Toward Optimal Resource Allocation for Task Offloading in Mobile Edge Computing

Wenzao Li[1,2(✉)], Yuwen Pan[1], Fangxing Wang[2], Lei Zhang[3], and Jiangchuan Liu[2]

[1] College of Communication Engineering, Chengdu University of Information Technology, Chengdu, China
lwz@cuit.edu.cn, yuwenpan@163.com

[2] School of Computing Science, Simon Fraser University, Burnaby, Canada
{fangxinw,jcliu}@sfu.ca

[3] College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
leizhang@szu.edu.cn

Abstract. Task offloading emerges as a promising solution in Mobile Edge Computing (MEC) scenarios to not only incorporate more processing capability but also save energy. There however exists a key conflict between the heavy processing workloads of terminals and the limited wireless bandwidth, making it challenging to determine the computing placement at the terminals or the remote servers. In this paper, we aim to migrate the most suitable offloading tasks to fully obtain the benefits from the resourceful cloud. The problem in this task offloading scenario is modeled as an optimization problem. Therefore, a Genetic Algorithm is then proposed to achieve maximal user selection and the most valuable task offloading. Specifically, the cloud is pondered to provide computing services for as many edge wireless terminals as possible under the limited wireless channels. The base stations (BSs) serve as the edge for task coordination. The tasks are jointly considered to minimize the computing overhead and energy consumption, where the cost model of local devices is used as one of the optimization objectives in this wireless mobile selective schedule. We also establish the multi-devices task offloading scenario to further verify the efficiency of the proposed allocating schedule. Our extensive numerical experiments demonstrate that our allocating scheme can effectively take advantage of the cloud server and reduce the cost of end users.

Keywords: Mobile edge computing · Task offloading · Genetic algorithm · Computing overhead · Allocating schedule

1 Introduction

The recent success of Internet-of-Things (IoT) [11,12,17] facilitates an explosive increase of mobile devices as well as computing tasks. It is reported that

X. Chu et al. (Eds.): QShine 2019, LNICST 300, pp. 50–62, 2020.
https://doi.org/10.1007/978-3-030-38819-5_4

more than 7 billion resource-limited devices are connected in the Internet of Things (IoT) in 2018 [1]. These interconnected devices further integrate as an intelligent information system and call for more smart applications [18]. Mobile devices are usually assigned with a variety of computing tasks for processing, while they mostly suffer from the constrained power supply and the limited processing capabilities [22]. Edge-cloud computing provides a promising opportunity by offloading the computing tasks of mobile devices to nearby servers to reduce the computation cost and save energy [14]. The limitation of shared wireless bandwidth however restricts the entire task offloading, only allowing a portion of tasks to move to the cloud servers [16]. The task offloading scheme can effectively reduce the computational burden for end devices. However, how to select an offloading task will affect the computing energy, the number of tasks, and bandwidth utilization. Generally, given the diverse optimization objectives, massive tasks and multiple constraints, such a resource allocation problem is quite complicated. To this end, we introduce a genetic algorithm that considers both the channel resource allocation and task cost to address this problem.

A 5G base station (BS) usually owns hundreds of ports, which can support to send and receive signals from many users at once on the same frequency [15]. The high density of users in the covered area of a BS will result in limited data processing given the limited wireless bandwidth. Then, we maximize the number of computation offloading tasks as well as tasks with a high cost in a period. We thus propose a genetic algorithm, named computation offloading with Genetic Algorithm (COGA), to solve the selected tasks of computation offloading in MEC. It can achieve superior task selection, computing cost, and energy efficiency during a period.

The contribution of this paper is summarized as follows. The paper models the computation offloading problem in the MEC scenario as an optimization problem. Energy consumption, computing latency and the number of tasks are jointly considered in our design purpose. Therefore, we propose a genetic algorithm COGA, which is based on a unified objective function to optimize multiple targets for concerned issues. We further propose Enhanced COGA, which considers the features of the computation offloading scenario to achieve smart offloading and energy saving. Numerical simulations demonstrate the superiority of our solution.

The rest of the paper is organized as follows. We present the computation offloading model in Sect. 2. Section 3 introduces the system model. Then, we propose a genetic algorithm introduced in Sect. 4. In Sect. 5, we present the numerical experiments and discuss the results to evaluate the performance. We finally discuss the related work in Sect. 6, and conclude paper in Sect. 7.

2 Proposed Computation Offloading System Model

The computation offloading is a complex problem, and the considered factors are inconsistent in different system models [5,13]. Given that those devices are all connected to a BS, carrying out the offloading strategy at a BS is a simple way

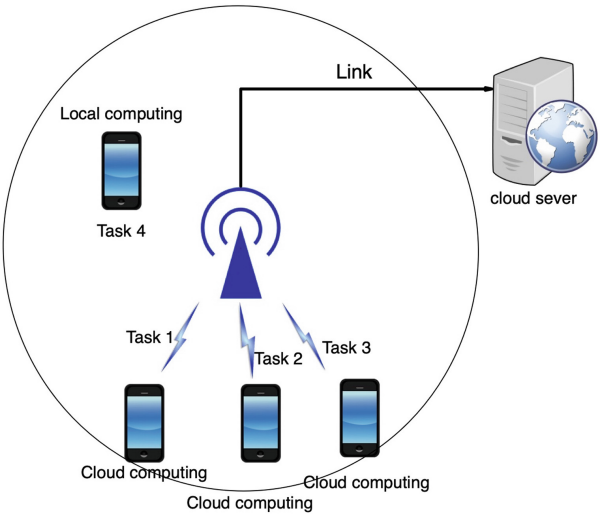

Fig. 1. The task offloading in multi-user MEC system

to reduce the terminal cost. We also consider the set $\mathcal{U} = \{u_1, u_2, ..., u_i, ..., u_{N_t}\}$ as the covered terminals by a BS. And these devices belong to a BS $\mathcal{B}$, which provides the wireless channel to the set $\mathcal{U}$. We assume that the set $\mathcal{U}$ will not be changed during the offloading period $\mathcal{T}$ and each device has one task offloading requirement. Generally, the number of devices and channel gain may be changed due to the mobility of users covered by the base station. Each device in the scenario owns more than one task for computation offloading and it will be further discussed in future works. Therefore, the task set $\mathcal{T}_{task} = \{L_1, L_2, ..., L_i, ..., L_{N_t}\}$, where $L_i \in u_i$. Each covered device only has one task in the model and the task of device is required to offload to the near cloud by centralized control of the BS. As shown in Fig. 1, the task offloading scenario has four important parts, the offloading tasks, the wireless channel, the local users and the remote cloud.

The communication mode, mobile device and cloud play as pivotal roles in the MEC. These models are introduced in detail as follows.

2.1 Communication Model

In the OFDMA communication system, the total bandwidth is partitioned into several sub-channels. The OFDMA-based cellular network can adopt the full-duplex (FD) ratio technology and FD based BS supports multiple half-duplex (HD) users [6]. The numerous subcarriers of each sub-channel can be assigned to a user in a centralized model. If the $g(i)$ represents the channel gain for mobile i, the channel bandwidth is B, and the transmission power is S_U. We unified use N_0 presents the average noise power, it is usually considered as Gaussian channel noise. Then, the maximum transmission bit rate C_U can be calculated by equation

$$C_U = B * log_2(1 + \frac{g_U S_U}{N_0}), \tag{1}$$

where the C_U represents the maximum data rate. It means the maximum data transfer rate can be provided to BS covered users. But it is restrained by some specific parameters of BS. In actual situations, the actual data transfer rate $C_S(U)$ is very below the theoretical value of C_U. Although there is data conflicting in the same spectrum, it can be improved by the physic layer channel access schedule such as CS-MUD and SCMA [2,7]. From the Eq. (1), the limited transfer data rate may not satisfy the ultra dense devices to offload to the cloud. For the reason of task processing efficiency, the more tasks benefited from the remote cloud, the better in MEC scenarios.

2.2 Mobile Device Computation Model in Task Computation Offloading

The energy consumption and computing time in local device are frequently discussed [5,20], we assume that device i has only one task $L_i \triangleq (da_i, f_i^l)$, the f_i^l denotes the CPU cycles of mobile devices per second. Then the local computing time can described as

$$t_i^l = \frac{d_i}{f_i^l}, \tag{2}$$

where the t_i^l can also be understood as execution time or the execution cost of a terminal user. Then, the energy consumption can be described as

$$e_i^l = \mathcal{J}_i d_i, \tag{3}$$

where the $\mathcal{J}_i$ denotes the coefficient consumed energy per CPU cycle. If we transform this problem as the energy consumed per bit of CPU processing, we can describe the energy consumption as

$$\eta_i^l = \frac{\mathcal{P}_i t_i}{\mathcal{D}_i}, \tag{4}$$

where the $\mathcal{P}_i$ represents the CPU power consumption. The t_i denotes the processing time of task L_i and the $\mathcal{D}_i$ represents the number of processing data for the task L_i. From the (3) and (4), the processed amount of data or the number of CPU cycles for the task are proportional to energy consumption. Note that the CPU keeps the computing frequency in the processing period and it is difficult to accurately calculate the energy consumption per cycle due to the complicated work model of a CPU. Hence, the $\mathcal{J}_i$ can be understood as a coefficient of energy computing. After we establish the computing time and energy consumption model with (2) and (3), then the task cost of local user u_i can be defined as

$$\mathcal{C}_i^l = \lambda_t t_i^l + \lambda_e e_i^l, \tag{5}$$

where the λ_t and λ_e denote the weight parameters, which influence the optimization target for the concerned network indicators. If the system cares about the energy consumption, then it can set $\lambda_t < \lambda_e$ and the $\lambda_t, \lambda_e \in [0, 1]$. It is the common processing way in the weight method.

2.3 Cloud Computing Model

The remote cloud is to consider to have sufficiency computing ability, and the computing energy has no constraint due to the power supply. Some studies focus on the overall operation reaching a reasonable balanced state. For instance, the task offloading need to satisfy $\mathcal{C}_i^l < \mathcal{C}_i^c$, where the $\mathcal{C}_i^c$ denotes the task computing cost in the cloud [5]. Some other studies do not only consider the computing capability of the remote cloud but also considers the capacity of the wireless channel. As shown in Subsect. 2.1, the channel capacity is limited by C_U and there are $\mathcal{M}$ sub-channels. Their respective bandwidths make up the available bandwidth B. To get the benefits of the cloud servers through the limited bandwidth which is a challenging question.

3 Distributed Computation Offloading in a BS

The proposed computation offloading has been discussed in this section. As represented in the prior section, a sub-channel is severed to user i. Therefore, the channel capacity can reach the maximum data rate as follow:

$$C_{\bar{s}}(i) = \lambda_\gamma \bar{B} * log_2(1 + \frac{g_i S_i}{N_0}). \tag{6}$$

The $\bar{B}$ represents the bandwidth of a sub-channel and λ_γ can be understood as channel utilization. Then we can formulated the channel data rate C_U as $C_U \geq \sum_{m=1}^{n} L_m(m) C_{\bar{s}}(m)$, where m denotes the determined offloading tasks. So the resource allocation problem under the constraints on the channel data rate can be formulated as follows:

$$\begin{aligned} \mathcal{Z}_{\mathcal{B}_m, \mathcal{B}_c} &= \begin{pmatrix} \max\limits_{\mathcal{B}_m} f(\mathcal{B}_m) \\ \max\limits_{\mathcal{B}_c} f(\mathcal{B}_c) \end{pmatrix} = \begin{pmatrix} \max\limits_{\mathcal{B}_m} \sum_{m=1}^{n} L_i(m) * C_{\bar{s}}(m) \\ \max\limits_{\mathcal{B}_c} \sum_{m=1}^{n} L_i(m) * \mathcal{C}_i^l(i) \end{pmatrix} \\ & s.t. \quad C_U \geq \sum_{m=1}^{n} L_m(m) C_{\bar{s}}(m) \\ & L_i \leq N_t, i \in N_t, C_i \leq C_{\bar{s}}(i) \end{aligned} \tag{7}$$

The Eq. (7) describes the demand for task offloading, the purpose of the scenario requires that maximum tasks need to offload to the cloud, but it should face the limited bandwidth. And the compute offloading still needs to consider the energy efficiency $\sum_{i=1}^{m} e_i^l$ and the compute time $t \sum_{i=1}^{m} t_i^l$. In the model, the cloud computing capabilities are considered to be sufficient. Both the energy

cost and computing time require the maximum value to reduce the burden of end-users. Obviously, this is a multi-objective optimization problem. And unfortunately, it is extremely challenging to obtain an optimal solution.

Then we need to make the decision of computation offloading. It can be considered that, which tasks should be selected to offloading. Similarly, the selected tasks should have the maximum cost. This strategy will minimize the overall burden of the terminals.

4 Task Computation Offloading Decision Based on Genetic Algorithm

In this section, the proposed optimized problem is formulated as the Eq. (7), and the offloading schedule is designed with a heuristic search method. To solve this challenging problem, this genetic algorithm focuses on closing the optimal performance.

4.1 Initialization Model

First, a matrix In is given to express the decision of computation offloading. The number of rows in the matrix represents the number of offloading decision combinations.

$$In = \begin{bmatrix} \delta_{1,1} & \delta_{1,2} & \dots & \delta_{1,N_t} \\ \delta_{2,1} & \delta_{2,2} & \dots & \delta_{2,N_t} \\ \dots & \dots & \dots & \dots \\ \delta_{m,1} & \delta_{m,2} & \dots & \delta_{m,N_t} \end{bmatrix}, \tag{8}$$

where the δ_{ij} means the task number j, the each matrix row represents the offloading task order form set $\mathcal{T}_{task}$. The matrix is established by random way at the beginning, each task index is unique and $\delta_{ij} \in [1, N_t]$. Besides, the initial matrix is executed in each period t_p. The size of the row is $n = N_t$, which denotes the number of pending offloading tasks in the BS. And the fitness function $\psi_f(\bar{N}_t, \mathcal{C}_i^l)$ need to be given according to the optimized object

$$\psi_f(\bar{N}_t, \mathcal{C}_i^l) = \lambda_{\bar{N}_t}^f M(\bar{N}_t) + \lambda_{\mathcal{C}}^f M(\mathcal{C}_i^l), \tag{9}$$

where the $\lambda_{\bar{N}_t}^f$ and $\lambda_{\mathcal{C}}^f$ denotes the coefficient separately. The function of $M()$ is a mapping function, which can solve the problem of adding non-similar physical dimensions. They are mapped in $[0, \delta]$, then it can be compared within a quantified range. And the $\bar{N}_t$ represents the determined offloading tasks, which are taken values from the matrix row $\delta_{i,\dots}$. Additional, the $\mathcal{C}_i^l$ means the cost value of determined offloading tasks, which is described as the Eq. (5).

4.2 The Selection Processing of COGA

After the COGA establishes the matrix with random values. Then, the COGA will consist of four phases of operation: roulette algorithm, elite retention strategy, cross operation, and mutation operation.

Roulette Algorithm (RA): In RA, there are three steps for matrix reorganization. First, the fitness value of matrix In should be calculated as the set $f^v(i = 1, 2, 3..., m)$. Second, the survival probability $p_r(r_i)$ is calculated by

$$p_r(r_i) = \frac{f^v(r_i)}{\sum_{j=1}^{m} f^v(r_j)}. \tag{10}$$

Third, $p_r(r_i)$ is used to establish the array, then the m times selecting operations will establish a new matrix In_r.

Elite Retention Strategy (ERS): The ERS just selected a maximum fitness value of a row in In_r, the *best* row is kept as the $m+1$ row. Thus, a new matrix In_e is established.

Cross Operation (CO): COGA randomly selects two rows of In_e (except the $m+1$ row) for *crossing operation*. And it generates an index p, and $p \in \{p \in N | 1 \leq p \leq N_t\}$, then the two rows will change the values before $\delta_{p,...}$ between the two rows with distance one or two position. In addition, a crossing probability P_c, which is introduced in [3], can be described as

$$P_c = \begin{cases} max(P_c) - \frac{max(P_c) - min(P_c) * (\alpha - \beta)}{\delta' - \beta}, \alpha > \beta, \\ max(P_c) \qquad\qquad\qquad\qquad\quad , \alpha \leq \beta \end{cases} \tag{11}$$

where $max(P_c)$ denote the maximum cross probability, and the $min(P_c)$ means the minimum cross probability. α represents the maximum fitness value of the two selected rows (the two orders of offloading decisions). β denotes the average of calculated fitness value for the whole matrix, and the δ' denotes the maximum calculated fitness value for the matrix In_e. If a randomly variable seed $S_d < P_c$, after CO, the new matrix In_c is established. It should be noted here that the task index should be unique in each row after CO. It can be achieved by traversing the task index.

Mutation Operation (MO): After the CO, we can get the new matrix In_c. The COGA operates the MO for each row of matrix In_c. First of all, the mutation probability P_m can be described as [3]

$$P_m = \begin{cases} max(P_m) - \frac{max(P_m) - min(P_m) * (\alpha - \beta)}{\delta' - \beta}, \alpha > \beta, \\ max(P_m) \qquad\qquad\qquad\qquad\quad , \alpha \leq \beta \end{cases} \tag{12}$$

The variables is similar to the Eq. (11). The $max(P_m)$ represents the maximum mutation probability and the $min(P_m$ represents the minimum mutation probability. The $max(P_c)$, $min(P_c)$ is set as 0.9% and 0.6%, respectively. At the same time, $max(P_m)$, $min(P_m$ is set as 0.9 percent and 0.6%, respectively. Each row of in_m will process the MO by the probability P_m. If a row is determined to be mutated, randomly swap the positions of the two index in the row. Such a MO iterates through the first m rows of the entire matrix In_c.

After the MO, the row with the lowest fitness value will be deleted. Then, the new matrix In_m becomes $m * N_t$. The COGA will iterate until the fitness value is stable.

In the computation offloading scenario, the number of tasks may be relatively high in a BS covered area. The searching process will be lengthy. Therefore, the CO stage is redesigned to strengthen the dramatic changes in each row. In the CO stage of COGA, the cross operation just runs at once and it is regardless of the number of initial tasks. Hence, the CO times $\phi(m)$ is designed as

$$\phi(m) = \frac{m}{\lambda_b + \lambda_g \delta}. \tag{13}$$

where the λ_b and the λ_g are constant coefficients, the value of $\phi(m)$ is positively related to m. The δ represents a random seed. Then, the more dramatic CO leads to less convergence consumption times.

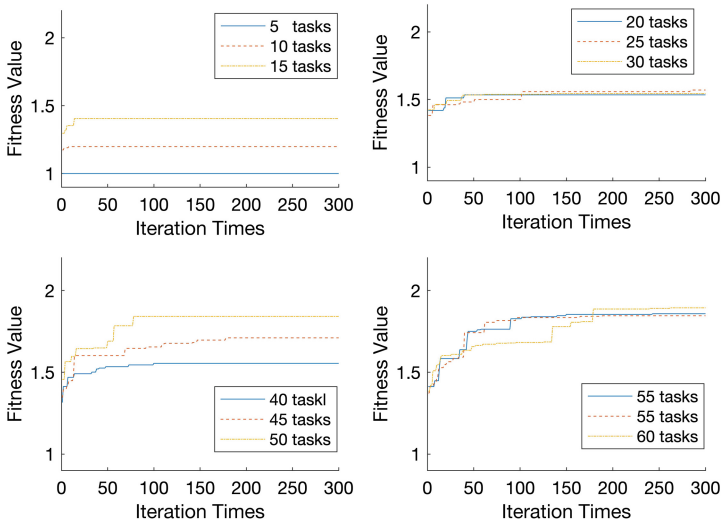

Fig. 2. Algorithm convergence under different task numbers

5 Simulation Results

We completed the proposed scenario and the COGA algorithm. Then, several simulation results are presented for evaluating the performance of the COGA strategy. In actual situations, mobile computing tasks vary in data size. The channel capacity of BS is also related to its own model. From the design principle of the algorithm, the size, number of tasks and the assumption of BS bandwidth do not affect the effectiveness of the proposed algorithm.

There are up to 60 mobile devices in a BS covered area, the channel gain is not considered in this scenario due to the principle of algorithm. The channel bandwidth $C_{\bar{U}} = 5\,\text{MHz}$, the transmission power $S_i = 100\,\text{mw}$ and the noise

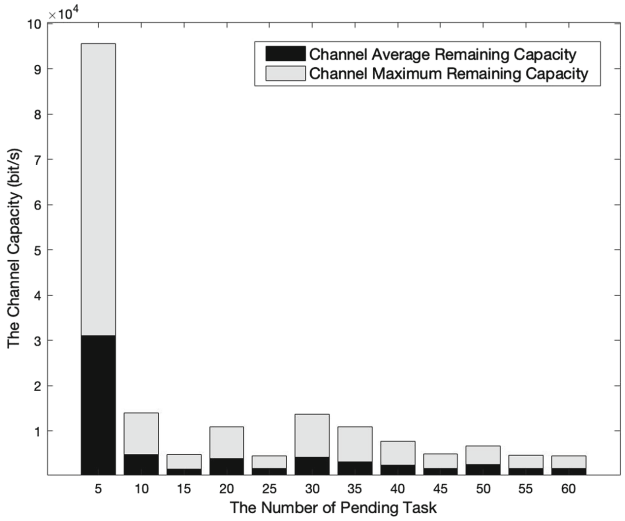

Fig. 3. The comparison of channel average remaining capacity and maximum remaining capacity

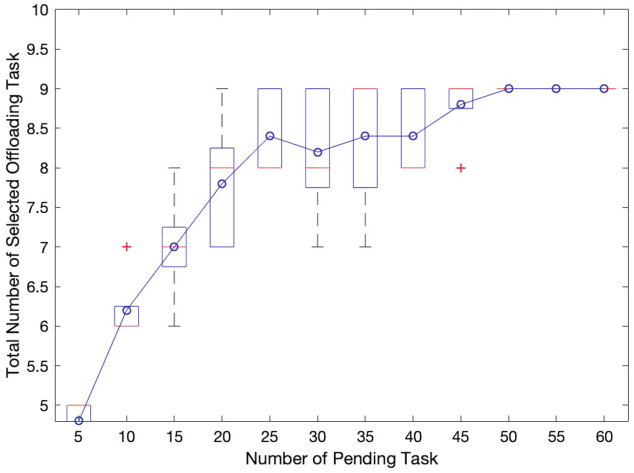

Fig. 4. The number of offloading task in different task density

$N_0 = -100\,\text{dbm}$ [5]. In this scenario, each device owns a task for offloading computing requirements, the $\mathcal{J}_i = 8.9 * 10^{-12}$ J/cycle. The amount of offloading data size is randomly generated in the range of $(0.125{*}10^5, 0.175{*}10^5)$ bit/s. The coefficient λ_t and λ_e are set as 0.3, 0.7, separately. Since the task data is randomly generated, the COGA runs to select offloading tasks. To verify the validity of the method, each experiment is carried out five times. The performance should be tracked at four aspects in different task density: the convergence status, the

remaining capacity of the channel, the offloading tasks and offloading cost of proposed GA algorithms.

To observe the convergence of the COGA, the fitness calculation process of twelve different number of task offloading scenarios are shown in Fig. 2. All of them can effective converge after a number of iterations. Besides, the more tasks have required the computation offloading, the more iterations are needed.

Figure 3 displays the average remaining capacity and maximum remaining capacity in different task density. When the limited channel capacity is calculated for 5 tasks, there is more capacity left due to the small data offloading requirements. From the results, the remaining capacity waves in a small range when the number of tasks increases. In the scene of intensive tasks, the remaining space is very small, therefore, it shows the effectiveness of residual capability control. If the channel capacity that can be used for task offloading is too small and the rate of all pending tasks are big sizes, then the remaining channel capacity will not change too much. We also give the results of the number of offloading tasks in different task density with the box figure.

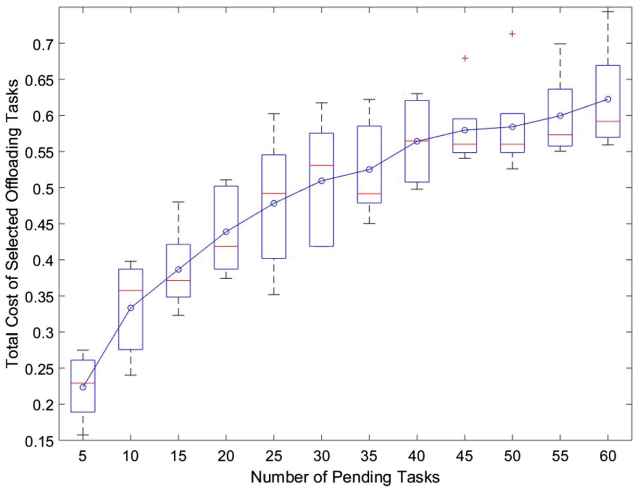

Fig. 5. The cost of offloading task in different task density

From the Fig. 4, the top horizontal line and the bottom horizontal line represents the maximum and minimum of 5 time's result, separately. The cycle denotes the average value, and the middle horizontal line means the medium number of tasks. The results are stable in the relatively intensive task scenes. It indicates that it near the best-optimized result in the current bandwidth situation. Similarly to the results in Fig. 4, the cost value of the proposed cost model also maintains a relatively high level in intensive task scenes. It presents that COGA has an outstanding optimization effect on the aspects of energy and computing time. However, both the task transmitting rate and the remaining

bandwidth determine how many tasks are offloaded. The fewer the number of offloading tasks, the less the cost on the end-users must be reduced (Fig. 5).

6 Related Work

In the research areas of computation offloading, the proposed approaches aim to energy [8], latency [4] and joint consideration [9]. Zhao et al. [21] discussed the task scheduling based on the consideration of computing limitation in the edge cloud. The task offloading strategy aims to reduce task latency by coordinating the heterogeneous cloud model. Li et al. [10] aim to optimize the formulated cost model in multi-users scenarios by deep reinforcement learning (DRL) method [19]. The results also achieve the proposed design purpose. The resource allocation approach in MEC is one of the key questions, and there are more studies for whole network performance progress. You et al. [20] concentrate on energy consumption and computation latency at multi-user scenarios. The proposed algorithm was designed based on priority policy to reduce the search cost. In similar task offloading scenarios, Chen et al. [5] formulate the several computation decision making among devices as a game in MEC.

From some of the recent researches above, the computation offloading is a complex problem. Each method focuses on the different aspects of MEC systems. The optimized problem ignores the task numbers and decreases the maximum cost of terminal devices which are covered in the communication area of BS. Then we proposed the COGA to optimize the number of offloading tasks and computing cost in MEC.

7 Conclusion

The computation offloading is still a hot issue in MEC, and it is hard to determine the importance of the tasks in the absence of a specific scenario. With the rapid expansion of mobile applications, more and more small computing tasks will appear in the future. It is reasonably prophesied that task computation offloading will become more and more important in the ultra dense network. Therefore, we proposed the COGA computation offloading approach based on the genetic algorithm, which decides the task offloading and adjusts the mobile terminal cost for energy efficiency and computing resource under limited channel capacity. Then, we consider the intensive tasks in BS scenarios, then we redesigned the cross operation for more rapid convergence. After plenty of simulations, several results indicated that the proposed algorithm can effectively determine the offloading tasks. The method has the flexibility for optimized targets, it still has some aspects that have not yet been discussed. For some instances, To make the offloading decision in the continuous-time slot, or some tasks with a large amount of data can not be treated fairly. The valuable problems deserve further discussions in future work.

Acknowledgement. This research was supported by China Scholarship Council (CSC), Fund of Applied Basic Research Programs of Science and Technology Department (No. 2018JY0290). The work of Lei Zhang was supported in part by the National Natural Science Foundation of China under Grant 61902257. The work of Fangxin Wang and Jiangchuan Liu is supported by a Canada NSERC Discovery Grant.

References

1. August 2018. https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/
2. Bockelmann, N., et al.: Massive machine-type communications in 5G: physical and MAC-layer solutions. IEEE Commun. Mag. **54**(9), 59–65 (2016)
3. Chen, R., Liang, C.Y., Hong, W.C., Gu, D.X.: Forecasting holiday daily tourist flow based on seasonal support vector regression with adaptive genetic algorithm. Appl. Soft Comput. **26**, 435–443 (2015)
4. Chen, S., Wang, Y., Pedram, M.: A semi-Markovian decision process based control method for offloading tasks from mobile devices to the cloud. In: 2013 IEEE Global Communications Conference (GLOBECOM), pp. 2885–2890. IEEE (2013)
5. Chen, X., Jiao, L., Li, W., Fu, X.: Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Trans. Netw. **24**(5), 2795–2808 (2015)
6. Di, B., Bayat, S., Song, L., Li, Y., Han, Z.: Joint user pairing, subchannel, and power allocation in full-duplex multi-user ofdma networks. IEEE Trans. Wirel. Commun. **15**(12), 8260–8272 (2016)
7. Du, Y., Dong, B., Chen, Z., Fang, J., Yang, L.: Shuffled multiuser detection schemes for uplink sparse code multiple access systems. IEEE Commun. Lett. **20**(6), 1231–1234 (2016)
8. Huang, D., Wang, P., Niyato, D.: A dynamic offloading algorithm for mobile computing. IEEE Trans. Wirel. Commun. **11**(6), 1991–1995 (2012)
9. Kwak, J., Kim, Y., Lee, J., Chong, S.: DREAM: dynamic resource and task allocation for energy minimization in mobile cloud systems. IEEE J. Sel. Areas Commun. **33**(12), 2510–2523 (2015)
10. Li, J., Gao, H., Lv, T., Lu, Y.: Deep reinforcement learning based computation offloading and resource allocation for MEC. In: 2018 IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6. IEEE (2018)
11. Liu, X., Cao, J., Yang, Y., Qu, W., Zhao, X., Li, K., Yao, D.: Fast rfidsensory data collection: trade-off between computation and communicationcosts. IEEE/ACM Trans. Netw. (2019)
12. Liu, X., Xie, X., Wang, S., Liu, J., Yao, D., Cao, J.: Efficient range queries for large-scale sensor-augmented RFID systems. In: EEE/ACM Trans. Netw. (TON) (2019, in press)
13. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutor. **19**(4), 2322–2358 (2017)
14. Mao, Y., Zhang, J., Song, S., Letaief, K.B.: Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems. IEEE Trans. Wireless Commun. **16**(9), 5994–6009 (2017)
15. Nordrum, A., Clark, K., et al.: Everything you need to know about 5G. IEEE Spectrum (2017)
16. Satyanarayanan, M.: The emergence of edge computing. Computer **50**(1), 30–39 (2017)

17. Teli, S.R., Zvanovec, S., Ghassemlooy, Z.: Optical internet of things within 5G: applications and challenges. In: 2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS), pp. 40–45. IEEE (2018)
18. Wang, F., Wang, F., Ma, X., Liu, J.: Demystifying the crowd intelligence in last mile parcel delivery for smart cities. IEEE Netw. **33**(2), 23–29 (2019)
19. Wang, F., et al.: Intelligent edge-assisted crowdcast with deep reinforcement learning for personalized QoE. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, pp. 910–918. IEEE (2019)
20. You, C., Huang, K., Chae, H., Kim, B.H.: Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Trans. Wirel. Commun. **16**(3), 1397–1411 (2016)
21. Zhao, T., Zhou, S., Guo, X., Niu, Z.: Tasks scheduling and resource allocation in heterogeneous cloud for delay-bounded mobile edge computing. In: 2017 IEEE International Conference on Communications (ICC), pp. 1–7. IEEE (2017)
22. Zhao, X., Zhao, L., Liang, K.: An energy consumption oriented offloading algorithm for fog computing. In: Lee, J.-H., Pack, S. (eds.) QShine 2016. LNICST, vol. 199, pp. 293–301. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60717-7_29