Optimizing the deployment of Virtual Network Functions in 5G networks with Model Based Testing*

Francisco Luque-Schempp[1], Laura Panizo[1], Maria-del-Mar Gallardo[1], and Pedro Merino-Gomez[1]

[1] Universidad de Malaga, Andalucia Tech, Spain
[2] {fls, laurapanizo, gallardo, pedro}@lcc.uma.es

Abstract. The new 5G mobile communication networks have an unprecedented dependency on software compared with previous generations. The software, running with virtualization techniques, will be the main enabler to dynamically share the network resources in a number slices. Each slice offers a full end-to-end network for one specific service and a set of users, and is implemented with the deployment and configuration of a chain of virtual network functions (VNFs) combined with other network resources (e.g. capacity in the base stations or priority of traffic). A new entity, called NFV MANO (network functions virtualization management and network orchestration) is expected to decide on the placement and resources assigned to the VNFs to keep alive all the slices. A major challenge in this new paradigm is how the MANO will dynamically assign the limited resources to the VNFs in all the slices in such a way that all the users always perceive the desired quality for the service. Usually, these resources are CPU, memory and storage space in the computational nodes. This paper addresses the use of formal methods to implement the elastic and dynamic placement and resource configuration of the VNFs. In particular, we discuss the use of model-based testing and model checking as underlying techniques for machine learning to create part of the intelligence that requires the MANO in 5G networks (sometimes call the book rule). The paper introduces the overall methodology centered in a non-deterministic parameterized model of the MANO that can be refined thanks to testing campaigns with real execution of slices in a 5G lab environment.

Keywords: Network Function Virtualization · 5G networks · Model based testing · dynamic configuration of slices.

1 Introduction

Nowadays, the economy and digital society are more and more dependant on mobile communication networks. Mobile networks are evolving to new paradigms

* The 5GENESIS project was funded by the European Unions Horizon 2020 research and innovation programme under grant agreement No. 815178.

in which softwarization is one of the key aspects. In particular, one of the most important features of 5G networks is the shift of core networks from proprietary to commercial off-the-shell hardware in order to flexibly and dynamically adapt networks to the changing market and user needs. To achieve these objectives, the 5G networks introduce the concept of *network slice* that is similar to a private network tailored to run a service with specific *Key Performance Indicators* (KPI). For instance, a critical service such as telesurgery imposes hard constraints on packet latency (under 5 ms) in order to control a surgeon robot over a cellular network. Other services, such as high-resolution content delivery, require high uplink speed. In 5G networks, each kind of service will run on a different network slice that satisfies their performance and quality of service requirements, but all these network slices will be deployed over the same underlying infrastructure.

The implementation of network slicing in 5G relies on enabling technologies from cloud computing domain, such as *Virtualized Network Functions* (VNFs). A VNF is a software implementation of a network equipment, such as a router, a firewall, a load balancer, or even the components that conform the mobile core network. One of the main characteristics of VNFs is that they are designed to run on the cloud. In the future 5G networks, the operators' cloud will be physically distributed among different locations, called *Points of Presence*, in order to dynamically deploy VNFs where they are needed to support the changing service demands. This challenging task is carried out by the Management and Orchestration (MANO) entity included in the 5G reference architecture (see Figure 1).

In this context, this paper faces the challenge of automatically generate useful information to help the orchestrator decide about the deployment, configuration and re-configuration of VNFs. The generation of this knowledge for the orchestrator basically uses the description of the VNFs and the service level agreements for each service (SLA) to predict the suitable deployment that satisfies the SLA. This problem has been previously addressed with different estimation tools [7, 8, 5, 10]. However, these previous approaches are focused on the use of of computational resources and do not consider the impact of a realistic 5G network in the final quality of service perceived by the users.

In this paper, we aim to develop a novel learning method to specialize the orchestrator considering the whole end-to-end network, including the users and the communication part. We propose to combine formal methods to generate and test the orchestration decisions and and a realistic end-to-end 5G network to run the VNFs. In order to achieve this aim, the described problem is transformed into a runtime analysis of extra-functional properties (relative to time and resources utilization) evaluated over event sequences that reflect executions of network service. Formal techniques such as *model-based testing*, *model checking* and *runtime verification* will be combined to carry out the runtime analysis mentioned above.

The paper is organized as follows. Section 2 introduces the main concepts and problems of 5G network slicing. Section 3 presents the overall proposal to

generate the MANO rule's book using model based testing and runtime verification techniques. Sections 4 focuss on specific aspects of the proposal: how to construct the model of the system and how to automatically generate test cases. Section 5 present related work. Finally, Section 6 summarizes the conclusions and future work.

2 Background: slicing in 5G networks

2.1 Architecture of the network

Figure 1 represents a 5G network. A 5G network is built upon the three main domains that present in the previous 4G technology: radio access network (composed of the user equipment and the gNb access nodes), the transport network (a logical connection thanks to switches, aggregation points and communication links) and the core network. The main differences with respect to 4G networks are the technologies in each domain and the deployment of the services as part of the network (AF component in the figure). In the radio part, the new standard is called 5G NR (5G new radio) and offers features like lower latency and higher capacity than 4G networks. In the core network, the functionality is implemented with a number of software modules and interfaces following the standards to create a 5G core. It is expected that these 5G core VNFs will be deployed as VNFs in a central cloud.

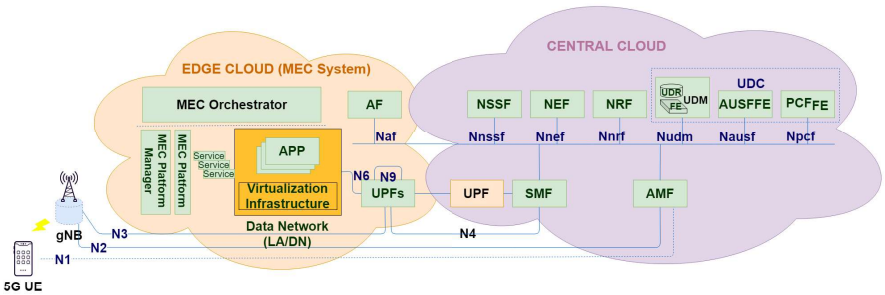

Fig. 1. 5G Architecture

From the service provider perspective, the main novelty is the integration of the software to implement the service as part of the network thanks to the deployment of their own VNFs jointly with the 5G core VNFs and other network-oriented VNFs (e.g. caches and firewalls) in the same cloud. When a service requires low latency is possible to deploy its VNFs following the *Multi-access Edge Computing* technology (MEC). Using the MEC implies that some 5G core components, like the User Plane Function (UPF), are also deployed in the MEC.

2.2 Slices for verticals

Network slicing allows *Mobile Network Operators* (MNOs) to manage multiple virtual networks using a common shared physical infrastructure. These virtual networks make possible a virtual partition of the RAN (Radio Access Network), the core network and the switching and aggregation network. Roughly speaking, Figure 1 represents one slice, and a second slice could be created by assigning part of the RAN resources and adding more VNFs in the Edge cloud and the Central cloud to provide functionalities of 5G core and services. Each virtual network is created to serve a specific service with specific requirements, that usually fit three profiles: *enhanced Mobile Broadband* (eMBB), *Ultra Reliable Low Latency Communications* (URLLC) and *massive Machine Type Communications* (mMTC). Figure 2 shows some usage scenarios where virtual networks mentioned above, known as *Network Slices*, help to provide a service that depends on use case requirements.

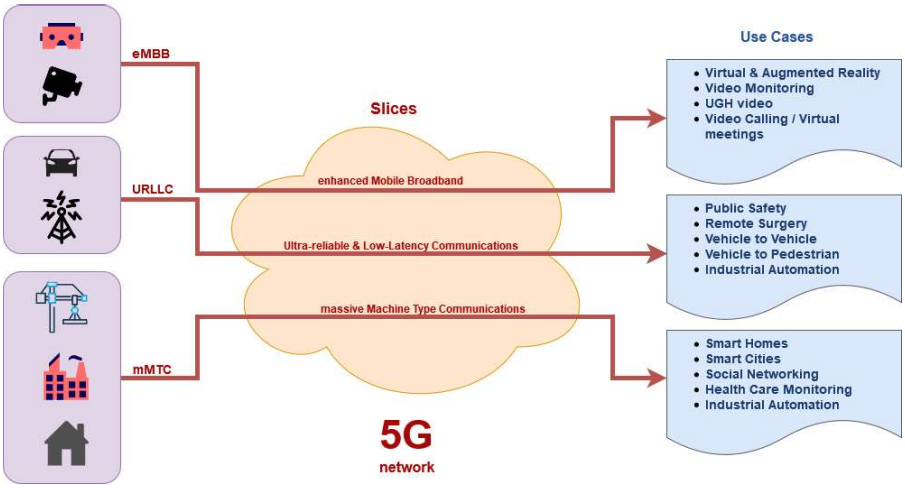

Fig. 2. 5G usage scenarios

2.3 NFV to support slices

As mentioned in the ETSI GS NFV 003 V1.4.1 [1], Network Functions Virtualization (NFV) is the "principle of separating network functions from the hardware they run on by using virtual hardware abstraction".

In 5G network the concept of NFV is especially important since it can be a significant transformation for this network, reducing cost or increasing flexibility, although the most important change that NFV introduces is the possibility to

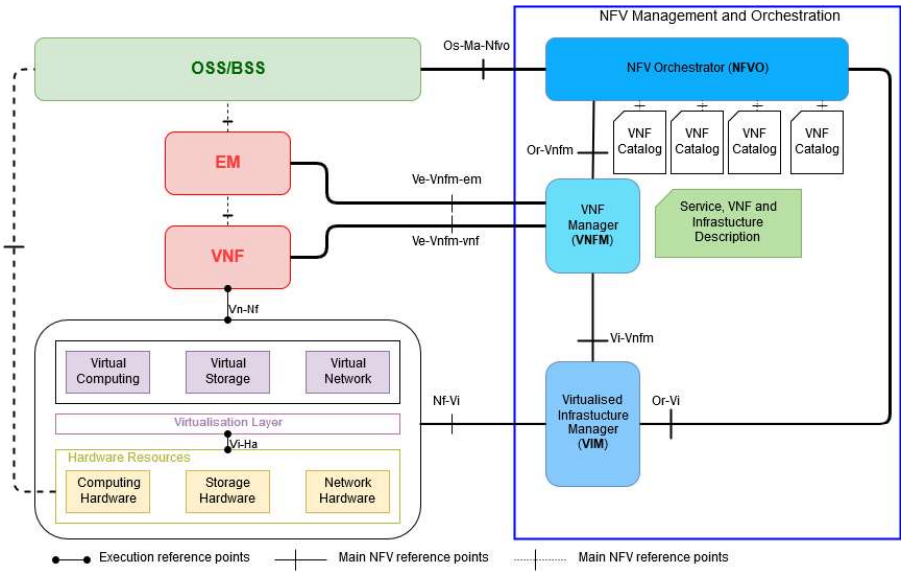

Fig. 3. ETSI MANO

provide different kinds of services and requirements on a common shared physical network through network slicing (see Section 2.2).

As mentioned in Section 1, the deployment and reconfiguration of the VNFs in a cloud environment is an open challenge. This task is carried out by the MANO, Figure 3 shows an overview of the MANO architecture proposed by the ETSI. Left part is composed of *Operations Support System* (OSS) / *Business Support System* (BSS), *Element Manager* (EM) + VNF and by last, *NFV Infrastructure* (NFVI) composed of virtual/hardware computing, storage and network (the hardware supporting the Edge and Central clouds in Figure 3). OSS/BSS are components that they allow to monitor, control and manage different kind of network services. EM provides network management of the virtualized and physical network elements. The VNF is an implementation of a network function that can be deployed on NFVI. Right part is composed of *Management and Orchestration* (MANO) layer, differentiating between *NFV Orchestration* (NFVO) + *VNF Manager* (VNFM) and *Virtualized Infrastructure Manager* (VIM). NFVO is responsible for orchestration and management of NFVI, software resources and realizing network services on NFVI. VNFM is responsible for control, management and monitorization of VNF life cycle, it also controls EM. The VIM is the Virtualized Infrastructure Manager, and in most real deployments is the well-known OpenStack software.

2.4 A running example

Figure 4 shows an example that illustrate how a service is deployed in a network slice. We assume that the MNO offers a simple slice to deploy a video on demand service for mobile users. The slice includes network components, such as an instance of a 5G core, and some service-oriented VNFs, for example a video server and the cache function. The orchestrator (the MANO) addresses the following 4 phase to properly configure, deploy and terminate the service.

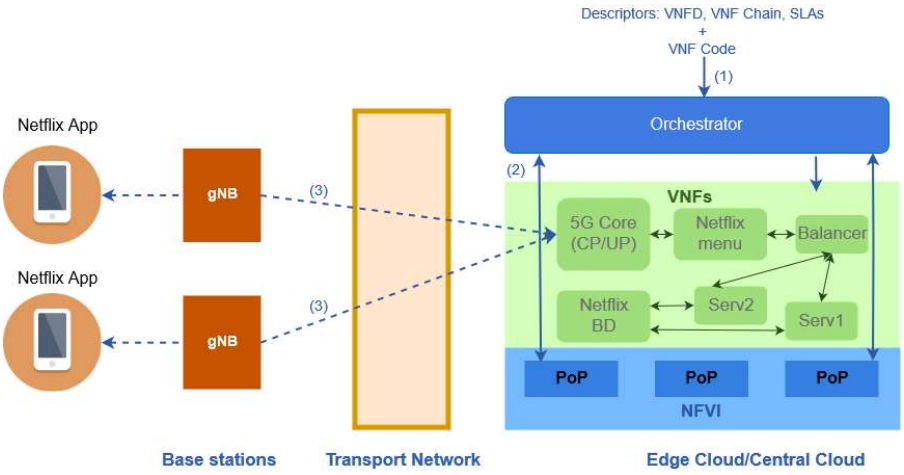

Fig. 4. Example of slice for video on demand

Phase 1: ***Network service descriptor processing.*** The MANO "orchestrator" processes the information necessary to deploy a network slice oriented to a specific service (Netflix in this case). This information includes the executable code of VNF (virtualized like a container, virtual machine, etc.) and the descriptors of the different VNFs (VNFDs), which are defined by ETSI [2] as follows: "A VNFD is a deployment template which describes a VNF in terms of deployment and operational behaviour requirements. It also contains connectivity, interface and virtualized resource requirements". Additionally, the VNF descriptor can include information about the quality of service expected by user (requirements), for example the value of parameters such as delay, bandwidth, number of simultaneous users, etc. and auto-scaling properties. In cloud computing terminology, these requirements are called *Service Level Agreement* (SLA) [1]. The fulfillment of SLAs are translated into the fulfillment of expected quality by users (e.g. max delay between video frames and max response delay). In 5G network terminology, these quality indicators are usually referred to as *Key Performance Indicators* (KPIs), and less frequency *Quality Performance Indicators* (QPIs). The MANO also processes the *Network Service Descriptor* (NSD) [3] that consists of information used by the *NFV Orchestrator* to instantiate a *Network Service* formed by one or more VNFs. By last, *Network Slice Template* (NST)

represents logical network function(s), resource linked to the services, and the most important, it represents network capabilities that are required by service used which in turn is closely related to *Service Level Agreement* (SLA) previously mentioned. In practice, all the descriptors are specified using description languages like TOSCA or YAML, which are widely used in cloud computing field. All the information is processed to generate internal models of objects described in VNFD, NSD and NST that the orchestrator will manage in the next phase. Listing 1.1 and 1.2 show the code of the VNFD of a *Video on Demand* (VoD) VNF, and the definition of the service type.

Listing 1.1. Excerpt of a VNFD

```
vnfd-catalog:
    vnfd:
    -   connection-point:
        -   name: eth0
            type: VPORT
        ...
        description: ...
        mgmt-interface: ...
        name: slice_VoD_vnfd
        vdu:
        -   count: 1
            id: vdu1
            image: UbuntuVoD
            interface:
            -   external-connection-point-ref: eth0
                ...
                virtual-interface:
                ...
            monitoring-param:
                    - id: metric_vdu1_cpu
                      nfvi-metric: cpu_utilization
            name: slice_VoD_vnfd-VM
            vm-flavor:
                memory-mb: 2048
                storage-gb: 100
                vcpu-count: 2
        monitoring-param:
        -   id: metric_vim_vnf1_cpu
            name: metric_vim_vnf1_cpu
            aggregation-type: AVERAGE
            vdu-monitoring-param:
                vdu-ref: vdu1
                vdu-monitoring-param-ref: metric_vdu1_cpu
        ...
```

Listing 1.2. Definition of the slice service type

```
-   SNSSAI-identifier:
        slice-service-type: eMBB/URLLC/mMTC
```

Phase 2: ***Network service deployment.*** The MANO performs the deployment, inter-connection and configuration of VNFs chain interacting with the points of presence of the operator computational infrastructure, commonly known as *Network Functions Virtualization Infrastructure* (NFVI). The deployment consist of locating each VNF at suitable point of presence. The configuration implies the allocation of resources (CPU, RAM, disk) and the interconnection with other VNFs and/or physical elements. At this point, the MANO must apply some optimization algorithm that identify how many resources allocate to achieve the performance and quality of service agreed in the SLA.

Phase 3: ***Network service execution and re-configuration.*** At this point, the VNFs implementing network and service functionality are running, and the final users (users of the service) start using the service uninterruptedly it for days, weeks or months. Some examples of services are the distribution of high-resolution video (Netflix is a popular service in this domain, in Figure 4 it is supposedly installed in network operator), a private communications service for security forces, an augmented reality service to offer sightseeing activities, a remote control service for critical infrastructures, etc. In this phase, the number of users and their location of users vary, producing a quite changing network environment. Moreover, the network and computational resources can be modified due to the deployment or elimination of other services. The orchestrator must monitor that VNFs fulfill the SLAs and, therefore, the users receive the expected quality (KPIs or QPIs mentioned above). In addition, the orchestrator must re-locate or re-configure VNFs if necessary. Currently, the algorithms that perform these task still admit a quite margin of improvement.

Phase 4: ***Network service termination.*** This phase is not represented in Figure 4, and is devoted to releasing the resources previously allocated to service, as well as the elimination of VNFs images. These actions are taken when users are not expected to be connected for a long period of time, and thus, it is worth it the elimination and subsequent creation.

3 Overall proposal to create the MANO rules

Figure 5 shows our approach to generate the so called *book of rules* of a MANO in order to manage a specific service according to the requirements and performance specified in the SLA.

The approach has two differentiate phases. The first one is devoted to the extraction of the MANO rules using model-based testing and run-time verification techniques. The second phase focuses on the validation of the rules in an emulated 5G environment. Both phases have as input the VNF and service descriptors (VNFDs and VSFDs), and the SLA. In addition, it is worth mentioning that both phases can happen in more than one iteration in order to extract fine-grained rules.

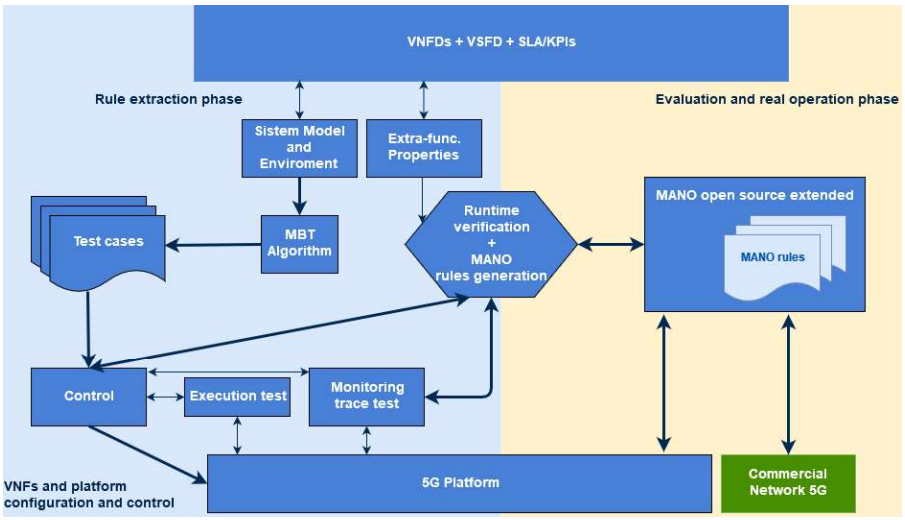

Fig. 5. overall approach.

In the first phase, shown on the left part of Figure 5, the VNFDs and VSFDs are transformed into a model of the service combined with a non-deterministic model of the MANO that includes a wide variety of rules to be applied to each network scenario. Using this model, we automatically generate test cases that show different management rules for different network scenario. The VNFs pass these test cases in a controlled environment, a testing platform for 5G, where they can be monitored. We use runtime verification techniques to determine whether the service and the VNFs satisfy the SLA. To do this, the SLA are translated into a set of extra-functional properties (the runtime monitors) that evaluate whether the execution each test case matches the desired SLA. The (non-)correct test cases help us to derive management rules for the MANO that satisfy the extra-functional properties.

The objective of the second phase is to validate the synthesized rules in a production environment. To this end, the rules are installed on a real MANO (shown on the right part of Figure 5) that manages and orchestrates the 5G testing platform (and can even manage a commercial 5G network). Again, using the runtime verification engine of the lab, we check the suitability of the synthesized rules during the normal operation of the service.

We propose to run multiple iterations of the approach. In each iteration, we will refine the NFV and MANO models based on the previous iteration results. These refined models can be used to extract new test cases that produce fine-grained rules. In the following section, we present some preliminary models of the NFVs and MANO.

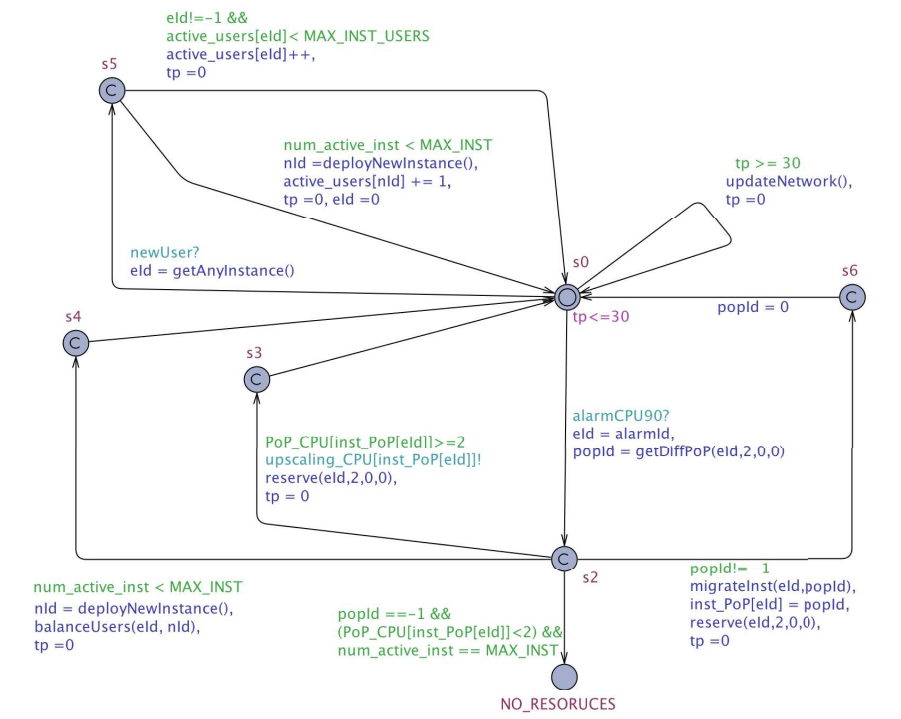

Fig. 6. Uppaal MANO Model

4 The initial model

In this section, we describe the construction of a non-deterministic MANO model from which the test cases can be produced as explained in Section 3. We assume that the MANO model consists of a number of MANO (sub-)models that execute concurrently. Each sub-model manages a unique service deployed on different PoPs to which users are connected. In consequence, in this section, we focus on the description of one of these sub-models. In the following, to simplify the presentation, we simply call it MANO model.

The MANO model is a state machine that responds to events provided by the infrastructure platform executing functions to preserve the required SLA. Thus, the MANO and the infrastructure communicate each other intensively over time. In addition, a complementary functionality of the MANO model is to periodically inspect the state of the infrastructure to carry out reconfiguration actions, if needed, even though no events are fired.

Figure 6 contains a prototype implementation of our proposal using Uppaal timed automata. State `s0` is the initial state of the monitor. At this state, the automaton may receive events such as `alarmCPU90` and `newUser` from the infrastructure through different synchronization channels. The first event occurs when

the platform detects that some instance is reaching the 90% in the CPU usage. The second one is a notification that a new user has connected to the service. The monitor responds to the first event by transiting to state `s2`. During the transition, the identifier of the PoP which has provoked the alarm is recorded in variable `eid`. The monitor also searches for a new PoP, with more resources, that can hold a new instance of the service, if needed. From `s2`, the automaton can go to states `s3`, `s4`, `s6` and `NO_RESOURCES`. This last state is an error which should not be never reached. The rest of states represent non-exclusive alternatives for the monitor: to increment the CPU resources in the PoP where the instance which fired the alarm is located, to deploy a new service instance and balance the users, to migrate the instance to a different PoP with enough CPU resources.

Similarly, the monitor responses to the `newUser` event transiting to state `s5` and finding an instance which can hold it. From `s5`, the automaton can jump to the initial state through two different transitions: assigning the new user to an existing service instance or deploying a new service instance for the user. As in the previous case, if both transitions are enabled, the automaton may select any of them in a non-deterministic way.

Finally, observe that the automaton has a clock variable `tp` that is used to periodically check the state network and update it, if necessary. Currently, we use a bi-dimensional array with the network parameters of interest that contain the network state at three ordered previous time instants. This information may be used, for instance, to discover when an user may have disconnected to release its resources.

We have not included transitions that deal with alarms due to the RAM or HDD usage in the model of Figure 6 to simplify the automaton. In addition, it is worth noting that the model is parametric wrt a number of constants that have to be calibrated such as the thresholds to fire alarms, the maximum number of service instances and users (`MAX_INST`, `MAX_INST_USERS`) and so on.

The concurrent execution of the MANO model and the infrastructure produces a set of traces (sequence of infrastructure states) that constitute the test cases to be analysed against the SLA. Figure 7 contains an example of a possible infrastructure execution which can be synchronized with the monitor of Figure 6 to produce test cases.

The execution shows changes in the network at certain time instants. For example, at time instant `t=9` the platform is already initialized and the values of HDD and RAM usage are updated (the second and third parameter of function `updateInfo`). At time instant `t=99`, a new user is connected and the infrastructure sends event `newUser` to the monitor.

As described in Section 3, the analysis of test cases may be used to iteratively improve the monitor rules (the transitions in the automaton of Figure 6).

The number of different traces generated in this way is very large due to the non-deterministic character of the monitor, and the range of values of the network variables. A way of pruning these traces is to use properties (described, for instance, in some temporal logic) to discard some non-interesting behaviours.

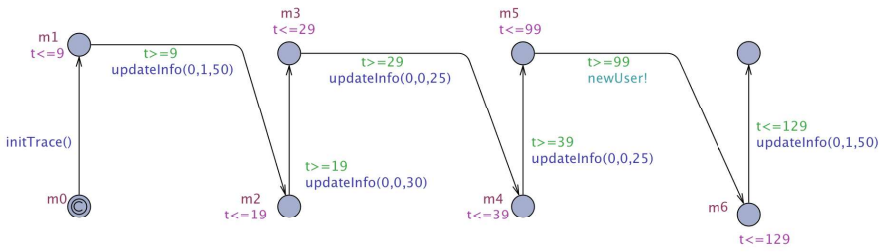

Fig. 7. An infrastructure execution

For instance, we could add the TCTL property "`A[](not monitor.s3)`" to the monitor to generate test cases in which the monitor has carried out at least an `upscaling_CPU` task. The Uppaal model checker tries to check if no trace is eventually at state `s3`. The counterexamples for this property are precisely the test cases of interest wrt the property specified.

5 Related work

The efficient management and orchestration of VNFs in the context of 5G networks is a interesting and challenging problem that can be addressed from different perspectives.

Currently, some proposals focus on the predicting the performance behavoiur of VNF chains (services deployed interconnecting VNFs) deployed in the cloud without considering the role of the mobile network. However, the network and the service users are an important part of the environment that stimulate and interacts with the VNFs, and thus, they must be taken into account to predict the service performance in terms of extra-functional properties, Service Level Agreements (SLAs), Key Performance Indicators (KPIs) and Quality Performance Indicators (QPIs).

Peuster and Karl [7] proposed a methodology to characterize the performance of VNF and VNF chains prior service deployment that can be part of a DevOps methodology. To this end, the authors execute the VNFs in different emulated network configurations and monitor the how different performance parameters evolve. The paper includes an evaluation of a profiler prototype built upon the emulation platform MeDICINE, which is based on an extension of Mininet that can execute production-ready VNFs (given as Dockers containers) in user-defined network topologies. The prototype can emulate the effect of the network and other services that compete for the resources. However, the emulation is limited by the capacity of the host machine that runs Mininet, which is far from emulating a real 5G network. The author states that the profile of the VNF can be used to improve the decision making of the MANO, but there is no insight of how to transform the profile into MANO rules.

Gym [8] is other framework for VNF profiling. In this case, the VNF is tested under different resource configurations of the infrastructure, which is mainly composed by servers where the VNFs runs. However, the the infrastructure lacks of the components of a mobile network. The authors' objective is to use the framework to build testbed for NFVs and services extending the framework with new components.

The characterization of VNFs performance has been also addressed from the analytical point of view. For instance, the tool Probius [5] aims to detect abnormal behaviours of NFVs due to performance uncertainties. Probius automatically generate all possible service chains with the given VNFs, collects and analyzes performance related features of each chain, and analyzes performance problems through anomaly detection and graph-based behaviour analysis, and it is able to point out the reasons of the VNFs performance issues.

The project 5GTANGO [10] proposed a testing approach for VNFs based on TTCN-3 test cases that can be manually or automatically generated using model-based testing techniques. However, the authors do not discuss which entities are included in the model, or what requirements guide the test generation algorithms.

In [7, 5, 10], the VNFs, the services and the test cases run in a emulated infrastructure that cannot properly represent a 5G network. Our proposal will be built upon a testbed that includes a complete LTE network, and that will be extended in the near future to behave as a 5G network. The testbed includes test automatization mechanisms that allow test execution and reporting without human intervention.

Formal methods have been also used to verify VNFs and VNF chains against reachability and safety properties in order to determine whether services are interfering, are isolated, or are accessed by unauthorized users. In [6] and [9] the analysis of properties is based on SMT solvers, such as Z3, combined with static analysis and symbolic model checking. These approaches accepts as input a logic formula and find the values (if any) that make the formula satisfiable. The main limitation of these approaches is the transformation of the VNF code into a model in solver's input language, this task is not trivial and is error prone. To minimize this problem, in [4] proposed a tool to automatically extract the VNF model from its code. The proposal is limited to VNFs implemented using a set of Java libraries to facilitate the coding task.

6 Conclusions

We have presented a novel approach to use formal methods in order to produce the knowledge for the MANO engines to manage 5G network efficiently. This is work in progress, and we are now defining the details for a first implementation. As far as using realistic networks is a major contribution, we plan to use two relevant testbeds at UMA. TRIANGLE testbed is a lab environment fully automated to emulated the network, so we can use it for the first phase to generate

the MANO rules. 5GENESIS Malaga is the field deployment to run services with real users, and it will be used for the validation phase.

References

1. ETSI GS NFV: Network Functions Virtualization (NFV); Terminology for Main Concepts in NFV. Tech. Rep. ETSI GS NFV 003, European Telecommunications Standards Institute (ETSI) (08 2018), v1.4.1
2. ETSI GS NFV-IFA: Network Functions Virtualization (NFV); Management and Orchestration; VNF Descriptor and Packaging Specification. Tech. Rep. ETSI GS NFV-IFA 011, European Telecommunications Standards Institute (ETSI) (08 2018), v2.5.1
3. ETSI GS NFV-MAN: Network Functions Virtualization (NFV); Management and Orchestration. Tech. Rep. ETSI GS NFV-MAN 001, European Telecommunications Standards Institute (ETSI) (12 2014), v1.1.1
4. Marchetto, G., Sisto, R., Virgilio, M., Yusupov, J.: A Framework for User-Friendly Verification-Oriented VNF Modeling. In: 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC). vol. 1, pp. 517–522 (07 2017). https://doi.org/10.1109/COMPSAC.2017.16
5. Nam, J., Seo, J., Shin, S.: Probius: Automated approach for vnf and service chain analysis in software-defined nfv. In: Proceedings of the Symposium on SDN Research. pp. 14:1–14:13. SOSR '18, ACM (2018). https://doi.org/10.1145/3185467.3185495
6. Panda, A., Lahav, O., Argyraki, K.J., Sagiv, M., Shenker, S.: Verifying isolation properties in the presence of middleboxes. CoRR **abs/1409.7687** (2014)
7. Peuster, M., Karl, H.: Understand your chains: Towards performance profile-based network service management. In: 2016 Fifth European Workshop on Software-Defined Networks (EWSDN). pp. 7–12. IEEE Computer Society (2016). https://doi.org/10.1109/EWSDN.2016.9
8. Rosa, R.V., Bertoldo, C., Rothenberg, C.E.: Take Your VNF to the Gym: A Testing Framework for Automated NFV Performance Benchmarking. IEEE Communications Magazine **55**(9), 110–117 (09 2017). https://doi.org/10.1109/MCOM.2017.1700127
9. Spinoso, S., Virgilio, M., John, W., Manzalini, A., Marchetto, G., Sisto, R.: Formal verification of virtual network function graphs in an sp-devops context. In: ESOCC. Lecture Notes in Computer Science, vol. 9306, pp. 253–262. Springer (2015)
10. Zhao, M., Le Gall, F., Cousin, P., Vilalta, R., Muoz, R., Castro, S., Peuster, M., Schneider, S., Siapera, M., Kapassa, E., Kyriazis, D., Hasselmeyer, P., Xilouris, G., Tranoris, C., Denazis, S., Martrat, J.: Verification and validation framework for 5g network services and apps. In: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). pp. 321–326 (Nov 2017). https://doi.org/10.1109/NFV-SDN.2017.8169878