A Distributed Algorithm for Multi-Stage Computation Offloading

Tobias Mahn, Dennis Becker, Hussein Al-Shatri, Anja Klein
Communications Engineering Lab, TU Darmstadt, Germany, {t.mahn, h.shatri, a.klein}@nt.tu-darmstadt.de

***Abstract*—A scenario consisting of several mobile users, an access point (AP) and a cloud server in a multi-stage hierarchy is considered. Each user has a computation task which can be computed locally or offloaded to the AP or to the cloud server. Considering the shared access channel between users and the AP, the shared computation resources at the AP and the shared backhaul link connection from the AP to the cloud, an energy minimization computation offloading problem with a time constraint is tackled. The time constraint guarantees that the offloading time will not exceed the local computation time. In this paper, we propose a distributed game theoretic algorithm which decomposes the offloading problem into the subproblems of resource allocation and offloading decisions. The algorithm works iteratively between the two subproblems as follows: The AP receives offloading decisions from the users and accordingly optimizes the fractions of the bandwidth on the access channel, the fractions of the backhaul link rate and the fractions of the computation resource at the AP for all offloading users. Based on the assigned resources, each user autonomously decides between local computation or offloading to the AP or to the cloud server and reports its decision to the AP. Our proposed algorithm is shown to require only limited signaling between users and AP and converges in significantly few iterations. Furthermore, the results show that our algorithm performs close to the optimal policy.**

I. Introduction

Computation offloading has seen a great interest in research as shown in a recent survey [1]. One of the reasons is the rapid growth of mobile devices and internet of things gadgets in our daily life that perform demanding tasks like high resolution photo editing, video processing for augmented reality or natural language recognition [1]. While the recent generations of mobile devices, e.g. smartphones, tablets and laptops, usually offer high computational resources, their battery is the limiting factor for such demanding purposes [2]. As studies have shown, mobile phone users have always been very cautious about their battery level [3], [4]. Hence, offloading tasks to a cloud server might be the solution for saving energy on the mobile devices and yet enabling a rich experience for the mobile users.

Besides computation offloading to a cloud server, mobile edge computing (MEC) is a new approach to the topic [5], [6]. Instead of a cloud server, a cloudlet with less computational capability than the cloud server is attached to the access point (AP). The cloudlet is closer to the mobile unit (MU) than the cloud server. This reduces the latency and enables more real time applications than offloading to a cloud server. Since access channel and remote computation resources are limited and shared among the MUs, each MU has to consider, where to offload its task and whether offloading is beneficial at all. The advantage of computation offloading over local computation of a task can be a shorter computation time, a reduction of energy consumption or a combination of both.

The authors of [7] and [6] investigate computation time minimization problems under the constraint of maximum allowed transmit power in MEC scenarios using optimization methods. The authors of [6] propose and compare the system offloading performance of different access schemes of the wireless radio channel for a single task per MU. In [7], the focus lies on device internal task scheduling of multiple tasks at a single MU. Transmission time of the task is modeled by the Shannon channel capacity and the task size. Offloading of the task is only possible if the transmission time is smaller than the time slot duration.

In [8] and [9], the authors formulate central algorithms for energy minimization with a maximum computation time constraint and propose optimization approaches. Both consider MEC computing scenarios and arbitarily splittable tasks. Multiple different optimization approaches of energy minimization problems are investigated in [8] for only a single MU. In [9], multiple MUs try to minimize their transmit power, while sharing the radio access and cloudlet computation resources. Simultaneous offloading of multiple MUs causes interference on the radio access channel and hence, the transmit power has to be optimized to ensure successful reception at the AP.

The authors of [10] and [11] combine energy and time minimization through a weighting variable as a joint objective. The energy reduction by offloading of a MU is compared to the maximum computation time required. The results computed by this formulation have proven to be strongly dependent on the choice of the weighting variable. Compared to the central appraoches in [6]- [9], distributed approaches based on game theory are proposed in [10] and [11]. These games can be executed in a distributed manner at each MU instead of a central computation at the AP. Nevertheless, the convergence to a Nash equilibrium requires full knowledge about all parameters in the system at every MU. While [10] considers only offloading to a cloud server, [11] investigates a multi-stage scenario. This multi-stage scenario combines MEC with an additional cloud server and a MU has to choose the most beneficial location for offloading. In [11], the radio access link to the AP is proposed as a simple transmission rate model.

This paper investigates a multi-stage offloading hierarchy with a combination of a cloudlet for MEC and a cloud server. Each MU has a non-splittable task. By offloading, the MU tries to minimize the energy spent for the computing or transmission of its task. Since the hardware of a MU requires a static power while transmitting or waiting for the result of the remote computation, this power is included in our energy model. The offloading time of a MU is defined as the sum of the time for transmission and the time for the computation at the AP or at the cloud server. A combination of energy and computation time in the objective function is avoided by a constrained version of an energy minimization problem. To ensure that offloading is always beneficial in terms of time delay, the offloading time is constrained to be less than the local computation time.

A novel distributed algorithm is proposed, where each MU takes an autonomous offloading decision on its task. The offloading decisions and the resource allocation are separated and executed iteratively. The resource allocation of the shared radio access channel, the shared backhaul link to the cloud and the shared computation resources at the AP is performed at the AP based on offloading decisions of the MUs. After receiving feedback from the AP about its fraction of the shared resources, every MU decides whether offloading or local computation is more beneficial. Furthermore, this decision can be performed without knowledge about the offloading decisions of the other MUs as a best response to the allocated resources. Our proposed algorithm requires significantly low signaling between AP and MUs and no direct signaling among the MUs.

The paper is structured as follows: Section II introduces the scenario. In Section III, we formulate the optimization problem to our model and show how it can be transformed into the hierarchical game formulation. Our theoretical results are tested numerically in Section IV.

II. System Model

A. Scenario

In this paper, we consider a computation offloading scenario with K MUs connected to a single wireless AP with an attached cloudlet. Furthermore, a cloud server with high computational capability is available through a limited capacity backhaul link to the AP. This scenario is shown in Figure 1 and could be a part of a larger network. While the MUs receive their energy from a limited battery, the AP, cloudlet and cloud server have access to energy through the power grid.

Every MU with index $k = 1, \dots, K$ has one task to be computed. This task is non-splittable and therefore, has to be either computed locally or can be offloaded to the cloudlet or the cloud server. The size l_k of the task of MU k is measured in bits. For the computation of the task, we introduce a complexity factor β_k which determines the number of CPU cycles needed for the computation of one bit of the task of MU k. This factor β_k is dependent on the type of task, e.g. the processing of a video stream can have a higher β_k than the processing of a text file. Exemplary values for β_k can be found in [2]. For a high task complexity factor β_k, even the computation of a small task might result in a huge energy consumption and processing time at the MU. So, offloading the task can be beneficial for a MU trying to save energy.

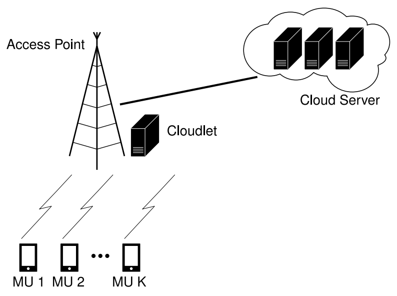

Fig. 1. Multi-stage computation offloading scenario with K MUs, a single AP with attached cloudlet and a cloud server

B. Local Computation

First, we are considering the case of MU k computing its task locally. Furthermore, every MU has a processing frequency $f_{\mathrm{MU},k}$, which is measured in CPU cycles per second. Using the assumptions, the local processing time is

$$T_{\mathrm{MU},k} = \frac{\beta_k l_k}{f_{\mathrm{MU},k}}. \quad (1)$$

$T_{\mathrm{MU},k}$ with the power $p_{\mathrm{calc},k}$ required for local processing at MU k in Watt leads to energy consumed for a local computation of MU k's task, which is given by

$$E_{\mathrm{MU},k} = p_{\mathrm{calc},k} \cdot \frac{\beta_k l_k}{f_{\mathrm{MU},k}}. \quad (2)$$

C. Computation at the AP

Now, offloading of the task of MU k to the AP with attached cloudlet is introduced. The cloudlet is assumed to be more powerful in terms of computation capability than the MUs, i.e. $f_{\mathrm{MU},k} \leq f_{\mathrm{AP}}$ with f_{AP} being the overall processing frequency of the cloudlet in cycles per second. If more than one MU decides for offloading its task to the AP, the limited computation resources are split and denoted by $f_{\mathrm{AP},k}$ for the assigned processing frequency to MU k for offloading its task. Only MUs offloading their task to the AP are allocated a non-zero processing frequency. Furthermore, all MUs offloading their task to the AP cannot exceed the total processing frequency of the cloudlet, i.e. $\sum_{k=1}^{K} f_{\mathrm{AP},k} \leq f_{\mathrm{AP}}$.

A possible strategy for the allocation of all shared communication and computation resources among the MUs offloading their tasks will be discussed in Section III.

Each MU k has a radio access to the AP through a channel using an orthogonal frequency-division multiplexing access (OFDMA) transmission scheme with bandwidth b_k. The overall bandwidth assigned to the MUs offloading their tasks has to be smaller than or equal to the total system bandwidth B, i.e. $\sum_{k=1}^{K} b_k \leq B$. All MUs computing their task locally, receive no bandwidth $b_k = 0$. For MU k, the resulting transmission rate is expressed by the Shannon channel capacity as

$$r_{\text{AP},k} = b_k \log_2\left(1 + \frac{p_{\text{trans},k}|h_k|^2}{\sigma^2}\right), \tag{3}$$

where $p_{\text{trans},k}$ denotes the transmission power of MU k, $|h_k|^2$ the wireless uplink channel gain from MU k to the AP and σ^2 the white Gaussian noise power. The channel gains $|h_k|^2$ and the noise power σ^2 can be assumed to be known by the AP. When offloading the task of MU k to the AP, the total time for offloading to the AP is the summation of transmission time and time for remote computation, i.e.

$$T_{\text{AP},k} = \frac{l_k}{r_{\text{AP},k}} + \frac{\beta_k l_k}{f_{\text{AP},k}}. \tag{4}$$

An additional transmission time for the result of the offloaded task back to the MU will be omitted, as the number of bits of the result can be assumed to be much smaller than the original task, similar to [6], [12].

For the transmission over the wireless channel, a transmit power of $p_{\text{trans},k}$ is needed by MU k. As the AP is connected to the power grid, only the energy consumed by the MUs is considered in our system model. A MU k requires a static power $p_{\text{static},k}$ during the transmission and during the computation of its task at the AP. This static power is much smaller than computation power $p_{\text{calc},k}$ or transmit power $p_{\text{trans},k}$ of MU k. Following from this, we can formulate the energy consumed for offloading the task to the AP as

$$E_{\text{AP},k} = p_{\text{trans},k} \cdot \frac{l_k}{r_{\text{AP},k}} + p_{\text{static},k} \cdot \left(\frac{l_k}{r_{\text{AP},k}} + \frac{\beta_k l_k}{f_{\text{AP},k}}\right). \tag{5}$$

It is assumed that the computation frequencies $f_{\text{MU},k}$ and powers $p_{\text{trans},k}$ and $p_{\text{static},k}$ are fixed values and can be transmitted to the AP by every MU when establishing the connection.

D. Offloading to the Cloud

Finally, a MU can also offload its task to the cloud server. This server is able to provide each MU a guaranteed processing frequency f_{cloud} that does not have to be shared. If a MU k decides to offload its task to the cloud server, the AP receives the task from the MU and transmits it over the shared backhaul link with total backhaul transmission rate R_{cloud}. The transmission rate assigned to MU k is $r_{\text{cloud},k}$ and we can formulate a similar constraint as for the other shared resources $\sum_{k=1}^{K} r_{\text{cloud},k} \leq R_{\text{cloud}}$. All MUs computing locally or offloading to the AP will receive no backhaul transmission rate, i.e. $r_{\text{cloud},k} = 0$.

Compared to the total time for offloading to the AP another additional transmission time $\frac{l_k}{r_{\text{cloud},k}}$ has to be considered, which leads to

$$T_{\text{cloud},k} = \frac{l_k}{r_{\text{AP},k}} + \frac{l_k}{r_{\text{cloud},k}} + \frac{\beta_k l_k}{f_{\text{cloud}}}. \tag{6}$$

The energy for offloading to the cloud server can be calculated as

$$\begin{aligned} E_{\text{cloud},k} =& p_{\text{trans},k} \cdot \frac{l_k}{r_{\text{AP},k}} + \\ & p_{\text{static},k} \cdot \left(\frac{l_k}{r_{\text{AP},k}} + \frac{l_k}{r_{\text{cloud},k}} + \frac{\beta_k l_k}{f_{\text{cloud}}}\right) \end{aligned} \tag{7}$$

III. Problem Formulation

A. Optimization Problem

In this section, the general computation offloading optimization problem (8) is formulated based on our system model, where the objective is the summation of the energies of all K MUs.

$$\begin{aligned} \underset{\mathbf{x}_k, \mathbf{r}_k}{\arg\min} \quad & \sum_{k=1}^{K} E_k, && (8) \\ \text{s.t.} \quad & x_{\text{AP},k} T_{\text{AP},k} \leq T_{\text{MU},k}, \quad \forall k, && (8a) \\ & x_{\text{cloud},k} T_{\text{cloud},k} \leq T_{\text{MU},k}, \quad \forall k, && (8b) \\ & \sum_{k=1}^{K} b_k \leq B, && (8c) \\ & \sum_{k=1}^{K} r_{\text{cloud},k} \leq R_{\text{cloud}}, && (8d) \\ & \sum_{k=1}^{K} f_{\text{AP},k} \leq f_{\text{AP}}, && (8e) \\ & b_k, r_{\text{cloud},k}, f_{\text{AP},k} \geq 0, \quad \forall k, && (8f) \\ & x_{\text{MU},k}, x_{\text{AP},k}, x_{\text{cloud},k} \in \{0,1\}, \quad \forall k, && (8g) \\ & x_{\text{MU},k} + x_{\text{AP},k} + x_{\text{cloud},k} = 1, \quad \forall k. && (8h) \end{aligned}$$

Three decision variables $x_{\text{MU},k}$, $x_{\text{AP},k}$ and $x_{\text{cloud},k}$ are introduced for the location in the network, where the task of MU k is computed. Constraint (8g) ensures the binarity of the variables. For a MU k, its task can be computed at only one location which is ensured by constraint (8h). Using these decision variables and the energies defined in (2), (5) and (7), the energy spend by MU k is

$$E_k = x_{\text{MU},k} E_{\text{MU},k} + x_{\text{AP},k} E_{\text{AP},k} + x_{\text{cloud},k} E_{\text{cloud},k}. \tag{9}$$

In our scenario, the time required for offloading the task of MU k to the AP or to the cloud shall not be longer than a local computation at the MU such that a good quality of experience is achieved for MU k, see constraints (8a) and (8b).

With constraints (8c)-(8e), the sum of transmission bandwidth b_k, backhaul link capacity $r_{\text{cloud},k}$ and computation resources at the cloudlet $f_{\text{AP},k}$ assigned to the MUs is limited to their individual upper bounds.

The notation of the optimization variables is simplified by defining a decision vector $\mathbf{x}_k = [x_{\text{MU},k}, x_{\text{AP},k}, x_{\text{cloud},k}]$ and a resource allocation vector $\mathbf{r}_k = [b_k, r_{\text{cloud},k}, f_{\text{AP},k}]$ for every MU k. The joint optimization (8) of the decision vectors $\mathbf{x}_k$ and the resource vectors $\mathbf{r}_k$ for all K users is a non-convex mixed integer problem and intractable. Therefore, a sub-optimum approach to this problem is proposed in the next subsection.

B. Distributed Algorithm

When considering that every MU tries to minimize its energy, which is required to obtain the result of its own computation task, it is rational to model the MUs as individual decision makers. A MU can only influence its own offloading decision under the consideration of the available resources, i.e. both its local resources and the ones provided by the network. Our approach separates the offloading decision, which will be performed at each MU, from the resource allocation, which will be executed by the AP.

The proposed algorithm is iterative. In the beginning, every MU transmits its initial decision vector $\mathbf{x}_k$, the size of its task l_k and the complexity factor β_k to the AP. The AP allocates the shared resources according to the initial decision vectors $\mathbf{x}_k$ and returns a resource vector $\mathbf{r}_k$ to each corresponding MU. Now, the first MU can redecide its initial decision, submit an updated decision vector $\mathbf{x}_1$ and the AP will recalculate the resource allocation. The algorithm terminates when no MU can save more energy by updating its decision. Throughout every iteration, the MUs only have to exchange the decision vectors $\mathbf{x}_k$ to the AP, while the AP has to return the updated resource allocation vectors $\mathbf{r}_k$ back to the corresponding MUs.

1) Resource Allocation: The AP solves (10), where the resource allocation is only dependent on the energy, because the computation time is unknown. Furthermore, the optimization is simplified in this problem formulation as the decision vectors $\mathbf{x}_k$ are given to the AP by the MUs.

$$\begin{aligned} \arg\min_{\mathbf{r}_k} \quad & \sum_{k=1}^{K} E_k, \\ \text{s.t.} \quad & (8c), (8d), (8e), (8f). \end{aligned} \tag{10}$$

The energy minimization objective can be combined with the three constraints for the shared resources (8c)-(8e) by Langrangian multipliers and problem (10) can be rewritten as

$$\begin{aligned} \mathcal{L} = & \sum_{k=1}^{K} E_k + \mu_1 \left(\sum_{k=1}^{K} b_k - B \right) + \\ & \mu_2 \left(\sum_{k=1}^{K} r_{\text{cloud},k} - R_{\text{cloud}} \right) + \mu_3 \left(\sum_{k=1}^{K} f_{\text{AP},k} - f_{\text{AP}} \right). \end{aligned} \tag{11}$$

As these resources are independent of each other, a closed form solution for the optimal allocation of each resource for MU k can be found. Taking the partial derivative with respect to b_k leads to

$$\frac{\partial \mathcal{L}}{\partial b_k} = -(x_{\text{AP},k} + x_{\text{cloud},k}) \frac{(p_{\text{trans},k} + p_{\text{static},k}) l_k}{b_k^2 \log_2 \left(1 + \frac{p_{\text{trans},k} |h_k|^2}{\sigma^2} \right)} + \mu_1 \tag{12}$$

and by setting it to zero, the fraction of the bandwidth can be found as

$$b_k = \sqrt{\frac{\frac{(x_{\text{AP},k} + x_{\text{cloud},k})(p_{\text{trans},k} + p_{\text{static},k}) l_k}{\log_2 \left(1 + \frac{p_{\text{trans},k} |h_k|^2}{\sigma^2} \right)}}{\mu_1}}. \tag{13}$$

The optimal value for the Langrangian multiplier μ_1 can be found by the partial derivative of (11) with respect to μ_1 as

$$\frac{\partial \mathcal{L}}{\partial \mu_1} = \sum_{k=1}^{K} b_k - B, \tag{14}$$

which leads to an optimal μ_1 of

$$\mu_1 = \left(\frac{1}{B} \sum_{k=1}^{K} \sqrt{\frac{(x_{\text{AP},k} + x_{\text{cloud},k})(p_{\text{trans},k} + p_{\text{static},k}) l_k}{\log_2 \left(1 + \frac{p_{\text{trans},k} |h_k|^2}{\sigma^2} \right)}} \right)^2. \tag{15}$$

Inserting (15) into (13), results in an equation for the optimal bandwidth $b_k = b_k^*$ allocated to MU k. The equation describes the fraction of the total bandwidth B that MU k will receive if it decides for offloading its task in relation to all other MUs that are offloading their tasks.

In a similar way, the optimal backhaul transmission rate for offloading the task of MU k to the cloud server can be found by the partial derivative with respect to $r_{\text{cloud},k}$, which can be written as

$$\frac{\partial \mathcal{L}}{\partial r_{\text{cloud},k}} = -x_{\text{cloud},k} \frac{p_{\text{static},k} l_k}{r_{\text{cloud},k}^2} + \mu_2 \overset{!}{=} 0 \tag{16}$$

and leads to

$$r_{\text{cloud},k} = \sqrt{\frac{x_{\text{cloud},k} p_{\text{static},k} l_k}{\mu_2}}. \tag{17}$$

The partial derivative with respect to μ_2

$$\frac{\partial \mathcal{L}}{\partial \mu_2} = \sum_{k=1}^{K} r_{\text{cloud},k} - R_{\text{cloud}} \overset{!}{=} 0 \tag{18}$$

leads to the equation for the optimal μ_2 of

$$\mu_2 = \left(\frac{1}{R_{\text{cloud}}} \sum_{k=1}^{K} \sqrt{x_{\text{cloud},k} p_{\text{static},k} l_k} \right)^2. \tag{19}$$

By inserting (19) into (17), the optimal allocated backhaul transmission rate $r_{\text{cloud},k} = r_{\text{cloud},k}^*$ for MU k is found. Each MU k is assigned the backhaul transmission rate $r_{\text{cloud},k}$ in relation to every other MU taking the decision for offloading to the cloud.

The third result is the optimal shared computation frequency at the AP assigned to the task of MU k. Taking the partial derivative with respect to $f_{\text{AP},k}$ results in

$$\frac{\partial \mathcal{L}}{\partial f_{\text{AP},k}} = -x_{\text{AP},k} \frac{p_{\text{static},k} \beta_k l_k}{f_{\text{AP},k}^2} + \mu_3 \overset{!}{=} 0 \tag{20}$$

Algorithm 1 Distributed Multi-Stage Offloading

```
Take any initial strategy profile $\mathbf{s}[0]$ and calculate corresponding optimal resource allocation $\{\mathbf{r}_k\}$
Set NE = False and $m = 0$
while NE == False do
    flag = 0; $k = 1$;
    while flag == 0 and $k \le K$ do
        AP calculates $\{\mathbf{r}_k^*\}$ for $(s_k', s_{-k}[m]), \forall s_k' \in \mathcal{S}_k$
        MU $k$ checks, if (8a) and (8b) are fulfilled, otherwise sets corresponding utility to $u_k(s_k', s_{-k}[m]) = \infty$
        if $u_k(\mathbf{s}[m]) > u_k(s_k', s_{-k}[m]), s_k' \in \mathcal{S}_k$ then
            Set $\mathbf{s}[m+1] = (s_k', s_{-k}[m])$; flag = 1; $m = m + 1$;
        else if $k == K$ then
            Set flag = 1; NE = True;
        else
            $k = k + 1$;
        end if
    end while
end while
return NE $\mathbf{s}^*$ of game $G$ and corresponding resource allocation $\{\mathbf{r}_k^*\}$
```

and the optimal value

$$f_{\text{AP},k} = \sqrt{\frac{x_{\text{AP},k} p_{\text{static},k} \beta_k l_k}{\mu_3}}. \tag{21}$$

The partial derivative with respect to μ_3 is

$$\frac{\partial \mathcal{L}}{\partial \mu_3} = \sum_{k=1}^{K} f_{\text{AP},k} - f_{\text{AP}} \overset{!}{=} 0 \tag{22}$$

leads to the equation for the optimal μ_3 of

$$\mu_3 = \left(\frac{1}{f_{\text{AP}}} \sum_{k=1}^{K} \sqrt{x_{\text{AP},k} p_{\text{static},k} \beta_k l_k} \right)^2. \tag{23}$$

The insertion of (23) into (21) leads to the result that each MU k offloading its task to the AP will receive the optimal fraction of the total computation frequency $f_{\text{AP}} = f_{\text{AP}}^*$ relative the to static powers, task sizes and complexity factors of all other MUs also offloading to the AP.

The optimal values for the three shared resources can be written as an optiomal resource allocation vector $\mathbf{r}_k^* = [b_k^*, r_{\text{cloud},k}^*, f_{\text{AP}}^*]$.

2) Offloading Decisions: For the offloading decisions of the MUs, we define a strategic form game. The players of the game are the MUs, which are defined by the set $\mathcal{K} = \{1, \ldots, K\}$. Every MU k has a set of strategies $(\mathcal{S}_k)_{k \in \mathcal{K}}$, written as

$$\mathcal{S}_k = \Big\{ s_k = (x_{\text{MU},k}, x_{\text{AP},k}, x_{\text{cloud},k}) \mid \tag{24}$$
$$x_{\text{MU},k} + x_{\text{AP},k} + x_{\text{cloud},k} = 1;$$
$$x_{\text{MU},k}, x_{\text{AP},k}, x_{\text{cloud},k} \in \{0, 1\} \Big\},$$

which defines the decision of MU k. The decisions of all K MUs can be expressed by a strategy profile $\mathbf{s} = (s_k, s_{-k})$, where $s_k \in \mathcal{S}_k$ is one possible strategy of MU k and $s_{-k} = (s_1, \ldots, s_{k-1}, s_{k+1}, \ldots, s_K) \in \mathcal{S}_{-k} = \prod_{j \neq k} \mathcal{S}_j, \forall k$ are the played strategies of the other MUs. In order to verify the timing constraints (8a) and (8b), the utility function of MU k is defined as

$$u_k(\mathbf{s}) = \begin{cases} \infty, & \text{for } x_{\text{AP},k} T_{\text{AP},k} > T_{\text{MU},k} \\ & \text{or } x_{\text{cloud},k} T_{\text{cloud},k} > T_{\text{MU},k} \\ E_k, & \text{else.} \end{cases} \tag{25}$$

Accordingly, the game can be defined as $\mathcal{G} = (\mathcal{K}, (\mathcal{S}_k)_{k \in \mathcal{K}}, (u_k)_{k \in \mathcal{K}})$.

As the resource allocation is executed independently from the offloading decisions, each MU is able to compute his utility function independent of the other MUs. Due to this, game $\mathcal{G}$ can be proven to be an exact potential game by defining a potential function $\Phi(\mathbf{s}) = \sum_{k=1}^{K} E_k$ and calculating the difference of two strategies s_k and s_k'. The existence of a Nash Equilibrium has been proven for an exact potential game [13].

The proposed algorithm is summarized in Algorithm 1.

IV. Numerical Results

For our simulations, a scenario with $K = 5$ MUs is considered. Each MU has a task of length l_k that is randomly chosen out of a uniform distribution between 2 MB and 10 MB. The computation frequencies are set as $f_{\text{MU},k} = 1$ GHz for all MUs, $f_{\text{AP}} = 2$ GHz for the cloudlet and $f_{\text{cloud}} = 4$ GHz for the cloud server. Powers are set to $p_{\text{calc},k} = 2$ W for local calculation, $p_{\text{trans},k} = 1$ W for transmitting the task and $p_{\text{static},k} = 1$ W as static power of the MUs hardware.

To model the wireless channel between the MUs and the AP, the signal-to-noise ratio is set to 0 dB and the channel gain $|h_k|^2$ from MU k to the AP is modeled randomly with uniform distribution and an average $E\{|h_k|^2\} = 1$. A total transmission bandwidth of $B = 100$ MHz and a total backhaul transmission rate of $R_{\text{cloud}} = 100$ Mbit/s are assumed.

The performance of our algorithm is assessed against full local computation and against the optimal policy, which is calculated by computing all possible 3^K combinations of offloading decisions for the K MUs. For each decision combination, the corresponding resource allocation problem is solved. For Figures 2 and 3, 100 Monte Carlo runs were performed per data point.

Figure 2 shows the total energy consumed by all K MUs as a function of the task complexity β_k. For low β_k no MU offloads and the results for all policies are equal. In the

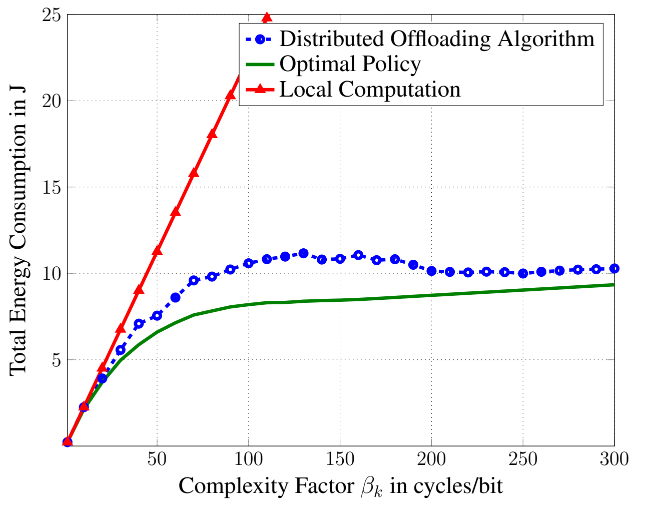

Fig. 2. Total energy consumption of the MUs for different task complexities

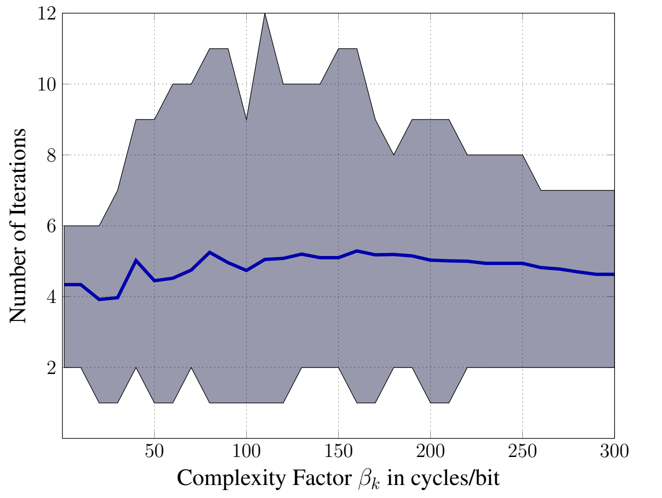

Fig. 3. Average number of iterations for different task complexities

plotted range for task complexity β_k from 1 to 300, not all MU are able to offload their tasks and meet the maximum time constraint. While the curve for local computation rapidly increases, the proposed algorithm and the optimal policy stay at a much lower level of about 10 J total energy consumption.

Although, our algorithm leads to suboptimal offloading decisions, the offloading gain $\frac{E_{\text{local}}}{E_{\text{offloading}}}$ of our algorithm is still close to the offloading gain of the optimal policy. For $\beta_k = 100$, our algorithm has a gain over local computation of 2.13, while the optimal policy has a gain of 2.75. At $\beta_k = 250$, the gain increases to 5.64 for our algorithm compared to 6.24 for the optimal policy. This result also proves the effectiveness of offloading in general for tasks with high computational complexity.

The average number of iterations m until the algorithm reaches a NE is shown in Figure 3 as a blue line. Furthermore, the gray area represents the variance of the number of iterations. Even for the overall maximum number of 12 iterations, the number of computations our algorithm requires is much smaller than the computation of the optimal policy.

V. Conclusion

We introduce a time constrained energy minimization problem for a multi-stage computation offloading scenario with multiple MUs connected to an AP. A distributed algorithm separating the resource allocation at the AP and the autonomous offloading decisions at the MUs is formulated. Due to the separation, the required knowledge at each entity in the network is limited and signaling is greatly reduced. The proposed algorithm has proven to be computationally efficient by reaching performance results close to the computationally exhaustive optimal policy with only a few iterations.

Acknowledgement

This work has been performed in the context of the DFG Collaborative Research Center (CRC) 1053 MAKI - subprojects B3 and C7.

Supported by DAAD with funds from the Federal Ministry of Education and Research (BMBF).

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[2] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing." *Proc. of the 2nd USENIX Conf. Hot Topics on Cloud Computing*, pp. 1–4, 2010.

[3] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *IEEE Computer Magazine*, vol. 43, no. 4, pp. 51–56, 2010.

[4] D. Ferreira, A. K. Dey, and V. Kostakos, "Understanding human-smartphone concerns: a study of battery life," in *Proc. of the International Conference on Pervasive Computing*. Springer, 2011, pp. 19–33.

[5] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing - a key technology towards 5G," *ETSI White Paper*, vol. 11, 2015.

[6] H. Q. Le, H. Al-Shatri, and A. Klein, "Efficient resource allocation in mobile-edge computation offloading: Completion time minimization," in *Proc. of the IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 2513–2517.

[7] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. of the IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 1451–1455.

[8] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, 2016.

[9] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 45–55, 2014.

[10] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, 2015.

[11] M.-H. Chen, M. Dong, and B. Liang, "Multi-user mobile cloud offloading game with computing access point," in *Proc. of 5th IEEE International Conference on Cloud Networking (Cloudnet)*, 2016, pp. 64–69.

[12] C. You and K. Huang, "Multiuser resource allocation for mobile-edge computation offloading," in *IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[13] S. Lasaulce and H. Tembine, *Game theory and learning for wireless networks: fundamentals and applications*. Academic Press, 2011.