Petri Net-Based Model for 5G and Beyond Networks Resilience Evaluation

Rui Li, Bertr Decocq, Anne Barros, Yiping Fang, Zhiguo Zeng

To cite this version:

Rui Li, Bertr Decocq, Anne Barros, Yiping Fang, Zhiguo Zeng. Petri Net-Based Model for 5G and Beyond Networks Resilience Evaluation. 2022 25th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), Mar 2022, Paris, France. pp.131-135, 10.1109/ICIN53892.2022.9758134 . hal-03648310

HAL Id: hal-03648310
https://hal.science/hal-03648310v1

Submitted on 7 Feb 2023

Petri Net-Based Model for 5G and Beyond Networks Resilience Evaluation

Rui Li*†, Bertrand Decocq*, Anne Barros†, Yiping Fang†,Zhiguo Zeng†
**Orange Innovation*
Châtillon, France
{rui.li, bertrand.decocq}@orange.com
†*Laboratoire Génie Industriel*
CentraleSupélec, Université Paris-Saclay
Gif-sur-Yvette, France
{rui.li, anne.barros, yiping.fang, zhiguo.zeng}@centralesupelec.fr

***Abstract*—The promise of telecommunication networks to deliver more demanding and complex applications requires them to become more flexible and efficient. To achieve better performance, telecommunication networks adopt technologies such as NFV (Network Function Virtualization). However, this evolution also brings more potential risks to the telecommunication network. Reliability and resilience are becoming critical for service delivery in the networks. To answer to service requirements of high level availability and reliability, a model with a global view of infrastructure, virtual network elements, and network layer structure is required. Toward this end, this paper presents a Petri Net method to model 5G and beyond telecommunication networks. We introduce an extended Petri Net to model physical infrastructure, virtual infrastructure, network services, their behaviors, and dependencies. We present a simulation result on network availability estimation. This result shows the potential of the Petri Net-based model to be applied to a complex telecommunication system resilience assessment.**

***Index Terms*—Petri Net, 5G networks, B5G, resilience, availability, modeling, simulation**

I. INTRODUCTION

Telecommunication networks are becoming indispensable for modern production and living. Facing the diverse and high requirements from a broad vertical industry, 5G and beyond networks are expected to be both efficient and reliable for service delivery. Keeping such systems at good performance during their whole life cycle is essential for service providers and operators. Although 5G has been under development for the last ten years, its resilience has not been studied enough.

5G is more service-oriented than 4G by offering a transition from a ”horizontal” service delivery model toward a ”vertical” one [1]. To better meet the different requirements and high demand, Network Function Virtualization (NFV) should be introduced for both RAN (Radio Access Network) and CN (Core Network) to better adjust the network configuration to the requirements. Thus, the resilience of the telecommunication networks is no longer an issue only for infrastructure but also for virtual elements and service delivery [2].

This study builds a Petri Net-based model to describe 5G and beyond networks. This model could be applied to communication service availability [3] (the ability to allow correct operation of the application) analysis, communication service reliability (the measure of continuous correct service delivery) estimation, and the resilience [4] (the ability to provide and maintain an acceptable level of service in the face of various faults and challenges) evaluation of the system.

While still at an early stage, this paper introduces a case study regarding network virtualization characteristics for testing Network Function Virtualization (NFV) self-healing in the dysfunctional mode and analyzing the network availability.

The paper is structured as follows. First, we present related work concerning telecommunication resilience analysis and modeling in section II. Then in section III, we focus on the telecommunication network model and, in particular explaining how an extended Petri Net models the virtualization characteristics. A case study on self-healing and the results are given in section IV. Section V concludes the work with some remarks and outlines the future works.

II. RELATED WORK

Recently, some research regarding the 5G and beyond telecommunication network resilience has been carried out. These studies mainly focus on network resilience optimization, only a few on resilience assessment. In optimization, while solving linear programming problems is still the mainstream of the research as found in [5]–[7], other methods such as Shortest Paths [8], Divide and Conquer [9] are also applied. The complexity of solving such a problem grows with the number of constraints. However, numerous new constraints on resource allocation and network management will be required if the virtualization layer is considered. In assessment, methods such as Reliability block diagram [10], Markov chain [11], [12] are addressed. These methods fail to model a complex telecommunication network by considering infrastructure, virtualization, network layers, and their dependencies. A model that can capture all network elements and their relationship for resilience assessment and optimization is still missing.

As a critical technique for the 5G and beyond, Network Virtualization has been particularly studied in some works. An availability model and analysis of a virtualized system based on Virtual Machines(VMs) are introduced in [13]. The authors in [14] present a performance modeling approach that goes into the microservice level to estimate the effect of resource configuration on the Quality of Service (QoS). Although VMs are already widely used in virtualization, containerization, as a novel and lightweight virtualization method, is believed to be a promising solution for 5G and beyond. However, rarely is container-based virtualization modeled. Because it makes the telecommunication network modeling even more complex [2].

Some recent works draw attention to Petri Net-based model to study telecommunication networks thanks to its convenience in modeling discrete event systems. In [15], a Petri Net model is proposed for Service Function Chain (SFC) reliability assessment. However, the model does not take into consideration the risks of infrastructure failures. A Queuing Petri Net model is applied in [16] to evaluate the QoS of a video streaming service. The dysfunctional state of the system is not yet considered in their study. In [17], the authors apply a Petri Net model to describe the probabilistic behaviors of network service. In [18], an extension of Petri Net is applied only to model the NFV MANO framework to analyze its availability. Finally, the authors in [19], introduce a Petri Net-based performance model for containerized applications deployed by Kubernetes. However, these works are limited and do not propose a comprehensive perspective by considering NFV characteristics, infrastructure behaviors, and QoS in functional and dysfunctional mode.

Inspired by the related work, this paper proposes an extended Petri Net approach to fill the gaps in related work. This extended Petri Net is given to model SFC, containerization-based NFV elements, and infrastructure layer of the network. It is also capable to estimate the performance and the resilience.

III. PETRI NET-BASED NETWORK MODEL

A. Timed Stochastic Colored Petri Net

1) Petri Net mathematical representation: Petri Net is also known as Place / Transition net. It is a widely used technique tracking systems' states, dynamics, and constraints. As well defined in [20], the Petri Net is a 5-tuple $\mathcal{N} = \langle P, T, F, W, M_0 \rangle$, where P is a finite set of places often representing the different states of a system. Places are graphically presented in circles. T is a finite set of transitions representing the state-changing process. Transitions are graphically presented in rectangles or squares. F is a finite set of arcs with $F \subseteq (P \times T) \cup (T \times P)$. W is a multi-set of arcs $(P \times T) \cup (T \times P) \to \mathbb{N}$ assigning the weight to inputs and outputs of a transition. M is the marking of the Petri Net graph and $M_0 = P \to \{m_1, m_2, ..., m_{|P|}\}$, therefore, assigning the initial marking of the graph. Tokens of the graph describe the dynamic and concurrent activities of systems. The marking in Petri Net records the token number of each place.

2) Extensions of Petri Net: The classical Petri Net is not directly applicable to telecommunication systems. Some state-changing processes in such a system could be stochastic, time-dependent, and require additional information. Some extensions of Petri Net can help modeling a complex network better.

One of the most important extensions of Petri Net is Stochastic Petri Net [21]. It includes a new set $R = \{r_1, r_2, ..., r_{|T|}\}$, representing the firing rate of each transition. This extension could be applied to describe a failure process in the telecommunication network.

In order to describe a time-dependent process, for instance, the packet transmission, Timed Petri Nets [22] are introduced. A new set $D : T \to \mathbb{Q}_0^+$ associates each transition with a specific non-negative number to represent the time factor.

Colored Petri Net attaches a value to a token. It indeed distinguishes different kinds of tokens that a place holds. This extension adds the following items [23], [24]:

1) Σ is a finite set of non-empty types, called color sets.
2) C is a function $P \to \Sigma$ defining the type of tokens allowed in a place.
3) G: $T \to \mathbb{B}$ associate the transition with a precondition g (Boolean expression). The transition will be fired only when g returns true value.
4) E is an arc expression function defined from F into expressions such that $\forall a \in F : C(E(a)) = C(p)$.
5) I is an initialization function mapping place $p \in P$ with an expression such that $I(p)$ is associated to $C(p)$.

Combining the extensions as aforementioned, we use a Timed Stochastic Colored Petri Net (TSCPN) to describe the 5G system. Such a TSCPN is a multi-tuple: $TSCPN = \langle \Sigma, P, T, F, W, m_0, C, G, E, I, R, D \rangle$

TSCPN is then applied to describe the different parts of the telecommunication system.

B. Composition of the network system

We divide the 5G and beyond networks into three layers.

The first layer is the service layer. In this layer, the network service is delivered by steering packets between a set of functions called the service function chain. We consider a network service where all its network functions engaged are virtualized. The service delivery is presented by a series of Virtual Network Functions(VNFs) connected via virtual links.

The second layer is the NFV elements layer. A virtual link in this layer is based on a physical transport network, and a VNF is a virtual functional building block hosted on a physical server. Unlike physical elements, virtual elements may have an unfixed size and an unfixed number of replicas.

The third layer is the infrastructure layer. Physical machines and physical links belong to this layer.

C. Basic hypothesis

1) Service function chain: We consider an End-to-End service with an ordered SFC. In this system, user equipment sends service request packets to the SFC in the network. We assume that every packet conveys a same size of data, and its SFC always follows the same order of VNFs.

An SFC is a series of VNFs connected by links. The transport network is considered as a perfectly reliable system. We only consider a fixed time delay spent on the transmission link between a user equipment and VNF, and between different VNFs.

2) VNF and Virtualization: A VNF is, in fact, an application that consists of several microservices. An example of 3 VNFs and their microservices are shown in Fig. 1. Each microservice is considered as a sub-function of VNF. We assume that if a request packet needs multiple microservices, they should be pursued in a given order. A packet cannot consult two different microservices at the same time.

Fig. 1. VNFs and their microservices.

There are many ways of virtualization. In this paper, we choose to model the deployment of these microservices in containers. Kubernetes is used as the system for automating deployment and managing containerized applications.

Pods are the smallest deployable units in Kubernetes. A pod is one or a cluster of containers with shared storage and network resources. We assume that only one container is deployed on a pod. For each container, it corresponds to a microservice the VNF supplier predefines. Pods are running on Kubernetes nodes. All these nodes are physical machines.

3) Infrastructure and resources: The infrastructures used to deliver an end-to-end function are physical links and physical servers. Each physical machine has a certain amount of CPU, storage, and network resources. Pods can only be hosted on the server with enough resources.

4) Orchestration and management: Kubernetes is an enabler for the orchestration and management of containerized applications. To evaluate the system resilience under failure, we consider self-healing operation. Kubernetes can regularly detect the healthiness of the pods or nodes. In case of failure, they will be terminated, and new ones will be created. Other operations such as auto-scaling, in which Kubernetes detects particular indicators and change the deployment manners accordingly, will be studied in future work.

D. Telecommunication network modeling

1) 5G Service Function Chain: This is the top layer Petri Net which represents the process of an SFC containing m VNFs as presented in Fig. 2. This pipeline style Petri Net consists of a set P of $2(m+1)$ places and a set T of $2m+1$ transitions. It signifies the progress of packet processing.

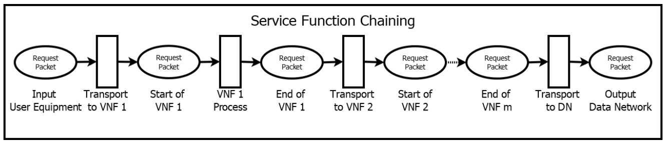

Fig. 2. Petri Net of SFC example.

At this level, a token refers to a packet that conveys a request message that needs to traverse the SFC. The color function C allows only 'packet' type tokens to stay at the places, $C = \{\text{'Packet'}\}$. These 'packet' tokens can also convey some values, including packet serial number, latency requirement, packet starting time, etc.

$P_{\text{sfc}} = \{p_{\text{UE}}, p_{\text{Start of VNF1}}, p_{\text{End of VNF1}}, ..., p_{\text{End of VNFm}}, p_{\text{DN}}\}$ represents different steps of the processing procedure. 'Packet' tokens at the place p_{DN} signify that the packets are successfully delivered. 'Packet' token information such as latency can be further investigated to verify if service is delivered correctly.

The transition set $T_{\text{sfc}} = \{T_{\text{tran}} \cup T_{\text{treat}}\}$ contains the transport and the packet treatment in SFC. The transitions $T_{\text{tran}} = \{t_{\text{Transport to VNF 1}}, t_{\text{Transport to VNF 2}}, ..., t_{\text{Transport to DN}}\}$ stand for the packets being transmitted from the previous place to the next one. A duration function D is attached to these transitions and returns a time delay for each packet transmission according to the distance and the transmission technology. The treatment transitions $T_{\text{treat}} = \{t_{\text{VNF 1 process}}, t_{\text{VNF 2 process}}, ..., t_{\text{VNF m process}}\}$ are expanded into sets of sub-networks explained in the following sections.

2) Virtual Network Functions: We expand the VNF i process transition $t_{\text{VNF i process}}$ as depicted in Fig. 3. This transition takes a token from the place $p_{\text{Start of VNF i}}$, and after processing, it returns the token to the place $p_{\text{End of VNF i}}$. Inside a VNF i, there are n microservices embedded in containers. The token color set is $\Sigma = \{\text{'Packet', 'Packet list'}\}$. The latter refers to a list of packets to be treated. When a packet gets into this i-th VNF, an immediate transition $t_{\text{service selection}}$ routes them to the microservice k it looks for with the help of guard function G. Then this packet is inserted to the 'Packet list' token at the place $p_{\text{MS queue k}}$. Microservice k transition $t_{\text{MS k-transition}}$ is enabled as long as there is at least one microservice pod with enough capacity left to treat this packet. After the treatment, the packet arrives at $p_{\text{End of MS in VNF i}}$. Two assertion transitions associated with guard functions will check if other microservices should be consulted before leaving current VNF.

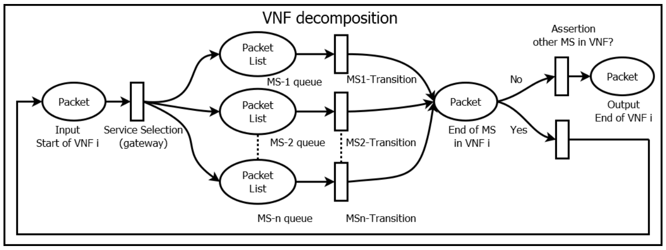

Fig. 3. Petri Net of VNF decomposition.

3) Microservices: Microservices level Petri Net explains the transition $t_{\text{MS k-transition}}$ in detail. The microservices applications are in the form of containers and they are embedded into pods. A new token color type, 'Pod', is added. These tokens are stored at $p_{\text{available pods}}$ and $p_{\text{failed pods}}$.

The microservice is modeled with two transitions, as shown in Fig. 4. The transition $t_{\text{MS bounding}}$ couples the first packet of the packet list token with an available pod. A Boolean guard function B associated with this transition checks if the pod is eligible to provide service for the packet. To complete the task, the packet will borrow a certain computation resources from the pod. These resources are seen as homogeneous by assumption. A function D assigns the processing time of the microservice to this transition. An output packet token will be sent to $p_{\text{bounded packets}}$ after this duration. The transition $t_{\text{MS process}}$ verifies the completion of the service. If a pod fails during the packet treatment, the packet needs to redo the same microservice. Otherwise, the borrowed resource will be released, and the packet will try the next microservice.

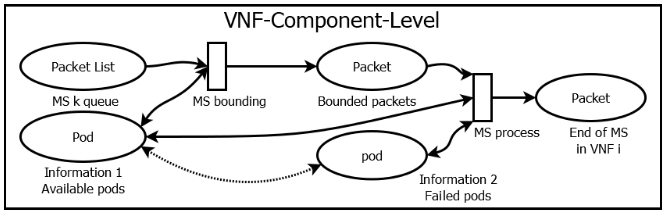

Fig. 4. Petri Net of Mirco-service treatment.

4) Failure and Self-healing: Numerous failures could happen in a telecommunication system. In this study, we mainly consider pod failure, one of the most common failures that affect the a vitrulized system's performance.

Due to space, only pod software failure is explained, as shown in Fig. 5. A stochastic transition $t_{\text{Stochastic pod failing process}}$ connects the state change of a pod token. We assume that all pods in our system are identical, and thus, they have the same mean time to failure (MTTF). We also assume that a pod failure is in accordance with the exponential distribution $X \sim \text{Exp}(\lambda)$, and with a constant rate of $\lambda = \text{MTTF}^{-1}$.

Kubernetes launches a liveness probe once in a while to detect the healthiness of pods. This time interval is called periodsecond of the probe. If a pod is unhealthy, Kubernetes starts the self-healing by terminating the pod and creating new one. The transition $t_{\text{Pods termination}}$ consumes the failed token after a graceful termination time. Transition $t_{\text{Pods creation}}$ will create a pod containing the same microservice on an available node with enough resources. We introduce place $p_{\text{Available node}}$ containing a new token color, 'Node', to represent the nodes.

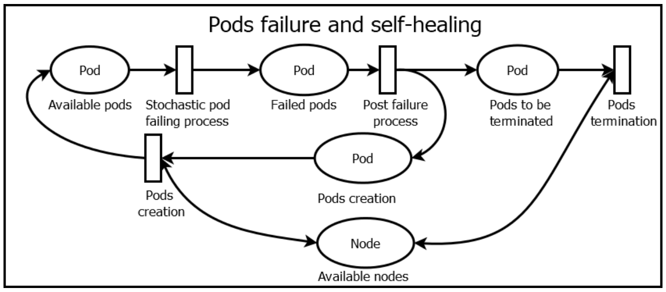

Fig. 5. Pod Self-healing process

IV. NETWORKS AVAILABILITY ESTIMATION

The first step of our work is to apply the model to estimate the system resilience by looking at the virtualization and infrastructure layer without mapping them to telecommunication services. By doing so, the availability of the network to provide services to the packets is estimated. Two major failures, the physical failure on nodes and the software failure on pods, are identified as the main risks to the system. When a failure occurs, the Kubernetes Master will do self-healing to ensure availability. We consider a system with one VNF that consists of two microservices. We assume that these microservices container pods are deployed on the same Data Center. For load balancing reasons, each microservice initially has three identical pod replicas. Other parameters are given in Table. I.

TABLE I
VNF PARAMETERS

Parameter	Value
Pod failure rate	MTTF = 1258 hours [25]
Pod termination time	30 seconds (fixed value)
Node failure rate	MTTF = 8760 hours
Node repair rate	MTTR = 0.5 hours
Average time for pod instantiating	5 seconds
Average time for node creation	1 second
Node capacity	3 pods per node
Data Center Capacity	4 servers
Self-healing probe periodsecond	0 (immediate), 2, 5 and 10 seconds

A microservice is considered available at time t, if the token quantity of such microservice at $p_{\text{available pods}}$ is greater than the desired replica quantity. The uptime of a microservice is the duration of time that a microservice is available. Then the average availability of a microservice i can be calculated as:

$$A_i = \frac{\text{microservice } i \text{ uptime}}{\text{total simulation time}}$$

In the first situation, we assume the self-healing detection is immediate, i.e., a failure on pod or node can be detected with no delay. We simulate the microservice behavior over 50 years. The average value of microservices' uptime over 20000 simulations is taken as the final result. We assume that the two microservices are from the same VNF supplier and are managed by the same Kubernetes Master. The results in Fig. 6 show that if the desired replica quantity is three, then the availability of a single microservice is 99.9996712% (5 nines). If the desired replica quantity is one (one pod is enough, but three initial pods bring high redundancy), then the availability of this microservice can achieve up to 9 nines. The overall availability for the VNF (at least three available replicas for both microservice 1 and 2) is 99.9993523%[1].

In the second situation, the effect of self-healing probe frequency t_p on system availability is studied. The result is shown in Fig. 7. We compare the overall VNF availability for t_p varying from 0 to 10 seconds. The longer the probe periodsecond, the lower the overall availability. The availability drops from 99.9993523% (5 nines) to 99.9987198%[2] (4 nines) by changing immediate detection to 10 seconds. Thus, the telecommunication network can consume less energy while satisfying the availability requirement by wisely optimizing the periodsecond if allowed, according to this result.

---

[1] 95% confidence interval [99.9993520304%, 99.9993526071%]

[2] 95% confidence interval [99.9987192945%, 99.9987202301%]

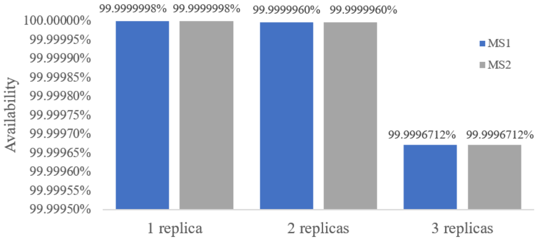

Fig. 6. Microservice availability with immediate detection.

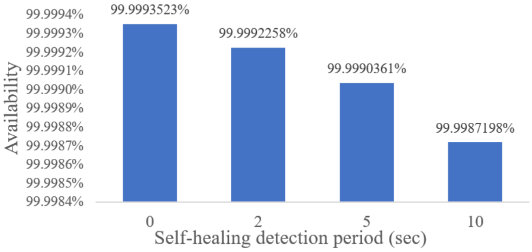

Fig. 7. Overall availability under immediate, 2, 5, and 10 seconds detection.

After 20000 simulation iterations, the results converge well. It took from half to two hours (depending on detection interval) to run these 20000 simulations in CPN tools on a personal computer equipped with Windows 10, 2.10 GHz CPU, and 8GB memory. Indeed, the computation time is proportional to the number of pods and inversely proportional to the periodsecond.

V. CONCLUSION

This paper presents a Petri Net-based model to analyze the performance and resilience of 5G and beyond networks. This model divides a telecommunication system into multiple layers and proves its ability to describe new features of 5G and beyond. The results of the Monte-Carlo simulation on VNF self-healing show the prospects of this model on telecommunication network availability analysis.

The results remain optimistic since other risks such as network failure or maintenance are not fully considered. In addition, more precise parameters need to be collected from our experts and suppliers. For the next step, the auto-scaling case study will be carried out to complete the service-level reliability and resilience analysis, and to see how networks adapt to different packet traffic. We also intend to expand the case study from one single VNF to an SFC and apply the model to simulate a real use case from the verticals.

REFERENCES

[1] A. Banchs, D. M. Gutierrez-Estevez, M. Fuentes, M. Boldi and S. Provvedi, “A 5G mobile network architecture to support vertical industries,” in IEEE Communications Magazine, vol. 57, no. 12, pp. 38–44, December 2019.

[2] R. Li, B. Decocq, A. Barros, Y. Fang, and Z. Zeng, “Complexity in 5G Network Applications and use cases,” In 31st European Safety and Reliability Conference, pp. 3054–3061, 2021.

[3] B. Sayrac, “5G Common Terminology,” 5G Smart Manufacturing, 2020, [Online] Available: https://5gsmart.eu/wp-content/uploads/5G-SMART-common-terminology.pdf.

[4] ResiliNets Wiki, [Online] Available: https://resilinets.org.

[5] T. Hößler, L. Scheuvens, N. Franchi, M. Simsek and G. P. Fettweis, “Applying reliability theory for future wireless communication networks,” 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), pp. 1-7, 2017.

[6] K. Sayad, B. Lemoine, A. Barros, Y. Fang, and Z. Zeng, “Dynamic orchestration of communication resources deployment for resilient coordination in critical infrastructures network,” In 31st European Safety and Reliability Conference, pp. 2055–2062, 2021.

[7] D. Harutyunyan, N. Shahriar, R. Boutaba and R. Riggio, “Latency-Aware Service Function Chain Placement in 5G Mobile Networks,” 2019 IEEE Conference on Network Softwarization, pp. 133–141, 2019.

[8] L. Qu, C. Assi, K. Shaban, and M.J. Khabbaz, “A reliability-aware network service chain provisioning with delay guarantees in NFV-enabled enterprise datacenter networks,” IEEE Transactions on Network and Service Management, vol. 14, no. 3, pp. 554–568, 2017.

[9] Y. Carlinet, N. Perrot and A. Alves-Tzitas, “Minimum-cost virtual network function resilience,” in INOC, pp. 37–42, 2019.

[10] M. Di Mauro, G. Galatro, M. Longo, F. Postiglione and M. Tambasco, “Comparative performability assessment of SFCs: the case of containerized IP multimedia subsystem,” in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 258–272, March 2021.

[11] M. Mosahebfard, J. Vardakas and C. Verikoukis, “Modelling the admission ratio in NFV-based converged optical-wireless 5G networks,” in IEEE Transactions on Vehicular Technology, vol. 70, no. 11, pp. 12024-12038, 2021.

[12] H. Farooq, M. S. Parwez and A. Imran, “Continuous time Markov chain based reliability analysis for future cellular networks”, Proc. IEEE Glob. Commun. Conf. (GLOBECOM), pp. 1–6, 2015.

[13] D.S. Kim, F. Machida, and K.S. Trivedi, “Availability modeling and analysis of a virtualized system,” in 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing, pp. 365–371, 2009.

[14] M.G. Khan, J. Taheri, M.A. Khoshkhologhi, A. Kassler, C. Cartwright, M. Darula, et al., “A performance modelling approach for SLA-aware resource recommendation in cloud native network functions,” in : 6th IEEE Conference on Network Softwarization, pp. 292–300, 2020.

[15] L. Rui, X. Chen, Z. Gao, W. Li, X. Qiu and L. Meng, “Petri Net-based reliability assessment and migration optimization strategy of SFC,” in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 167–181, March 2021.

[16] S. Schneider, A. Sharma, H. Karl and H. Wehrheim, “Specifying and analyzing virtual network services using queuing Petri Nets,” 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 116–124, 2019.

[17] M. Di Mauro, M. Longo, F.Postiglione and M. Tambasco, “Availability modeling and evaluation of a network service deployed via NFV,” in International Tyrrhenian Workshop on Digital Communication, Springer, Cham, pp. 31–44, 2017.

[18] B. Tola, Y. Jiang and B. E. Helvik, “On the resilience of the NFV-MANO: an availability model of a cloud-native architecture,” 2020 16th International Conference on the Design of Reliable Communication Networks (DRCN) , pp. 1–7, 2020.

[19] V. Medel, O. Rana, J. A. Bañares, and U. Arronategui, “Modelling performance and resource management in kubernetes,” in Proceedings of the 9th International Conference on Utility and Cloud Computing, Association for Computing Machinery, New York, pp. 257–26, 2016.

[20] T. Murata, “Petri Nets: Properties, analysis and applications,” Proceedings of the IEEE, vol. 77, no. 4, pp. 541-580, April 1989.

[21] M. Marsan, G. Conte, G. Balbo, “A class of generalized stochastic Petri Nets for the performance evaluation of multiprocessor systems,” ACM Trans. Comput. Syst, vol, 2. pp. 93-122, 1984.

[22] L. POPOVA-ZEUGMANN, “Time petri nets,” Time and Petri Nets, Springer, Berlin, Heidelberg, pp. 31–137, 2013.

[23] K. Jensen, “9,” Volume 1, Basic Concepts. Springer, Berlin, 1992.

[24] D. Liu, J. Wang, S. Chan, J. Sun, L. Zhang, “Modeling workflow processes with colored Petri Nets,” Computers in Industry, vol. 49. pp. 267–281, 2002.

[25] S. Sebastio, R. Ghosh, A. Gupta and T. Mukherjee, “ContAv: a Tool to assess availability of container-based systems,” 2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA), pp. 25–32, 2018.