Joint Proportional Task Offloading and Resource Allocation for MEC in Ultra-dense Networks with Improved Whale Optimization Algorithm

Yanling Zeng, Wen Chen*, Zhangbin Tang and Jiaxing Wu

College of Information Science & Technology, DongHua University, Shanghai, China
Corresponding author: chenwen@dhu.edu.cn

Abstract. Computationally intensive and latency sensitive applications put forward strict requirements on mobile user equipments (UEs) to provide the required computation capacity and delay. Mobile edge computing (MEC) in ultra-dense network (UDN) is considered as a promising solution that can reduce the computation limitations of UEs and extend their battery life by computation offloading. But the dense deployment of UEs, and the limited resources of small cell base stations (SBSs), which imposes more delays on computation offloading in MEC environment. In this paper, using the characteristics of UDN, we study the computation offloading management scheme based on the weighted sum of task completion time and energy consumption in MEC system. Then, we propose a computation offloading framework and formulate joint task offloading and resource allocation problem in UDNs to minimize overall cost consumption. Based on Genetic algorithm (GA) and Whale optimization algorithm (WOA), we adopt a heuristic algorithm named as GAWOA to solve this problem. Numerical results show that the algorithm has a fast convergence speed and better performance than other benchmark algorithms.

1. Introduction
Mobile user equipments (UEs), e.g., smartphones and tablets, are gaining huge popularity due to their growing functionality. Although the capabilities of UEs keep increasing, running computation-intensive tasks on UEs results in higher power consumption and longer latency [1]. However, the UEs usually have limited resources, such as limited battery energy and local CPU computing capability may suffer from a poor computing experience [2]. To solve these limitations, mobile edge computing (MEC) has recently become a new paradigm that allows data to be processed on the edge of the network, where is close to the UEs. Accordingly, researchers have tended to offload computing tasks to nearby base stations rather than applying remote cloud. By offloading computing tasks to nearby base station, performance qualities such as computation delay and energy consumption can be progressively reduced [3]. Due to the intensive deployment of small cell BSs, Ultra-dense network (UDN) have been considered as a promising paradigm to provide a huge access capacity to widely distributed UEs.

There have been quite a few studies on computation offloading problems. Next, the relevant work in these aspects is summarized: objective function (energy-efficient, delay-efficient and cost-efficient), computing mode (partial and full offloading). Starting from the energy-efficient MEC system, Dai et al. [4] applied user association with computation offloading to minimize energy consumption of users and MEC servers in a multi-task scenario. By jointly optimizing the computation offloading decision, spectrum, power and computing resource allocation, the energy-saving computation offloading management scheme in the MEC system was studied in [5]. Paper [6] proposed an edge network
architecture that enables edge nodes to collaborate, thus minimizing the total energy consumption with a finite tolerance delay. In [7], the author focused on task offloading and channel resource allocation based on MEC in 5G UDN to minimize the total energy consumption of the system while meeting the latency requirements of mobile users. In a delay-efficient scenario, Literature [8] developed an online SBS peer offloading framework by using the Lyapunov technology, so as to maximize the long-term system performance and keep the SBS’s energy consumption under a single long-term constraint. Reference [9] jointly optimized the offloading decision and computing resource allocation to minimize the average task duration in the case of limited device battery capacity. Considering a cost-efficient scenario, by considering the completion time and energy, the computational offloading problem of moving edge calculation is transformed into the system cost minimization problem in [10]. In order to maximize the benefits of task offloading for users in [11], the joint problem of computation offloading and resource allocation is studied. In [12], the author modeled the multi-user task offloading problem for mobile-edge cloud computing in a multi-channel wireless environment as a multi-user computation offloading game. The work [13] studied the task offloading problem in ultra-dense network, with the purpose of using the idea of software defining the network to save the battery life of user devices while minimizing the delay. In [14], the author considered edge cloud with limited computation power and sells it among users as a separable resource. The paper in [15] proposed joint load balancing and offloading, so that each vehicle can choose a VEC server and offload parts of its computation task for edge execution which is helpful to improve the utility of the system.

In this paper, we study the task offloading problem for MEC in UDNs, we consider a MEC system with multiple SBSSs and multiple UEs, and study the cost-efficient computation offloading problem by optimizing the offloading decision, the offloading ratio, channel bandwidth allocation, computation capacity allocation. Since it is difficult to solve joint task offloading and resource allocation optimization problems, a heuristic algorithm based on Whale optimization algorithm (WOA) is improved to solve the problem.

2. System Model

2.1. Network Model

We consider an ultra-dense networks (UDNs) consists of a macro base station (MBS), N small-cell base stations (SBSs), and U User Equipments (UEs). The CDSA (control-data separation architecture) model in this paper is an emerging model based on [16], as shown in Fig. 1. MBS provides wide area coverage and processing for most control signaling, while SBS provides the required data services. Each UE can receive signals from multiple SBSSs. The UEs and SBSSs are randomly distributed over the coverage of MBS, and each SBS is equipped with a MEC server. We denote the set of all UEs and the set of base stations as \( U = \{1, 2, \ldots, i, \ldots, U\} \) and \( N = \{1, 2, \ldots, j, \ldots, N\} \), respectively.

![Figure 1. System Model](image)

Each UEs has a delay-sensitive task that can be processed locally or offloaded to the SBS to process. Each of the task can be logically divided into several partitions (in bit) for computation offloading, and partitions can be processed locally or on SBS. When the task request from UEs is received, each SBS can process the task itself, or offload it to other SBSSs with additional computing
resource to process. The computation task of UE $i$ can be described by $Q_i = (d_i, f_i, T_i^{\text{max}})$. For task $Q_i$, $d_i$ denote the data for computation (in bits), (for example, the program codes or input parameters), regardless of the task is local computing or edge computing, we assume that $d_i$ remains the same, $f_i$ denotes the number of required CPU cycles for computing one bit of task $Q_i$, $T_i^{\text{max}}$ denotes the maximum latency of UE $i$ to accomplish the task $Q_i$.

In this paper, we consider parallel task offloading. In other words, tasks can be computed both locally and at base station. Here, we define the ratio of offloaded amount of bits to the total input data bits of UE $i$ as $\lambda_{i,j}(0 \leq \lambda_{i,j} \leq 1)$. UE $i$ offloads $\lambda_{i,j}d_i$ to the SBS and processed the rest $(1 - \lambda_{i,j})d_i$ locally. Then, the offloading decision of UE $i$ is denoted by $x_{i,j} \in \{0, 1\}$, when $x_{i,j} = 1$, the task of UE $i$ is offloaded to SBS $j$, otherwise, the task of UE $i$ is processed locally.

2.2. Communication Model

If UE $i$ send request to SBS $j$, the spectral efficiency of the uplink between UE $i$ and SBS $j$ can be described by Shannon capacity $R_{i,j}$, where $\gamma_{i,j}$ is the uplink signal to noise ratio (SNR) of UE $i$. $\gamma_{i,j}$ can be expressed as $\gamma_{i,j} = \frac{p_i h_{i,j}}{\sigma^2}$, where $p_i$ is the transmission power of UE $i$, $h_{i,j}$ is the channel gain between UE $i$ and the SBS $j$, $\sigma^2$ is the power of the additive white Gaussian noise. For each UE, the received signal intensity should be considered due to the distance between the base station and UE. Therefore, the received SNR should be greater than a threshold.

The bandwidth of SBS $j$ is denoted as $B_j$ (in Hz). For UE $i$ offloading to SBS $j$, SBS $j$ allocates certain bandwidth resources $B_{i,j}$ to UE $i$, so the uplink data rate of UE $i$ connected to SBS $j$ can be expressed as

$$R_{i,j} = B_{i,j} \cdot \eta_{i,j}$$ (1)

When $\lambda_{i,j}d_i$ bits offloaded to the SBS $j$, then the transmission delay and energy consumption for UE $i$ offloading the task $Q_i$ to the SBS $j$ can be expressed as the following, respectively

$$T_{i,j}^{\text{comm}} = \frac{\lambda_{i,j}d_i}{R_{i,j}^{\text{comm}}} = \frac{\lambda_{i,j}d_i}{B_{i,j}^{\text{comm}} \eta_{i,j}}$$ (2)

$$E_{i,j}^{\text{comm}} = P_i^{\text{comm}} T_{i,j}^{\text{comm}} = \frac{P_i^{\text{comm}} \lambda_{i,j}d_i}{B_{i,j}^{\text{comm}} \eta_{i,j}}$$ (3)

where $P_i^{\text{comm}}$ is the total power consumption (in watt) of UE $i$ for transmission.

2.3. Computation Model

1) Local Computing. When the computation tasks of the UE are assigned to executed locally, only the UE’s local CPU works. For local computing, we denote $F_i^L$ as the local computation capacity (i.e., CPU cycles per second) of UE $i$ when processing task $Q_i$. Thus, when $(1 - \lambda_{i,j})d_i$ data bits are processed at the UE $i$, the local computation delay can be calculated as

$$T_i^L = \frac{(1-\lambda_{i,j})d_i}{F_i^L}$$ (4)

Given the $P_i^L$ as the local computation power consumption (in watt) of UE $i$. Also, the local energy consumption can be expressed as

$$E_i^L = P_i^L T_i^L = \frac{P_i^L (1-\lambda_{i,j})d_i}{F_i^L}$$ (5)

2) Edge Computing. For edge computing, UE $i$ offloads $\lambda_{i,j} \cdot d_i$ part of task $Q_i$ to SBS $j$. If SBS $j$ has sufficient computation capacity, it will process this part, otherwise $\lambda_{i,j} \cdot d_i$ will be further offloaded to other SBS to process [9] [14]. The total computation capacity of the SBS $j$ is denoted as $F_j$ (i.e., CPU cycles per second). For UE $i$ offloading to SBS $j$, SBS $j$ allocates certain computation resources $F_{i,j}$ to UE $i$. Thus, the computation delay of UE $i$ offloaded to SBS $j$ is calculated as
When UE $i$ offloads the tasks to SBS, UE $i$ first according to the remaining computing capacity and bandwidth of each SBS in the candidate SBS set, select the SBS that can meet the time delay and energy consumption requirements of UE. When the UE completes the task offloading, the occupied resource is released and the SBS set is updated. Thus, when SBS $j$ finally execute the task, the total delay includes the communication delay between the UE $i$ and the SBS $j$, and the computing delay of the SBS $j$. Then the total delay of task $Q_i$ processed by the SBS $j$ can be expressed by

$$T_{i,j}^{off} = T_{i,j}^{comm} + T_{i,j}^{comp}$$

As mentioned in [5], we ignored the computing energy consumption, for the SBSs can have rich energy. Thus, the total energy consumption of UE $i$ can be given by

$$E_{i,j}^{off} = E_{i,j}^{comm}$$

According to equation (1)-(8), the computation delay and energy consumption of UE $i$ are given by, respectively

$$T_i = (1 - \sum_{j=1}^{N} x_{i,j})T_i^{L} + \sum_{j=1}^{N} x_{i,j}T_{i,j}^{off}$$
$$E_i = (1 - \sum_{j=1}^{N} x_{i,j})E_i^{L} + \sum_{j=1}^{N} x_{i,j}E_{i,j}^{off}$$

In the mobile edge computing environment, there exist some delay-sensitive terminal applications and some energy-sensitive IoT devices, combined with the computation delay and energy consumption for completing the task of UE $i$ executed by SBS $j$, using $\alpha$,$(1-\alpha)$ to represent the weighted parameter of time delay and energy consumption respectively, the total cost of the task for completing the task of UE $i$ is expressed as:

$$C_i = \alpha T_i + (1 - \alpha)E_i$$

3. Problem Formulation and Analysis

In this chapter, we consider both the application experience and energy consumption saving of the UEs, and design a computing offloading scheme for multi-terminal device tasks, which involves computation resources, offloading decision-making, offloading ratio, channel bandwidth. The objective of this paper is to minimize the cost consumption combining computation delay and energy consumption of all UEs. The battery capacity of UE $i$ is denoted as $E_i^{max}$. Therefore, the optimization objective is expressed as:

$$C = \sum_{i=1}^{U} C_i$$

Therefore, the main problem addressed in this paper is

$$P_0 \text{ min} \begin{array}{c} C \\ x,\lambda,F,B \end{array}$$

s.t. $$T_i \leq T_i^{max}, i \in U$$
$$E_i \leq E_i^{max}, i \in U$$
$$0 \leq \sum_{i=1}^{N} \lambda_{i,j} \leq 1, i \in U, j \in N$$
$$\sum_{i=1}^{N} x_{i,j} \leq 1, i \in U$$
5. Numerical Simulation

In this section, we first evaluate the performance of the adopted algorithm with the other baseline algorithms, then verify the effectiveness of the offloading decision. We consider an MEC system and the coverage radius of SBS is 20m. The number of UEs are 100, the number of SBSs are 10, data size to offload is assumed to be distributed in [0.1-0.5] MB, CPU cycles of per bit of offloading task is assumed to be distributed in [500-1000] cycles/bit, local computational capacity is 500 MHz, computational capacity of SBSs is 80GHz, local computational power consumption is 0.5 W, transmission power consumption of UE is 0.3 W.
We compare GAWOA with the genetic algorithm GA, and whale optimization algorithm WOA in order to evaluate the performance. We compare the GAWOA algorithm with local computing only: all computing tasks are executed locally, no task offloaded, and offloading only: all computing tasks are offloaded to the base station and not executed locally, in order to verify the effectiveness of the offloading decision.

Simulation of cost consumption is computed according to Equation (11) and it is illustrated in Fig.2, the cost consumption of the GAWOA algorithm is verified when compared to GA and WOA algorithms. As we can observed from Fig.2, GAWOA gets the best cost consumption, while WOA keeps the highest cost consumption. The cost consumption of GA is between WOA and GA. That is because GAWOA combines the advantages of GA, which is better at searching globally and WOA, which is better at the searching broadly. Hence, GAWOA has a better performance than GA and WOA.

![Figure 2. The convergence of three Algorithms](image_url)

![Figure 3. Cost of different channel bandwidths](image_url)

As we can seen from Fig.3, increased the system bandwidth provides UEs with a greater chance to reduce the offloading cost and select to offload. When the channel bandwidth is low, it is inefficient to offload all computing tasks. This is because the channel bandwidth decreases will result the offloading time increases. When the system bandwidth is bigger than 8MHz, the offloading only scheme is more beneficial than the local only scheme. We can also observed that GAWOA obtains dramatically lower cost consumption compared to the other two algorithms. When the system bandwidth reaches 10MHz, the cost consumption of the offloading only scheme is as low as that of the GAWOA algorithm. This is reasonable when the system bandwidth increase, the offloading time will decrease.

Fig.4 illustrates the cost consumption achieved by other two algorithms versus the number of UEs from 60 to 100. As indicated in Fig.4, GAWOA can achieve relatively low computational cost compared to the two baseline algorithms. This is because in our proposed framework, offloading decisions, bandwidth, and computation resources are jointly optimized, so computation offloading is beneficial to the offloading scheme. It can also be seen from Fig.4 that when UE is large enough, the local only scheme will incur higher computation cost than the offloading only scheme. This is due to UE’s limited computing resources, which result in a large delay. When UE is small, GAWOA have the same performance as offloading only scheme because it has sufficient computing resources.

Fig.5 show the cost consumption under different weighted parameters from 0 to 1. The offloading result shows that the weighted parameters have a certain effect on the offloading decisions of UEs and then affect the system performance. The cost consumption increases as the weighted parameter α increases, this is due to the fact that UE is more concerned with computation delay and have more strict requirements for delay. When more UE select to offloading, the competition among UEs for the limited computation resource at the SBS which result in higher delay and further affect the system cost consumption.
6. Conclusion

In this paper, we study a cost-efficient task offloading problem in the MEC system with UDNs, taking computation resource allocation, channel bandwidth allocation, offloading decision-making and offloading ratio into account. First, we describe the computation offloading model, and the objective function is formulated. Second, we combine GA and WOA and design an algorithm named GAWOA to solve this problem. Then the performance and convergence of GAWOA are analyzed. Finally, the effectiveness of this algorithm in improving the system performance is studied by simulation.

7. References

[1] D Liu, A Hafid, and L Khoukhi 2018 *IEEE Global Communications Conf.* (ABU Dhabi: IEEE Globecom) pp 1–6
[2] J Liu, Y Mao, J Zhang, and K B Letaief 2016 *IEEE Int. Symp. on Information Theory* (Barcelona: ISIT) pp 1451–55
[3] X Tao, K Ota, M Dong, H Qi, and K Li 2017 *IEEE Wireless Communications Letters*. 6 774–7
[4] Y Dai, D Xu, S Maharjan, and Y Zhang 2018 *IEEE Transactions on Vehicular Technology*. 67 12313–25
[5] F Guo, H Zhang, H Ji, X Li, and V C M Leung 2018 *IEEE/ACM Transactions on Networking*. 26 2651–64
[6] T T Vu, N V Huynh, D T Hoang, D N Nguyen, and E Dukkiewicz 2018 *IEEE Global Communications Conference* (ABU Dhabi: IEEE Globecom) 1–6
[7] X Chen, Z Liu, Y Chen, and Z Li 2019 *IEEE Access*. 7 184172–82
[8] L Chen, S Zhou, and J Xu 2018 *IEEE/ACM Transactions on Networking*. 26 1619–32
[9] Y Wang, X Tao, X Zhang, P Zhang, and Y T Hou 2019 *IEEE Transactions on Vehicular Technology*. 68 2763–2776
[10] H Yu, Q Wang, and S Guo 2018 *IEEE Int. Conf. on Networking* (Calabria: ICNSC) pp 1–4
[11] T X Tran and D Pompili 2019 *IEEE Transactions on Vehicular Technology*. 68 856–68
[12] X Chen, L Jiao, W Li, and X Fu 2016 *IEEE/ACM Transactions on Networking*. 24 2795–2808
[13] M Chen and Y Hao 2018 *IEEE Journal on Selected Areas in Communications*. 36 587–97
[14] M Liu and Y Liu 2018 *IEEE Wireless Communications Letters*. 7 420–3
[15] Y Dai, D Xu, S Maharjan, and Y Zhang 2019 *IEEE Internet of Things Journal*. 6 4377–87
[16] W Bai, Y Li, T Yao, and H Zhang 2018 *IEEE Access*. 6 60891–9