



**HAL**  
open science

## Discontinuous Computation Offloading for energy-efficient mobile edge computing

Mattia Merluzzi, Nicola Di Pietro, Paolo Di Lorenzo, Emilio Calvanese  
Strinati, Sergio Barbarossa

► **To cite this version:**

Mattia Merluzzi, Nicola Di Pietro, Paolo Di Lorenzo, Emilio Calvanese Strinati, Sergio Barbarossa. Discontinuous Computation Offloading for energy-efficient mobile edge computing. IEEE Transactions on Green Communications and Networking, 2022, 6 (2), pp.1242-1257. 10.1109/TGCN.2021.3125543 . cea-04870258

**HAL Id: cea-04870258**

**<https://cea.hal.science/cea-04870258v1>**

Submitted on 13 Jan 2025

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Copyright

# Discontinuous Computation Offloading for Energy-Efficient Mobile Edge Computing

Mattia Merluzzi, *Member, IEEE*, Nicola di Pietro, Paolo Di Lorenzo, *Senior Member, IEEE*,  
Emilio Calvanese Strinati, *Member, IEEE*, Sergio Barbarossa, *Fellow, IEEE*

**Abstract**—We propose a novel strategy for energy-efficient dynamic computation offloading, in the context of edge-computing-aided beyond 5G networks. The goal is to minimize the energy consumption of the overall system, comprising multiple User Equipment (UE), an access point (AP), and an edge server (ES), under constraints on the end-to-end service delay and the packet error rate performance over the wireless interface. To reduce the energy consumption, we exploit low-power sleep operation modes for the users, the AP and the ES, shifting the edge computing paradigm from an *always on* to an *always available* architecture, capable of guaranteeing an on-demand target service quality with the minimum energy consumption. To this aim, we propose an online algorithm for dynamic and optimal orchestration of radio and computational resources called *Discontinuous Computation Offloading (DisCO)*. In such a framework, end-to-end delay constraints translate into constraints on overall queuing delays, including both the communication and the computation phases of the offloading service. DisCO hinges on Lyapunov stochastic optimization, does not require any prior knowledge on the statistics of the offloading traffic or the radio channels, and satisfies the long-term performance constraints imposed by the users. Several numerical results illustrate the advantages of the proposed method.

**Index Terms**—Edge Computing, Beyond 5G, Green Networking, Computation Offloading, Energy Efficiency.

## I. INTRODUCTION

With the advent of beyond 5G networks [1], [2], mobile communication systems are evolving from a pure communication framework to service enablers, building on the tight integration of communication, computation, caching, and control functionalities [3], [4]. Indeed, future networks will serve a plethora of new applications, not only addressed to mobile end users, but also for whole different sectors (*verticals*), such as Industry 4.0, Internet of Things (IoT), autonomous driving, remote surgery, Artificial Intelligence (AI) etc. These new services have very different requirements and they generally involve massive data processing within low

end-to-end (E2E) delays (in the order of ms). Among several technology enablers at different layers (e.g., AI, network function virtualization, millimeter-wave communications), a prominent role will be played by Edge Computing, whose aim is to move cloud functionalities (e.g., computing and storage resources) at the edge of the network, to avoid the relatively long delays necessary to reach central clouds. Edge Computing is also the object of an ETSI Industry Specification Group, called Multi-Access Edge Computing (MEC) [5]. In 5G networks, MEC functionalities will be placed behind the User Plane Function (UPF), thus in the core network or virtualized locally at the Access Point (AP) [6]. MEC is foreseen to enable several novel applications and use cases [7], relying on the enhanced performance of new beyond 5G technologies, due to the massive volume of data to be transferred within low-latency and/or extremely high-reliability constraints [8]. Recent surveys on MEC are available in [9], [10].

In this paper, we focus on *computation offloading* services, in which the execution of applications is transferred from mobile devices (or sensors in IoT environments) to a nearby edge server (ES) [10]. Computation offloading helps reducing the User Equipment's (UE) energy consumption and/or the overall delay of the service. When an application is offloaded, the overall service time is composed of the uplink transmission time of input data, the processing time of this input at the ES, and the time needed to send the results back to the UE [11], [12]. In edge-computing-aided networks, a critical aspect for real-life implementations is the limited energy made available by the battery at the mobile device, the need for frequent battery recharge, and the high energy consumption of network elements, due to the dense deployment of APs and ESs necessary to enable the described ecosystem. In traditional mobile networks, a large portion of the power is consumed at the AP site [13], [14]. With the deployment of ESs, the power consumption will certainly increase, so that new methods are essential to reduce the impact of the ICT industry on the global carbon footprint [15]. In such a context, the main target of our paper is the energy efficiency of the overall network, comprising UE, AP, and ES.

## II. RELATED WORK AND CONTRIBUTION

In the context of mobile networks, several works focus on novel strategies to reduce system power consumption. In general, it is well-known that a large portion of the power is consumed by the AP only for being in active state (RF chains, power amplifiers, cooling, etc.) [14]. Thus, most of the works

M. Merluzzi and E. Calvanese Strinati are with Univ. Grenoble Alpes, CEA, Leti, F-38000 Grenoble, France, France.

Email: mattia.merluzzi@cea.fr, emilio.calvanese-strinati@cea.fr.

N. di Pietro is with Athonet, via Cà del Luogo 6/8, 36050, Bolzano Vicentino (VI), Italy. Email: nicola.dipietro@athonet.com.

P. Di Lorenzo and S. Barbarossa are with the Department of Information Engineering, Electronics, and Telecommunications of Sapienza University, via Eudossiana 18, 00184 Roma, Italy.

E-mail: paolo.dilorenzo@uniroma1.it, sergio.barbarossa@uniroma1.it.

This work was partly supported by the European Commission through the H2020 project Hexa-X (Grant Agreement no. 101015956), by H2020 EU/Taiwan Project 5G CONNI, Nr. 861459, by the CPS4EU project, which has received funding from the ECSEL Joint Undertaking (JU) under grant agreement Nr. 826276, and by MIUR under the PRIN Liquid\_Edge contract.

in the literature propose strategies based on possible ON/OFF behavior of the APs, known as Discontinuous Transmission (DTX) [14], [16]–[21], by which some components of the AP are put in low-power sleep states when possible, e.g., in case of low traffic. In the context of edge computing and computation offloading, there exists a wide literature [12], [22]–[31]. In particular, [25] proposes a dynamic computation offloading strategy, based on Lyapunov stochastic optimization, to reduce a weighted sum of UE and ES power consumption. [26] extends the strategy to a multi-server multi-cell scenario, introducing average delay and reliability constraints on the queue lengths. In [27], a joint dynamic computation offloading strategy was proposed with reliability guarantees, incorporating ultra-reliable low-latency communications and energy harvesting devices. All these works mainly focus on power consumption at the UE and ignore the network. The authors of [28] propose a dynamic strategy aimed at minimizing the average power consumption of mobile devices, under a latency constraint and a constraint on the ES average power consumption, without considering the AP. In [29], an auction-based incentive mechanism is proposed to maximize the revenue of a mobile network operator under delay constraints. In [30], the authors present a multi-objective approach to minimize the execution delay, energy consumption, and monetary cost of the smart devices with service rate constraints. The work in [31] proposes a scheduling and resource provisioning strategy to minimize edge nodes' power consumption under delay and resource constraints. Recent contributions consider the energy consumption of both radio access and MEC network [32]–[39]. In particular, in [32], a scheduling strategy is proposed to find a trade-off between task completion ratio and throughput, hinging on Lyapunov optimization, while [33] aims at minimizing the long-term average delay under a long-term average power consumption constraint. In [34], the long-term average energy consumption of a MEC network is minimized under a delay constraint, using a MEC sleep control. Also, in [35] the problem is formulated as the minimization of the energy consumption under a mean service delay constraint, optimizing the number of active base stations and the computation resource allocation at the ES, while considering a sleep mode for both APs and ESs. In [36], Lyapunov optimization is used to reduce the energy consumption of a fog network while guaranteeing an average response time. The authors of [37] minimize the offloading service delay with Lyapunov optimization under constraints on the user's and edge nodes' energy consumption. In [38], the authors exploit Lyapunov optimization, Lagrange multiplier, and sub-gradient techniques to optimize devices' and APs' energy consumption under delay constraints, exploiting AP sleep states. The authors of [39] propose a method to minimize a weighted sum of users and MEC energy consumption under delay constraints, considering a dedicated time for wireless charging.

Another class of recent works propose data-driven solutions as, e.g., Deep Reinforcement Learning (DRL) [40]–[43]. In [40], a decentralized approach based on DRL is proposed to minimize a weighted sum of user local powers, offloading powers, and buffering delays. In [41], the authors solve the problem of computation offloading with a deep

Q-network aimed at minimizing the energy consumption of MEC nodes and users under task delay constraints. DRL is also exploited for content caching in [42], where the authors aim at maximizing the content provider saving costs, with an incentive mechanism used to motivate end nodes to participate in the offloading process. In [43], DRL is used to minimize a weighted sum of energy consumption and delay in an IoT scenario. While [40]–[43] exploit pure data-driven solutions, other recent results show the possibility of merging model-based optimization with the power of data-driven optimization [44]–[47]. The common point of all these works is the lack of a *holistic* view of APs' sleep control, radio resource allocation, ESs' sleep and CPU scheduling, and UE's sleep control, under E2E delay constraints, involving average and out-of-service events, which is the goal of this paper. Our main challenge, not addressed in the available literature, is to design a strategy able to deal with complex time-varying scenarios with unknown statistics and several discrete optimization variables, involving heterogeneous entities (i.e. UE, APs and ESs), looking for low-complexity solutions able to run online.

#### A. Our Contribution

In this paper, we extend and improve our preliminary results of [48]. In contrast with the state of the art, we *simultaneously* optimize the modulation and coding scheme selection and the power of both the UE's uplink and the AP's downlink radio transmission, the CPU frequency allocation at the ES, and the duty cycles of all the network elements. We propose a dynamic computation offloading strategy based on Lyapunov stochastic optimization that minimizes the weighted sum of UE's, AP's, and ES's long-term average energy consumption, under an *average end-to-end delay constraint* and a *reliability constraint*. The latter is defined as the probability that the end-to-end delay exceeds a prescribed threshold. These constraints are handled through the definition of an uplink queue of data to be offloaded by each UE, a computation queue at the ES, and a downlink queue of results at the AP. These constraints translate into a constraint on the average length of the sum of the three queues and a probabilistic bound on the maximum total queue length, as in [48]. However, differently from [48], we introduce the sleep mode operation at the UE's side and an adaptive algorithm to translate the probabilistic constraint on the queue lengths into a reliability constraint on the *actual* end-to-end delay. Our proposed strategy does not require any *a priori* knowledge of the statistics of the radio channels or of the data arrivals. In particular, starting from a non-convex non-differentiable long-term average optimization problem with unknown statistics, we devise an algorithm that solves a deterministic problem on a per-slot basis, yielding an asymptotically optimal solution of the original problem (as a consequence of Proposition 1, as explained in Section IV). The proposed optimal solution of each deterministic problem has very low computational complexity and can be found via Algorithm 1, 2, and 3, presented in Section V. Several numerical results show the performance of our strategy, also compared with other methods, due to the fact that it takes into account the whole network energy consumption, reducing that

of all agents *simultaneously*, thus achieving a globally green solution.

### III. SYSTEM MODEL

To capture the dynamic aspects of the problem, we consider time as organized in slots  $t = 1, 2, 3 \dots$  of equal duration  $\tau_l$ . In the following, we present: the UE's, AP's, and ES's energy consumption model; the queueing model used to handle the delay constraints; the reliability performance over the radio interface in terms of Packet Error Rate (PER).

#### A. Energy consumption model

Computation offloading generally entails three phases: an uplink phase, where a UE sends data to the AP, a processing phase at the ES, and a downlink phase for the transmission of results to the UE [10], [12]. In our dynamic scenario, the overall slot duration  $\tau_l$  is divided into two portions: a period of  $\tau_s$  seconds dedicated to control signaling and transition among sleep/active states, and a period of  $\tau$  seconds for the actual three phases of computation offloading. Indeed, we assume that the AP, the ES, and the UE can enter low-power *sleep states* for energy saving purposes during the slot fraction reserved to offloading (not for the whole slot duration, due to the need for control signaling and state transitions at each time slot). When in sleep state, the AP and the UE cannot receive nor transmit and the ES cannot process data, thus consuming less power. Our goal is to optimize the long-term fraction of time that the entities spend in sleep state with the aim of minimizing the overall system energy consumption, but guaranteeing a targeted Quality of Service (QoS) measured by the overall delay of the computation offloading service. Since at the beginning of every slot each network element must be active, before the end of the slot all network elements wake up (if they were sleeping), to be active at the beginning of the next slot for control signaling. Thus,  $\tau_s$  comprises a portion at the beginning of the slot, devoted to control signaling and eventual transition from active to sleep, and a portion at the end of the slot, needed to wake up if in sleep state, to be active for control signaling at the beginning of the next slot, as depicted in the top right part of Fig. 1. Finally, the total duration of a time slot is  $\tau_l = \tau_s + \tau$ . It should be noted that, when dimensioning  $\tau_s$ , the transition time has to be taken into account. Thus, in this paper, we exploit sleep operation modes compatible with the slot duration from a transition time point of view, as it will be clarified in the following sections. The transition energy is neglected, as typically done in works related to DTX [19]–[21], [34], [35], [38], [49]. However, our model can be easily extended to take into account the transition energy consumption, being just an additional term of power consumed during a sleep phase.

1) *AP's Energy Consumption*: Nowadays, around 80% of the total power consumption of the wireless networks is consumed at the AP [49], which consumes a considerable fraction of its total power only for being in active state [13], [49]. Let us denote by  $p_a^{\text{on}}$  the overall power consumption of the AP for being in active state. This parameter generally includes the consumption of power amplifiers, power supply,

analog front-end, digital baseband, and digital control. In active state, the AP can transmit and/or receive. Thus, we denote by  $p^d(t)$  the overall downlink transmit power. Let us note that the AP can enter a low-power sleep mode to save energy whenever possible, without compromising the QoS. Obviously, the deeper the sleep mode, the higher the energy saving, but also the higher the time needed to wake-up, i.e., the minimum sleep period. In Section VI, we will present more specific considerations on the sleep modes, active and sleep power consumption of the AP, and transition times (e.g. from sleep to active). To control the active and sleep state of the AP, we introduce the binary variable  $I_a(t) \in \{0, 1\}$ , which equals 1 if and only if the AP is in active state at time slot  $t$ . In each slot, the AP is forced to be active for the first portion of  $\tau_s$  seconds to perform Channel State Information acquisition and control signaling. For simplicity, we neglect the transmit power necessary for this reduced exchange of information, thus taking into account only the active state power  $p_a^{\text{on}}$  during the signaling period. Then, the AP energy consumption at time slot  $t$  is

$$E_a(t) = \tau (I_a(t) (p_a^{\text{on}} + p^d(t)) + (1 - I_a(t))p_a^s) + \tau_s p_a^{\text{on}}, \quad (1)$$

where  $p_a^s$  represents the (low) power consumed by the AP in sleep mode. The power consumed in the receiver chain is neglected, as it is typically much smaller than the other contributions.

2) *UE's Energy Consumption*: Going beyond [48], we assume that all  $K$  UE can switch their radio equipment to a low-power sleep mode whenever possible. In particular, we assume that UE  $k$  (for  $k = 1, \dots, K$ ) consumes a generic power  $p_k^{\text{on}}$  only for being active. Also, we denote by  $p_k^u(t)$  the power necessary to transmit, assuming that it is a monotone increasing function of the transmit power  $p_k^{\text{tx}}(t)$ . In Section VI, we will be more specific with a model from the literature for the typical values of  $p_k^{\text{on}}$ , the function linking  $p_k^u(t)$  and  $p_k^{\text{tx}}(t)$ , and the transition times. Recalling that the UE is always active at the beginning and the end of the slot for control signaling, the total energy consumption of the UE is

$$E_u(t) = \sum_{k=1}^K \left[ \tau (I_k(t) (p_k^{\text{on}} + p_k^u(t)) + (1 - I_k(t))p_k^s) + \tau_s p_k^{\text{on}} \right], \quad (2)$$

where  $I_k(t)$  equals one if UE  $k$  is active in time slot  $t$ , and 0 otherwise.

3) *ES's Energy Consumption*: As pointed out in [50], the power management of a CPU is all about efficiently (and dynamically) controlling both current and voltage in order to minimize power while providing a desired performance. Power-saving techniques can be divided into two main categories: *turn it off* and *turn it down*. The first one consists in switching off some components of the CPU, which are then put into low-power sleep states. In modern processors, there exist several possible idle states, called C-states [50], which allow the processor to enter more or less deep sleep modes. Obviously, a deeper sleep mode provides higher energy sav-

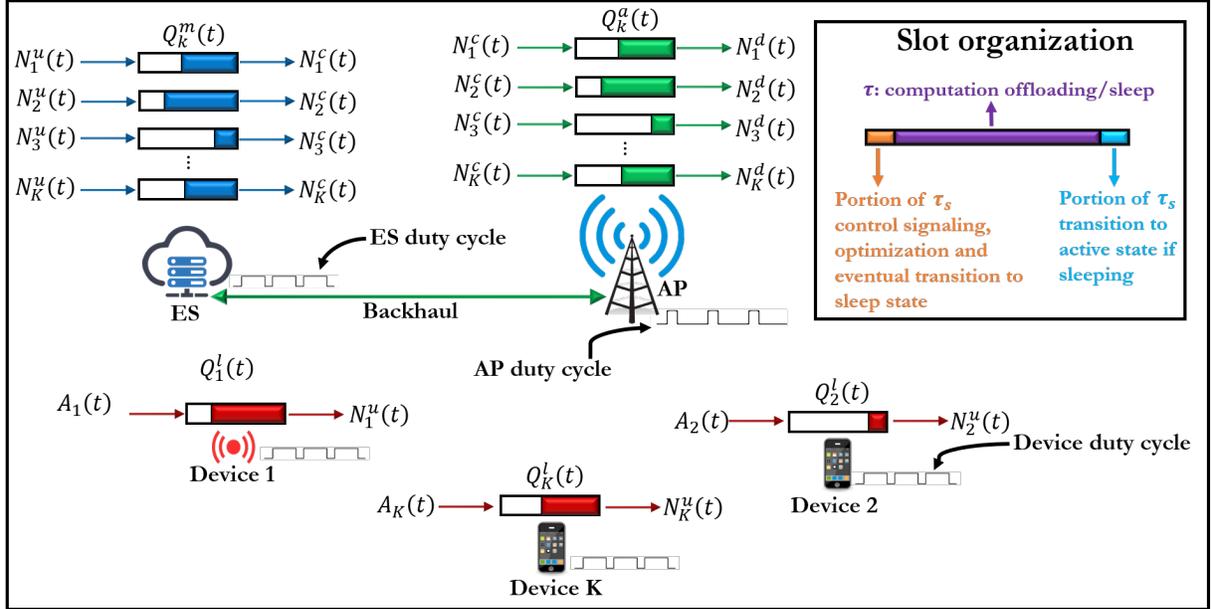


Fig. 1: Network model

ings, but requires more time to wake up. This defines a trade-off between energy consumption and latency. Furthermore, C-states can operate on each core separately or on the entire CPU package [51].

In this paper, we adopt C-states operating on a specific core, dedicated to treat the offloaded tasks of all the UE of our system. In particular, we consider two states: the C0-state, in which the CPU core is active and executing some thread, and the C $x$ -state ( $x = 1, 2, \dots$ ), in which the CPU clock frequency is driven to zero. The transition time from C $x$  to C0 depends on the specific choice of the C $x$  state. For instance, for C1-state, it is in the order of  $\mu\text{s}$  [51]. In Section VI, we will present more specific considerations on the choice of the C $x$  state, based on the duration of the slot. In our model, the CPU core consumes a power  $p_m^{\text{on}}$  just for being in active state (C0-state) and a power  $p_m^{\text{s}}$  in sleep state (C $x$ -state). Moreover, when the ES is active, the dynamic power spent for computation is  $p_m^c(t) = \kappa f_c^3(t)$ , where  $f_c(t)$  is the CPU cycle frequency at time slot  $t$  and  $\kappa$  is the effective switched capacitance of the processor [52]. We suppose that it is possible to use dynamic voltage frequency scaling to scale down the frequency [53], thus reducing the dynamic power consumption. In particular, we assume that  $f_c$  can be selected from a finite set  $\mathcal{F} = \{0, \dots, f_{\text{max}}\}$  and we introduce the binary variable  $I_m(t) \in \{0, 1\}$ , which equals 1 if and only if the ES is in active state. Then, recalling that  $\tau_l = \tau_s + \tau$ , the energy consumption in each time slot is given by

$$E_m(t) = \tau (I_m(t)p_m^{\text{on}} + (1 - I_m(t))p_m^{\text{s}} + p_m^c(t)) + \tau_s p_m^{\text{on}}, \quad (3)$$

where  $I_m(t) = \mathbf{1}\{f_c(t)\}$ , with  $\mathbf{1}\{\cdot\}$  the indicator function; note that  $p_m^c(t) = 0$  whenever  $f_c(t) = 0$ , because  $p_m^c(t) = \kappa f_c^3(t)$ . Then, from (1), (2), (3), the total energy consumption in slot  $t$  is:

$$E_{\text{tot}}(t) = E_u(t) + E_m(t) + E_a(t). \quad (4)$$

### B. Delay and queuing model

Computation offloading involves three main steps: an uplink transmission phase of input data from the UE; a computation phase at the ES; a downlink transmission phase of results back to the UE. We consider a dynamic scenario, in which new input data units are continuously generated from an application at the UE's side and have to be offloaded and processed at the ES. To model the system dynamics, we use a simple queuing model, taking into account the three phases of computation offloading. This model allows us to characterize the total delay experienced by a data unit from its generation at the mobile side until the reception of its corresponding result, sent by the AP to the UE. In particular, the considered queuing model is depicted in Fig. 1. Specifically, in Fig. 1, we can notice three different queues: *i*) A local communication queue at each device (red) of data buffered before uplink transmission; *ii*) A remote computation queue at the ES (blue) of data buffered before being processed; *iii*) A downlink communication queue (green) of results buffered before being sent back to the devices. Accordingly, each data unit experiences three different delays: a communication delay, including buffering at the UE; a computation delay, including buffering at the ES; a communication delay, including buffering at the AP. As we will show later, we take into account these three sources of delay jointly, as in [27]. For the multiple access over the radio channel, we consider a simple Frequency Division Multiple Access, both for the uplink and the downlink.

1) *Uplink communication queue:* In uplink, allocating bandwidth  $B_k^u$  to UE  $k$ , the symbol duration is  $T_k^{s,u} = \frac{1}{B_k^u}$ . Since the time for data transmission is  $\tau = \tau_l - \tau_s$ , under the assumption of a perfect pulse shaping, UE  $k$  can transmit  $N_k^{s,u}(t) = \left\lfloor \frac{\tau}{T_k^{s,u}} \right\rfloor = \lfloor \tau B_k^u(t) \rfloor$  symbols at time  $t$ . Assuming that bits are encoded against radio channel noise into packets of fixed length  $N_b$  bits, employing an  $M$ -QAM modulation,

the number of packets transmittable at time  $t$  is given by:

$$N_k^{p,u}(t) = \left\lfloor \frac{N_k^{s,u}(t) \log_2(M_k^u(t)) R_k^{c,u}(t)}{N_b} \right\rfloor \quad (5)$$

where  $M_k^u(t)$  is the modulation order and  $R_k^{c,u}(t)$  is the channel coding rate. In particular, we assume that the uplink Modulation and Coding Scheme (MCS) pair  $m_k^u = (M_k^u(t), R_k^{c,u}(t))$  is chosen from a discrete finite set  $\mathcal{M}_k^u$ . Also, we assume that a data unit has to be transferred in one time slot, i.e., it cannot be split and partially transmitted over different time slots. Thus, the number of data units that UE  $k$  can send at time slot  $t$  over the radio interface is

$$N_k^u(t) = \left\lfloor \frac{N_k^{p,u}(t) N_b}{S_k^i} \right\rfloor, \quad (6)$$

where  $S_k^i$  is the size in bits of an input data unit. Then, the local queue of input data units to be offloaded evolves as

$$Q_k^l(t+1) = \max(0, Q_k^l(t) - N_k^u(t)) + A_k(t), \quad (7)$$

where  $A_k(t)$  is the number of newly arrived data units generated by the application running at UE  $k$ ;  $A_k(t)$  is modeled as a random process whose statistics are not known *a priori*.

2) *Remote computation queue*: We assume that the number of input data units processed by the ES to serve UE  $k$  is proportional to the number of CPU cycles allocated for this task. Given the computation rate  $f_k(t)$  assigned to user  $k$ , measured in CPU cycles per second, and defining the coefficient  $J_k$  as the ratio between the number of processed data and the number of CPU cycles, the number of data units processed by the ES for UE  $k$  at time slot  $t$  is

$$N_k^c(t) = \lceil \tau f_k(t) J_k \rceil. \quad (8)$$

Hence, the queue of data waiting for being processed by the ES for UE  $k$  evolves as

$$Q_k^m(t+1) = \max(0, Q_k^m(t) - N_k^c(t)) + \min(Q_k^l(t), N_k^u(t)). \quad (9)$$

3) *Downlink communication queue*: Finally, we define  $K$  queues at the AP, containing the computation results to be sent back to the UE. We assume that every processed input data unit produces one output data unit, with size  $S_k^o$  possibly different from  $S_k^i$ . The queue evolves as:

$$Q_k^a(t+1) = \max(0, Q_k^a(t) - N_k^d(t)) + \min(Q_k^m(t), N_k^c(t)), \quad (10)$$

where  $N_k^d(t)$  is the number of output data units sent back to user  $k$  in downlink, which is computed as

$$N_k^d(t) = \left\lfloor \frac{N_k^{p,d}(t) N_b}{S_k^o} \right\rfloor, \quad (11)$$

where  $N_k^{p,d}(t)$  is the number of packets sent in downlink, given similarly as in (5), using  $B_k^d(t)$  for the bandwidth assigned to UE  $k$  for downlink communication at time  $t$ .  $M_k^d(t)$  is the downlink  $M$ -QAM modulation order, and  $R_k^{c,d}(t)$  is the channel coding rate for downlink. As for the uplink, the pair  $m_k^d = (M_k^d(t), R_k^{c,d}(t))$  belongs to a discrete set  $\mathcal{M}_k^d$ .

4) *End-to-end delay constraints*: As already mentioned, the overall delay experienced by a data unit is the time elapsed from its generation at the mobile side, to the moment the user receives back the result associated with it. By Little's law [54], the average overall service delay is proportional to the average queue length. Then, the overall delay is directly related to the sum of the uplink and downlink communication queues and the computation queue  $Q_k^{\text{tot}}(t) = Q_k^l(t) + Q_k^m(t) + Q_k^a(t)$ . In particular, given a data unit arrival rate  $A_k^{\text{avg}} = \mathbb{E}\{A_k(t)/\tau\}$ , the long-term average end-to-end delay experienced by a data unit generated by UE  $k$  is  $\bar{D}_k^\infty = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E}\{Q_k^{\text{tot}}(t)/A_k^{\text{avg}}\}$ , where the expectation is taken with respect to the random radio channel and data arrival realizations. Our first aim is to guarantee a constraint on the long-term average delay  $D_k^{\text{avg}}$ , written as:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E}\{Q_k^{\text{tot}}(t)\} \leq Q_k^{\text{avg}} = D_k^{\text{avg}} A_k^{\text{avg}}, \quad \forall k. \quad (12)$$

As a second objective, we want to ensure a long-term probabilistic constraint on the E2E delay experienced by data units:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \Pr\{D_k(t) > D_k^{\text{max}}\} \leq \epsilon_k, \quad \forall k, \quad (13)$$

where  $D_k^{\text{max}}$  is a predefined threshold,  $0 < \epsilon_k < 1$ , and  $D_k(t)$  represents the overall delay experienced by a generic data unit whose result is received back by UE  $k$  at time  $t$ . The aim of this constraint is to reduce the variability of the delay. As mentioned before, there is a direct dependence between the overall delay and the overall queue length, therefore we can translate (13) into the following probabilistic constraint on the sum of the queues:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \Pr\{Q_k^{\text{tot}}(t) > \delta_k Q_k^{\text{avg}}\} \leq \epsilon_k, \quad \forall k, \quad (14)$$

with  $\delta_k > 1$  conveniently chosen to convert the delay threshold into a queue-length threshold. In principle, there is no direct analytical relation between  $\delta_k Q_k^{\text{avg}}$  and  $D_k^{\text{max}}$ , but we will propose in Section V-C an online method to appropriately select and adapt  $\delta_k$ . Finally, note that (14) can be equivalently recast as the expectation of a Bernoulli random variable as

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E}\{u(Q_k^{\text{tot}}(t) - \delta_k Q_k^{\text{avg}})\} \leq \epsilon_k,$$

where  $u(\cdot)$  is the unitary step function. In the sequel, the event  $\{Q_k^{\text{tot}}(t) > \delta_k Q_k^{\text{avg}}\}$  will be termed as “*out-of-service*”, and  $\epsilon_k$  will be the required *out-of-service probability*.

### C. Packet error rate performance

To satisfy a target performance in terms of packet loss, we want to guarantee that the uplink and downlink PER, denoted respectively  $\text{PER}_k^u$  and  $\text{PER}_k^d$  for UE  $k$ , do not exceed some targeted thresholds  $\theta_k^u$  and  $\theta_k^d$ . In this sense, given the radio channel state at time  $t$ , recalling that the transmit power  $p_k^{\text{tx}}(t)$  used by UE  $k$  is a function of the chosen MCS  $m_k^u \in \mathcal{M}_k^u$ , we define  $p_k^{\text{tx}, \min}(m_k^u, t) = \min\{p_k^{\text{tx}}(t) : \text{PER}_k^u \leq \theta_k^u\}$ . A

minimum target PER translates into a minimum target SNR  $\bar{\gamma}_k$ . Thus, the minimum transmit power is  $p_k^{\text{tx},\min} = \frac{\bar{\gamma}_k N_0 B_k^u}{h_k^u}$ , where  $N_0$  is the noise power spectral density at the receiver, and  $h_k^u$  is the time-varying uplink channel power gain. The same discussion is valid for the downlink transmission.

#### IV. PROBLEM FORMULATION

In this section, we formulate our optimization problem, aimed at minimizing the long-term average weighted sum of the UE's, AP's and ES's energy consumption, as defined in (1), (2), (3), whose value at time slot  $t$  is given by the convex combination:

$$E_{\text{tot}}^w(t) = \alpha_1 E_u(t) + \alpha_2 E_a(t) + \alpha_3 E_m(t), \quad (15)$$

where  $\alpha_i \geq 0$ ,  $\forall i$ , and  $\sum_{i=1}^3 \alpha_i = 1$ , with the coefficients  $\alpha_i$  chosen in order to explore alternative priority mechanisms assigned to different energy consumption sources, as clarified later on. The long-term optimization problem is then:

$$\min_{\Psi(t)} \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E} \{ E_{\text{tot}}^w(t) \} \quad (16)$$

subject to

- (a) Eqn. (12);
- (b) Eqn. (13);
- (c)  $m_k^u(t) \in \mathcal{M}_k^u$ ,  $\forall k, t$ ;
- (d)  $m_k^d(t) \in \mathcal{M}_k^d$ ,  $\forall k, t$ ;
- (e)  $p_k^{\text{tx},\min}(m_k^u, t) I_k(t) \leq p_k^{\text{tx}}(t) \leq p_k^{\text{tx},\max} I_k(t)$ ,  $\forall k, t$ ;
- (f)  $I_k(t) \in \{0, 1\}$ ,  $\forall k, t$ ;
- (g)  $p_k^{\text{d},\min}(m_k^d, t) I_a(t) \leq p_k^{\text{d}}(t) \leq p_k^{\text{d},\max} I_a(t)/K$ ,  $\forall k, t$ ;
- (h)  $I_a(t) \in \{0, 1\}$ ,  $\forall t$ ;
- (i)  $f_c(t) \in \mathcal{F}$ ,  $\forall t$ ;
- (j)  $f_k(t) \geq 0$ ,  $\forall k, t$ ;
- (k)  $\sum_{k=1}^K f_k(t) \leq f_c(t)$ ,  $\forall t$ ;

where  $\Psi(t) = [\{\Phi_k(t)\}_{k=1}^K, f_c(t), I_a(t)]$ , with  $\Phi_k(t) = [m_k^u(t), m_k^d(t), p_k^{\text{tx}}(t), p_k^{\text{d}}(t), f_k(t), I_k(t)]$ . The constraints in (16) have the following meaning: (a) the average end-to-end delay of each user does not exceed  $D_k^{\text{avg}} = Q_k^{\text{avg}}/A_k^{\text{avg}}$ ; (b) the out-of-service probability is lower than a threshold  $\epsilon_k$ ; (c)-(d) the uplink and downlink MCS belong, respectively, to  $\mathcal{M}_k^u$  and  $\mathcal{M}_k^d$ ; (e) the uplink transmit power of each UE guarantee the PER constraints and is lower than some fixed budget  $p_k^{\text{tx},\max}$ ; (f) the indicator variable of each UE's sleep state is binary; (g) the downlink transmit power of each UE guarantee the PER constraints and is lower than some fixed budget  $p_k^{\text{d},\max}/K$ ; (h) the indicator of the AP's sleep state is binary; (i) the computation frequency of the ES is selected from a discrete set  $\mathcal{F}$ ; (j) the CPU cycle frequency assigned to UE  $k$  is non-negative; (k) the sum of all CPU cycle frequencies assigned to all UE does not exceed the ES's total computation frequency  $f_c$ .

Clearly, the problem formulation in (16) raises many issues in terms of high complexity and hard tractability. First of all, in (16), the objective function and the long-term constraints (cf.

(12), (13)) cannot be computed *a priori*, since the statistics of radio channels and data arrivals are not supposed to be known. Furthermore, even by assuming perfect knowledge of the statistics, several discrete variables over a long-term time horizon are involved, thus making the problem to exhibit exponential computational complexity, in principle. Nevertheless, hinging on Lyapunov stochastic optimization [55], we are able to transform (16) into a *pure stability problem*, which is solved in a per-slot fashion that requires only the observation of instantaneous realizations. Building on stochastic optimization theory, we prove the convergence and the asymptotic optimality of the proposed strategy. Furthermore, we show that the per-slot problem enables a low-complexity solution, even in the presence of the discrete variables, thanks to the decoupling across different slots. Optimality is asymptotically achieved thanks to the introduction of virtual queues that allow the algorithm to keep track online of how well the method is behaving in the real case. In general, different approaches can be followed when the system model (or part of it) is not known or to handle complexity efficiently. For instance, in [41], the authors approximate the original long-term problem, which is a mixed integer linear program that exhibits exponential complexity, using a Markov decision process that is then solved via DRL. The main difference of our work with respect to the data-driven solution of [41] is that, by exploiting the mathematical models presented in Section III and keeping track of the instantaneous (real and virtual) queues' state, it is possible to split the original problem into a series of consecutive simpler problems that do not need a reinforcement method to be solved, but rather enjoy closed form expressions and fast iterative solutions, with asymptotic theoretical guarantees.

#### A. Lyapunov stochastic optimization

We present now a way to guarantee the long-term constraints, based on Lyapunov stochastic optimization. The solution depends on the definition of two *virtual queues* for each UE. For each UE  $k = 1, \dots, K$ , the first virtual queue  $Z_k(t)$ , used to impose constraint (a) in (16), evolves as

$$Z_k(t+1) = \max(0, Z_k(t) + Q_k^{\text{tot}}(t+1) - Q_k^{\text{avg}}). \quad (17)$$

Similarly, for constraint (b), we define a virtual queue  $Y_k(t)$  that evolves as

$$Y_k(t+1) = \max(0, Y_k(t) + \mu_k (u \{Q_k^{\text{tot}}(t+1) - \delta_k Q_k^{\text{avg}}\} - \epsilon_k)), \quad (18)$$

where  $\mu_k > 0$  is a scalar stepsize. The *mean rate stability* of the queues is defined as [55, p. 17]:

$$\lim_{T \rightarrow \infty} \frac{\mathbb{E}\{Z_k(T)\}}{T} = 0, \quad \forall k, \quad \lim_{T \rightarrow \infty} \frac{\mathbb{E}\{Y_k(T)\}}{T} = 0, \quad \forall k. \quad (19)$$

In particular, the mean-rate stability of  $Z_k(t)$  and  $Y_k(t)$  is sufficient to ensure constraints (a) and (b) in (16) [55]. We now introduce the *Lyapunov function* [55, p. 32]:

$$L(\Theta(t)) = \frac{1}{2} \sum_{k=1}^K [Z_k^2(t) + Y_k^2(t)],$$

where  $\Theta(t) = [\{Z_k(t)\}_k, \{Y_k(t)\}_k]$ . From  $L(\Theta(t))$ , we can define the *conditional Lyapunov drift* [55, p. 33], which is the conditional expected variation of  $L(\Theta(t))$  over one slot

$$\Delta(\Theta(t)) = \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t)) | \Theta(t)\}. \quad (20)$$

Minimizing (20) is enough to achieve (19), but may yield the drawback of an unnecessary energy consumption. For this reason, we need to integrate the objective function of (16) in the drift, obtaining the *drift-plus-penalty function* [55, p. 39]:

$$\Delta_p(\Theta(t)) = \Delta(\Theta(t)) + V \cdot \mathbb{E}\{E_{\text{tot}}^w(t) | \Theta(t)\}, \quad (21)$$

where  $V$  is a trade-off parameter used to tune the relative importance given to the objective function with respect to the average virtual queue backlog.  $\Delta_p(\Theta(t))$  is just a ‘‘penalized’’ version of (20). Then, the parameter  $V$  is used to trade off the average weighted energy consumption in (15) and the average E2E delay, as it will be also clarified in Section VI with the numerical results. Now, proceeding as in [55], we minimize an upper bound of (21) in each time slot, whose derivation is described in the appendix (cf. (40)). In particular, our method requires the solution of the following optimization problem in each time slot:

$$\begin{aligned} \min_{\Psi(t)} \sum_{k=1}^K & \left[ -2Q_k^l(t)N_k^u(t) + 4Q_k^m(t)(N_k^u(t) - N_k^c(t)) \right. \\ & + 4Q_k^a(t)(N_k^c(t) - N_k^d(t)) + Z_k(t) [\max(0, Q_k^l(t) - N_k^u(t)) \\ & + \max(0, Q_k^m(t) - N_k^c(t)) + \max(0, Q_k^a(t) - N_k^d(t))] \\ & + \mu_k Y_k(t) u \{ \max(0, Q_k^l(t) - N_k^u(t)) + A_k(t) \\ & + \max(0, Q_k^m(t) - N_k^c(t)) + \min(Q_k^l(t), N_{k,\max}^u) \\ & + \max(0, Q_k^a(t) - N_k^d(t)) + \min(Q_k^m, N_{k,\max}^c) \\ & \left. - \delta_k Q_k^{\text{avg}} \right] + V E_{\text{tot}}^w(t) \end{aligned} \quad (22)$$

subject to  $\Psi(t) \in \mathcal{Z}(t)$ ,

where  $\mathcal{Z}(t)$  is the feasible set defined by (c)-(k) of (16). Now, at every  $t$ , the *Min Drift-Plus-Penalty Algorithm* observes the queue states  $Q_k^l(t)$ ,  $Q_k^m(t)$ ,  $Q_k^a(t)$ ,  $\Theta(t)$  and the random events  $h_k(t)$ ,  $A_k(t)$  and produces a control decision  $\Psi(t) \in \mathcal{Z}(t)$  based on the solution of (22). The non-convex non-differentiable objective function in (22) is difficult to optimize. Thus, we proceed by finding a suitable approximation of (22) that simplifies the solution but still provides optimality guarantees. In particular, we hinge on the concept of  $\Gamma$ -additive approximation [55, p. 59]:

*Definition 1:* For a given constant  $\Gamma$ , a  $\Gamma$ -additive approximation of the drift-plus-penalty algorithm is one that, for a given state  $\Theta(t)$  at slot  $t$ , chooses a (possibly randomized) action  $\Psi(t) \in \mathcal{Z}(t)$  that yields a conditional expected value of the objective function in (22) that is within a constant  $\Gamma$  from the infimum over all possible control actions.

To find a suitable  $\Gamma$ -approximation, we first introduce the following upper bound, used to get rid of the non-linearity introduced by the  $\lfloor \cdot \rfloor$  operator in (22). In particular, since we can write  $x - 1 \leq \lfloor x \rfloor \leq x$ , we have  $\max(0, Q_k^m(t) -$

$\lfloor \tau f_k(t) J_k \rfloor) \leq \max(0, Q_k^m(t) - \tau f_k(t) J_k + 1)$ . Then, adding without loss of generality the following additional constraint:

$$f_k(t) \leq \min \left( \frac{Q_k^m(t) + 1}{\tau J_k}, f_c \right), \quad \forall k, t,$$

we have  $\max(0, Q_k^m(t) - \tau f_k(t) J_k + 1) = Q_k^m(t) - \tau f_k(t) J_k + 1$ . Finally, to deal with the non-linearity introduced by the step function  $u\{\cdot\}$ , we note that  $u\{x - A\} \leq u\{x\} \leq x + 1$ ,  $\forall x, A \geq 0$ . Applying these bounds to the objective function of (22) and removing the constant terms, the problem can be re-formulated as follows (we omit the temporal index  $t$  for ease of notation):

$$\begin{aligned} \min_{\Psi} \sum_{k=1}^K & \left[ (4Q_k^m - 2Q_k^l) N_k^u - \tilde{Q}_k \tau f_k J_k - 4Q_k^a N_k^d \right. \\ & \left. + (Z_k + \mu_k Y_k) (\max(0, Q_k^l - N_k^u) \right. \\ & \left. + \max(0, Q_k^a - N_k^d)) \right] + V E_{\text{tot}}^w \end{aligned}$$

subject to

$$(a) \quad \Psi \in \mathcal{Z}; \quad (b) \quad f_k \leq \min \left( \frac{Q_k^m + 1}{\tau J_k}, f_c \right), \quad \forall k; \quad (23)$$

where  $\tilde{Q}_k = 4(Q_k^m - Q_k^a) + Z_k + \mu_k Y_k$  and  $\Psi$  and  $\mathcal{Z}$  are defined as for (22). The following theoretical result applies.

*Proposition 1:* Suppose that the channel gains  $\{h_k(t)\}_k$  and the data arrivals  $\{A_k(t)\}_k$  are i.i.d over time, that (16) is feasible, and that  $\mathbb{E}\{L(\Theta(0))\} < \infty$ ; then, solving (23) in each time slot guarantees that all virtual queues are mean-rate stable (i.e., (19) holds) and  $E_{\text{tot}}^w$  is such that:

$$\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E}\{E_{\text{tot}}^w(t)\} \leq E_{\text{tot}}^{w,\text{opt}} + \frac{\zeta + \Gamma}{V}, \quad (24)$$

where  $E_{\text{tot}}^{w,\text{opt}}$  is the infimum time average energy achievable by any policy that meets the required constraints, and  $\zeta$  is a positive constant defined in the appendix (cf. (41)).

*Proof:* The proof follows from the fact that the control policy deriving from the solution of (23) is a  $\Gamma$ -additive approximation of the drift-plus-penalty algorithm in (22). This holds true because, for any given state  $\Theta(t)$  of the physical and virtual queues at slot  $t$ , function (22) is bounded from above due to the finite size of the feasible set  $\mathcal{Z}(t)$ , for all  $t$ . Thus, the conditional expected value of the objective function in (22), evaluated in the solution of (23), is within a constant  $\Gamma$  from the global optimum of problem (22). The derivations leading to (22) and (41) are given in the Appendix. The main claim comes as a direct consequence of [55, Th. 4.8]. ■

*Remark 1:* The main consequence of Proposition 1 is that the mathematically optimal long-term solution of (16) is achieved by optimally solving (23), as  $V$  increases. This will be also clearly visible in the numerical results of Section VI.

*Remark 2:* The algebraic manipulations that led to (23), decouple the radio and computation optimization variables and allow us to optimally split the main problem into two different sub-problems: i) radio resource allocation problem, both in uplink and downlink; ii) ES CPU scheduling problem. We now present a low-complexity algorithm that achieves the globally optimal solution of (23).

## V. SOLUTION OF THE PER-SLOT OPTIMIZATION PROBLEM

### A. Radio Resource Allocation

The problem for radio resource allocation involves: (i) the decision on the UE and AP sleep state, (ii) the uplink transmit power and MCS selection, and (iii) the downlink transmit power and MCS selection. Then, omitting the temporal index  $t$ , defining  $\Gamma = [\{m_k^u\}_k, \{m_k^d\}_k, \{p_k^{\text{tx}}\}_k, \{p_k^d\}_k, \{I_k\}_k, I_a]$ , and recalling (2), (4), and (15), the first sub-problem reads as:

$$\begin{aligned} \min_{\Gamma} \sum_{k=1}^K & \left[ (4Q_k^m - 2Q_k^l)N_k^u - 4Q_k^a N_k^d \right. \\ & + (Z_k + \mu_k Y_k) (\max(0, Q_k^l - N_k^u) + \max(0, Q_k^a - N_k^d)) \\ & \left. + V\alpha_1 E_k \right] + V\alpha_2 E_a \\ \text{subject to } & \Gamma \in \mathcal{Z}', \end{aligned} \quad (25)$$

where  $\mathcal{Z}'$  is the feasible set for the radio resources according to (c)-(h) of (16). Now, to solve (25), we can distinguish between two different cases:

**Case 1:**  $I_a = 0$ . In this case, since the AP is in sleep mode and cannot receive nor transmit, no user can transmit or receive and  $I_k = 0$ , for all  $k$ . Thus, the trivial solution is  $N_k^u = N_k^d = p_k^{\text{tx}} = p_k^d = 0$ . Moreover, recalling that each UE and the AP are forced to be in active state for a period  $\tau_s$  necessary for control signaling, the energy consumption of each user is simply given by  $E_k = \tau p_k^s + \tau_s p_k^{\text{on}}$ , while the energy consumption of the AP is  $E_a = \tau p_a^s + \tau_s p_a^{\text{on}}$ . Then, the minimum value of the objective function of (25) in this case, is given by:

$$\begin{aligned} L^s = \sum_{k=1}^K & (Z_k + \mu_k Y_k) (Q_k^l + Q_k^a) \\ & + V \left[ \alpha_1 \sum_{k=1}^K (\tau p_k^s + \tau_s p_k^{\text{on}}) + \alpha_2 (\tau p_a^s + \tau_s p_a^{\text{on}}) \right]. \end{aligned} \quad (26)$$

This value will be compared with the solution obtained in the following second case.

**Case 2:**  $I_a = 1$ . In this case, the AP is available for transmission and/or reception, and the radio resources in uplink and downlink can take different values. In particular each UE can optimize its  $I_k$ . Thus, we can distinguish between the case  $I_k = 0$ , in which no transmission or reception occurs for UE  $k$ , and the case  $I_k = 1$ , in which the uplink and the downlink resources can take any value of the feasible set. When  $I_k = 0$ , we have  $N_k^u = N_k^d = p_k^{\text{tx}} = p_k^d = 0$  and the part of the objective function associated with each UE is

$$L_k^s = (Z_k + \mu_k Y_k) (Q_k^l + Q_k^a) + V\alpha_1 (\tau p_k^s + \tau_s p_k^{\text{on}}). \quad (27)$$

On the other hand, in the case  $I_k = 1$ , the optimization of each uplink and downlink variable is independent from the others. We now show the solutions for each user in the case  $I_k = 1$ .

1) *Optimal Uplink Radio Resource Allocation:* As already mentioned, in this work we assume that the spectral resources (i.e., the bandwidth) are assigned *a priori*. This makes the

problem separable among different UE and can be formulated, for all  $k$  with  $I_k = 1$ , as follows:

$$\begin{aligned} \min_{\{m_k^u, p_k^{\text{tx}}\}} & (4Q_k^m - 2Q_k^l)N_k^u + (Z_k + \mu_k Y_k) \max(0, Q_k^l - N_k^u) \\ & + V\alpha_1 \tau p_k^u + V\alpha_1 (\tau + \tau_s) p_k^{\text{on}} \\ \text{subject to} & \\ (a) \quad & m_k^u \in \mathcal{M}_k^u; \quad (b) \quad p_k^{\text{tx}, \min}(m_k^u) \leq p_k^{\text{tx}} \leq p_k^{\text{tx}, \max}, \end{aligned} \quad (28)$$

where we recall that  $p_k^u$  is a (given) monotone increasing function of  $p_k^{\text{tx}}$  (cf. Section III-A2, and  $N_k^u$  is a function of  $m_k^u$  (cf. (6)). Since  $\mathcal{M}_k^u$  is discrete and finite, (28) can be easily solved via an exhaustive search over all possible schemes in  $\mathcal{M}_k^u$  (with linear complexity in the cardinality of  $\mathcal{M}_k^u$ ), where the optimal choice for the transmit power for each  $k$  is  $p_k^{\text{tx}} = p_k^{\text{tx}, \min}(m_k^u)$ . Note that it might happen that  $p_k^{\text{tx}, \min}(m_k^u) \geq p_k^{\text{tx}, \max}$ . In this case, the selected MCS cannot be used to guarantee the required PER; thus, the solution of (28) has to be searched in the subset of  $\mathcal{M}_k^u$  that satisfies the constraint on the PER. We denote by  $m_k^{u, \text{opt}}$  and  $p_k^{\text{tx}, \text{opt}}$  the optimal values of the MCS and the uplink transmit power, respectively. Then, the optimal value of  $N_k^u$  is obtained by plugging  $m_k^{u, \text{opt}}$  in (6). Finally, If no MCS can be used to guarantee the required PER, the user  $k$  does not transmit, i.e.,  $N_k^{u, \text{opt}} = p_k^{\text{tx}, \text{opt}} = 0$ .

2) *Optimal Downlink Radio Resource Allocation:* The downlink resource allocation is similar to the uplink case, so that the following subproblem of (25) is solved for each user in each slot:

$$\begin{aligned} \min_{\{m_k^d, p_k^d\}} & -4Q_k^a N_k^d + (Z_k + \mu_k Y_k) \max(0, Q_k^a - N_k^d) + V\alpha_2 p_k^d \\ \text{subject to} & \\ (a) \quad & m_k^d \in \mathcal{M}_k^d; \quad (b) \quad p_k^{d, \min}(m_k^d) \leq p_k^d \leq p_k^{d, \max}/K, \end{aligned} \quad (29)$$

where  $N_k^d$  is a function of  $m_k^d$  (cf. (11)). The solution of this problem is obtained, as for the uplink case, via an exhaustive search over the feasible values of  $m_k^d$ . We denote by  $m_k^{d, \text{opt}}$  and  $p_k^{d, \text{opt}}$  the optimal solutions of (29). The optimal value of  $N_k^d$ , denoted by  $N_k^{d, \text{opt}}$  is obtained by plugging  $m_k^{d, \text{opt}}$  into (11). Then, let  $L_k^a$  be the following quantity, resulting from the UE's active state:

$$\begin{aligned} L_k^a = & (4Q_k^m - 2Q_k^l)N_k^{u, \text{opt}} - 4Q_k^a N_k^{d, \text{opt}} \\ & + (Z_k + \mu_k Y_k) (\max(0, Q_k^l - N_k^{u, \text{opt}}) \\ & + \max(0, Q_k^a - N_k^{d, \text{opt}})) \\ & + V\alpha_1 (\tau p_k^{u, \text{opt}} + (\tau + \tau_s) p_k^{\text{on}}) + V\alpha_2 \tau p_k^{d, \text{opt}}. \end{aligned} \quad (30)$$

The optimal value of  $I_k$ , denoted by  $I_k^{\text{opt}}$ , is then chosen based on the comparison between (27) and (30). In particular,  $I_k^{\text{opt}} = 1$  if  $L_k^a < L_k^s$ , and  $I_k^{\text{opt}} = 0$ , otherwise. Finally, letting

$$L^a = \sum_{k=1}^K (I_k^{\text{opt}} L_k^a + (1 - I_k^{\text{opt}}) L_k^s) + V\alpha_2 (\tau + \tau_s) p_a^{\text{on}}, \quad (31)$$

the optimal value of  $I_a$ , denoted by  $I_a^{\text{opt}}$ , is chosen based on the comparison between (26) and (31). In particular,  $I_a^{\text{opt}} = 1$

---

**Algorithm 1** Radio Resource Allocation
 

---

In each time slot  $t$ , observe  $Q_k^l, Q_k^m, Q_k^a, Z_k, Y_k, h_k^u, h_k^d, \forall k$ .

**S1.** Solve (28) and (29) to find for each UE  $k$  the values  $m_k^{u,\text{opt}}, p_k^{\text{tx,opt}}, m_k^{d,\text{opt}}, p_k^{d,\text{opt}}$ . Plug  $m_k^{u,\text{opt}}$  and  $m_k^{d,\text{opt}}$  into (6) and (11) to find  $N_k^{u,\text{opt}}$  and  $N_k^{d,\text{opt}}$ , respectively.

**S2.** Compute  $L_k^s$  and  $L_k^a$  from (27) and (30), respectively,  $\forall k$ .

**S3. for**  $k = 1, \dots, K$  **do**

**if**  $L_k^s \leq L_k^a$  **then**

$I_k^{\text{opt}} = 0, \quad N_k^{d,\text{opt}} = N_k^{u,\text{opt}} = p_k^{d,\text{opt}} = p_k^{\text{tx,opt}} = 0.$

**else**

$I_k^{\text{opt}} = 1.$

**end**

**end**

**S4.** Compute  $L^s$  and  $L^a$  from (26) and (31), respectively.

**S5. if**  $L^s \leq L^a$  **then**

$I_a^{\text{opt}} = 0, \quad N_k^{d,\text{opt}} = N_k^{u,\text{opt}} = p_k^{d,\text{opt}} = p_k^{\text{tx,opt}} = 0, \forall k.$

**else**

$I_a^{\text{opt}} = 1.$

**end**

---

if  $L^a < L^s$ , and  $I_a^{\text{opt}} = 0$ , otherwise. The overall procedure for the optimal radio resource allocation in uplink and downlink is summarized in Algorithm 1.

### B. Optimal CPU scheduling

The sub-problem of (25) for CPU scheduling at the ES is formulated as follows:

$$\begin{aligned}
 \min_{\Phi} \quad & V\alpha_3\tau \left( I_m(p_m^{\text{on}} - p_m^s) + p_m^s + \kappa f_c^3 \right) \\
 & - \tau \sum_{k=1}^K \tilde{Q}_k f_k J_k + V\alpha_3\tau_s p_m^s \\
 \text{subject to} \quad & \\
 (a) \quad & f_c \in \mathcal{F}; \quad (b) \quad 0 \leq f_k \leq \min \left( \frac{Q_k^m + 1}{\tau J_k}, f_c \right), \quad \forall k; \\
 (c) \quad & \sum_{k=1}^K f_k \leq f_c,
 \end{aligned} \tag{32}$$

with  $\Phi = [f_c, \{f_k\}_k]$ , and we recall that  $I_m = \mathbf{1}\{f_c\}$ . From (32), we notice that, for a fixed  $f_c$ , the problem is linear in the variables  $\{f_k\}_k$  and can be efficiently solved via a fast iterative algorithm. Thus, we can perform a search for the optimal value of  $f_c$  within  $\mathcal{F}$ . In particular, the overall procedure to select the optimal  $f_c$ , the ES's sleep variable  $I_m$ , and the optimal scheduling frequencies  $\{f_k\}_k$  is described in Algorithm 2. Steps S2-S6 find the optimal CPU resource allocation for a given  $f_c$ : to minimize  $L_c$ , we need to allocate the maximum possible CPU frequency to the UE with the highest  $\tilde{Q}_k$ ; if this leaves some available CPU frequency (cf. step S3), the same principle is applied to the remaining UE. Note also that the  $|\mathcal{F}|$  iterations over the possible  $f_c$  (steps S1-S7) can be easily parallelized, being independent from each other. From a complexity point of view, even when not parallelized, it is important to notice that Algorithm 2 requires, in the worst case,  $K \times |\mathcal{F}|$  iterations.

---

**Algorithm 2** ES CPU Scheduling
 

---

In each time slot  $t$ , observe  $Q_k^m, Q_k^a, Z_k, Y_k$ .

Define the  $|\mathcal{F}| \times 1$  vector of the available CPU frequencies  $\varphi = \mathcal{F}$ . Define the  $|\mathcal{F}| \times K$  matrix  $F = \{F_{ik}\}_{i,k}$ , and the  $|\mathcal{F}| \times 1$  vector  $L = \{L_i\}_{i=1}^{|\mathcal{F}|}$ . Set  $F_{ik} = 0 \forall i, k$ , and  $L_i = 0 \forall i$ .

**for**  $i = 1, \dots, |\mathcal{F}|$  **do**

**S1.** Let  $\tilde{\varphi} = \varphi_i$ ,  $I_m = \mathcal{I}\{\tilde{\varphi}\}$ , and  $\mathcal{U} = \{k = 1, \dots, K\}$ .

**while**  $\tilde{\varphi} > 0$  and  $\mathcal{U} \neq \emptyset$  **do**

**S2.**  $\tilde{k} = \arg \max_{k \in \mathcal{U}} \{J_k \tilde{Q}_k\}$ .

**S3.**  $F_{i\tilde{k}} = \min((Q_{\tilde{k}}^m + 1)/(\tau J_{\tilde{k}}), \tilde{\varphi})$ .

**S4.**  $\mathcal{U} = \mathcal{U} \setminus \{\tilde{k}\}$ .

**S5.**  $\tilde{\varphi} = \tilde{\varphi} - F_{i\tilde{k}}$ .

**end**

**S6.** Define  $\bar{k} = \{k : \tilde{Q}_k \leq 0\}$ , and set  $f_{\bar{k}} = 0$ .

**S7.** Compute the objective function  $L_c$  of (32) with  $f_c = \varphi_i$  and  $f_k = F_{ik}, \forall k$ ; save it in  $L_i$ .

**end**

**S8.** Find  $i^* = \arg \min_i \{L_i\}$ ,  $f_c^{\text{opt}} = \varphi_{i^*}$ ,  $f_k^{\text{opt}} = F_{i^*k} \forall k$ .

---

### C. End-to-end probabilistic delay constraint adaptation

We now present an online adaptation method to set the parameter  $\delta_k$  so that (14) accurately represents (13). Given a starting point  $\delta_k(0)$ , the parameter is updated at each time slot as follows:

$$\delta_k(t) = \max(\delta_k(t-1) - \nu_k(t)(P_k(D_k^{\text{max}}, W_k^t) - \epsilon_k), 1), \tag{33}$$

where  $P_k(D_k^{\text{max}}, W_k(t))$  is a moving estimate of the out-of-service probability evaluated on the set  $W_k^t$  (of size  $|W_k^t|$ ) composed by the last data units received by UE  $k$  until  $t$ :

$$P_k(D_k^{\text{max}}, W_k^t) = \frac{1}{|W_k^t|} \sum_{w \in W_k^t} \mathcal{I}\{D_k^w > D_k^{\text{max}}\}, \tag{34}$$

where  $D_k^w$  is the end-to-end delay of the  $w$ -th data unit in  $W_k^t$ . In particular,  $|W_k(t)|$  is the minimum between a given value (chosen to accurately estimate the probability), and the actual number of received data until time  $t$ , due to the fact that, at the beginning, there might be no sufficient data to estimate the probability. Furthermore,  $\nu_k(t)$  is a stepsize sequence, typically chosen either constant or using the diminishing rule

$$\nu_k(t) = \frac{\nu_k(0)}{t^{\beta_k}}, \quad \beta_k \in (0, 1]. \tag{35}$$

The rationale behind the adaptation rule in (33) is the following: first, we know from the theoretical analysis of Section IV and from Proposition 1 that, for a given  $\delta_k$ , Algorithm 1 and 2 yield a solution to (16) that satisfies (14). Therefore, if (14) is satisfied but the current estimated  $P_k(D_k^{\text{max}}, W_k^t)$  is actually greater than the desired value  $\epsilon_k$ , it means that  $\delta_k Q_k^{\text{avg}}$  misrepresents  $D_k^{\text{max}}$  and it is actually greater than it should. Consequently,  $\delta_k$  has to decrease at the next time step in order to impose a tighter threshold and let  $\delta_k Q_k^{\text{avg}}$  better represent  $D_k^{\text{max}}$ . The opposite happens instead, when  $P_k(D_k^{\text{max}}, W_k^t) < \epsilon_k$ , to achieve a lower-energy solution.

---

**Algorithm 3** Discontinuous Computation Offloading (DisCO)

**Input data:**  $K, \mathcal{F}, B_k^u, B_k^d, p_k^{\text{tx,max}}, p_k^{\text{d,max}}, J_k, \mathcal{M}_k^u, \mathcal{M}_k^d, \alpha_1, \alpha_2, \alpha_3, D_k^{\text{avg}}, D_k^{\text{max}}, \epsilon_k, \mu_k$ .

In each slot  $t$  do:

**S1.** Find the optimal radio and computation resource allocation with Algorithms 1 and 2, respectively, and run accordingly the computation offloading procedure.

**S2.** Update the physical and virtual queues as in (7), (9), (10), (17) and (18), respectively.

**S3.** Update  $\nu_k$  as in (35), estimate  $P_k(D_k^{\text{max}}, W_k^t)$  as in (34), and update  $\delta_k$  as in (33).

---

Finally, the overall dynamic strategy is described in Algorithm 3, which will be termed as DisCO.

## VI. NUMERICAL RESULTS

We present here simulation results to assess the performance of our online optimization strategy. All simulations are performed in Matlab®, with the following fixed settings.

**Fixed settings.** We consider a picocell placed at the center of a square area of side 150 m. We assume an FDD system, with total available bandwidth  $B = 10$  MHz equally split between uplink and downlink. We consider a total time slot duration  $\tau_l = 10$  ms, with  $\tau_s = 1$  ms the portion of the slot used for control signaling and optimization, i.e. where all entities are in active state. Then, the slot duration for data transmission and computation is  $\tau = 9$  ms. Therefore, the sleep modes of all entities have to be selected according to these values, taking into account the specific transition times. For the AP power consumption, among the different models available in the literature, we exploit that of [49], which provides a tool, available online, to model the power consumption of base stations of different kinds, with details on the specific components (power amplifiers, supply power, etc.). However, our proposed optimization strategy is not constrained to the use of this model; it is more general and can be applied to different models. Recalling the notation of Section III-A1, in the case of a picocell, the AP active power is  $p_a^{\text{on}} = 2.2$  W. The maximum transmit power of the AP is set to 251 mW [56], so that the maximum transmit power of each user is  $251/K$  mW. In [49], four possible Sleep Modes (SM) are defined, with different minimum sleep periods, corresponding to the OFDM symbol, the sub-frame duration, the radio frame duration, and a standby mode. For our simulations, we exploit Sleep Mode 2 from [49], whose minimum sleep time is 1 ms, while a single transition (e.g. from sleep to active) requires 0.5 ms [49], so that a total duration  $\tau_s = 1$  ms is enough, considering 0.5 ms at the beginning of the slot (for optimization and eventual transition to sleep state) and 0.5 ms at the end of the slot to wake up and being active at the beginning of the next slot (see up right part of Fig. 1). The power consumption in sleep mode 2 is  $p_a^s = 278$  mW [49].

The channel model is taken from [57], with a carrier frequency of 28 GHz, and a Rayleigh fading with unit variance. The noise power spectral density is  $N_0 = -174$  dBm/Hz, with an additional noise figure of 5 dB both at UE and at the

AP. For the UE power consumption, recalling the notation of Section III-A2, we exploit the empirical model of [58], where it is shown that the active power is about  $p_k^{\text{on}} = 0.9$  W, and is also affected by transmit powers above 10 mW, consuming an additional 0.6-1.5 W. Here, we assume a maximum transmit power  $p_k^{\text{tx,max}} = 100$  mW per UE. According to [58], the power  $p_k^u(t)$  consumed to transmit is a monotone increasing function of the transmit power  $p_k^{\text{tx}}(t)$ . For the sleep operation, similarly to the AP case, two different states are defined [58]: a light sleep mode, with power consumption  $p_k^s = 346$  mW and sub-ms transition time, and a deep sleep mode, with  $p_k^s = 20.3$  mW and much longer transition time (around 10 ms). In this paper, we exploit the light sleep operation, due to the sub-ms transition time. For the numerical model presented in Section III-C, we can choose all  $M$ -QAM modulations with  $M \in \{4, 16, 64, 256\}$ , coupled with coding rates in  $\{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ , both in uplink and in downlink, so that  $\mathcal{M}_k^u$  and  $\mathcal{M}_k^d$  have 28 elements. The packet length used in Section III-C is 1500 bytes. The ES has a maximum CPU cycle frequency  $f_{\text{max}} = 4.5 \times 10^9$  CPU cycle/s and an effective switched processor capacitance  $\kappa = 10^{-27}$  W ·  $\left(\frac{\text{s}}{\text{CPU cycle}}\right)^3$  [52]. The vector of all possible CPU cycle frequencies is  $\varphi = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1] \times f_{\text{max}}$ . Finally, recalling the notation of Section III-A3, the power consumption in active state is  $p_m^{\text{on}} = 20$  W, whereas the sleep state power consumption is  $p_m^s = 10$  W.

*Energy-Delay trade-off:* As a first numerical result, we illustrate the performance of DisCO in terms of energy-delay trade-off. We run our simulations with random configurations of the following parameters: the input and output data size  $S_k^i = 10^x$ ,  $S_k^o = 10^y$  bits, with  $x$  and  $y$  uniformly randomly generated (u.r.g.) in [2, 3] and [1, 3], respectively; we assume Poisson arrivals with  $A_k^{\text{avg}}$  u.r.g. in [5, 15] data units; finally,  $J_k = 10^{-z}$  data/CPU cycle, with  $z$  u.r.g. in [2, 5]. The simulation has run for  $T = 10^5$  slots and it has been repeated over 100 independent realizations of the above random parameters and of  $K = 5$  users' positions, uniformly distributed in a square of side 150 m. All UE have an average delay requirement  $D_k^{\text{avg}} = 100$  ms and, for this simulation,  $\delta_k$  is fixed for all  $k$ , with  $\delta_k = [1.5, 1.6, 1.7, 1.8, 1.9]$ . The out-of-service constraint is  $\epsilon_k = 10^{-2}$ , with stepsize  $\mu_k = 10, \forall k$  (cf. (18)). We assume the bandwidth to be equally shared among all UE and a target PER of  $10^{-4}$ , both in uplink and downlink. In Fig. 2a, we show the trade-off between the long-term average of (15) and the average E2E delay, defined in (12). This trade-off is obtained by increasing the Lyapunov parameter  $V$  (cf. (21)) from right to left, as shown in the figure. We plot this trade-off for different settings of the weighting parameters  $\alpha_i, i = 1, 2, 3$  in (15), which also correspond to proper customization of works previously appeared in the literature to our framework and system model. *i)* **UE-centric setting (▲):** This strategy is obtained by setting  $\alpha_1 = 1, \alpha_2 = \alpha_3 = 0$  (cf. (15)), to only consider the UE energy consumption. This strategy could be possibly related to our previous work [27], where only the UE's energy consumption is optimized. *ii)* **AP-centric setting (◆):** This strategy is obtained by setting  $\alpha_2 = 1, \alpha_1 = \alpha_3 = 0$ , to consider only the AP energy consumption. A radio-centric

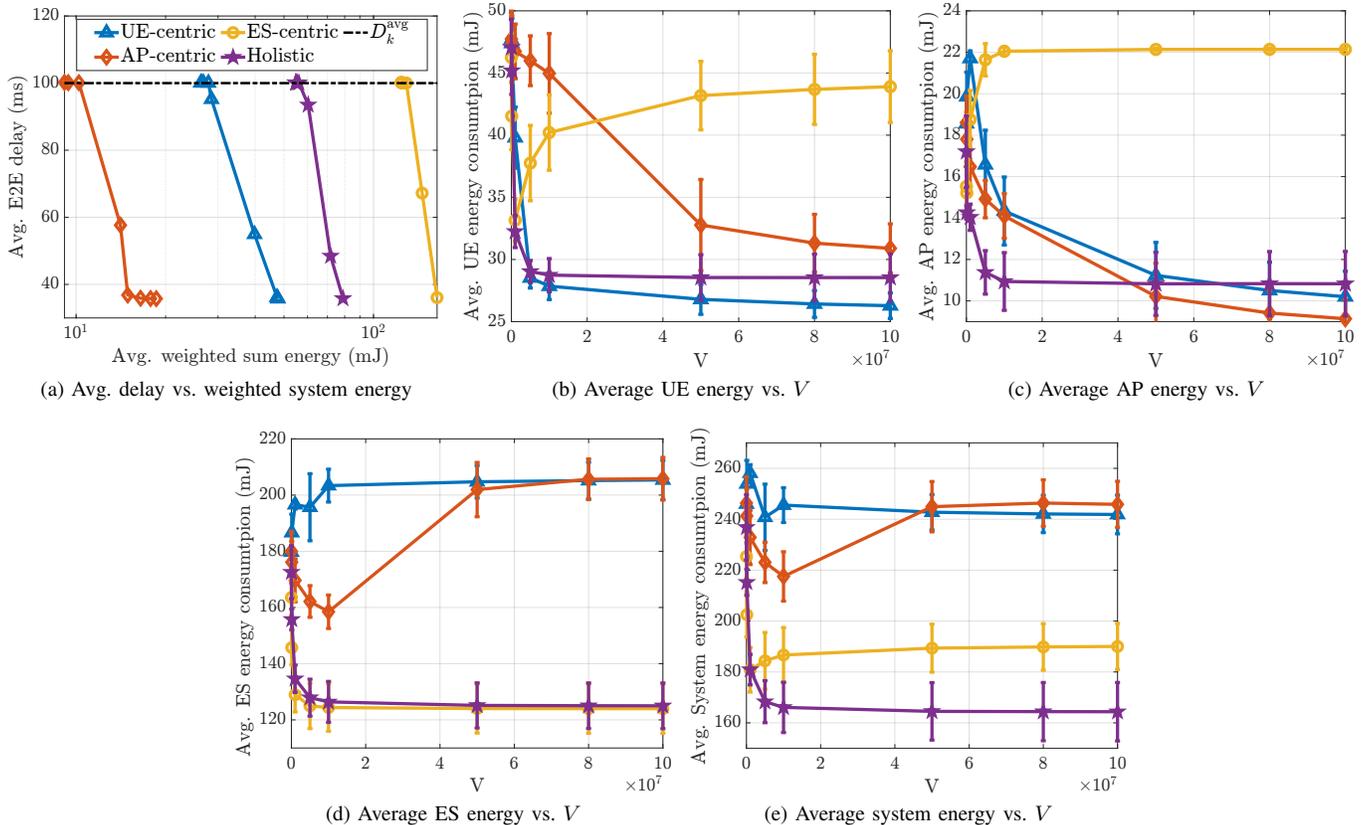


Fig. 2: Energy-delay trade-off.

optimization is proposed in [38], where the authors aim to minimize the sum of UE’s and AP’s energy consumption in a multi-AP scenario. *iii*) **ES-centric setting** (●): This strategy is obtained by setting  $\alpha_3 = 1$ ,  $\alpha_1 = \alpha_2 = 0$ , to only consider the ES energy consumption; *iv*) **Holistic solution** (★): This strategy is obtained by setting  $\alpha_1 = \alpha_2 = \alpha_3 = 1/3$ , to take into account the *overall network energy consumption*. We can notice how the average weighted energy decreased as  $V$  increases, while the average E2E delay increases until reaching its maximum value  $D_k^{avg}$  imposed by constraint (a) of (16), as suggested by the theoretical result in Proposition 1, for all strategies. Looking at Fig. 2a, one may conclude that the AP-centric strategy (◆) is the best one because it achieves the best trade off between average weighted sum energy and delay. However, this does not give any clue on the energy consumption of the single agents and the network. Therefore, we now wonder what is the behavior of the single sources of energy consumption. Let us notice that, for the highest values of  $V$ , all strategies reach the same E2E delay, so that we can compare them in terms of energy consumption, given an E2E delay. Thus, in Fig. 2b-e, we show the long-term average energy consumption of all users, the AP, the ES, and the overall energy consumption (the sum of the three), all as a function of the Lyapunov tradeoff parameter  $V$  (cf. (21)), with the same value of  $V$  as for Fig. 2a. Some comments follow:

(a) **UE-centric setting** (▲). In this setting, the energy consumption of the UE (Fig. 2b) reaches its lowest level,

while the energy consumption of the ES (Fig. 2d) is not optimized. Instead, the AP’s energy consumption (Fig. 2c) reaches a level very close to its lowest, obtained with the AP-centric setting (◆). This is due to the fact that the AP tends to operate in sleep mode when no UE transmits or requests results back, which happens often in the user-centric setting.

- (b) **AP-centric setting** (◆). In this case, the energy consumption of both the AP and the UE approach lower values, for similar reasons as the previous case. This suggests that there exists a strong link between the two entities, since they must be active at the same time when they need to communicate. We can interpret (a) and (b) as “radio-centric” solutions.
- (c) **ES-centric setting** (●). This solution, yields the lowest possible energy consumption for the ES as expected, but it is detrimental for the radio part, incurring additional energy consumption for the AP and the users.
- (d) **Holistic solution** (★). This solution aims at minimizing the overall system energy consumption, This is the most interesting and promising strategy, since it is globally “green” and it reaches very close-to-optimal energy consumption for each agent (UE, AP, ES). This suggests that the three sources of energy consumption can be minimized jointly without detrimental effects on the single agents. Practically, the choice of the  $\alpha_i$  is based on the particular needs of the telecom operator, the MEC

operator, or the UE, but could be also based on a global and holistic energy reduction policy. In this paper, we do not tackle the problem of optimizing the  $\alpha_i$  for the different needs and leave it for future investigation.

This first result motivates us to fix  $\alpha_i = 1/3$ ,  $i = 1, 2, 3$  (holistic solution) for the next simulations.

**Reliability:** Fig. 3a focuses on the out-of-service constraint, i.e. constraint (b) of (16), and shows the effectiveness of the adaptive parameter  $\delta_k$  in (14). The scenario is composed of 4 UE, Poisson arrivals with  $A_k^{\text{avg}} = 5$ ,  $S_k^i = 1000$  and  $S_k^o = 100$  bits,  $J_k = 10^{-4}$  data/CPU cycle,  $D_k^{\text{avg}} = 100$  ms,  $D_k^{\text{max}} = [250, 200, 150, 120]$  and a reliability requirement  $\epsilon_k = 10^{-3}$ , with  $\mu_k = 20$ . The adaptation of  $\delta_k$  is obtained with starting point  $\delta_k(0) = 1 \forall k$ ,  $\nu_k(0) = [15, 5, 4, 3]$ ,  $k = 1, 2, 3, 4$ , and the diminishing rule in (35) uses  $\beta_k = 1/2$ ,  $\forall k$ . The probability of exceeding the desired maximum delay ( $P_k(D_k^{\text{max}}, W_k^t)$  in (34)) is estimated over the most recent  $10^4$  data result arrivals (i.e.  $|W_k(t)| = 10^4$ ). The target PER is  $10^{-4}$ , and the trade-off parameter is  $V = 5 \times 10^6$ . The simulation is run for  $10^5$  slots. Then, Fig. 3a shows the reliability function (also known as survivor function), defined as  $1 - \text{CDF}(D_k)$ , with  $\text{CDF}(D_k)$  being the cumulative distribution function of the end-to-end delay experienced by all data of user  $k$ . The delay is measured by timestamping each data unit. Thus, each curve in Fig. (3a) shows the probability that the end-to-end delay of each data unit exceeds the value on the abscissa. The black dotted horizontal line represents the requirement  $\epsilon_k$  on the out-of-service probability (cf. (14)). For each UE, the points corresponding to  $D_k^{\text{max}}$ ,  $k = 1, 2, 3, 4$  are circled; they all lie below the horizontal black dotted line and the reliability constraint is met. We also show, for each UE, the average energy consumption  $E_k^{\text{avg}} = \frac{1}{T} \sum_{t=1}^T \mathbb{E}\{E_k(t)\}$ . In particular, the average system energy consumption resulting from the minimum delay strategy (i.e., always transmit) is 245 mJ, while the average system energy consumption necessary to achieve the result of Fig. 3a is much lower (160 mJ). The evolution of  $\delta_k(t)$  over time and its convergence are illustrated in Fig. 3b. As expected, a lower  $D_k^{\text{max}}$  requires a lower  $\delta_k$ . Finally, Fig. 3c illustrates the instantaneous out-of-service probability obtained via the adaptive strategy, which flattens around  $\epsilon_k$  after a transient interval. Note that the choice  $\delta_k(0) = 1$  is conservative and helps limiting the out-of-service probability when the convergence of the algorithm is not reached yet. Then, over time, the constraint is relaxed thanks to the adaptation rule of  $\delta_k$ , which helps reducing the energy consumption.

**Comparison of different sleep modes strategies:** We now compare DisCO with four different resource strategies, which correspond to specific customization of other works to our setting. *i) Equal  $f_k$ 's:* resources are optimized (including  $f_c$ ) but the CPU frequencies are equally allocated to each user by the ES (i.e. without Algorithm 2 for CPU scheduling). *ii) No sleep:* resources are optimized but the network elements cannot be turned to sleep states. This could be possibly related to our previous work [27], where we jointly optimize radio and computation resources in a user-centric fashion,

not exploiting sleep modes. *iii) Radio sleep:* resources are optimized but the sleep state of the ES is not available. This is analogous to the approach of [38], once customized to our system model (i.e. single AP), where the authors only exploit AP sleep states. *iv) ES sleep:* resources are optimized but the AP and the users cannot enter the sleep states. This is coherent with the results of [34], where a sleep state at the ES is considered, but no sleep is exploited for UE and AP. Also, we propose a different strategy for bandwidth allocation, based on the following heuristic: let  $\tilde{Q}_k^u = 4Q_k^m - 2Q_k^l + (Z_k + \mu_k Y_k)Q_k^l$  and  $\mathcal{K}_u^+ = \{k : \tilde{Q}_k^u > 0\}$  for the uplink; similarly, let  $\tilde{Q}_k^d = 4Q_k^a + (Z_k + \mu_k Y_k)Q_k^a$  and  $\mathcal{K}_d^+ = \{k : \tilde{Q}_k^d > 0\}$  for the downlink. We define the below uplink (downlink) bandwidth allocation rule:

$$B_k^u = \begin{cases} \frac{\tilde{Q}_k^u}{\sum_{i \in \mathcal{K}_u^+} \tilde{Q}_i^u} B_u, & \text{if } k \in \mathcal{K}_u^+, \\ 0, & \text{otherwise,} \end{cases}$$

$$B_k^d = \begin{cases} \frac{\tilde{Q}_k^d}{\sum_{i \in \mathcal{K}_d^+} \tilde{Q}_i^d} B_d, & \text{if } k \in \mathcal{K}_d^+, \\ 0, & \text{otherwise,} \end{cases} \quad (36)$$

where  $B_u$  and  $B_d$  are the total available uplink and downlink bandwidths, respectively. This heuristic for the allocation of spectral resources is based on the fact that all the information about the status of a certain UE's quality of service lies in the physical and virtual queues. Thus, a UE with a higher  $\tilde{Q}_k^u$  ( $\tilde{Q}_k^d$  for the downlink part), which is defined based on the objective function of (25), needs more resources to drain its queues.

We run our simulations with random configurations of the following parameters:  $S_k^i = 10^x$ ,  $S_k^o = 10^y$  bits, with  $x$  and  $y$  u.r.g. in  $[1, 3]$ . We assume Poisson arrivals with  $A_k^{\text{avg}}$  u.r.g. in  $[1, 20]$  data units. Finally,  $J_k = 10^{-z}$  data/CPU cycle, with  $z$  u.r.g. in  $[2, 5]$ . We consider a scenario with 10 users, all with an average delay requirement  $D_k^{\text{avg}} = [80, 85, 90, 95, 100, 105, 110, 115, 120, 125]$  ms, and  $\delta_k = [1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4]$ . The simulation is run for  $10^4$  slots and the results are averaged over 100 independent realizations of the above parameters and UE' positions. In Fig. 4a, we observe the non-negligible gain of DisCO in terms of average system energy consumption, when compared to all the proposed alternative strategies. The heuristic for bandwidth allocation described in (36) and termed as "DisCO (BW Heur.)" in Fig. 4a achieves an additional gain around 10% with respect to DisCO with equal bandwidth allocation. Of course, other heuristics can be investigated and integrated with our strategy. For instance, at each  $t$  (or a longer time scale), it is possible to compare the solutions obtained with different bandwidth allocation strategies and select the best one, if this is compatible with a practical implementation. A recent contribution suggests this possibility, with a parallel GPU based implementation [59].

**The effect of the arrival rate:** In Fig. 4b, we compare the average system energy consumption of DisCO with other strategies, considering different values of the parameter  $A_k^{\text{avg}}$ ,  $\forall k$ . The scenario involves 15 UE;  $S_k^i = 10^x$ ,  $S_k^o = 10^y$  bits, with  $x$  and  $y$  u.r.g. in  $[2, 3]$  and  $[1, 3]$ , respectively;  $J_k = 10^{-z}$

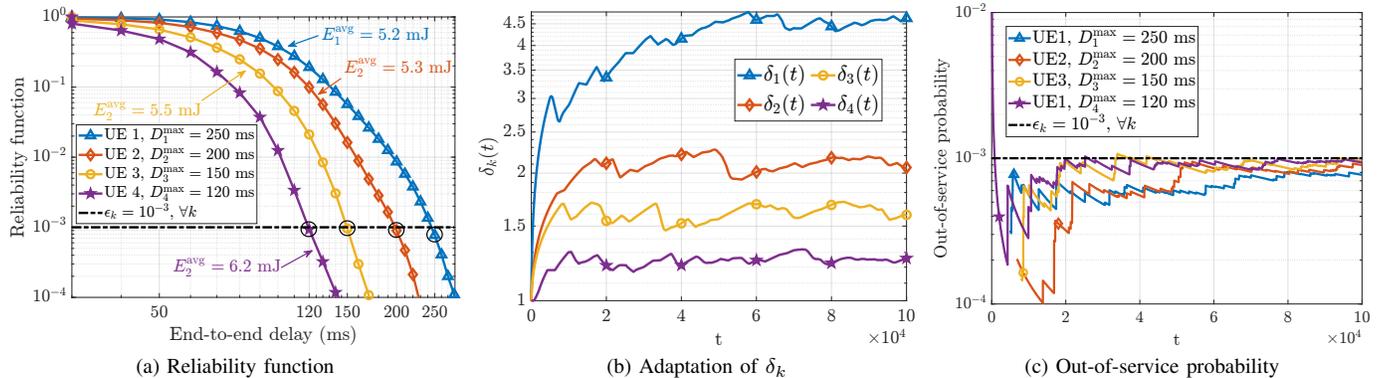


Fig. 3: Performance in terms of reliability

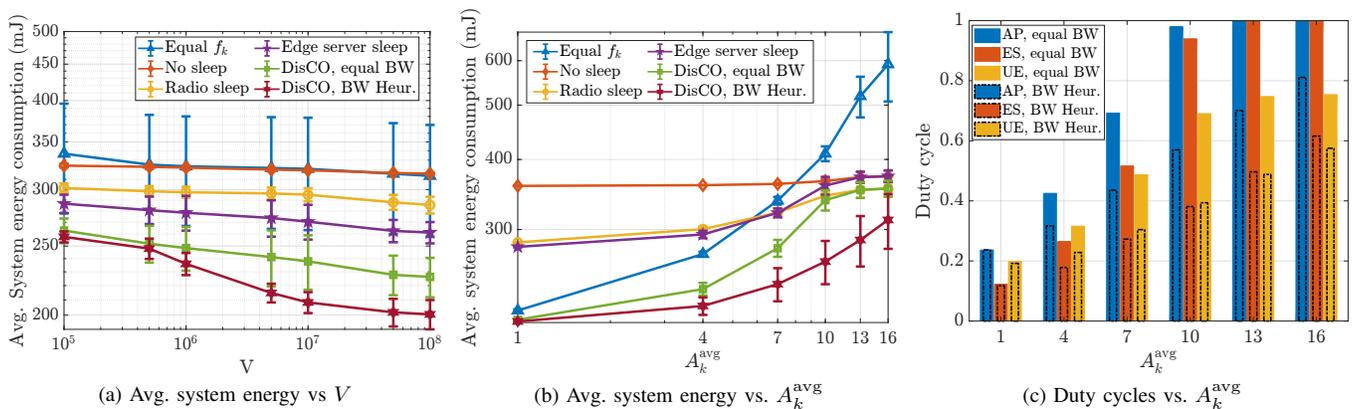


Fig. 4: Comparison of DisCO with other strategies

data/CPU cycle, with  $z$  u.r.g. in [2, 5]; the average delay constraint is  $D_k^{\text{avg}} = 100$  ms,  $\delta_k = 2$ ,  $\epsilon_k = 10^{-2}$ ,  $\mu_k = 10$ ,  $\forall k$ . The Lyapunov trade-off parameter is  $V = 5 \times 10^7$ . The simulation is run for  $10^4$  slots and the results are averaged over 100 independent realizations of the above parameters and UE positions. Fig. 4b shows how DisCO is able to yield a large gain compared to the other strategies, except for high arrival rates, where there are less degrees of freedom to exploit the sleep mode operations. In particular, the duty cycles (fraction of activity time) obtained with DisCO are shown in Fig. 4c, as a function of  $A_k^{\text{avg}}$ . We considered the same setting used for Fig. 4b, using DisCO with equal bandwidth allocation, and with the heuristic described in (36). Fig. 4c shows that, for high  $A_k^{\text{avg}}$ , the duty cycles of DisCO are close to 1 (i.e., always active), thus explaining the similar energy consumption as the strategies without sleep control. However, with our proposed heuristic for bandwidth allocation, we achieve a non-negligible gain in terms of activity time with respect to the equal bandwidth allocation strategy. This result further motivates taking into account the physical and virtual queues in prioritizing the scheduling of the users.

## VII. CONCLUSIONS

In this paper, we proposed a dynamic resource allocation algorithm for computation offloading that jointly exploits low-power sleep modes of UE, AP, and ES to reduce the system

energy consumption with guaranteed E2E average delay and reliability. Via Lyapunov stochastic optimization, we solved a long-term problem, using a dynamic algorithm that works on a per-slot basis, without assuming any prior knowledge on the statistics of data arrivals and radio channels, and with theoretical guarantees. Several numerical results show the performance gain offered by our proposed online strategy, and how a holistic view of the system can be beneficial for all agents and for the global energy consumption. In this paper, we focused on a multiuser setting with a single AP and single ES. Future investigations should include optimized scheduling of spectral and time radio resources in a multi-cell multi-server scenario, where the cooperation among multiple APs and ESs can help reducing the overall energy consumption. Furthermore, non-cooperative methods, including purely game-theoretic approaches or incentive-based mechanisms (see e.g., [29], [42]), are worth of being investigated, as a way to achieve distributed and efficient solutions, while minimizing signaling overhead. Finally, due to the partial knowledge of the communication and computation models involved, it is worth investigating both (partial) data-driven approaches, e.g. DRL methods, and (partial) model-based approaches, where whichever information, albeit limited, is incorporated and exploited to find efficient solutions.

## APPENDIX

Here, we present the derivation of the upper bound of the Lyapunov drift-plus-penalty that leads to the per-slot optimization strategy in (22). First of all, note that, given a generic virtual queue  $X(t)$  evolving as  $X(t+1) = \max(0, X(t) + x(t+1) - \bar{x})$ , and defining  $\Delta_X(t) = \frac{X^2(t+1) - X^2(t)}{2}$ , we can always write  $\Delta_X(t) \leq \frac{(x(t+1) - \bar{x})^2}{2} + X(t)x(t+1) - X(t)\bar{x}$  [55, p. 59]. Then, for the virtual queue  $Z_k(t)$  defined in (17), we can write

$$\begin{aligned} \Delta_Z(t) &\leq \frac{(Q_k^{\text{tot}}(t+1) - Q_k^{\text{avg}})^2}{2} + Z_k(t)Q_k^{\text{tot}}(t+1) \\ &\quad - Z_k(t)Q_k^{\text{avg}} = \frac{1}{2} (Q_k^{\text{tot}}(t+1))^2 + \frac{1}{2} (Q_k^{\text{avg}})^2 \\ &\quad - Q_k^{\text{tot}}(t+1)Q_k^{\text{avg}} + Z_k(t)Q_k^{\text{tot}}(t+1) - Z_k(t)Q_k^{\text{avg}} \\ &\leq (Q_k^l(t+1))^2 + (Q_k^m(t+1) + Q_k^a(t+1))^2 + \frac{1}{2} (Q_k^{\text{avg}})^2 \\ &\quad + Z_k(t)Q_k^{\text{tot}}(t+1) - Z_k(t)Q_k^{\text{avg}} \leq (Q_k^l(t+1))^2 \\ &\quad + 2(Q_k^m(t+1))^2 + 2(Q_k^a(t+1))^2 + \frac{1}{2} (Q_k^{\text{avg}})^2 \\ &\quad + Z_k(t)Q_k^{\text{tot}}(t+1) - Z_k(t)Q_k^{\text{avg}}. \end{aligned} \quad (37)$$

Now, for  $A, b \geq 0$  we have  $(\max(0, Q - b) + A)^2 \leq Q^2 + A^2 + b^2 + 2Q(A - b)$  [55, p. 33]; recalling (7), (9) and (10) and applying the upper bound to all queues, we can write

$$\begin{aligned} \Delta_Z(t) &\leq (Q_k^l(t))^2 + (A_{k,\text{max}})^2 + (N_{k,\text{max}}^u)^2 + 2(Q_k^m(t))^2 \\ &\quad + 2Q_k^l(t)(A_k(t) - N_k^u(t)) + 2(N_{k,\text{max}}^u)^2 + 2(N_{k,\text{max}}^c)^2 \\ &\quad + 4Q_k^m(t)(N_k^u(t) - N_k^c(t)) + 2(Q_k^a(t))^2 + 2(N_{k,\text{max}}^c)^2 \\ &\quad + 2(N_{k,\text{max}}^d)^2 + 4Q_k^a(t)(N_k^c(t) - N_k^d(t)) + \frac{1}{2} (Q_k^{\text{avg}})^2 \\ &\quad + Z_k(t)(\max(0, Q_k^l(t) - N_k^u(t)) + A_k(t) \\ &\quad + \max(0, Q_k^m(t) - N_k^c(t)) + \min(Q_k^l(t), N_{k,\text{max}}^u)) \\ &\quad + \max(0, Q_k^a(t) - N_k^d(t)) \\ &\quad + \min(Q_k^m(t), N_{k,\text{max}}^c) - Q_k^{\text{avg}}), \end{aligned} \quad (38)$$

where we used the fact that  $\min(Q_k^m(t), N_k^c(t)) \leq \min(Q_k^m(t), N_{k,\text{max}}^c)$ , and  $\min(Q_k^l(t), N_k^u) \leq \min(Q_k^l(t), N_{k,\text{max}}^u)$ , where  $N_{k,\text{max}}^u$  is the maximum number of uplink transmitted data units, and  $N_{k,\text{max}}^c$  is the maximum number of computable data units, given (8). For the virtual queue  $Y_k(t)$  (cf. (18)), we can write

$$\begin{aligned} \Delta_Y(t) &\leq \frac{\mu_k^2 (u\{Q_k^{\text{tot}}(t+1) - \delta_k Q_k^{\text{avg}}\} - \epsilon_k)^2}{2} \\ &\quad + \mu_k Y_k(t) (u\{Q_k^{\text{tot}}(t+1) - \delta_k Q_k^{\text{avg}}\} - \epsilon_k) \\ &\leq \frac{\mu_k^2 (1 - \epsilon_k)^2}{2} + \mu_k Y_k(t) (u\{Q_k^{\text{tot}}(t+1) - \delta_k Q_k^{\text{avg}}\} - \epsilon_k), \end{aligned} \quad (39)$$

where we used the fact that  $u\{\cdot\} \leq 1$ . Finally, plugging (38) and (39) into (21), we can write

$$\begin{aligned} \Delta_P(\Theta(t)) &\leq \zeta + \mathbb{E} \left\{ \sum_{k=1}^K \left[ \chi_k(t) - 2Q_k^l(t)N_k^u(t) \right. \right. \\ &\quad + 4Q_k^m(t)(N_k^u(t) - N_k^c(t)) + 4Q_k^a(t)(N_k^c(t) - N_k^d(t)) \\ &\quad + Z_k(t)(\max(0, Q_k^l(t) - N_k^u(t)) \\ &\quad + \max(0, Q_k^m(t) - N_k^c(t)) + \max(0, Q_k^a(t) - N_k^d(t))) \\ &\quad + \mu_k Y_k(t) u \left\{ \max(0, Q_k^l(t) - N_k^u(t)) + A_k(t) \right. \\ &\quad + \max(0, Q_k^m(t) - N_k^c(t)) + \min(Q_k^l(t), N_{k,\text{max}}^u(t)) \\ &\quad + \max(0, Q_k^a(t) - N_k^d(t)) + \min(Q_k^m(t), N_{k,\text{max}}^c(t)) \\ &\quad \left. \left. - \delta_k Q_k^{\text{avg}} \right\} \right] + VE_{\text{tot}}^w(t) \left| \Theta(t) \right\}, \end{aligned} \quad (40)$$

where  $\zeta$  is a positive constant given by

$$\begin{aligned} \zeta &= \sum_{k=1}^K \left[ (A_{k,\text{max}})^2 + 3(N_{k,\text{max}}^u)^2 + 4(N_{k,\text{max}}^c)^2 \right. \\ &\quad \left. + 2(N_{k,\text{max}}^d)^2 + \frac{(Q_k^{\text{avg}})^2}{2} + \frac{\mu_k^2(1 - \epsilon_k)^2}{2} \right], \end{aligned} \quad (41)$$

and  $\chi_k(t)$  is a constant at time slot  $t$  (i.e. it does not depend on the optimization variables), which reads as follows:

$$\begin{aligned} \chi_k(t) &= (2Q_k^l(t) + Z_k(t))A_k(t) + (Q_k^l(t))^2 + 2(Q_k^m(t))^2 \\ &\quad + 2(Q_k^a(t))^2 + Z_k(t)(\min(Q_k^l(t), N_{k,\text{max}}^u(t)) \\ &\quad + \min(Q_k^m(t), N_{k,\text{max}}^c(t)) - Q_k^{\text{avg}}) - \mu_k Y_k(t)\epsilon_k. \end{aligned} \quad (42)$$

Then, the Min-Drift-plus penalty algorithm proceeds by opportunistically minimizing (40) in each time slot, leading to the problem in (22), where all the constant terms (with respect to the variables) are omitted.

## REFERENCES

- [1] S. Ahmadi, *5G NR: Architecture, Technology, Implementation, and Operation of 3GPP New Radio Standards*. Elsevier Science, 2019.
- [2] E. Calvanese Strinati *et al.*, "6G: The Next Frontier: From Holographic Messaging to Artificial Intelligence Using Subterahertz and Visible Light Communication," *IEEE Veh. Tech. Magazine*, vol. 14, no. 3, pp. 42–50, Sep. 2019.
- [3] S. Barbarossa, S. Sardellitti, E. Ceci, and M. Merluzzi, "The edge cloud: A holistic view of communication, computation, and caching," in *Chapter 16 of Cooperative and Graph Signal Processing*. Academic Press, 2018, pp. 419–444.
- [4] A. Ndikumana *et al.*, "Joint Communication, Computation, Caching, and Control in Big Data Multi-access Edge Computing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2019.
- [5] "ETSI Multi-Access Edge Computing," Available Online at <https://www.etsi.org/technologies/multi-access-edge-computing>.
- [6] S. Kekki *et al.*, "MEC in 5G networks," *ETSI white paper*, vol. 7, p. 1–28, 2018.
- [7] ETSI, "Multi-Access Edge Computing (MEC); Phase 2: use Cases and Requirements," October 2018.
- [8] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi, "5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View," *IEEE Access*, vol. 6, pp. 55 765–55 779, 2018.
- [9] Q.-V. Pham, F. Fang, H.-N. Vu, M. Le, Z. Ding, L. B. Le, and W.-J. Hwang, "A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art," *ArXiv*, vol. abs/1906.08452, 2019.

- [10] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.
- [11] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 45–55, Nov. 2014.
- [12] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Net.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.
- [13] G. Auer *et al.*, "How much energy is needed to run a wireless network?" *IEEE Wireless Comm.*, vol. 18, no. 5, pp. 40–49, Oct. 2011.
- [14] S. Tombaz *et al.*, "Energy Performance of 5G-NX Wireless Access Utilizing Massive Beamforming and an Ultra-Lean System Design," in *Proc. of IEEE GLOBECOM 2015*, 2015, pp. 1–7.
- [15] J. Malmudin and D. Lundén, "The Energy and Carbon Footprint of the Global ICT and E&M Sectors 2010–2015," *Sustainability*, vol. 10, no. 9, pp. 1–31, August 2018.
- [16] R. Bonnefoi, C. Moy, and J. Palicot, "Power Control and Cell Discontinuous Transmission Used As a Means of Decreasing Small-Cell Networks' Energy Consumption," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 4, pp. 899–914, Dec. 2018.
- [17] P. Chang and G. Miao, "Interference-aware distributed control of cell discontinuous transmission," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6.
- [18] J. Kim, H. Lee, and S. Chong, "Traffic-Aware Energy-Saving Base Station Sleeping and Clustering in Cooperative Networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 2, pp. 1173–1186, 2018.
- [19] A. De Domenico, R. Gupta, and E. Calvanese Strinati, "Dynamic Traffic Management for Green Open Access Femtocell Networks," in *2012 IEEE 75th Vehicular Technology Conference (VTC Spring)*, 2012, pp. 1–6.
- [20] A. De Domenico, E. Calvanese Strinati, and A. Capone, "Enabling Green cellular networks: A survey and outlook," *Computer Communications*, vol. 37, pp. 5–24, 2014.
- [21] A. De Domenico and D. Kténas, "Reinforcement learning for interference-aware cell DTX in heterogeneous networks," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6.
- [22] W. Labidi, M. Sarkiss, and M. Kamoun, "Energy-optimal resource scheduling and computation offloading in small cell networks," in *Proc. of ICT 2015*, Sydney, NSW, Australia 2015, pp. 313–318.
- [23] C. You, K. Huang, H. Chae, and B. H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wir. Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.
- [24] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.
- [25] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.
- [26] C. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic Task Offloading and Resource Allocation for Ultra-Reliable Low-Latency Edge Computing," *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4132–4150, 2019.
- [27] M. Merluzzi, P. Di Lorenzo, S. Barbarossa, and V. Frascolla, "Dynamic Computation Offloading in Multi-Access Edge Computing via Ultra-Reliable and Low-Latency Communications," *IEEE Transactions on Signal and Information Processing over Networks*, pp. 1–1, 2020.
- [28] D. Han, W. Chen, and Y. Fang, "Joint Channel and Queue Aware Scheduling for Latency Sensitive Mobile Edge Computing with Power Constraints," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2020.
- [29] H. Zhou, X. Chen, S. He, J. Chen, and J. Wu, "DRAIM: A Novel Delay-Constraint and Reverse Auction-Based Incentive Mechanism for WiFi Offloading," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 4, pp. 711–722, 2020.
- [30] F. Sufyan and A. Banerjee, "Computation Offloading for Distributed Mobile Edge Computing Network: A Multiobjective Approach," *IEEE Access*, vol. 8, pp. 149 915–149 930, 2020.
- [31] J. Fang, Y. Chen, and S. Lu, "Energy-Efficient Resource Provisioning Strategy for Reduced Power Consumption in Edge Computing," *Applied Sciences*, vol. 10, no. 17, 2020. [Online]. Available: <https://www.mdpi.com/2076-3417/10/17/6057>
- [32] L. Li, Q. Guan, L. Jin, and M. Guo, "Resource Allocation and Task Offloading for Heterogeneous Real-Time Tasks With Uncertain Duration Time in a Fog Queueing System," *IEEE Access*, vol. 7, pp. 9912–9925, 2019.
- [33] L. Chen, S. Zhou, and J. Xu, "Energy efficient mobile edge computing in dense cellular networks," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.
- [34] S. Wang, X. Zhang, Z. Yan, and W. Wang, "Cooperative Edge Computing with Sleep Control under Non-uniform Traffic in Mobile Edge Networks," *IEEE Internet Things J.*, Oct. 2018.
- [35] P. Chang and G. Miao, "Resource Provision for Energy-Efficient Mobile Edge Computing Systems," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.
- [36] Y. Nan, W. Li, W. Bao, F. C. Delicato, P. F. Pires, Y. Dou, and A. Y. Zomaya, "Adaptive Energy-Aware Computation Offloading for Cloud of Things Systems," *IEEE Access*, vol. 5, pp. 23 947–23 957, 2017.
- [37] Q. Wu, J. Zhou, J. Zhou, J. Weng, Q. Liu, Y. Xing, and S. Xu, "A Computation Offloading Algorithm for Cloud Edge Collaborative Network Based on Sleep Mechanism," in *2021 International Wireless Communications and Mobile Computing (IWCMC)*, 2021, pp. 317–322.
- [38] B. Yu, L. Pu, Q. Xie, J. Xu, and J. Zhang, "U-MEC: Energy-Efficient Mobile Edge Computing for IoT Applications in Ultra Dense Networks," in *WASA*, 2018.
- [39] R. Malik and M. Vu, "Energy-efficient Joint Wireless Charging and Computation Offloading In MEC Systems," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–1, 2021.
- [40] X. W. Z. Chen, "Decentralized computation offloading for multi-user mobile edge computing: a deep reinforcement learning approach," *J Wireless Com Network 2020*, 188 (2020), 2020.
- [41] H. Zhou, K. Jiang, X. Liu, X. Li, and V. C. M. Leung, "Deep Reinforcement Learning for Energy-Efficient Computation Offloading in Mobile Edge Computing," *IEEE Internet of Things Journal*, pp. 1–1, 2021.
- [42] H. Zhou, T. Wu, H. Zhang, and J. Wu, "Incentive-Driven Deep Reinforcement Learning for Content Caching and D2D Offloading," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2445–2460, 2021.
- [43] R. Zhao, X. Wang, J. Xia, and L. Fan, "Deep reinforcement learning based mobile edge computing for intelligent Internet of Things," *Physical Communication*, vol. 43, p. 101184, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1874490720302615>
- [44] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless Networks Design in the Era of Deep Learning: Model-Based, AI-Based, or Both?" *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7331–7376, 2019.
- [45] S. Bi, L. Huang, H. Wang, and Y.-J. A. Zhang, "Lyapunov-guided Deep Reinforcement Learning for Stable Online Computation Offloading in Mobile-Edge Computing Networks," *IEEE Tran. on Wireless Communications*, pp. 1–1, 2021.
- [46] S. Bae, S. Han, and Y. Sung, "A Reinforcement Learning Formulation of the Lyapunov Optimization: Application to Edge Computing Systems with Queue Stability," *Available online: https://arxiv.org/abs/2012.07279*, 2020.
- [47] M. Sana, M. Merluzzi, N. di Pietro, and E. Calvanese Strinati, "Energy Efficient Edge Computing: When Lyapunov Meets Distributed Reinforcement Learning," *Available online: https://arxiv.org/abs/2103.16985*, 2021.
- [48] M. Merluzzi, N. di Pietro, P. Di Lorenzo, E. Calvanese Strinati, and S. Barbarossa, "Network Energy Efficient Mobile Edge Computing with Reliability Guarantees," in *Proc. of IEEE GLOBECOM 2019*, Dec 2019, pp. 1–6.
- [49] B. Debaille, C. Desset, and F. Louagie, "A Flexible and Future-Proof Power Model for Cellular Base Stations," in *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, May 2015, pp. 1–7.
- [50] C. Gough, I. Steiner, and W. Saunders, *Energy Efficient Servers: Blueprints for Data Center Optimization*. Apress, 2015.
- [51] L. Brochard *et al.*, *Energy-Efficient Computing and Data Centers*. Wiley, 2019.
- [52] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *J. VLSI Signal Process. Syst.*, vol. 13, no. 2-3, pp. 203–221, Aug. 1996. [Online]. Available: <http://dx.doi.org/10.1007/BF01130406>
- [53] E. Le Sueur and G. Heiser, "Dynamic Voltage and Frequency Scaling: The Laws of Diminishing Returns," in *Proc. HotPower*, 2010, pp. 1–8.
- [54] J. D. C. Little, "A Proof for the Queuing Formula:  $L = \lambda W$ ," *Oper. Res.*, vol. 9, no. 3, p. 383–387, Jun. 1961.
- [55] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool Publishers, 2010.

- [56] "IMEC Power Model for Wireless Base Stations," <https://www.imec-int.com/en/powermodel>.
- [57] S. Sun, T. S. Rappaport *et al.*, "Propagation Path Loss Models for 5G Urban Micro- and Macro-Cellular Scenarios," in *2016 IEEE 83rd VTC Spring*, May 2016, pp. 1–6.
- [58] M. Lauridsen, L. Noël, T. B. Sorensen, and P. Mogensen, "An Empirical LTE Smartphone Power Model with a View to Energy Efficiency Evolution," *Intel Technology Journal*, vol. 18, pp. 172–193, 03 2014.
- [59] Y. Huang, S. Li, Y. T. Hou, and W. Lou, "GPF: A GPU-based Design to Achieve 100 us Scheduling for 5G NR," in *MobiCom '18*, 2018.



**Mattia Merluzzi** (Member, IEEE) received the M.S. degree in Telecommunication Engineering and the Ph.D. degree in Information and Communication Technologies from Sapienza University of Rome, Italy, in 2017 and 2021, respectively. He is currently a research engineer at CEA-Leti, Grenoble, France, where he is involved in the research team of the H2020 project Hexa-X. He has participated in the H2020 EU/Japan project 5G-Miedge, the H2020 EU/Taiwan project 5G CONNI and the MIUR funded PRIN Liquid Edge. His primary research

interests are in edge computing, beyond 5G systems, stochastic optimization, and edge machine learning. He was the recipient of the 2021 GTTI (Italian National Group on Telecommunications and Information Theory) Award for the Best Ph.D. thesis.



**Nicola di Pietro** received the B.S. degree in mathematics from the University of Padova, Italy, in 2008. In 2010, he received the M.S. degree in mathematics jointly from the University of Padova, Italy, and the University of Bordeaux, France, within the framework of the international ALGANT program. He received the Ph.D. degree in mathematics from the University of Bordeaux, France, in 2014. During the years of his doctoral studies, he was a Research Engineer with the European R&D Center of Mitsubishi Electric in Rennes, France. From 2014

to 2016, he was an Associate Post-Doctoral Fellow at Texas A&M University at Qatar. From 2017 to 2021, he was a Research Engineer with CEA-Leti in Grenoble, France. He is now a System Engineer at Athonet, Italy. He is author of several papers and patents, and his research interests are 5G networks, edge computing, information theory, and lattice error-correcting codes.



**Paolo Di Lorenzo** (Senior Member, IEEE) received the M.Sc. and the Ph.D. degrees in electrical engineering from Sapienza University of Rome, Rome, Italy, in 2008 and 2012, respectively. He is currently an Associate Professor with the Department of Information Engineering, Electronics, and Telecommunications, Sapienza University of Rome. In 2010, he held a visiting research appointment with the Department of Electrical Engineering, University of California at Los Angeles, Los Angeles, CA, USA. From May 2015 to February 2018, he was

an Assistant Professor with the Department of Engineering, University of Perugia, Perugia, Italy. He has participated in the FP7 European research projects FREEDOM, on femtocell networks; SIMTISYS, on moving target detection and imaging using a constellation of satellites; and TROPIC, on communication, computation, and storage over collaborative femtocells. He is a Principal Investigator of the research unit in the H2020 European project RISE 6G. His research interests include signal processing theory and methods, distributed optimization, mobile edge computing, machine learning, and graph signal processing. Prof. Di Lorenzo is currently an Associate Editor for the IEEE Transactions on Signal and Information Processing Over Networks. He was the recipient of the three best student paper awards, respectively, at IEEE SPAWC10, EURASIP EUSIPCO11, and IEEE CAMSAP11. He was also the recipient of the 2012 GTTI (Italian National Group on Telecommunications and Information Theory) Award for the Best Ph.D. thesis.



**Emilio Calvanese Strinati** (Member, IEEE) obtained his Engineering Master degree in 2001 from Sapienza University of Rome and his Ph.D in Engineering Science in 2005. He then started working at Motorola Labs in Paris in 2002. Then in 2006 he joined CEA-Leti as a research engineer. From 2007, he becomes a PhD supervisor. From 2010 to 2012, he has been the co-chair of the wireless working group in GreenTouch Initiative which deals with design of future energy efficient communication networks. From 2011 to 2016 he was the Smart

Devices & Telecommunications European collaborative strategic programs Director. Between December 2016 and January 2020 is was the Smart Devices & Telecommunications Scientific and Innovation Director. From 2017 to 2018 he was one of the three moderators of the 5G future network expert group. Between 2016 and 2018 he was the coordinator of the H2020 joint Europe and South Korea 5GCHAMPION project that showcased at the 2018 winter Olympic Games, 5G technologies in realistic operational environments. Since July 2018 he is the coordinator of the H2020 joint Europe and South Korea 5G-AllStar project. Since 2018 he holds the French Research Director Habilitation (HDR). In 2021 he started the coordination of the H2020 European project RISE-6G, focusing on the design and operation of Reconfigurable Intelligent Surfaces in future high frequency 6G networks. Since February 2021 he is also the director of the New-6G (Nano Electronic & Wireless for 6G) initiative, dedicated to the required convergence between microelectronic & telecom, hardware & software, network & equipment for upcoming 6G technologies. He has published around 150 papers in international conferences, journals and books chapters, given more than 200 international invited talks, keynotes and tutorials. He is the main inventor or co-inventor of more than 70 patents. He has organized more than 100 international conferences, workshops, panels and special sessions on green communications, heterogeneous networks and cloud computing hosted in international conferences as IEEE GLOBCOM, IEEE PIMRC, IEEE WCNC, IEEE ICC, IEEE VTC, EuCNC, IFIP, EuCNC and European Wireless. He is the general chair of EuCNC 2022.



**Sergio Barbarossa** (Fellow, IEEE) received his MS and Ph.D. EE degree from the Sapienza University of Rome, where he is now a Full Professor and Senior Research Fellow of the Sapienza School of Advanced Studies. He has held visiting positions at the Environmental Research Institute of Michigan ('88), Univ. of Virginia ('95, '97), and Univ. of Minnesota ('99). He is an IEEE Fellow, EURASIP Fellow, and he has been an IEEE Distinguished Lecturer. He received the IEEE Best Paper Award from the IEEE Signal Processing Society in the years

2000, 2014, and 2020. He received the Technical Achievements Award from the EURASIP Society in 2010. He coauthored the papers that received the Best Student Paper Award at ICASSP 2006, SPAWC 2010, EUSIPCO 2011, and CAMSAP 2011. He has been the scientific coordinator of several EU projects on wireless sensor networks, small cell networks, distributed mobile cloud computing, and edge computing in 5G networks. He is now leading a national project on edge learning and he is involved in two H2020 European projects on 5G networks for Industry 4.0 and on reconfigurable intelligent surfaces. His current research interests are in the area of mobile edge computing and machine learning, graph signal processing, and distributed optimization. From 1997 to 2003, he was a member of the IEEE Technical Committee for Signal Processing in Communications. He served as an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING (1998-2000 and 2004-2006), the IEEE SIGNAL PROCESSING MAGAZINE, and the IEEE TRANSACTIONS ON SIGNAL AND INFORMATION PROCESSING OVER NETWORKS. He has been the General Chairman of the IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC), 2003 and the Technical Co-Chair of SPAWC, 2013. He has been the Guest Editor for Special Issues on the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, EURASIP Journal of Applied Signal Processing, EURASIP Journal on Wireless Communications and Networking, the IEEE SIGNAL PROCESSING MAGAZINE, and the IEEE SELECTED TOPICS ON SIGNAL PROCESSING.