Joint Computation and Bandwidth Resources Allocation for Generalized Computing Model in Mobile Edge Computing

Qiaobin Kuang\textsuperscript{1}, Jie Gong\textsuperscript{2}, Xiang Chen\textsuperscript{1,a)}, and Xiao Ma\textsuperscript{2}

\textsuperscript{1}School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China.
\textsuperscript{2}School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China.
\textsuperscript{a)} chenxiang@mail.sysu.edu.cn

Abstract: Mobile edge computing (MEC) has restricted computation resource and limited bandwidth resource when serving multiple users simultaneously. In general, a user’s task can not be always parallelized. In this paper, we study a joint computation and bandwidth resource allocation problem, considering a generalized task computing model. The computation capacity is modeled as a non-decreasing concave function about allocated computation resource, and the joint computation and bandwidth resources allocation problem is solved by cvx opt. framework. Simulation results show that our proposed scheme is greatly superior to the serial processing benchmark scheme for Non-linear model. With non-linear model, higher computational multiplexing gain can be achieved, however, for linear model, there is no gain that can be obtained from the proposed scheme.

Keywords: Mobile edge computing, computation offloading, resource allocation

Classification: Wireless communication technologies

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, “A survey on mobile edge computing: The communication perspective,” IEEE Commun. Surveys Tuts., vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017. DOI:10.1109/COMST.2017.2745201

[2] P. Mach and Z. Becvar, “Mobile edge computing: A survey on architecture and computation offloading,” IEEE Commun. Surveys Tuts., vol. 19, no. 3, pp. 1628–1656, Mar. 2017. DOI:10.1109/COMST.2017.2682318

[3] Y. Mao, J. Zhang, and K. B. Letaief, “Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems,” in Proc. IEEE WCNC, San Francisco, CA, USA, Mar. 2017, pp. 1–6. DOI: 10.1109/WCNC.2017.7925615

[4] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, “Offloading in mobile edge computing: Task allocation and computational frequency scaling,” IEEE Trans. Commun., vol. 65, no. 8, pp. 3571–3584, Aug. 2017. DOI: 10.1109/TCOMM.2017.269660
1 Introduction

With the emergence of novel mobile applications such as face recognition, augmented reality technology and autonomous driving, the requirement for high-intensity computation and low latency is becoming more stringent. However, mobile users are often resource constrained due to their physical limitation. Mobile edge computing (MEC) which deploys computation resources at access points (APs) to offload mobile users’ computing tasks, is a promising paradigm to solve the problem, as it not only can provide sufficient computing capability at the mobile edge, but also shortens the response time compared with centralized clouds [1, 2]. A large number of previous papers in mobile edge computing have investigated computation offloading problems. For a single-user MEC system, [3, 4] presented task offloading schedule approaches. In multi-user case, it is crucial on how to jointly allocate radio and computation resources [5–7].

As seen in the existing works, computation time is not always inversely proportional to the allocated computation resource, as task may not be fully parallelized. In particular, some parts of task are so highly integrated or relatively that cannot be partitioned and some components’ outputs are the inputs of others [1]. Even if a large amount of parallel computation resources are allocated to one user, due to the partition limitation of the user’s task, some parallel computation resources are not utilize, that is, as the allocated computation resources increase, it does not mean that the computation time will decrease. To the best of our knowledge, there is no generalized model on the relation between computation time and computation resource.

In this work, we propose a new joint computation and bandwidth resources allocation approach for generalized computing model in an MEC system. Each user requests system bandwidth resources to transfer their respective tasks, and the edge server allocates certain computation resources to each user. Specifically, we model the generalized computing model as a
non-decreasing concave function to represent the relation between computation capacity and computation resource. We have jointly optimized the allocation of both computation and bandwidth resources to minimize the overall latency of the system. The simulation results show that the proposed method greatly reduces the latency of the MEC system compared to random serial processing (RSP) method for non-linear model. However, for linear computing model, the RSP method outperforms the joint computation and bandwidth resources allocation approach.

2 System Model

Consider an MEC system consisting of an AP and $K$ mobile users. The AP contains an MEC server and the set of mobile users is denoted by $\mathcal{K}$, with $|\mathcal{K}| = K$. All these users are equipped with a single antenna. The total available frequency band of the MEC system is $B$, and is orthogonally allocated to the mobile users.

Each mobile user $i \in \mathcal{K}$, would like to offload its computation-intensive task to the MEC server to reduce its own energy consumption and latency. Let a pair $(l_i, w_i)$ represents the corresponding information of the mobile user $i \in \mathcal{K}$, in which $l_i$ is the number of task input-bits at the mobile user $i$, and the execution of the task needs $w_i$ CPU cycles. In our setting, the AP serves as a central controller to decide how to assign the orthogonal frequency band and the CPU of the server for mobile users. The control signalling over-head is ignored for simplicity. The whole offloading process can be divided into the following steps.

2.0.1 Computation Offloading from mobile users to the AP

In this case each mobile user $i \in \mathcal{K}$ will occupy bandwidth $b_i$, to offload its task to the AP. The data rate of user $i$ can be denoted as

$$ R_i(b_i) = b_i \log_2 \left( 1 + \frac{h_i p_i}{N} \right), \tag{1} $$

where $p_i$ is the transmission power of user $i$ which is determined by the base station according to some power control algorithm, $h_i$ is the channel power gain between the user $i$ and the AP, and $N$ denotes the power of the additive white Gaussian noise. Therefore, the time for offloading the task of user $i$ can be calculated as

$$ T_i^{tran}(b_i) = \frac{l_i}{R_i(b_i)}, \forall i \in \mathcal{K}. \tag{2} $$

Furthermore, the sum bandwidth allocated to the users can not exceed the total budget, i.e.

$$ \sum_{i \in \mathcal{K}} b_i \leq B. \tag{3} $$

2.0.2 Computing at AP

In the MEC server in AP, a virtual machine is associated to user $i$ for its task computation. The total computation resource of all the virtual machines will not exceed the maximum computation resource of the system $F$. Let $f_i$,
denote the computation resource assigned to user \(i\), due to the nature of computation tasks, the computation capacity for user \(i\) is not always linear to the computation resource. Thus, we define a function \(g(f_i)\) to represent the relation between computation resource and capacity, where \(g(\cdot)\) can be any non-decreasing and concave function. Accordingly, we can obtain the time of computing at the MEC server for user \(i\), which can be denoted as

\[
T_{i}^{\text{comp}}(f_i) = \frac{w_i}{g(f_i)}, \forall i \in \mathcal{K}.
\]  

(4)

The total limitation of the computation resources of the AP is

\[
\sum_{i \in \mathcal{K}} f_i \leq F.
\]  

(5)

2.0.3 Sending back Computing result

After completing the task computing, the result is sent back to each user. Assume the result of the computation is much smaller than the input-bits. Therefore, the time to send back the result can be ignored.

According to the offloading process, the total latency for user \(i\) to complete its task can be expressed as

\[
T_i(b_i, f_i) = T_i^{\text{tran}}(b_i) + T_i^{\text{comp}}(f_i), \forall i \in \mathcal{K}.
\]  

(6)

Since AP computes the tasks of users in parallel, the process latency of the system is given by

\[
T = \max_{i \in \mathcal{K}} \alpha_i T_i(b_i, f_i),
\]  

(7)

where \(\alpha_i\) is the positive weight for user \(i\) to complete its task. Our target is to minimize the weighted latency of the system for users’ tasks execution, by efficiently optimizing the computation and bandwidth resources allocation of the MEC system. The decision variables include the bandwidth allocation \(b_i\) and the computation resource allocation \(f_i\). The problem is formulated as:

\[
\text{(P1)} : \min_{b_i, f_i} \max_{i \in \mathcal{K}} \alpha_i T_i(b_i, f_i)
\]  

s.t. \(b_i \geq 0, f_i \geq 0, \forall i \in \mathcal{K},
\]  

(3), (5),

(8)

where (3), (5) are the constraint of bandwidth and computation resource of the MEC system, respectively.

Note that problem (P1) is convex, and we solve it by standard convex optimization techniques such as the interior point method [8].

3 Numerical Results

In this part, the numerical results are presented to validate the performance of the proposed joint computation and bandwidth resources allocation design for generalized computing model. The linear model \(g_l(f) = \beta_l f\) denotes the tasks of users can be parallelized completely. Therefore, we use a natural logarithmic function \(g_{\log}(f) = \beta_{\log} \log(1 + rf)\) as the chosen non-linear
function to present the limited parallel processing, where $\beta_{log}$ and $r$ are the parameters of the function. Compared to the following benchmark scheme without such a joint design under two computation models.

- **random serial processing**: The scheme is to arbitrarily choose a user to transmit at first, and then transmit one of the remaining users’ tasks when the transmission of the previous user’ task is completed, and at the same time, the server is serving the previous user. In short, the channel and the server are available for only one user simultaneously. In this paper, we don’t consider the energy consumption. Therefore, in order to reduce the latency of the system, we choose the maximum data rate and computation capacity for those users. The maximum data rate for user $i$ is the same in both computing models, i.e., $R_{i,max} = B \log_2 \left(1 + \frac{h_{pi}}{N_0}\right)$. The maximum computation capacity for both linear computing resource (LCR) models and non-linear computing resource (nLCR) models are $g_{l,max} = \beta_l F$, $g_{log,max} = \beta_{log} \log(1+rF)$, respectively. We assume that the finish time of the first user for both transmission and computation for LCR model is $T_{l,1} = \frac{\alpha_{l} l_{1} R_{1,max}}{g_{l,max}} + \frac{\alpha_{1} w_{1}}{g_{l,max}}$. Accordingly, the finish time for the second user is $T_{l,2} = \max\{\frac{\alpha_{l} l_{1} R_{1,max}}{g_{l,max}} + \frac{\alpha_{2} l_{2} R_{2,max}}{g_{l,max}}, T_{l,1}\} + \frac{\alpha_{2} w_{2}}{g_{l,max}}$. In the end, we can get a recursion formula as $T_{l,k} = \max\{\sum_{i=1}^{k} \frac{\alpha_{i} l_{i} R_{i,max}}{g_{l,max}}, T_{l,k-1}\} + \frac{\alpha_{k} w_{k}}{g_{l,max}}$, which is the $k$ users’ system latency. Similarly, for nLCR models, the recursion formula can be obtained as $T_{log,k} = \max\{\sum_{i=1}^{k} \frac{\alpha_{i} l_{i} R_{i,max}}{g_{log,max}}, T_{log,k-1}\} + \frac{\alpha_{k} w_{k}}{g_{log,max}}$.

In this simulation, we assume the distance between the users and the server satisfies the discrete uniform distribution of interval $[120, 150]$ meters. The path-loss between users and the server is denoted as $\beta_0 (d/d_0)^{-\zeta}$, where $\beta_0 = -60$ dB corresponds to the path-loss at the reference distance of $d_0 = 10$ m, $d$ denotes the distance from the transmitter to the receiver, and $\zeta = 3$ is the path-loss exponent. Furthermore, we set $\alpha_{i} = 1$, $p_{i} = 26$ dBm, $\forall i \in K$, $\beta_l = 0.98$, $\beta_{log} = 20$, $r = 10^{3}$, $B = 40$ MHz, $N_0 = -100$ dBm, $F = 5$ GHz. The input-bits of the tasks $l_{i}$ is yield the uniform distribution of interval $(0, 1)$ Mbits, the required CPU cycles of users $w_{i}$ satisfy the discrete uniform distribution of interval $[1, 3] \times 10^{9}$ cycles. The reasonability of the selected parameters above is supported by the references [4] and [7]. Different values of the above parameters will influence the performance of the proposed model. Therefore, we choose the commonly used values in the literature of computation offloading in MEC to guarantee the well behaving of the model and the appropriateness to model an actual CPU.

Fig.1 shows the performance gains of the linear computing resource (LCR) models and non-linear computing resource (nLCR) models for different numbers of users, where the performance gain is expressed as $\eta_{l,gain} = 1 - \frac{T_{l}}{T_{l}'}$ and $\eta_{log,gain} = 1 - \frac{T_{log}}{T_{log}'}$ for linear and non-linear models, respectively. Denote $T_{l}$ and $T_{log}$ as the system latency of proposed scheme for LCR and nLCR models, respectively. $T_{l}'$ and $T_{log}'$ are the system latency of RSP. As shown in Fig.1, the nLCR model has a gain under any number of users, and with
the increasing of user number, the amplitude of the gain is increasing up to 1. This means that under the nLCR model, the user’s parallel processing capacity is limited, even if a user is allocated a lot of parallel computation resources, these computation resources can not be fully utilized due to the limitation of the user’s task partition. When the system serves users in parallel, these parallel computation resources can be more fully utilized, therefore, the proposed joint computation and bandwidth resources allocation scheme can achieve superior performance to RSP. Notice that the relation between the two times is essentially determined by the practical applications. For example, when focusing on the computation offloading problem as in our paper, the computation time is dominant. But when the communication time is dominant, our proposal still has equal performance compared to RSP.

As the number of users increases, however, the LCR model has no gain, meaning that the user’s task can be processed completely in parallel. In this case, simultaneously serving these users does not bring gain, and also wastes channel resources, because all users’ tasks must have been transmitted before they are computed, thus, the channels allocated are idle. For the LCR model, the channels are always occupied for RSP, thus, serial processing is more resource efficient than the proposed scheme. The gain of serial processing by transmission and computation over time multiplexing tends to be stable with the number of users increasing.

4 CONCLUSIONS
In this paper, we investigated a computation and bandwidth resource allocation scheme. Numerical results presented that, compared to the benchmark scheme, the proposed scheme can achieve remarkable gain for non-linear computing resource model. For linear computing resource model, the joint computation and bandwidth resource allocation scheme is inferior to random serial processing. How to combine resource optimization with more MEC servers and more users is worth further investigation in the future work.