Latency-Constrained Dynamic Computation Offloading with Energy Harvesting IoT Devices

Conference Paper · May 2019

DOI: 10.1109/INFCOMW.2019.8845302

3 authors:

Mattia Merluzzi
Sapienza University of Rome
12 PUBLICATIONS **83** CITATIONS
SEE PROFILE

Paolo Di Lorenzo
Università degli Studi di Perugia
65 PUBLICATIONS **1,669** CITATIONS
SEE PROFILE

S. Barbarossa
Sapienza University of Rome
279 PUBLICATIONS **10,582** CITATIONS
SEE PROFILE

Latency-Constrained Dynamic Computation Offloading with Energy Harvesting IoT Devices

Mattia Merluzzi, Paolo Di Lorenzo, and Sergio Barbarossa
Sapienza Univ. of Rome, DIET Dept, Via Eudossiana 18, 00184, Rome, Italy
e-mail: {mattia.merluzzi,paolo.dilorenzo, sergio.barbarossa}@uniroma1.it

***Abstract*—In this paper, we address the problem of dynamic computation offloading with Multi-Access Edge Computing (MEC), considering an Internet of Things (IoT) environment where computation requests are continuously generated locally at each device, and are handled through dynamic queue systems. In such context, we consider simple devices (e.g., sensors) with limited battery and energy harvesting capabilities. Hinging on stochastic optimization tools, we devise a dynamic algorithm that jointly optimize radio (e.g., power, energy) and computation (e.g., CPU cycles) resources, while guaranteeing a certain out of service probability (defined as the probability that the sum of local and remote queues exceeds a predefined value) and stability of the device batteries around prescribed operating levels. The method requires the solution of a convex optimization problem per time slot, and does not require apriori knowledge of channel, task and energy arrival distributions. Numerical results illustrate the advantages of the proposed method.**

***Index Terms*—Computation offloading, Mobile Edge computing, Internet of Things, queues, stochastic optimization.**

I. Introduction

The 5G revolution will radically change the concept of mobile networks, since it aims to integrate many services on the same network infrastructure, requiring a flexible network design. The mobile data traffic is experiencing a very high growth [1], also due to a plethora of new services such as Internet of Things, Industry 4.0, automated driving, etc. In the era of data mining, a large amount of data to be processed will be collected/generated by sensors and/or objects, such as IoT devices. In some applications, there is the need to process these data within very short delays, but tiny sensors are not capable of running sophisticated programs. For this reason, data are usually processed in the cloud, where computation resources are virtually infinite. However, the delay to reach the cloud through the public internet is typically much larger than the low latency requirements of some applications. To overcome this issue, MEC brings cloud computing functionalities at the edge of the network, enabling the offloading of sophisticated applications from mobile devices and tiny sensors to small data centers, called Mobile Edge Hosts (MEH). Typically, MEHs are located at the Radio Access Point (RAP), or at an aggregation point of the core network, thus guaranteeing low latency services and high energy efficiency. However, since MEHs have much smaller computation capabilities than the cloud due to space limitation and CAPEX costs, the available resources (i.e., radio, computation, energy) have to be managed properly to provide satisfactory Quality of Service (QoS). In particular, since the end-to-end delay comprises a communication time and a computation time, in a user/application centric architecture it is important to manage these resources in a joint and dynamic manner [2].

This work was supported by H2020 EUJ Project 5G-MiEdge, Nr. 723171.

Related works. As already mentioned, due to the limited computation resources of MEC, resource allocation strategies for computation offloading are an important and pivotal topic in the research community, see, e.g., [2]–[11]. The interested reader can refer to the recent survey in [12]. In [13], the edge cloud is seen as a holistic system comprising communication, computation, and storage resources. Computation offloading strategies can be split into two cathegories: *static* and *dynamic* strategies. The static formulation deals with short time applications, in which users request for a computation with a well defined computational demand; whereas, long-lived applications must be treated in a dynamic fashion, since the application continuously generates data to be processed, without necessarily knowing apriori the statistics of the data. Concerning the static formulation, the work [2] shows the benefit of the joint optimization of radio and computation resources. Recently the sinergy of MEC and mmWave communication has been studied in [3], [4], where the authors investigate the benefits of a high capacity radio access, taking into account the detrimental effect of blocking events, typical of mmWave communications. In particular, in [3], statistical independent blocking event are considered, while in [4] the investigation is extended to the statistical dependent events due to large obstacles, and the overbooking of radio and computation resources is studied as a countermeasure. In [10], the authors jointly assign UE's to mmWave APs and MEHs with two different algorithms: a penalized Successive Convex Approximation based strategy, and a many-to-one matching game algorithm. The authors of [14] devised an algorithm to minimize the energy consumption, in TDMA and OFDMA systems, while in [15] the offloading decision and the allocation of radio and computation resources are jointly optimized. In [7], the authors investigate dynamic strategies using stochastic optimization, with the aim of minimizing the long-term average power consumption under constraints on the mean rate stability of the computation queues with a single MEH. The work [8] presents a fog-enabled D2D scenario with an assignment algorithm between devices. User assignment is

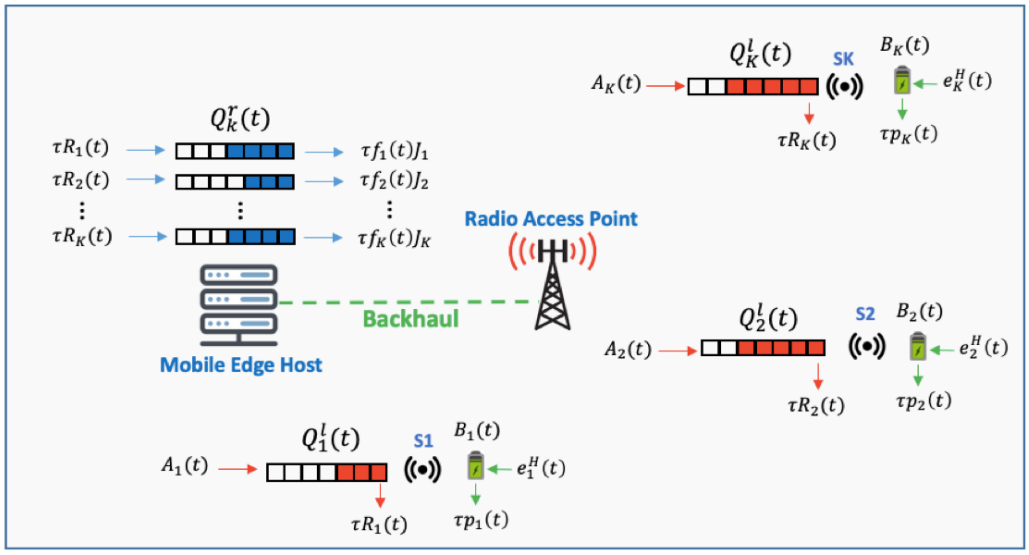

Fig. 1: Network scenario.

also addressed in [9], with the aim of minimizing the average delay under energy constraints, using a penalty function that discourages frequent handovers, while hinging on a Multi-armed bandit algorithm to learn the optimal penalty parameter.

All the aforementioned works do not address the problem of dynamic computation offloading while keeping the computation queues under a certain threshold, in order to limit the service delay. Latency-constrained dynamic computation offloading was first addressed in [11], where the authors introduced a probabilistic constraint on the computation queues, written as a bound on the probability of exceeding a certain value, handling it with extreme value theory. Then, the work in [16] extends [11] by considering a scenario with multiple APs and MEHs, where user assignment is handled with a many-to-one matching game with externalities. Meanwhile, energy harvesting (EH) techniques have attracted a lot of interest in IoT in order to cope with the battery-limited nature of sensor devices, enabling the possibility to collect energy from renewable sources such as wind, sun, vibration, and heat [17]. EH naturally introduce *dinamicity* in the problem due to the intermittent arrivals of energy from the environment and the variability over time of the battery levels at each sensor. In this context, the works in [18], [19], study the optimal packet communication strategy to maximize the net bit rates while stabilizing the data queue in EH communications. An energy scheduling strategy for remote estimation in the case of a single EH sensor is proposed in [20]. Finally, a computation offloading framework based on energy harvesting is introduced in [6], where the authors devise an algorithm that minimizes an execution cost comprising latency and task dropping, while at the same time keeping the battery level stable around a predefined value.

Contribution. In this paper, we propose the first strategy for dynamic multi-user computation offloading with energy harvesting IoT devices, aimed at minimizing the long-term average energy consumption under a bound on the out of service probability, while guaranteing stability of the batteries around a prescribed operating level. The approach is novel with respect to the literature, since it enables dynamic computation offloading with guarantees in terms of both QoS and energy management. In particular, differently from [6], we consider a dynamic multi-user scenario where task arrivals are handled via computation queues, and we consider QoS requirements in terms of the probability that the sum of local and remote queues does not exceed a prescribed level. Also, we differ from [16], which does not consider the possibility to harvest energy from the environment, and impose QoS requirements separately on local and remote queues. The proposed method requires the solution of a convex problem in each time slot, so that it can be solved using efficient numerical tools [21]. Simulation results assess the performance of our solution, illustrating how, in its simplicity, it guarantees out of service probability and stability of the IoT devices' batteries.

II. Problem Formulation

Let us consider a scenario where K devices wish to offload computations to a MEH, which is connected to a RAP via a high capacity backhaul, as in the example of Fig. 1. Time is divided into slots of equal duration τ. The EH process is modeled as successive energy packet arrivals, i.e., $E_k^A(t)$ units of energy arrive at sensor i at the beginning of the t-th time slot. The energy arrivals $E_k^A(t)$ are i.i.d. among different slots, and are upper bounded by $E_{\max}^A$ [22]. In each time slot, part of the arrived energy, say, $E_k^H(t) \leq E_k^A(t)$, will be harvested and stored in the battery, and it will be available for data transmission from the next slot. Let us denote the battery level of node k at time slot t as $B_k(t)$. The transmit energy is subject to the energy causality constraint $e_k(t) \leq B_k(t)$ for all t, so that the battery level evolves according to:

$$B_k(t+1) = B_k(t) - e_k(t) + E_k^H(t), \quad \text{for all } k, t, \qquad (1)$$

Of course, from (1), the battery level is determined by the balance between the energy spent for transmission [i.e., $e_k(t)$] and the one harvested from the environment [i.e., $E_k^H(t)$].

From the radio perspective, letting $p_k(t)$ be the transmit power of sensor k, and considering a fixed transmission time interval τ, the energy consumption related to data transmission at time t is given by $e_k(t) = p_k(t)\tau$. Thus, considering a frequency division multiple access, the maximum data rate between sensor k and the AP at time t is given by:

$$R_k(t) = \beta_k(t) W \log_2 \left(1 + \frac{h_k(t) e_k(t)}{N_0 \beta_k(t) W \tau}\right), \tag{2}$$

where $\beta_k(t)$ is the portion of the bandwidth allocated to sensor k, $h_k(t)$ is the radio channel, W is the total available bandwidth, and N_0 is the noise power spectral density. Each sensor keeps a local (at the sensor) queue of bits to be transmitted to offload computation to the MEH; also, the MEH keeps the computation queues of all sensors (cf. Fig. 1). The local data queue of sensor k, say, $Q_k^l(t)$, takes on input the new data arrivals $A_k(t)$, and it is drained by transferring data to the MEH via the RAP, thus evolving as:

$$Q_k^l(t+1) = \max\left(Q_k^l(t) - \tau R_k(t), 0\right) + A_k(t). \tag{3}$$

Similarly, the remote queue $Q_k^r(t)$ is fed by the data transmitted by the sensors, and it is drained by the computation power of the MEH, as follows:

$$\begin{aligned} Q_k^r(t+1) \; = \; & \max\left(Q_k^r(t) - \tau f_k(t) J_k, 0\right) \\ & + \min\left(Q_k^l(t), \tau R_k(t)\right), \end{aligned} \tag{4}$$

where $f_k(t)$ is the total computation power (in CPU cycles/s) assigned to sensor k during time slot t; J_k denotes the number of bits per CPU cycle, i.e., a parameter that depends on the specific application required by sensor k. In such a scenario, the sum of the local and remote computation queues represents a measure of the latency experienced by data before to be processed by the MEH, and is denoted by:

$$Q_k^{\mathrm{tot}}(t) = Q_k^l(t) + Q_k^r(t). \tag{5}$$

The goal is to find an optimal resource allocation strategy to minimize the long-term average energy consumption, while guaranteeing a bound on the out of service probability and a stable battery level. Mathematically, the problem is cast as:

$$\begin{aligned} & \min_{\mathbf{\Psi}(t)} \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathbb{E}\left\{e_k(t)\right\} \\ & \text{subject to} \\ & (a) \quad \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \Pr\left\{\mathrm{Q_k^{tot}(t) > Q_k^{max}}\right\} \le \epsilon_{\mathrm{k}}, \quad \forall \mathrm{k}; \\ & (b) \quad 0 \le e_k(t) \le \min\left(e_k^{\max}, B_k(t)\right), \quad \forall k, t; \\ & (c) \quad 0 \le f_k(t) \le f_{\max}, \qquad \forall k, t; \\ & (d) \quad \sum_{k=1}^{K} f_k(t) \le f_{\max}, \quad \forall t; \\ & (e) \quad 0 \le E_k^H(t) \le E_k^A(t), \quad \forall k, t; \end{aligned} \tag{6}$$

where $\mathbf{\Psi}(t) = \left[\{e_k(t)\}_k, \{f_k(t)\}_k, \{E_k^H(t)\}_k\right]$; ϵ_k is the out-of-service probability, while $e_k^{\max}$ and $f_{\max}$ are the energy budget of device k and the computational power of the MEH, respectively. The constraints have the following meaning: (a) ensures that the probability for the total queue in (5) to exceed a maximum value does not exceed the required out of service probability; (b) ensures that the transmit energy of each sensor is non negative and does not exceed the maximum energy budget, given by the minimum between the maximum transmit energy and the current battery level; (c) forces the computation resources allocated to each user to be non negative and not greater than the computation power of the MEH; (d) guarantees that the sum of the computation resources allocated to each user is at most equal to the computational power of the MEH; finally, constraint (e) sets the bounds on the maximum harvestable energy in each time slot.

III. Algorithm development

We handle problem (6) using stochastic optimization [23]. First of all, we define the virtual queue $Y_k(t)$ associated to the constraint (a) in (6). Then, we equivalently recast (a) as:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left\{\mathbf{1}\left\{\mathrm{Q_k^{tot}(t) > Q_k^{max}}\right\}\right\} \le \epsilon_{\mathrm{k}}, \tag{7}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. Since the indicator function in (7) can be rewritten as

$$\mathbf{1}\left\{Q_k^{\mathrm{tot}}(t) > Q_k^{\max}\right\} = u\left\{Q_k^{\mathrm{tot}}(t) - Q_k^{\max}\right\}, \tag{8}$$

where $u(\cdot)$ denotes the unitary step function. The virtual queue $Y_k(t)$ associated to the first constraint in (6) evolves as:

$$\begin{aligned} Y_k(t+1) = \max\big[0, & Y_k(t) \\ & + \mu\left(u\left\{Q_k^{\mathrm{tot}}(t+1) - Q_k^{\max}\right\} - \epsilon_k\right)\big], \end{aligned} \tag{9}$$

where μ is a step-size used to control the convergence of the algorithm. Note that the use of the step size does not change the problem, since it comes just from the scalar multiplication of both sides of constraint (7) by a factor μ. Now, using the approach of [24], [6], to stabilize the battery level around a desired value θ_k, we introduce the virtual queues $\widetilde{B}_k(t)$, which evolve as:

$$\widetilde{B}_k(t) = B_k(t) - \theta_k, \tag{10}$$

$k = 1, \ldots, K$, where $B_k(t)$ is given by (1). Having introduced the virtual queues $Y_k(t)$ and $\widetilde{B}_k(t)$ for each device k, we define the following Lyapunov function:

$$L(\mathbf{\Theta}(t)) = \frac{1}{2} \sum_{k=1}^{K} \left[Y_k^2(t) + \widetilde{B}_k^2(t)\right], \tag{11}$$

where $\mathbf{\Theta}(t) = \left[\mathbf{Y}(t), \widetilde{\mathbf{B}}(t)\right]$, and $\mathbf{Y}(t), \widetilde{\mathbf{B}}(t)$ are the vectors whose elements are the virtual queues of all sensors. Then, the Lyapunov drift is defined as [23]:

$$\Delta(\mathbf{\Theta}(t)) \triangleq \mathbb{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t)) | \mathbf{\Theta}(t)\}, \tag{12}$$

where the expectation is taken with respect to the channel and arrival rate (of data and energy) realizations, and it depends

on the control policy. The Lyapunov drift defined in (12) leads to the mean-rate stability of the virtual queues, i.e.,

$$\lim_{T\to\infty} \frac{\mathbb{E}[Y_k(T)]}{T} = 0, \quad k = 1, \ldots, K, \tag{13}$$

$$\lim_{T\to\infty} \frac{\mathbb{E}[\widetilde{B}_k(T)]}{T} = 0, \quad k = 1, \ldots, K, \tag{14}$$

but it can also lead to an unnecessary energy consumption. To balance mean-rate stability of virtual queues and long-term average energy consumption, we introduce the *drift-plus-penalty* function given by [23]:

$$\Delta_p(\mathbf{\Theta}(t)) = \Delta(\mathbf{\Theta}(t)) + V \cdot \mathbb{E}\left\{ \sum_{k=1}^{K} e_k(t) \big| \mathbf{\Theta}(t) \right\} \tag{15}$$

where V is a control parameter used to balance the aforementioned energy/queues tradeoff. The proposed algorithm proceeds by minimizing a proper upper bound of (15). In particular, it is possible to prove that an upper bound of (15) is given by [25]:

$$\begin{aligned}
&\Delta_p(\mathbf{\Theta}(t)) \leq C \\
&+ \mathbb{E}\Bigg\{ \sum_{k=1}^{K} \Bigg[\mu Y_k(t) \max\Big(0, \max(0, Q_k^l(t) - \tau R_k(t)) \\
&+ \max(0, Q_k^r(t) - \tau f_k(t) J_k) + \delta_k(t) \Big) \\
&+ \widetilde{B}_k(t) \left(E_k^{\mathrm{H}}(t) - e_k(t) \right) + V \cdot e_k(t) \Bigg] \Bigg| \mathbf{\Theta}(t) \Bigg\},
\end{aligned} \tag{16}$$

where C is a positive constant, and the term $\delta_k(t)$ is given by $\delta_k(t) = \tau R_{k,\max}(t) + A_k(t) - Q_k^{\max} + 1$, with

$$R_{k,\max}(t) = \beta_k(t) W \log_2 \left(1 + \frac{h_k(t) \min\left(e_k^{\max}, B_k(t)\right)}{\beta_k(t) W N_0 \tau} \right)$$

being an upper bound on the data rate of device k at time t. Thus, the algorithm proceeds by greedily minimizing instantaneous values of the upper bound in (16), thus obtaining the following dynamic control policy:

$$\begin{aligned}
\min_{\mathbf{\Psi}(t)} \quad & \sum_{k=1}^{K} \Bigg[\mu Y_k(t) \max\Big(0, \max(0, Q_k^l(t) - \tau R_k(t)) \\
& + \max(0, Q_k^r(t) - \tau f_k(t) J_k) + \delta_k(t) \Big) \\
& + \widetilde{B}_k(t)(E_k^H(t) - e_k(t)) + V \cdot e_k(t) \Bigg] \\
\text{subject to} \quad & \mathbf{\Psi}(t) \in \mathcal{Z}(t)
\end{aligned} \tag{17}$$

where $\mathcal{Z}(t)$ is the set of feasible actions according to the constraints $(b)-(d)$ of problem (6). It is easy to show that (17) is a convex optimization problem [21] when $V > \widetilde{B}_k(t)$ for all k, t[1], but it has a non-differentiable objective function. To tackle this issue, we first perform a simple change of variable,

[1]Exploiting the upper-bound in (20), a sufficient condition to guarantee $V > \widetilde{B}_k(t)$ is to set $V > E_{\max}^A$. This condition always holds in practice, since we are interested in large values of V, which lead to low average energy expenditures of the IoT devices.

Algorithm 1 : Dynamic Latency-constrained Computation Offloading Algorithm with Energy Harvesting

Data: K, N_{slot}, τ, J_k, V, P_t, W, $\{A_{k,\max}\}_k$, $\{Q_k^{\max}\}_k$, $\{\epsilon_k\}_k$, $f_{\max}$. Set μ, $\{Y_k(0)\}_k$, $\{\widetilde{B}_k(0)\}_k$;
For $t = 1 : N_{\text{slot}}$
(S.1): Observe the radio channels $\{h_k(t)\}_k$ and the harvested energy arrivals $\{E_k^A(t)\}_k$;
(S.2): Set the optimal harvested energies $\{E_k^H(t)\}_k$ to

$$E_k^H(t) = E_k^A(t) \cdot \mathbf{1}\{\widetilde{B}_k(t) \leq 0\}, \quad k = 1, \ldots, K; \tag{19}$$

(S.3): Solve problem $\mathcal{P}$ to find the optimal transmission energies $\{e_k(t)\}_k$ and computation resources $\{f_k(t)\}_k$;
(S.4): Update $Q_k^r(t)$ as in (4);
(S.5): Observe $A_k(t)$ and update $Q_k^l(t)$ as in (3);
(S.6): Update $B_k(t)$ as in (1);
(S.7): Update $Y_k^l(t)$ and $\widetilde{B}_k(t)$ as in (9) and (10) , respectively;
End

in order to use the data rate $R_k(t)$ as a variable of the problem. In particular, the transmit energy is given by:

$$e_k(t) = \frac{\beta_k(t) W \tau}{h_k(t)} \left[\exp\left(\frac{R_k(t) \log_e(2)}{\beta_k(t) W} \right) - 1 \right]. \tag{18}$$

Then, exploiting the convex epigraph form [21], it is possible to show that (17) can be equivalently recast as [25]:

$$\begin{aligned}
\min_{\mathbf{\Omega}(t)} \quad & \sum_{k=1}^{K} \Bigg[\mu Y_k(t) \Phi_k(t) + \widetilde{B}_k(t) E_k^H(t) \\
& + (V - \widetilde{B}_k(t)) \cdot \frac{\beta_k(t) W \tau}{h_k(t)} \exp\left(\frac{R_k(t) \log_e(2)}{\beta_k(t) W} \right) \Bigg]
\end{aligned}$$

subject to

$$\begin{aligned}
&(a) \quad 0 \leq R_k(t) \leq R_{k,\max}(t), \quad \forall k, t; \\
&(b) \quad 0 \leq f_k(t) \leq f_{\max}, \quad \forall k, t; \\
&(c) \quad \sum_{k=1}^{K} f_k(t) \leq f_{\max}, \quad \forall t; \\
&(d) \quad 0 \leq E_k^H(t) \leq E_k^A(t), \quad \forall k, t; \\
&(e) \quad \Phi_k(t) \geq \max(0, \delta_k(t)), \quad \forall k, t; \\
&(f) \quad \Phi_k(t) \geq Q_k^l(t) - \tau R_k(t) + \delta_k(t), \quad \forall k, t; \\
&(g) \quad \Phi_k(t) \geq Q_k^r(t) - \tau f_k(t) J_k + \delta_k(t), \quad \forall k, t; \\
&(h) \quad \Phi_k(t) \geq Q_k^l(t) - \tau R_k(t) + Q_k^r(t) \\
&\qquad\qquad - \tau f_k(t) J_k + \delta_k(t), \quad \forall k, t;
\end{aligned} \tag{$\mathcal{P}$}$$

where $\mathbf{\Omega}(t) = [\{R_k(t)\}_k, \{f_k(t)\}_k, \{\Phi_k(t)\}_k, \{E_k^H(t)\}_k]$. Now, problem $(\mathcal{P})$ is convex and differentiable, and can be solved using powerful numerical tools as interior point methods [21]. The MEH is the entity that runs the optimization algorithm for resource allocation, collecting all the needed information. In fact, almost all functions in $(\mathcal{P})$ are linear, except for (18), which is however convex. The overall dynamic procedure is described in Algorithm 1. In particular, Step (S.2) of Algorithm 1 is obtained by minimizing (17) with respect to $\{E_k^H(t)\}_k$, with the constraint $0 \leq E_k^H(t) \leq E_k^A(t)$.

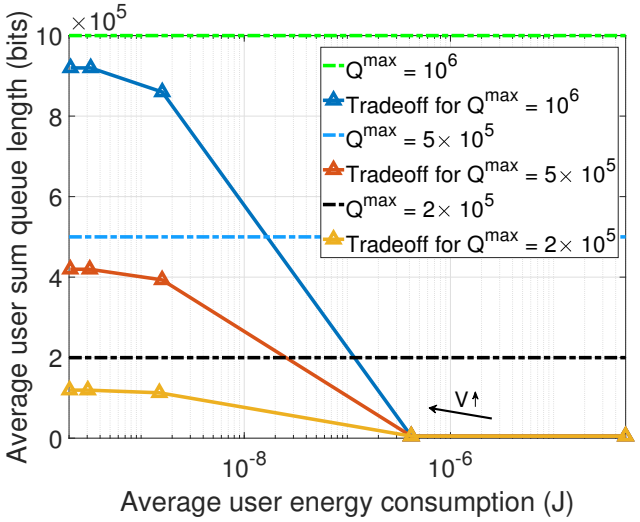

Fig. 2: Average user queue length vs average user energy consumption, for different $Q^{\max}$

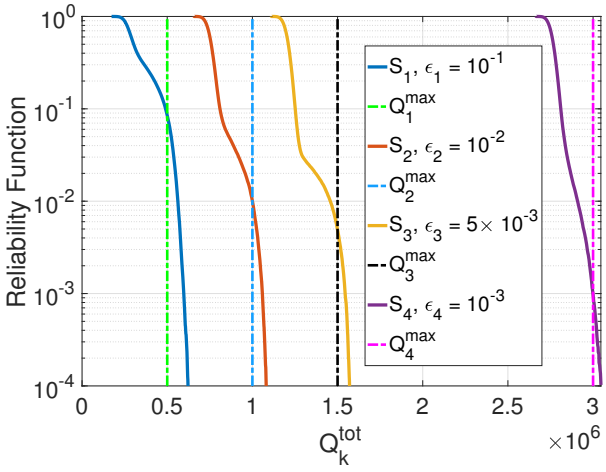

Fig. 3: Probability of exceeding the value on the abscissa.

Since (17) is linear with respect to $\{E_k^H(t)\}_k$, from (19), each node i collects the maximum harvestable energy $E_k^H(t)$ when $B_k(t) \leq \vartheta_k$; whereas, for $B_k(t) > \vartheta_k$, node k does not harvest any energy. Consequently, merging (1) with (19), we have:

$$B_k(t) \leq \vartheta_k + E_{\max}^A, \qquad \text{for all } k, t. \tag{20}$$

Step (S.3) of Algorithm 1 requires the solution of the optimization problem $\mathcal{P}$, which is convex with respect to the variables $\{R_k(t)\}_k$, $\{f_k(t)\}_k$, and $\{\Phi_k(t)\}_k$. Finally, the further steps update all the batteries, queues, and virtual queues according to (1), (3), (4), and (9).

IV. Numerical results

In this section, we show the performance of our algorithm through numerical results obtained by simulation in MATLAB environment, using the *fmincon* function from the optimization toolbox. Since problem $(\mathcal{P})$ is convex, *fmincon* converges to the global optimal solution very efficiently. In our simulations, we used a carrier frequency equal to $f_c = 3$ GHz, an available bandwidth of 10 MHz, a noise power spectral density of -174 dBm/Hz. The RAP is placed at the center of a square of side 100 m, whereas the positions of the K sensors are selected at random within the considered area. The channel value is obtained using the Friis path-loss, and considering a Rayleigh fading with zero mean and unit variance. We consider a single MEH associated with the RAP, having a computational power $f_{\max} = 3 \times 10^9$ CPU cycles/s. The conversion parameter J_k is set to 10^{-1} bits/CPU cycle for all k. The maximum transmit energy of each user is $e_k^{\max} = 5$ mJ, since the maximum transmit power is $P_k = 500$ mW and the transmission time interval is $\tau = 10$ ms. In Fig. 2, we show the tradeoff between the average user queue length and the average user energy consumption, for different requirements on $Q_k^{\max}$, which is chosen to be equal for all k. In this simulation, we considered a scenario with 30 sensors, with data arrivals uniformly distributed between 0 and $A_{k,\text{avg}} = 5 \times 10^3$ bits for all k. Also, the energy arrival rates at each sensor follow the uniform distribution between 0 and a maximum energy arrival $E_{\max}^A = 10^{-4}$ J. We consider three different QoS requirements given by $Q_k^{\max} = [1, 5, 10] \times 10^6$ bits, with an out of service probability bound equal to $\epsilon_k = 10^{-2}$. In the simulation, we considered 5000 time slots with $\tau = 10$ ms, and we averaged over 100 realizations of sensors' position. For the virtual queues $\{Y_k(t)\}_k$, we used a step-size $\mu = 50$. The energy/delay tradeoff is explored by letting the parameter V to vary along the curves reported in Fig. 2. In particular, the value of V increases going from right to left, as shown in the figure. Thus, as we can notice from Fig. 2, increasing the value of V, the average energy decreases. Also, the curves tend to the bound $Q_k^{\max}$ at large values of V (since a lower transmission energy determines a larger overall delay in terms of queue length), while at the same time not exceeding this value due to the constraint on the out of service probability. It is worth to remark that the method becomes pretty insensitive to V above a certain value, thus enabling flexibility in the choice of the tradeoff parameter. The only drawback of increasing V, and thus finding the minimum energy value, is the larger time needed to guarantee convergence of the algorithm.

As a further example, in Fig. 3, we show the behavior of the queues' reliability, defined as $1 - \text{CDF}(Q_k^{\text{tot}}(t))$, where $\text{CDF}(\cdot)$ is the cumulative distribution function. In particular, we illustrate the queues' reliability of 4 sensors, considering different values of $Q_k^{\max}$, ϵ_k, and θ_k. We run the simulation for 300000 slots and we averaged the results over the last 250000 slots, selecting $V = 10^{13}$. Each curve in Fig. 3 shows the probability that Q_k^{tot} is greater than the value on the abscissa, while the vertical lines represent the maximum requirements $Q_k^{\max}$, $k = 1, 2, 3, 4$, for the queue of each sensor. From Fig. 3, we notice that the proposed method enables all sensors to meet the required constraint on the out of service probability. Finally, in Fig. 4, we illustrate the temporal behavior of the battery level for the 4 sensors, using the same simulation setting considered to obtain Fig. 3. The curves are averaged

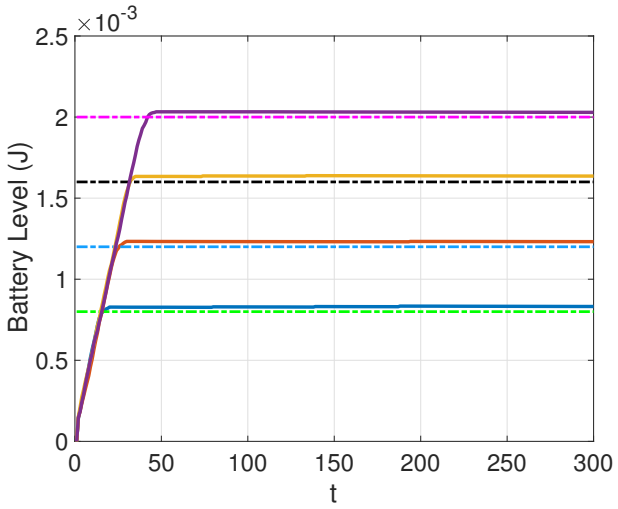

Fig. 4: Instantaneous battery level vs. iteration index

over 100 independent simulations. In Fig. 4, we can notice the effectiveness of the proposed algorithm in stabilizing the battery level of all sensors around prescribed values $\{\theta_k\}_k$, within a precision of at most $E^A_{\max} = 10^{-4}$ J, thus confirming the theoretical finding in (20).

V. Conclusions and Future Work

In this paper we studied the problem of dynamic resource allocation for computation offloading with MEC, with a constraint on the out of service probability, considering an IoT scenario with energy harvesting low-power devices. The problem is formulated as the minimization of the long-term average energy consumption subject to constraints on the out of service probability, while stabilizing the battery level of all devices around predefined thresholds. Stochastic optimization tools are used to solve the problem in a dynamic fashion, without assuming apriori knowledge of channel statistics and task arrivals. Moreover, the method defines a strategy to select the optimum harvested energy in each time slot, without any knowledge on the energy arrival statistics. Since the sum of the local and remote computation queues is considered as a metric to quantify the service delay, our algorithm naturally performs a joint optimization of radio and computation resources. The proposed strategy requires the solution of a convex optimization problem in each time slot, so that it can be handled via efficient and fast numerical tools. Several research directions are still open considering, e.g., distributed implementations of the proposed algorithm.

References

[1] Ericsson, "Ericsson mobility report," *Available Online*, Jun. 2018.

[2] 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.

[3] S. Barbarossa, E. Ceci, M. Merluzzi, and E. Calvanese-Strinati, "Enabling effective mobile edge computing using millimeterwave links," in *Proc. of IEEE Int. Conf. Commun. Work.*, May 2017, pp. 367–372.

[4] S. Barbarossa, E. Ceci, and M. Merluzzi, "Overbooking radio and computation resources in mmw-mobile edge computing to reduce vulnerability to channel intermittency," in *Proc. of 2017 Eur. Conf. Net. Commun. (EuCNC)*, Jun. 2017, pp. 1–5.

[5] W. Labidi, M. Sarkiss, and M. Kamoun, "Energy-optimal resource scheduling and computation offloading in small cell networks," in *Proc. of 2015 22nd International Conference on Telecommunications (ICT)*, Sydney, NSW, Australia 2015, pp. 313–318.

[6] 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.

[7] 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, Sept 2017.

[8] Y. Yang, S. Zhao, W. Zhang, Y. Chen, X. Luo, and J. Wang, "DEBTS: Delay energy balanced task scheduling in homogeneous fog networks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2094–2106, 2018.

[9] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE Journal on Sel. Areas in Comm.*, vol. 35, no. 11, pp. 2637–2646, Nov 2017.

[10] S. Sardellitti, M. Merluzzi, and S. Barbarossa, "Optimal association of mobile users to multi-access edge computing resources," in *Proc. of 2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2018, pp. 1–6.

[11] L. Chen-Feng, M. Bennis, and H.V. Poor, "Latency and reliability-aware task offloading and resource allocation for mobile edge computing," in *Proc. of 2017 IEEE Globecom Workshops (GC Wkshps)*, Singapore 2017, pp. 1–7.

[12] 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.

[13] 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*. 2018, pp. 419 – 444, Academic Press.

[14] 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.

[15] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, pp. 11255–11268, 2017.

[16] L. Chen-Feng, M. Bennis, M. Debbah, and H.V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *[Online]. Available: arXiv:1812.08076*.

[17] Shashank Priya and Daniel J Inman, *Energy harvesting technologies*, vol. 21, Springer, 2009.

[18] Meng-Lin Ku, Yan Chen, and KJ Ray Liu, "Data-driven stochastic models and policies for energy harvesting sensor communications," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 8, pp. 1505–1520, 2015.

[19] Vinod Sharma, Utpal Mukherji, Vinay Joseph, and Shrey Gupta, "Optimal energy management policies for energy harvesting sensor nodes," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, 2010.

[20] Ashutosh Nayyar, Tamer Başar, Demosthenis Teneketzis, and Venugopal V Veeravalli, "Optimal strategies for communication and remote estimation with an energy harvesting sensor," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2246–2260, 2013.

[21] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

[22] Longbo Huang and Michael J Neely, "Utility optimal scheduling in energy-harvesting networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 4, pp. 1117–1130, 2013.

[23] Michael J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*, Morgan and Claypool Publishers, 2010.

[24] M. J. Neely and L. Huang, "Dynamic product assembly and inventory control for maximum profit," in *49th IEEE Conference on Decision and Control (CDC)*, Dec.

[25] M. Merluzzi, P. Di Lorenzo, S. Barbarossa, and V. Frascolla, "Joint resource allocation for latency-constrained dynamic mobile edge computing," *Submitted to IEEE Transactions on Mobile Computing*, 2019.