	Description	Value [a)]
$1/\lambda_{VM}$	mean time to (DB,FB,HA) failure	3000 hours
$1/\lambda_{VMM}$	mean time to hypervisor failure	5000 hours
$1/\lambda_{HW}$	mean time to hardware failure	60000 hours
$1/\mu_{VM}$	mean time to (DB,FB,HA) repair	1 hour
$1/\mu_{VMM}$	mean time to hypervisor repair	2 hours
$1/\mu_{HW}$	mean time to hardware repair	8 hours

[a)] We assume the same failure/repair rate values (being deployed on similar VMs) for database, functional blocks and HAproxy.

Table 4. Input parameters for the SRN representing the VNF

Parameter	Description	Value
$1/\lambda_{hw}$	mean time to hardware failure	60000 hours
$1/\lambda_{sw}$	mean time to software failure	3000 hours
$1/\lambda_{vmm}$	mean time to hypervisor failure	5000 hours
$1/\mu_{sw}$	mean time to easy software repair	7 minutes
$1/\mu_{sw1}$	mean time to tough software repair	2 hours
$1/\mu_{vmm}$	mean time to hypervisor repair	10 minutes
$1/\alpha_p$	mean time to provisioning	20 minutes
$1/\alpha_m$	mean time to migration	20 minutes
$1/\alpha_s$	mean time to scaling (S-I/S-O) procedures	12 hours
p_{sw}	probability of successful software repair	0.98
p_{vmm}	probability of successful hypervisor repair	0.99

Table 5. Availability Results of the whole Network Service

Setting	Redundancy Level	A_{NS}
S_1	$L_{c1}=2$, $L_{c2}=3$, $M_{c1}=1$, $M_{c2}=1$, $N=3$	0.99998444
S_2	$L_{c1}=2$, $L_{c2}=3$, $M_{c1}=1$, $M_{c2}=2$, $N=3$	0.99999080
S_3	$L_{c1}=2$, $L_{c2}=3$, $M_{c1}=2$, $M_{c2}=2$, $N=3$	0.99999084
S_4	$L_{c1}=2$, $L_{c2}=3$, $M_{c1}=1$, $M_{c2}=1$, $N=4$	0.99998686
S_5	$L_{c1}=2$, $L_{c2}=3$, $M_{c1}=1$, $M_{c2}=2$, $N=4$	0.99999323
S_6	$L_{c1}=2$, $L_{c2}=3$, $M_{c1}=2$, $M_{c2}=2$, $N=4$	0.99999326

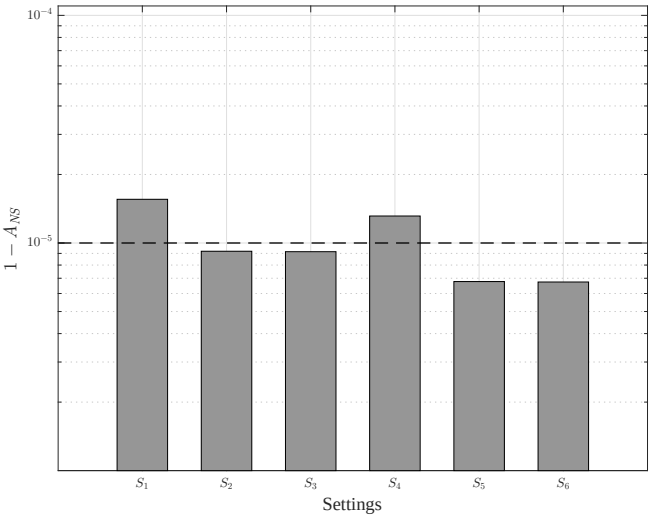

Fig. 4. Unavailability $1 - A_{NS}$ of the Network Service for 6 settings $S_1, ..., S_6$ representing various replicas arrangements. The horizontal dashed line represents the "five nines" requirement ($1 - A_{NS} = 10^{-5}$).

(setting S_2). By incrementing M_{c1} (setting S_3), a negligible increment of NS availability is obtained. Similar considerations can be provided for the $N = 4$ case (setting S_4, S_5, S_6). Given a "five nines" availability constraint, S_2 is the minimal cost setting in terms of deployed replicas.

5.1 Sensitivity Analysis

Generally, a sensitivity analysis concerns the robustness of the system with respect to deviations of some system parameters from their nominal values. We analyze the influence of two crucial parameters on the availability of the minimal cost setting S_2, such as:

- λ_{SW}, influencing the transition T_{fsw} of the VNF model reported in Fig. 3 governing the software failures;
- $\lambda_{VMM}(= \lambda_{vmm})$, influencing both the transition T_{fVMM} of the VIM model (Fig. 2), and the transition T_{fvmm} of the the VNF model (Fig. 3) (since we assume the same hypervisor for both).

Figure 5 shows that the reciprocal of failure rate of the software part can be relaxed from 3000 hours (nominal value) to 2500 hours by still guaranteeing the "five nines" requirement indicated with a horizontal dashed line. Similarly, Fig. 6 shows that the working value of 5000 hours for $1/\lambda_{VMM}$ can be reduced to 4300 hours with no side effects on the desired availability condition.

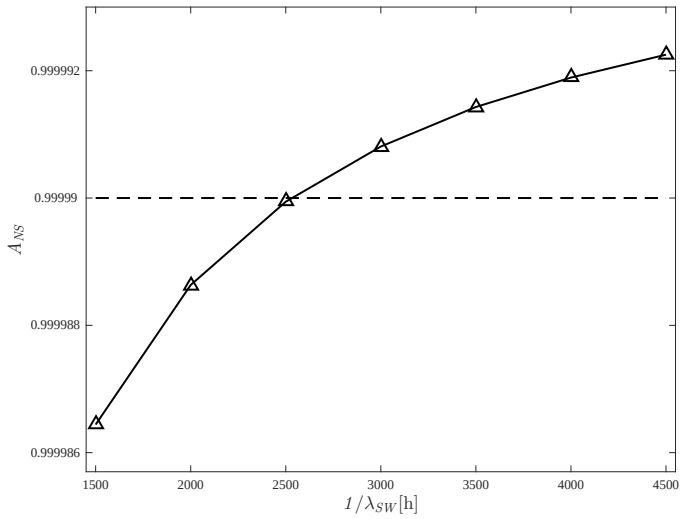

Fig. 5. Influence of the software failure rate on the overall system.

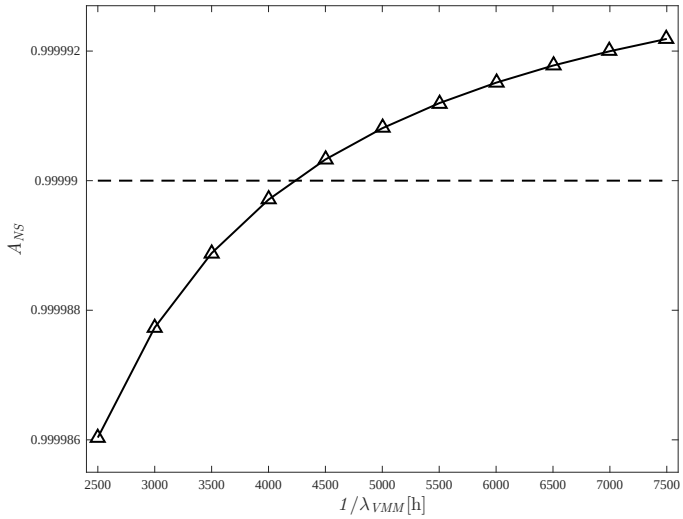

Fig. 6. Influence of the hypervisor failure rate on the overall system (same hypervisor for both VIM and VNFs).

6 Concluding Remarks

The NFV represents a cutting-edge paradigm in the telecommunication and networking industry, that allows to offer advanced communication services by deploying software-based network appliances, namely VNFs, replacing traditional physical equipments. The VNFs, and their interconnections, namely the SFC, are managed by the VIM, a critical element in charge of supervising the whole NFV environment and, at an higher level, the whole Virtualized Infrastructure. In this work we proposed an availability evaluation of a Network Service (NS), resulting from the aggregation of VIM subsystem and an SFC. Such a composition has been modeled by hierarchical approach where a Reliability Block Diagram represents the high-level interconnections between VIM and SFC, and Stochastic Reward Nets modeling each subsystem characterized by failure and repair events. A steady-state availability analysis, has been performed to evaluate different NS configurations. Besides, the robustness of the whole system with regard to variations of some critical parameters has been evaluated. The proposed availability and sensitivity analyses provide useful indications to network and telco operators about, the deployment of the number of component replicas satisfying a certain availability requirement, and perception about the influence of characteristic parameters on the overall architecture. Future works will be devoted at considering more sophisticated interconnections among components, including the virtual and physical links.

References

1. ETSI. Network Functions Virtualisation: An introduction, benefits, enablers, challenges and call for action. Technical report, 2012.
2. A. M. Medhat, T. Taleb, A. Elmangoush, G. A. Carella, S. Covaci, and T. Magedanz. Service function chaining in next generation networks: State of the art and research challenges. *IEEE Commun. Mag.*, 55(2):216–223, 2017.
3. T. Taleb, A. Ksentini, and B. Sericola. On service resilience in cloud-native 5g mobile systems. *IEEE J. Sel. Areas Commun.*, 34(3):483–496, 2016.
4. ETSI. Network Functions Virtualisation (NFV) reliability; report on models and features for end-to-end reliability. Technical report, 2016.
5. Y. Yamato, Y. Nishizawa, S. Nagao, and K. Sato. Fast and reliable restoration method of virtual resources on OpenStack. *IEEE Trans. Cloud Comput. (In Press)*, 2015.
6. H. Khazaei, J. Mii, V. B. Mii, and N. B. Mohammadi. Availability analysis of cloud computing centers. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 1957–1962, 2012.
7. X. Zhang, C. Lin, and X. Kong. Model-driven dependability analysis of virtualization systems. In *2009 Eighth IEEE/ACIS International Conference on Computer and Information Science*, pages 199–204, 2009.
8. J. Dantas, R. Matos, J. Araujo, and P. Maciel. An availability model for eucalyptus platform: An analysis of warm-standy replication mechanism. In *2012 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1664–1669, 2012.
9. M. Di Mauro, F. Postiglione, and M. Longo. Reliability analysis of the controller architecture in Software Defined Networks. In L. Podofillini, B. Sudret, E. Stojadinovic, B. Zio, and W. Kröger, editors, *Safety and Reliability of Complex Engineered Systems*, pages 1503–1510. Taylor & Francis Group, 2015.
10. M. Di Mauro, F. Postiglione, M. Longo, R. Restaino, and M. Tambasco. Availability evaluation of the virtualized infrastructure manager in Network Function Virtualization environments. In L. Walls, M. Revie, and T. Bedford, editors, *Risk, Reliability and Safety: Innovating Theory and Practice*, pages 2591–2596. Taylor & Francis Group, 2017.
11. M. Di Mauro, M. Longo, and F. Postiglione. Performability evaluation of software defined networking infrastructures. In *ValueTools 2016 - 10th EAI International Conference on Performance Evaluation Methodologies and Tools*, pages 1–8, 2016.
12. W. Kuo and Z. Ming. *Optimal Reliability Modeling: Principles and Applications*. Wiley, 2002.
13. J.K. Muppala, G. Ciardo, and K.S. Trivedi. Stochastic Reward Nets for reliability prediction. In *Communications in Reliability, Maintainability and Serviceability*, pages 9–20, 1994.
14. D. M. Nicol, W. H. Sanders, and K. S. Trivedi. Model-based evaluation: from dependability to security. *IEEE Trans. Depend. Sec. Comput.*, 1(1):48–65, 2004.
15. J.K. Muppala, M. Malhotra, and K.S. Trivedi. *Markov Dependability Models of Complex Systems: Analysis Techniques*. Springer Berlin Heidelberg, 1996.
16. R. De S. Matos, P. Maciel, F. Machida, K. Dong Seong, and K. Trivedi. Sensitivity analysis of server virtualized system availability. *IEEE Trans. Rel.*, 61(4):994–1006, 2012.
17. Robin A. Sahner and K.S. Trivedi. Reliability modeling using SHARPE. *IEEE Trans. Rel.*, 36(2):186–193, 1987.