Distributed Algorithm for Energy Efficient Joint Cloud and Edge Computing with Splittable Tasks

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

***Abstract*—The considered hierarchical multi-level offloading scenario consists of multiple mobiles units (MUs), an access point (AP) with attached cloudlet for mobile edge computing (MEC) and a cloud server. Each user has an arbitrarily splittable task and three possible options for the computation of fractions of this task, which are local computation, offloading to the cloudlet and offloading to the cloud server. We decompose a non-linear central energy minimization problem into subproblems and propose a distributed algorithm that separates the allocation of shared communication and computation resources by the AP from the offloading decisions by the MUs. The AP assigns fractions of the shared bandwidth of the radio access channel, the shared backhaul transmission link to the cloud server and the shared computation frequency at the cloudlet according to offloading decisions of the MUs by solving closed-form expressions which are derived in this paper. Given the available resources, each MU solves a linear optimization problem to calculate the optimal fractions of its task to be computed locally or offloaded. In numerical simulations, the algorithm is proven to be stable and reaching results close to the optimal policy.**

I. Introduction

Mobile devices like smartphones or laptops play an increasingly important role in the daily life, but their battery capacity limits the execution of computation intensive tasks. A study revealed that most users are very concerned about keeping the battery level as high as possible [1]. Computation offloading has been proposed as a solution to the battery capacity bottleneck. A user transmits the task to be computed to a cloud server, receives the result of the computation back and thus he consumes less energy for wireless transmission of the task as compared to computing locally [2]. Besides offloading to a cloud server with high latency, mobile edge computing (MEC) is an alternative approach to computation offloading [3], [4]. In MEC a small-scale server called cloudlet is attached to a base station or access point (AP). The cloudlet can be used for time-critical applications with high computational complexity like augmented reality or speech processing [5].

As shown in a survey on computation offloading [6], it is a broad field of research that includes considerations on multi-layer computing architectures [7], placement of required services for computation [8] or the interaction between users, providers of computation servers and service operators [9]. This paper focuses on the influence of shared computation and communication resources on the offloading decision of a mobile unit (MU).

Most papers considering a similar aspect of computation offloading treat smaller scenarios with a single offloading location in the network, e.g. a cloudlet for MEC [5], [10]. As offloading of a task requires transmission of the task to the cloudlet and the result back to the user as well as computation at the cloudlet, the authors of [5] and [10] propose a joint optimization of communication and computation resources. The authors of [10] formulate an energy minimization problem. MUs with a stream of non-splittable tasks offload a task only if the resources allocated to a MU are sufficient. The proposed algorithm is efficiently modelled by queueing theory. The authors of [5] consider a time minimization problem. The maximum computation time of the MUs or the cloudlet shall be minimized under the assumption that each MU has a single splittable task. MUs can decide to process one fraction of the task locally, while the rest is offloaded to the cloudlet. An algorithm based on bisection search is proposed to find the minimum total computation time in the network. The algorithms formulated in [5], [10] are calculated by a central entity, e.g. the AP, which is assumed to have knowledge about all parameters in the network.

Recent papers like [11] and [12] propose hierarchical network models with three levels, one for the MUs, one for the AP and one for the cloud server. MUs and the cloud server have no direct connection to each other. Therefore, offloading to the cloud server requires two hops and all offloaded data is handled by the AP. In [11], a flow management optimization for queues of offloaded tasks is introduced. MUs have taken the offloading decision already when their tasks are added to a queue. The authors of [12] propose a multi-level network consisting of one AP with attached cloudlet and one cloud server. Their model combines energy minimization and minimization of the maximum computation time in one objective function. Energy and time are connected through a weighting variable whose choice has a strong influence on the optimization results. Although the algorithm is based on game theory and could be executed distributed by all MUs, the convergence to a Nash equilibrium is achieved when computing with full knowledge centrally at the AP or at one of the MUs.

In this paper, a multi-level offloading scenario is investigated which combines a low latency MEC system with a cloud server. Multiple MUs have to share the available communication and computation resources. Each MU has an arbitrarily splittable computation task whose fractions can be computed locally, at the cloudlet and at the cloud server. Every MU has to decide on the splits of its task considering its available resources. An energy minimization problem limited by a maximum computation time constraint is formulated. The energy minimization problem is decomposed into subproblems and

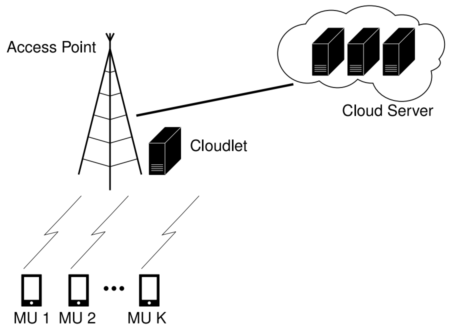

Fig. 1. Multi-level computation offloading scenario with K MUs, a single AP with attached cloudlet and a cloud server connected over a backhaul link

approximately solved by an iterative algorithm. Our proposed algorithm is executed distributedly by the AP and all MUs. In an iterative way, the MUs decide autonomously on the fractions to be computed locally and to be offloaded based on the available communication and computation resources, while the AP allocates the shared resources according to the offloading decisions. The proposed algorithm requires only a limited set of parameters at each entity in the network and thereby, reduces signaling of the parameters by the MUs to the AP or among the MUs.

The paper is structured as follows: Section II introduces the scenario. We formulate the optimization problem to our model in Section III and show how it can be transformed into a distributed algorithm in Section IV. In Section V, the theoretical results are tested numerically.

II. System Model

A. Scenario

A multi-tier computation offloading scenario consisting of K MUs, a single AP with an attached cloudlet and a cloud server is considered. This scenario is shown in Figure 1 and can be part of a larger network with multiple APs. The MUs are connected to the AP through a shared radio access channel. The AP has a connection over a backhaul link with limited capacity to the cloud server. AP and cloud server have unlimted access to energy as they are connected to the power grid. Therefore, only the energy consumed by the MUs is considered in our model.

Each MU k with $k \in \{1, \dots, K\}$ has a splittable task of length l_k, measured in bits. The task of MU k is assumed to be arbitarily splittable, so that one fraction $x_{\text{MU},k}$ is computed locally, one fraction $x_{\text{AP},k}$ is computed at the AP, and another fraction $x_{\text{cloud},k}$ is offloaded to the cloud server with $x_{\text{MU},k} + x_{\text{AP},k} + x_{\text{cloud},k} = 1$. Since the computation of diverse types of tasks can require a different number of CPU cycles [13], a complexity factor β_k for the task of MU k is introduced. This complexity factor β_k determines the number of CPU cycles required to compute one bit of the task of MU k.

B. Local Computation

Each MU k has a processing frequency $f_{\text{MU},k}$, measured in cycles per second. If MU k decides to compute the whole task locally, the total local computation time can be defined as

$$T_{\text{MU,total},k} = \frac{\beta_k l_k}{f_{\text{MU},k}}. \tag{1}$$

The local computation time scales linearly with the number of bits to be computed and therefore, the computation of a fraction of the task takes an equivalent fraction of computation time, i.e.

$$T_{\text{MU},k} = \frac{x_{\text{MU},k}\beta_k l_k}{f_{\text{MU},k}}. \tag{2}$$

By introducing a calculation power $p_{\text{calc},k}$ the hardware of MU k requires during computation, the local computation energy $E_{\text{MU},k}$ can be written as the product of calculation power and local computation time as

$$E_{\text{MU},k} = p_{\text{calc},k} \cdot \frac{x_{\text{MU},k}\beta_k l_k}{f_{\text{MU},k}} \tag{3}$$

and the energy for total local computation can be denoted by $E_{\text{MU,total},k}$.

C. Computation at the AP

The first possible location for offloading the task or a fraction of it is the cloudlet attached to the AP, which will be called offloading to the AP in the following. The cloudlet is a small server with a processing frequency f_{AP} that is assumed to be higher than the computation frequencies of the MUs, i.e. $f_{\text{MU},k} \leq f_{\text{AP}}$. If only one MU is offloading to the AP, it receives the full processing frequency for its computation. Otherwise, the computation frequency at the AP is shared and the fraction allocated to MU k is denoted as $f_{\text{AP},k}$. The total allocated computation frequency cannot exceed the maximum available processing frequency, i.e. $\sum_{k=1}^{K} f_{\text{AP},k} \leq f_{\text{AP}}$. A possible approach for the allocation of this shared resource will be discussed in Sections III and IV.

Each MU is connected to the AP through a wireless radio channel which uses an orthogonal frequency-division multiple access (OFDMA) transmission scheme. If MU k offloads a fraction of its task, it receives a fraction b_k of the total available bandwidth B. In total, the bandwidth allocated to offloading MUs cannot exceed the total bandwidth B, i.e. $\sum_{k=1}^{K} b_k \leq B$. The transmission rate of MU k to the AP with bandwidth b_k, transmission power $p_{\text{trans},k}$, the uplink channel gain $|h_k|^2$ and white Gaussian noise power σ^2 is expressed by the Shannon channel capacity as

$$r_{\text{AP},k} = b_k \log_2\left(1 + \frac{p_{\text{trans},k}|h_k|^2}{\sigma^2}\right). \tag{4}$$

Channel gains $|h_k|^2$ and noise power σ^2 are assumed to be known at the AP.

The offloading time for a fraction of the task of MU k to the AP can be written as the sum of three actions: transmission of the task to the AP, computation at the AP and transmission of the result back to MU k. In our model, the result is assumed to be much smaller than the original task and transmission of the result to the MU is omitted, similar to [5], [14]. The time for transmission and computation at the AP can be written as

$$T_{\text{AP},k} = \frac{x_{\text{AP},k} l_k}{r_{\text{AP},k}} + \frac{x_{\text{AP},k} \beta_k l_k}{f_{\text{AP},k}}. \tag{5}$$

To model the energy consumed by MU k for offloading a fraction of its task to the AP, two powers are required. Transmission power $p_{\text{trans},k}$ models the power of the modem required in the transmission phase and static power $p_{\text{static},k}$ describes the idle power of the remaining hardware components of the MU. The static power has to be considered during transmission and while the MU is waiting for the results of the computation at the AP. Then, the energy for offloading a fraction of the the task to the AP is

$$E_{\text{AP},k} = (p_{\text{trans},k} + p_{\text{static},k}) \cdot \frac{x_{\text{AP},k} l_k}{r_{\text{AP},k}} + p_{\text{static},k} \cdot \frac{x_{\text{AP},k} \beta_k l_k}{f_{\text{AP},k}}. \tag{6}$$

D. Computation at the Cloud

The cloud server is the second possible offloading location. It is assumed to have plenty of computation resources available and offer a fixed processing frequency f_{cloud} to each MU offloading a fraction there. The frequency is higher than the local processing frequency of a MU, i.e. $f_{\text{MU},k} \leq f_{\text{cloud}}$.

Access to the cloud server is possible over a backhaul link from the AP with a total backhaul transmission rate R_{cloud}. A MU has to send the fraction of its task to be computed at the cloud over the shared access channel to the AP which then forwards it over the backhaul link to the cloud. Offloading by multiple MUs requires sharing of the backhaul link and the assigned backhaul transmission rate for MU k is denoted by $r_{\text{cloud},k}$. The sum of all assigned rates has to be less than the total backhaul transmission rate, i.e. $\sum_{k=1}^{K} r_{\text{cloud},k} \leq R_{\text{cloud}}$.

Due to the backhaul transmission an additional latency has to be considered when offloading a fraction of a task to the cloud. During this interval, MU k stays idle and only requires the static power $p_{\text{static},k}$. The time for transmission and offloading is formulated as

$$T_{\text{cloud},k} = \frac{x_{\text{cloud},k} l_k}{r_{\text{AP},k}} + \frac{x_{\text{cloud},k} l_k}{r_{\text{cloud},k}} + \frac{x_{\text{cloud},k} \beta_k l_k}{f_{\text{cloud}}}, \tag{7}$$

and the corresponding energy is

$$\begin{aligned} E_{\text{cloud},k} = (p_{\text{trans},k} + p_{\text{static},k}) \cdot \frac{x_{\text{cloud},k} l_k}{r_{\text{AP},k}} + \\ p_{\text{static},k} \cdot \left(\frac{x_{\text{cloud},k} l_k}{r_{\text{cloud},k}} + \frac{x_{\text{cloud},k} \beta_k l_k}{f_{\text{cloud}}} \right). \end{aligned} \tag{8}$$

III. Optimization Problem

The total energy for the computation of the task of MU k is expressed by

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

A decision vector $\mathbf{x}_k = [x_{\text{MU},k}, x_{\text{AP},k}, x_{\text{cloud},k}]$ and a resource vector $\mathbf{r}_k = [b_k, r_{\text{cloud},k}, f_{\text{AP},k}]$ is assigned to each MU k. To simplify the notation of the optimization arguments, the vectors of all K MUs are summarized in a decision matrix $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_K]$ and a resource matrix $\mathbf{R} = [\mathbf{r}_1 \ldots \mathbf{r}_K]$, respectively.

To minimize the energy consumed by all MUs in the system, the following optimization problem can be formulated:

$$\begin{aligned}
\underset{\mathbf{X},\mathbf{R}}{\arg\min} \quad & \sum_{k=1}^{K} E_k, && (9)\\
\text{s.t.} \quad & x_{\text{AP},k} T_{\text{AP},k} \leq T_{\text{MU,total},k}, \quad \forall k, && (10a)\\
& x_{\text{cloud},k} T_{\text{cloud},k} \leq T_{\text{MU,total},k}, \quad \forall k, && (10b)\\
& \sum_{k=1}^{K} b_k \leq B, && (10c)\\
& \sum_{k=1}^{K} r_{\text{cloud},k} \leq R_{\text{cloud}}, && (10d)\\
& \sum_{k=1}^{K} f_{\text{AP},k} \leq f_{\text{AP}}, && (10e)\\
& b_k, r_{\text{cloud},k}, f_{\text{AP},k} \geq 0, \quad \forall k, && (10f)\\
& 0 \leq x_{\text{MU},k}, x_{\text{AP},k}, x_{\text{cloud},k} \leq 1, \quad \forall k, && (10g)\\
& x_{\text{MU},k} + x_{\text{AP},k} + x_{\text{cloud},k} = 1, \quad \forall k. && (10h)
\end{aligned}$$

Constraints (10a) and (10b) represent the maximum allowed computation time. To ensure that offloading is always beneficial in terms of time, offloading of fractions or the whole task of MU k shall not take longer than the local computation of the whole task would take. The sums of the shared resources of the access channel, the backhaul link from the AP to the cloud server and the computation frequency at the AP are limited to their respective upper bounds in constraints (10c)-(10e). Each resource assigned to every MU has to be non-negative, which is expressed by (10f). The fraction of the task of MU k to be computed at each possible location has to be also non-negative and has an upper bound of 1, for the case the whole task is computed at one location. This is formulated by constraint (10g). The last constraint (10h) ensures that the whole task of each MU is computed.

IV. Distributed Algorithm

Although the proposed offloading problem can be formulated as shown in Section III, this formulation has disadvantages. First, the optimization problem is non-linear and non-convex and thus, not efficiently solvable. Another disadvantage is the centrality of the optimization problem. The solution has to be calculated at one location in the network, e.g. at the AP. Every MU has to share all parameters required for the calculation and will receive an offloading decision back. Therefore, a MU has no active part in the decision process.

In this section, a distributed algorithm is proposed that separates resource allocation at the AP and the offloading decisions by the MUs into two subproblems. Before starting the algorithm, task length l_k, complexity factor β_k and initial offloading decisions of the MUs are transmitted to the AP. Initial offloading decisions of the MUs can, e.g. be modeled by full local computation of the tasks or by the corresponding fractions when assuming an equal distribution of the shared resources among all K MUs. Afterwards, the MUs are sorted in descending order according to their respective possible energy saving.

Now, an iterative process begins and the shared resources are assigned based on these offloading decisions at the AP. After a MU receives feedback about the allocated fraction of the shared resources, the MU can autonomously reconsider and possibly, update the offloading decision. One after another, all MUs receive information about their available resources and can redecide on the fractions of their tasks to be computed locally or offloaded.

The proposed algorithm is summarized in Figure 2. In the following subsections, the sorting, the resource allocation and the offloading decisions are explained in more details.

A. Sorting

Until now, the assignment of the numbers $1, \dots, K$ to the K MUs is random. Since the total computation energy of all K MUs should be minimized, the MU with the highest possible reduction of its energy consumption by offloading shall decide first. The optimal ranking would be calculated by $\arg\max_{k \in \mathcal{K}} |E_{\text{MU,total},k} - E_k|$ with $\mathcal{K}$ being the set of all K MUs. This metric is not applicable, because it requires the offloading decisions and shared resources to be known. Therefore, more simple metrics for ordering have to be evaluated. Under the assumption that each MU will offload its whole task and will receive all communication and computation resources, possible metrics are: the task length l_k, the number of required CPU cycles $\beta_k l_k$, or the required energy for offloading to the AP $E_{\text{AP},k}$. Each metric sorts the MUs in descending order.

B. Resource Allocation

Assuming that offloading decisions of all MUs are available at the AP, the shared resources have to be optimized accordingly. Optimization problem (9) can be rewritten only in dependence of the shared resource matrix $\mathbf{R}$ as

$$\begin{aligned} \arg\min_{\mathbf{R}} \quad & \sum_{k=1}^{K} E_k, \\ \text{s.t.} \quad & (10c), (10d), (10e), (10f). \end{aligned} \tag{11}$$

This formulation is still non-convex, but can be reformulated by a Lagrange function $\mathcal{L}$ as

$$\begin{aligned} \mathcal{L} = & \sum_{k=1}^{K} E_k + \mu_1 \left(\sum_{k=1}^{K} b_k - B \right) + \\ & \mu_2 \left(\sum_{k=1}^{K} r_{\text{cloud},k} - R_{\text{cloud}} \right) + \mu_3 \left(\sum_{k=1}^{K} f_{\text{AP},k} - f_{\text{AP}} \right). \end{aligned} \tag{12}$$

Since the shared wireless bandwidth, backhaul link transmission rate and computation frequency at the AP are independent of each other, the derivation of this Lagrangian function leads to one separate equation for the optimal allocation of each resource. The corresponding KKT-conditions and the derivation can be found in the Appendix.

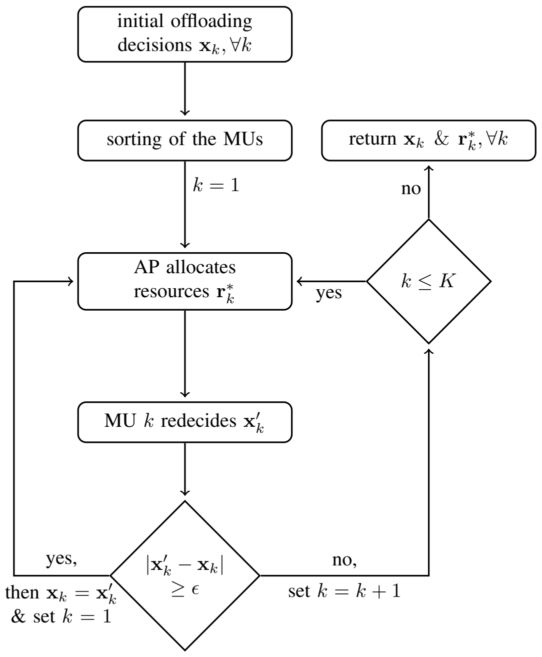
initial offloading decisions $\mathbf{x}_k, \forall k$
sorting of the MUs
$k = 1$
AP allocates resources $\mathbf{r}_k^*$
MU k redecides $\mathbf{x}_k'$
$|\mathbf{x}_k' - \mathbf{x}_k| \geq \epsilon$
yes, then $\mathbf{x}_k = \mathbf{x}_k'$ & set $k = 1$
no, set $k = k + 1$
$k \leq K$
yes
no
return $\mathbf{x}_k$ & $\mathbf{r}_k^*, \forall k$

Fig. 2. Flow diagram of the iterative algorithm

The optimal fraction of the total bandwidth that MU k requires to offload the desired fraction of its task is

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

the optimal fraction of the backhaul transmission rate is

$$r_{\text{cloud},k}^* = R_{\text{cloud}} \cdot \frac{\sqrt{x_{\text{cloud},k} p_{\text{static},k} l_k}}{\sum_{k=1}^{K} \sqrt{x_{\text{cloud},k} p_{\text{static},k} l_k}} \tag{14}$$

and the optimal fraction of the total available processing frequency at the AP can be calculated by

$$f_{\text{AP},k}^* = f_{\text{AP}} \cdot \frac{\sqrt{x_{\text{AP},k} p_{\text{static},k} \beta_k l_k}}{\sum_{k=1}^{K} \sqrt{x_{\text{AP},k} p_{\text{static},k} \beta_k l_k}}. \tag{15}$$

These results show that a MU k offloading a fraction of its task to one or both of the remote locations, will receive a fraction of the shared communication and computation resources relative to the overall amount of offloaded data by all MUs.

C. Offloading Decisions

After the separation of offloading decisions and resource allocation, the remaining optimization problem for the offloading decisions becomes

$$\begin{aligned} \arg\min_{\mathbf{X}} \quad & \sum_{k=1}^{K} E_k, \\ \text{s.t.} \quad & (10a), (10b), (10g), (10h). \end{aligned} \tag{16}$$

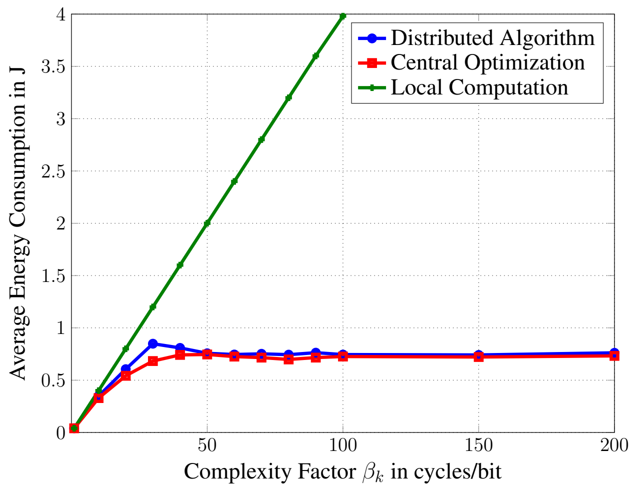

Fig. 3. Total energy consumption of the MUs for different task complexities

The energy of each MU k is independent of all other MUs, as the shared resources are assigned before the update of the offloading decision. Therefore, optimization problem (16) can be simplified to

$$\begin{aligned} \underset{\mathbf{x}}{\arg\min} \quad & E_k, \\ \text{s.t.} \quad & (10a), (10b), (10g), (10h). \end{aligned} \tag{17}$$

This formulation is a standard linear optimization problem that can be solved efficiently and fast by available solvers and can also be computed on a MU.

V. Numerical Results

The scenario consists of $K = 5$ MUs, each with a task of length $l_k = 10$ MB and a computation frequency of $f_{\text{MU},k} =$ 1 GHz. The cloudlet has a shared computation frequency of $f_{\text{AP}} = 4$ GHz and the cloud server provides $f_{\text{cloud}} = 8$ GHz to each MU. Every MU has a local calculation power of $p_{\text{calc},k} = 500$ mW, a transmission power of $p_{\text{trans},k} = 200$ mW and a static power of $p_{\text{static},k} = 20$ mW. The radio access channel is modeled with a total bandwidth of $B = 100$ MHz, a white Gaussian noise power of $\sigma^2 = 10^{-13}$ W and a channel gain of $|h_k|^2 = 1/d^{-3}$ with d being a randomly chosen distance of the MU from the AP in a range from 10 m to 50 m. The shared backhaul transmission link to the cloud server has a total transmission rate of $R_{\text{cloud}} = 100$ Mbit/s. For initialization, the shared resources are allocated equally among all MUs and their initial offloading decisions are calculated accordingly.

For the first result shown in Figure 3, the average energy required of a MU is calculated for different complexity factor β values in a range of from 1 to 200. Performing 50 Monte Carlo runs per data point, the proposed algorithm is tested against the central optimization problem (9) calculated by the BARON non-convex solver and against a full local computation of the tasks. While local computation linearly increases, the two policies with offloading consume much less energy and the curves increase only slightly. The distributed algorithm shows an energy consumption close to the central solution. Only for complexity factors β_k between 30 and 100 a larger deviation is visible. In this range, not all MUs are able to offload with their allocated resources and the calculation of a solution requires more iterations for settlement. For higher values of β_k, all MUs offload fully their tasks to either the AP or the cloud server and the distributed algorithm nearly reaches the central optimization result.

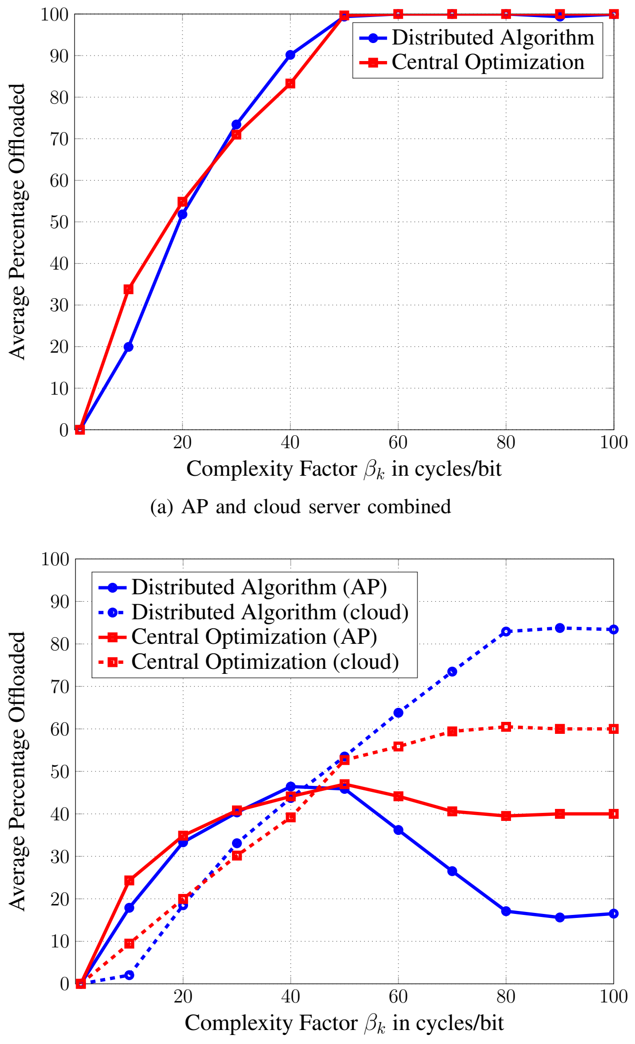

Fig. 4. Percentage of offloaded data for different task complexities

When investigating the average percentage of offloaded computations per MU in Figure 4, more differences between the central optimization problem and the proposed distributed algorithm become apparent. In Figure 4a, the sum of offloaded data to the AP and to the cloud server is shown. The curves of the central optimization and the distributed algorithm are comparable in steepness and saturation point. The central optimization results in more offloaded data for a complexity factor of 10, while the distributed algorithm is offloading slightly more data at a complexity factor of

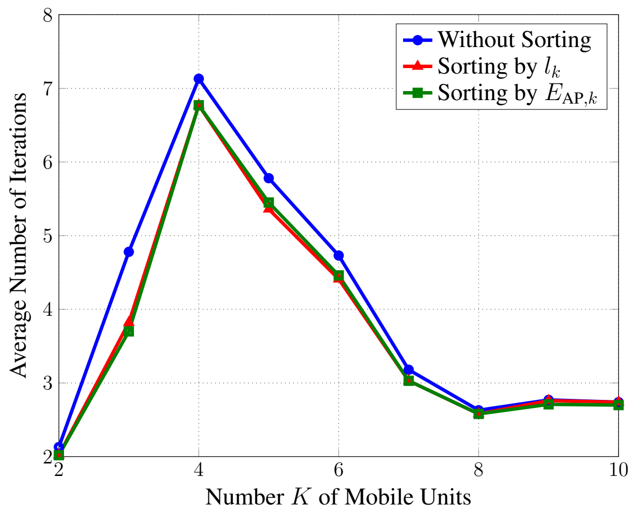

Fig. 5. Average number of iterations for different numbers of MUs

40. A closer inspection of the location to which the MUs offload the tasks reveals different offloading decisions of the MUs in the central optimization compared to the distributed algorithm. This is shown in Figure 4b. For complexity factors between 1 and 50, both methods are behaving in a similar way. First, one MU starts to offload its task to the AP. When the increasing complexity factor leads to a higher maximum local computation time, a second MU starts to offload to the cloud server with higher latency. At complexity factors higher than 50, all MUs decide for full offloading of the tasks. A MU calculating its offloading decision based on the distributed algorithm is missing global knowledge about the decisions of the other MUs. Therefore, it is visible that a MU prefers the cloud server with fixed computation frequency over the shared computation capabilities at the AP, despite the higher latency for the transmission.

For the results in Figure 5, each MU is modeled to have a random task size between 5 MB and 25 MB and β_k is fixed to 50. For different numbers K of MUs, the average number of iterations is computed with 100 Monte Carlo runs per data point. For $K = 4$ and $\beta_k = 50$, a steep increase of the required iterations is visible in Figure 5. Like in the result described in Figure 3, at this point not all MUs are able to offload. With a further increasing number of MUs, the initially available resources per MU decrease and more MUs decide for local computation. Therefore, the remaining MUs reach their final offloading decisions in a few iterations. Both curves representing an initial sorting of the MUs according to the task length l_k or to the energy $E_{\text{AP},k}$ for computing at the AP are nearly identical. Even the simple sorting by a decreasing task size can reduce the average number of required iterations.

VI. Conclusion

A hierarchical multi-level scenario with two locations for computation offloading is investigated and a central energy minimization problem with a maximum computation time constraint is formulated. The problem is remodeled into a distributed, iterative algorithm that requires only a limited set of parameters at the MUs and the AP. Thereby, signaling is reduced and each MU is able to take an autonomous offloading decision. In numerical simulations, the proposed algorithm achieves results close to the original non-convex energy minimization problem, while having a significantly lower computation time.

Acknowledgement

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

SPONSORED BY THE

Federal Ministry of Education and Research

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

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] R. Vilalta, V. Lopez, A. Giorgetti, S. Peng, V. Orsini, L. Velasco, R. Serral-Gracia, D. Morris, S. De Fina, F. Cugini *et al.*, "Telcofog: A unified flexible fog and cloud computing architecture for 5g networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 36–43, 2017.

[8] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service entity placement for social virtual reality applications in edge computing," in *Proc. of the IEEE INFOCOM 2018 - Conference on Computer Communications*, 2018.

[9] H. Zhang, Y. Zhang, Y. Gu, D. Niyato, and Z. Han, "A hierarchical game framework for resource management in fog computing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 52–57, 2017.

[10] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Joint allocation of computation and communication resources in multiuser mobile cloud computing," in *Proc. of the IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2013, pp. 26–30.

[11] K. Kaur, S. Garg, G. S. Aujla, N. Kumar, J. J. Rodrigues, and M. Guizani, "Edge computing in the industrial internet of things environment: Software-defined-networks-based edge-cloud interplay," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 44–51, 2018.

[12] 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.

[13] 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.

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

Appendix

The gradient of (12) with respect to the shared resources of MU k is

$$\nabla_{\boldsymbol{r}_k}\mathcal{L}(\boldsymbol{r}_k,\boldsymbol{\mu}_k) = \begin{pmatrix} -(x_{\text{AP},k}+x_{\text{cloud},k})\dfrac{(p_{\text{trans},k}+p_{\text{static},k})l_k}{b_k^2\log_2\left(1+\frac{p_{\text{trans},k}|h_k|^2}{\sigma^2}\right)}+\mu_1 \\ -x_{\text{cloud},k}\dfrac{p_{\text{static},k}l_k}{r_{\text{cloud},k}^2}+\mu_2 \\ -x_{\text{AP},k}\dfrac{p_{\text{static},k}\beta_k l_k}{f_{\text{AP},k}^2}+\mu_3 \end{pmatrix} \tag{18}$$

and the remaining KKT-conditions are

$$\sum_{k=1}^{K} b_k^* - B \leq 0, \tag{19}$$

$$\sum_{k=1}^{K} r_{\text{cloud},k}^* - R_{\text{cloud}} \leq 0, \tag{20}$$

$$\sum_{k=1}^{K} f_{\text{AP},k}^* - f_{\text{AP}} \leq 0, \tag{21}$$

$$\mu_1^*,\mu_2^*,\mu_3^* \geq 0, \tag{22}$$

$$\mu_1^*\left(\sum_{k=1}^{K} b_k^* - B\right) = 0, \tag{23}$$

$$\mu_2^*\left(\sum_{k=1}^{K} r_{\text{cloud},k}^* - R_{\text{cloud}}\right) = 0, \tag{24}$$

$$\mu_3^*\left(\sum_{k=1}^{K} f_{\text{AP},k}^* - f_{\text{AP}}\right) = 0. \tag{25}$$

As the three shared resources are independent of each other, which is shown in (18), only one μ_i is active per resource.

In the following, only the proofs of (25) for the shared computation frequency $f_{\text{AP},k}^*$ will be discussed, since the proofs for b_k^* and $r_{\text{cloud},k}^*$ can be formulated similarly. From (18), the shared computation frequency $f_{\text{AP},k}$ can be calculated as

$$f_{\text{AP},k} = \sqrt{\frac{x_{\text{AP},k}p_{\text{static},k}\beta_k l_k}{\mu_3}} \tag{26}$$

The derivative of (12) with respect to μ_3 leads to

$$\sum_{k=1}^{K} f_{\text{AP},k} - f_{\text{AP}} = 0 \tag{27}$$

and the optimal value of the Lagrangian multiplier μ_3^* can be calculated as

$$\mu_3^* = \left(\frac{1}{f_{\text{AP}}}\sum_{k=1}^{K}\sqrt{x_{\text{AP},k}p_{\text{static},k}\beta_k l_k}\right)^2. \tag{28}$$

The optimal fraction of the shared computation frequency k $f_{\text{AP},k}^*$ at the AP for MU can be found by inserting (28) in (26).

With the results for $f_{\text{AP},k}^*$ and μ_3^*, the validity of the KKT-conditions can be discussed. The variables $p_{\text{static},k}$, β_k, l_k and f_{AP} are positive. The offloading decision $x_{\text{AP},k}$ is only zero if MU k is not offloading a fraction of its task to the AP. For μ_3^* this results in

$$\mu_3^* \begin{cases} = 0, & \text{if } x_{\text{AP},k} = 0,\ \forall k \\ > 0, & \text{else.} \end{cases} \tag{29}$$

The upper bound for (21) can be calculated by

$$\sum_{k=1}^{K} f_{\text{AP}} \cdot \frac{\sqrt{x_{\text{AP},k}p_{\text{static},k}\beta_k l_k}}{\sum_{k=1}^{K}\sqrt{x_{\text{AP},k}p_{\text{static},k}\beta_k l_k}} - f_{\text{AP}} = 0 \tag{30}$$

$$\sum_{k=1}^{K} f_{\text{AP}}\sqrt{x_{\text{AP},k}p_{\text{static},k}\beta_k l_k} - f_{\text{AP}}\sum_{k=1}^{K}\sqrt{x_{\text{AP},k}p_{\text{static},k}\beta_k l_k} = 0. \tag{31}$$

This result shows that if at least one MU offloads a fraction of its task to the AP, the resources will be allocated completely. If no MU offloads a fraction to the AP, the inequality in (22) reaches its minumum value, i.e.

$$\sum_{k=1}^{K} f_{\text{AP},k}^* - f_{\text{AP}} = \begin{cases} -f_{\text{AP}}, & \text{if } x_{\text{AP},k} = 0,\ \forall k \\ 0, & \text{else.} \end{cases} \tag{32}$$

Therefore, inequalities (21) and (22) are always fulfilled.

The orthogonality in (25) between the optimal Lagrangian multiplier μ_3^* and the constraint for the shared computation resources (21) has to be investigated. For the left multiplicand μ_3^*, result (29) can be used, and for the right multiplicand, result (32). Either μ_3^* is inactive when no MU offloads a fraction to the AP or the available computation frequency is fully allocated and constraint (21) becomes zero. Therefore, (25) is always zero and the orthogonality is fulfilled.