CURSA-SQ: A Methodology for Service-Centric Traffic Flow Analysis

Marc Ruiz, Franco Coltraro, and Luis Velasco

***Abstract*—The rapid availability of new services makes that network operators cannot exhaustively test their impact on the network or anticipate any capacity exhaustion. This situation will be worse with the imminent introduction of the 5G technology and the kind of totally new services that it will support. In addition, the increasing complexity of the network makes unreachable analyzing its behavior in front of the specific traffic that needs to be supported, which prevents from training human operators and much less, machine learning algorithms that might automatize network operation. In this paper, we present CURSA-SQ, a methodology to analyze the network behavior when the specific traffic that would be generated by groups of service consumers is injected. CURSA-SQ includes input traffic flow modelling with second and sub-second granularity based on specific service and consumer behaviors, as well as a continuous G/G/1/k queue model based on the logistic function. The methodology allows to accurately study traffic flows at the input and outputs of complex scenarios with multiples queues systems, as well as other metrics such as delays, while showing noticeable scalability. Application use cases include, packet and optical network planning, service introduction assessment, and autonomic networking, just to mention a few.**

***Index Terms*—Service-based traffic generation, logistic queue model, aggregated traffic models.**

I. Introduction

The advent of 5G networks will open the possibility to service providers to offer new services such as Video-On-Demand (VoD) contents in mobile devices with 4K ultra-high definition (UHD) and Virtual Reality [1]. Such exciting scenario will impose enormous challenges for network operators and vendors, since those new services will require stringent quality of service (QoS) from the network. To achieve these challenges, the adoption of optical technologies increasing the capacity and flexibility of fronthaul [2] and backhaul networks [3] is needed. However, before 5G deployment and service commercialization, the impact on the traffic injected to Multiprotocol Label Switching (MPLS)-over-optical metro and core networks need to be considered so they can be adequately planned.

Besides, the introduction of 5G will increase even more the complexity of multilayer networks, so they need to be complemented with new architectures for control and management to facilitate network operation and to evolve towards an autonomic networking paradigm [4]. Autonomic networking is based on three fundamental pillars that define the *observe–analyze–act* (OAA) loop [5]. Monitoring heterogeneous network elements (*observe*) produces huge amounts of metered data containing relevant information about network performance. Monitoring data is then processed by statistical and/or machine learning algorithms (*analyze*) [6] aiming at detecting and identifying some evidence requiring some further actions to be taken (*act*). Some examples include the reconfiguration of virtual network topologies following traffic changes [7]-[8] and the dimensioning of next planning steps based on traffic prediction [9]; both require the analysis of monitoring data to model and characterize network traffic.

Nonetheless, a crucial fact affecting the observe step is hindering the research on autonomic networking: no real monitoring data is available for the targeted networking scenarios. Incipient services to be supported by 5G network technologies limit the availability of real monitoring data to only what it can be obtained from test-beds, which, in most of the cases, do not represent those realistic scenarios that autonomic networking pursues. To overcome the lack of real monitoring data, analyzing *synthetically generated traffic data* becomes a requirement to train and test data analytics algorithms and to validate network optimization procedures before they enter into operation in a real network.

Trying to replicate the observed self-similarity and long-range dependency in packet network traffic [10], several theoretical models have been based on stochastic processes. These models can be used within *discrete-event simulators* [11] to generate discrete random input (*packet*) traffic propagated by a queue system that models the network under study. For instance, the authors in [12] presented a traffic generator for optical packet switching studies. This methodology achieves accurate traffic measurements for different single network elements (e.g., interfaces, links, connections), as well as additional traffic-related measurements, like end-to-end delay. However, traffic generation based on discrete stochastic processes requires a set of parameters to be fit, which entails having real traffic traces.

Looking at the literature, many research contributions related to queue systems make use of discrete-event simulation assuming M/M/1 queues [13], where arrivals follow a Poisson distribution and holding times are exponentially distributed. Even recent research works presenting hot topic networking use cases rely on such model [14]. However, these M/M/1 memoryless processes do not fit with the behavior of real dynamic traffic, as suggested in [15]; this can be extended to the traffic generated by 5G services.

Manuscript received April 3, 2018.

Marc Ruiz, Franco Coltraro, and Luis Velasco (lvelasco@ac.upc.edu) are with the optical communications group (GCO) at Universitat Politècnica de Catalunya (UPC), Barcelona, Spain.

For this very reason, current research effort is in producing models to generate synthetic realistic cellular traffic data for 5G networks and applications [16]. Consequently, discrete-event network simulation must evolve from service-agnostic M/M/1 models to more realistic and accurate G/G/1 ones, where both arrivals and departures are distributed following a wide range of statistical functions.

Nevertheless, the scalability remains a major issue; realistic simulations would consist in generating days or even weeks of traffic sourced by thousands of consumers in a network modelled as a system of queues that packets need to traverse. For instance, several weeks of traffic need to be analyzed to obtain accurate traffic models that would be required for traffic forecasting [8]. In consequence, it is required that execution time must be orders of magnitude smaller than simulation time, which is not typically achieved when huge amounts of packets need to be processed [17]. Therefore, the use of discrete-event simulation is clearly impractical.

Aiming at providing faster and more scalable approaches, the authors in [18] proposed a hybrid discrete-continuous fluid-flow approach that can considerably speed up the simulation of complex flow networks as compared to traditional queueing models. In fact, *continuous queue models* can be used to simulate G/G/1 queue systems. Among different models, the Vickrey's point-queue model [19] allows formulating an uncapacitated queue system as a differential equation that depends on input and output *traffic flows*. Although theoretically this continuous model scales much better than discrete ones based on packets, it has some additional limitations, such as *i*) the restriction of using infinite queues; *ii*) the impossibility to obtain packet-level measurements such as delay; and *iii*) the impossibility to use practical numerical methods for solving differential equations, such as ordinary differential equation (ODE) methods.

Assuming that a continuous queue system propagating flows instead of packets is adopted, a question still remains: how input flows entering the network need to be generated. Authors in [8] proposed a methodology to generate input traffic flows for distinct service types according to different daily patterns of expected traffic and distributions for the variance around that expectation. However, this approach makes difficult to generate profiles of new, incipient services, where there is no clear evidence on how the traffic generated by the service consumers behaves. In that case, a methodology to derive input traffic models as a function of additional information, such as the expected behavior of service consumers and the characteristics of the service, is needed.

In this paper, we propose a fast, accurate, attainable, and scalable *service-centric traffic flow analysis methodology* based on statistical flow characterization and continuous queuing models, named CURSA-SQ. Starting from the packet traffic generated by single service consumers, CURSA-SQ generates synthetic network traffic, as well as other related traffic variables resulting from the activity of consumers and providers of 5G services for a wide range of use cases.

The rest of the paper is organized as follows. Section II overviews the proposed CURSA-SQ methodology. Our proposal of queuing building blocks for synthetic traffic generation is presented in Section III. Moreover, a continuous queue model is formally defined, extending the Vickrey's point-queue model supporting capacitated queues and the use of ODE integrators for its computation. Input traffic flow modelling is detailed in Section IV by means of statistical formulation aiming at producing traffic models of flows aggregating multiple consumers. Finally, the proposed generation procedure is presented. CURSA-SQ numerical validation is presented in Section V and an illustrative application targeting the analysis of the impact of traffic evolution in a multilayer metro network scenario is eventually presented. Finally, Section VI concludes the paper.

II. Service-Centric Traffic Flow Analysis

In this section, we present a general overview of the CURSA-SQ methodology. Without loss of generality, let us consider a scenario where a network operator provides connectivity between *service consumers* and *service providers*. This can be extended to other scenarios like machine-to-machine communications, etc. Fig. 1 illustrates the scenario, where service is requested by the consumers; the *upstream* traffic arrives from service consumers in a network node that aggregates and forwards it toward the selected service provider, whereas in the *downstream* direction, such node forwards the traffic coming from a service provider (in response to service requests) to the specific service consumer.

We are interested in studying and generating traces of the aggregated traffic flows as a function of consumers traffic flows (hereafter, *input traffic*) and the characteristics of the network node (e.g., link capacity). To reduce the number of input traffic flows, we group consumers of the same type of service and with the same characteristics. For instance, consumers of a VoD service of a specific provider (e.g., Netflix) with 4K definition. Finally, a consumer group can be served from one or more locations of the same provider.

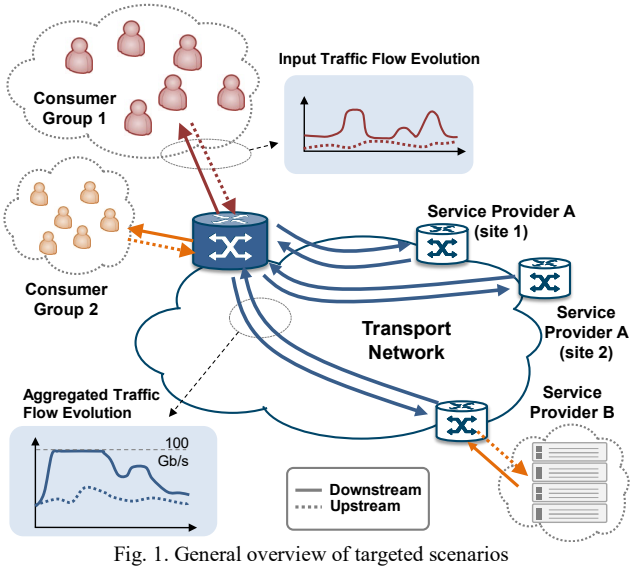

Fig. 1. General overview of targeted scenarios

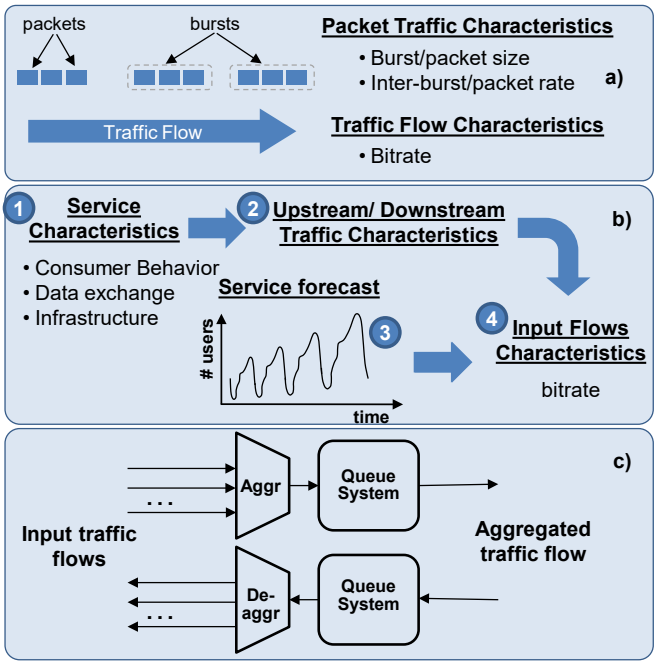

Fig. 2. Overview of the CURSA-SQ Methodology

In the absence of capacity constraints in the node, the aggregated traffic flows would follow a distribution as simple as just the summation of the input traffic. However, this scenario is not realistic in practice as a result of the limited capacity of the interfaces, the size of the buffers, etc. In consequence, the aggregated traffic will appear limited as it is suggested in Fig. 1.

Regarding the location of the network node, it does not need to be at the network edge; any intermediate location between consumers and providers can be subject of study even if upstream or downstream traffic is previously altered by capacity constraints, traffic shaping, etc., aa seen at the node.

We will use different traffic flow generators for upstream and downstream traffic. Those generators will generate traffic flows, in terms of bitrate, with granularity T fine enough to study flows (in the order of hundreds of milliseconds) but several orders of magnitude higher than those typical times and sizes of packet-based traffic generation (Fig. 2a). In the upstream direction, one single flow generator per consumer group will be used to produce the traffic flow for all the active consumers in the group; this flow generator will be located at the consumer group location and will target one or more service provider's sites. In the downstream direction, each service provider's site will contain a flow generator to produce the traffic flows toward the consumer groups.

The generation process is summarized in Fig. 2b; it is based on first characterizing each service (labeled 1 in Fig. 2b) to find the upstream and downstream traffic characteristics (2) for one single service consumer. Then, the traffic flow bitrate is generated by scaling the traffic characteristics to the number of active consumers forecasted for a given time period (3), while transforming the characteristics from the discrete to the continuous domain (4). The following groups of characteristics have been identified:

1. *Consumer behavior*: these characteristics capture the behavior of the consumers of a specific service. Assuming a VoD service, key characteristics of active consumers are the time between consecutive content reproductions, the duration of the content, and the completion rate of every content according to its duration [20].

2. *Data exchange*: these characteristics focus on how the service generates the data to be transferred according to consumers' activity. Continuing with the VoD service example, when a content reproduction is requested by a user, a certain amount of audio and video (media) is sent to the user to fill an initial buffer. After that, media segments of a given short duration (e.g., 5 sec.) are regularly sent following a typical ON/OFF pattern until the content finishes or the reproduction is stopped [21].

3. *Consumer infrastructure:* these characteristics allow adapting the data exchange to packet traffic since network infrastructure can impact the service. In a VoD service, video quality is adapted as a function of the throughput [22]. This could impact on the size of media segments and consequently, on the packet traffic characteristics.

The above service-related characteristics are not deterministic, but they follow statistical distributions. Therefore, by analyzing them, the packet traffic that every individual consumer introduces in the network can be modeled in terms of a few *random variables* capturing how bursts (and even packets) are generated by a single active consumer. The most relevant random variables are: *i*) *inter-arrival burst rate*, defined as the rate between consecutive bursts; *ii*) *burst size*, defined as the number of bytes transmitted in a burst; *iii*) *inter-arrival packet rate*, as the rate between consecutive packets in a burst; and *iv*) *packet size*, as the total amount of bytes (headers included) of a packet.

Once input traffic flows are generated in terms of bitrate for every period and every direction, they are used to generate aggregated traffic flows. To this end, a number of upstream input upstream traffic flows are aggregated, and the resulting flow feeds a queue system (Fig. 2c). The reverse process is followed in the downstream direction; the downstream input traffic flows are aggregated (not showed in the figure) and the resulting traffic flow enters a queue system; at the output, a disaggregator separates the resulting flow into the defined traffic flows.

The aggregator, queue system, and disaggregator in Fig. 2c are the building blocks that, by concatenating them, allow to study more complex problems, such as the one depicted in Fig. 3 targeting at modelling a packet switch in a typical MPLS-over-optical metro network. The switch consists of a number of access optical interfaces of a given speed (e.g., 10 Gb/s) connecting access networks, few high-speed (e.g., 100 Gb/s) optical metro interfaces, and several MPLS Label-Switched Paths (LSP). In the upstream, the consumers traffic flows are first aggregated and a queue system representing an access network ensures that the capacity of the switch

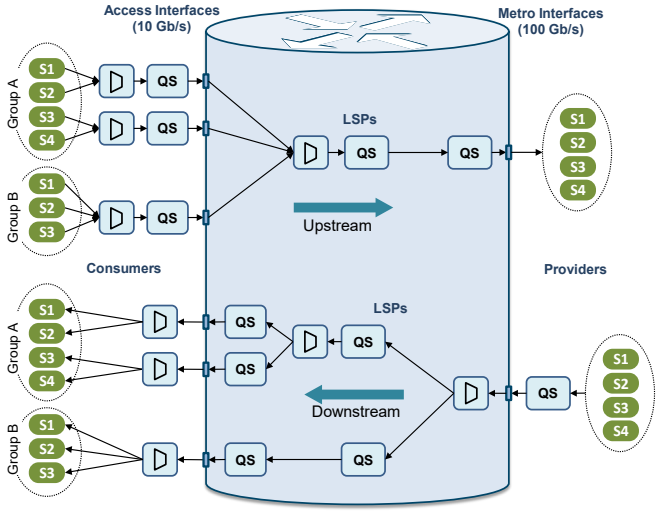

Fig. 3. Application example: Metro packet switch modelling

interface is not exceeded. Next, flows are aggregated into LSPs that are routed through the network. A queue system is used to ensure that the capacity of each LSP is not exceeded. Finally, LSPs leave the packet switch through metro interfaces, where another queue system enforces the capacity of the interface.

In the downstream, traffic flows from the service provider arrive through the metro interfaces, where queue systems model network devices between the service providers and the switch. Traffic is disaggregated into the configured LSPs (note that the LSP configuration is different in the upstream and the downstream to model asymmetric traffic). Finally, the traffic is disaggregated and sent through the access interfaces.

In the next section, we develop a general queuing module and queue model that supports the CURSA-SQ methodology. Although input flow generators are a crucial part to inject traffic in the proposed queue system, its formal development will be faced in the subsequent section together with the algorithm that exploits the proposed models.

III. CURSA-SQ Queue System and Model

In this section, we first define the general queuing module of the CURSA-SQ methodology, which is built with the above defined building blocks; then, we define our continuous queue model based on the logistic function.

A. Building blocks and queuing module

Traffic flow measured after every queue system is the result of the aggregation and propagation of traffic flows through the system under study, such as that in Fig. 3. In this subsection, we define a general *queuing module* that relates the aggregator, queue system, and disaggregator and allow keeping track of traffic flows.

To generalize the queuing module, let us consider a number n of input traffic flows that will be aggregated, queued, and finally disaggregated into m output traffic flows (Fig. 4). The traffic input flows are aggregated to generate the input flow X, which is converted to capacity units (bytes) and stored in a

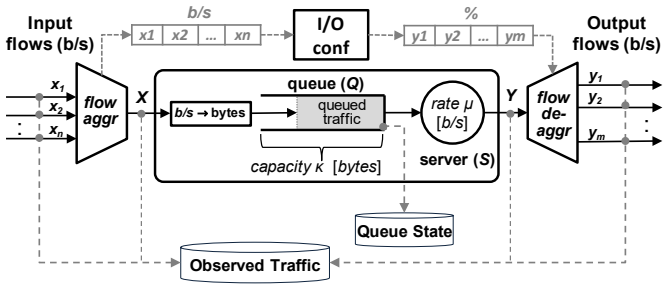

Fig. 4. Queuing module and building blocks

First-In-First-Out (FIFO) queue (Q) with capacity k bytes, provided that enough capacity is available in the queue. The server (S) processes queued data at a rate μ, which is configured according to the throughput of the element that the queue system models; e.g., if the queue system represents an interface, μ is the interface speed. According to the heterogeneity of input flows and server rates, we consider that the queue system follows a G/G/1/k model with FIFO discipline [13]. The flow Y leaving the queue system can be divided into m output traffic flows, to allow configuring the proportion of traffic to be forwarded to every single output; flow splitting is computed according to the measured magnitude of every input flow and the output configuration.

Finally, queuing modules store traces with granularity T of the traffic flows measured at every single input and output in a centralized repository. Additionally, queues' capacity usage is continuously monitored, aggregated and stored in a different repository for capacity and delay analysis purposes.

B. Logistic queue model

According to the queuing module scheme in Fig. 4, a continuous input flow X is processed by a capacitated queuing system to generate output flow Y. Let us consider that X is known in advance and it is defined in the time interval $[t_0, t_1]$; hence, $X(t)$ can be evaluated for any t in the interval. Before entering into the mathematical details of the proposed queue model, it is worth defining the used notation:

$X(t)$	Input bitrate (b/s) at time t
$Y(t)$	Output bitrate (b/s) at time t
$Q(t)$	Bytes in queue at time t
k	Queue capacity (bytes)
μ	Server rate (b/s)
Δt	Small time interval << T

For the sake of completeness, let us start from the *flow conservation equality* [23], i.e., the general relation between input and output flows of a queue system and the state of the queue, when a small time interval Δt is considered:

$$Q(t+\Delta t) = Q(t) + \frac{1}{8}\cdot\left(X(t) - Y(t)\right)\cdot\Delta t \qquad (1)$$

The queue state after Δt equals the queue state before Δt plus the difference between input and output flows during the interval. By considering the limit when $\Delta t \to 0$, we obtain the following differential equation $Q'(t)$, where the factor of 1/8 converts bits in a flow to bytes in a queue.

$$Q'(t)=\frac{dQ(t)}{dt}=\frac{1}{8}\cdot\left(X(t)-Y(t)\right)\cdot \tag{2}$$

Assuming that input flow $X(t)$ is known, the differential equation can be solved to obtain output flow $Y(t)$ as a function of both $X(t)$ and $Q(t)$. In addition, since we aim at modelling capacitated queues, some amount of input flow will be dropped when $Q(t)$ reaches its capacity.

The proposed capacitated logistic queue model starts from adapting the well-known Vickrey's point-queue model in [19]. Equation (3) presents a first approach based on that model, which characterizes a queue where the output flow is limited by the rate of the server. In case of an empty queue, $Y(t)$ equals $X(t)$ if the server rate is faster than $X(t)$; otherwise, output flow is constantly μ while the queue is not empty.

$$Y(t)=\begin{cases}\min\{\mu,X(t)\} & \text{if } Q(t)=0\\ \mu & \text{if } Q(t)>0\end{cases} \tag{3}$$

Note that eq. (3) presents a discontinuity, which makes solving eq. (2) computationally challenging since numerical methods such as ODE cannot be used. For this very reason, we propose a continuous formulation of the output flow based on an exponential function, as follows:

$$Y(t)=\mu+\left(\min\{\mu,X(t)\}-\mu\right)\cdot \mathrm{e}^{-8\cdot\lambda\cdot\frac{Q(t)}{\mu}} \tag{4}$$

Equation (4) keeps the main features of eq. (3), namely: *i*) when $Q(t)$ is empty, $Y(t)$ equals the minimum of μ and $X(t)$, and *ii*) if $X(t)$ is greater than μ, $Y(t)$ is fixed to μ. In addition, eq. (4) applies a smooth, exponential-based queue emptying when $X(t)$ is smaller than μ. The exponent depends on the current size of the queue; in particular, it is proportional to $Q(t)/\mu$, i.e., the expected emptying time of the current queue.

To illustrate the difference between the point-queue model in eq. (3) and the logistic-queue model in eq. (4), Fig. 5 presents an example of an input traffic flow with two plateaus, where the first one exceeds the capacity of the queue. Fig. 5a shows the input flow, as well as the output flows when each queue model is applied, whereas Fig. 5b shows the queue usage for each queue model. It is clear, in view of Fig. 5, that the logistic approach softens queue emptying without adding any other effect on the queue. To control the sharpness of the logistic function when the queue is almost empty, the exponent is weighted by parameter λ that can be computed as the expected average intensity, estimated as $\lambda\sim\mathrm{avg}(X)/\mu$.

The derived logistic model assumes an uncapacitated queue, which is not realistic in general. To guarantee that $Q(t)$ cannot exceed the queue capacity k, we propose to limit $X(t)$ as a function of $Q(t)$ based on the logistic function (see eq. (5)). The limited input flow, $\hat{X}(Q(t),t)$, approximates $X(t)$ when $Q(t)$ is far from k and tends to 0 as soon as $Q(t)$ approximates k.

$$\hat{X}\left(Q(t),t\right)=\frac{X(t)}{1+h\cdot \mathrm{e}^{\rho\cdot\left(Q(t)-k\right)}}. \tag{5}$$

Weighting coefficient h can be setup proportionally to the relation between the intensity of the aggregated input flow and μ, e.g., h=-(1-max(X)/μ). Parameter ρ adjust the slope of the descent when the queue is almost full; to guarantee a good convergence of ODE integrators, its value needs to be tuned to allow a fast descent with a smooth enough function.

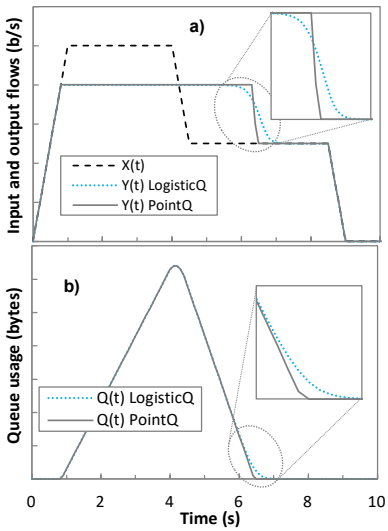

Fig. 5. Queuing module and building blocks.

Finally, eq. (6) presents the differential equation defining the capacitated logistic queue model and it can be solved by considering that at some starting time t_0 the queue had an initial value Q_0 (equal to 0 if the queue was empty). The differential equation is computed in the time interval $[t_0, t_{max}]$.

$$\begin{aligned}Q'(t)=\ & \frac{1}{8}\cdot\Big[\hat{X}\left(Q(t),t\right)-\\ & -\left[\mu+\left(\min\left\{\mu,\hat{X}\left(Q(t),t\right)\right\}-\mu\right)\cdot \mathrm{e}^{-8\cdot\lambda\cdot\frac{Q(t)}{\mu}}\right]\Big],\\ & t\in\left[t_0,t_{\max}\right]\quad where\ Q\left(t_0\right)=Q_0\end{aligned} \tag{6}$$

Let Q_0 and $\hat{X}$ be ≥ 0 and μ and k be > 0. Then, the system in eq. (6) satisfies the following properties: *i*) there exists a unique and positive solution for the defined time interval; *ii*) if $\hat{X}<\mu$ then, the queue gets empty exponentially fast; *iii*) a flow arriving in the system at t_1 will exit from it before other input flow arriving at $t_2 > t_1$ (FIFO property), and *iv)* $Q(t) \leq k$, $\forall t$ (capacitated queue). Proofs of the above properties can be found in [24].

IV. Input Traffic Flows and Traffic Analysis

Owing to the fact that even simple studies entail generating input flows that aggregate many service consumers and creating a complex system of queues to propagate traffic flows through the outputs, a meaningful part of the CURSA-SQ methodology is devoted to reducing the computational effort of generating large amount of fine granular traffic flows while ensuring the required accuracy. To this end, in this section we first propose statistical and mathematical models to generate aggregated input flows feeding the queue systems in practical

execution times. Next, the general CURSA-SQ methodology to generate traffic flows that takes advantage of the input flow generation and queue models is detailed.

A. Input traffic flow characterization

Four random variables were enumerated in Section II characterizing the traffic activity of one single consumer. From the perspective of a flow aggregating several individual active consumers, the effect of both packet size and inter-arrival packet rate variables can be neglected compared to burst size and burst inter-arrival rate. Since such traffic characteristics do not depend on the number of active consumers, the main source of input flow variations is precisely the evolution of consumers over time. Variations in the expected number of active consumers need to be modeled to capture any pattern, such as periodic behaviors (e.g., a daily pattern) or evolutionary trends (e.g., an annual increment).

With the above in mind, let us define the following random variables to model the traffic flow of a specific consumer group aggregating consumers of the same service:

- *ibr* Inter-arrival burst rate (s^{-1}), defined as the rate of consecutive bursts.
- *bs* Burst size (in bits for convenience)
- *r* Consumer maximum flow rate (b/s)
- γ Traffic burstiness degree
- $u(t)$ Number of active consumers at time *t*
- $x(t)$ Bitrate (b/s) generated by a consumer group or service provider site
- *T* Traffic generation granularity (s)

Since bitrate is expressed in b/s units, it seems natural to consider T = 1sec. as a reference. Aiming at supporting a wide range of statistical distributions and functions for traffic characterization and consumers time evolution, we consider a modelling approach based on computing approximations of the expectation (E) and variance (V) of $x(t)$ based on the expectation and variance of *ibr*, *bs*, and $u(t)$; these can be easily obtained assuming prior knowledge on service traffic random variables distribution and active consumers models. Note that the product of *ibr* and *bs* results into a new random variable representing the bitrate generated by one single user.

$E(x(t))$ can be approximated as the product of the expected number of users and the expected single user bitrate:

$$E\left(x(t)\right) \approx E\left(u\left(t\right)\right) \cdot E\left(bs \cdot ibr\right) = E\left(u\left(t\right)\right) \cdot E\left(bs\right) \cdot E\left(ibr\right) \quad (7)$$

Regarding the variance and assuming that *bs* and *ibr* are independent, the variance of the individual user bitrate can be derived according to well-known expressions to estimate the variance of the product of two independent variables [25]:

$$V\left(bs \cdot ibr\right) = V\left(bs\right) \cdot V\left(ibr\right) + E\left(bs\right)^2 \cdot V\left(ibr\right) + E\left(ibr\right)^2 \cdot V\left(bs\right) \quad (8)$$

Then, $V(x(t))$ can be approximated as the sum of the variance of individual users. According to the definition of a consumer group and the independence assumption, $V(x(t))$ can be estimated as:

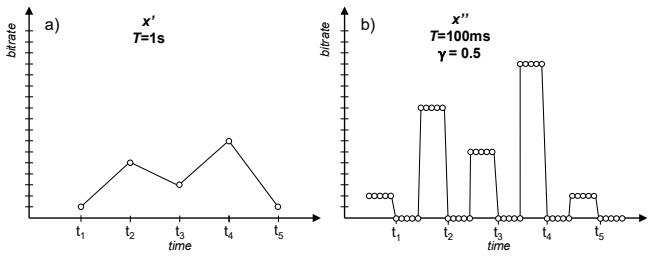

Fig. 6. Traffic generated with second (a) and sub-second (b) granularity

$$V\left(x(t)\right) \approx E\left(u\left(t\right)\right) \cdot V\left(bs \cdot ibr\right) \quad (9)$$

The model in eq. (7) and eq. (9) allows generating random traffic flows with the selected T. To that aim, a pseudo-random generator function ϕ following a given distribution, e.g., uniform, Gaussian, etc., can be used to generate random traffic $x'(t)$ according to $E(x(t))$ and $V(x(t))$.

$$x'(t) = \min\left\{u(t) \cdot r, \Phi\left(E\left(x\left(t\right)\right), V\left(x\left(t\right)\right)\right)\right\} \quad (10)$$

where $u(t) \cdot r$ is the maximum traffic that the consumer group can inject/receive due to access speed constraints.

Although eq. (10) works fine generating random traffic flows for $T \geq 1$ sec. traffic flows with sub-second granularity need to be generated to estimate queuing delays. Such sub-second scale generation must reproduce the nature of a bursty traffic with on-off periods producing short intervals of high activity that fill queues up.

To this aim, a flow $x''(t,i)$ with sub-second granularity is generated from $x'(t)$; index i represents the i-th interval T within the one-second interval centered in t. To allow computing maximum expected delays, a worst case of traffic bursty behavior is considered, as sketched in Fig. 6. Specifically, the example of $x'(t)$ flow in Fig. 6a is used to produce the $x''(t,i)$ with T=100 ms. in Fig. 6b; every bitrate sample in $x'(t)$ is transformed into 10 samples in $x''(t,i)$. Within every one-second interval, a first *on* period where bitrate can exceed that of $x'(t)$ is followed by an *off* period where bitrate is fixed to 0. Note that the summation of all samples in $x''(t,i)$ within one-second interval equals the bitrate in $x'(t)$. The number and magnitude of samples in the *on* period depends on the degree of burstiness γ of the traffic of the consumer group, and it is computed as follows:

$$\gamma = \frac{bs/r}{bs/r + 1/ibr}. \quad (11)$$

γ thus, represents the proportion of time within a second where traffic is actually generated. Then, the generation of random traffic samples with sub-second interval is defined as:

$$x''(t,i) = \begin{cases} \min\left\{u\left(t\right) \cdot r, \gamma^{-1} \cdot x'(t)\right\}, & T \cdot \sum_{j=0..i} x''(t,j) < x'(t) \\ 0, & T \cdot \sum_{j=0..i} x''(t,j) \geq x'(t) \end{cases} \quad (12)$$

Finally, it is worth noting that, if $T > 1$ sec., $x'(t)$ can be easily computed by averaging random samples generated with 1 sec granularity, whereas $x''(t,i)$ do not need to be computed.

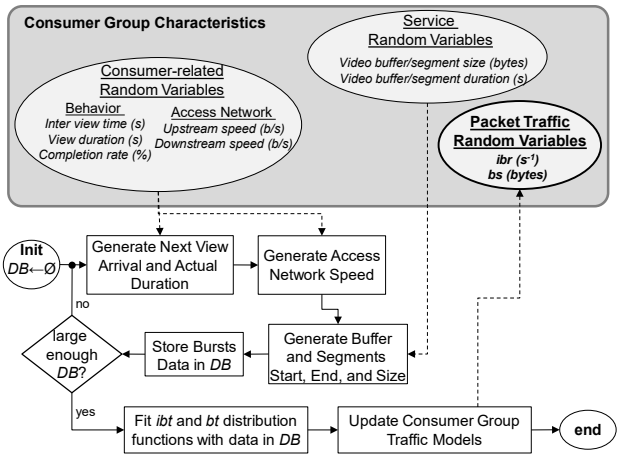

Fig. 7. CURSA-SQ methodology applied to VoD traffic analysis

B. *Traffic flow analysis*

With the input flow generation and queue models already presented, complex networking use cases can be reproduced. Then, we now focus on developing a general methodology for traffic flow generation using as guiding example the network scenario depicted in Fig. 3, where a metro packet switch connects consumers of different services in access networks to service providers located in the metro/core segment.

As presented in Section II, input flows characteristics depend on those of the consumer group. Fig. 7 sketches the procedure that can be followed to characterize traffic distributions from the consumer group characteristics. For the sake of clarity, the VoD service will be used as an example.

Random samples of the initial buffering and subsequent media segments will be generated according to both consumer-related and service-related characteristics; each sample consists of a tuple <*startTime*, *duration*, *size*>. Consumer-related characteristics are used to determine the starting time of a new video reproduction and the actual duration, considering not only video duration distribution but also completion rate. Moreover, upstream/downstream access speed is stochastically determined for the whole reproduction.

According to the previous values and the configuration of service-related characteristics, media segment samples are randomly generated and stored for further modelling purposes. As soon as enough samples to accurately fit traffic characteristics are available, *ibr* and *bs* statistical distributions are obtained by testing several known distributions and returning the one maximizing some common goodness-of-fit indicator such as the logarithm of the likelihood function [26].

Once burst traffic characteristics are available, traffic flow can start to be generated. In general, any scenario under study can be modeled as a number of unidirectional graphs, with three types of nodes: aggregators, disaggregators and queue systems. For instance, in the network scenario in Fig. 3, two unidirectional graphs (upstream and downstream) can be observed, where the path connecting consumers to providers and vice versa involves three queue systems (stages).

Table I presents the pseudocode of the traffic flow generation algorithm; the algorithm receives upstream (G_{up}) and downstream (G_{down}) graphs, a time window [*t_start*, *t_end*] of interest, and the granularity T. To avoid dealing with flows as large as the duration of the selected time window, short flows of duration $\delta \geq 1$s are generated and propagated. Note that, according to T, such flows will be either $x'(t)$ or $x''(t,i)$. After initializing the database where the traffic flows will be stored and computing δ as a function of T, all input flow generators are retrieved, and the availability of traffic models is checked before running consumer group traffic model procedures (lines 1-6). A traffic modelling procedure needs to be executed if a traffic model for the consumer group is not yet available or if some consumer-related and/or service-related characteristics have changed becoming thus an existing traffic model obsolete.

TABLE I TRAFFIC FLOW GENERATION ALGORITHM

INPUT G_{up}, G_{down}, *t_start*, *t_end*, T
OUTPUT *Traffic*

```
1:   Traffic ←Ø; δ←max(T,1s)
2:   nl_up, nl_down ←getNumStages(G_up, G_down)
3:   CG←getCGs (G_up, G_down)
4:   for each cg∈ CG do
5:     if trafficModelAvailable(cg) then continue
6:     fitTrafficModel(cg) (see Fig. 7)
7:   t←t_start
8:   while t < t_end do
9:     for G ∈{G_up, G_down}
10:      I_l ←generateCGFlows(getCGs(G),[t, t+δ], T)
11:      for l=1..(getNumStages(G)-1) do
12:        O_l, Q_l ← propagateFlows(G_l, I_l)
13:        storeFlowData(Traffic, I_l,O_l,Q_l)
14:        if l ≠ getNumStages(G)-1 then I_l+1 ← O_l
15:    t ← t+T
16:  return Traffic
```

Next, traffic flows with the required granularity are generated by propagating flows in both upstream and downstream directions (lines 8-15). To this aim, input flows of duration T are generated from the burst traffic statistical distributions following the input flow generation described in the previous subsection (line 10). Then, flows are propagated through the different stages and input and output traffic data, as well as queue state, are stored (lines 11-13). Note that the output of one stage is used as the input of the following one. Finally, when the time window is completed, the database with the traffic flows is retrieved (line 16).

V. NUMERICAL RESULTS

In this section, we first present the characteristics of the services that will be considered and then, numerically validate the CURSA-SQ methodology by comparing the results against traditionally packet-based generation and simulation. Finally, the illustrative application use case in Fig. 3 considering different types of services is used to generate examples of traffic analysis that can be carried out with the proposed methodology.

TABLE II SERVICES TRAFFIC CHARACTERISTICS

Service	$E(ibr)$ (s^{-1})	$V(ibr)$ (s^{-1})	$E(bs)$ (MB)	$V(bs)$ (MB)
VoD	0.25	2.54e-5	3.84	1.21
Gaming	1.33	0.19	0.14	0.02
Internet	1.66	0.40	0.12	0.04

A. Services characteristics

For the subsequent studies, we will consider three different services, namely: *VoD*, *Gaming*, and *Internet*. According to the CURSA-SQ methodology, relevant studies available in the literature providing consumer-related and service-related random variables characterization were used to characterize traffic sourced by consumer groups. Table II summarizes the expectation and variance of *ibr* and *bs* for all these services.

Let us detail the characterization of the VoD (recall the characteristics identified in Section II). Regarding consumer behavior, according to the study presented in [20], the idle time y that an active user spends (e.g., deciding which content to watch) follows the power law probability distribution $p=\alpha \times y^{-\beta}$, with parameters α=0.43 and β=1.2. On the other hand, the duration of the content selected by a user approximates an exponential distribution with a typical mean around 30 minutes and a reasonable maximum of 4 hours [27]. However, users usually stop a reproduction before its completion time. Completion rate depends on the content duration; the longer the duration is, the smaller the completion rate. A Weibull distribution with scale and shape parameters around 75 and 0.8 fits with a large variety of contents' duration [20]. Regarding service-related VoD characteristics, we adopt a typical on-off pattern consisting of an initial 10-20 sec transmission of media contents, followed by a number of 2 sec media segments, until the reproduction finishes [21]. According to the previously defined statistical distributions, we simulated the activity of a single consumer and stored the time stamp and size of 10.000 traffic bursts. The analysis of this data lead to the VoD consumers traffic characteristics detailed in Table II, that indicates long spaced bursts of large number of bytes.

A similar procedure was followed to characterize gaming and Internet consumers' traffic from key statistical distributions detailed in [28]-[30]. The resultant traffic characteristics differ from that of VoD in both, the frequency of bursts (high *ibr*) and its size (small *bs*). Note that Internet traffic is the one that shows the highest variance in terms of *ibr*, which translates into a less regular traffic pattern.

B. Validation results

Aiming at validating the CURSA-SQ methodology including the aggregated input traffic flow model and the logistic queue model, we developed a *packet-based* simulation environment for benchmarking purposes. Specifically, a packet input traffic generator produces packets streams creating of a fixed size creating 1500-byte Ethernet frames, according to the specific mean and variance of *ibr* and *bs*; a packets stream is generated independently for each individual user. Then, the aggregated packets stream is sent to a simple queue system with one *discrete* queue, which processes packet by packet. This combination of packet-based traffic generation and discrete queue simulation provides the baseline performance for comparison purposes.

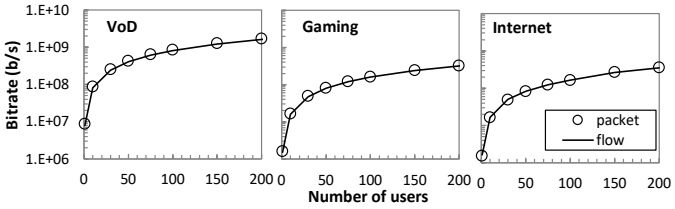

Fig. 8. Input traffic vs. users

TABLE III RELATIVE ERRORS OF AGGREGATED TRAFFIC FLOWS

	VoD		Gaming		Internet	
users	*mean*	*max*	*mean*	*max*	*mean*	*max*
10	6%	57%	4%	14%	4%	15%
50	5%	34%	2%	5%	3%	4%
100	4%	15%	2%	2%	2%	3%
200	4%	10%	1%	1%	2%	2%

The CURSA-SQ methodology and the discrete simulator were implemented in Python 2.7 and executed in an Intel i7-4790K -based computer with 16 GB RAM running Ubuntu 16.04.3 LTS.

For each defined service, we considered a scenario with a single consumer group configured with a constant number of users. For the sake of a fair comparative analysis, we run several executions with incremental number of users. Every execution generated a random flow of one day long and T = 1sec. according to eq. (10) that was used for input flow comparison purposes. Then, a sub-second flow with T = 50ms was generated according to eq. (12) to evaluate the performance of the logistic queue model; both discrete and logistic queues were configured with a 10 Gb/s server.

Fig. 8 shows the average bitrate of the traffic flows of each consumer group against the number of users, using flow-based and packet-based generation. As shown, flow-based generation accurately matches the correlation between generated bitrate and number of users that packet-based generation produced. A detailed accuracy analysis is presented in Table III, where mean and maximum errors of flow-based generation w.r.t. packet-based generation are detailed for every service and different number of users. Mean errors are not higher that 6%, whereas maximum error remarkably decreases with the number of users, reaching no more that 15% in the worst case (for the VoD service) when 100 or more users are considered. Note that gaming and Internet services experience maximum errors not higher that 15% even with 10 users. In light of these results, the accuracy of the proposed statistical methodology to generate aggregated input flows is validated assuming scenarios with a medium/high number of consumers per group.

A comparison between discrete and logistic queues is

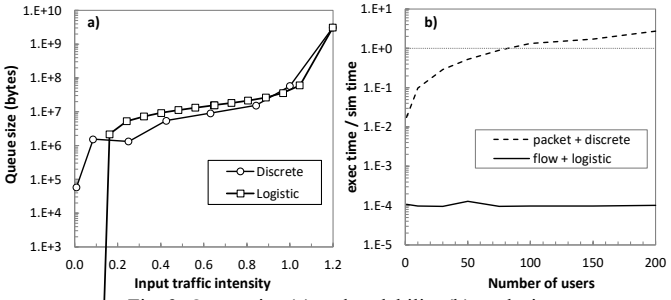

Fig. 9. Queue size (a) and scalability (b) analysis

shown in Fig. 9a for downstream VoD traffic. In Fig. 9a, the maximum queued traffic is plot as a function of the traffic intensity, computed as the quotient between the average of the aggregated input flow and the speed of the queue server. Note that when the traffic intensity is under about 0.15 the logistic queue is unable to reproduce the behavior of the discrete queue, as for low traffic intensities the discrete behavior becomes more dominant. However, for the scenarios of interest entailing a meaningful traffic intensity, queued traffic evolves similar in both cases, which entails a key numerical evidence to validate not only the logistic queue model but also the procedure to generate aggregated input flows with sub-second granularity.

Looking at analyzing the scalability of both packet-based and flow-based approaches, Fig. 9b presents the total execution time (input flow generation plus queue simulation) as a function of the number of consumers aggregated in the flow. For illustrative purposes, execution time is presented relative to the simulated time, so a value equal to 1 entails simulating the same amount of time that is needed for running the simulation (e.g., 1 day of simulation takes 1 day of execution). As it can be observed, CURSA-SQ runs in few seconds independently of the number of users; this contrasts with the packet-based approach, which execution time is dependent on the number of users and few orders of magnitude larger than that of CURSA-SQ. In addition, the packet-based approach is not practical when a large number of users need to be considered, as its execution time exceeds the simulated time.

In light of the previous results, we can conclude that the proposed CURSA-SQ methodology leads to similar results in terms of flow characteristics and queue behavior that the classical packet-based flow generation and discrete queue simulation, and with excellent scalability. In consequence, this methodology can be used to generate traffic for network analysis purposes in complex scenarios.

C. Illustrative use case: QoS evolution analysis

Let us now apply the CURSA-SQ methodology on the example of the packet switch in a MPLS-over-optical metro network in Section II; specifically, on the downstream direction, depicted in Fig. 10 for convenience, where optical interfaces are labeled. In this example, we consider the VoD, gaming and Internet services with the traffic characteristics in Table II, as well as a new 4K UHD VoD service, which doubles the number of bytes transmitted per burst w.r.t. the standard VoD service. The expected evolution in the number of users of each of the services is represented in Fig. 11a.

According to expected consumer evolution, we generated daily traffic during the period under analysis, measuring the traffic at every interface. Such traffic traces can be used for many purposes, like to train and validate machine learning algorithms for autonomic networking. In this paper, let us exemplify its use for network planning, particularly to anticipate when and where capacity exhaustion will seriously limit the network performance.

Fig. 11b shows the average traffic at the peak hour for every interface, where it can be observed how the traffic in interface II reaches its 10 Gb/s capacity in 2019 Q4, so network planning decisions (e.g., upgrading optical interface II to 40Gb/s or adding a new interface together with optical connectivity towards the access, etc.) should be made before that date. This illustrates the application of the proposed CURSA-SQ methodology as a tool for network planners.

Let us now analyze the traffic between the service provider and consumer group A during a typical day before interface II capacity is exceeded, e.g., in 2019 Q2; Fig. 12a shows traffic monitored at every interface. The correlation between the traffic measured at each interface is guaranteed by the queue system that models the switch; the traffic queued at each interface is shown in Fig. 12b. Note that the combination of different interface speed and service mix in the flows leads to completely different behavior on the queued traffic. In line with the interface capacity analysis presented before, interface II is the one that shows largest queued traffic. Nonetheless, interface IV shows some queued traffic peaks during evenings.

With the data generated before, extended network performance analysis can be done, e.g., for examining the delay of traffic flows from metro to access. Fig. 12c presents the maximum delay of two paths, namely from metro interface

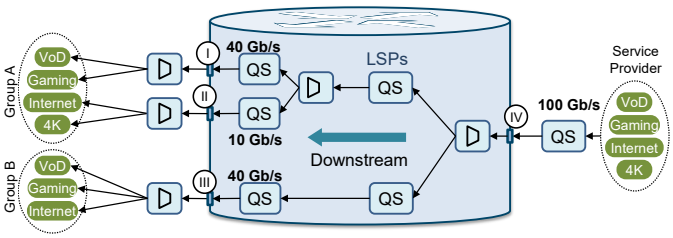

Fig. 10. Consumers evolution scenario

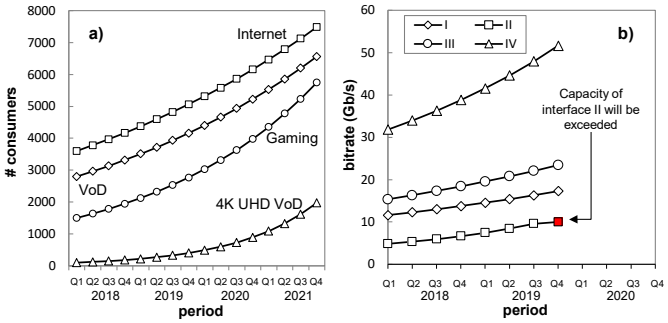

Fig. 11. Consumers (a) and capacity (b) evolution.

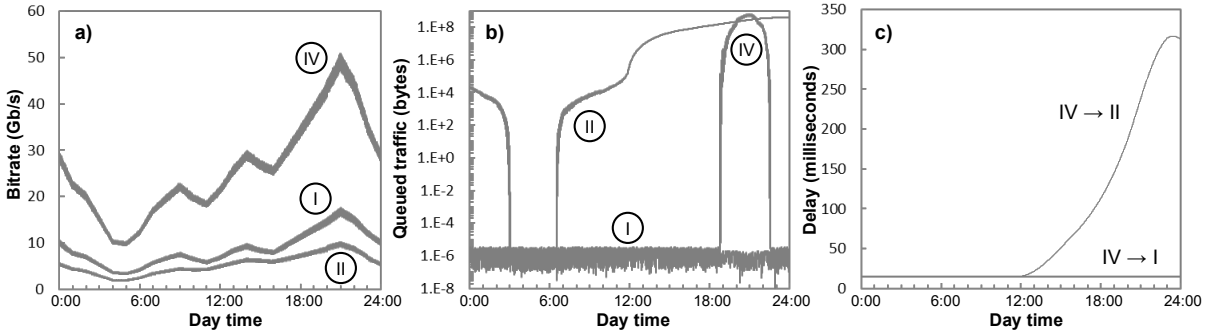

Fig. 12. Bitrate (a), queued traffic (b), and delay (c) for a single day in 2019 Q2.

IV to access interface I (IV->I), and from metro interface IV to access interface II (IV->II). As shown, path IV->I experiences an almost constant delay, even when interface IV experiences a peak of queued traffic during night hours. In this case, queue usage is very small (hundreds of MB) compared to the speed of the interface (100 Gb/s), which minimizes the impact of queued traffic on the accumulated delay. Conversely, queue usage for interface II entails a clear impact on the delay in path IV->II, reaching a maximum value of 300ms, which is evidence of the need for planners to make some decision to support the expected demand increase.

VI. Concluding Remarks

The CURSA-SQ methodology has been proposed to generate accurate synthetic traffic flows based on service characteristics and consumers behavior, and to analyze its impact on the network infrastructure. Input traffic flow modelling was statistically formulated aiming at producing traffic models of flows aggregating a number of consumers, where second and sub-second granularities were considered. In addition, a continuous queue model was formally defined, being a key component to create queuing systems.

The numerical validation of the CURSA-SQ methodology showed high accuracy and extraordinary scalability, compared to the classical packet-based generation and simulation. Once validated, the CURSA-SQ methodology was applied on an illustrative application use case focused on a metro switch scenario, where traffic from different types of services were generated. Useful studies for network planning purposes were carried out, analyzing capacity exhaustion in the interfaces to anticipate upgrading processes, as well as delay as an indicator of the QoS of the provided services.

Finally, just to mention that the range of applications of the CURSA-SQ methodology is remarkable and interestingly includes the use of traffic traces to train and validate machine learning algorithms for autonomic networking scenarios.

Acknowledgements

The research leading to these results has received funding from the European Community's through the METRO-HAUL project (G.A. n° 761727), from the AEI/FEDER TWINS project (TEC2017-90097-R) and from the Catalan Institution for Research and Advanced Studies (ICREA).

References

[1] Huawei, "Best-UHD Industry Development White Paper," [On-line] http://www.bpl-business.com/ pdf/ exhibitors/ huawei/ Best-uhd-industrydevelopment-white-paper.pdf, 2016.

[2] L. Velasco *et al*., "Meeting the Requirements to Deploy Cloud RAN over Optical Networks," IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 9, pp. B22-B32, 2017.

[3] M. Ruiz *et al*., "Planning Fixed to Flexgrid Gradual Migration: Drivers and Open Issues," IEEE Comm. Magazine, vol. 52, pp. 70-76, 2014.

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

[5] Ll. Gifre, J.-L. Izquierdo-Zaragoza, M. Ruiz, and L. Velasco, "Autonomic Disaggregated Multilayer Networking," IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. 482-492, 2018.

[6] D. Rafique and L. Velasco, "Machine Learning for Optical Network Automation: Overview, Architecture and Applications," (Invited Tutorial) IEEE/OSA Journal of Optical Communications and Networking (JOCN), 2018.

[7] F. Morales *et al*., "Virtual Network Topology Adaptability based on Data Analytics for Traffic Prediction," (Invited) IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 9, pp. A35-A45, 2017.

[8] F. Morales, Ll. Gifre, F. Paolucci, M. Ruiz, F. Cugini, P. Castoldi, and L. Velasco, "Dynamic Core VNT Adaptability based on Predictive Metro-Flow Traffic Models," IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 9, pp. 1202-1211, 2017.

[9] L. Velasco, F. Morales, Ll. Gifre, A. Castro, O. González de Dios, and M. Ruiz, "On-demand Incremental Capacity Planning in Optical Transport Networks," IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 8, pp. 11-22, 2016.

[10] W. Leland, M. Taqqu, W. Willinger, D. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," IEEE/ACM Transactions on Networking, vol. 2, pp. 1-15, 1994.

[11] G. Fishman, "Discrete-Event Simulation: Modeling, Programming, and Analysis," Springer Series in Operations Research and Financial Engineering, 2001.

[12] F. Xue and S. J.B. Yoo, "On the Generation and Shaping Self-similar Traffic in Optical Packet-switched Networks," OPNETWORK, 2002.

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

[14] M. Raza *et al.*, "Dynamic Slicing Approach for Multi-Tenant 5G Transport Networks [Invited]," IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. A77-A90, 2018.

[15] K. Gaizi, F. Abdi, and F. Abbou, "Realistic dynamic traffic generation for WDM Optical Networks," in Proc. ISSC, pp. 1-4, 2016.

[16] F. Malandrino, C. Chiasserini, and S. Kirkpatrick, "Cellular Network Traces Towards 5G: Usage, Analysis and Generation," in IEEE Transactions on Mobile Computing, vol. 17, pp. 529-542, 2018.

[17] R. M. Fujimoto *et al*., "Large-scale network simulation: how big? how fast?" in Proc. MASCOTS, pp. 116-123, 2003.

[18] B Melamed, S Pan, Y Wardi, “Hybrid Discrete-Continuous Fluid-Flow Simulation,” in Proc. SPIE, vol. 4526, pp. 263-270, 2001.

[19] K. Han, T. L. Friesz, T. Yao, “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.

[20] L. Huang, B. Ding, Y. Xu, and Y. Zhou, “Analysis of User Behavior in a Large-Scale VoD System,” in Proc. NOSSDAV, 2017.

[21] A. Rao *et al*, “Network characteristics of video streaming traffic,” in Proc. CoNEXT, 2011.

[22] H. Azwar and Hendrawan, “H.265 video delivery using dynamic adaptive streaming over HTTP (DASH) on LAN network,” in Proc. International Conference on Telecommunication Systems Services and Applications (TSSA), 2014.

[23] T. Konstantopoulos, M. Zazanis, and G. De Veciana, “Conservation laws and reflection mappings with an application to multiclass mean value analysis for stochastic fluid queues,” Stochastic Processes and their Applications, vol. 65, pp. 139-146, 1996.

[24] F. Coltraro, “A logistic queue model for network traffic modelling and simulation,” M.Sc. Thesis, Universitat Politecnica de Catalunya, 2017. https://upcommons.upc.edu/handle/2117/106777.

[25] G. Casella and R. Berger, “Statistical Inference,” 2nd ed., Duxbury/Thomson Learning, 2002.

[26] J. Rao and C. Wu, “Empirical Likelihood Methods,” Handbook of Statistics, vol. 29, Elsevier, 2009.

[27] Y. Choi, J. Silvester, and H. Kim, “Analyzing and Modeling Workload Characteristics in a Multiservice IP Network,” in IEEE Internet Computing, vol. 15, pp. 35-42, 2011.

[28] W. Feng *et al*, “A Traffic Characterization of Popular On-Line Games,” IEEE/ACM Transactions on Networking, vol. 13, pp. 488-500, 2005.

[29] D. Drajic *et al*., “Traffic generation application for simulating online games and M2M applications via wireless networks,” in Proc. WONS, pp. 167-174, 2012.

[30] X. Wu *et al*., “Packet size distribution of typical Internet applications,” in Proc. ICWAMTIP, pp. 276-281, 2012.