Modeling and Assessing Connectivity Services Performance in a Sandbox Domain

Marc Ruiz, Mario Ruiz, Fatemehsadat Tabatabaeimehr, Lluís Gifre,
Sergio López-Buedo, Jorge E. López de Vergara, Óscar González, and Luis Velasco

***Abstract*—The automation of Network Services (NS) consisting of virtual functions connected through a multilayer packet-over-optical network requires predictable Quality of Service (QoS) performance, measured in terms of throughput and latency, to allow making proactive decisions. QoS is typically guaranteed by overprovisioning capacity dedicated to the NS, which increases costs for customers and network operators, especially when the traffic generated by the users and/or the virtual functions highly varies over the time. This paper presents the PILOT methodology for modeling the performance of connectivity services during commissioning testing in terms of throughput and latency. Benefits are double: first, an accurate per-connection model allows operators to better operate their networks and reduce the need for overprovisioning; and second, customers can tune their applications to the performance characteristics of the connectivity. PILOT runs in a *sandbox domain* and constructs a scenario where an efficient traffic flow simulation environment, based on the CURSA-SQ model, is used to generate large amounts of data for Machine Learning (ML) model training and validation. The simulation scenario is tuned using real measurements of the connection (including throughput and latency) obtained from a set of active probes in the operator network. PILOT has been experimentally validated on a distributed testbed connecting UPC and Telefónica premises.**

***Index Terms*—Sandbox domain, Network Automation, Performance Modeling.**

I. INTRODUCTION

MORE and more connectivity services are requiring not only stringent, but also more predictable Quality of Service (QoS) performance, measured in terms of key performance indicators (KPI) such as throughput and latency (delay). Accelerated by such requirements, new solutions for the control and orchestration of the optical transport network are being proposed (see e.g. [1]). Such services are supported by a packet layer on top of an optical network, where the latter covers core and metro segments and provides high capacity with low latency and high reliability [2]. Solutions currently under research to guarantee the requested performance are Network Function Virtualization (NFV) and *network slicing*, where NFV Network Services (NS) consist of interconnected Virtual Network Functions (VNFs) placed in different Central Offices (CO). Note that, as specific network resources are reserved to every NFV NS, the performance is guaranteed at the cost of high overprovisioning unless dimensioning is carefully carried out. Even though the performance is bounded, it cannot be precisely estimated as a function of the input traffic, which might be of interest for both network operators to reduce overprovisioning, and for customers to implement autonomic NFV services (see, e.g., [3]-[5]).

The performance of a layer 2 (L2) / layer 3 (L3) packet connection can be assessed during the commissioning phase through active monitoring, as we demonstrated in our previous paper in [6], using a 100 Gb/s active probe. The methodology is different from that proposed by the IP Performance Measurement (IPPM) Working Group [7] that uses dissimilar measurements for each performance indicator. We measure a packet connection by using an active probe at the source to inject a train of numbered and timestamped packets; when the train arrives to the other end of the connection, another active probe measures throughput, by using the reception times of every packet, and latency, by comparing the transmission timestamp with the reception time of each packet. Note that this latency measurement requires a common reference clock for the active probes, which is provided by a Global Positioning System (GPS) receiver to achieve the needed accuracy [8]. In the case that the connectivity is implemented by a point-to-multipoint (p2mp) multicast connection [9], instead of a point-to-point (p2p) one, every probe in a destination will measure the performance. Following this procedure, packet connection performance, i.e., one-way packet delay, delay variation (jitter), packet loss, and throughput, can be measured (see [10], [11] for details). However, as the length of each measurement train and the packet separation are constant, the obtained measurements can be considered as a bound, since they are not related to the specific traffic that the connection will support.

Machine Learning (ML) [12] can help to improve the predictability, as well as to assess the performance of connectivity services; ML models (e.g., Artificial Neural Networks -ANN) can be trained and used to estimate the performance of end-to-end packet connections. Note that by considering ML models for such estimation, the details of the network are abstracted and thus, they can be shared with the

Manuscript received November 22, 2019; revised February 11, 2020; accepted February 19, 2020.

Marc Ruiz, Fatemehsadat Tabatabaeimehr, and Luis Velasco (lvelasco@ac.upc.edu) are with the Optical Communications Group at Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. Mario Ruiz, Sergio López-Buedo, and Jorge López de Vergara are with Universidad Autónoma de Madrid (UAM) and Naudit HPCN, Madrid, Spain. Lluís Gifre is with Nokia Bell-Labs, Nozay, France. Óscar González is with Telefónica, Madrid, Spain.

Cite as: M. Ruiz, M. Ruiz, F. Tabatabaeimehr, L. Gifre, S. López-Buedo, J.E. López de Vergara, Ó. González, L. Velasco, “Modeling and Assessing Connectivity Services Performance in a Sandbox Domain”, *IEEE/OSA Journal of Lightwave Technology*, 2020. doi:10.1109/JLT.2020.2975641

final customers. However, to obtain accurate models, training and validation procedures need to be carried out, which entails the availability of a large amount of data. To this end, the control and orchestration of the interconnection network [13] can be augmented with a Monitoring and Data Analytics (MDA) controller [14] running besides the Software-Defined Networking (SDN) controller, to collate measurements from the packet nodes, which must be also labeled with the traffic measured at the input of the connection. In that regard, many works can be found in the literature regarding the use of collected data for network automation. In these works, data analytics algorithms running in the MDA controller are able to discover knowledge to automate the network operation [15], as well as to detect anomalies and degradations during commissioning testing and operation at both the optical and the packet layers (see, e.g., [16]-[18]). Such detection triggers the notification to the SDN controller for network reconfiguration and maintenance [19].

However, obtaining specific data to model a given connection takes a long time to collect those data and as NFV NSs might be highly dynamic, a different approach is required to reduce the time to create the training and testing datasets. For this reason, the use of a *sandbox domain*, where ML models can be trained with data from the network and from simulation, is required in [20]. In this regard, the behavior of the queues in packet nodes along a connection can be studied using realistic and accurate G/G/1 queues [21] in combination with realistic input traffic. Aiming at providing fast and scalable approaches, *continuous queue models* can be used to simulate G/G/1 queue systems. Among different models, the Vickrey's point-queue model [22] allows formulating an uncapacitated queue system as a differential equation that depends on input and output *traffic flows*. Concerning this, in our previous work in [23], we proposed a methodology named CURSA-SQ to analyze traffic flows by modeling both service traffic and the behavior of the queues. The CURSA-SQ methodology is based on: *i*) modeling the input traffic related to a given service *s*; and *ii*) a continuous capacitated G/G/1/k queue model with a First-In-First-Out (FIFO) discipline based on the logistic function.

As CURSA-SQ is able to reproduce the characteristics of a packet connection, it can be used to generate the large dataset needed for ML training and validation, thus reducing the amount of real measurements that would otherwise be obtained by the active probes, as well as helping to obtain end-to-end ML models of packet connections. In CURSA-SQ, a packet connection is modeled as a path (p2p) or a tree (p2mp), where every output interface is represented by a queue that includes the transmission delay introduced by the links. We also assume that the client defines the expected traffic characteristics at NS set-up time, which can be obtained from ad-hoc or available studies (see, e.g., [24], [25]).

As in our previous paper in [6], we also assume that active probes are equipped in the COs. However, in this work the probes can be programmed to measure the performance of the connection as measured at the destination(s) for specific configurations of packet trains that follow the expected traffic. In this regard, the authors in [26] used specific packet trains to measure different scenarios like network congestion and daily variations. However, those configurations are not related to real traffic conditions. In particular, such measurements help to calibrate the scenario in CURSA-SQ. Specifically, the contributions of this work are:

- Section II presents the PILOT methodology for modeling connectivity services in a typical infrastructure in a sandbox domain. COs are interconnected by a multilayer packet-over-optical network. PILOT uses active probes deployed in the COs to obtain real measurements that will be used to tune a CURSA-SQ-based simulation scenario reproducing the real deployment with high accuracy. Eventually, the CURSA-SQ-based simulator is used to generate large amounts of realistic synthetic data for ML training and validation.
- The key PILOT components are detailed in Section III, including: *i*) how the customers need to specify the traffic mix that every connection will support; *ii*) how to sample such traffic mix to minimize the amount of real measurements that need to be performed; and *iii*) how the real measurements are used to tune the CURSA-SQ-based simulation scenario.
- A workflow is proposed in Section IV to integrate the PILOT methodology, which includes the active probes for real measurements, in the provisioning of NSs. The hardware implementation of the active probe is also summarized, including how measurements are performed and results are reported.

The discussion is supported by the experiments in Section V, where the active probes are firstly validated and then, the proposed workflow and the PILOT methodology are experimentally assessed.

II. Modeling and Assessing Connection KPIs

In this section, we detail our proposal for modeling and assessing connection performance, based on one-way active network measurements. Fig. 1 presents an infrastructure where a multilayer network interconnects three COs. In the control, orchestration, and management (COM) plane, an NFV

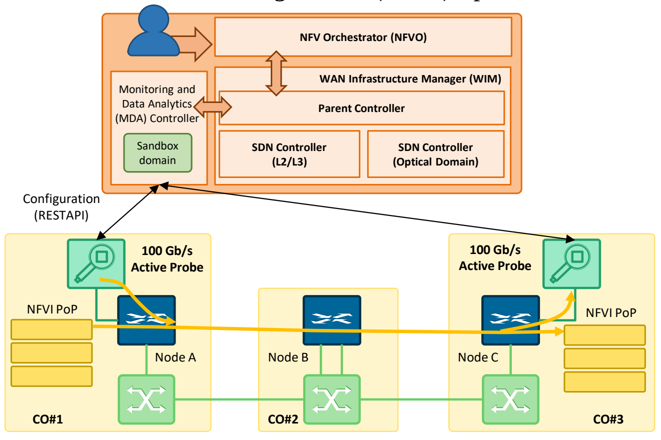


Fig. 1. Reference architecture of control and data planes.

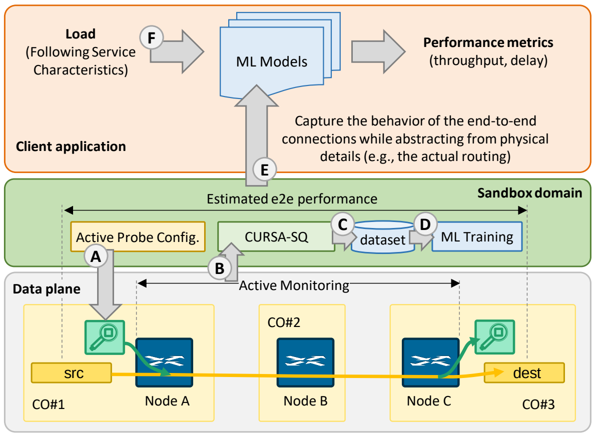


Fig. 2. Sandbox domain proposed in this paper.

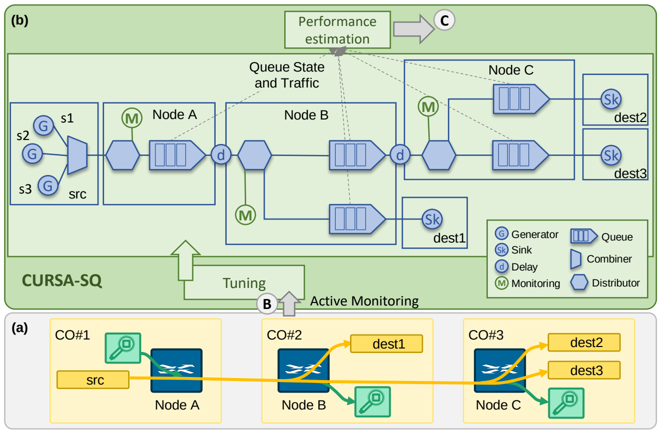


Fig. 3. Example of p2mp connection (a) and CURSA-SQ-based simulation (b).

orchestrator (NFVO) deals with the deployment of NFV NSs. The multilayer interconnection network is controlled by a hierarchical SDN architecture, where a Parent controller is on top of per-layer SDN controllers, one for the packet and another for the optical layer. Finally, an MDA controller collects monitoring data from the underlying infrastructure, trains ML models in the sandbox domain, and applies data analytics techniques on these data.

Given that one of the main aspects of network automation is to guarantee that provisioned connections actually meet the requested performance, active probes are equipped at the packet layer to measure the performance of packet connections. The active probes are installed in every CO and connected through a 100GbE interface of a L2 switch in the internal CO network. Being the active probes connected to interfaces configured in trunk mode, the probes can tag the generated Ethernet frames with the desired VLAN ID, selecting the VLAN to be measured. The active probes have been developed to be integrated in the above described COM system. To that end, they expose a REST-API-based northbound interface (NBI) through which the MDA controller can configure them and initiate a measurement session on a specific packet connection.

In line with the ETSI NFV architecture [27], an NFV Infrastructure (NFVI) is deployed across multiple points of presence (NFVI-PoP) for supporting the instantiation of VNFs. In the example in Fig. 1, a simple NS that consists of two VNFs in CO#1 and CO#3 interconnected by a unidirectional packet connection is shown. The packet circuit has resources reserved in Nodes A, B and C, and its performance is measured by the active probe in CO#1 that acts as a sender and that in CO#3 that acts as a receiver. Although the active probes provide really accurate measurements between them, note that because they are connected to the packet node that provides the external connectivity to the CO (Node A and C), those measurements do not include parts of the network, like the internal PoPs' networks. For these very reasons, as well as to generate the large training and validation dataset, we propose to use CURSA-SQ to emulate the complete connectivity set-up and to generate useful models for the customers. However, real measurements are strictly needed to tune the CURSA-SQ scenario, which includes *additional* delays (e.g., transmission and other processing-related delays, see [30] for a complete delay model) and fine tune of the queues.

Fig. 2 illustrates the above concepts, where the active probes measure the performance of the circuit between Node A and Node C. The proposed PILOT methodology runs inside a sandbox domain in the MDA controller, and it includes a module to configure the probes to perform the required measurements based on the characteristics of the services provided by the customer using two random variables (inter-arrival burst rate -IBR- and the burst size -BS) (step A in Fig. 2); the resulting measurements are collected and used to tune a network simulator based on CURSA-SQ (B). Once enough data is generated (C), ML models are trained and validated (D) and they can be shared to the connection's customer (E), which will use them to estimate performance metrics based on the load (F). Note that the produced ML models provide a way to reproduce the behavior of the connection without revealing the internal routing or other network details, which facilitates being shared to end customers. An alternative to ML models would be providing abstracted end-to-end performance data, at the cost of moving large volumes of data.

To illustrate the network simulation process, Fig. 3a shows an example of a slightly more complex p2mp connection defined between a source VNF and a set of destinations, i.e., <*src*, {*dest1*, *dest2*, *dest3*}>. Let us assume that the customer has specified that such p2mp connection will be used to convey a set of services S (in the example S={$s1$, $s2$, $s3$}). Using the details of the p2mp connection received from the Parent SDN controller, which include the route of the tree, the resources actually reserved, the traffic specifications, and the QoS constraints. In the sandbox domain, the PILOT algorithm defines the scenario for CURSA-SQ-based simulation (Fig. 3b). The CURSA-SQ scenario includes a traffic generator (G) in the source VNF for each of the services specified, a sink node (Sk) for each VNF destination, dimensioned queues for each output interfaces in the route of the tree, and delay nodes (d) emulating the delay introduced by the links.

Next, PILOT configures the active probes to measure the performance at every destination CO in the packet connection that will be used to tune the CURSA-SQ scenario, e.g., to ensure that any additional delay is included in the real set-up.

Furthermore, in the case of p2mp connections, the probes join a multicast group created by the Parent SDN controller specifically for the commissioning tests; the source probe uses the multicast group as destination IP address for the generated packets. Once the probes are configured, PILOT generates measurements configurations that include the definition of bursts mimicking the specified mix of services at meaningful values of IBR and BS random variables. Once the results are received from the destination probes (step B), PILOT uses them to tune the simulation scenario and runs CURSA-SQ to generate a large amount of labeled data for ML training and validation (C). The next section describes the PILOT methodology in depth.

III. Combining Measurements and Synthetic Data

The general scheme of the PILOT methodology is sketched in Fig. 4. PILOT entails three sequential stages to be carried out to produce accurate ML models for each packet connection. PILOT relies on the specification of the traffic mix that the connection will support. Specifically, the mix of traffic is defined in terms of services characterized by, at least, IBR and BS random variables, and a scaling factor. Such specification of the traffic mix is used to generate meaningful active probe configurations in terms of packet trains that are generated by the active probes and which measurements are used to tune the CURSA-SQ scenario. Once experimentally assessed, synthetic data that reproduces the real connection is generated, and accurate ML models can be trained and validated. The following sub-sections elaborate on key PILOT methodology components.

A. *Traffic mix specification*

As introduced in the previous section, we assume that a traffic specification is received for each connectivity request. Such specification includes the characterization of the set services *S* that the connection will support. Service characterization must include, at least, the statistical distribution and associated parameters of the IBR and BS *burst-level* random variables plus the scaling. The characterization can be provided by the customer or the network operator, and can be based on specific measurements or on studies available in the literature (see [23]).

From the received service characterization, we define the traffic specification χ_s for service *s*, as follows:

$$\chi_s = \{IBR_s \sim f(\theta_s), BS_s \sim g(\vartheta_s)\}, \forall s \in S \qquad (1)$$

where *f* and *g* denote probability distribution functions with their respective parameters. In line with [23], we consider that IBR and BS can be treated as independent variables; indeed, *f* and *g* can belong to distinct families of probability distributions. In addition, the characterization of the services at *packet-level* can lead to a more precise configuration of active probe measurements. In that regard, *packet size* (PS) is an additional random variable that could be included in χ_s.

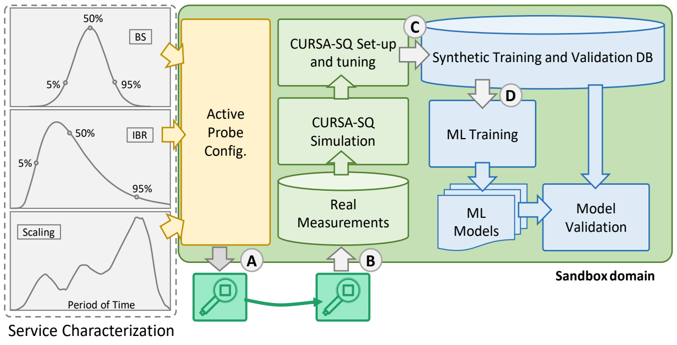


Fig. 4. Overview of the PILOT Methodology.

The expected demand needs also to be included for scaling of each service. We denote this input as $u_s(t)$, where the scaling (e.g., number of individual users) of each service *s* is defined as a function of time. The specific time range is defined by the customers according to their interests, and it can cover from hours/days/weeks (e.g., a typical tidal profile that periodically repeats on time) to months (e.g., the expected user evolution during connection lifetime). Note that this flexible definition of the expected load opens the possibility to carry out several analysis leading to different KPI modelling for short, medium, and long-term applications. For the sake of simplicity, we assume the same time range for all the services in a connection.

The statistical properties of the services and their expected demand in time are key to understand and define the *traffic flow* $x(t)$ (bitrate, defined in b/s) injected into the connection. For modelling and simulation purposes (mainly for generation), we consider to model services separately, and therefore, the expectation (*E*) and variance (*V*) of each service in the flow can be computed as follows:

$$E\left(x_s(t)\right) = u_s(t) \cdot E(BS_S) \cdot E(IBR_S) \qquad (2)$$

$$V\left(x_s(t)\right) = u_s(t) \cdot V(BS_S \cdot IBR_S) \qquad (3)$$

where the variance of the product of BS and IBR can be estimated from well-known approximations of the variance of the product of two independent variables [28]. Note that the connection traffic flow $x(t)$ is the aggregation of all $x_s(t)$.

B. *Traffic Sampling and Measurements Configurations*

Let us now detail the procedure for sampling the traffic flows $x_s(t)$ to obtain real measurements for those traffic samples using the active probes under realistic traffic conditions (see Fig. 4). We have redefined the synthetic packet generation in the active probes for this purpose (see Section IV.A), where a measurement request is defined by a number of packet bursts, each containing packets of a given size. The definition of the bursts and the delay between two consecutive ones can be defined to reproduce a desired traffic pattern. The objective is then to define how measurement configurations are created to follow the main statistical characteristics of the specified traffic mix for the connection.

Fig. 5 summarizes the concept behind the generation of measurement configurations. Service characteristics are processed by a sample generator module that generates a set of

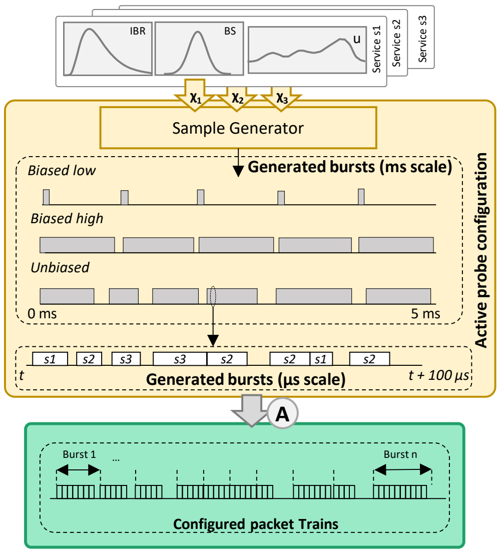


Fig. 5. Active probe configuration procedure.

samples of a given duration of bursty traffic; the duration, e.g., 5 ms, is defined by the capacity of the connection. The sample generator module firstly computes the expectation and variance of IBR and BS random variables based on their probability distributions. Then, several time values conveniently spaced in the time range of $u_s(t)$ are selected, thus covering relevant traffic mixes for low, medium, and high loads. For each selected time and mix, samples are generated according the expectation and variance references. In particular, three classes of samples are considered: *i*) *unbiased* samples, where E(IBR) and E(BS) are used for all the services; *ii*) *biased low*, where both E(IBR) and E(BS) are decreased by their respective variance values V(IBR) and V(BS), and *iii*) *biased high*, where both E(IBR) and E(BS) are increased by their respective variance values V(IBR) and V(BS). Regardless of the class, a sample is generated as a sequence of BS and IBR values (mixing services) around their expectation with some additional random variation defined within their variance magnitude. Note that unbiased samples allow measuring KPIs in average cases, which is intended for computing throughput and average latency. On the contrary, biased samples are designed either for measuring additional delays in the absence of queued traffic (*low*) or stressing the connection capacity to compute maximum latency and packet losses (*high*).

It is important to analyze the generated samples at different time resolutions. Assuming *self-similarity*, at coarse resolution (ms scale) traffic can be seen as a sequence of on/off periods of mixed services, whereas, at a finer resolution (µs scale), the sequence of on/off periods can be seen between bursts of differentiated services. This degree of detail is required to generate precise active probe configuration.

From the generated samples, a procedure to adjust and configure bursts of trains of packets in the active probe is required. Thus, a burst in a measurement configuration corresponds to a total (or partial) burst in a sample. Note that, to regularize the length of the bursts of packet trains, a minimum and maximum number of packets can be setup.

C. *CURSA-SQ tuning and ML model training*

The real measurements for the set of meaningful configurations obtained are used to tune and validate the CURSA-SQ scenario that will be eventually used to generate synthetic data for model training and validation purposes.

CURSA-SQ requires configuring a set of traffic generators and this can be done according to equations (2) and (3), as proposed in [23]. However, to reproduce by simulation exactly the same sampled scenarios measured by the active probe, CURSA-SQ traffic generation needs to be altered with the deviation introduced in unbiased measurements.

The generated traffic flows are then propagated through a system of continuous queues Q that models the connection, as represented in Fig. 3. Let us assume that every queue $q \in Q$ is characterized by a unique and common buffer with capacity k (in bytes) and a server rate μ (in b/s). Moreover, $q(t)$ represents the queue state, i.e., the number of bytes in the queue at time t. From such state, partial KPIs are computed for each individual queue. Then, computing connection KPIs, i.e., throughput and latency measurements between the source and all the destinations, is simply the aggregation of partial KPIs computed in queues, as well as in delay nodes.

Despite of the fidelity of the CURSA-SQ-based simulation setup to represent a real connection, there are two main unknowns that need to be discovered after analyzing real measurements. First, the magnitude of the additional delay to be introduced by delay nodes can be easily computed after analyzing the mean latency obtained for biased low measurements. Second, as measurements are correlated to some traffic behavior at the burst-level, but they are actually propagated packet-by-packet during measurements, a mismatch between theoretical and measured traffic behavior can exist. To solve this issue, a *correction factor* (multiplier) can be applied to both expectation and variance configured in the generators in order to fit the characteristics of the expectation and variance measured. A good reference for this purpose is to compare the difference between minimum and maximum latency in the simulation and in the experiments for the case of the biased high monitoring samples. The multiplication factors can be easily obtained to correct the deviation between simulation and experimental values. After this tuning operation, unbiased measurements can be used to validate the accuracy of the simulation environment.

Once CURSA-SQ is tuned and the KPIs obtained by simulation match the experimental ones for the measured samples, a large set of synthetic KPI measurements for a wide range of connection loads along the whole time period can be easily obtained. The data is eventually used for producing ML models that allow the customer to estimate KPIs for each connection destination point as a function of the expected traffic mix. In particular, we use ANNs that, given the aforementioned input, return an output vector with estimation of average throughput, average latency, maximum latency, and packet loss (if any).

IV. Active Measurements

In this section, we first describe the main characteristics of the packet generation and measurements process, and then, we present the proposed workflow to be carried out when a new packet connection is requested.

A. *Synthetic Packet Generation and Measurements*

We have extended the functionality of the implementation in [6] to send a configurable number of bursts of packet trains, each with different characteristics, as defined in Section III. The format of the generated packets is compatible with iPerf, a well-known active measurement tool [29]. Measurements are identified by an ID that is included in every packet, transmitted as an UDP datagram. Every generated packet is also labeled with two IDs: one to identify the burst it belongs to and another for the sequence number. In addition, each packet includes a transmission timestamp with nanosecond resolution (precision of 3.1ns). Finally, the rest of the payload of the UDP datagrams is defined by a pattern selected according to a concrete Bit Error Rate Test (BERT), such as pseudorandom (PRBS), all zeros, all ones, alternated, etc. The generated Ethernet frames are tagged with the desired VLAN ID. To send the packets deterministically following the bursts specifications, the transmitter is implemented as a Finite State Machine. Packets are spaced to perform measurements on packet connections with throughput up to 100 Gb/s in the most demanding scenario.

At the receiver side, UDP datagrams are timestamped and then checked and correlated to the measurement. After that, they are parsed to retrieve the information needed to compute the performance metrics. For statistical reasons, the specified bursts can be generated several times, named *repetitions*. Then, performance metrics are computed individually for every received packet and accumulated into two different sets of counters to compute *measurement-wise* and *repetition-wise* performance indicators, which are then summarized with maximum, arithmetic mean, and minimum values.

B. *Proposed Workflow*

The proposed workflow is shown in Fig. 6. It starts when an operator requests the deployment of a new NS through the NFVO's Graphical User Interface (GUI), which defines the interconnected VNFs, as well as the specification of the traffic that can be expected and the QoS constraints in terms of throughput and latency (message 0 in Fig. 6). That request triggers the set-up of a number of packet connections, which the NFVO requests through the Parent controller's NBI and includes the traffic specification and the QoS constraints (1). Once the requested connectivity is set-up, the Parent controller updates the MDA controller with the connection ID and its attributes (2). Next, the Parent controller creates an IP multicast group that will be used exclusively for running the commissioning tests, and requests the MDA controller to assess whether the configuration of the connection meets the QoS constraints and to model its performance under the specified traffic (3). Upon the reception of that request in the MDA controller, the PILOT application is triggered. PILOT first

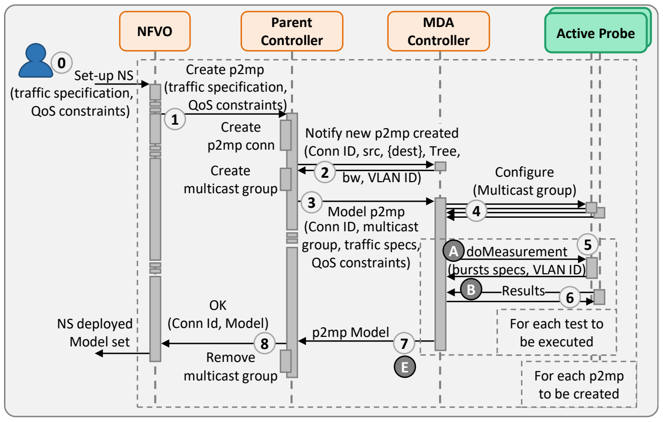


Fig. 6. Proposed workflow for a p2mp connection.

determines the probes that will be involved in the tests and requests them to join the IP multicast group (4). In addition, PILOT determines the set of tests to be executed, and for every test, it requests the active probe in the source of the connection to run them specifying the composition of the packet trains that the probe needs to generate. The request includes the VLAN ID that has been configured so the active probe can use it for tagging the generated Ethernet frames (5).

To measure the performance of a connection, the active probe in the source CO injects trains of Ethernet frames that are received by the active probe in each remote CO. Once the performance of the connection in every destination CO is measured, the obtained results are notified to the MDA controller (6). With such measurements, the PILOT application configures the scenario and runs CURSA-SQ with specific parameters to generate a training dataset that is subsequently used to generate a ML model for every destination of the connection, as detailed in Section III. New tests are afterwards executed to validate the connection model. Once a ML model for the connection is produced, the MDA controller replies the Parent controller (7), which in turn replies the NFVO (8) and tears down the IP multicast group.

The proposed PILOT methodology, the active probes and the workflow are experimentally assessed in the next section.

V. Experimental Assessment

A. *Packet Connectivity Performance Measurements*

The active probes were firstly evaluated locally in a setup in UAM-Naudit premises in Madrid, Spain, (Fig. 7a) where the active probes were used to evaluate the performance of the connectivity between two L2 Ethernet switches connected through a L3 router with two 1 GbE interfaces.

The active probe functionality is implemented using a mix of Verilog and High-Level Synthesis, in a Field Programmable Gate Array (FPGA). We base our development on the Alpha Data ADM-PCIE-9V3 High-Performance Network Accelerator card, which features a Xilinx's Virtex Ultrascale+ FPGA (XCVU3P-2-FFVC1517), as well as 8 GB of DDR RAM and two 100 Gb/s Ethernet QSFP28 cages (Fig. 7b).

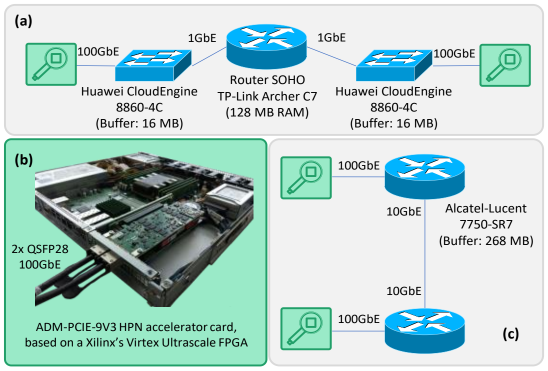
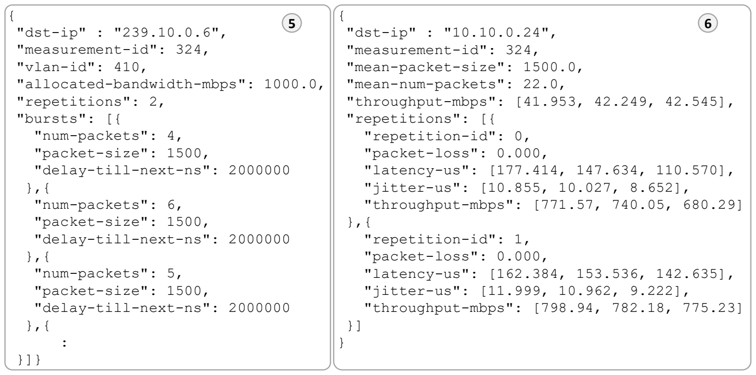

(a)
1GbE
1GbE
100GbE
100GbE
Huawei CloudEngine 8860-4C (Buffer: 16 MB)
Router SOHO TP-Link Archer C7 (128 MB RAM)
Huawei CloudEngine 8860-4C (Buffer: 16 MB)
(b)
2x QSFP28 100GbE
ADM-PCIE-9V3 HPN accelerator card, based on a Xilinx's Virtex Ultrascale FPGA
100GbE
10GbE
Alcatel-Lucent 7750-SR7 (Buffer: 268 MB)
10GbE
100GbE
(c)

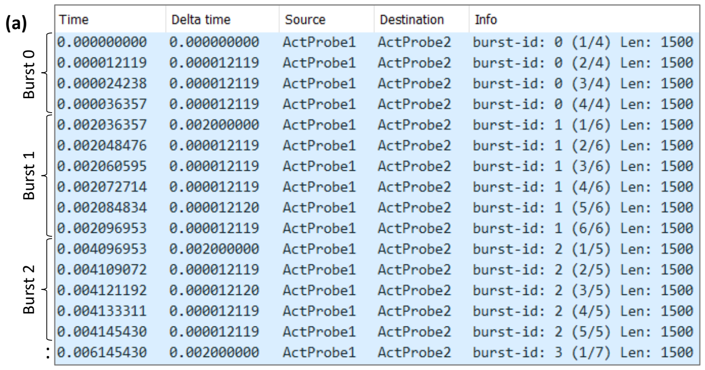
Fig. 7. Testbed scenarios and active probes.



```
{                                                    5
 "dst-ip" : "239.10.0.6",
 "measurement-id": 324,
 "vlan-id": 410,
 "allocated-bandwidth-mbps": 1000.0,
 "repetitions": 2,
 "bursts": [{
   "num-packets": 4,
   "packet-size": 1500,
   "delay-till-next-ns": 2000000
  },{
   "num-packets": 6,
   "packet-size": 1500,
   "delay-till-next-ns": 2000000
  },{
   "num-packets": 5,
   "packet-size": 1500,
   "delay-till-next-ns": 2000000
  },{
      :
 }]}
```

```
{                                                    6
 "dst-ip" : "10.10.0.24",
 "measurement-id": 324,
 "mean-packet-size": 1500.0,
 "mean-num-packets": 22.0,
 "throughput-mbps": [41.953, 42.249, 42.545],
 "repetitions": [{
   "repetition-id": 0,
   "packet-loss": 0.000,
   "latency-us": [177.414, 147.634, 110.570],
   "jitter-us": [10.855, 10.027, 8.652],
   "throughput-mbps": [771.57, 740.05, 680.29]
 },{
   "repetition-id": 1,
   "packet-loss": 0.000,
   "latency-us": [162.384, 153.536, 142.635],
   "jitter-us": [11.999, 10.962, 9.222],
   "throughput-mbps": [798.94, 782.18, 775.23]
 }]
}
```

Fig. 8. JSON messages for measurement configuration and results.



(a)

	Time	Delta time	Source	Destination	Info
Burst 0	0.000000000	0.000000000	ActProbe1	ActProbe2	burst-id: 0 (1/4) Len: 1500
	0.000012119	0.000012119	ActProbe1	ActProbe2	burst-id: 0 (2/4) Len: 1500
	0.000024238	0.000012119	ActProbe1	ActProbe2	burst-id: 0 (3/4) Len: 1500
	0.000036357	0.000012119	ActProbe1	ActProbe2	burst-id: 0 (4/4) Len: 1500
Burst 1	0.002036357	0.002000000	ActProbe1	ActProbe2	burst-id: 1 (1/6) Len: 1500
	0.002048476	0.000012119	ActProbe1	ActProbe2	burst-id: 1 (2/6) Len: 1500
	0.002060595	0.000012119	ActProbe1	ActProbe2	burst-id: 1 (3/6) Len: 1500
	0.002072714	0.000012119	ActProbe1	ActProbe2	burst-id: 1 (4/6) Len: 1500
	0.002084834	0.000012120	ActProbe1	ActProbe2	burst-id: 1 (5/6) Len: 1500
	0.002096953	0.000012119	ActProbe1	ActProbe2	burst-id: 1 (6/6) Len: 1500
Burst 2	0.004096953	0.002000000	ActProbe1	ActProbe2	burst-id: 2 (1/5) Len: 1500
	0.004109072	0.000012119	ActProbe1	ActProbe2	burst-id: 2 (2/5) Len: 1500
	0.004121192	0.000012120	ActProbe1	ActProbe2	burst-id: 2 (3/5) Len: 1500
	0.004133311	0.000012119	ActProbe1	ActProbe2	burst-id: 2 (4/5) Len: 1500
	0.004145430	0.000012119	ActProbe1	ActProbe2	burst-id: 2 (5/5) Len: 1500
:	0.006145430	0.002000000	ActProbe1	ActProbe2	burst-id: 3 (1/7) Len: 1500

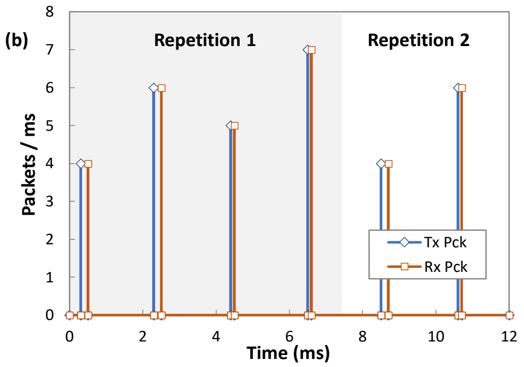


Fig. 9. Packets generated for a configured measurement (a) and aggregated generated and received packets (b).

These network interfaces are directly mapped to the FPGA, which is in charge of generating and receiving packets. To this end, a 512-bit width bus working at 322 MHz is implemented, so it is possible to work at 100 Gb/s even with the smallest packets and send and timestamp them accurately.

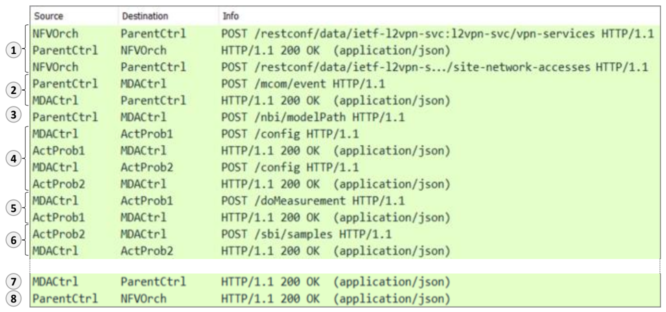
Fig. 8 shows the JSON messages used for measurement configuration and results, which correspond to messages 5 and 6 in the workflow. Message 5 includes the following fields (see Fig. 8): *i*) `dst-ip` specifies the IP address of the remote probe (unicast IP address) or probes (the multicast address group created by the Parent controller); *ii*) `measurement-id` uniquely identifies the measurement for correlation purposes; *iii*) `vlan-id` specifies the VLAN tag to be used; *iv*) `allocated-bandwidth-mbps` defines the bandwidth allocated for the connection and is used to define the inter-packet time; *v*) `repetitions` defines the number of times that the list of bursts needs to be sequentially repeated; vi) `bursts` specifies a list of bursts, where each burst is specified by: *a*) `num-packets` defines the number of IP packet to be sent in the burst; *b*) `packet-size` defines the size of each packet in octets; the used BERT type is PRBS by default; *c*) `delay-till-next-ns` defines the delay in nanoseconds until the next burst after the end of last packet in the current burst.

With these specifications, the source probe generates the packets for the measurement. Fig. 9a shows the generated packets, where one can easily identify the packets trains belonging to the first bursts and the correspondence with the specifications in message 5 in Fig. 8. In addition, to graphically illustrate the measurement configured, as well as the effects of the intermediate router, Fig. 9b shows the sequence of bursts measured at the output of the source active probe and at the input of the destination one, where some delay and jitter can be observed.

Every probe participating in the measurement returns the results to the MDA controller (6). The message includes (Fig. 8): *i*) `dst-ip` is the IP of the probe that reports the results; *ii*) `measurement-id` for correlation purposes; *iii*) `mean-packet-size` is the average size of the packets received; *iv*) `mean-num-packets` is the average number of packets received in every repetition; *v*) `throughput-mbps` specifies the maximum, average and minimum throughput measured for the measurement; and *vi*) `repetitions` contains a list with the results for each repetition of bursts sent, where each repetition include: *a*) `repetition-id` for correlation purposes; *b*) `packet-loss` reports the packet loss in percentage; *c*) `latency-us`, `jitter-us` and `throughput-mbps` report a list with the maximum, average and minimum latency, jitter, and throughput, respectively, measured during the burst, i.e., excluding inter-burst time, for each repetition independently.

B. *Workflow*

Once the active probes have been assessed, they were deployed in Telefónica premises in Madrid, Spain together with other commercial equipment and the Parent controller. Telefónica premises are connected to UPC ones in Barcelona, Spain, where the MDA controller is deployed. Such distributed testbed is used to carry out the complete experimental assessment and validate the proposed workflow. The setup is shown in Fig. 7c, where two active probes were deployed in Telefónica premises and connected to two Alcatel-Lucent routers through 100 Gb/s optical interfaces; the routers are



```
    Source       Destination   Info
  ┌ NFVOrch      ParentCtrl    POST /restconf/data/ietf-l2vpn-svc:l2vpn-svc/vpn-services HTTP/1.1
1 ┤ ParentCtrl   NFVOrch       HTTP/1.1 200 OK  (application/json)
  └ NFVOrch      ParentCtrl    POST /restconf/data/ietf-l2vpn-s.../site-network-accesses HTTP/1.1
2 ┌ ParentCtrl   MDACtrl       POST /mcom/event HTTP/1.1
  └ MDACtrl      ParentCtrl    HTTP/1.1 200 OK  (application/json)
3   ParentCtrl   MDACtrl       POST /nbi/modelPath HTTP/1.1
  ┌ MDACtrl      ActProb1      POST /config HTTP/1.1
4 ┤ ActProb1     MDACtrl       HTTP/1.1 200 OK  (application/json)
  │ MDACtrl      ActProb2      POST /config HTTP/1.1
  └ ActProb2     MDACtrl       HTTP/1.1 200 OK  (application/json)
5 ┌ MDACtrl      ActProb1      POST /doMeasurement HTTP/1.1
  └ ActProb1     MDACtrl       HTTP/1.1 200 OK  (application/json)
6 ┌ ActProb2     MDACtrl       POST /sbi/samples HTTP/1.1
  └ MDACtrl      ActProb2      HTTP/1.1 200 OK  (application/json)

7   MDACtrl      ParentCtrl    HTTP/1.1 200 OK  (application/json)
8   ParentCtrl   NFVOrch       HTTP/1.1 200 OK  (application/json)
```

Fig. 10. Exchanged messages in the workflow.

connected through a 10 Gb/s optical link thus creating a virtual link at the packet layer.

The exchanged messages during the set-up of a packet connection are presented in Fig. 10, where the messages are identified as in the workflow in Fig. 6 for the sake of clarity. When an operator in the NFVO needs to deploy a new NS, it specifies the traffic as well and the QoS constraints relevant for every connectivity service. The NFVO requests then to the Parent controller the creation of a new connection that is replied with a unique identifier for the connection. Next, the NFVO issues a second request defining the configuration attributes for the service to the Parent controller (messages 1).

Upon reception of the configuration request, the Parent controller creates the p2p or p2mp packet connection. Once the connection is set-up, it issues a notification to the MDA controller (2) through an interface named M-COM [17], which updates its operational databases; the attributes of the connection include: the connection ID, the ingress and egress connection point(s) of the connection, the path/tree topology, its bandwidth and the VLAN ID. Then, the Parent controller creates a multicast group that will be used for the commissioning testing and issues a request to the MDA controller to model the new packet connection (3); that request contains the connection ID, the multicast group IP address, the traffic specifications and the QoS constraints for this service.

Upon the reception of the request, the MDA controller first configures the active probes (4). Next, the MDA issues a request (5) to the sender probe to start the measurement process by generating the sequence of bursts that emulate the service to be implemented according to the traffic specifications. When a measurement is completed, the receiving probes issue requests to the MDA controller with the results of the measurements (6). At this point, new measurements can be requested by the PILOT algorithm in the MDA controller. After finishing the sequence of measurements, PILOT computes the model for the connectivity service and sends the computed model to the Parent controller (7), which in turn sends it to the NFVO (8) and removes the multicast group used for the tests. Eventually, the NFVO replies the operator with the NS creation and reports the computed models for the NS.

C. *Methodology validation*

The collected measurements were used to generate a relevant set of experimental measurements to tune a CURSA-SQ scenario emulating the real setup. Biased and unbiased samples were generated in the range of normalized load [0.1, 0.95], defined as the average traffic volume over the connection capacity (i.e., 10 Gb/s). We used the statistical service characterization and demand profiles in [23] for generating a mix of different services including Video-On-Demand, Online Gaming, and Internet services.

The results in Fig. 11 show the experimental measurements and the simulation data for two different configurations of the CURSA-SQ scenario, focusing on throughput and latency (both average and maximum) KPIs analysis. Precisely, Fig. 11a and Fig. 11b show the results obtained from unbiased samples for average analysis, whereas Fig. 11c focuses on biased high ones, which are relevant to illustrate the real behavior of maximum latency. In the first CURSA-SQ configuration, simulations have been conducted before tuning CURSA-SQ with the real measurements. The results show a slight reduction of throughput estimation, whereas latency is clearly underestimated due to: *i*) the lack of additional delays consideration, and *ii*) a different latency slope for high loads (clearly visible at loads 0.85 and 0.9). In the second CURSA-SQ configuration, simulations were conducted after tuning CURSA-SQ (according to Section III) using biased low monitoring samples at normalized load 0.1 and biased high ones at normalized load 0.9, for additional delay computation and generators corrections, respectively. As it can be observed, the evolution of all three KPIs as a function of the load closely matches with the experimental measurements.

Let us to illustrate the need to tune CURSA-SQ with measurements over packet trains that follow the expected traffic (as detailed in Section III) instead of over packet trains generically configured, i.e., independent to the expected traffic. In the latter, the active probes can generate two types of trains: *i*) with small packets and large inter-packet time to measure minimum latency; and *ii*) with large packets and small inter-packet time to measure maximum throughput [10]. Fig. 12 compares the performance at the highest loads of tuning CURSA-SQ based on real measurements over both active probes configurations, for throughput (a), average delay (b) and maximum delay (c). The results show a sub-estimation of the delay as large as 70%. In view of the results, we can conclude that the proposed packet trains configuration results in precise measurements and thus better CURSA-SQ tuning.

An additional analysis of the gain introduced by the proposed tuning methodology is depicted in Fig. 13, where the relative errors between simulated and real KPIs are computed for both CURSA-SQ configurations. It is worth noting that the remarkable reduction of error achieved by the tuning procedure. In fact, error below 3% for estimated average and 9% for maximum latency are obtained, which validates the simulation environment as accurate synthetic monitoring data generator for KPI model training purposes.

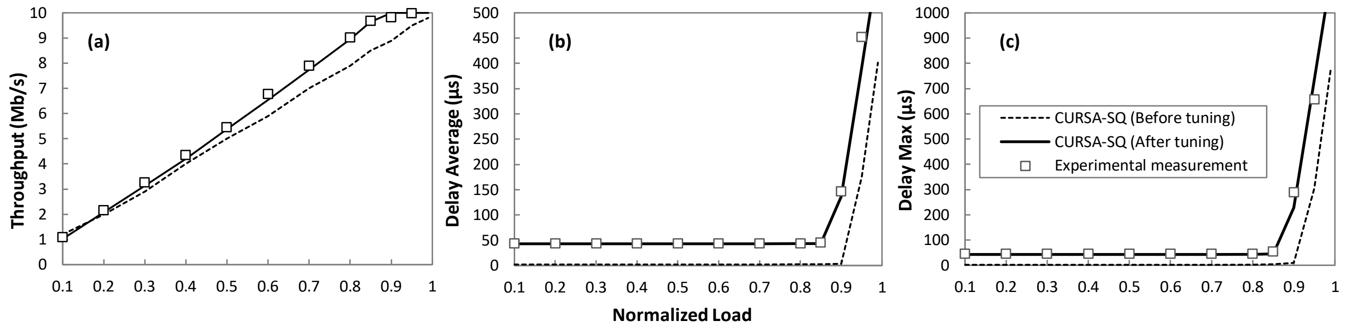


Fig. 11. Experimental and simulation results for KPI estimation: a) throughput, b) average, and c) maximum latency.

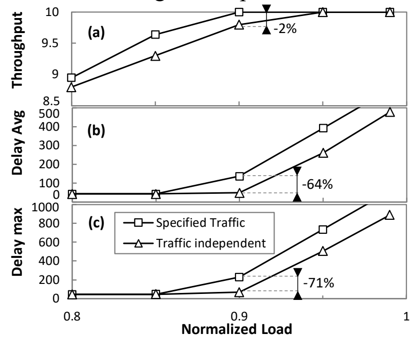


Fig. 12. CURSA-SQ tuning as a function of the injected packet trains.

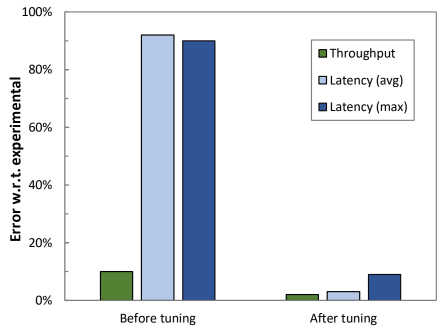


Fig. 13. Relative error of CURSA-SQ-based simulation.

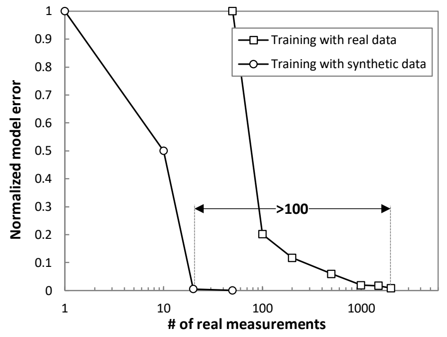


Fig. 14. Prediction error of ML models vs # of real measurements.

D. *Real vs Synthetic data for ML training*

Once the simulation environment has been validated, let us study the benefits of using PILOT with CURSA-SQ tuned with real measurements sampling the specified traffic mix, as compared to using just real measurements to generate labeled data for ML training and validation. To this end, we conducted a simulation-based study in a more complex scenario, according to the scheme represented in Fig. 3, where the delay nodes (d) implement a non-linear function that follows the behavior found in the previous subsection.

Fig. 14 shows the prediction error (normalized between the minimum and maximum) of the trained ML models as a function of the number of real measurements conducted. Note that the approach based on synthetic samples has been configured to perform ML training using 10,000 training data samples regardless of the number of real measurements used for the initial tuning. On the contrary, in the approach of ML training with real data, the number of training data samples equals that of real measurements. In both cases, an ANN with a single input for the normalized NS load and one output for each KPI was trained (Section III.C). A single hidden layer consisting in 10 hidden neurons with the logistic activation function was configured. The training procedure converged to accurate ANNs in less than one minute in a conventional desktop computer, which points out a negligible computational burden. The results clearly show the benefits of using the proposed PILOT methodology: the minimum model error is achieved with less than 100 times of experimental measurements as compared with the training with real data. In fact, note that the minimum error with the PILOT methodology is achieved when the approach based on real measurements has not yet enough data to initiate ANN training. Therefore, we can conclude that using CURSA-SQ tightened with relevant active measurements allows obtaining accurate KPI models for packet connections during commissioning testing.

VI. Concluding Remarks

A methodology named PILOT has been proposed and experimentally demonstrated to provide predictable connectivity services. The PILOT methodology facilitates reducing the cost of overprovisioning at both, the packet and the optical layer. PILOT is based on three main pillars that allow the generation of accurate ML models to estimate the QoS, in terms of throughput and latency, during commissioning testing: *i*) an efficient traffic flow simulation environment, named CURSA-SQ, to produce large amounts of labeled data for ML training and validation purposes; *ii*) real measurements to tune the CURSA-SQ scenario by discovering additional delay and throughput bottlenecks, which are usually not constant but related to the actual traffic load; and *iii*) specification of the estimated traffic mix that the connection will support are used in the process of data generation, which includes real measurements and traffic generation for the simulation scenario.

PILOT is carried out during the provisioning phase of NSs as part of commissioning testing. In this regard, a workflow has been proposed that executes PILOT for every connection related to the NS being provisioned. As a result, a ML model is produced for every connection and the set of models are returned to the client at the end of the provisioning phase.

The active probes were first experimentally validated in a local setup and then integrated in a distributed testbed connecting UPC and Telefónica premises, respectively, in Barcelona and Madrid, Spain, where the PILOT methodology and the proposed workflow were experimentally assessed. The results show noticeable accuracy of the produced ML models after the scenario in CURSA-SQ was tuned with real measurements. Last but not least, the benefits of using PILOT with CURSA-SQ were compared to using just real measurements to generate labeled data for ML training and validation in terms of number of real measurements needed to train accurate ML models. The results show that PILOT requires around 20 real measurements, which is 100 times less than performing ML training directly with real measurements.

ACKNOWLEDGMENT

This work was partially supported by the EC through the METRO-HAUL (G.A. nº 761727) project, from the Spanish MINECO/FEDER TWINS (TEC2017-90097-R) and TRÁFICA (TEC2015-69417-C2-1-R) projects and from the Catalan Institution for Research and Advanced Studies.

REFERENCES

[1] S. Fichera *et al*, "Latency-Aware Resource Orchestration in SDN-Based Packet Over Optical Flexi-Grid Transport Networks," IEEE/OSA J. of Optical Communications and Networking, vol. 11, pp. B83-B96, 2019.

[2] L. Velasco *et al*., "Saving CAPEX by Extending Flexgrid-based Core Optical Networks towards the Edges," IEEE/OSA J. of Optical Communications and Networking, vol. 5, pp. A171-A183, 2013.

[3] L. Velasco *et al*., "An Architecture to Support Autonomic Slice Networking [Invited]," IEEE/OSA J. of Lightwave Technology, vol. 36, pp. 135-141, 2018.

[4] L. Velasco *et al*., "A Control and Management Architecture Supporting Autonomic NFV Services," Photonic Network Communications, vol. 37, pp. 24-37, 2019.

[5] M. Ruiz *et al*., "Big Data-backed Video Distribution in the Telecom Cloud," Computer Communications, vol. 84, pp. 1-11, 2016.

[6] J. López de Vergara *et al*., "Demonstration of 100 Gbit/s Active Measurements in Dynamically Provisioned Optical Paths," in Proc. European Conference on Optical Communication, 2019.

[7] IETF IP Performance Measurement (IPPM) Working Group. [On-line] https://datatracker.ietf.org/wg/ippm/about/.

[8] M. Ruiz *et al*., "Harnessing Programmable SoCs to Develop Cost-effective Network Quality Monitoring Devices," in Proc. FPL, 2016.

[9] M. Ruiz and L. Velasco, "Serving Multicast Requests on Single Layer and Multilayer Flexgrid Networks," IEEE/OSA J. of Optical Communications and Networking, vol. 7, pp. 146-155, 2015.

[10] M. Ruiz *et al*., "Accurate and affordable packet-train testing systems for multi-Gb/s networks," IEEE Comm. Magazine, vol. 54, pp. 80-87, 2016.

[11] R. Leira *et al*., "High-speed optical networks latency measurements in the microsecond timescale with software-based traffic injection," Optical Switching and Networking, vol. 29, pp. 39-45, 2018.

[12] D. Rafique and L. Velasco, "Machine Learning for Optical Network Automation: Overview, Architecture and Applications," (Invited Tutorial) IEEE/OSA J. of Optical Communications and Networking, vol. 10, pp. D126-D143, 2018.

[13] R. Casellas *et al*., "Control, Management, and Orchestration of Optical Networks: Evolution, Trends, and Challenges," IEEE/OSA J. of Lightwave Technology, vol. 36, pp., 2018.

[14] L. Velasco *et al*., "Monitoring and Data Analytics for Optical Networking: Benefits, Architectures, and Use Cases," IEEE Network Magazine, vol. 33, pp. 100-108, 2019.

[15] F. Morales *et al*., "Dynamic Core VNT Adaptability based on Predictive Metro-Flow Traffic Models," IEEE/OSA J. of Optical Communications and Networking, vol. 9, pp. 1202-1211, 2017.

[16] Ll. Gifre *et al*., "Autonomic Disaggregated Multilayer Networking," IEEE/OSA J. of Optical Communications and Networking, vol. 10, pp. 482-492, 2018.

[17] L. Velasco *et al*., "Building Autonomic Optical Whitebox-based Networks," IEEE/OSA J. of Lightwave Technology, vol. 36, pp. 3097-3104, 2018.

[18] A. P. Vela *et al*., "Soft Failure Localization during Commissioning Testing and Lightpath Operation [Invited]," IEEE/OSA J. of Optical Communications and Networking, vol. 10, pp. A27-A36, 2018.

[19] L. Velasco *et al*., "Designing, Operating and Re-Optimizing Elastic Optical Networks," (Invited Tutorial) IEEE/OSA J. of Lightwave Technology, vol. 35, pp. 513-526, 2017.

[20] Focus group on Machine Learning for Future Networks including 5G, "Unified architecture for machine learning in 5G and future networks," Technical Specification ITU-T FG-ML5G-ARC5G, 2019.

[21] U. Bhat, *An Introduction to Queueing Theory: Modeling and Analysis in Applications*, Birkhäuser Basel, 2015.

[22] K. Han *et al*., "A partial differential equation formulation of Vickrey's bottleneck model, part I: Methodology and theoretical analysis", Transportation Research Part B: Methodological, vol. 49, pp. 55-74, 2013.

[23] M. Ruiz et al., "CURSA-SQ: A Methodology for Service-Centric Traffic Flow Analysis," IEEE/OSA J. of Optical Communications and Networking, vol. 10, pp. 773-784, 2018.

[24] M. Dischinger *et al*., "Characterizing residential broadband networks," in Proc. ACM Special Interest Group on Data Communication, 2007.

[25] A. Sivanathan *et al*., "Characterizing and Classifying IoT Traffic in Smart Cities and Campuses," in Proc. IEEE International Conference on Computer Communications Workshops, 2017.

[26] B. Villa and P. Heegaard, "Detecting Period and Burst Durations in Video Streaming by means of Active Probing," in Proc. International Conference on Computer Communication and Management, 2013.

[27] "Network functions virtualisation (NFV); Architectural framework," ETSI GS NFV 002 (V1.2.1), 2014.

[28] G. Casella and R. Berger, "Statistical Inference," 2nd ed., Duxbury/Thomson Learning, 2002.

[29] ESnet, "iPerf3: A TCP, UDP, and SCTP network bandwidth measurement tool," https://github.com/esnet/iperf, 2019.

[30] N. Finn *et al*. (Eds.), "DetNet Bounded Latency," IETF draft work-in-progress, 2019.