Flow Assignment and Processing on a Distributed Edge Computing Platform

Franco Davoli, *Life Senior Member, IEEE*, Mario Marchese, *Senior Member, IEEE*, and Fabio Patrone, *Member, IEEE*

***Abstract*—The evolution of telecommunication networks toward the fifth generation of mobile services (5G), along with the increasing presence of cloud-native applications, and the development of Cloud and Mobile Edge Computing (MEC) paradigms, have opened up new opportunities for the monitoring and management of logistics and transportation. We address the case of distributed streaming platforms with multiple message brokers to develop an optimisation model for the real-time assignment and load balancing of event streaming generated data traffic among Edge Computing facilities. The performance indicator function to be optimised is derived by adopting queuing models with different granularity (packet- and flow-level) that are suitably combined. A specific use case concerning a logistics application is considered and numerical results are provided to show the effectiveness of the optimisation procedure, also in comparison to a "static" assignment proportional to the processing speed of the brokers.**

***Index Terms*—Flow assignment, resource allocation, distributed computing, MEC, 5G.**

I. Introduction

DEEP changes are affecting the worldwide telecommunication infrastructure. New emerging use cases and a mix of new and traditional applications require a profound evolution of the overall telecommunication network to address the growing number of connected users and the ensuing traffic volume. The integration of technologies such as Software Defined Networking (SDN) [1], Network Functions Virtualization (NFV) [2], and Mobile Edge Computing (MEC) [3], [4] is leading to network softwarization, which brings telecommunication networks closer to computer systems for what concerns traffic flow management and resource allocation [5]. The consolidation of the fifth generation of mobile networks (5G) is further strengthening this aspect [6].

Manuscript received November 4, 2021; revised March 2, 2022; accepted April 29, 2022. This work was supported in part by the European Commission, through the H2020 5G PPP Projects MATILDA under Grant 761898 and in part by 5G-INDUCE under Grant 101016941. The review of this article was coordinated by Prof. Abbas Jamalipour. *(Corresponding author: Fabio Patrone.)*

Franco Davoli and Fabio Patrone are with the Department of Electrical, Electronic and Telecommunications Engineering, and Naval Architecture (DITEN), University of Genoa, 16145 Genova, Italy, and also with the National Laboratory of Smart and Secure Networks (S2N), National Inter-University Consortium for Telecommunications (CNIT), 16145 Genoa, Italy (e-mail: franco.davoli@unige.it; f.patrone@edu.unige.it).

Mario Marchese is with the Department of Electrical, Electronic and Telecommunications Engineering, and Naval Architecture (DITEN), University of Genoa, 16145 Genova, Italy, and also with the CNIT University of Genoa Research Unit, 16145 Genova, Italy (e-mail: mario.marchese@unige.it).

Digital Object Identifier 10.1109/TVT.2022.3172792

In this scenario, newly developed resource allocation and network control solutions can be based on computationally powerful techniques, such as deep learning [7]–[9]. The related problems may have commonalities with similar issues in computing systems and datacentres, and the boundary between communications and computing is becoming increasingly blurred [10]. Typically, general-purpose computing devices are going to replace special purpose telecommunications equipment and to host multiple tenants that act as Network Service Providers (NSPs) for their fixed or mobile customers that run applications on their User Equipment (UE). UEs may benefit from computing resources that can be partially local, i.e. available on the very same UEs, and partially residing in a remote datacentre or at the mobile edge. Edge Computing resources may be located at micro-datacentres (μDCs) deployed at the access network premises, i.e., in the vicinity of users, in order to reduce the response latency [11].

In this context, distributed streaming platforms like Kafka [12] can be implemented with brokers residing in the network edge and sharing the computational and storage resources of μDCs, to support a number of different applications, such as in the Internet of Things (IoT) framework, in logistics and transportation, and in website tracking [13]. Typically, multiple incoming data streams[1] with Quality of Service (QoS) requirements, e.g., on latency, will be generated by data producers and distributed to different brokers, according to some criterion. After event storage at the brokers, consumer client applications asynchronously retrieve the data for processing. In a typical publish/subscribe configuration, data is pushed to the broker from the producer and pulled from the broker by the consumer.

Many logistics/production processes involve monitoring of goods, especially during the transportation between parts' suppliers and production sites. Examples of this include manufacturing processes where the final product requires the assembly of many complex and delicate component parts (see, e.g., [14], [15]). In other environments, measurements by multiple sensor nodes are collected and processed in a distributed infrastructure to provide quality control (e.g., temperature variations in the meat industry [16]). The 5G and MEC evolution

[1]We note that the term "stream" is often adopted in the terminology of data distribution platforms, like Kafka, to indicate a sequence of data packets that are related to a given set of events of a certain category (a "topic" in the publish/subscribe parlance). In the following, we will use the terms "data stream" and "flow" interchangeably.

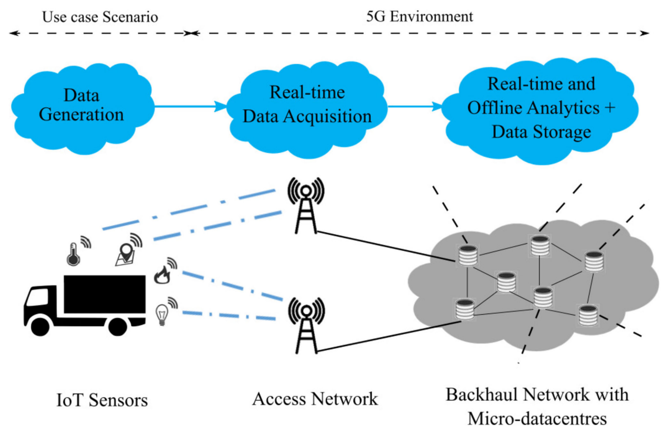

Fig. 1. DLPMA scenario with main modules and components (adapted from [17]; courtesy of BIBA – Bremer Institut für Produktion und Logistik GmbH, Bremen, Germany).

allows an unprecedented and sophisticated real-time monitoring of the goods being transported.

In this paper, we consider a general queuing model that involves packet-level processing before storage at the brokers and we model the incoming traffic generated by each flow as bursts of packets. On top of this, we build an optimisation scheme for the assignment and load balancing of incoming flows to the brokers. Incoming flows are characterised by statistical models with much longer time scales than the packet traffic they generate.

The paper is organised as follows. We describe the logistics scenario in more detail in Section II. Section III contains the mathematical problem formulation, along with the description of the control architecture and data traffic models, in the case of homogeneous traffic flows. The case of heterogeneous traffic flows with different statistical characteristics or performance requirements, which may stem, for instance, from data streams generated by different "topics," is discussed in Section IV. Section V reports our results obtained by using the proposed optimisation schemes and comparing them to a static allocation strategy independent of the traffic characteristics. The results have been obtained through both numerical evaluation and network simulation. Conclusions are drawn in Section VI.

II. A Logistic Use Case

The specific use case we consider in this paper regards a logistics scenario stemming from the MATILDA 5G PPP H2020 European Project [17], where it has been conceived as one out of five different use cases to demonstrate the project outcomes. It has been termed specifically "Distributed Logistics-Production & Maintenance Application" (DLPMA) use case. It is described in detail in one of the project deliverables [18] and its functional components are represented in Fig. 1. The general goal of the MATILDA project was to deliver a holistic and innovative fifth-generation mobile network (5G) framework to undertake the design, development and orchestration of 5G-ready vertical applications (vApps) and 5G network services over programmable infrastructures [19]. Eventually, the specific use case was not selected for a final demonstration, favouring, instead, a robotic arm control by the same project partner, but it still represents a good example of the situation we wish to model in this paper.

The scenario refers to a transport to be tracked, starting from a supplier and moving to a production facility. The goal is to monitor the loaded goods in real-time, as they are supposed to be very fragile and to need sensitive handling. To this purpose, data from sensors, such as temperature, humidity, and vibrations are transmitted in real-time, as is customary in IoT applications, in order to allow the customers to monitor their goods all over the duration of the transport. At the same time, data are collected and stored at customers' facilities for further analysis aimed at optimising the transportation process.

The data collection process at the application level is implemented through a publish/subscribe mechanism provided by a Kafka message broker and a distributed streaming platform. Kafka is the message broker which customers can subscribe to in order to consume data that are produced asynchronously and need to be made available to the various customers. The goods are monitored through the above-mentioned sensors for temperature, humidity, and vibrations. Further, a GPS-module is implemented for tracking. All sensor data are collected by a centre node on board the transport (a *producer* in Kafka terminology) that manages sensor and GPS data and sends them via 5G to the application modules implemented as cloud or fog/edge services.

The DLPMA platform provides various functionalities, such as, among others, real-time data analytics, positioning and housekeeping of goods, also based on offline analytics and past process history, which are all computationally-intensive processes.

Without going into details of the platform and of the data treatment and formats, we concentrate here on an abstract description of the arriving data streams that must be processed in real time by the analytics module. We assume that a fleet of multiple transportation means generates streams of measurement data. Moreover, actually going beyond the specific implementation considered in the MATILDA project, and with a perspective that addresses MEC applications [3], [4], we also consider a distributed edge computing scenario, where multiple μDCs may be deployed in various geographical zones traversed by the transports, as they move along toward their destinations. These μDCs may be characterised by different computational and storage resources, so that they may provide processing by Virtual Machiness (VMs) at different computational speeds. A μDC can then host different VMs which may act as brokers and also perform computational activities related with data analytics on the received data streams. Kafka actually allows partitioning the data streams among multiple brokers.

Quoting almost verbatim from the Kafka website,[2] the process can be summarized as follows: producers send data to the brokers, directing data of each specific partition to the partition leader. The alive servers and the partition leaders they host for a certain topic are discovered by their answers to specific requests for metadata issued by the producers that act as clients. The

[2][Online]. Available: https://kafka.apache.org/documentation/#theconsumer

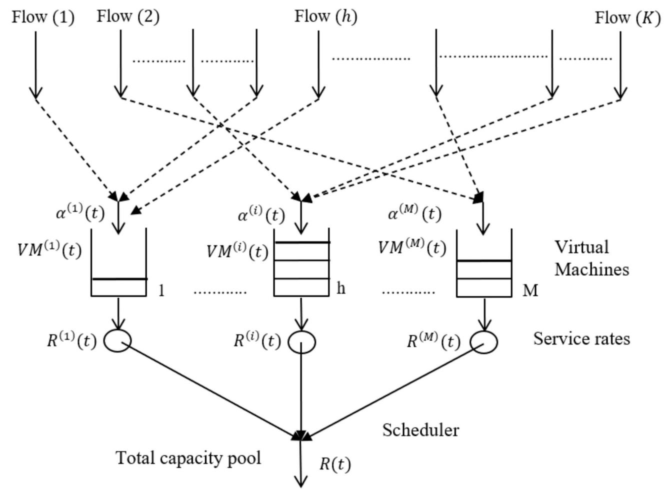

Fig. 2. Flow assignment problem.

clients can decide upon the choice of the server on the basis of a given policy; the determination of the latter is precisely our goal.

For each operational zone traversed, then, we can suppose to have the presence of VMs located in different μDCs that are available and discoverable to receive the streams generated by the transports in a certain coverage area. The on-board producer decides upon the assignment of the streams to μDCs and their specific VMs; the assignment lasts for the duration of the stream, which is composed by multiple batches of measurements to be processed regarding a certain product. Then, our goal is to provide the producing clients with a policy on the dispatching of the various flows generated by the data streams to the VMs located at Edge μDCs, in order to balance the load and minimise the overall average processing delay, including both stream handling for local storage and possible pre-processing of the data. We model the incoming traffic flow generated by each active data stream as being constituted by bursts of packets and adopt a simple but general model for the queuing systems that represent packet-level processing. Specifically, we model each VM as a processing unit, in front of which the packets generated by the assigned flows are queued and served in FIFO order. On top of this, we construct our optimisation scheme to implement the assignment and load balancing of incoming flows to the processing queues, over time periods within which they are served with constant rates. The flows originate at the transports at random instants, last for a random time period, and are characterised by statistical models with much longer time scales than the packet traffic they generate. The modelling and optimisation problem we consider here is a slight modification of the one we treated in [20] in the context of NFV.

III. Flow Modelling and Optimisation Problem Statement - Homogeneous Traffic Case

The abstract representation as a queuing system of our logistics use case with distributed storage and computation is represented in Fig. 2. Each queue corresponds to a VM, residing in a specific μDC, that may be equipped with different computational resources, depending on its location and hardware configuration. For this reason, we suppose that, in general, each VM residing in a certain μDC may have been assigned one virtual CPU (processor) characterised by a certain computational speed (processing capacity). These speeds represent the service rates $R^{(1)}(t), \ldots, R^{(M)}(t)$, satisfying $\sum_{i=1}^{M} R^{(i)}(t) = R(t)$, where M is the total number of VMs assigned to our logistic application, along with a total processing capacity pool $R(t)$.

Assuming $R^{(1)}(t), \ldots, R^{(M)}(t)$ fixed, we consider each queue with its own independent buffer in stationary conditions and so we drop the dependence on t in the following. Incoming flows are distributed among the processors on the basis of coefficients $\zeta^{(1)} \geq 0, \ldots, \zeta^{(M)} \geq 0$, $\sum_{i=1}^{M} \zeta^{(i)} = 1$, to be determined through an optimisation procedure that will be described later; for the time being, they are considered fixed. In other words, each incoming flow is assigned randomly to a processor upon its birth according to the probability distribution determined by the coefficients. The model we investigate in this section can basically correspond to the traffic generated by a single topic. More specifically, we assume that the topic collects measurements that derive from similar sensing devices that monitor the same type of physical quantity (e.g., temperature) referring to a specific application domain (e.g., the monitoring of the meat quality [16] that we mentioned in the Introduction), even if they may originate from different transportation means and even belong to different final users. Therefore, we assume the data being generated by these processes to possess similar statistical features and performance requirements. The case in which multiple different topics can give rise to differentiated flows will be modelled in the next Section. We also remark explicitly that we do not differentiate VMs according to their location in a μDC or another. Each VM represents an active server performing the brokerage and other analytical functionalities that may be required. Obviously, as the transports move across different geographical zones, all parameters in our model, in terms of number of VMs, workload generated by the data traffic and processing capacities being offered, may change. However, the time scales of such changes would be orders of magnitude larger than the ones characterising the data traffic, so that we can consider successive optimisations in quasi-stationary conditions.

To clarify the relation between our analytical queueing model with the publish-subscribe mechanism handled by Kafka, we consider in some more detail the data collection process on-board the transports, which will also form the basis for the generation of realistic simulation results to be compared with analytical ones in Section V. Given a set of on-board sensors collecting measurements data that pertain to a certain topic, we assume the data they generate to be collected by a gateway situated on the transport. Data packets arriving at the gateway are aggregated in batches (packet bursts) by a coalescing process before transmission over the wireless channel; this can be done by waiting to collect a certain number of packets before transmitting them, up to the expiration of a certain time period, as in the packet coalescing process applied in Green Ethernet cards (see, e.g., [21]). Each time the transport enters the area covered by a partition leader, a new connection is set up, representing a data stream (flow) composed by the packet bursts of such aggregated data source. Given the movement of the transports, we assume these traffic flows to be generated according to a birth-death model. Thus, the input process to each VM-operated server queue at a μDC is composed by the flows assigned by the flow distribution policy to that VM, and each flow – that remains active for the duration of the assignment, i.e., until the transport remains under the coverage area of the responding partition leader – carries packet bursts.

It should also be noted at this point that the traffic characteristics measured at the sources might be altered before arriving at the servers' queues during network traversal, owing to different paths of the 5G network and the relative parameters: delay, bandwidth, congestion, slicing, impairments, etc. However, we believe that these modifications may be mitigated in the 5G and edge computing environment for two main reasons: i) we can associate traffic slices with QoS requirements to different topics stemming from different vertical applications; ii) being the serving VMs situated in edge μDCs, the traversal of switching nodes should be minimal. In any case, we believe that the bursty nature of the traffic (packet bursts of random length) can be maintained. Variations in burst interarrival times and in the first and second moments of burst length can be estimated by monitoring the traffic arriving at the input to the queues (in the same way as the same parameters should be estimated at the sources – possible methods are, e.g., those indicated in [22]). As regards the effect of mobility, as we divide the territory into different coverage areas, transitions between adjacent areas imply the re-establishment of connections, and the possible variations of the traffic parameters within a new connection can be taken into account in the same way.

We also assume that the packet bursts within each active flow are generated according to a Poisson model with Long-Range-Dependent (LRD) burst length.[3] For each queue i, we consider the average waiting time $W^{(i)}(a^{(i)})$, with input rate $a^{(i)} = \zeta^{(i)} m\lambda\beta$ [pkts/s], $i = 1, \ldots, M$, calculated according to an $M^X/G/1$ queuing model [23], so taking into account the traffic generation at the flow level (i.e., the LRD traffic entering the queue is the aggregate of LRD traffic streams produced by the individual flows coming from different transports). The aggregate burst rate is determined by the presence of m total active flows, each with a burst generation rate equal to λ and average burst length (in packets) β. From classical $M^X/G/1$ queuing theory, the expression of the average waiting time in queue $W^{(i)}(a^{(i)})$ is given as in formula (1) below (see [23], where the expression is derived for the average queue length; the average waiting time is then obtained from Little's Theorem by dividing the average queue length by the packet input rate $a^{(i)}$). We have used the notation $W^{(i)}(a^{(i)})$ to indicate explicitly the dependence of the average waiting time on $a^{(i)}$, as defined above and, hence, on the fraction of active flows $\zeta^{(i)} m$ entering queue i; namely, the average waiting time is conditional to the number of flows m – and, as such, may be further averaged with respect to the probability distribution of the flows – and is a function of the parameter $\zeta^{(i)}$.

$$W^{(i)}(a(i)) = \frac{\rho^{(i)2}}{2\zeta^{(i)} m\lambda\beta \left(1 + \sigma^2_{S^{(i)}}/\overline{S^{(i)2}}\right)\left(1 - \rho^{(i)}\right)} + \\ + \frac{\rho^{(i)}\left(\overline{X^2}/\beta - 1\right)}{2\zeta^{(i)} m\lambda\beta\left(1 - \rho^{(i)}\right)} \qquad (1)$$

where $S^{(i)}$ is the service time depending on the amount of operations N_p to be performed per packet and on the processing speed $R^{(i)}$. $E\{S^{(i)}\} = E\{N_p\}/R^{(i)} = \overline{N_p}/R^{(i)} = 1/\mu^{(i)}$, $\overline{S^{(i)2}}$ is the mean square value and $\sigma^2_{S^{(i)}}$ the variance. $\rho^{(i)} = \zeta^{(i)} m\lambda\beta/\mu^{(i)}$ is the queue utilisation and $\overline{X^2}$ the mean square value of the burst length.

It is worth noting that more general models could be also considered. For instance, if another Key Performance Indicator (KPI), such as the energy consumption, is included to be traded off with latency, the $M^X/G/1/SET$ model could be adopted to account for set up times for processor wakeup, as done in [24] and [25] in the case of deterministic service times.

The case of flows with unequal burst generation rates can be handled in a similar way if service separation with static partitions [26] is applied: services giving rise to flows with similar statistical nature and similar requirements are grouped into classes and assigned to a subset of processors for each class. More general formulations are possible, as indicated in [26] and as briefly discussed in [20]. We consider this case explicitly in the next section.

As the time scales at the burst- and flow-level are widely different, it follows that variations in the number of flows should be considered on a much longer time scale with respect to the timing of events describing the dynamics of packets in the queue. Based on this consideration, we decide to ignore non-stationary behaviours and assume that a stationary state in the queue probabilities is reached almost instantaneously between birth and death events at the flow level. A precise analysis of a somehow related problem, based on Courtois' decomposition, can be found in [27].

Under this flow distribution strategy and homogeneous flows assumption, the same burst generation model holds for the flows assigned to each processing VM. Therefore, we can examine each queue as separate from the others, conditioned to the presence of m total flows in the system, and consider it as an $M^X/G/1$ queue.

In order to avoid instability, the following condition must be satisfied for each queue

$$\rho^{(i)} = \zeta^{(i)} m\lambda\beta/\mu^{(i)} < 1, \text{ i.e. } m^{(i)} \equiv m\zeta^{(i)} < \frac{\mu^{(i)}}{\lambda\beta} \qquad (2)$$

so that the maximum number of flows $m^{(i)}_{max}$ acceptable by queue i is equal to $\lfloor \mu^{(i)}/\lambda\beta \rfloor$, $\lfloor x \rfloor$ being the largest integer less than or equal to x.

This condition also imposes the presence of a Call Admission Control (CAC) to limit the maximum number of flows totally

[3] For the time being, we do not need to specify the exact probability distribution of the burst length. As can be seen from (1), the expression of the average queuing delay, conditional to a given number of active flows that generate burst arrivals in the queue, depends only on the first and second moments of the burst length distribution. In Section V, where we will need to generate burst arrivals for simulation results, we will adopt a Pareto distribution of the burst length with finite mean and variance.

acceptable in the system to

$$m_{max} = \sum_{i=1}^{M} \left\lfloor \frac{\mu^{(i)}}{\lambda\beta} \right\rfloor \tag{3}$$

By recalling now that the expression (1) of the average waiting time is conditional to the number of active flows generating burst arrivals in the queue, and that the flows are assumed to be described by a birth-death model, we can further average out the waiting time with respect to the distribution of the flows. Let λ_f and μ_f, respectively, be the parameters of the independent exponential distributions describing flow interarrival times and durations, and let $A_f = \lambda_f/\mu_f$ [Erlangs] denote the traffic intensity of the flows. Then, the probability $p_m^{(i)}$ that m flows are active (producing bursts) on the i^{th} VM's queue is given by an M/M/$m_{max}^{(i)}$/$m_{max}^{(i)}$ queuing model as

$$\begin{aligned} p_m^{(i)} &= p_0^{(i)} \prod_{j=0}^{m-1} \frac{\left(\zeta^{(i)} A_f\right)^j}{j+1} \\ &= \frac{\left(\zeta^{(i)} A_f\right)^m / m!}{\sum_{j=0}^{m_{max}^{(i)}} \frac{\left(\zeta^{(i)} A_f\right)^j}{j!}} \qquad m = 0, 1, \ldots, m_{max}^{(i)} \end{aligned} \tag{4}$$

Thus, we can write

$$\overline{W}^{(i)} = \frac{1}{\left(1 - p_0^{(i)}\right)} \sum_{m=1}^{m_{max}^{(i)}} p_m^{(i)} W^{(i)} \left(\zeta^{(i)} m \lambda \beta\right) \tag{5}$$

for the average delay per queue with respect to the total number of flows and considering the presence of at least one active flow at the i^{th} VM, and

$$\overline{W} = \sum_{i=1}^{M} \overline{W}^{(i)} \zeta^{(i)} \tag{6}$$

for the total average delay over all flows.

The upper limit of the sum in (5) is necessary as a consequence of condition (2).

There is a final condition to be accounted for. From (4), the blocking probabilities of each VM are given by

$$\begin{aligned} P_B^{(i)} &= p_{m_{max}^{(i)}}^{(i)} \\ &= \frac{\left(\zeta^{(i)} A_f\right)^{m_{max}^{(i)}} / m_{max}^{(i)}!}{\sum_{j=0}^{m_{max}^{(i)}} \frac{(\zeta^{(i)} A_f)^j}{j!}} \qquad i = 0, 1, \ldots, M \end{aligned} \tag{7}$$

The blocking probabilities are required to be less than a given threshold $\overline{P_B}$, assumed to be the same for all VMs. Then, an optimisation problem can be posed for the selection of the traffic spreading coefficients as

$$\min_{\substack{\zeta^{(1)} \geq 0, \ldots, \zeta^{(M)} \geq 0 \\ \sum_{i=1}^{M} \zeta^{(i)} = 1 \\ P_B^{(1)} \leq \overline{P_B}, \ldots, P_B^{(M)} \leq \overline{P_B}}} \overline{W} \tag{8}$$

IV. Flow Modelling and Optimisation Problem Statement - Heterogeneous Traffic Case

In the case in which data streams are characterised by different statistical parameters in terms of average flow and burst generation rates, and/or average flow duration, burst length, and amounts of operations per packet, and, possibly, by different performance requirements, the flow model would correspond, in general, to a stochastic knapsack [26], [28]. As suggested in [26] and already anticipated in the previous Section, in this case the most advisable and manageable model is that of Service Separation, whereby only flows with the same statistical characteristics and performance requirements are multiplexed together and feed the same queue for the VM they are assigned to, with their bursts. A reasonable way of handling the allocation of resources in this case consists of grouping flows with similar characteristics into classes and to perform per-class resource assignments. Here again, for the reasons recalled in Section III, where we have mentioned possible modifications in the statistical characteristics of the traffic caused by network traversal, the classification (and traffic parameters' estimation) should be performed at the entrance of the specific VM queues.

Then, let us consider having K such classes and let $\lambda^{(k)}$ be the packet generation rate, $\beta^{(k)}$ the average burst length, and $\overline{N_p}^{(k)}$ the average number of requested operations characterising class-k packets. The overall processing capacity resource pool of R units can be partitioned into K groups, with R_k units assigned to the k-th group, $k = 1, \ldots, K$, according to some criterion. In particular, let $\theta^{(k)}(m^{(k)})$ be a function that represents the minimum processing capacity that is required to satisfy packet-level QoS requirements for $m^{(k)}$ active class-k flows, whose generated packets are multiplexed in the same buffer. We consider here the simplest possible class of strategies for resource allocation which, by following [26], is termed *Service Separation with Static Partitions (SSSP)*; in particular, we consider the *Complete Partitioning (CP)* case, defined as follows.

Let $R_1 > 0, \ldots, R_K > 0$, with $R_1 + \cdots + R_K = R$, be a partition of the total processing capacity and let $R^{(k)}$ denote now the processing capacity dedicated to serve the buffered packets of class k, with $E\{S^{(k)}\} = 1/\mu^{(k)} = \overline{N_p}^{(k)}/R^{(k)}, k = 1, \ldots, K$.

Consequently, under CP, an arriving class-k flow is admitted iff

$$\theta^{(k)}(m^{(k)} + 1) \leq R_k \tag{9}$$

with $\theta^{(k)}(\cdot)$ corresponding to the following criterion:

$$\theta^{(k)}(m^{(k)}) = \min\{0 < R^{(k)} \leq R_k : W^{(k)}(m^{(k)}) \leq \hat{W}^{(k)}\} \tag{10}$$

where

$$\begin{aligned} W^{(k)}(m^{(k)}) = & \frac{\rho^{(k)2}}{2m^{(k)}\lambda^{(k)}\beta^{(k)}\left(1+\sigma^2_{S^{(k)}}/\overline{S^{(k)2}}\right)\left(1-\rho^{(k)}\right)} \\ & + \frac{\rho^{(k)}\left(\overline{X^{(k)2}}/\beta^{(k)} - 1\right)}{2\, m^{(k)}\lambda^{(k)}\beta^{(k)}\left(1 - \rho^{(k)}\right)} \end{aligned} \tag{11}$$

with

$$\rho^{(k)} = \frac{m^{(k)}\lambda^{(k)}\beta^{(k)}}{\mu^{(k)}} = \frac{m^{(k)}\lambda^{(k)}\beta^{(k)}\overline{N_p}^{(k)}}{R^{(k)}} \tag{12}$$

$\hat{W}^{(k)}$ being a desired upper bound on the average delay of class-k packets.

TABLE I
NUMERICAL VALUES OF THE MODEL'S PARAMETERS

$A_f = 10$ [Erlangs]	$\lambda = 10$ [bursts/s]
$\beta = 1.5$ [pkts/burst]	$\overline{P_B} \leq 0.1$
$\overline{X^2} = 3$	$\alpha^{(i)} = 10 \quad i = 1, \ldots, M$
$R^{(i)} = 2,000,000 - i \cdot 200,000$ [opers/s] $\quad i = 1, \ldots, M$	$\overline{N_p} = 1000$ [opers/pkt]
$\mu^{(i)} = R^{(i)} / \overline{N_p} \quad i = 1, \ldots, M$	$\delta^{(i)} = (\alpha^{(i)} - 1)/(\alpha^{(i)} \cdot \mu^{(i)}) \quad i = 1, \ldots, M$
$\overline{S^{(i)2}} = (\delta^{(i)2} \cdot \alpha^{(i)})/(\alpha^{(i)} - 2) \quad i = 1, \ldots, M$	$\sigma^2_{S^{(i)}} = \overline{S^{(i)2}} - 1/\mu^{(i)2} \quad i = 1, \ldots, M$

Conditions $W^{(k)}(m^{(k)}) \leq \hat{W}^{(k)}$, for $k = 1, \ldots, K$, define the so called "feasibility region" or "schedulable region," i.e., the region in the space of traffic loads $m^{(k)}\lambda^{(k)}\beta^{(k)}$ [pkts/s] (or $m^{(k)}\lambda^{(k)}\beta^{(k)}\overline{N_p}^{(k)}$ "computational units"/s) of all classes within which the desired packet-level QoS requirements are satisfied. It is worth noting that in all cases, like the present one, where an analytical model is available, the feasibility region under Service Separation is easily computable. In other words, the availability of an analytical packet-level model makes relatively easy to define an analytically expressible packet-level criterion and naturally lends a notion of capacity of the underlying statistical multiplexer which allows a clear definition of the region over which the flows of the various classes can range.

Given the presence of a CAC, there is, actually, another performance index that might become of interest in this case: the blocking probability of flows, also identified as Grade of Service (GoS). In the CP case, the blocking probabilities at individual queues are easily calculated, similarly to what has been done in the preceding section: the queuing model outlined above for the flow level would indeed be of type $M/M/m^{(k)}_{max}(R_k)/m^{(k)}_{max}(R_k)$, $m^{(k)}_{max}(R_k)$ being the maximum number of acceptable flows as a function of R_k

$$m^{(k)}_{max} = \left\lfloor \frac{\mu^{(k)}_{max}}{\lambda^{(k)}\beta^{(k)}} \right\rfloor = \left\lfloor \frac{R_k}{\overline{N_p}^{(k)}\lambda^{(k)}\beta^{(k)}} \right\rfloor \tag{13}$$

The blocking probabilities so just correspond to the Erlang B formula

$$\begin{aligned} P^{(k)}_B(R_k) &= EB\left[A^{(k)}_f, m^{(k)}_{max}(R_k)\right] \\ &= \frac{\left(A^{(k)}_f\right)^{m^{(k)}_{max}(R_k)} / m^{(k)}_{max}(R_k)!}{\sum_{j=0}^{m^{(k)}_{max}(R_k)} \frac{\left(A^{(k)}_f\right)^j}{j!}} \end{aligned} \tag{14}$$

Then, a general optimisation criterion at the flow level can be that of minimising an overall index of the type $\overline{P}_B(R_1, \ldots, R_K) = \sum_{k=1}^{K} P^{(k)}_B(R_k)$, or $P^{max}_B(R_1, \ldots, R_K) = \max P^{(k)}_B(R_k)$, $k = 1, \ldots, K$, with respect to the number of active processors and their allocation among classes, under given low-level constraints on delay and, possibly, on power consumption, if we want to add this KPI to the optimisation, by suitably changing the queuing models. In our numerical results in the next Section we will adopt the first criterion, i.e., we will seek

$$\min_{\substack{R_1>0,\ldots,R_K>0 \\ \sum_{j=1}^{K} R_j = R}} \overline{P}_B(R_1, \ldots, R_K) \tag{15}$$

It is worth noting that CP is a simple resource allocation strategy for the minimization of a flow-level criterion like the average or the maximum blocking probability, where the functional form of the strategy is fixed a priori, and the optimisation problem becomes a parametric one like (15). Its rationale relies principally on the fact that it allows excluding portions of the feasibility regions where a certain class might be greatly privileged with respect to the others.

Other choices are possible and are extensively discussed in [26]. In general, the optimum (unconstrained) functional form of the resource allocation strategy is difficult to obtain, though some properties that allow restricting it can be found [29], [30].

V. PERFORMANCE EVALUATION

We present and comment here numerical results for the evaluation of the proposed method. They have been obtained by using the optimisation tools available in the Python library Scipy,[4] and, in particular, the SLSQP (Sequential Least SQuares Programming) optimisation method.

Table I summarises the numerical values of the considered reference scenario.

We have assumed a continuous approximation of the burst length, with a Pareto distribution with location parameter $\delta = 1$ and shape parameter $\alpha = 3$. Besides, we have assumed a Pareto distribution of the service time of each VM with shape parameter $\alpha^{(i)}$ and location parameter $\delta^{(i)}$ as reported in Table I.

A. Homogeneous Traffic Case

Different tests have been performed. Their aim was to set the $\zeta^{(i)}$ values in order to minimise the total average delay $\overline{W}$ over all traffic flows by considering the problem defined in (8). The proposed strategy has been assessed in different traffic flow conditions and with a different number of VMs in the scenario.

Figs. 3 and 4 show the values of the coefficients $\zeta^{(i)}$ obtained as the output of the optimisation tool considering a different number of VMs (M) from 2 to 5 and by varying the burst arrival rate λ in the range [5,100] with discrete steps of 5 bursts/s and the flow traffic intensity A_f in the same range, respectively. In both cases, the parameters $R^{(i)}$, $\mu^{(i)}$, $\delta^{(i)}$, $\overline{S^{(i)2}}$, and $\sigma^2_{S^{(i)}}$ will assume numerical values in accordance with the related equations in Table I.

The trends of the coefficients $\zeta^{(i)}$ show that at high traffic the proposed strategy tends to distribute the incoming traffic flows to all the available VMs proportionally to their processing speeds $R^{(i)}$. Indeed, at higher λ and A_f values, each $\zeta^{(i)}$ gets closer and closer to a sort of asymptotic value which is $R^{(i)} / \sum_{j=1}^{M} R^{(j)}$. Instead, at low traffic, the proposed strategy steers higher percentages of incoming traffic flows than the

[4] www.scipy.org

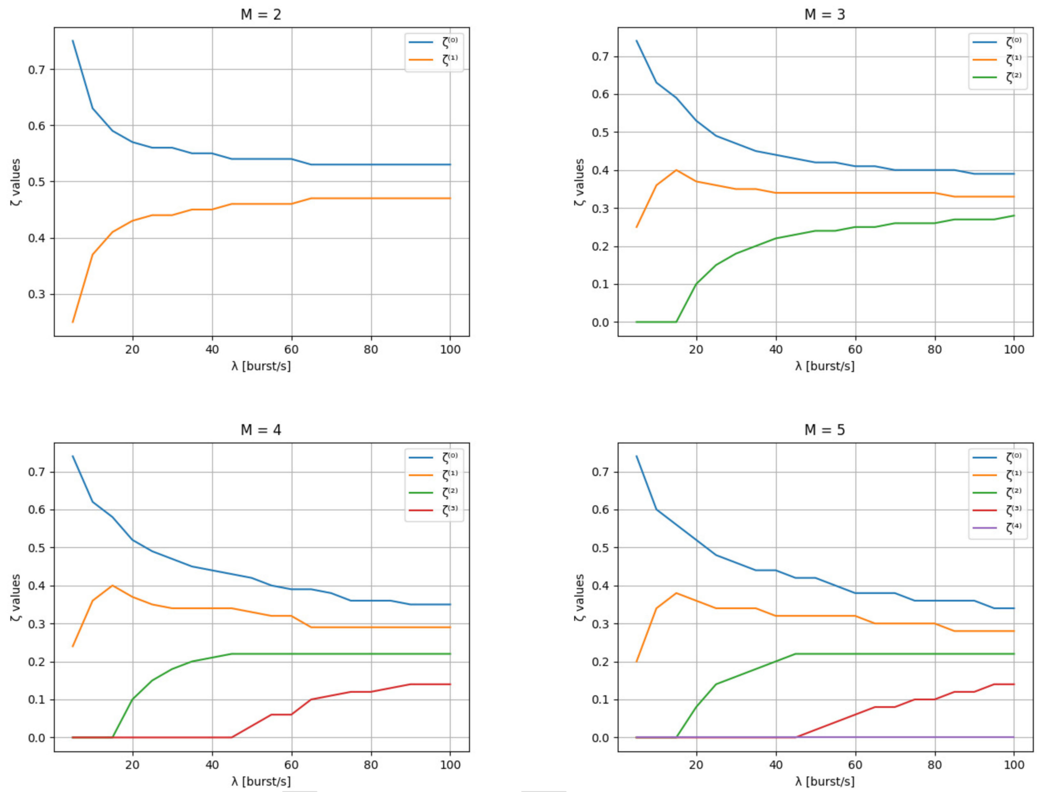

Fig. 3. Values of the coefficients $\zeta^{(i)}$ obtained by considering different numbers of VMs M from 2 to 5 and varying the value of λ in the range [5,100] [bursts/s].

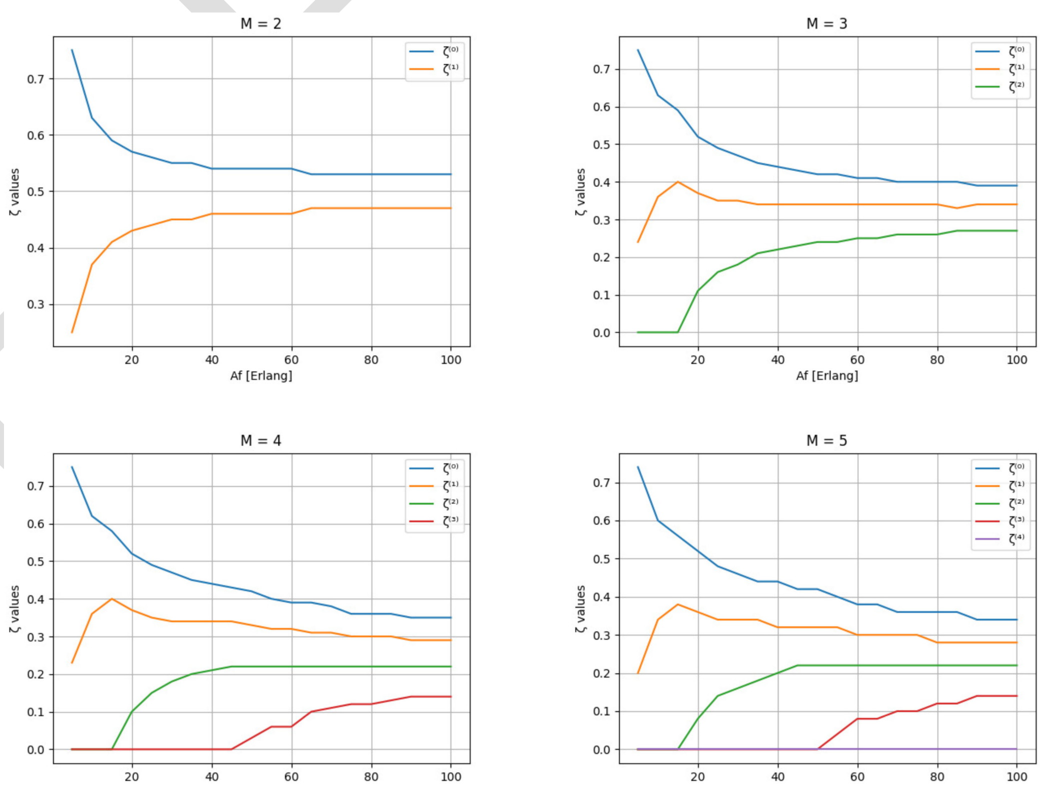

Fig. 4. Values of the coefficients $\zeta^{(i)}$ obtained by considering different numbers of VMs M from 2 to 5 and varying the value of A_f in the range [5,100] [Erlang].

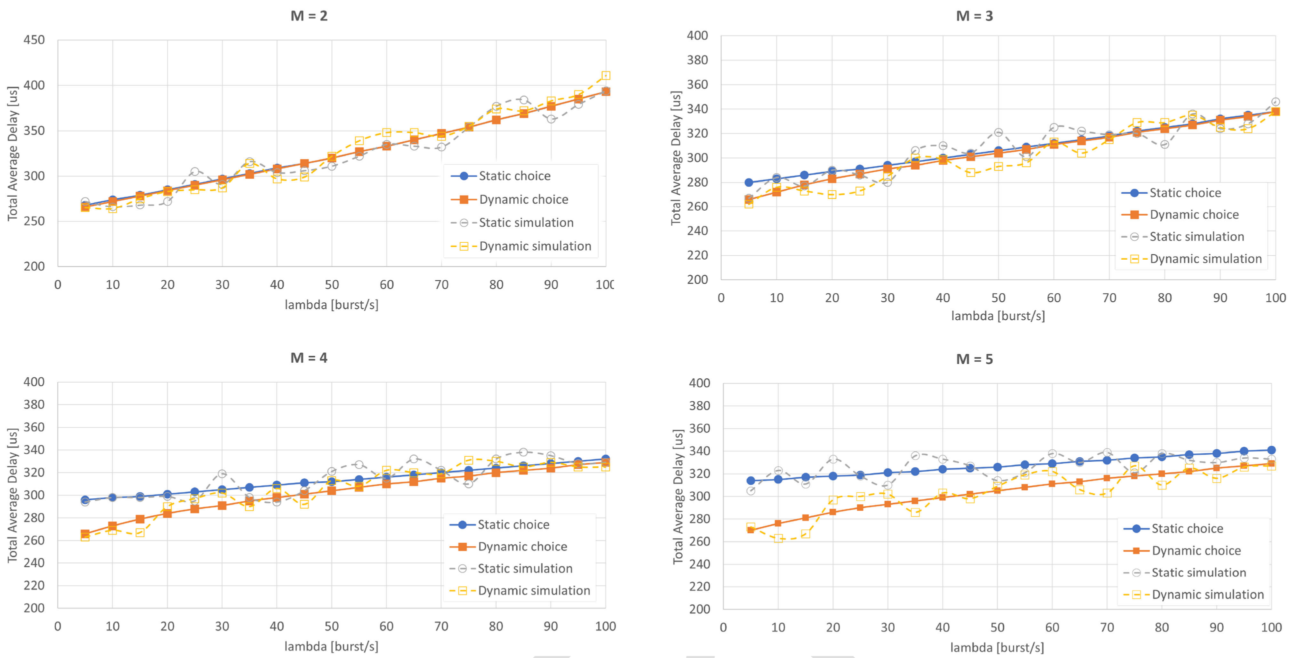

Fig. 5. Total average delay over all traffic flows $\overline{W}$ obtained by considering different numbers of VMs M from 2 to 5 and varying the value of λ in the range [5,100] [bursts/s]: comparison among the four considered cases (dynamic vs static choice and optimisation tool vs network simulator).

related asymptotic values to the VMs with higher processing speed. The main reason is that when λ and A_f are small, the queues of the "fastest" VMs do not increase significantly. They are almost able to process each single burst before the arrival of the next one, so they are almost always the best choice to reduce the obtained delay. In the extreme case that the processing speed of one VM is higher than the arrival rate of the corresponding bursts, the VM's buffer will be always almost empty and the presence of other VMs would not give additional benefits to the system in terms of lower delay. This aspect is also the reason why the system automatically and gradually "enables" more VMs with increasing values of λ and A_f. This behaviour can be seen by looking at Figs. 3 and 4 with M greater than 2, up to the case when even the highest considered values of λ and A_f are not high enough to let the system "enable" the "slowest" VM (with $M = 5$).

Concerning the total average delay over all traffic flows $\overline{W}$, i.e., the performance index to be minimised, we decided to compare the results obtained by using the optimisation tool with others obtained through a more realistic network simulation. We used the software Network Simulator 3 (NS3) to simulate the scenario depicted in Fig. 1. In detail, we simulated a network composed of 10 moving trucks as data sources generating data flows and, within each of them, burst data packets with the same statistical distributions and the same numerical values reported in Table I.

Fig. 5 shows the $\overline{W}$ values obtained with different numbers of VMs M from 2 to 5 and by varying the burst arrival rate λ in the range [5,100] [bursts/s] with discrete steps of 5 bursts/s. Each shown value is a mean value obtained by executing the same simulation for 20 rounds. The four trends are related to:

- *Dynamic Choice:* the coefficients $\zeta^{(i)}$ are dynamically computed by considering the optimisation problem defined in (8) and the results obtained through the Python-based optimisation tools.
- *Static Choice:* the coefficients $\zeta^{(i)}$ are statically set depending only on the processing speeds $R^{(i)}$, i.e., $\zeta^{(i)} = R^{(i)} / \sum_{j=1}^{M} R^{(j)}$, and the results obtained through the Python-based tools, by using analytical calculations.
- *Dynamic Simulation:* coefficients $\zeta^{(i)}$ dynamically computed but results obtained through the network simulator.
- *Static Simulation:* coefficients $\zeta^{(i)}$ statically set but results obtained through the network simulator.

The obtained total average delay grows with increasing λ. The difference between the two considered strategies decreases, which is due to the trend of the values of coefficients $\zeta^{(i)}$ exhibited in the optimisation procedure. At low traffic, the $\zeta^{(i)}$ values set with the static choice differ from the ones set with the dynamic choice. At high traffic, the static values converge to the asymptotic values of the dynamic and traffic-dependent choice, and so the $\overline{W}$ trends get closer to each other. The results obtained by using the optimisation tool and the network simulator follow the same trend with a deviation within $\pm 5\%$, confirming the reliability of the proposed model also in a more realistic test environment.

Other tests have been performed with constant $\lambda = 10$ [bursts/s] by varying the traffic intensity of the flows A_f in the range [5,100] [Erlangs] with discrete steps of 5 Erlangs. Results have been obtained in the four considered cases and are shown in Fig. 6. Their trends are similar to the ones obtained changing the λ values for the same reason.

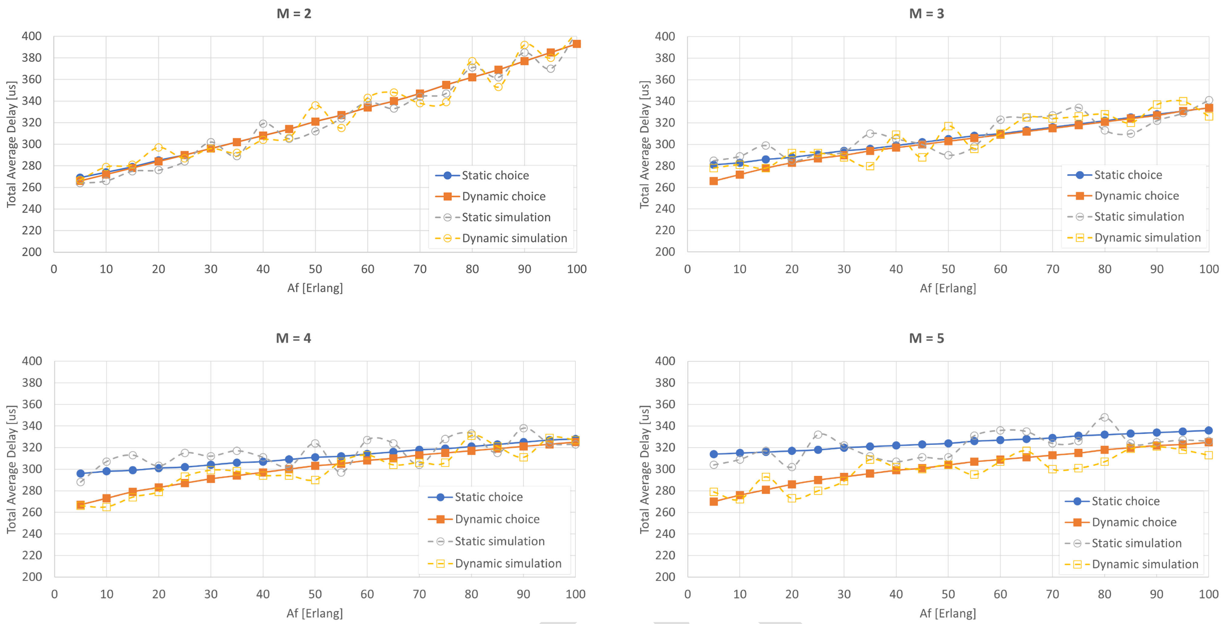

Fig. 6. Total average delay over all traffic flows $\overline{W}$ obtained by considering different number of VMs (M) from 2 to 5 and changing the value of A_f in the range [5,100] [Erlangs] to the maximum possible value to have a feasible solution: comparison among the four considered cases (dynamic vs static choice and optimisation tool vs network simulator).

Unpleasantly, we were not able to retrieve any real data traces from a real case logistic scenario to test our model by using real data flows as input. However, we tried to overcome this issue by using a hybrid approach based on both the optimisation tool and the network simulator. At the first step, we used the simulated network configuring the data sources to generate packets with different statistics than the ones considered in the analytical model. In detail, we considered a variable number of moving trucks, each of them equipped with a variable number of different sensors that generate one data packet every 10 seconds. Each truck moves within a predefined urban scenario, as shown in Fig. 7, from a point A, representing the truck storage station, to a point B, representing its arrival point, such as a shop or a warehouse. All trucks start their journey from the same point A but each of them has a different destination point B.

All trucks do not become active and moving at the same time, but they subsequently leave point A every 300 [s] to reach their unique destinations. All paths have different length and the measured average time the trucks need to reach their destinations is 3000 [s]. In this way, each truck represents a traffic flow that is active only while the truck is moving, and so 300 and 3000 [s] are the flow interarrival time and the average flow duration, respectively. The scenario is also composed of a node representing a terrestrial and fixed gateway to collect the packets from all the trucks and a node representing the μDC system where the packets are processed. All trucks aggregate a certain and fixed amount of packets generated by each on-board sensor before transmitting them to the gateway. In this way, we obtained that different packet bursts are affected by different generation delays, mainly related to the time to wait inside the truck before transmission and to the different propagation times due to the different distances between truck and gateway. After this step,

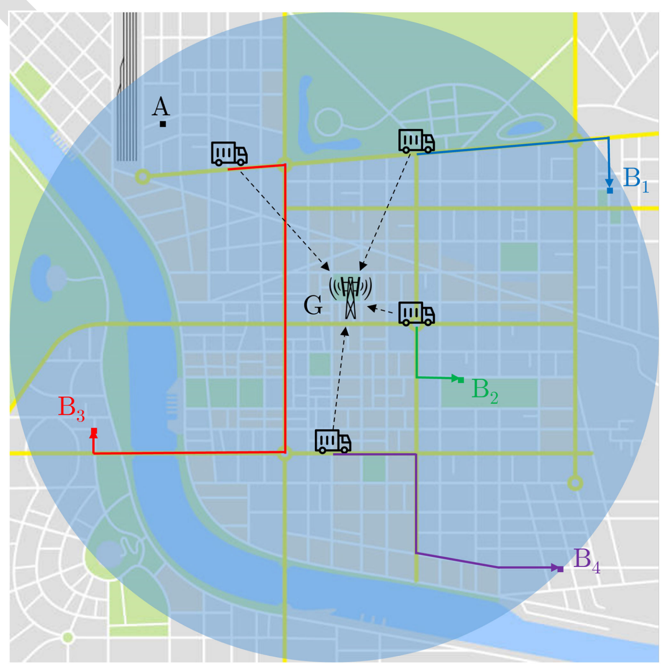

Fig. 7. Example of the considered urban map within the network simulator.

we measured the numerical values of the model's parameters, such as the queue service time and related statistics, directly within the simulator, and we used these measured values as input to the optimisation tool to assess the feasibility of the proposed approach even in case of a different and more realistic traffic flow configuration.

Figs. 8 and 9 compare the results in terms of the Total Average Delay measured within the network simulator and the

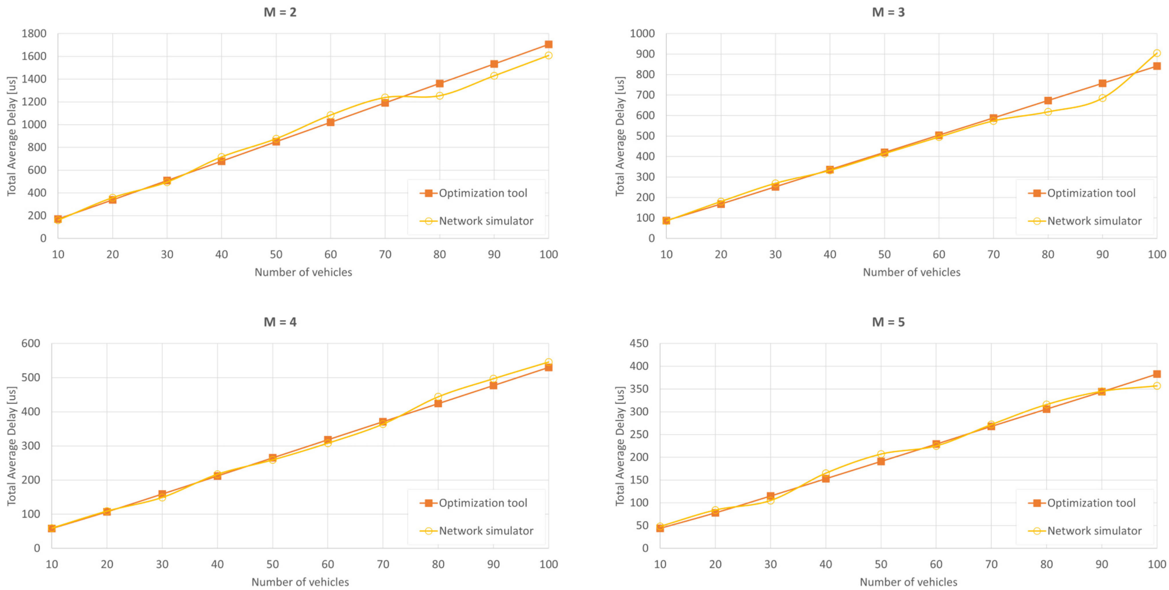

Fig. 8. Total average delay over all traffic flows $\overline{W}$ obtained by considering different number of VMs (M) from 2 to 5 and changing the number of vehicles within the considered network from 10 to 100: comparison among the results obtained with the optimisation tool and the network simulator.

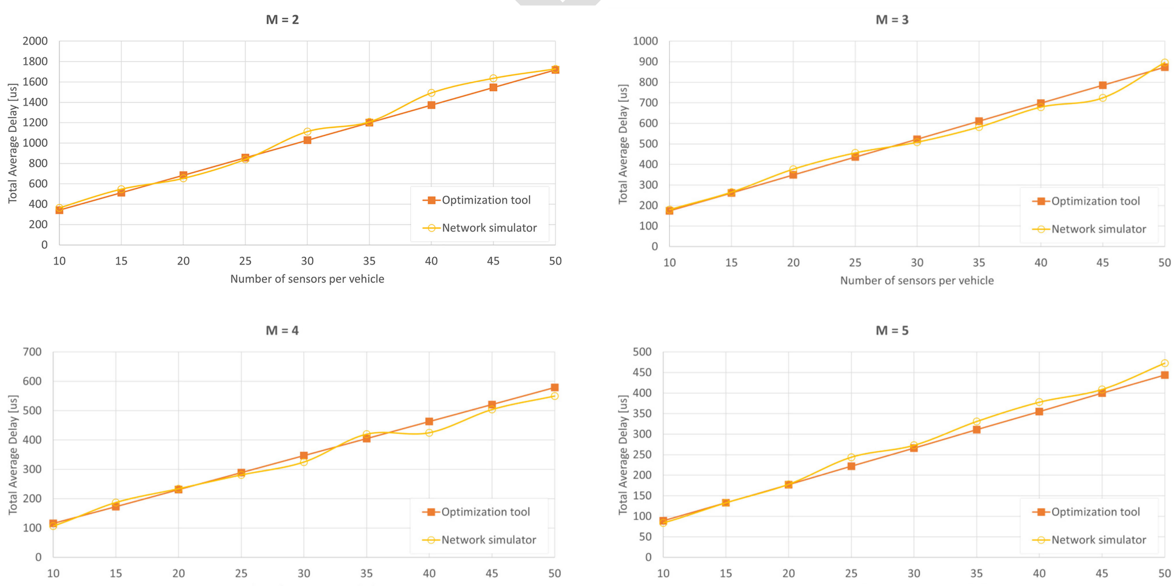

Fig. 9. Total average delay over all traffic flows $\overline{W}$ obtained by considering different number of VMs (M) from 2 to 5 and changing the number of sensors within each vehicle of the considered network from 10 to 50: comparison among the results obtained with the optimisation tool and the network simulator.

ones computed by the optimisation tool. We made different tests by changing the number of trucks in the network from 10 to 100 with 10 trucks steps, 10 sensors per truck, and the number of sensors within each truck from 10 to 50 with 5 sensors steps, 10 trucks in the network, in order to analyse which is the impact on the obtained performance with modifications that affect multiple and different parameters of the proposed model.

These data further confirm the reliability of the proposed model showing a deviation between the results obtained with the network simulator and the optimisation tool set with the statistic information measured within the simulator within $\pm 10\%$.

B. Heterogeneous Traffic Case

An additional analysis has been performed by considering the presence of traffic flows with different statistical characteristics. In this case, the overall available computational capacity $R = 8,000,000$ [opers/s] is divided into K computational resource units with capacity R_k properly sized to minimise the mean (over the classes) blocking probability. Results have been obtained by changing two of the traffic flow parameters. In detail, Figs. 10 and 11 show how the R_k values change by varying the number of considered traffic flow classes K along with λ and A_f values of one of the classes, while keeping

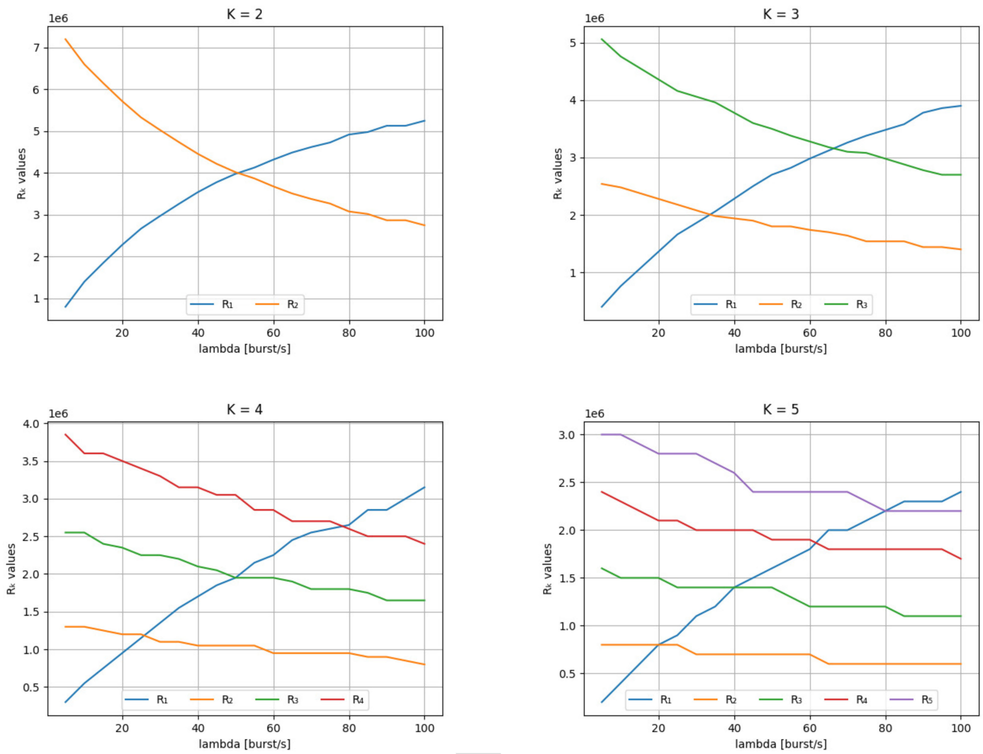

Fig. 10. Processing capacity of the traffic flow classes obtained by considering different number of classes K from 2 to 5 and changing the value of λ of the first class in the range [5,100] [bursts/s].

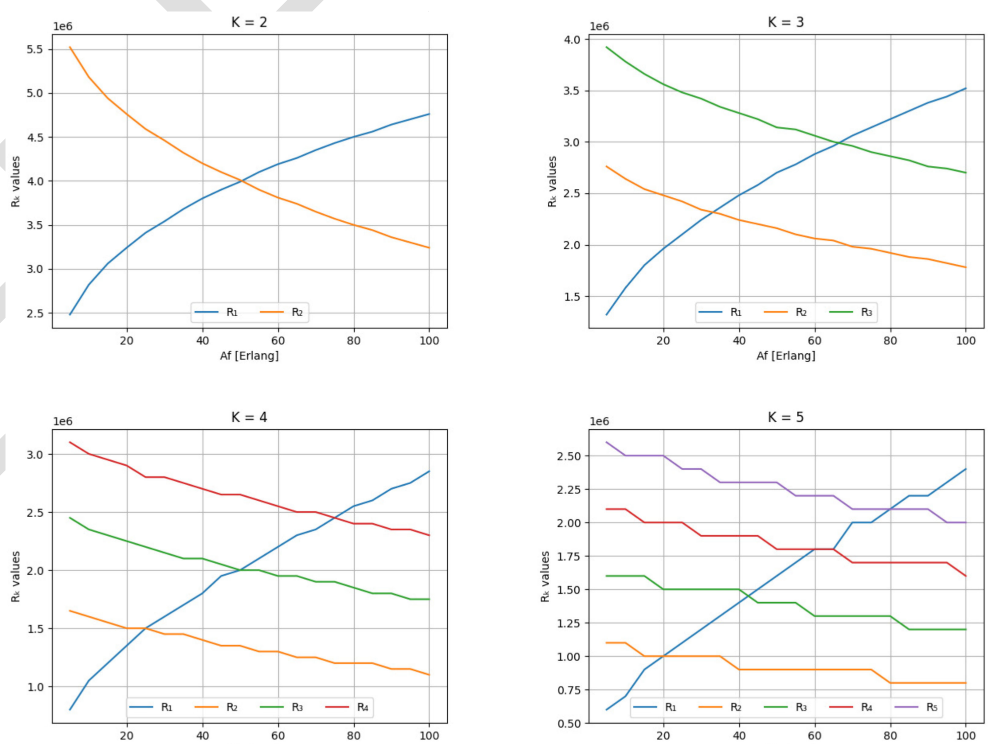

Fig. 11. Processing capacity of the traffic flow classes obtained by considering different number of classes K from 2 to 5 and changing the value of A_f of the first class in the range [5,100] [Erlangs].

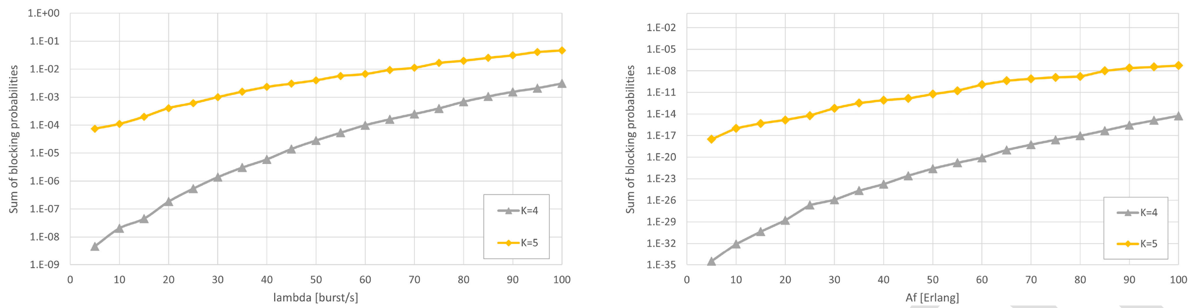

Fig. 12. Sum of blocking probabilities $\overline{P}_B(R_1, \ldots, R_K)$ obtained with $K = 4$ and $K = 5$ and varying the value of λ in the range [5,100] [bursts/s] and the value of A_f in the range [5,100] [Erlangs].

TABLE II
NUMERICAL VALUES OF λ WITH HETEROGENEOUS FLOWS (WITH FIXED $A_f = 10$ [ERLANGS])

K	λ [bursts/s]
2	[variable, 50]
3	[variable, 33, 66]
4	[variable, 25, 50, 75]
5	[variable, 20, 40, 60, 80]

TABLE III
NUMERICAL VALUES OF A_f WITH HETEROGENEOUS FLOWS (WITH FIXED $\lambda = 10$ [BURSTS/S])

K	A_f [Erlang]
2	[variable, 50]
3	[variable, 33, 66]
4	[variable, 25, 50, 75]
5	[variable, 20, 40, 60, 80]

the other ones fixed. Tables II and III show the values set for these two parameters in the performed tests with different K values. The values of the other variables have been kept as indicated in Table I, where the index i, $i = 1, \ldots, M$, would be now substituted by the index k, $k = 1, \ldots, K$, ranging over the classes.

The results obtained in the tests with heterogeneous traffic flows confirm the same trend shown in the previous results and the validity of the proposed solution. When the flow traffic intensity or the burst generation rate of one class increase, the system automatically allocates a bigger portion of the overall available computational resources to that class, consequently lowering the other classes' portions accordingly.

Fig. 12 shows the trends of the sum of the blocking probabilities of all K classes $\overline{P}_B(R_1, \ldots, R_K)$ obtained with $K = 4$ and $K = 5$ and by varying λ and A_f values of the first class in the same range as in all previous results, while keeping fixed the other classes' values as reported in Tables II and III.

The $\overline{P}_B(R_1, \ldots, R_K)$ results show increasing trends by increasing both considered λ and A_f values, which is in line with what expected, i.e., with higher traffic also the probability that the system will not be able to satisfy all the incoming flows is higher. We decided to show only the results obtained with $K = 4$ and $K = 5$, because the ones obtained with $K = 2$ and $K = 3$ assume values too low to be significant.

VI. CONCLUSION

The advent of 5G and MEC has enabled much greater capabilities for real-time monitoring and optimisation in many application areas, including logistics and transport. Leveraging on these technologies, we have considered an optimisation problem for the dispatching of analytic and decision support calculations in μDCs located at the edge (access and backhaul networks). The considered system is based on IoT real-time data collected by a set of goods being transported from supply centres to production facilities. Our emphasis has been centred on the operational complexity, represented by the statistical distribution of the number of operations per second to be performed on the data, and on the statistical nature of the network traffic generated by sensor measurements. We have defined two optimisation problems based on this modelling scheme, aimed at minimising the overall average delay in the system's response. The models stem from a real transportation scenario which has been derived from one of the MATILDA project's use cases. Results obtained through a numerical evaluation have shown a reasonable behaviour of the optimised solution based on the model's parameters, which is also confirmed by the results obtained by using a network simulator. Future work will consider the identification and adaptation of the network simulator on the basis of measurements derived from real sensor-generated data.

ACKNOWLEDGMENT

The authors would like to thank BIBA GmbH, Bremen, Germany, for the permission to refer to their logistics use case.

REFERENCES

[1] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[2] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surv. Tut.*, vol. 18, no. 1, pp. 236–262, Jan.–Mar. 2016.

[3] European Telecommunications Standards Institute, "Multi-access edge computing (MEC); framework and reference architecture," ETSI GS MEC 003 v2.2.1, 2020.

[4] F. Giust *et al.*, "MEC deployments in 4G and evolution towards 5G," *ETSI White Paper*, vol. 24, pp. 1–24, 2018.

[5] T. Koketsu Rodrigues, J. Liu, and N. Kato, "Offloading decision for mobile multi-access edge computing in a multi-tiered 6G network," *IEEE Trans. Emerg. Topics Comput.*, to be published, doi: 10.1109/TETC.2021.3090061.

[6] A. Manzalini *et al.*, "Towards 5G software-defined ecosystems: Technical challenges, business sustainability and policy issues," IEEE SDN White paper, pp. 1–16, 2016. [Online]. Available: https://discovery.ucl.ac.uk/id/eprint/10043678/

[7] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp. 1529–1541, Jul.–Sep. 2021.

[8] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11158–11168, Nov. 2019.

[9] F. Tang, Y. Zhou, and N. Kato, "Deep reinforcement learning for dynamic uplink/downlink resource allocation in high mobility 5G HetNet," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 12, pp. 2773–2782, Dec. 2020.

[10] X. Chen, W. Li, S. Lu, Z. Zhou, and X. Fu, "Efficient resource allocation for on-demand mobile-edge cloud computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8769–8780, Sep. 2018.

[11] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2092–2104, Feb. 2020.

[12] Apache Kafka, "Kafka: A distributed streaming platform," Accessed: May 10, 2022. [Online]. Available: https://kafka.apache.org/

[13] Y. Qiao, Z. Xing, Z. M. Fadlullah, J. Yang, and N. Kato, "Characterizing flow, application, and user behavior in mobile networks: A framework for mobile Big Data," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 40–49, Feb. 2018.

[14] L. Barreto, A. Amaral, and T. Pereira, "Industry 4.0 implications in logistics: An overview," *Procedia Manuf.*, vol. 13, pp. 1245–1252, 2017.

[15] D. G. Pascual, P. Daponte, and U. Kumar, *Handbook of Industry 4.0 and SMART Systems*. Boca Raton, FL, USA: CRC Press, 2019.

[16] N. Yamani and A. Al-Anbuky, "Neuro wireless sensor network architecture: Cool stores dynamic thermal mapping," in *Proc. IEEE Sensors Appl. Symp.*, 2011, pp. 45–50.

[17] MATILDA, "A holistic, innovative framework for design, development and orchestration of 5G-ready applications and network services over sliced programmable infrastructure," Accessed: May 10, 2022. [Online]. Available: https://www.matilda-5G.eu

[18] "Industry 4.0 smart factory implementation report – first demonstration phase," MATILDA project Deliverable D6.5, Available upon request, 2019.

[19] R. Bruschi, J. F. Pajo, F. Davoli, and C. Lombardo, "Managing 5G network slicing and edge computing with the MATILDA telecom layer platform," *Comput. Netw.*, vol. 194, 2021, Art. no. 108090.

[20] F. Davoli, M. Marchese, and F. Patrone, "Flow assignment in multi-core network processors," in *Proc. Adv. Optim. Decis. Sci. Soc., Serv. Enterprises*, 2019, pp. 493–503.

[21] A. Chatzipapas and V. Mancuso, "An M/G/1 model for gigabit energy efficient ethernet links with coalescing and real-trace-based evaluation," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2663–2675, Oct. 2016.

[22] R. Bolla, R. Bruschi, F. Davoli, and J. F. Pajo, "A model based approach towards real-time analytics in NFV infrastructures," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 2, pp. 529–541, Jun. 2020.

[23] H. C. Tijms, *A First Course in Stochastic Models*. Hoboken, NJ, USA: Wiley, 2003.

[24] R. Bolla, R. Bruschi, A. Carrega, and F. Davoli, "Green networking with packet processing engines: Modeling and optimization," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 110–123, Feb. 2014.

[25] R. Bolla, R. Bruschi, A. Carrega, F. Davoli, and J. F. Pajo, "Corrections to: "Green networking with packet processing engines: Modeling and optimization"," *IEEE/ACM Trans. Netw.*, to be published, doi: 10.1109/TNET.2017.2761892.

[26] K. Ross, *Multiservice Loss Models for Broadband Telecommunication Networks*. Berlin, Germany: Springer, 1995.

[27] S. Ghani and M. Schwartz, "A decomposition approximation for the analysis of voice/data integration," *IEEE Trans. Commun.*, vol. 42, no. 7, pp. 2441–2452, Jul. 1994.

[28] J. Kaufman, "Blocking in a shared resource environment," *IEEE Trans. Commun.*, vol. 29, no. 10, pp. 1474–1481, Oct. 1981.

[29] M. Cello, G. Gnecco, M. Marchese, and M. Sanguineti, "Optimality conditions for coordinate-convex policies in CAC with nonlinear feasibility boundaries," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1363–1377, Oct. 2013.

[30] M. Cello, G. Gnecco, M. Marchese, and M. Sanguineti, "Narrowing the search for optimal call-admission policies via a nonlinear stochastic knapsack model," *J. Optim. Theory Appl.*, vol. 164, no. 3, pp. 819–841, 2015.

Franco Davoli (Life Senior Member, IEEE) is currently a Professor Emeritus with the Department of Electrical, Electronic, and Telecommunications Engineering, and Naval Architecture (DITEN) of the University of Genoa, Genoa, Italy. In 2004 and 2011, he was a Visiting Erskine Fellow with the University of Canterbury, Christchurch, New Zealand. His research interests include dynamic resource allocation in multiservice networks and in the future Internet, wireless mobile 5G/6G and satellite networks, multimedia communications and services, and in flexible, programmable, and energy-efficient networking. He has coauthored more than 380 scientific publications in international journals, book chapters, and conference proceedings. He was a Principal Investigator in a large number of projects and has served in several positions in the Italian National Consortium for Telecommunications, an independent organization joining 38 universities all over Italy. He was a Co-founder and the Head for the term 2003–2004, of the CNIT National Laboratory for Multimedia Communications, Naples, Italy, and the Vice-President of the CNIT Management Board during 2005–2007. He is currently the Head of the CNIT National Laboratory of Smart and Secure Networks, based in Genoa, Italy, and a coordinator of the H2020 5G PPP 5G-INDUCE project.

Mario Marchese (Senior Member, IEEE) was born in Genoa, Italy, in 1967. He received the Laurea degree (*cum laude*) and the Ph.D. degree in telecommunications from the University of Genoa, Genoa, Italy, in 1992 and 1997, respectively. From 1999 to January 2005, he was with the Italian Consortium for Telecommunications (CNIT), University of Genoa Research Unit, where he was the Head of Research. From February 2005 to January 2016, he was an Associate Professor with the University of Genoa. Since February 2016, he has been a Full Professor with the University of Genoa. He is the author of the book *Quality of Service over Heterogeneous Networks*, (John Wiley & Sons, Chichester, 2007), and author or coauthor of more than 300 scientific works, including international journals, international conferences, and book chapters. His main research interests include networking, quality of service over heterogeneous networks, software defined networking, satellite DTN and nano-satellite networks, networking security. From 2006 to 2008, he was the Chair of the IEEE Satellite and Space Communications Technical Committee. He is the winner of the IEEE ComSoc 2008 Satellite Communications Distinguished Service Award in recognition of significant professional standing and contributions in the field of satellite communications technology.

Fabio Patrone (Member, IEEE) was born in Genoa, Italy, in 1988. He received the bachelor's and master's degree in telecommunication engineering from the University of Genoa, Genoa, Italy, in 2010 and 2013, respectively, and the Ph.D. degree at the Satellite Communications and Networking Laboratory, with a thesis on routing and scheduling algorithms in satellite delay and disruption tolerant networks. He is currently an Assistant Professor with the University of Genoa. His main research interests include satellite networks and DTN networks, in particular design of routing, scheduling, and congestion control algorithms for satellite communication (SatCom) networks, study of integration solutions between SatCom and terrestrial networks within the 5G framework, exploiting new networking paradigms, such as software defined networking and network function virtualization, study of Internet of Things solutions based on satellite/aerial networks.