information

MDPI

Article

Genetic Algorithm-Based Optimization of Offloading and Resource Allocation in Mobile-Edge Computing

Zhi Li [1,2] and Qi Zhu [1,2,*]

[1] Jiangsu Key Laboratory of Wireless Communications, Nanjing University of Posts and Telecommunications, Nanjing 210003, China; 18852853125@163.com
[2] Engineering Research Center of Health Service System Based on Ubiquitous Wireless Networks, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
* Correspondence: zhuqi@njupt.edu.cn

Received: 20 December 2019; Accepted: 31 January 2020; Published: 3 February 2020

Abstract: Mobile edge computing (MEC) can use a wireless access network to serve smart devices nearby so as to improve the service experience of users. In this paper, a joint optimization method based on the Genetic Algorithm (GA) for task offloading proportion, channel bandwidth, and mobile edge servers' (MES) computing resources is proposed in the scenario where some computing tasks can be partly offloaded to the MES. Under the limitation of wireless transmission resources and MESs' processing resources, GA was used to solve the optimization problem of minimizing user task completion time, and the optimal offloading task strategy and resource allocation scheme were obtained. The simulation results show that the proposed algorithm can effectively reduce the task completion time and ensure the fairness of users' completion times.

Keywords: mobile edge computing; offloading; genetic algorithm; resource allocation

1. Introduction

With the advent of the 5G network, various smart services are constantly emerging. Billions of intelligent terminal devices are needed to handle a large number of tasks, but the limited number of terminal devices restricts the capabilities to complete them. Mobile edge computing (MEC) [1] enables terminal devices to not only access the network wirelessly but to also offload some computing tasks to the mobile edge servers (MES) to reduce the distance between the servers and the MES and to shorten the completion time of tasks [2]. In the MEC network, offloading strategies and resource allocation will directly affect the performance of the system, so it has become a research hotspot recently [3].

In recent years, scholars have carried out studies on the MEC offloading problem. Offloaded tasks can be divided into partly and fully offloading tasks [4]. A partly offloading task means that users can offload tasks partly or offload all of them at once, and a fully offloading task means that users can only choose to handle all of the tasks locally or in the MES [5]. In [6], an alternating direction multiplier algorithm was proposed to solve the problem of satisfying users' minimum delay and minimizing energy consumption. In [7], a two-step traffic allocation approach was proposed for jointly optimizing channel bandwidth and MES computing resources to minimize latency for all users. In [8], an algorithm based on game theory was proposed to jointly optimize channel bandwidth and MES computing resources to minimize the overall time and energy consumption. In [9], an algorithm based on Q learning was proposed to solve the problem of minimizing the energy consumption of the system. In [10], a potential game algorithm was proposed to solve the problem of minimizing users' task completion time. In [11], task offloading was mapped into a queuing model, which solved the problem of minimizing task offloading time. In [12], the cache strategy was added to the offloading model and the best edge computing offloading method with cache enhancement scheme was proposed in the

paper, which made users' delay shorter and energy consumption less. In [13], user task uploads were mapped into the queuing model to describe network dynamics, and game theory was used to find out the best scheme for computing offload and transmission scheduling. Authors in [14] further considered a wireless powered MEC and minimized the probability of successful computations. The performance optimization of multi-user wireless powered MEC system was later studied in [15,16]. In [17], users offloaded independent tasks to the edge devices and downloaded results from them over prescheduled time slots. Energy consumption at both the user and edge devices was considered herein. In [18], the optimized user offloading strategy algorithm based on game theory was put forward in a transmission queue model, and the optimization goal was defined as the overall time computing task to all users, but it only considered each user's offloading proportion and did not take channel resources and MES computing resources into account. So far, according to the partly offloading task model, there has been a limited joint allocation of partly offloading strategies and communication resources to minimize the completion time mechanism of user devices' (UDs') overall task.

Although there are many literatures about joint optimization strategies and resources allocation to minimize completion time of the UD's tasks or energy, few of them conduct partly offloading scenarios, especially proportional offloading. The main contributions of this paper are as follows:

- We solve the problem of minimizing the overall completion time in the scenario of multiple mobile devices and the one edge server, as well as the UD's task, which can be divided proportionally. We also propose a joint optimization algorithm for users' task partly offloading and resource allocation to solve the problem to solve the problem.
- We propose a joint optimization algorithm of offloading and resource allocation based on the Genetic Algorithm (GA) under the partial offloading task model. A strategy combination composed of the user's offloading proportion, bandwidth, and computing resources is an individual, and each factor in the individual is a gene. Then, different individuals are combined into a population matrix, and the optimal user's offloading proportion and resource allocation combination is finally obtained through selection, crossover, and mutation operations.
- Simulation results demonstrate that our proposed algorithm can effectively shorten the completion time and guarantee fairness among users. For example, when the UD's number is 10, the total completion time in this paper is 12.1% lower than that in literature [18].

The rest of this paper is organized as follows: Section 2 gives the system model and problem description of a multi-user single edge server. Section 3 presents the optimization problem analysis and the joint optimization algorithm. In Section 4, the simulation results are presented and analyzed. Finally, Section 5 summarizes this paper.

2. System Model

As shown in Figure 1, a BS is located at the center of a cell with a MES, M users' devices (UDs) are randomly distributed inside the cell and each of them is represented as UD_m. Each UD_m has a computation task $\mathrm{U_m} = \{f_m, D_m, L_m\}$, f_m represents users' local computing resources, D_m represents the size of the task yet to be handled and L_m represents the number of CPU cycles required to process 1 byte task, where $m \in (1, 2, 3 \ldots M)$.

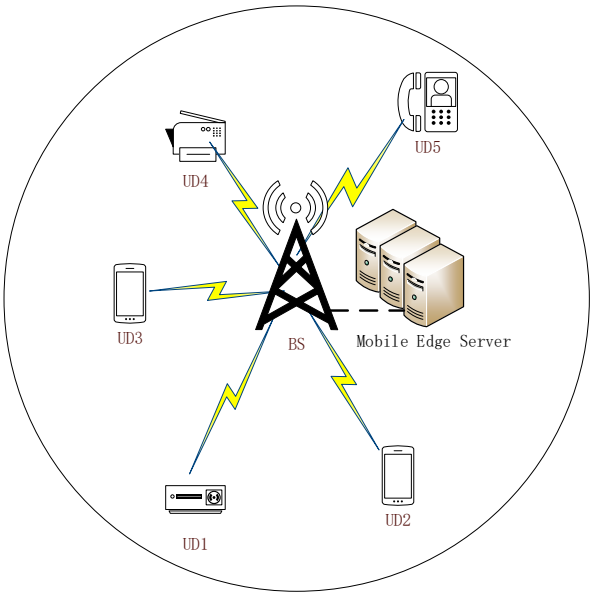

Figure 1. System model.

Each UD is needed to offload P_m proportion of tasks to the MES for processing ($P_m \in [0,1]$), and the remaining $(1-P_m)$ proportional computing tasks are left to be locally handled, so the local processing time of UD_m can be obtained as:

$$T_{m\_local} = \frac{(1-P_m) * D_m * L_m}{f_m} \tag{1}$$

Users upload the offloading data through FDMA (Frequency Division Multiple Access). The total bandwidth of the system is $B_{\max}$, and the bandwidth allocated to UD_m is B_m, so the uplink task data rate of UD_m is given by:

$$R_m = B_m * \log_2(1 + SNR_m) \tag{2}$$

Therefore, the uplink time of UD_m can be expressed as:

$$T_{m\_trans} = \frac{P_m * D_m}{R_m} \tag{3}$$

Assuming that the total computing resources of the MES is $F_{\max}$, and after the UD_m uploads part of the tasks to the edge server, the server will allocate F_m computing resources to process these tasks. The time that the MES processes the offloading tasks of UD_m can be expressed as

$$T_{m\_mec} = \frac{P_m * D_m * L_m}{F_m} \tag{4}$$

After finishing handling the task, the edge server will send the processed data back to the UDs. The download time is T_{back}, which has a small impact on the overall time compared with other times [19] and can be ignored. Therefore, the processing time of the user offloading task can be obtained as:

$$T_{m\_offload} = T_{m\_trans} + T_{m\_mec} = \frac{P_m * D_m * L_m}{F_m} + \frac{P_m * D_m}{R_m} \tag{5}$$

Since the UD's task offloading and local task processing are carried out simultaneously, the total completion time of UD_m is expressed as:

$$T_m = Max(T_{m\_local}, T_{m\_offload}) \tag{6}$$

Since M UDs are handling tasks at the same time, the overall completion time of the system can be obtained as:

$$T = Max(T_1, T_2, T_3 \dots T_M) \tag{7}$$

The objective of this paper is to minimize the total completion time by jointly optimizing the UD_m's task offloading proportion P_m, channel bandwidth B_m, and MES' allocated resources F_m, which not only reduce the total completion time but also ensures fairness among UDs. The above optimization problem is formulated as:

$$\begin{aligned} \min_{B_m, F_m, Pm} \quad & T \\ s.t. \quad & C1: \ T_m \le \tfrac{L_m * D_m}{f_m} \ m \in M \\ & C2: \ \sum_{m \in M} F_m \le F_{\max} \\ & C3: \ \sum_{m \in M} B_m \le B_{\max} \\ & C4: \ 0 \le P_m \le 1 \ m \in M \end{aligned} \tag{8}$$

C1 shows that the processing time after users' offload should not be longer than the time when all the tasks are processed locally, which will cause users not to offload. C2 shows that the sum of MES computing resources allocated to each user should not be larger than the computing resources of MES itself; C3 shows that the sum of bandwidth resources allocated to each user cannot be larger than the channel bandwidth itself; C4 represents the offload proportion of UD_m, and each UD can choose not to offload, partially offload, or fully offload, depending on the number of users, local computing resources, MES computing resources, and channel bandwidth.

3. Algorithm Formulation

The above optimization problem (8) is a non-convex optimization problem, which cannot be solved by common optimization methods such as the Lagrange multiplier method or the KKT condition. GA, as a kind of heuristic algorithm [20], does not process object parameter itself, but rather through a parameter set encoded by multiple genetic individuals, in other words, it evaluates multiple solutions in the search space. It has a good global search capability and, through the gene selection, crossover, and mutation genetic operation, the search is not easy to fall into local optimum and can rapidly and accurately solve complex problems [21].

In this paper, a genetic algorithm was adopted to solve the optimization problem (8). The encoding method is shown in Figure 2. n different strategy combinations $\{P, F, B\}$ are coded to form a population matrix, as shown in (9). $\{P, F, B\}$ is an individual in the population matrix, and each value in $\{P, F, B\}$ becomes a gene for the individual in the population matrix. For each strategy combination in the matrix, the $\{P, F, B\}$ with the lowest task completion time is selected as the optimal individual in an iteration. Crossover and mutation operators, through different combinations between $\{P, F, B\}$, achieve a new population matrix into the next iteration and finally get the global optimal value.

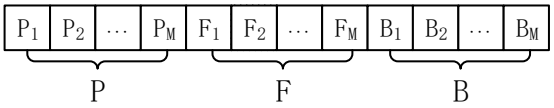

Figure 2. Encoding method.

3.1. Population Formation and Optimal Individual Selection

Assuming that there are M UDs, each of them needs to optimize the three variables P_m, F_m, B_m, so a n row 3*M column population matrix K is generated. The row set at 4 to 6 times the number of column is easier to make the algorithm convergence and reduce the iteration complexity, so the n is 4*3*M. The population K is formulated as

$$K = \begin{bmatrix} k_{1,1}, k_{1,2} \dots k_{1,M}, k_{1,M+1} \dots k_{1,2M}, k_{1,2M+1}, \dots k_{1,3M} \\ k_{2,1}, k_{2,2} \dots k_{2,M}, k_{2,M+1} \dots k_{2,2M}, k_{2,2M+1}, \dots k_{2,3M} \\ \dots\dots \\ k_{n1,}, k_{n,2} \dots k_{n,M}, k_{n,M+1} \dots k_{n,2M}, k_{n,2M+1}, \dots k_{n,3M} \end{bmatrix} \tag{9}$$

where $k_{i,j}(i \in [1,n], j \in [1,m])$ is the offloading proportion of the UD j in the strategy combination i, $k_{i,j}(i \in [1,n], j \in [m+1,2m])$ is the allocated MES computing resources of the UD $j-m$ in the strategy combination i. $k_{i,j}(i \in [1,n], j \in [2m+1,3m])$ is the allocated channel bandwidth of the UD $j-2m$ in the strategy combination i.

The optimization problem proposed in this paper is a minimization problem, T is defined as the fitness function in the algorithm, and each iteration process has the selection process of excellent individual $\{P,F,B\}$ (for simplicity, we define it as K_i) in the population to enter the next genetic operation. This is to avoid the optimization falling into the local optimal, so the subsequent iteration process can converge to the global optimal solution faster. The specific process is shown in Algorithm 1.

Algorithm 1 Optimal individual selection

1. **Input:** K, $U_m = \{f_m, D_m, L_m\}$, $B_{\max}$, $F_{\max}$.
2. **For** $i = 1:N$:
3. **For** $j = 1:M$:
4. $T_{local} = \left(1 - k_{i,j}\right)/f_j$;
5. $T_{offload} = \left(\left(k_{i,j}\right) * D_j\right)/\left(k_{i,j+2M} * log2\left(1 + SNR_j\right)\right) + \left(k_{i,j} * D_j * L_j\right)/k_{i,j+M}$;
6. $T_j = Max\left(T_{local}, T_{offload}\right)$;
7. **End**
8. $T_i = MAX(T_1, T_2 \dots T_m)$;
9. **End**
10. $T = (T_1, T_2 \dots T_n).Q_n = T_n/sum(T).Q = (Q_1, Q_2 \dots Q_n)$.
11. Generate a random number *rand* ([0, 1]). Calculate where x is in Q. Select the individuals K_i to whom the scope belongs.
12. **Output:** K_i

3.2. Crossover and Mutation operation

After the selection of an excellent individual, crossover and mutation operations are carried out in different combinations of $\{P,F,B\}$ in K. Two strategy combinations i and i' in a group are as parents, and crossover operations mean that the B_m, F_m, P_m of the same UD are randomly exchanged in the two strategy combinations. The specific process is shown in Figure 3.

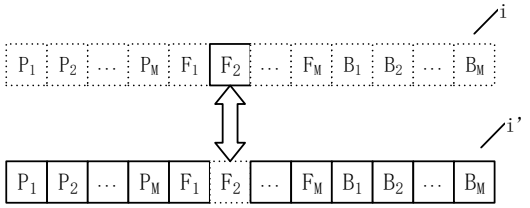

Figure 3. Crossover operation.

If the exchanged variables exceed the limits of the constraints, a new individual is selected or a set of strategy combinations are used for cross-operations.

The mutation operation randomly changes the B_m, P_m, F_m in a combination of strategy, so the algorithm has a higher global search capability, and it also makes the algorithm's local search capability stronger and maintains the diversity of the group. The specific operation process is shown in Figure 4. If the changed variable exceeds the limit of the constraint, a new random value is selected for mutation operation. The specific process is shown in Algorithm 2.

Algorithm 2 Crossover and mutation

Input K
  For $i = 1 : n/2$:
  Generate a random number m ($m \in [0, 3M]$).Change the value of $k_{2*i-1,m}$ and $k_{2*i,m}$.
  End.
  Get new population matrix K^1.
  Perform Algorithm 1 on K^1. Get new greatest individual k_{ii}.
  For $i = 1 : n$:
    Generate a random number b($b \in [0, 3M]$).
    If: $0 \le b \le M$:
      Generate c($c \in [0, 1]$);
    Else if : $M + 1 \le b \le 2M$:
      Generate c($c \in [0, F_{\max}]$);
    Else if : $2M + 1 \le b \le 3M$:
      Generate c($c \in [0, B_{\max}]$);
    If c satisfies the condition of (8):
      $k_{i,b} = c$;
    else return (8)
  End.
  Get new population matrix K^2
  Perform Algorithm 1 on K^2.Get new greatest individual k_{iii}.
Output: k_{ii}, k_{iii}

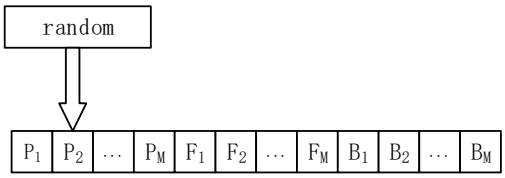

Figure 4. Mutation operation.

3.3. Algorithm Overview

In this algorithm, the optimal individual selection, crossover, and mutation operations are used to obtain the three best strategy combinations k_i, k_{ii}, and k_{iii}, and they are add to the next iteration to continue the genetic operation. For each iteration, the three strategy combinations will be substituted into the fitness function T to get the minimum completion time, which will be compared with the minimum completion time in the last iteration. If the difference is less than *exp*(precision), iteration will be stopped and the best $\{P, F, B\}$ will be output in $[k_i, k_{ii}, k_{iii}]$; otherwise, the next genetic operation will be continuously conducted until the emergence of the max iteration numbers (*iter*).The specific process is shown in Algorithm 3.

Algorithm 3 A joint optimization algorithm of offloading and resource allocation

Input $U_m = \{f_m, D_m, L_m\}$, $B_{\max}$,$F_{\max}$,*exp*,*iter*.
Randomly generate a 3 row 3∗M column matrix K'
For $k = 1 : iter$:
  Randomly generate a n row 3∗M column matrix K
  Replace the first 3 lines of K with K'
  Do Algorithm 1 and Algorithm 2 on K. Get k_i,k_{ii},k_{iii}.
  $K'' = [k_i, k_{ii}, k_{iii}]$
  If $abs((\min(T(K'')) - \min(T(K')))) \le exp$:
    break;
  Else:
    If $\min(T(K'')) \le \min(T(K'))$:
      $K' = K''$;
End
Output: $Argmax(T(K''))$;

4. Simulation Results and Discussions

In this section, the performance of the proposed algorithm was evaluated through Matlab simulation. The simulation scenario is shown in Figure 1. The simulation parameters are cited in Table 1.

Table 1. Simulation parameters.

Parameter	Value
MES computational resource $F_{\max}$	{2.5, 5, 10, 15} GHz/s
Channel bandwidth $B_{\max}$	[10, 20] MHz
Data task D_m	[8, 12] MB
Needed cpu cycles to calculate 1 bit task L_m	1000 cycles/byte
exp	0.001
UD's local computational capacity f_m	[0.8, 1.2] GHz/s
n	4 * 3 * M
UD numbers M	[2, 30]
SNR_m	[5, 25] dB

Figure 5 shows the simulation results of the total completion time with the number of iterations for 10 users and 15 users. It can be seen from the figure that the overall task completion time of the genetic algorithm tends to converge after 20 iterations.

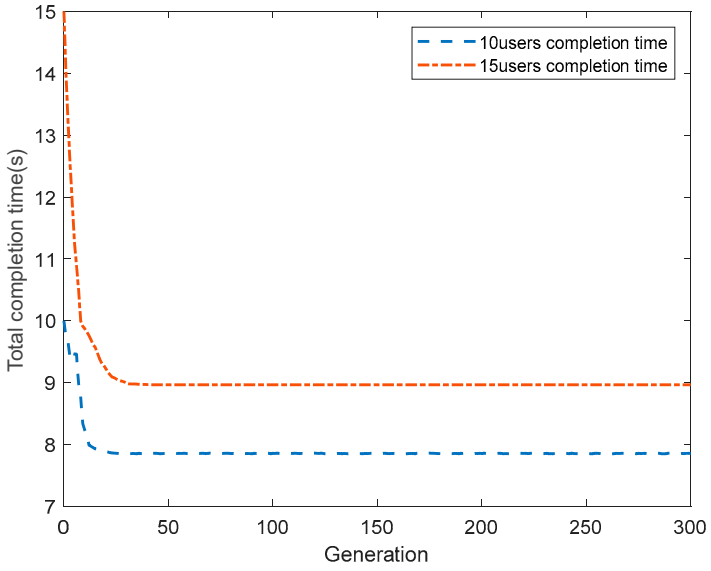

Figure 5. Total completion time with generation times.

Figure 6 shows the simulation results of the comparison of the UD numbers and the total completion time T when $B_{\max} = 10$ MHz, $F_{\max} = 5$ GHz/s and compares the algorithm of this paper with the algorithm of [12], the algorithm that does not offload, the algorithm that only optimizes bandwidth (average allocating computing resource), and the algorithm that only optimizes computing resource (average allocating bandwidth). It illustrates that if there is no offloading, the average completion time will be around 10 seconds. As the number of UDs increases, other algorithms approach this value. Algorithm [18] does not consider the allocation of bandwidth and MES computing resources, resulting in a waste of resources, so the overall completion time of this paper is faster. In conclusion, both the optimized bandwidth and the computing resources can get a fast total completion time.

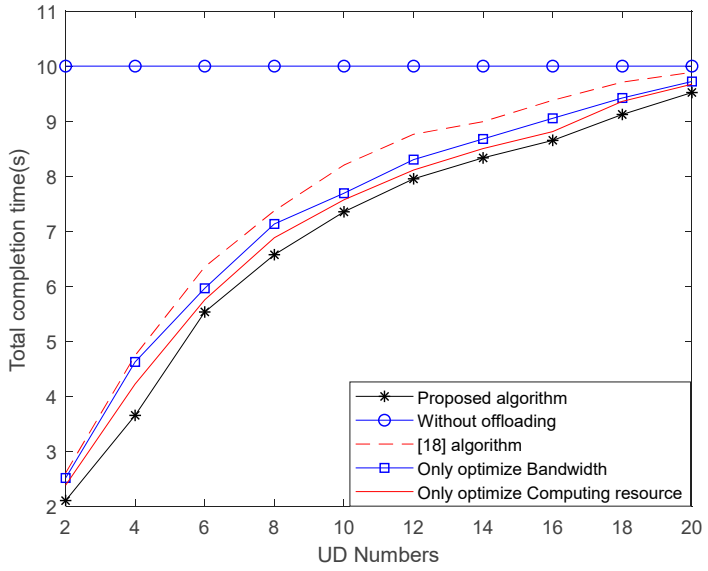

Figure 6. Total completion time with different user device (UD) numbers.

Figure 7 shows the simulation results of the comparison of the UD numbers and the UD's fairness when $B_{\max} = 10$ MHz, $F_{\max} = 5$ GHz/s and compares the algorithm of this paper with the algorithm of [18]. In this paper, the standard deviation of time was used to reflect the fairness of users [22] and the metric was given by:

$$Fairness = \frac{\sqrt{\sum_{m=1}^{M} \left(t_m - \bar{t}\right)^2}}{M} \tag{10}$$

where $\bar{t}$ is the average task completion time of all UDs. The figure illustrates that the larger the distributable resources are, the stronger the fairness is. Compared with the literature [18], the UD's completion time in this paper was fairer.

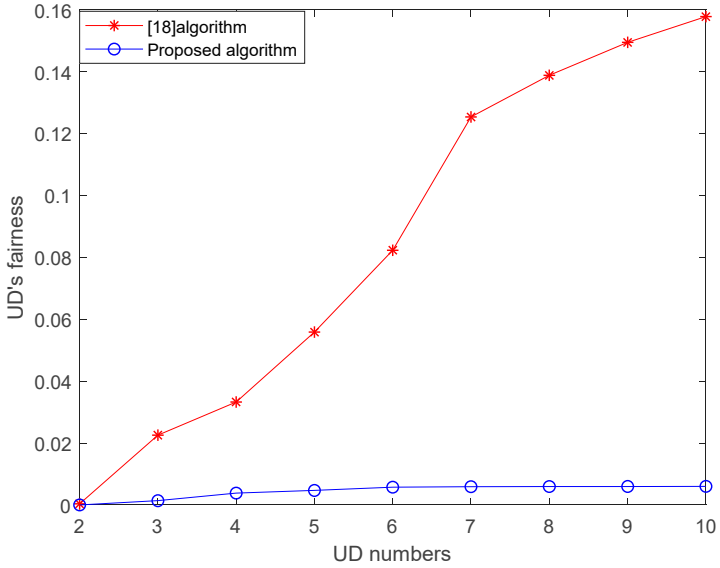

Figure 7. UD's fairness with different UD numbers.

Figure 8 shows the simulation results of T vary with $B_{\max}$. According to the figure, the completion time decreases in line with an increase in the bandwidth that can reduce the transmission time.

However, when the bandwidth increases to a certain value, the time reduction tends to be flat, because the transmission time is small when the bandwidth is wide. While the MEC computing power does not change, T_{m_mec} does not change either. So, when the transmission time is not an order of T_{m_mec}, the change in completion time will be much smaller.

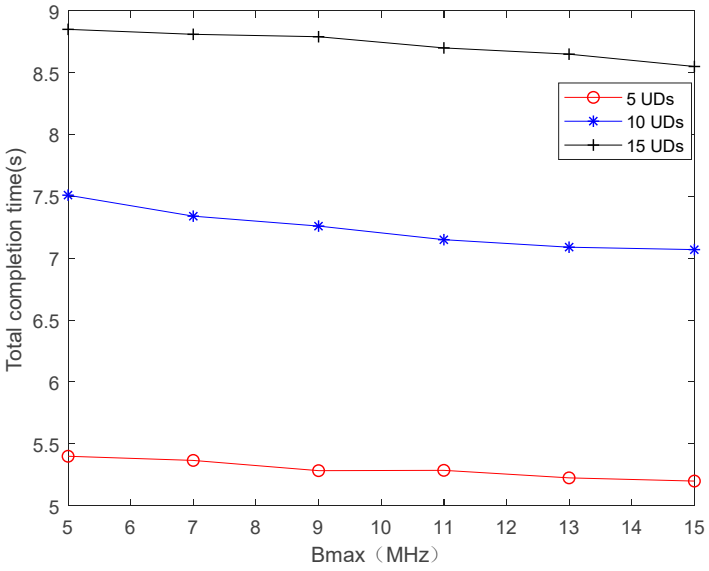

Figure 8. Total completion time T varies with $B_{\max}$.

Figure 9 shows the simulation results of T vary with $F_{\max}$. According to the figure, as the $F_{\max}$ increases, the total completion time T continuously decreases. Since the MES calculation time is larger than the data transmission time in the scenario assumed in this paper, the trend of time decline is larger than that of Figure 8. Similar to Figure 8, when the $F_{\max}$ increases to a certain extent, the completion time tends to be stable. Comparing Figures 8 and 9 with Figure 6, it can be seen that jointly optimizing MES computing resources and communication resources is more effective in reducing task completion time than considering only one resource.

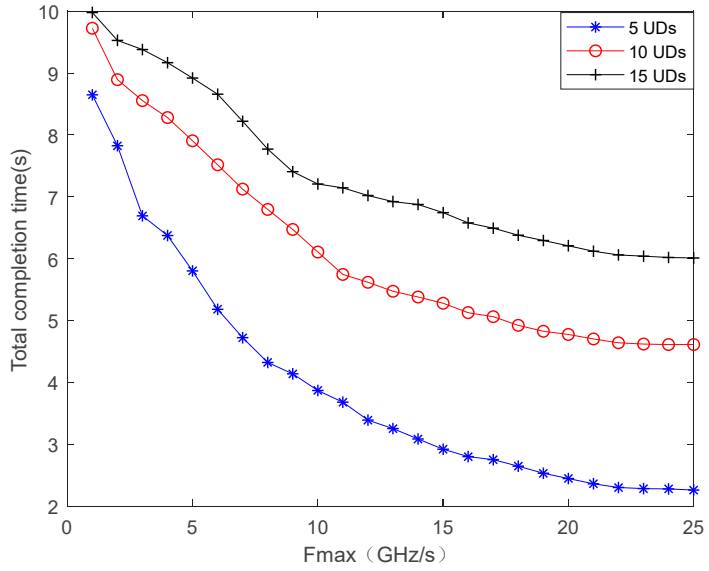

Figure 9. Total completion time T varies with $F_{\max}$.

Figure 10 shows that the simulation results of the UD's average offload proportion vary with UD numbers when $B_{\max} = 10$ MHz. According to the figure, as the number of UDs increases, the UD's average offload proportion in the three environments gradually decreases. This is because, although the number of UDs is increasing, $F_{\max}$ and $B_{\max}$ do not change. If the offload proportion is the same, then T_{m_local} will not change but $T_{m\_offload}$ will gradually rise. T slows down, and hence the average offload proportion decreases as the number of UDs increases. And when the number of users is the same, $F_{\max}$ is larger and, therefore, the average UD's offload proportion is larger.

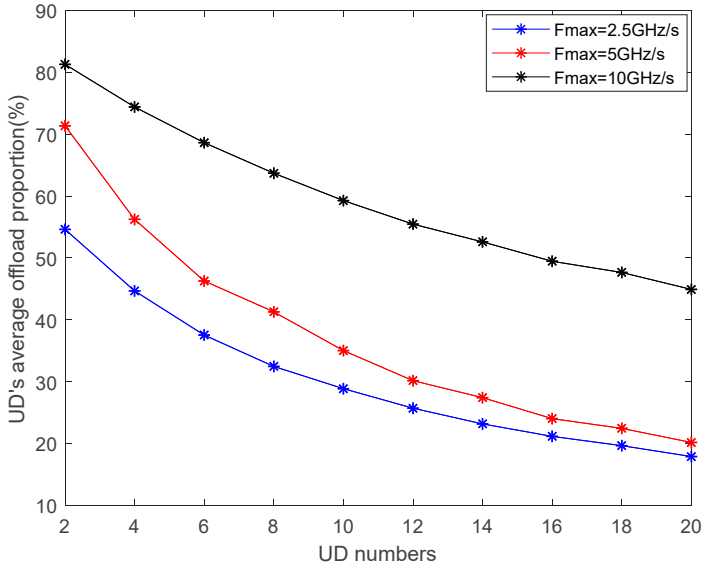

Figure 10. UD's average offload proportion varies with UD numbers.

5. Conclusions

This paper minimizes the overall completion time in the scenario of multiple mobile devices and a one edge server and proposes a joint optimization algorithm for user task offloading and resource allocation to solve the problem of minimizing the overall completion time. The simulation results show that, compared with a single allocation offload strategy, the joint allocation of resources can make the overall completion time shorter. Optimizing bandwidth or MES computing resources separately can reduce the task offload completion time, however, it will encounter bottlenecks. Jointly optimizing communications and computing resources can make tasks offload faster. However, the scenario in this article is relatively simple. It only considers the user's completion time and does not consider energy consumption. It does not take into account the true distance and does not consider the situation of multiple edge servers and base stations. This will be the direction of further research in the future.

Author Contributions: Investigation: Z.L. and Q.Z.; Writing-original draft, Z.L. and Q.Z. All authors have read and agreed to the published version of the manuscript.

Funding: National natural science foundation of China (61971239, 61631020). zhuqi@njupt.edu.cn

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Abbas, N.; Zhang, Y.; Taherkordi, A.; Skeie, T. Mobile Edge Computing: A Survey. *IEEE Internet Things J.* **2017**, *5*, 1–12. [CrossRef]
2. Wang, S.; Zhang, X.; Zhang, Y.; Wang, L.; Yang, J.; Wang, W. A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications. *IEEE Access* **2017**, *5*, 6757–6779. [CrossRef]
3. Xu, X.; Liu, J.; Tao, X. Mobile Edge Computing Enhanced Adaptive Bitrate Video Delivery with Joint Cache and Radio Resource Allocation. *IEEE Access* **2017**, *5*, 16406–16415. [CrossRef]

4. Hong, Q.L.; Al-Shatri, H.; Klein, A. Optimal Joint Power Allocation and Task Splitting in Wireless Distributed Computing. In Proceedings of the 11th International ITG Conference on Systems, Communications and Coding, Hamburg, Germany, 6–9 February 2017.
5. Chen, M.H.; Liang, B.; Min, D. Joint offloading decision and resource allocation for multi-user multi-task mobile cloud. In Proceedings of the IEEE International Conference on Communications, Kuala Lumpur, Malaysia, 22–27 May 2016.
6. Li, L.; Zhang, X.; Liu, K.; Jiang, F.; Peng, J. An Energy-Aware Task Offloading Mechanism in Multiuser Mobile-Edge Cloud Computing. *Mob. Inf. Syst.* **2018**, *18*, 1–12. [CrossRef]
7. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing. *IEEE Trans. Netw.* **2016**, *24*, 2795–2808. [CrossRef]
8. Xu, C. Decentralized Computation Offloading Game for Mobile Cloud Computing. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *26*, 974–983.
9. Zhao, H.; Wang, Y.; Sun, R. Task Proactive Caching Based Computation Offloading and Resource Allocation in Mobile-Edge Computing Systems. In Proceedings of the 14th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; pp. 232–237.
10. Liu, K.; Peng, J.; Li, H.; Zhang, X.; Liu, W. Multi-device task offloading with time-constraints for energy efficiency in mobile cloud computing. *Future Gener. Comput. Syst.* **2016**, *64*, 1–14. [CrossRef]
11. Cardellini, V.; Personé, V.D.; Di Valerio, V.; Facchinei, F.; Grassi, V.; Presti, F.L.; Piccialli, V. A game-theoretic approach to computation offloading in mobile cloud computing. *Math. Program.* **2015**, *157*, 421–449. [CrossRef]
12. Yu, S.; Langar, R.; Fu, X.; Wang, L.; Han, Z. Computation Offloading With Data Caching Enhancement for Mobile Edge Computing. *IEEE Trans. Veh. Technol.* **2018**, *67*, 11098–11112. [CrossRef]
13. Yi, C.; Cai, J.; Su, Z. A Multi-User Mobile Computation Offloading and Transmission Scheduling Mechanism for Delay-Sensitive Applications. *IEEE Trans. Mob. Comput.* **2019**, *19*, 29–43. [CrossRef]
14. You, C.; Huang, K.; Chae, H. Energy Efficient Mobile Cloud Computing Powered by Wireless Energy Transfer. *IEEE J. Sel. Areas in Commun.* **2016**, *34*, 1757–1771. [CrossRef]
15. Bi, S.; Zhang, Y.J. Computation Rate Maximization for Wireless Powered Mobile-Edge Computing with Binary Computation Offloading. *IEEE Trans. Wirel. Commun.* **2017**, *17*, 4177–4190. [CrossRef]
16. Meng, S.; Wang, Y.; Miao, Z.; Sun, K. Joint optimization of wireless bandwidth and computing resource in cloudlet-based mobile cloud computing environment. *Peer-to-Peer Netw. Appl.* **2018**, *11*, 462–472. [CrossRef]
17. Xing, H.; Liu, L.; Xu, J.; Nallanathan, A. Joint Task Assignment and Resource Allocation for D2D-Enabled Mobile-Edge Computing. *IEEE Trans. Commun.* **2019**, *67*, 4193–4207. [CrossRef]
18. Nowak, D.; Mahn, T.; Al-Shatri, H.; Schwartz, A.; Klein, A. A Generalized Nash Game for Mobile Edge Computation Offloading. In Proceedings of the 6th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (Mobile Cloud), Bamberg, Germany, 26–29 March 2018.
19. Prékopa, A. On logarithmic concave measures and functions. *Acta Sci. Math.* **1973**, *34*, 335–343.
20. Galletly, J. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*; Oxford University Press: New York, NY, USA, 1999.
21. Morris, G.M.; Goodsell, D.S.; Halliday, R.S.; Huey, R.; Hart, W.E.; Belew, R.K.; Olson, A.J. Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function. *J. Comput. Chem.* **2015**, *19*, 1639–1662. [CrossRef]
22. Al-Kanj, L.; Poor, H.V.; Dawy, Z. Optimal Cellular Offloading via Device-to-Device Communication Networks with Fairness Constraints. *IEEE Trans. Wirel. Commun.* **2014**, *13*, 4628–4643. [CrossRef]