Near Real-Time Estimation of End-to-End Performance in Converged Fixed-Mobile Networks

Alvaro Bernal [(1)], Matias Richart [(2)], Marc Ruiz [(1)], Alberto Castro [(2)], and Luis Velasco [(1)*]

[(1)] Universitat Politècnica de Catalunya (UPC), Barcelona, Spain.
[(2)] Universidad de la República, Montevideo, Uruguay
*Corresponding author: lvelasco@ac.upc.edu

***Abstract*—The independent operation of mobile and fixed network segments is one of the main barriers that prevents improving network performance while reducing capital expenditures coming from overprovisioning. In particular, a coordinated dynamic network operation of both network segments is essential to guarantee end-to-end Key Performance Indicators (KPI), on which new network services rely on. To achieve such dynamic operation, accurate estimation of end-to-end KPIs is needed to trigger network reconfiguration before performance degrades. In this paper, we present a methodology to achieve an accurate, scalable, and predictive estimation of end-to-end KPIs with sub-second granularity near real-time in converged fixed-mobile networks. Specifically, we extend our CURSA-SQ methodology for mobile network traffic analysis, to enable converged fixed-mobile network operation. CURSA-SQ combines simulation and machine learning fueled with real network monitoring data. Numerical results validate the accuracy, robustness, and usability of the proposed CURSA-SQ methodology for converged fixed-mobile network scenarios.**

***Keywords*—Converged Fixed-Mobile Networks; Real-Time KPI estimation; Shared Medium Modelling**

I. INTRODUCTION

Fixed-mobile networks have been traditionally operated as two separated network segments, where the Evolved Packet Core (EPC) in the Radio Access Network (RAN) facilitates the mobility of User Equipment (UE) and provides Quality of Service (QoS) looking at meeting the needs of services and users with diverse characteristics, whereas the fixed network provides connectivity services among Evolved NodeB (eNB) / Next Generation NodeB stations and with the mobile core [1], [2]. Although this separation simplifies network operation as the RAN assumes that enough *resources* (i.e., connections with fixed capacity) are allocated in the fixed network, it imposes resource overprovisioning to the fixed network; enough resources need to be allocated in the fixed network trying to avoid network congestion that would degrade the QoS perceived by the end-users (i.e., increased end-to-end delay and reduced throughput). Note that overprovisioning increases network capital expenditures (CAPEX). However, since the fixed networks have been traditionally able to easily meet the requirements of 2G/3G/4G mobile networks, network operators have not paid much attention to such overprovisioning.

Nonetheless, large traffic variations can be expected not only in the RAN but also in the fixed network as a result of the increment of the bitrate available in the RAN, the different type of services (e.g., video streaming, P2P, gaming, and so on), and the mobility of UEs. Note that: *i*) the number of active UEs in a given cell fluctuates not only at macroscopic scale (daily) but also at microscopic one (second) according to complex behavioral aspects [3]; *ii*) the traffic generated by the different services is not constant, as non-deterministic on/off patterns are commonly observed [4]; and *iii*) UEs' mobility, including within the same cell and among neighboring cells, impacts on the end-to-end latency and throughput in both upstream and downstream directions [5]. Such traffic variations push the amount of resources to be overprovisioned in the fixed network.

In addition, the stringent requirements imposed by 5G is making that fixed networks need to be redesigned while fostering the convergence of mobile and fixed networks, where the optical transport network is extended toward the edge [6]. In this regard, operators are attending with significalt interest to the definition of the next-generation cell site Gateway (CSGw) connecting current and upcoming 5G mobile cell sites, to the transport network [7]; the CSGw includes, among others, Multiprotocol Label Switching (MPLS) capabilities, so traffic engineering techniques can be applied to *packet flows*. Note that MPLS improves the routing and increases the traffic engineering possibilities and it can be used to implement the data plane, e.g., to support the S1 interface [8].

In this context, looking at limiting CAPEX derived from the required overprovisioning, the convergence of mobile and fixed networks needs to be complemented with some level of coordination at the control plane. In fixed networks the Software-defined Network (SDN) is already consolidated and it is able to control the network not only for connectivity provisioning but also to re-configure the allocated resources in response to network conditions. In contrast, the separation of control and user planes in the RAN is currently under research looking at controlling eNBs and enabling the deployment of specific policies on top of a centralized SDN controller (see, e.g., [8]-[11]).

Although an SDN controller could be in charge of the RAN, for the sake of simplicity from the overall network orchestration perspective (which it is currently under definition [12]), in this paper, we assume that an SDN controller is in charge of the fixed network, whereas the EPC in the mobile core provides policy control in the RAN. The SDN controller can re-configure the allocated connectivity resources in the fixed network as a function of the needs of the cells in the RAN, targeting at ensuring the desired QoS, while avoiding resource overprovisioning. To this end, real-time Key Performance Indicators (KPI) and resource utilization monitoring are needed to evaluate the QoS and detect bottlenecks in the converged network [13]. Those KPIs (latency and throughput) measured end-to-end, i.e., from UEs to the mobile core, and vice versa, in addition to packet loss measured in the access-metro nodes in the fixed network, can be of paramount importance for an optimal operation oriented to guarantee QoS, as defined by those end-to-end KPIs, with efficient resource usage. In the case some degradation is detected, the SDN controller could re-configure the fixed network to adapt resources to current network conditions.

The above approach is purely *reactive* as it consists in following the changes in the network conditions, so it would be desirable to anticipate (near) future conditions, so that the adaptation can be performed in a *proactive* manner. In that regard, Machine Learning (ML) techniques [14] can be used to make predictions about future traffic and position of UEs [15]. In fact, specialized monitoring and data analytics (MDA) architectures that include a MDA controller running besides the SDN controller have been proposed to collect monitoring data from the network devices, analyze such data by means of ML-based algorithms, and issue recommendations to the SDN controller in the case of detecting some degradation (see, e.g., [16]-[18]). In addition, ML algorithms have been proposed to reconfigure the virtual network topology (VNT) based on predicted traffic [19], to detect packet traffic anomalies [20], and to compose aggregated traffic models based on a set of models for individual traffic flows [21]. However, such mobility and traffic predictions are not enough to identify bottlenecks in the network and predict the QoS perceived by the users. Although such analysis can be currently performed using discrete-event based network simulation (e.g., the well-known *ns-3* open source network simulator [22]), its use for network operation can be discarded due to scalability issues.

Given this, we propose a tool that uses traffic prediction and UEs' mobility as inputs to compute future network conditions and estimate QoS-related end-to-end KPIs for the current network configuration. In this respect, in our previous work in [23], we proposed a methodology named CURSA-SQ to analyze traffic flows in a fixed network by modelling service traffic and the behavior of the queues in packet nodes. CURSA-SQ enables near real-time traffic analysis (with sub-second granularity) due to its better performance and scalability compared to traditional discrete-event based simulations. Starting from the general CURSA-SQ methodology, in this paper, we present the needed extensions to enable its application in converged fixed-mobile network scenarios, where each cell in the RAN is modeled as a shared medium controlled by the cell's scheduler. Specifically, the contributions are:

- Section II introduces the converged fixed-mobile network scenario and motivates the need of end-to-end KPI's estimation as the key to verify the performance of services from UEs to applications running in datacenters connected to the access-metro and core fixed network, as well as to the Internet. Our proposal for CURSA-SQ as the tool able to produce such estimation is overviewed, where its output can be used by a performance analysis module to issue recommendations that can help to proactively reconfigure resources in the network. However, for such estimation to be useful for the network operation, it needs to be near real-time, and therefore, the considered estimation window should not be larger than some (few) minutes and be ready in a shorter time, which imposes stringent requirements.

- Our proposal to extend the general CURSA-SQ for shared medium and mobility is presented in Section III, together with the computation of end-to-end KPIs in networking scenarios. Once KPIs are estimated, a *performance analyzer* module can determine whether the required QoS will be met, identify bottlenecks and issue recommendations to the SDN controller or the mobile core to *anticipate* any degradation. Finally, the SDN controller could properly re-configure the available resources to ensure that those requirements are met.

- Section IV focuses on the configuration of CURSA-SQ that needs to be carried out before every simulation. Among others, traffic disaggregation and projection are key elements that produce the needed input traffic for the simulation. In the output, the results of CURSA-SQ need to be evaluated against real measurements to detect significant deviations.

The discussion is supported by the simulation results presented in Section V, where the results from CURSA-SQ are validated against those of the ns-3 network simulator for a pure mobile network scenario. In addition, a sensitivity study is carried out to analyze the dependence of the CURSA-SQ simulation against errors in the traffic prediction and configuration of the simulation. Three interesting scenarios are eventually configured on a realistic converged fixed-mobile network to illustrate the usefulness of the proposed approach to simulate network conditions that can be used afterward by a performance analysis module to anticipate performance degradation and identify bottlenecks in the network.

II. ESTIMATING END-TO-END KPIS IN A FIXED-MOBILE NETWORK

In this section, we overview our proposal to extend CURSA-SQ, as well as various other modules needed to evaluate end-to-end KPIs in converged fixed-mobile network scenarios.

Fig. 1a illustrates the considered fixed-mobile network scenario, where eNBs in the RAN are connected to packet nodes in the fixed access-metro network through CSGws. In the control plane, we assume an SDN controller in charge of the access-metro network that includes CSGws and packet nodes, as well as an MDA controller collecting monitoring data from the access-metro nodes. To support the S1 interface between eNBs and the mobile core, MPLS tunnels can be set-up in the access-metro network to facilitate traffic flow management (see [8]).

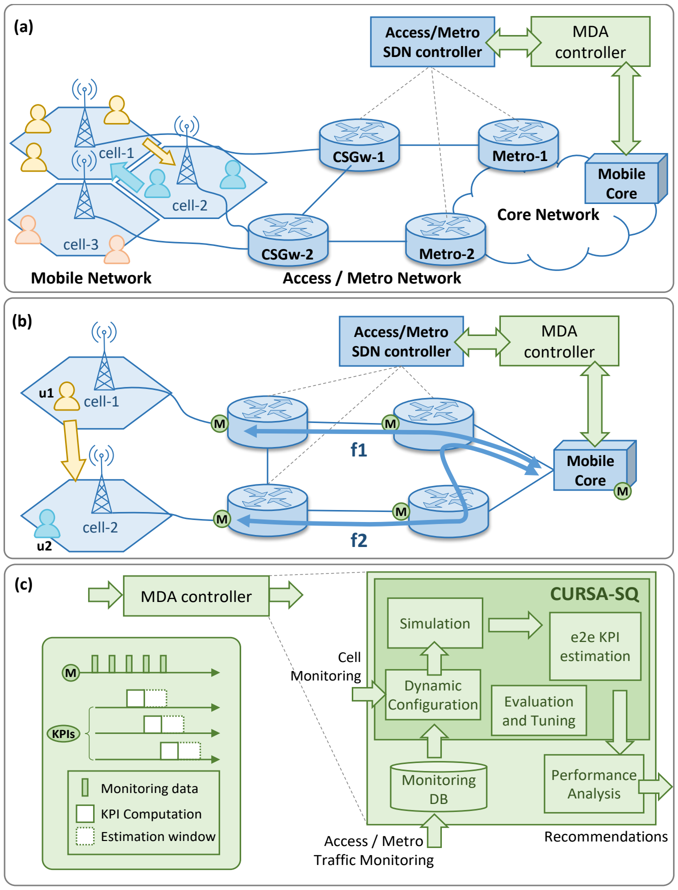

Fig. 1 Converged fixed-mobile network (a), mobility use case (b), and near real-time KPI estimation workflow (c).

Aiming at enabling near real-time access-metro and mobile KPI estimation, monitoring data collected continuously from network devices can be used for analysis purposes. However, variations in the input traffic to the access-metro network due to users' activity and mobility requires data that is available in the EPC with a full view of the cells. In addition, the evaluation of mobile services requires analyzing the behavior of the access-metro network. ML models can be fed with these data to forecast relevant variables, e.g., the number of active UEs, their position, and their mobility among cells within the next short time window (e.g., next 1-2 minutes). With such prediction, as well as with some known deterministic network parameters, CURSA-SQ can be used to simulate network conditions in a future time window and evaluate KPIs in every networking device, as well as end-to-end. Such evaluation, together with some recommendations, can be of paramount importance for the fixed and the mobile network operation.

For illustrative purposes, a case of mobility is represented in Fig. 1b. Let us consider that a group of UEs in cell-1 (labeled *u1*) moves towards the neighboring cell-2 where, among others, a second group of UEs (*u2*) is located. For the sake of simplicity, let us assume that users in both groups consume the same service, e.g., a P2P service generating large *symmetric traffic* between both groups. Two traffic flows are highlighted (*f1* and *f2*) connecting the CSGws to the mobile core. In the example, *u1* throughput would decrease and latency would increase as a consequence of cell handover. In addition, *u2* would also experience such an effect in case *u1* mobility would cause congestion in cell-2. Analyzing traffic flows, an increase in traffic flow *f2* would produce congestion and thus traffic loss if the capacity allocated to the MPLS tunnels supporting that flow is reached. CURSA-SQ produces outputs to the performance analysis module, which can issue recommendations to the SDN controller and the mobile core to manage resource allocation and traffic engineering policies. As an example, in view of the predicted congestion and traffic loss, the performance analysis module can issue a recommendation to the access-metro SDN controller advising to increase capacity allocated to MPLS tunnels.

Our proposal for near real-time KPI estimation is presented in Fig. 1c. Monitoring data collected from the access/metro SDN controller, as well as the UEs' cell assignment and throughput from the mobile core is stored in the MDA controller repository. Such data is used to estimate KPIs in the access/metro network for the next time window. Note the overlap between the computation period needed to estimate the KPIs for the next time window and the period of validity of the previous window.

Although measurements of the amount of bitrate entering every interface of a node in the access-metro network can be provided (in particular, those connecting the eNBs in the RAN), they are not enough to compute per-service and per-UE KPIs with enough accuracy. In fact, as such measurements would entail installing expensive deep packet inspection (DPI) devices to examine the contents of every packet entering the access-metro network, we assume that no DPI devices are installed. To overcome the lack of per-service and per-UE real-time measurements, CURSA-SQ includes a *dynamic configuration* module that, among other tasks, finds a feasible traffic disaggregation given the aggregated measured traffic and the information related to the UEs in the RAN; specifically, likely per-service flows are estimated, so that their summation produces an aggregated estimated flow that statistically behaves similar to the aggregated measured one. With such estimated traffic disaggregation, the dynamic configuration module prepares the scenario to run a simulation phase for the next short time window, and an *end-to-end KPI estimation* module computes the KPIs based on the results of the simulation phase that are sent to the *performance analysis* module in the MDA controller; the latter can carry out some evaluation on the estimated end-to-end KPIs and send timely recommendations to the SDN controller and mobile core. Last but not least, an *evaluation and tuning* module waits until real aggregated monitoring data measured from the network is available, compares them to the results estimated by the *simulation* module for the same time period, and uses the results to tune specific parameters in the dynamic configuration module.

The next two sections are devoted to defining the extensions to CURSA-SQ to evaluate the KPIs in a fixed-mobile network and to define the rest of the modules and their iteration within the CURSA-SQ architecture.

III. COMPUTING KPIS ON A FIXED-MOBILE NETWORK

For the sake of clarity and completeness, in this section we first summarize the general CURSA-SQ queue model from [23]. The CURSA-SQ queue model is a continuous G/G/1/k queue model with a First-In-First-Out (FIFO) discipline [24] based on the logistic function. Next, we extend the CURSA-SQ queue model to include shared medium and mobility, and finally, we focus on end-to-end KPI computation to enable analyzing the performance from UEs to the mobile core.

A. The general CURSA-SQ queue model

Fig. 2 presents the CURSA-SQ queuing module, where a number *n* of continuous input traffic flows are aggregated into a single input flow *X*(*t*) that is queued in the capacitated queuing system and finally leaves the queue as output traffic flow *Y*(*t*). As the queue has a limited capacity *k*, traffic loss *l*(*t*) can appear if the input traffic flow exceeds the available capacity in the queue at time *t*; the available capacity depends on the amount of data in the queue, named *queue state* *q*(*t*). Therefore, we define $\hat{X}(q(t),t)$ as the amount of input flow actually stored in the queue at time *t*. The queued data remains for a time *d*(*t*) in the queue until the server in the queue module processes them at a constant rate μ. The server rate can be selected according to the throughput of the element that the queue system models (e.g., a 10Gb/s network interface). The used notation is summarized in Table I.

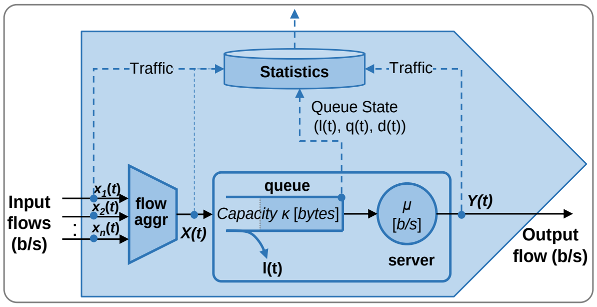

Fig. 2 General CURSA-SQ queuing module.

Table I Notation

Var/param	Description
$X(t)$	Input flow (b/s) at time *t*
k	Capacity of the queue (bytes)
$q(t)$	Bytes in queue at time *t*
$\hat{X}(q(t), t)$	Input flow (b/s) actually stored in the queue at time *t*
$l(t)$	Traffic loss in the queue at time *t*
$d(t)$	delay in the queue at time *t*
$Y(t)$	Output flow (b/s) at time *t*
μ	Server bitrate (b/s)

Equation (1) reproduces the differential equation from [23] defining the capacitated logistic queue model proposed in the general CURSA-SQ methodology. The model can be solved in the time interval [t_0, t_{max}] by initializing the queue with an initial value Q_0 at the starting time t_0.

$$q'(t)=\frac{1}{8}\cdot\left[\hat{X}\left(q(t),t\right)-\left(\mu+\left(\min\left\{\mu,\hat{X}\left(q(t),t\right)\right\}-\mu\right)\cdot \mathrm{e}^{-8\cdot\lambda\cdot\frac{q(t)}{\mu}}\right)\right],\quad t\in\left[t_0,t_{\max}\right]\, where\ q\left(t_0\right)=q_0\,. \tag{1}$$

The CURSA-SQ methodology allows measuring the performance of each queue. Specifically, two KPIs can be measured: *i*) flow loss is defined as the difference between the input flow and the flow actually stored in the queue, given the state of the queue at time *t*. For convenience, let us define the relative loss *l*(*t*) as the percentage of flow lost (see eq. (2)); *ii*) the delay in a queue at time *t*, *d*(*t*), is defined as the time needed to empty the queue given the queue's server rate (see eq. (3)).

$$l(t)=1-\frac{\hat{X}(t)}{X(t)}\in\left[0,1\right], \tag{2}$$

$$d(t)=8\cdot\frac{q(t)}{\mu}\,. \tag{3}$$

A statistics block in the queuing module (Fig. 2) collects the input and output traffic flow and measures the queue's KPIs. Note that the disaggregated output traffic flows can be easily computed assuming that loss distributes uniformly among the input traffic flows.

B. *Extension for Shared Medium with Mobility*

Let us consider a scenario with a single cell and several UEs connected; all the traffic between UEs and the base station shares the same physical medium, and thus, its capacity. It is clear that the capacity of the shared medium is not evenly distributed among the UEs in the cell, as the capacity that every UE perceives depends on the signal-to-interference-plus-noise ratio (SINR) and thus, on its specific geo-localization. In this regard and aiming to reduce the size of the problem and thus its computation time, UEs can be aggregated into groups following a *similarity* criterion in terms of e.g., their perceived capacity or their perceived signal quality. Note that the similarity criterion should be considered together with the cell's scheduler in order to achieve the most accurate results. In this paper, we assume the *proportionally fair* (PF) policy [25] and group UEs by the similarity in their perceived capacity. The size of the groups varies with time as UEs move, so mobility is modeled by updating the size of two or more groups.

According to the general CURSA-SQ methodology [23], traffic generation can be fairly aggregated in *mobile entities* with similar characteristics, including those related to packet/flow traffic, service, and infrastructure. Under this assumption, a shared medium of capacity C can be modeled as a system of queues, where we define a different queue for each group of UEs consuming the same service (see Fig. 3a-b). Then, given a set S of services and a set N_c of groups of UEs, a cell c is modeled as a set E_c of $|N_c|\times|S|$ mobile entities, each with a traffic generator and a queue. Let us now define the traffic generated by a mobile entity e=<n, s>, which can be characterized by the number of UEs $u(t)$ in group n and the traffic profile defining service s. In particular, two random variables are used to model the traffic of every single UE related to a given service s: *i*) the inter-arrival burst rate, and *ii*) the burst size. Then, the expectation and variance of both random variables are conveniently scaled using $u(t)$ to generate aggregated traffic traces (see [23] for more details).

As for mobility, as introduced above, it is modeled by updating the value of $u(t)$ of two or more entities in a correlated manner. Note that by updating $u(t)$, both intra- and inter- cell mobility can be implemented, depending on whether the entities are in the same or in different cells.

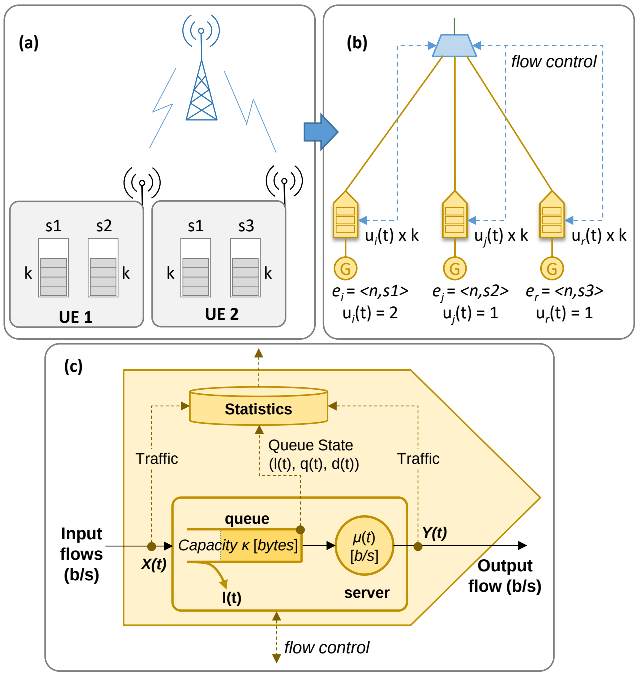

Fig. 3 Example of cell modeling of two UEs in a group (a-b). CURSA-SQ queuing model for a mobile entity (c).

Regarding queues, they are characterized by their capacity and their server rate. The capacity of the queue in an entity e is given by the buffer size (k) typically allocated by the Radio Link Control (RLC) protocol [25] times the number of UEs $u(t)$ in the entity. As for the server rate μ, we redefine that in the original CURSA-SQ model as being a function of time ($\mu(t)$), which converts the system into a *non-autonomous* Ordinary Differential Equation (ODE). In the cell model in Fig. 3b, all the queues are connected to an element that aggregates and disaggregates the traffic of the cell and emulates the shared medium of the cell, while implementing the PF scheduler; specifically the aggregator implements flow control by tuning the value of the server rate $\mu_e(t)$ for each mobile entity *e* in the cell *c*. Then, $\mu_e(t)$ can be modeled as:

$$\mu_e(t) = C_c \cdot \left[\alpha_c(t) \cdot g\left(q_e(t), q(t)\right) + (1-\alpha_c(t)) \cdot f_e \right], \tag{4}$$

where: *i*) $\alpha_c(t)$ is a weighting factor in [0,1]; *ii*) $g(\cdot)$ models the cell's scheduler policy with $q_e(t)$ being the current state of the local queue and $q(t)$ that of all queues in the cell; and *iii*) f_e is the fixed proportion of the capacity of the cell that mobile entity *e* will perceive, computed as $\mathrm{SINR}_e / \sum_{e' \in Ec} \mathrm{SINR}_{e'}$. The scheduler can manage the capacity sharing among the different mobile entities as a function of every request.

We implement the PF scheduler by solving the problem for every single cell for every time *t* in the given time window in two stages: *i*) the initial stage (*stage 0*) assumes α_c=0, i.e., $\mu_e(t)$ is proportional to the SINR perceived by entity *e*. This stage returns a value for $q_e(t)$, denoted $q^0{}_e(t)$; *ii*) the second stage uses a value of α_c that balances the server bitrate assigned to each entity in order to introduce fairness. α_c can be estimated as eq. (5):

$$\alpha_c(t) = \frac{1}{|N_c| \cdot |S| \cdot k} \cdot \sum_{e \in Ec} q_e^0(t). \tag{5}$$

As evaluating α_c for every time *t* in the given time window would heavily impact on the performance of the proposed method, we run stage 0 for longer periods where α_c is kept constant, and transform eq. (5) by computing the maximum for the period. Note that eqs. (4) and (5) focus on providing fairness among the entities in the cell; the resulting $\mu_e(t)$ value is evenly shared by all the UEs in the entity, and hence to preserve fairness among UEs, size of the entities should be balanced as much as possible.

Fig. 3c shows the details of the queue for mobile entities, where, similarly to the general CURSA-SQ queue module in Fig. 2, it stores the incoming flow traffic in a queue of capacity *k*, where the flow remains until it leaves at the programmed server rate $\mu_e(t)$.

By solving the model in eq. (1) with the shared medium extension in eq. (4) for a time interval, *per-entity* throughput, delay, packet loss and others KPIs can be obtained; *per-UE KPIs* are computed proportionally from *per-entity* ones. Regarding KPI computation, equations (2) and (3) can also be applied to compute the traffic loss and the delay in the queue for mobile entities. Note that the server rate in eq. (3) becomes $\mu_e(t)$ now.

C. *End-to-end KPI computation*

Once the queue models are defined, we can use them to simulate complex network scenarios consisting of a set of flows *F*, each combining fixed and mobile network segments, in order to analyze the evolution of the KPIs queue by queue, as well as flow by flow. For the latter, we still need to define how end-to-end KPIs are computed. To this end, we made use of the queue models defined above, which will be used here with the index of a particular queue *q*.

The throughput of a flow *f*, $T_f(t)$, can be computed as a function of the input flow and the accumulative loss along the route followed by the flow. Let us define the route followed by a flow as a set *H* of consecutive route segments, $H = \{h_1, h_2, \ldots h_n\}$, where each segment *h* consists of a set *P* of parallel paths, each conveying a proportion β_{fhp} of the flow. As flow loss $l(t)$ was defined above as a percentage, we can easily compute the accumulated loss as a function of the loss in every segment *h*, $l_h(t)$ (eq. (6)). In turn, $l_h(t)$ can be computed as the sum of the weighted loss along each path in the segment, which are finally computed as the product of the loss in every queue of the path (eq. (7)). Note that the traffic of a given flow distributes among all the paths in every segment of its route, and so the sum of proportions β_{fhp} for all the paths of every segment equals 1 (eq. (8)).

$$T_f(t) = X_f(t) \cdot \prod_{h \in H(f)} \left(1 - l_h(t)\right) \quad \forall f \in F \tag{6}$$

$$1 - l_h(t) = \sum_{p \in P(h)} \beta_{fhp} \cdot \prod_{q \in Q(p)} \left(1 - l_q(t)\right) \quad \forall h \in H \tag{7}$$

$$\sum_{p \in P(h)} \beta_{fhp} = 1 \quad \forall h \in H(f), f \in F \tag{8}$$

Similarly, the end-to-end delay of a flow *f*, $D_f(t)$, can be computed as the sum of the delay of every segment of the route of the flow (eq. (9)), where the delay of a segment is defined as the maximum delay of the paths of the segment; the delay of a path is defined as the summation of the delay in every queue along such path (eq. (10)).

$$D_f(t) = \sum_{h\in H(f)} d_h(t) \quad \forall f \in F \tag{9}$$

$$d_h(t) = \max_{p\in P(h)} \left\{ \sum_{q\in Q(p)} d_q(t) \right\} \quad \forall h \in H \tag{10}$$

IV. Near Real-Time KPI Estimation

Once the extensions to the general CURSA-SQ model for shared medium and mobility and end-to-end KPIs computation for fixed-mobile network scenarios have been presented, in this section, we focus on the rest of CURSA-SQ modules that are needed to produce accurate KPI estimation for the next time window. Fig. 4 presents a more detailed architecture of the CURSA-SQ module running inside the MDA controller; in particular, the queue models presented in Section III are implemented in the *Simulation* module, whereas end-to-end KPI estimation is performed in the related module. For illustrative purposes, Fig. 5 partially shows the simulation set-up configured for the scenario considered in Fig. 1, where three cells, each serving several entities, are considered.

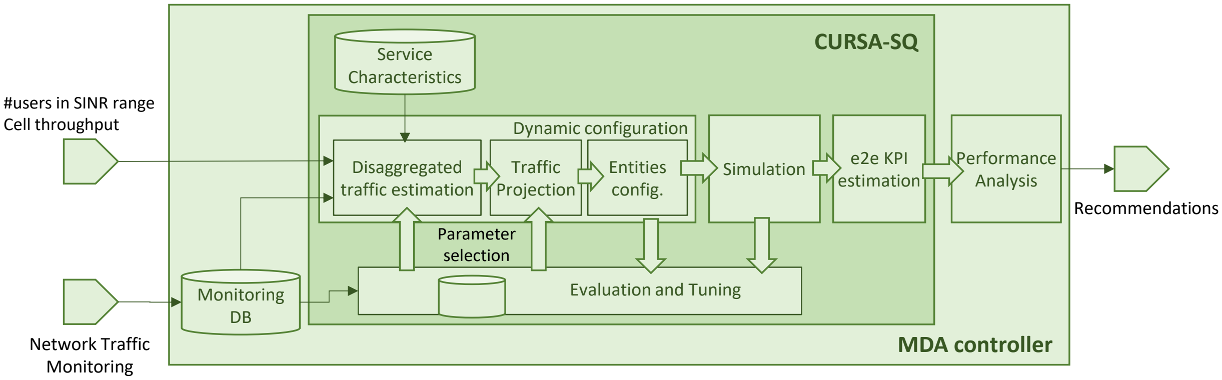

Fig. 4 Inputs and outputs and CURSA-SQ building blocks.

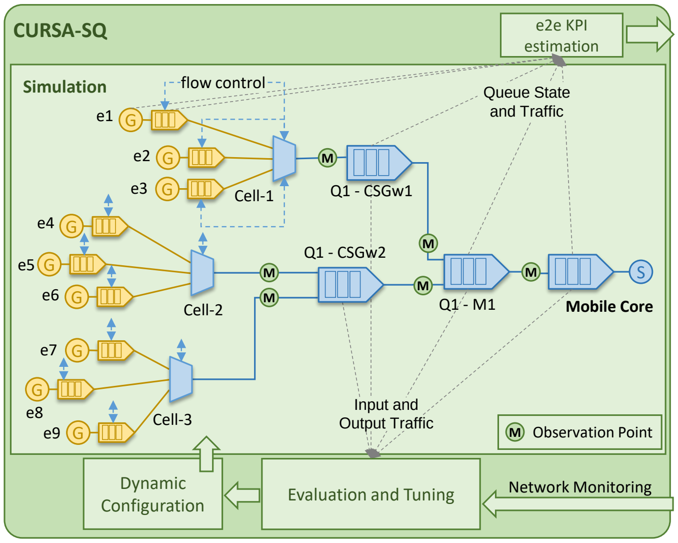

Fig. 5 CURSA-SQ simulation and statistics utilization.

In the architecture, the *dynamic configuration* module includes three submodules. The *entities configuration* submodule is in charge of the definition of the mobile entities set ($E_c = \{<n, s>\}$) in every cell *c*, where every entity *e* represents a set of UEs under similar perceived SINR consuming one service (see Section III.B). However, for that module to provide a proper configuration, especially for the traffic to be generated and consumed by every entity, a per-entity traffic estimation is needed. Let us assume that we are given the total cell throughput, as well as an initial discretization M_c, where every m specifies the number of UEs perceiving similar SINR. Then, two modules are involved in the prediction of the future traffic disaggregated by service and such given discretization: first, the *disaggregated traffic estimation* submodule finds a likely disaggregation of the cell's throughput into the set of pairs <*m*, *s*>, and next, the *traffic projection* submodule forecasts future traffic for each pair <*m*, *s*>. The *entities configuration* submodule creates the final mobile entity set by finding the optimal grouping of pairs <*m*, *s*> into each entity *e*=<*n*, *s*>. The last module is that of evaluation and tuning, which evaluates the accuracy of the estimation by comparing it against the real traffic conditions measured from the network. These modules are detailed next.

A. Disaggregated Traffic Estimation

This module deals with solving an optimization problem for every cell *c*, which can be stated as follows:

Given:

- the measured throughput of the cell *c* defined as a set of statistics (W_c), including mean, max, min, variance, 95th percentile, and so forth,
- the total number of UEs in a cell and an initial discretization M_c of the cell in terms of the perceived SINR,
- a set of services *S* and their traffic characteristics,
- an initial percentage (v_{cs}) of traffic volume for every service *s*. This initial percentage can be used to consider smooth evolutionary solution when coming from the previous time interval, as well as serve as a way to bias the optimization problem,
- weights γ_1 and γ_2.

Output: the percentage of traffic volume for every pair <*m*, *s*> for the cell *c*, denoted as θ_{csm}.

Objective: minimize the difference between the given throughput and the estimated one in terms of the defined statistics, weighted by γ_1, as well as the difference between the initial per-service traffic percentages and the estimated ones, weighted by γ_2. Note that the first term ensures that the results are as close as possible to the observed throughput, whereas the second one forces the results to follow a smooth evolution as introduced above.

The objective function Φ_c is defined for every cell *c*:

$$\Phi_c = \gamma_1 \cdot \sum_{w \in Wc} \left(f_w(\theta, M_c, S) - W_c(w) \right)^2 + \gamma_2 \cdot \sum_{s \in S} \left(v_{cs} - \sum_{m \in M} \theta_{csm} \right)^2 \quad \forall c \in C\,, \tag{11}$$

where, specific functions $f_w(\cdot)$ are defined to compute the average for every traffic statistic given the unknown percentages of traffic volume, the initial discretization, and the characteristics of the services. It is worth noting that by properly tuning weights, optimization emphasis can range from a smooth or static (γ_1=0, γ_2=1) evolution to a more dynamic (γ_1=1, γ_2=0) one.

The disaggregated traffic estimation optimization problem can be solved using the gradient descent algorithm [27] and the results, together with the set of statistics for the measured throughput, are stored in a database.

B. Traffic Projection

The traffic projection module applies interpolation to the estimated historical disaggregated traffic and measured cell throughput to forecast per <*m*, *s*> pair traffic and cell throughput in the next time window. The size of the historical data considered for interpolation is defined by parameter h_w.

This module uses *structural models* based on the state-space models that allow using more than one correlated time series [26]. In particular, the accuracy of interpolations of per <*m*, *s*> pair estimated traffic increases when the measured cell throughput is considered since the aggregation of all pairs <*m*, *s*> is naturally correlated to the overall cell throughput. In addition, the method can potentially identify the structural components of the considered time series (e.g., trend or seasonality), making it possible to deal with very different behaviors with just one single modeling approach.

C. Entities Configuration

Finally, the entities configuration reduces the number of possible generators to configure for the simulation step by grouping <*m*, *s*> pairs with similar characteristics to create mobile entities. The problem can be stated as follows:

Given:

- the number |*S*| of considered services,
- the set of pairs <*m*, *s*> and the projected traffic for every pair,
- a number $|N_c|$ of entities to be created by grouping <*m*, *s*> pairs with similar perceived SINR.

Output: the $|N_c|\times|S|$ mobile entities, including its assigned SINR and its projected traffic.

Objective: minimize the error between the SINR assigned to each entity and the SINR of each <*m*, *s*> pair weighted by the number of the pair. As a secondary objective, we are interested in obtaining entities representing a balanced number of UEs to achieve fairness among UEs in the cell.

The entities configuration optimization problem can be solved using the *k*-means clustering algorithm [14] complemented with a final phase for balancing focused on the secondary optimization objective.

D. Evaluation and tuning

Once the network scenario has been simulated and the KPIs have been estimated for the next time window, the results are temporally stored and can be used to evaluate their precision by comparing them against the real traffic conditions measured from the network. Note that the dynamic configuration module requires different configuration parameters that need to be tuned according to the specific scenario under study. Specifically: *i*) the disaggregated traffic estimation submodule includes parameters γ_1, γ_2, and v_{cs}. Note that the value of parameter v_{cs} can be equal to the values of the θ_{csm} from the previous estimation window, so the values selected for γ_1 and γ_2 can result into more dynamic or more persistent models; *ii*) in the traffic projection submodule, the parameter to evaluate and tune is the size of the historical time window, h_w; using large historical time windows, more importance will be given to the trend in the model, whereas with small sizes the model will detect changes in the injected traffic to the network; and *iii*) the $|N_c|$ parameter can be adjusted in the entities configuration submodule; note that the higher the number of entities, the higher the accuracy of the simulated traffic characteristics. In addition, $|N_c|$ can be dimensioned to facilitate the balancing of UEs among entities, which is a key aspect for the sake of system fairness.

In order to evaluate the current value of the above configuration parameters, an optimization problem is solved to find the optimal value of the configuration parameters, *Params**, for the estimation window corresponding to the monitoring data. The optimization problem is based on comparing the traffic measured at different points during the CURSA-SQ simulation and KPI estimation against the traffic that is measured from the network. Specifically, we assume the availability of measurements at the input of the access/metro network corresponding to the traffic from/to the RAN, as well as some type of measurements of aggregated traffic per service that can be obtained, e.g., at the output of the mobile core. In addition, other monitoring points can be considered as well. We consider that these monitoring data has a coarser granularity than that from the mobile network, which uses fine grained data for near real-time management.

Then, the optimization problem for evaluation can be stated as follows:

Given for time period *T*:

- the measured traffic at the interconnection between the RAN and the access/metro network,
- the aggregated traffic per service monitored at the output of the access/metro network,
- the traffic measurements from the CURSA-SQ simulation module.

Output: the set of optimal CURSA-SQ configuration parameters, $Params^*=<\gamma_1^*, \gamma_2^*, v_{cs}^*, h_w^*, |N_c|^*>$.

Objective: minimize the distance between the characteristics of the traffic observed in the real network and those of the traffic generated during the simulation.

Once solved the evaluation problem, tuning of the configuration parameters can be performed for the next estimation window. As the optimal values for the configuration parameters correspond to a historical time period *T*, the tuning module stores the optimal values just obtained and predicts the values of the parameters to the next time window based on their historical evolution. Then, considering a confidence interval for such predictions, parameters are updated only if their current value is outside the corresponding confidence interval.

V. Simulation Results

In this section, we numerically study and validate the proposed CURSA-SQ methodology for end-to-end performance estimation in converged fixed-mobile networks. To this aim, we firstly focus on evaluating the performance of the proposed extension for shared medium proposed in Section III.B by means of the ns-3 network simulator. Second, we carry out a sensitivity analysis of the dynamic configuration module in Section IV as a function of the reliability and precision of its submodules. Finally, we present a set of use cases to illustrate the use of CURSA-SQ for near real-time fixed-mobile network analysis.

A. Validation of the mobile network simulation

To validate CURSA-SQ extension for shared medium, we run several simulations to compare the performance of our Matlab implementation against that of the ns-3 network simulator implementing the LTE module [28] modeling the full LTE Radio Protocol and the EPC, including the core network interfaces, protocols, and entities. Both CURSA-SQ and ns-3 run on an i7-8700 server with 16GB RAM and Ubuntu 18.04.

For this comparative study, a scenario with one single cell was simulated consisting of a base station with a three-sectored antenna, an EPC, and several UEs receiving the traffic (UDP packets) injected by a random bursty traffic generator. Each sector was modeled as a parabolic antenna with a 3dB beam width of 70 degrees and a maximal attenuation of 20dB. For the sake of simplicity, we considered interference-free radio links with line of sight between the base station and the UEs. The ns-3 scenario was configured with the PF scheduler, 1ms transmission time interval, and 5MHz downlink bandwidth. According to the adaptive modulation and coding model in [29], the simulator finds the best *modulation and coding scheme* for a given channel condition. For the sake of a fair comparative analysis, components f_e and $g(\cdot)$ in eq. (4) have been modeled to match the abovementioned configuration. In addition, packet traces generated and used in ns-3 simulation were aggregated in flows with a granularity of 250 ms and used in the CURSA-SQ simulation.

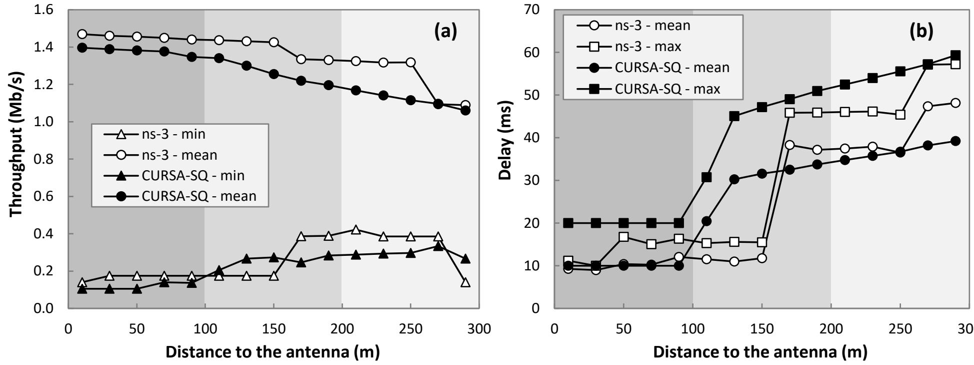

Fig. 6 Throughput (a) and latency (b) in the radio segment vs. distance to the antenna.

Table II Relative difference between CURSA-SQ and ns-3

	Peak-average ratio		
	[1, 1.2)	**[1.2, 1.4)**	**[1.4, 1.7)**
Throughput–min	3.5%	6.8%	10.0%
Throughput–mean	1.5%	3.7%	5.3%
Delay–mean	13.2%	14.1%	17.4%
Delay–max	20.5%	15.3%	12.9%

Fig. 6 shows the results of the simulation of 15 UEs located between 10 and 290 m from the antenna; CURSA-SQ was configured with one single UE per entity, i.e., 15 entities. The obtained minimum and average throughput, and the average and maximum delay per-UE are presented in Fig. 6a and Fig. 6b, respectively, where similar values for both simulation environments can be observed. The larger deviations are for the estimation of the delay for medium distances (100-200m), where CURSA-SQ overestimates the delay as a consequence of the intrinsic nature of the continuous queue model. However, the impact of such overestimation is minor as they could lead to conservative decisions for the performance analysis module.

Table II provides an extended comparison of the previous results in terms of the relative difference for quantifying relevant KPIs. Several repetitions with different random traffic traces and mobility patterns were simulated. The results in Table II are segmented by different peak/average traffic ratios of the traces; the higher ratio the more bursty the injected traffic. Note that throughput errors typically remain below 10%, whereas higher delay estimation errors are caused by the CURSA-SQ overestimation illustrated in Fig. 6.

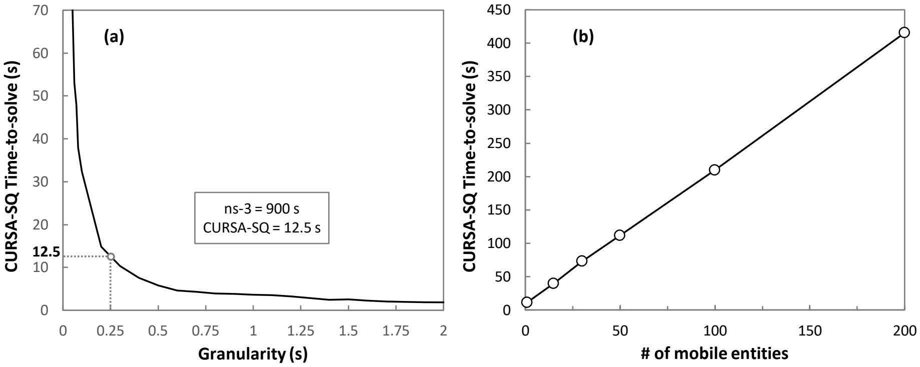

Fig. 7 CURSA-SQ time-to-solve vs. granularity (a), and vs. the number of mobile entities in the cell (b)

Let us now evaluate CURSA-SQ in terms of scalability and its applicability for near real-time KPI estimation. To this aim, let us consider that, in order to make and implement operational decisions, simulations of 2-minute time windows need to be carried out. Fig. 7a shows the time-to-solve one-entity queue system model as a function of the granularity configured in CURSA-SQ. The impact of reducing the granularity is two-fold: while the precision of KPI estimation and the amount of information for performance analysis and decision-making increases, the time-to-solve also increases, which can impact negatively for near real-time operation. As it can be observed, sub-second granularities can be achieved with low time-to-solve times. Specifically, by selecting 250ms granularity, just 12.5 seconds were needed; this is remarkably lower than the simulated 2-minute time-window (10% of the simulated time), which enables its use for near real-time operation. Note that the ns-3 simulation required ~15 min, i.e., 7.5 times the simulated time. Assuming such granularity, Fig. 7b shows the CURSA-SQ time-to-solve when the number of mobile entities in a cell increases; the results show a clear linear trend that is related to the number of calls to the ODE solver, which confirm the applicability of CURSA-SQ for a wide range of realistic scenarios.

B. Evaluation of the dynamic configuration module

Once the extension of CURSA-SQ for shared medium has been validated in the mobile network segment to compute KPIs assuming a perfect configuration of the system, let us now evaluate the dynamic configuration module, which in case of inaccuracy could introduce some error in the KPI estimation. Recall that the dynamic configuration module finds the best configuration of entities based on the estimation of the disaggregated traffic estimation and the traffic projection submodules. To this aim, we conduct a sensitivity analysis of the dynamic configuration module in the case of errors in the entities' configuration and traffic estimation. Without loss of generality, we focus on the impact of dynamic configuration errors in the estimation of the KPIs on the mobile segment using the configuration of the cell and UEs from the previous section. For the sake of clarity, we analyze both types of errors separately.

Let us first concentrate on analyzing traffic estimation errors. To this end, we configured one entity per UE and assumed the traffic flow traces generated from the ns-3 simulator as the real traffic while added an unbiased Gaussian error to emulate an overall prediction error from the disaggregated traffic estimation and traffic projection submodules. Fig. 8a presents the evolution of the error of the estimation for the minimum throughput and the maximum delay as a function of the normalized error introduced. A close-to-linear relation is observed between the error introduced in the traffic estimation and the error introduced in KPIs estimation when the former is below 30%; above that value, traffic estimation is too poor to be used for KPI estimation. Note that the 30% limit is not stringent, as it is expected that the traffic estimation can remain largely below this threshold. Under such assumption, the results illustrate two very positive properties of the CURSA-SQ methodology: *i*) the estimation of the error introduced by the disaggregated traffic estimation and traffic projection submodules provides a likely estimation of the error in the KPI estimation; *ii*) improving the quality of these submodules with the help of the evaluation and tuning module, will proportionally improve the quality of the KPI estimation.

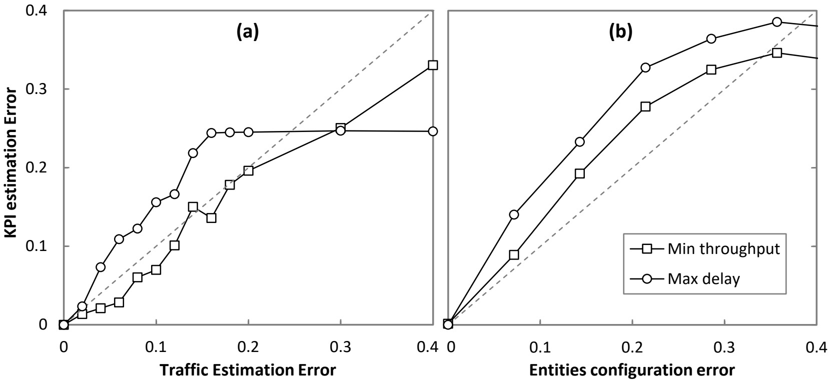

Fig. 8 Error in throughput and delay estimation vs. error in traffic estimation (a) and entities configuration error (b).

Let us now focus on evaluating the entities configuration submodule, assuming perfect traffic projection. Assuming that one entity per UE (15 entities) is the optimal configuration for the current scenario, we introduced error by reducing the number of entities, thus aggregating several UEs per entity. We computed a relative error as the ratio between the number of entities configured over the optimum one. Fig. 8b plots minimum throughput and maximum delay errors as a function of the error introduced by an inaccurate entities' configuration. A linear evolution of KPI estimation error is again observed with configuration error below 40%, which leads to similar conclusions than those for the traffic estimation error.

In view of the results, we conclude that the errors introduced by the dynamic configuration module have a moderate impact on KPI estimation. Moreover, by evaluating the differences between simulated and real traffic measurements, a rough but likely estimation of the error between estimated and real KPIs could be obtained.

C. *Performance of End-to-End KPI Estimation*

Once CURSA-SQ has been validated for the mobile network segment model, let us now focus on analyzing its performance for end-to-end KPI estimation. To this aim, we configured a more complex network scenario that considers mobile and fixed network segments; specifically, we configured the topology depicted in Fig. 5 that includes three cells, two CSGws, one metro router, and the mobile core. Every cell has been scaled up to a capacity of 1 Gb/s, whereas 10 Gb/s links have been considered for the fixed access-metro segment. We split every cell into 5 SINR zones, where every zone includes users consuming three types of services namely, Video-On-Demand (VoD), Gaming, and Internet; thus, every cell has been configured with 5 x 3 = 15 entities. The statistical characteristics of the services are in line with those in [23].

Three scenarios are considered for the evaluation, where each scenario is defined by the number of UEs in every entity and their mobility during the simulation time. The first and simplest scenario (*S1 - Normal*) reproduces the case where both the number of UEs and the background traffic remains constant along the simulation. The second scenario (*S2 - Cell congestion*), represents the case of UEs mobility between cells 1 and 2. Finally, the third scenario (*S3 - Metro congestion*) represents the case where the background traffic experiences a remarkable increment due to some external cause out of the analyzed RAN.

Let us assume that the initial time of the simulation (t_0) corresponds to the current time, whereas the final time (t_w) represents the end of the forecasted time window; then, $t_w = t_0 + 2$min. To properly configure entities for the whole duration of the simulation, the evolution of the number of UEs per cell and service is received from some external system; with this, linear traffic forecast from t_0 to t_w is considered. Finally, a realistic scenario is emulated by configuring an extra entity injecting background Internet traffic at every CSGw. Indeed, although for the sake of simplicity only three cells are simulated in detail, a denser RAN is considered where the access-metro segment supports such traffic. Table III and Table IV report the total number and the mobility of UEs per cell and the evolution of background traffic per CSGw, respectively, for the three considered scenarios at t_0 and t_w.

Table III Number and mobility of UEs per scenario and cell ($t_0 \rightarrow t_w$)

Cell	S1 - Normal	S2 - Cell Congestion	S3 - Metro Congestion
1	72 → 72	**18 → 126**	72 → 72
2	72 → 72	**126 → 18**	72 → 72
3	72 → 72	72 → 72	72 → 72

Table IV Evolution of background traffic (Gb/s) per scenario and cell ($t_0 \rightarrow t_w$)

CSGw	S1 - Normal	S2 - Cell Congestion	S3 - Metro Congestion
1	2.2 → 2.2	2.2 → 2.2	2.2 → 2.2
2	2.3 → 2.3	2.3 → 2.3	2.3 → **7.9**

Fig. 9 plots the evolution of the normalized load (computed as the ratio between the effective throughput and the bitrate) as a function of the time during the simulation window for all the three scenarios and the defined observation points (Fig. 5) in the cells, whereas Fig. 10 plots those for CSGws and metro nodes. As expected from the characteristics of scenario *S1*, steady behavior is observed in Fig. 9a and Fig. 10a, where the load is moderately low in every cell, as well as in the interfaces in the nodes of the fixed network. In scenario *S2*, the impact of mobility on cells' normalized load is clearly visible (Fig. 9b and Fig. 10b); while normalized load stays steady for cell 3 as in scenario 1, it largely increases in cell 1 as soon as decreases in cell 2, as a result of UEs moving from one cell to the other. In fact, the normalized load occasionally exceeds 0.8, which could lead to significant latency as a result of large traffic queuing. Although variations in the normalized load are observed in the CSGws, they are negligible compared to those in the cells as a result of traffic aggregation. Finally in scenario *S3*, cells' normalized load remain constant, while that in the observation point in CSGw2 increases causing metro congestion at the end of the simulated time window (Fig. 9c and Fig. 10c); this allows to conclude that the cause of congestion is not located in the RAN.

The above analysis was based on the traffic resulting from CURSA-SQ simulation in specific observation points. Other powerful results from CURSA-SQ simulation are end-to-end KPIs that allow evaluating whether future network conditions will actually impact the performance experienced by the end users. Fig. 11 plots end-to-end delay measured for the UEs in cell 1 under the considered scenarios; minimum, average, and maximum delay per-UE is plotted as a function of time. Let us also assume that analysis is carried out to detect whether end-to-end delay will exceed a threshold value set at 200 ms.

In scenario *S1*, users will experience delay below the threshold although some user will occasionally experience delay higher than the threshold. Given this, the performance analysis module would not issue any recommendation to the SDN controller, so the current network configuration is kept invariant. The analysis of scenario *S2* would clearly identify increasing maximum delay, which eventually will violate the threshold by far. Note however, that average delay stays well under the threshold. In this case, the analysis of the normalized load at the observation points (Fig. 9b and Fig. 10b) would allow to clearly identify that the cause of such end-to-end delay violation is congestion in cell 1; this conclusion can be notified to the mobile core. Finally, in scenario *S3* average and minimum delay follow the evolution of the maximum delay and exceed the threshold at the end of the simulation window. This fact (i.e., all users being equally affected), jointly with the analysis of traffic in the observation points (Fig. 9c and Fig. 10c), clearly identify congestion in the metro segment. The conclusion of this analysis can be notified to the SDN controller.

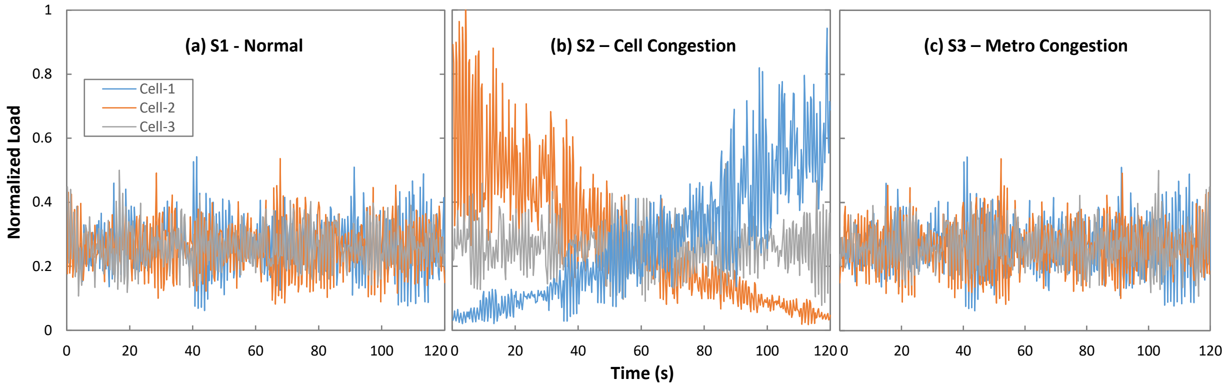

Fig. 9 Normalized load in the RAN as a function of time for every scenario

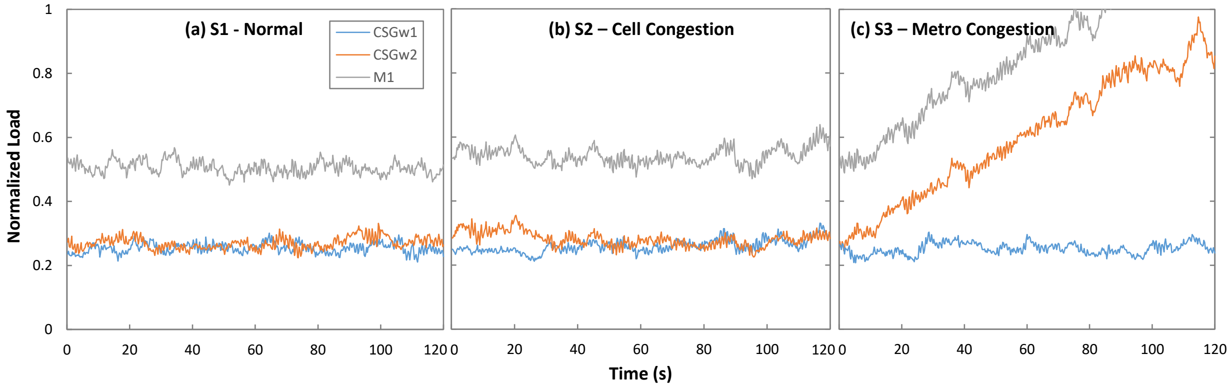

Fig. 10 Normalized load in the fixed network as a function of time for every scenario

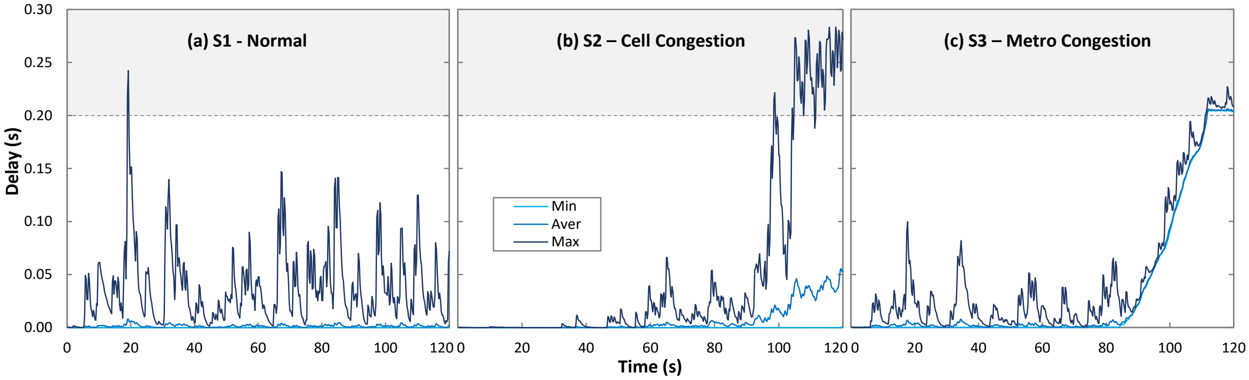

Fig. 11 End-to-end delay experienced by users in cell 1 in every scenario

A finer analysis can be carried out at the mobile entity level. Fig. 12a-b show the evolution of the throughput and delay vs. time, respectively, for the two mobile entities with the highest and the lowest SINR in cell 1 under scenario S2. It can be easily observed in Fig. 12a that the throughput experienced by both entities under low load conditions (for time < 70 approximately) is similar independently of the signal quality and shows the fairness of the cell scheduler. As soon as the load of the cell increases (for time > 70), the throughput of the entity with the lowest SINR is limited by the quality of the signal. Even though, the observed delay (Fig. 12b) is similar for both entities.

Finally, Fig. 13 proposes an alternative analysis of the end-to-end delay, where 10s intervals are averaged to study the maximum delay introduced by every element; it follows eq. (9) and plots the accumulative evolution of the maximum delay at every observation point. This representation allows an even more clear identification of the element partially responsible for large end-to-end delay. In particular, cell congestion can be identified as the cause of the large end-to-end delay observed in scenario *S2*, whereas metro congestion is identified in scenario *S3*.

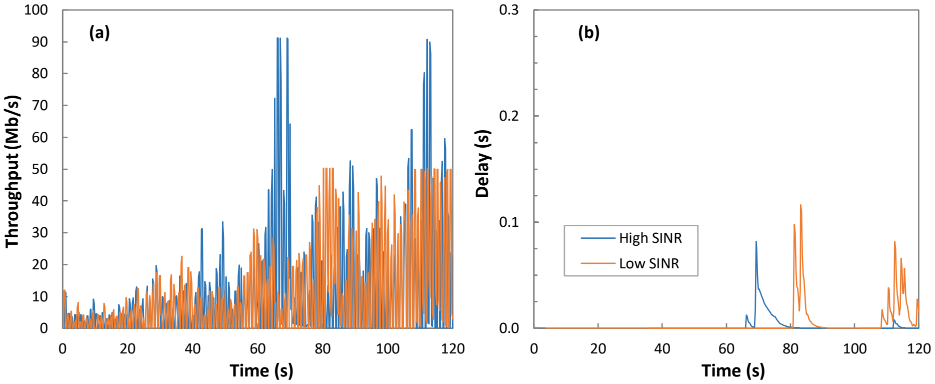

Fig. 12 Throughput (a) and delay (b) vs. time for the mobile entities with the highest and the lowest SINR.

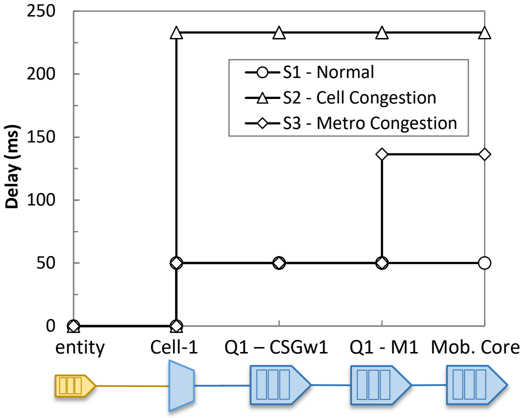

Fig. 13 Components of end-to-end delay

VI. Concluding Remarks

There is a clear need to estimate end-to-end KPI's in scenarios of converged fixed-mobile networks as the key to verify the performance of services. In this context, an extension to the general CURSA-SQ methodology for shared medium and mobility features has been presented, where UEs are grouped into entities as a function of the SINR that they perceive and the service that users consume. Entities are modeled as queue systems, where the service rate varies with time and depends on the actual SINR and the cell's scheduler policy.

The extension to CURSA-SQ and the end-to-end KPI estimation were complemented with additional modules aiming at enabling end-to-end KPIs estimation based on the simulation of network conditions. Specifically: *a*) the dynamic configuration module that configures the scenario before simulating every time window. This module consists of three submodules to: *i*) find a probable disaggregation of the cell's throughput into flows; *ii*) forecast future traffic for every flow; and *iii*) create the mobile entity set; and *b*) the evaluation and tuning module that evaluates the accuracy of the estimation by comparing it against the real traffic conditions measured from the network. Once the scenario is simulated for the next time window and KPIs estimation are computed, they can be used by a performance analysis module to detect performance degradation, identify their causes and issue recommendations that can help the SDN controller and the mobile core to proactively reconfigure resources in their domains.

CURSA-SQ was validated against the ns-3 network simulator for a pure mobile network scenario. CURSA-SQ estimations probed to be accurate while simulating 120s with granularity 250ms in just 12.5s. These results highlight the outstanding scalability of the proposed method for near real-time computation.

Once validated, a sensitivity study was carried out to analyze the dependence of the CURSA-SQ simulation against errors in the traffic prediction and configuration of the simulation. Interestingly, KPI estimation showed a linear relation with the error in the traffic estimation, as well as with the error in the entities' configuration, which can be corrected by the evaluation and tuning module by comparing the differences between simulated and real traffic measurements and tuning parameters in the dynamic configuration module.

Three scenarios were eventually configured on a realistic converged fixed-mobile network to illustrate the usefulness of the proposed approach to simulate network conditions and estimate end-to-end KPIs. Scenario *S1* reproduces a normal case where both the number of UEs and the background traffic remains constant, scenario *S2* represents the case of UEs mobility, and scenario *S3* represents the case where the background traffic experiences a remarkable increment due to some external cause. The results of the three use cases illustrate the type of analysis that should be carried out to anticipate performance degradation and precisely identify bottlenecks in the network.

In light of the obtained results, we can conclude that CURSA-SQ offers a precise and scalable methodology to simulate traffic conditions and to estimate end-to-end KPI with sub-second granularity near real-time in converged fixed-mobile networks. Posterior analysis can be carried out to identify the components that have more impact on the overall end-to-end figures, so that congestion or other effects in different parts of the network can be easily identified and localized.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Commission for the H2020-ICT-2016-2 METRO-HAUL project (G.A. 761727), from the AEI/FEDER TWINS project (TEC2017-90097-R), and from the Catalan Institution for Research and Advanced Studies (ICREA).

REFERENCES

[1] L. Velasco, A. Castro, A. Asensio, M. Ruiz, G. Liu, C. Qin, R. Proietti, S.J. Ben Yoo, “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.

[2] A. Asensio, M. Ruiz, L.M. Contreras, and L. Velasco, “Dynamic Virtual Network Connectivity Services to Support C-RAN Backhauling,” IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 8, pp. B93-B103, 2016.

[3] L. Huang, B. Ding, Y. Xu, and Y. Zhou, “Analysis of User Behavior in a Large-Scale VoD System,” in Proc. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), 2017.

[4] A. Rao, A. Legout, Y. Lim, D. Towsley, Ch. Barakat, and W. Dabbous, “Network characteristics of video streaming traffic,” in Proc. Conference on emerging Networking Experiments and Technologies (CoNEXT), 2011.

[5] A. Shams, S. Abied and M. Hoque, “Impact of user mobility on the performance of downlink resource scheduling in Heterogeneous LTE cellular networks,” in Proc. Int. Conference on Electrical Engineering and Information Communication Technology (ICEEICT), 2016.

[6] L. Velasco, P. Wright, A. Lord, and G. Junyent, “Saving CAPEX by Extending Flexgrid-based Core Optical Networks towards the Edges,” (Invited Paper) IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 5, pp. A171-A183, 2013.

[7] Telecom Infra Project: Open Optical & Packet Transport Working group, [on-line] https://oopt.telecominfraproject.com/, accessed July 2019.

[8] F. López, U. Silva, D. Campelo, R. Oliveira, S.-J. Lim, and L. García, “QoS Management and Flexible Traffic Detection Architecture for 5G Mobile Networks,” Sensors, vol. 19, pp. 1335, 2019.

[9] H. Huang, P. Li, S. Guo, W. Zhuang, “Software-defined wireless mesh networks: architecture and traffic orchestration,” IEEE Network vol. 29, pp. 24-30, 2015.

[10] A. Betzler, D. Camps-Mur, E. Garcia-Villegas, I. Demirkol, and J. Aleixendri, “SODALITE: SDN Wireless Backhauling for Dense 4G/5G Small Cell Networks,” IEEE Transactions on Network and Service Management, 2019.

[11] E. Coronado, S. Khan and R. Riggio, “5G-EmPOWER: A Software-Defined Networking Platform for 5G Radio Access Networks,” in IEEE Transactions on Network and Service Management, 2019. DOI: 10.1109/TNSM.2019.2908675.

[12] 5G PPP Architecture Working Group, “View on 5G Architecture - version 3.0 for Public consultation” (white paper), 2019.

[13] Test, Measurement, and KPIs Validation Working Group, “Validating 5G Technology Performance,” (white paper) 5GPPP, 2019.

[14] D. Rafique and L. Velasco, “Machine Learning for Network Automation: Overview, Architecture, and Applications,” IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. D126-D143, 2018.

[15] R. di Taranto, S. Muppirisetty, R. Raulefs, D. Slock, T. Svensson and H. Wymeersch, “Location-Aware Communications for 5G Networks: How location information can improve scalability, latency, and robustness of 5G,” IEEE Signal Processing Magazine, vol. 31, pp. 102-112, 2014.

[16] L. Velasco, A. Chiadò Piat, O. González, A. Lord, A. Napoli, P. Layec, D. Rafique, A. D'Errico, D. King, M. Ruiz, F. Cugini, and R. Casellas, "Monitoring and Data Analytics for Optical Networking: Benefits, Architectures, and Use Cases," accepted in IEEE Network Magazine, 2019 (DOI: 10.1109/MNET.2019.1800341).

[17] 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.

[18] L. Velasco, Ll. Gifre, J.-L. Izquierdo-Zaragoza, F. Paolucci, A. P. Vela, A. Sgambelluri, M. Ruiz, and F. Cugini, "An Architecture to Support Autonomic Slice Networking [Invited]," IEEE/OSA Journal of Lightwave Technology (JLT), vol. 36, pp. 135-141, 2018.

[19] F. Morales, M. Ruiz, Ll. Gifre, L. M. Contreras, V. López, and L. Velasco, "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.

[20] A. P. Vela, M. Ruiz, L. Velasco, "Distributing Data Analytics for Efficient Multiple Traffic Anomalies Detection," Elsevier Computer Communications, vol. 107, pp. 1-12, 2017.

[21] 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.

[22] The Network Simulator - ns-3, [on-line] http://www.nsnam.org/, accessed July 2019

[23] M. Ruiz, F. Coltraro, and L. Velasco, "CURSA-SQ: A Methodology for Service-Centric Traffic Flow Analysis," IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. 773-784, 2018.

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

[25] S. Sesia, I. Toufik, and M. Baker, *LTE - The UMTS Long Term Evolution - from theory to practice*, Wiley, 2009.

[26] A. Dort-Golts, "Short-Term Forecasting: Simple Methods to Predict Network Traffic Behavior," in Proc. International Conference on Next Generation Wired/Wireless Networking (NEW2AN), 2014.

[27] S. Ruder, "An overview of gradient descent optimization algorithms," cite arxiv:1609.04747, 2016.

[28] ns-3 LTE Module, [on-line] https://www.nsnam.org/docs/models/html/lte-design.html, accessed July 2019

[29] M. Mezzavilla, M. Miozzo, M. Rossi, N. Baldo, and M. Zorzi, "A lightweight and accurate link abstraction model for the simulation of LTE networks in ns-3," in Proc. ACM Int. Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM), 2012.