Energy-Efficient Resource Allocation for Mobile Edge Computing-Based Augmented Reality Applications

Ali Al-Shuwaili and Osvaldo Simeone

Abstract—Mobile edge computing is a provisioning solution to enable Augmented Reality (AR) applications on mobile devices. AR mobile applications have inherent collaborative properties in terms of data collection in the uplink, computing at the edge, and data delivery in the downlink. In this letter, these features are leveraged to propose a novel resource allocation approach over both communication and computation resources. The approach, implemented via Successive Convex Approximation (SCA), is seen to yield considerable gains in mobile energy consumption as compared to conventional independent offloading across users.

Index Terms—Mobile edge computing, Augmented reality, Resource allocation, Shared offloading, Multicasting.

I. Introduction

Augmented Reality (AR) mobile applications are gaining increasing attention due to the their ability to combine computer-generated data with the physical reality. AR applications are computational-intensive and delay-sensitive, and their execution on mobile devices is generally prohibitive when satisfying users' expectations in terms of battery lifetime [1]–[3]. To address this problem, it has been proposed to leverage mobile edge computing [3]–[6]. Accordingly, users can offload the execution of the most time- and energy-consuming computations of AR applications to cloudlet servers via wireless access points.

The stringent delay requirements pose significant challenges to the offloading of AR application via mobile edge computing [3], [6]. A recent line of work has demonstrated that it is possible to significantly reduce mobile energy consumption under latency constraints by performing a joint optimization of the allocation of communication and computational resources [7], [8]. These investigations apply to generic applications run independently by different users. However, AR applications have the unique property that the applications of different users share part of the computational tasks and of the input and output data [3], [4]. In this paper, we propose to leverage this property to reduce communication and computation overhead via the joint optimization of communication and computational resources.

To illustrate the problem at hand, consider the class of AR applications that superimpose artificial images into the real world through the screen of a mobile device. The block diagram of such applications shown in Fig. 1 identifies the following components [3], [4]: (*i*) *Video source*, which obtain raw video frames from the mobile camera; (*ii*) *Tracker*, which

A. Al-Shuwaili and O. Simeone are with the Center for Wireless Information Processing (CWiP), Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: ana24@njit.edu, and osvaldo.simeone@njit.edu).
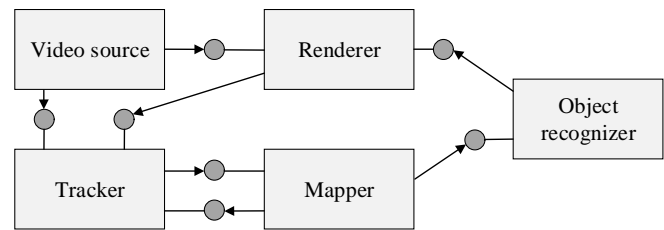

Fig. 1: Example of a component-based model of an AR application [3]. The application includes the Video source and Renderer components, which need to be executed locally on the mobile device, and the three main components of Mapper, Tracker and Object recognizer, which may be offloaded.

tracks the position of the user with respect to the environment; (*iii*) *Mapper*, which builds a model of the environment; (*iv*) *Object recognizer*, which identifies known objects in the environment; and (*v*) *Renderer*, which prepares the processed frames for display. The Video source and Renderer components must be executed locally at the mobile devices, while the most computationally intensive Tracker, Mapper and Object recognizer components can be offloaded. Moreover, if offloaded, Mapper and Object recognizer can collect inputs from all the users located in the same area, limiting the transmission of redundant information in the uplink across users. Also, the outcome of the Mapper and Object recognizer components can be multicast from the cloud to all co-located users in the downlink.

In this work, unlike prior papers [7], [8], we tackle the problem of minimizing the total mobile energy expenditure for offloading under latency constraints over communication and computation parameters by explicitly accounting for the discussed *collaborative* nature of AR applications. Section II introduces the system model. The resource allocation problem is formulated and tackled by means of a proposed Successive Convex Approximation (SCA) [9] solution in Section III. Numerical results are finally provided in Section IV.

II. System Model

We consider the mobile edge computing system illustrated in Fig. 2, in which K users in a set $\mathcal{K} = \{1, \ldots, K\}$ run a computation-intensive AR application on their single-antenna mobile devices with the aid of a cloudlet server. The server is attached to a single-antenna Base Station (BS), which serves all the users in the cell using Time Division Duplex (TDD) over a frequency-flat fading channel. Following the discussion in Sec. I, we assume that the offloaded application has shared
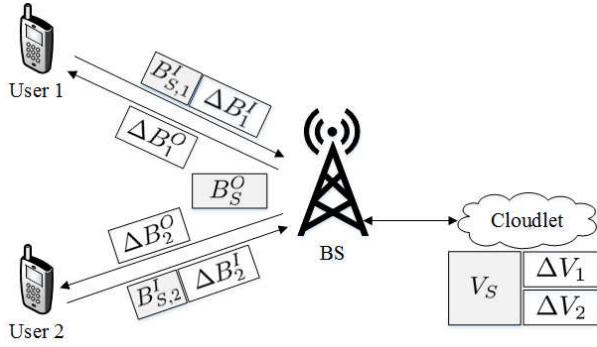
Fig. 2: Offloading of an AR application to a cloudlet attached to the BS. Shared data and computations are shaded.

Fig. 3: Data frame structure for a system with $K = 3$ users. The preamble containing training sequences is not shown.

inputs, outputs and computational tasks, which pertain to the Tracker, Mapper and Object recognizer components.

To elaborate, let us first review the more conventional set-up, studied in, e.g., [7], [8], in which users perform the offloading of *separate* and independent applications. In this case, offloading for each user k would require: (*i*) *Uplink:* transmitting a number B_k^I input bits from each user k to the cloudlet in the uplink; (*ii*) *Cloudlet processing:* processing the input by executing V_k CPU cycles at the cloudlet; (*iii*) *Downlink:* transmitting B_k^O bits from the cloudlet to each user k in the downlink. In contrast, as discussed in Sec. I, the collaborative nature of the Tracker, Mapper and Object recognizer components (recall Fig. 1) can be leveraged to reduce mobile energy consumption and offloading latency, as detailed next. We note that the non-collaborative components can potentially also be carried out locally if this reduces energy consumption (see, e.g., [7][8]). We leave the study of the optimization of this aspect to future work.

1) Shared uplink transmission: A given subset of $B_S^I \leq \min_k\{B_k^I\}$ input bits is shared among the users in the sense that it can be sent by *any* of the users to the cloudlet. For example, the input bits to the offloaded Object recognizer component in Fig. 1 can be sent by any of the users in the same area. As a result, each user k transmits a fraction of $B_{S,k}^I$ bits of the B_S^I shared bits, which can be optimized under the constraint $\sum_k B_{S,k}^I = B_S^I$, as well as $\Delta B_k^I = B_k^I - B_{S,k}^I$ bits that need to be uploaded exclusively by user k.

2) Shared cloudlet processing: Part of the computational effort of the cloudlet is spent producing output bits of interest to all users. An example is the computational task of updating the model of the environment carried out by the mobiles. Therefore, we assume that $V_S \leq \min_k\{V_k\}$ CPU cycles are shared, whereas $\Delta V_k = V_k - V_S$ CPU cycles are to be executed for each user k.

3) Multicast downlink transmission: Some of the output bits need to be delivered to all users. For example, a co-located group of users may need the output bits from the Mapper component for a map update. To model this, we assume that a subset of $B_S^O \leq \min_k\{B_k^O\}$ output bits can be transmitted in multicast mode to all users in the cell, while $\Delta B_k^O = B_k^O - B_S^O$ bits need to be transmitted to each user in a unicast manner.

The frame structure is detailed in Fig. 3. Not shown are the training sequences sent by the users prior to the start of the data transmission frame, which enable the BS to estimate the uplink
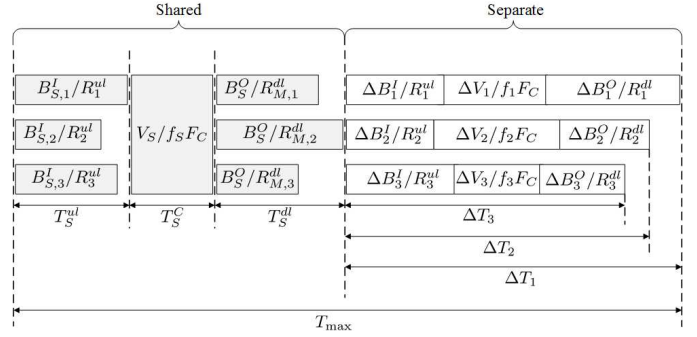
Channel State Information (CSI), and hence also the downlink CSI due to reciprocity. The CSI is assumed to remain constant for the frame duration. As seen in Fig. 3, in the data frame, the shared communication and computation tasks are carried out first, followed by the conventional separate offloading tasks, as discussed next.

1) Uplink transmission: The achievable rate, in bits/s, for transmitting the input bits of user k in the uplink is given by

$$R_k^{ul}(P_k^{ul}) = \frac{W^{ul}}{K} \log_2\left(1 + \frac{\gamma_k P_k^{ul}}{N_0 W^{ul}/K}\right), \quad (1)$$

where P_k^{ul} is the transmit power of the mobile device of user k; the uplink bandwidth W^{ul} is equally divided among the K users, e.g., using OFDMA; γ_k is the uplink and downlink channel power gains of user k; and N_0 is the noise power spectral density at the receiver. Referring to Fig. 3 for an illustration, the time, in seconds, necessary to complete the shared uplink transmissions is defined as $T_S^{ul} = \max_k B_{S,k}^I / R_k^{ul}(P_k^{ul})$, whereas the time needed for user k to transmit the separate ΔB_k^I bits is $\Delta B_k^I / R_k^{ul}(P_k^{ul})$. The corresponding mobile energy consumption due to uplink transmission is

$$E_k^{ul}(P_k^{ul}, B_{S,k}) = \left(\frac{P_k^{ul}}{R_k^{ul}(P_k^{ul})} + l_k^{ul}\right)\left(B_{S,k}^I + \Delta B_k^I\right), \quad (2)$$

where l_k^{ul} is a parameter that indicates the amount of energy spent by the mobile device to extract each bit of offloaded data from the video source. In (2) and in subsequent equations, we make explicit the dependence on variables to be optimized.

2) Cloudlet processing: Let F_C be the capacity of the cloudlet server in terms of number of CPU cycles per second. Also, let $f_k \geq 0$ and $f_S \geq 0$ be the fractions, to be optimized, of the processing power F_C assigned to run the ΔV_k CPU cycles exclusively for user k and the V_S shared CPU cycles, respectively, so that $\sum_{k \in \mathcal{K}} f_k \leq 1$ and $f_S \leq 1$. As shown in Fig. 3, the execution time for the shared CPU cycles is $T_S^C = V_S/(f_S F_C)$ and the time needed to execute ΔV_k CPU cycles of interest to user k remotely is $\Delta V_k/(f_k F_C)$.

3) Downlink transmission: The common output bits B_S^O are multicast to all users. Let P_M^{dl} be the transmit power for multicasting, which is subject to the optimization. The resulting achievable downlink rate for user k is given by

$$R_{M,k}^{dl}(P_M^{dl}) = W^{dl} \log_2\left(1 + \frac{\gamma_k P_M^{dl}}{N_0 W^{dl}}\right), \quad (3)$$

with $R_M^{dl}(P_M^{dl}) = \min_k R_{M,k}^{dl}(P_M^{dl})$, with W^{dl} being the downlink bandwidth. The downlink transmission time to multicast

B_S^O bits can hence be computed as $T_S^{dl} = B_S^O / R_M^{dl}(P_M^{dl})$ (see Fig. 3). The ΔB_k^O output bits intended exclusively for each user k are sent in a unicast manner in downlink using an equal bandwidth allocation, with rate

$$R_k^{dl}(P_k^{dl}) = \frac{W^{dl}}{K} \log_2 \left(1 + \frac{\gamma_k P_k^{dl}}{N_0 W^{dl}/K} \right), \qquad (4)$$

where P_k^{dl} is the BS transmit power allocated to serve user k. The overall downlink mobile energy consumption for user k is

$$E_k^{dl}(P_k^{dl}, P_M^{dl}) = \left(\frac{\Delta B_k^O}{R_k^{dl}(P_k^{dl})} + \frac{B_S^O}{R_{M,k}^{dl}(P_M^{dl})} \right) l_k^{dl}, \quad (5)$$

where l_k^{dl} is a parameter that captures the mobile receiving energy expenditure per second in the downlink. We note that we leave the optimization of the bandwidth allocation across users for uplink and downlink to future work.

III. ENERGY-EFFICIENT RESOURCE ALLOCATION

In this section, we tackle the minimization of the mobile sum-energy required for offloading across all users under latency and power constraints. Stated in mathematical terms, we consider the following problem:

$$\underset{\mathbf{z}}{\text{minimize}} \quad \sum_{k \in \mathcal{K}} E_k^{ul}(P_k^{ul}, B_{S,k}) + E_k^{dl}(P_k^{dl}, P_M^{dl})$$

s.t.
C.1 $\frac{\Delta B_k^I}{R_k^{ul}(P_k^{ul})} + \frac{\Delta V_k}{f_k F_C} + \frac{V_S}{f_S F_C}$
$+ \frac{\Delta B_k^O}{R_k^{dl}(P_k^{dl})} \leq T_{\max} - T_S^{ul} - T_S^{dl}, \forall k \in \mathcal{K},$

C.2 $\frac{B_{S,k}^I}{R_k^{ul}(P_k^{ul})} \leq T_S^{ul}, \forall k \in \mathcal{K},$

C.3 $\frac{B_S^O}{R_{M,k}^{dl}(P_M^{dl})} \leq T_S^{dl}, \forall k \in \mathcal{K},$

C.4 $\sum_{k \in \mathcal{K}} f_k \leq 1; 0 \leq f_S \leq 1; f_k \geq 0, \forall k \in \mathcal{K},$

C.5 $\sum_{k \in \mathcal{K}} B_{S,k} = B_S^I,$

C.6 $\sum_{k \in \mathcal{K}} P_k^{dl} \leq P_{\max}^{dl}; P_M^{dl} \leq P_{\max}^{dl}; P_k^{ul} \leq P_{\max}^{ul}, \forall k \in \mathcal{K}.$

(P.1)

The optimization variables are collected in vector $\mathbf{z} \triangleq \left(\mathbf{P}^{ul}, \mathbf{B}_S^I, \mathbf{f}, \mathbf{P}^{dl}, P_M^{dl}, T_S^{ul}, T_S^{dl} \right)$, where $\mathbf{P}^{ul} \triangleq (P_k^{ul})_{k \in \mathcal{K}}$, $\mathbf{B}_S^I \triangleq (B_{S,k}^I)_{k \in \mathcal{K}}$, $\mathbf{f} \triangleq ((f_k)_{k \in \mathcal{K}}, f_S)$, $\mathbf{P}^{dl} \triangleq (P_k^{dl})_{k \in \mathcal{K}}$, and we defined $\mathcal{Z}$ as the feasible set of problem (P.1). As illustrated in Fig. 3, constraints C.1-C.3 enforce that the execution time of the offloaded application to be less than or equal to the maximum latency of $T_{\max}$ seconds. Constraints C.4-C.5 impose the conservation of computational resources and shared input bits, and C.6 enforces transmit power constraints at BS and users.

Problem (P.1) is not convex because of the non-convexity of the energy function $E_k^{ul}(P_k^{ul}, B_{S,k}^I)$ and of the latency constraints C.2, which we can rewrite as $g_k(P_k^{ul}, B_{S,k}^I) \leq T_S^{ul}, \forall k \in \mathcal{K}$. We address this issue by developing an SCA solution following [9]. Theorem 2 in [9] shows that the SCA algorithm converges to a stationary point of the non-convex problem (P.1). Furthermore, such convergence requires a number of iterations proportional to $1/\epsilon$, where ϵ measures the desired accuracy in terms of the stationarity metric $\|F(\mathbf{z})\|_2^2$ defined in [10, Eq. (6)].

In order to apply the SCA method, we need to derive convex approximants for the functions $E_k^{ul}(P_k^{ul}, B_{S,k})$ and $g_k(P_k^{ul}, B_{S,k}^I)$ that satisfy the conditions specified in [9, Sec.

II]. Using such approximants, we obtain the SCA scheme detailed in Algorithm 1. In the algorithm, at each iteration v, the unique solution $\hat{\mathbf{z}}(\mathbf{z}(v)) \triangleq \left(\hat{\mathbf{P}}^{ul}, \hat{\mathbf{B}}_S^I, \hat{\mathbf{f}}, \hat{\mathbf{P}}^{dl}, \hat{P}_M^{dl}, \hat{T}_S^{ul}, \hat{T}_S^{dl} \right)$ of the following strongly convex problem

$$\hat{\mathbf{z}}(\mathbf{z}(v)) \triangleq \underset{\mathbf{z}}{\arg\min} \sum_{k \in \mathcal{K}} \tilde{E}(\mathbf{z}_k; \mathbf{z}_k(v))$$

s.t. **C.2** $\tilde{g}_k \left(P_k^{ul}, B_{S,k}; P_k^{ul}(v), B_{S,k}^I(v) \right) \leq T_S^{ul}, \forall k \in \mathcal{K},$

C.1, C.3 − C.6 of (P.1), (P.2)

is obtained, where we have defined $\mathbf{z}_k \triangleq \left(P_k^{ul}, B_{S,k}, f_k, f_S, P_k^{dl}, P_M^{dl}, T_S^{ul}, T_S^{dl} \right)$ as well as $\tilde{E}_k(\mathbf{z}_k; \mathbf{z}_k(v)) \triangleq \tilde{E}_k^{ul}(\mathbf{z}_k; \mathbf{z}_k(v)) + E_k^{dl}(P_k^{dl}, P_M^{dl})$. The approximants functions $\tilde{E}_k^{ul}(.; .)$ and $\tilde{g}_k(.; .)$ are discussed next. The approximant $\tilde{E}_k^{ul}(\mathbf{z}_k; \mathbf{z}_k(v))$ around the current feasible iterate $\mathbf{z}_k(v)$ can be obtained following [9, Sec. III, Example #8] as

$$\tilde{E}_k^{ul}(\mathbf{z}_k; \mathbf{z}_k(v)) = \frac{P_k^{ul}(v) \left(B_{S,k}^I(v) + \Delta B_k^I \right)}{R_k^{ul}(P_k^{ul})}$$

$$+ \frac{P_k^{ul}(v) \left(B_{S,k}^I + \Delta B_k^I \right)}{R_k^{ul}(P_k^{ul}(v))} + \frac{P_k^{ul} \left(B_{S,k}^I(v) + \Delta B_k^I \right)}{R_k^{ul}(P_k^{ul}(v))} \qquad (6)$$

$$+ \bar{E}_k^{ul}(\mathbf{z}_k; \mathbf{z}_k(v)) + l_k^{ul} \left(B_{S,k}^I + \Delta B_k^I \right),$$

where $\bar{E}_k^{ul}(\mathbf{z}_k; \mathbf{z}_k(v)) \triangleq (\mathbf{z}_k - \mathbf{z}_k(v))^T \Psi (\mathbf{z}_k - \mathbf{z}_k(v))$, with Ψ being a diagonal matrix with non-negative elements $\tau_{P^{ul}}, \tau_{B_S^I}, \tau_f, \tau_{f_S}, \tau_{P^{dl}}, \tau_{P_M^{dl}}, \tau_{T_S^{ul}}$ and $\tau_{T_S^{dl}}$. For the second approximant, in light of the relation $g(x_1, x_2) = x_1 x_2 = 1/2(x_1 + x_2)^2 - 1/2(x_1^2 + x_2^2)$, a convex upper bound is obtained as requested by SCA by linearizing the concave part of $g_k(P_k^{ul}, B_{S,k}^I)$ [9, Sec. III, Example #4], which results in

$$\tilde{g}_k \left(P_k^{ul}, B_{S,k}^I; P_k^{ul}(v), B_{S,k}^I(v) \right) = \frac{1}{2} \left(\left(B_{S,k}^I + \frac{1}{R_k^{ul}(P_k^{ul})} \right)^2 \right.$$

$$- \left(B_{S,k}^I(v) \right)^2 - \left(\frac{1}{R_k^{ul}(P_k^{ul}(v)^2)} \right) \right) - \left(B_{S,k}^I(v) \left(B_{S,k}^I - B_{S,k}^I(v) \right) \right.$$

$$\left. - \frac{R_k^{ul}(P_k^{ul}(v))}{\left(1 + \frac{\gamma_k P_k^{ul}(v)}{N_0 W^{ul}/K} \right) R_k^{ul}(P_k^{ul}(v))^4} \left(\frac{1}{R_k^{ul}(P_k^{ul})} - \frac{1}{R_k^{ul}(P_k^{ul}(v))} \right) \right).$$

(7)

The convexity of (7) is established by noting that the second term in the right-hand side is the reciprocal of the rate function (concave and positive) and the fourth power (convex and non-decreasing) of a convex function is convex [11].

Algorithm 1 SCA Algorithm

1: *Initialization*: $\mathbf{z}(0) \in \mathcal{Z}$; $\alpha = 10^{-5}$; $\epsilon = 10^{-5}$; $v = 0$; $\tau_{P^{ul}}, \tau_{B_S^I}, \tau_f, \tau_{f_S}, \tau_{P^{dl}}, \tau_{P_M^{dl}}, \tau_{T_S^{ul}}, \tau_{T_S^{dl}} > 0$.
2: Compute $\hat{\mathbf{z}}(\mathbf{z}(v))$ from (P.2).
3: Set $\mathbf{z}(v+1) = \mathbf{z}(v) + \delta(v)(\hat{\mathbf{z}}(\mathbf{z}(v)) - \mathbf{z}(v))$, with $\delta(v) = \delta(v-1)(1 - \alpha\delta(v-1))$.
4: If $\|F(\mathbf{z}(v))\|_2^2 \leq \epsilon$, stop.
5: Otherwise, set $v \leftarrow v + 1$, and return to step 2.

IV. NUMERICAL RESULTS

In this section, we provide numerical examples with the aim of illustrating the advantages that can be accrued by leveraging the collaborative nature of AR applications for mobile edge computing. We consider a scenario where eight users are

randomly deployed in a small cell. The radio channels are Rayleigh fading and the path loss coefficient is obtained based on the small-cell model in [12] for a carrier frequency of 2 GHz. The users' distances to the BS are randomly uniformly selected between 100 and 1000 meters and the results are averaged over multiple independent drops of users' location and of the fading channels. The noise power spectral density is set to $N_0 = -147$ dBm/Hz. The uplink and downlink bandwidth is $W^{ul} = W^{dl} = 10$ MHz. The uplink and downlink power budgets are constrained to the values $P_{\max}^{ul} = 50$ and $P_{\max}^{dl} = 60$ dBm, respectively. The cloudlet server processing capacity is $F_C = 10^{10}$ CPU cycles/s [3]. We also set $l_k^{ul} = 1.78 \times 10^{-6}$ J/bit [13], $l_k^{dl} = 0.625$ J/s [14] and $\epsilon = 10^{-5}$.

The size of the input data generally depends on the number and size of the features of the video sources obtained by the mobiles that are to be processed at the cloudlet. Here we select $B_k^I = 1$ Mbits, which may correspond to the transmission of 1024×768 images [5]. A fraction of the input bits $B_S^I = \eta B_k^I$ bits can be transmitted cooperatively by all users for some parameter $0 \leq \eta \leq 1$. The required CPU cycles of the offloaded components is set to $V_k = 2640 \times B_k^I$ CPU cycles, representing a computational intensive task [15]. The shared CPU cycles are assumed to be $V_S = \eta V_k$ for the same sharing factor η. The output bits are assumed to equal the amount of input bits $B_k^O = B_k^I = 1$ Mbits with shared fraction $B_S^O = \eta B_k^O$. Practical latency constraints for AR applications are of the order of 0.01 s [4], [5]. Throughout our experiments, we found that the accuracy $\epsilon = 10^{-5}$ was obtained within no more than 25 iterations.

For reference, we compare the performance of the proposed scheme, in which uplink and downlink transmissions and cloudlet computations are shared as described, with the following offloading solutions: *(i) Shared Cloudlet Processing and Downlink Transmission*: CPU cycles and output data are shared as described in Sec. II, while the input bits B_k^I are transmitted by each user individually, i.e., we set $B_{S,k}^I = 0$ and $\Delta B_k^I = B_k^I$ for all $k \in \mathcal{K}$; and *(ii) Shared Uplink Transmission*: Only the input bits are shared as discussed, while no sharing of computation and downlink transmission takes place, i.e., $\Delta V_k = V_k$ and $\Delta B_k^O = B_k^O$ for all $k \in \mathcal{K}$. We also include for reference the result obtained by solving problem (P.1) using the global optimization BARON software running on NEOS server with a global optimality tolerance of 10^{-6} [16]. As shown in Fig. 4, for $T_{\max} = 0.05$ s and $\eta = 0.3$, the *Shared Cloudlet Processing and Downlink Transmission* scheme achieves energy saving about 37% compared to separate offloading (which sets $\Delta B_k^I = B_k^I$, $\Delta V_k = V_k$ and $\Delta B_k^O = B_k^O$ for all $k \in \mathcal{K}$). This gain can be attributed to the increased time available for uplink transmission due to the shorter execution and downlink transmission periods, which reduces the associated offloading energy. Under the same conditions, the energy saving of around 50% with respect to separate offloading brought by *Shared Uplink Transmission* is due to the ability of the system to adjust the fractions of shared data transmitted by each user in the uplink based on the current channel conditions. These two gains combine to yield the energy saving of the proposed shared data offloading scheme with respect to the conventional separate offloading of around 63%. Both separate and shared offloading schemes have similar energy performance for the
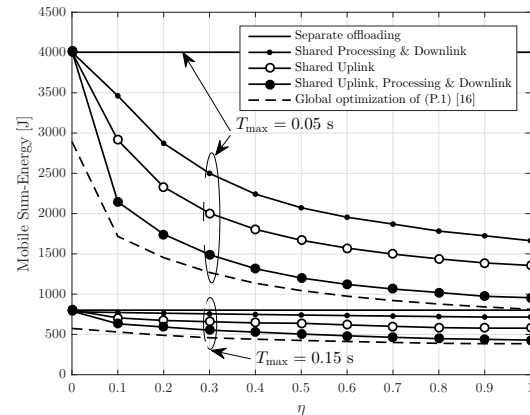

Fig. 4: Average mobile sum-energy consumption versus the fraction η of shared data in uplink and downlink and of shared CPU cycles executed at the cloudlet.

relaxed delay requirement of $T_{\max} = 0.15$ s, which can be met with minimal mobile energy expenditure even without sharing communication and computation resources. The figure also shows that SCA yields a solution that is close to the global optimum, for this example.

References

[1] D. Van Krevelen and R. Poelman, "A survey of augmented reality technologies, applications and limitations," *International J. of Virtual Reality*, vol. 9, no. 2, pp. 1–20, Jan. 2010.

[2] F. Liu *et al.*, "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 14–22, Jun. 2013.

[3] T. Verbelen *et al.*, "Leveraging cloudlets for immersive collaborative applications," *IEEE Perv. Comput.*, vol. 12, no. 4, pp. 30–38, Dec. 2013.

[4] S. Bohez *et al.*, "Mobile, collaborative augmented reality using cloudlets," in *Proc. Mobilware*, Bologna, Italy, Nov. 2013.

[5] J. M. Chung *et al.*, "Adaptive cloud offloading of augmented reality applications on smart devices for minimum energy consumption," *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 8, pp. 3090–3102, Aug. 2015.

[6] M. Satyanarayanan *et al.*, "The case for VM-based cloudlets in mobile computing," *IEEE Perv. Comput.*, vol. 8, no. 4, pp. 14–23, Dec. 2009.

[7] A. Al-Shuwaili *et al.*, "Joint uplink/downlink optimization for backhaul-limited mobile cloud computing with user scheduling," *ArXiv preprints:1607.06521*, Jul. 2016.

[8] S. Sardellitti *et al.*, "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.

[9] G. Scutari *et al.*, "Parallel and distributed methods for constrained nonconvex optimization-part I: Theory," *IEEE Trans. Signal Process.*, to appear, 2016. On-line: arXiv:1410.4754.

[10] L. Cannelli *et al.*, "Asynchronous parallel algorithms for nonconvex big-data optimization. part II: Complexity and numerical results," *arXiv preprint:1701.04900*, Jan. 2017.

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

[12] 3GPP TR 36.814, "Technical specification group radio access network; Further advancements for E-UTRA, physical layer aspects," Mar. 2010.

[13] F. Renna *et al.*, "Query processing for the internet-of-things: Coupling of device energy consumption and cloud infrastructure billing," in *Proc. IEEE IoTDI*, Berlin, Germany, pp. 83-94, Apr. 2016.

[14] N. Balasubramanian *et al.*, "Energy consumption in mobile phones: A measurement study and implications for network applications," in *Proc. ACM SIGCOMM*, Chicago, IL, pp. 280–293, Nov. 2009.

[15] A. Miettinen *et al.*, "Energy efficiency of mobile clients in cloud computing," in *Proc. USENIX*, Boston, MA, USA, pp. 4-11, Jun. 2010.

[16] N. Sahinidis, "Baron for solving nonconvex optimization problems to global optimality," Mar. 2017. On-line: https://neos-server.org/neos/solvers/go:BARON/AMPL.html.