Liao *et al. Journal of Cloud Computing: Advances, Systems and Applications* (2021) 10:15
https://doi.org/10.1186/s13677-021-00232-y

Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH Open Access

Adaptive offloading in mobile-edge computing for ultra-dense cellular networks based on genetic algorithm

Zhuofan Liao[1], Jingsheng Peng[1], Bing Xiong[1] and Jiawei Huang[2*]

Abstract

With the combination of Mobile Edge Computing (MEC) and the next generation cellular networks, computation requests from end devices can be offloaded promptly and accurately by edge servers equipped on Base Stations (BSs). However, due to the densified heterogeneous deployment of BSs, the end device may be covered by more than one BS, which brings new challenges for offloading decision, that is whether and where to offload computing tasks for low latency and energy cost. This paper formulates a multi-user-to-multi-servers (MUMS) edge computing problem in ultra-dense cellular networks. The MUMS problem is divided and conquered by two phases, which are server selection and offloading decision. For the server selection phases, mobile users are grouped to one BS considering both physical distance and workload. After the grouping, the original problem is divided into parallel multi-user-to-one-server offloading decision subproblems. To get fast and near-optimal solutions for these subproblems, a distributed offloading strategy based on a binary-coded genetic algorithm is designed to get an adaptive offloading decision. Convergence analysis of the genetic algorithm is given and extensive simulations show that the proposed strategy significantly reduces the average latency and energy consumption of mobile devices. Compared with the state-of-the-art offloading researches, our strategy reduces the average delay by 56% and total energy consumption by 14% in the ultra-dense cellular networks.

Keywords: Mobile edge computing offloading, 5G, Latency, Energy, Genetic algorithm

Introduction

The development of mobile applications facilitates people's life, such as augmented reality and intelligent interconnection, etc. These applications are characterized by a large amount of computation and a low tolerance for latency. However, mobile devices are limited in computation and endurance capacity. Mobile edge computing (MEC) technology has been regarded as an effective solution to the above constraints. Mobile devices get powerful computing capacity and lower energy consumption when the computing tasks are offloaded to edge servers [1]. Moreover, the integration of the MEC technology and next generation networks, such as the fifth/sixth generation networks (5G/6G), can save bandwidth resources and improve user experience [2, 3].

Though enjoying the super-high data transmission rate brought by the next generation networks, mobile devices still face new problems, such as the network densification [4] and low signal penetration. Many existing works formulated the MEC problem in 4G cellular networks as multi-user to one single-server model [5–7], and their concern was task local processing or remote offloading to the base station. Compared with 4G, the frequency of the 5G network is much higher, which means the faster its attenuation will be. Considering the development of antenna technology and the compensation of radio frequency technology, the 5G construction density of China

*Correspondence: jiaweihuang@csu.edu.cn
[2]School of Computer Science and Engineering, Central South University, Lushan South Road, 410083 Changsha, China
Full list of author information is available at the end of the article

Unicom and China Telecom should be about 1.7 times that of the 4G base station, and 3.2 times for China Mobile [8]. The ultra density of base stations means overlapped coverage of mobile devices, which brings a new challenge for offloading decisions: which accessible BS is the best one to offload tasks to achieve low latency and energy consumption?

In this paper, we formulate the offloading decision problem in the ultra-dense BSs environment as a nonlinear integer optimization problem, which is proved to be NP-Hard [9]. Then a suboptimal algorithm is designed to solve this problem. First, by considering the preference of both proximity and workload, mobile users are grouped to one BS with the highest preference as a destination to offload tasks. With the result of grouping, the original problem is decomposed into several distributed 0-1 integer optimization sub-problems. After that, a binary-coded Genetic Algorithm Based Distributed Offloading Strategy (GABDOS) is proposed to solve these sub-problems in parallel. The main contributions of this paper are as follows:

1. A Multi-User-to-Multi-Servers edge computing offloading problem, termed as MUMS, is proposed, which is especially for the ultra-dense 5G cellular networks.
2. The multi-user-to-multi-servers problem is divided and conquered by two phases, which are server selection and offloading decision. For the server selection phases, mobile users are grouped to the BS with the highest preference, which fully considers physical distance and workload of base stations.
3. After the grouping, the original multi-user-to-multi-servers problem is converted into several parallel offloading decision subproblems, each of which can be formulated as a nonlinear integer optimization problem. To get a fast near-optimal solution, a binary-coded Genetic Algorithm Based Distributed Offloading Strategy (GABDOS) is designed to get an adaptive offloading decision.
4. Convergence analysis of the genetic algorithm is given and extensive simulations show the performance of our solution from the impact of user amount, user device CPU frequency, and trade-off parameter for offloading latency and energy of users.

The rest part of the paper is organized as follows. The related works are reviewed in Related works section. The system model is elaborated in System model section. Problem formulation and solutions section gives the problem formulation and detailed solutions. Performance evaluations are described in Performance evaluation section, and finally, the paper is concluded in Conclusion section.

Related works

By deploying servers near users, MEC can provide additional computing services for mobile devices. Users can get lower latency and less energy consumption by offloading their computing tasks to edge servers. In order to fully realize the convenience of edge computing under the limitation of channel resources and computing capacity of edge servers, scholars used different excellent methods to optimize edge computing.

Common methods are primal-dual optimization [10, 11], nonlinear optimization [7, 12, 13] and mixed-integer linear programming problem [14, 15]. In [10], an online truthful mechanism based on the primal-dual optimization framework integrating computation and communication resource allocation is proposed . For multi-user wireless powered mobile edge computing (WP-MEC) systems, an online computation rate maximization (OCRM) algorithm is proposed by jointly managing the radio, computational resources, allocating time for energy transfer and data transmission [11].

Qi et al.[7] formulated the offloading decision as a resource scheduling problem with single or multiple objective functions and constraints. Bai et al.[12] conceived an energy-efficient computation offloading technique for UAV-MEC systems and formulated many energy-efficiency problems, which are then transformed into convex problems. In [13], Alghamdi et al. tackled the MEC problem by adopting the principles of optimal stopping theory contributing to two time-optimized sequential decision-making models. Guo et al. [14] forced on the problem of assigning resources for offloading the computationally intensive tasks of mobile applications. The original static offloading problem was formulated as a mixed-integer linear programming problem, and then be solved by an efficient heuristic approach based on congestion awareness. A cloud-mobile edge computing collaborative task offloading scheme with service orchestration (CTOSO) is proposed in [15]. In [16], a novel Baseline Data based Verifiable Trust Evaluation (BD-VTE) scheme is proposed to guarantee credibility at a low cost.

In recent years, optimization methods based on reinforcement learning [17–20] and artificial intelligence [21, 22] have emerged. A reinforcement learning-based online computation offloading approach for block chain-empowered mobile edge computing was proposed in [17]. In [18], deep reinforcement learning is first proposed to solve the offloading problem of multiple service nodes for the cluster and multiple dependencies for mobile tasks in large-scale heterogeneous MEC. Gao et al. investigated a DNN based MEC scheme considering multiple mobile devices and one MEC server in [19]. A Q-learning based flexible task scheduling with global view (QFTS-GV) scheme is proposed to improve task scheduling success rate, reduce delay, and extend lifetime for the

IoT in [20]. Miao et al. [21] put forward a new intelligent computation offloading based MEC architecture in combination with artificial intelligence (AI) technology. A matrix completion-based Sampling Points Selection joint Intelligent Unmanned Aerial Vehicle (UAVs) Trajectory Optimization (SPS-IUTO) scheme for data acquisition is proposed in [22]. With different system models, existing researches have contributed a lot to solve the above problems in MEC. However, most of them do not consider the queuing delay after the task is offloaded to the server, and also rarely consider the channel interference between mobile devices.

Genetic algorithm (GA) is a mature optimization algorithm, which is very stable in solving optimization problems. GA can obtain the global optimal solution which is independent of the initial conditions. Also, GA has strong robustness and is suitable for solving complex optimization problems. In some researches, GA has been applied to the optimization of MEC and fog computing.

Xu et al.[23] proposed an energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks, which adopt Non-dominated Sorting Genetic Algorithm II (NSGA-II) to realize multi-objective optimization to shorten the offloading time of the computing tasks and reduce the energy consumption of the ECNs. Goudarzi et al. [24] used GA and adjust it for the multisite offloading problem. Also, genetic operators are modified to reduce ineffective solutions and hence obtain the best possible solutions in a reasonable time. A modified GA, named energy sensitive GA, is developed and integrated to get the optimized task offloading result in [25], which is critical to most cloud robotic network applications.

Du et al. [26] minimize the total system overhead of the MEC system by jointly optimizing computation offloading decision making and communication channel allocation with GA. Li et al. solved a cost-minimization problem for the MEC problem using an improved genetic algorithm [27], but the offloading latency is not taken into account and tested. Wu et al. [28] proposed a distributed priority offloading mechanism with joint offloading proportion and transmission (PROMOT) energy algorithm based on Genetic Algorithm. PROMOT uses the offloading probability, which is given by the prior information of past offloading feedbacks, to determine if a task should be processed locally or remotely. Many of the above studies have cleverly combined GA with MEC. One disadvantage of GA is its uncertain convergence. The poor parameter setting is easy to lead to local optimum or slow convergence. However, many of the above works do not consider the convergence of the genetic algorithm.

Different from existing researches, in this paper, (1) a user grouping strategy is proposed considering both geographical distribution and base station load of the servers. (2) Queuing theory is introduced to analyze the queueing time of tasks and signal interference between mobile devices is fully considered. (3) To improve the scalability of the algorithm, the original multi-user-to multi-server problem is divided into several independent subproblems, which can be processed independently and in parallel. (4) To optimize the convergence performance of GA, the crossover probability(C_P) and mutation probability(M_P) are fully analyzed and the optimal value is recommended. Finally, a genetic algorithm-based distributed offloading strategy is proposed to obtain the optimal solution with high efficiency and low deployment overhead.

System model

Without loss of generalit, Orthogonal Frequency Division Multiple Access (OFDMA) technology is adopted for the 5G environment in this work. Each BS is equipped with an edge server with limited computing capacity. The data transmission delay between the BS and its edge server is so small that it can be counted as zero. The geographical distribution of users follows the Geometric distribution. On the other hand, each user device generates a computing task obeying the Poisson distribution in each time slot. And each computing task can not be divided, i.e., the task must be processed either locally or offloaded.

Network model

As shown in Fig. 1, m base stations are denoted as set $B = \{b_1, b_2, \ldots, b_m\}$, each of which is fixed with an edge server. The set of m servers is marked as $S = \{s_1, s_2, \ldots, s_m\}$. End users $U = \{u_1, u_2, \ldots, u_n\}$ are distributed in the area with geometric distribution. The set of computing tasks generated by all users is denoted as $T = \{t_1, t_2, \ldots, t_n\}$. Because of the high density of 5G BSs, the coverage area of servers will overlap with each other. That is to say, most users will be covered by more than one BS.

About the mobility of end-users. In the area of high population density, users move at 2-3 kilometers per hour and vehicles run at about 20 kilometers per hour, whose shift is relatively small for a 5G cellular station with a coverage radius of 500 to 1000 meters. Therefore, it can be assumed that the end-users' position is approximately stationary in each given time slot.

Communication model

As mentioned earlier, user devices transmit data with the BS through the wireless channel. The data transmission rate is determined by the general communication model, Shannon-Hartley theorem [29]:

$$r_{u_i,b_j} = \frac{W_{b_j}}{N} \log_2 \left(1 + \frac{p_{u_i} g_{u_i,b_j}}{\omega + \sum_{k=1,k\neq i}^{N} p_{u_i} g_{u_i,b_j}} \right), \quad (1)$$

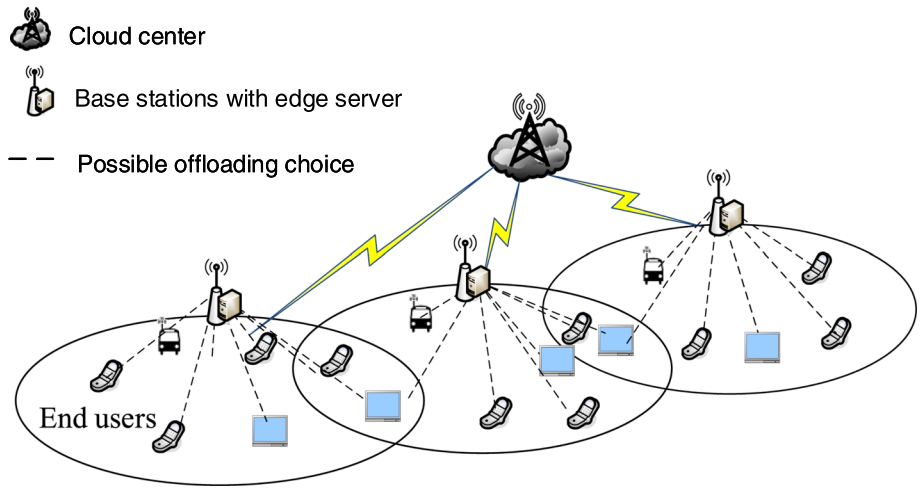

Fig. 1 Network model

where W_{b_j} denotes the bandwidth of BS b_j, N denotes the number of users the BS b_j served, p_{u_i} denotes the transmission power of user u_i, g_{u_i,b_j} denotes the channel gain between user u_i and BS b_j, and ω denotes background noise. Because the edge server is equipped on the BS, the transmission delay between them is very small, which can be regarded as zero. Therefore, the data transmission rate between the BS and the user equipment can directly be used to represent the data transmission rate between the user equipment and the edge server:

$$r_{u_i,s_j} = r_{u_i,b_j}. \tag{2}$$

The notations of this paper are summarized in Table 1.

Computation model

F_{s_j} is used to represent the CPU frequency of the edge server s_j. W_{b_j} is the bandwidth of BS b_j. For a task t_i, two indicators are used to describe it, one is the size of input data D_{t_i}, the other is the CPU cycle C_{t_i} it needs. Due to the small amount of data in the calculation result, the latency of the result back to the user's device is ignored.

Local processing model

If a task t_i is processed locally, its energy consumption and latency can be easily calculated, because the data transmission is not needed to be considered. Its latency will only be related to the computing capacity of the mobile device and the CPU cycle needed to calculate the task:

$$L_{t_i}^{l} = \frac{C_{t_i}}{F_{u_i}}. \tag{3}$$

Its energy consumption can be calculated by the following formula:

$$E_{t_i}^{l} = C_{t_i}\alpha. \tag{4}$$

where α represents the energy consumption generated by the device in each CPU cycle. In this paper, the model in [30] is used to calculate the value of α:

$$\alpha = \kappa \left(F_{u_i}\right)^2, \tag{5}$$

where κ is the energy consumption coefficient.

MEC processing model

In the model of this work, we assume that it is serial computing rather than parallel computing when the tasks are offloaded to the edge server, so we should consider the queuing situation of the tasks. If a task t_i is offloaded to the edge server for processing, its latency includes three parts: transmission latency, queuing latency, and calculation latency, and its energy consumption is equal to the transmission energy consumption. The following formula can calculate the transmission latency of the task:

$$L_{t_i}^{off,s_j} = \frac{D_{t_i}}{r_{u_i,s_j}}. \tag{6}$$

We consider an M/M/1 queue on the edge server s_j. Therefore, the queuing latency and calculation latency mentioned above will be combined as processing latency. According to queuing theory, average processing delay can be calculated by the following formula:

$$L^{soj,s_j} = \frac{1}{\mu_{s_j} - \lambda_{s_j}}, \tag{7}$$

where μ_{s_j} represents the task processing rate on the server s_j and λ_{s_j} is the task arrival rate. In this case, the energy consumption generated by the equipment only includes data transmission energy consumption:

$$E_{t_i}^{off,s_j} = \frac{D_{t_i}}{r_{u_i,s_j}} p_{u_i}. \tag{8}$$

Table 1 Notations

Symbol	Definition
S	The set of edge services
B	The set of BSs
U	The set of all users
N	The number of users the BS b_j served
T	The set of all computing tasks
r_{u_i,b_j}	The transmission rate between user i and BS j
W_{b_j}	The bandwidth of BS j
p_{u_i}	The transmission power of user i
g_{u_i,b_j}	The channel gain between user i and BS j
ω	Background noise
F_{b_j}	The CPU frequency of BS j
D_{t_i}	The input data size of task i
C_{t_i}	The required Cpu cycles of task i
m_{t_i,s_j}	The offloading decision of task i
$L^l_{t_i}$	The local processing latency of task t_i
$E^l_{t_i}$	The local processing energy consumption of task t_i
λ_{b_j}	Task arrival rate on BS j
μ_{b_j}	Service rate of BS j
$L^{off,s_j}_{t_i}$	The transmission latency of task t_i from user u_i to service s_j
L^{soj,s_j}	The average processing delay on service s_j
$E^{off,s_j}_{t_i}$	The transmission energy consumption of task t_i from user u_i to service s_j
Δ	Time frame length
β	Trade-off parameter between latency and energy consumption
H_{t_i}	Overhead of task t_i
$pre^{b_j}_{u_i}$	The preference from user u_i to BS b_j
$load_{b_j}$	The load of BS b_j
C_P	The crossover probability of GA
M_P	The mutation probability of GA

Problem formulation and solutions

In this section, the problem is formulated mathematically and the proposed solution is described in detail.

Problem formulation

The ultimate optimization goal of this work is to make an offloading decision for balanced latency and user energy consumption. For a single user, its possible offload options include local processing and offloading task to a reachable BS server for processing. k_i is used to represent the offloading decision of task t_i. If k_i is equal to 0, it indicates that the task is processed locally, otherwise, k_i indicates the BS server index of task offloading. Then, a set $K = \{k_1, k_2, \ldots, k_n\}$ is used to represent the offloading decisions of all users.

For a certain offloading decision k_i, the latency of task t_i can be calculated according to the following formula:

$$L_{t_i} = \begin{cases} L^l_{t_i} & \text{if } k_i = 0; \\ L^{off,s_{k_i}}_{t_i} + L^{soj,s_{k_i}} & \text{if } k_i \neq 0. \end{cases} \tag{9}$$

Similarly, the following formula can be used to calculate the energy consumption of t_i:

$$E_{t_i} = \begin{cases} E^l_{t_i} & \text{if } k_i = 0; \\ E^{off,s_{k_i}}_{t_i} & \text{if } k_i \neq 0. \end{cases} \tag{10}$$

Here, a user overhead H_{u_i} is introduced, which is equal to user task latency plus energy consumption. Since these two values are of similar magnitude, overhead is not normalized.

$$H_{t_i} = \beta L_{t_i} + (1 - \beta) E_{t_i}, \tag{11}$$

where β is a trade-off parameter. When β is close to 1, it pays more attention to latency performance. When β is close to 0, it pays more attention to energy consumption performance. Users can set it according to their own needs. Thus, the total overhead of all users can be calculated by following formula:

$$H_T = \sum_{t_i \in T} H_{t_i}. \tag{12}$$

The optimization objective of this paper is to reduce the total latency and energy consumption of users, which can be summarized as the following optimization objective function:

$$\begin{aligned} &\min \quad \text{object} \min_{M} H_T \\ &s.t. \quad \begin{cases} C1 : m_i \in \{0, 1, \ldots, m\}, \forall m_i \in M \\ C2 : \lambda_{s_j} < \mu_{s_j}, \quad \forall s_j \in S \end{cases} \end{aligned} \tag{13}$$

Constraint (1) indicates that offloading selection is an integer optimization problem, and constraint (2) indicates that the task arrival rate on the BS server is less than the processing rate. This problem is a very difficult nonlinear integer optimization problem [9], which motivates us to explore an efficient scheme to find suboptimal solutions.

User grouping strategy based on preferences

To reduce the complexity of the original problem, we first group the users. This means that the user will first select a specific BS as the destination for task offloading. Generally speaking, an intuitive solution is to assign mobile users to the nearest BS since a closer distance means better communication conditions. However, when the spatial distribution of mobile users is uneven, this solution will lead to an overload of BS for high user density. Therefore, this proposes the strategy of preference

first to allocate users. $pre_{u_i}^{b_j}$ is used to express the preference from user i to BS j, which is determined by the following formula:

$$pre_{u_i}^{b_j} = \frac{1}{\frac{d_{u_i}^{b_j}}{C_{b_j}} + \frac{load_{b_j}}{n}}, \tag{14}$$

where $d_{u_i}^{b_j}$ is distance between user i and BS j. C_{b_j} represents the signal coverage of BS j. The user device will request load data from the base station. Then $load_{b_j}$ indicates the load of BS j, that is, the number of users that have been allocated to it. n is the total number of users.

We traverse all users and use grouping strategy to decide where to offload its task, and here is a detailed description of the user grouping strategy based on preferences.

1. First, the user calculates the distance to all the reachable BSs and queries the workload of all BSs.
2. Second, the user calculates its preference for all reachable BSs according to Formula (14).
3. Finally, the user selects a BS offloading task with the greatest preference value.

Algorithm 1 is the pseudo-code of SPF.

Algorithm 1: User grouping strategy based on preferences

Input: User set: U
BS set: B
Output: User assignment results A

```
for user in U do
    Get reachable BS set B_reahable;
    for base in B_reahable do
        Calculate preferences pre_user^base
    Find the BS with the largest preference b_max
    A[user] ← b_max
return A;
```

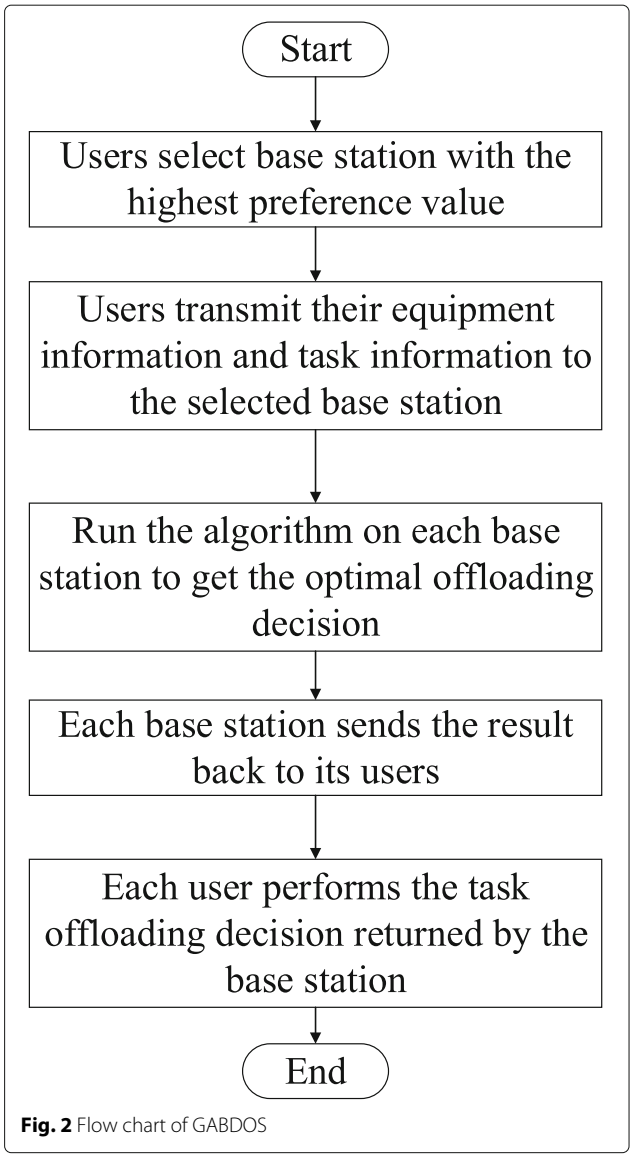

Fig. 2 Flow chart of GABDOS

Genetic algorithm based distributed offloading strategy

Mobile users need to select one of the reachable edge servers for task offloading. To obtain the optimal offloading option is a combinatorial explosion problem. In this paper, a genetic algorithm based distributed offloading strategy is proposed. The general idea of the strategy is as follows and Fig. 2 gives an intuitive flow chart of process.

- First, according to the user grouping strategy based on preferences (Algorithm 1), mobile users are assigned to the BS with the highest preference as a destination to offload tasks. The preference is determined by a tradeoff computation of both distance and workload. On the one hand, the closer the distance between the BS and the user, the larger the preference value is. On the other hand, the smaller the BS's workload, the larger the preference value.
- Then, the original problem will be decomposed into multi-user-to-single-server subproblems, which means that each BS is responsible for only a few users. In each group, users either process locally or offload tasks. So the subproblems are 0-1 selection problems. Because these subproblems are independent of each other, they can be processed in a distributed parallel.
- Finally, genetic algorithm (GA) is used on each edge server to solve the 0-1 selection problems and get near-optimal offloading decisions.

The optimization objective of this paper is modeled as a very complex 0-1 nonlinear optimization problem. Moreover, there is no polynomial-time complexity algorithm for this problem. Therefore, this paper uses a heuristic algorithm to reduce the complexity and obtain the approximate solution. GA is a method to search the optimal solution by simulating the natural evolution process [31], and it's widely used in optimization problems.

In this paper, due to the 0-1 selectivity of sub-problems, the binary coding is adopted to encode genes. Each chromosome represents the offloading decision of a sub-problem. If the gene of a user is 0, the task will be processed locally, and if 1, the task will be offloaded.

In the theory of evolution, fitness refers to the adaptability of an individual to the environment and the ability of the individual to reproduce. The fitness function of the GA is also called the evaluation function, which is used to judge the quality of individuals in the group. It is evaluated according to the objective function of the problem. The optimization goal of our model is to make the total latency and energy consumption of users as small as possible, so the following fitness function is adopted:

$$f = \frac{1}{H_T}. \tag{15}$$

The formula above shows that, the larger the total overhead, the lower the fitness.

Three operators in GA are as follows:

1 Selection operator: Roulette wheel selection is used in this paper. The basic idea is that the probability of each individual being selected is proportional to its fitness.
2 Crossover operator: As shown in Fig. 3, a two-point crossover operator is adopted, which refers to the random selection of two points in an individual gene, and then carry out gene exchange. The crossover probability is denoted as C_P. After extensive preliminary tests and convergence analysis, C_P is set to 0.5.
3 Mutation operator: Uniform mutation is adopted here, which means every gene is mutated according to a certain probability. The mutation probability is denoted as M_P. M_P is set to 0.001 according to preliminary test results.

The setting of C_P and M_P are detailed in Impact of trade-off parameter on GABDOS section Algorithm 2 is the pseudo-code of GA.

Time complexity analysis of proposed offloading strategy

Assuming there are N BSs and M users, the proposed offloading strategy can be divided into two parts. First, each user needs to traverse all the reachable base stations to select the base station with the highest preference. Therefore, the time complexity of this part is $O(N \times M)$. Then, each base station needs to calculate the optimal offloading decision. Since all base stations can process the step in parallel, the time complexity of a certain base station can represent the time complexity of this part. Since GA is adopted here, the time complexity of this part is mainly related to the parameters of GA. G is used to represent the generation of GA. P is used to represent the population.The size of the individuals is equal to the number of users. Therefore, the time complexity in this part can be summed up as $O(G \times P \times M)$. The time complexity of the latter part accounts for the main part of the whole

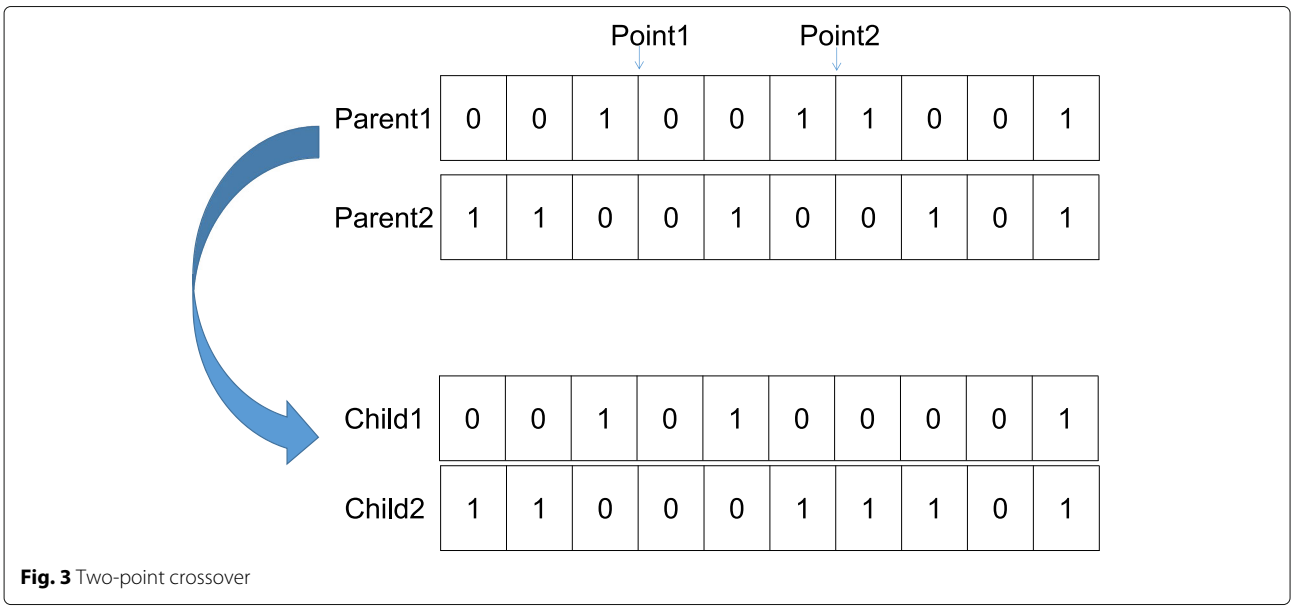

Fig. 3 Two-point crossover

Algorithm 2: Genetic Algorithm Based Distributed Offloading Strategy

Input: Initial population: P_0
Population size: N
The largest generation: G
The maximum fitness set of each generation population: $F = \{f_1, f_2, \ldots, f_N\}$
Maximum fitness difference threshold: K
Crossover probability: C_P
Mutation probability: M_p
Output: Optimal individual O

```
g ← 0;
F ← ∅;
while g < G do
    for i = 1 to N do
        Evaluate fitness of P_g;
        f_g ← The maximum fitness;
        O ← Maximum fitness individual;
    for i = 1 to N do
        Select operation to P_g;
    for i = 1 to N/2 do
        if random(0,1) < C_p then
            Crossover operation to P_t;
    for i = 1 to N do
        if random(0,1) < M_p then
            Mutation operation to P_g;
    if g ≠ 0 && f_g − f_{g−1} < K then
        break;
    P_{g+1} ← P_g;
    g ← g + 1;
return O;
```

algorithm. The convergence of GA has a great impact on its time complexity, so its convergence described in detail in the next section.

Performance evaluation

In this section, the impact of various indicators on user latency and energy consumption is tested, including user amount, the computing capacity, and the transmission power of user equipment. At the same time, the influence of the trade-off parameter β and the convergence of GA is also evaluated.

The following offloading strategies are used as benchmarks for this work:

1 All local processing strategy (ALP). In this extreme case, all tasks of the user are processed locally, which is the optimal solution for the minimum latency without considering users' energy consumption.
2 All offloading strategy (AOS), which means all the user tasks are offloaded to a random reachable edge server. Generally, offloading local tasks to the edge server will help reduce latency and save energy for users. However, due to the limited channel resources, if all users' tasks are offloaded, it is likely to cause serious signal interference and channel preemption, thus greatly reducing the efficiency of data transmission. Eventually, the energy consumption and latency of AOS will be greatly increased.
3 PROMOT [28], which is based on the Genetic technology as our work, is a priority offloading mechanism with joint offloading proportion and transmission (PROMOT) energy algorithm. It uses an offloading probability to determine if a task should be processed locally or remotely. The probability is computed from the prior information of past offloading feedbacks.

As shown in Fig. 4, the simulation is set in a $500m \times 500m$ square area. There are 9 BSs in total, and the plane positions of all users are randomly distributed. The basic parameters of the simulation are summarized in Table 2.

Performance for offloading efficiency

The proposed grouping strategy and GABDOS are compared with the ALP (referred to as the optimal solution for the lowest latency), the AOS and the PROMOT, in terms of the total overhead, the average latency and the energy consumption of user devices.

Impact of user amount on offloading efficiency

Figure 5 gives the overhead of different solutions with different user density. The overhead is a comprehensive measurement of user task latency plus energy consumption. It can be concluded that GABDOS has minimal overhead with various user density. This advantage is more and more obvious with the increase of the user amount compared with AOS. This is because the GABDOS can split the load of BS, while the PROMOT did not consider the workload variation of BSs. When the user amount is too large, AOS will lead to fierce channel resources and server resources preemption, thus significantly improving the latency and energy consumption of users. Compared to ALP, although GABDOS will cost user devices' energy consumption for offloading, it leads to an effective improvement in latency performance.

Figure 6 gives the latency performance. As expected, without considering energy consumption, ALP reaches the optimal latency for processing all tasks locally, which in practice can be used as the lower bound of latency. GABDOS performs very close to ALP which has the

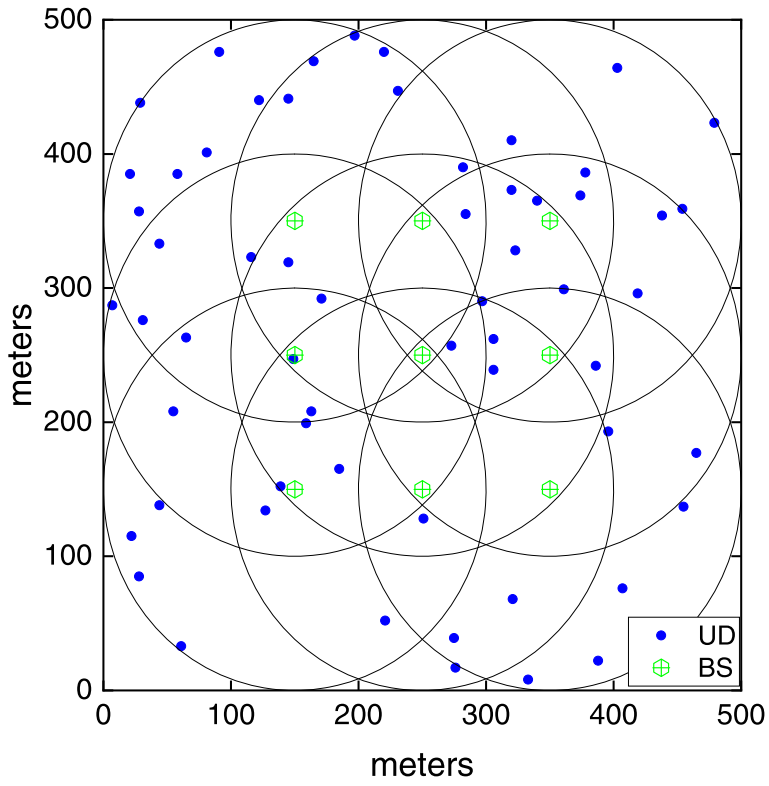

Fig. 4 Simulation setup

best latency performance, PROMOT places in the middle, while AOS performs the worst. This because when all users offload the task in AOS, they will interfere with each other and occupy limited channel resources. This will inevitably lead to a decrease in data transmission rate, thus increasing the transmission latency. At the same time, AOS will make the edge server overburdened and increase the task processing latency. PROMOT has two highest delay points at user amount 60 and 100, respectively. This may because when the user density exceeds some threshold, it spends time collecting the prior offloading information to refresh its offloading probability to make a further decision.

Table 2 Simulation parameters

Parameter	value
BS signal coverage	150m [8]
Edge server CPU frequency	$10GH_z$ [9]
BS bandwidth	$10MH_z$ [9]
User device transmission power	$0.5W$ [32]
Data volume of task	[600,1000]Kb [9]
Calculation cycles of task	[400,600]MegaCycles [9]
Energy consumption factor	5×10^{-27} [32]
Background noise	100dBm [9]
channel gain	140.7+36.7log10d[km] [33]
Time slot	5s
Trade-off parameter	0.5
Population size	100
The largest generation	500
Maximum fitness difference threshold	0.0001
C_P	0.5
M_P	0.001

Figure 7 shows the energy consumption performance of these strategies. When the user amount is less than 50, AOS has the best energy performance. Because, in this situation, the channel resources are relatively abundant, even if all users offload tasks, they can get a very high transmission rate. However, with the increase of user amount, the total energy consumption of AOS first exceeds that of GABDOS and PROMOT, and finally exceeds that of ALP. Because with the increase of the user amount, AOS will lead to a sharp decline in the channel transmission rate, thus greatly increasing the transmission energy consumption. Obviously, when the user amount is large, GABDOS has the best energy performance.

Impact of user device CPU frequency on offloading efficiency

In this section, the impact of the computing power of mobile devices on the total latency and energy consumption is tested.

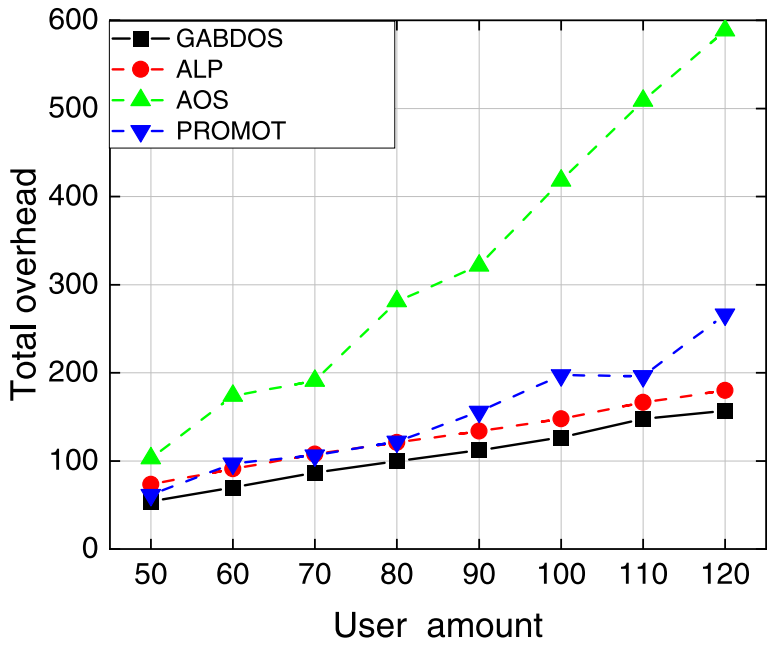

Fig. 5 Total overhead with different user amount

It can be concluded from Fig. 8 that as the computing capacity of mobile devices increases, the decreasing effect of average latency becomes smaller and smaller. The curve of GABDOS and ALP is very close, which shows that GABDOS has made full use of the latency performance gain brought by the increase of device CPU frequency. PROMOT is more insensitive to the increase of device CPU frequency since it makes the offloading decision based on prior probability. The reason why the decreasing effect of average latency becomes smaller with the computing capacity of mobile devices increasing will be discussed in our future work.

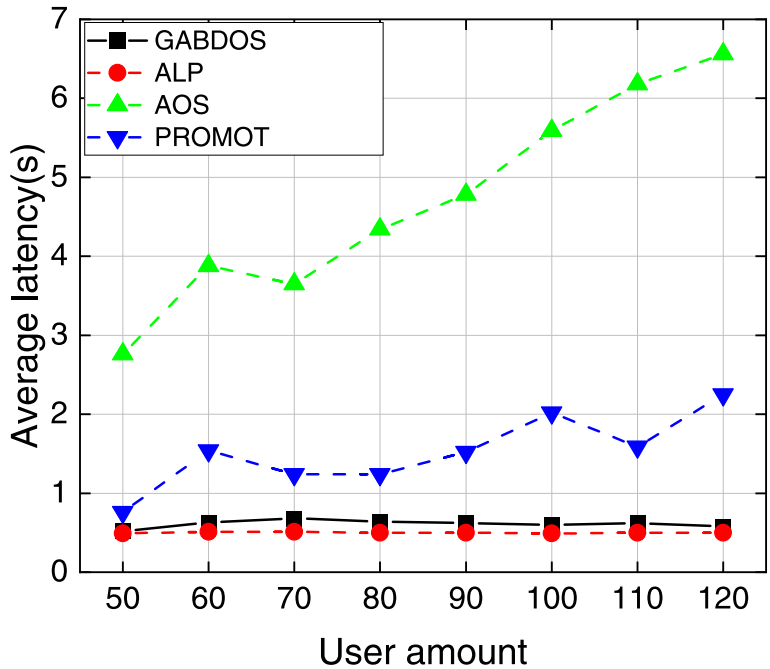

Fig. 6 Average latency with different user amount

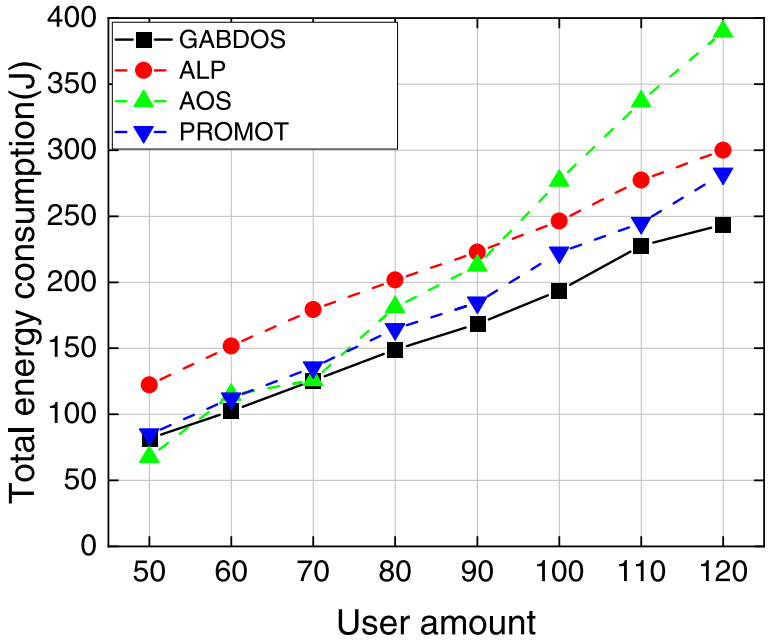

Fig. 7 Total energy consumption with different user amount

From Fig. 9, it can be told that all the curves except AOS are almost linear growth. GABDOS has the lowest energy consumption when the device CPU frequency is higher than 0.2 GHz. PROMOT consumes the most energy except AOS when the device CPU frequency is low. With the increase of device CPU frequency, PROMOT's energy consumption exceeds ALP. That's because the higher the device CPU frequency is, the higher the energy consumption of local processing tasks is. The above simulation results show that it is uneconomical to blindly increase the computing capacity of mobile devices.

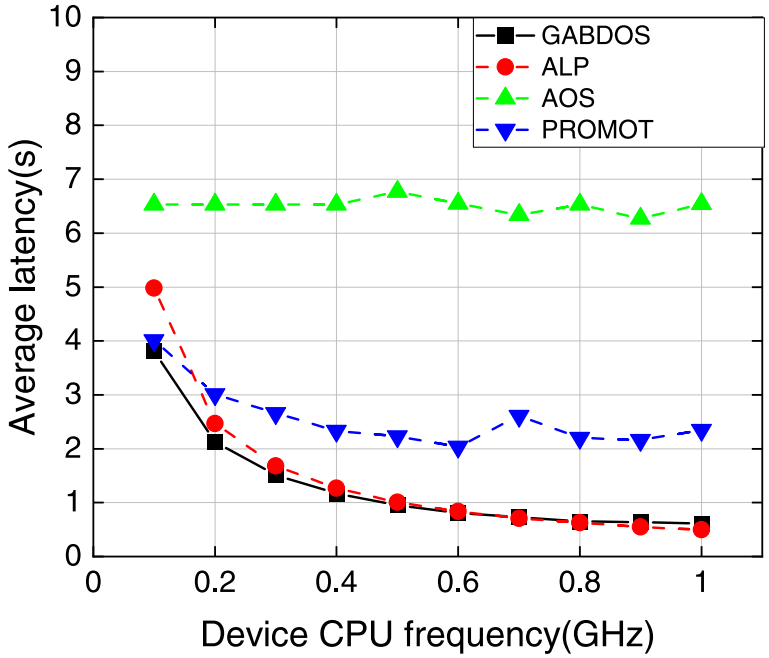

Fig. 8 Impact of user device CPU frequency on average latency

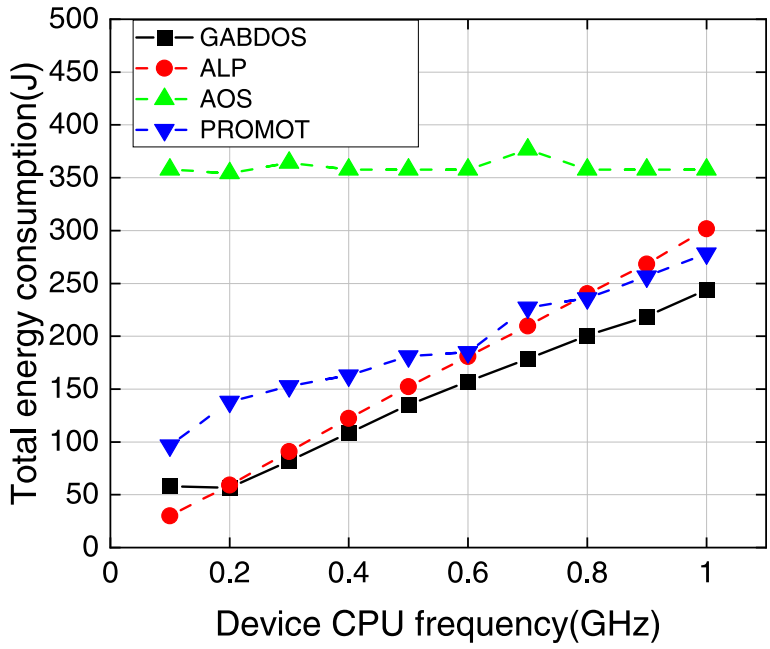

Fig. 9 Impact of user device CPU frequency on total energy consumption

Impact of trade-off parameter on GABDOS

The trade-off parameter β can be used to adjust the user's optimization preference. If the optimization latency is focused on, the β value can be set closer to 1. If the energy consumption optimization is focused on, the β value can be set closer to 0. For example, when the remaining power of the user's equipment is small and the task is not sensitive to latency, like image recognition, then the value of β can be set closer to 0. On the contrary, if the user's device is fully charged and many tasks are delay-sensitive, such as online games, β can be set closer to 1, so that the user can get a better

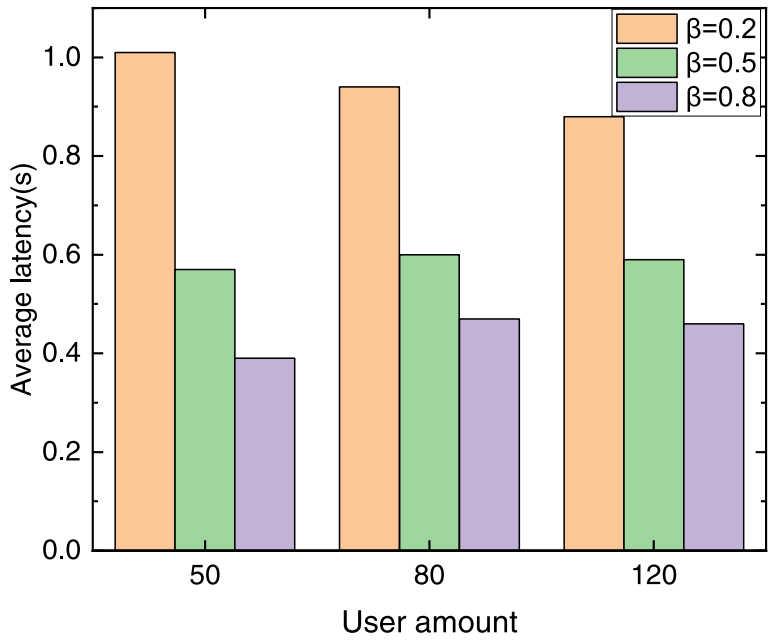

Fig. 10 Impact of β on average latency

experience. It can be concluded that the proposed GABDOS is flexible.

It can be concluded from Fig. 10 that with an increase of β, the average latency of users has a significant decline. From Fig. 11, it can be concluded that the total energy consumption of users increases with an increase of β. This means that adjusting β can effectively control the objective of the optimization.

Convergence analysis of GABDOS

Convergence is a matter of concern in GA. Since the proposed GABDOS is based on genetic algorithm, it is necessary to analyze the convergence of the algorithm.

Impact of crossover and mutation probability

It is known that the crossover probability C_P and the mutation probability M_P have a direct impact on the convergence of the genetic algorithm. The larger C_P is, the faster the new individuals will be generated. However, excessive C_P will lead to the destruction of the genetic model. On the contrary, if C_P is too small, the search process will be slow or even stagnant. For M_P, a small value will lead to the slow generation of new individuals, and it is easy for the algorithm to fall into the local optimum. A too-large M_P will make the algorithm become a pure random search algorithm.

To derive the optimal setting of C_P and M_P, the convergent generation and maximum fitness of the algorithm under different crossover and M_P are tested. As shown in Fig. 12, the algorithm converges extraordinary fast when M_P is small. However, through Fig. 13, it can be found out that the maximum fitness of the population is not optimal. It shows that the algorithm does not converge to the optimal solution, but falls into the local optimum. On the other hand, it is almost difficult for the algorithm to converge when M_P is set to a large number since the algorithm is close to random search. In the same way, we can see from Fig. 12 that the convergence of the algorithm is very slow when M_P is set very largely. Finally, according to Figs. 12 and 13 we confirm that the optimal crossover and M_P are 0.5 and 0.001 respectively.

Convergence of GA

In this subsection, we analyzed the maximum fitness in each generation when C_P and M_P were set to 0.5 and 0.001 respectively. The x-axis of Fig. 14 represents the generation of algorithm iteration, and the y-axis is maximum fitness in each generation of the population. The maximum fitness individual in the population corresponds to a specific offloading strategy. The larger its value is, the smaller the user's overhead is. Also, it can be concluded that the algorithm converges in the 50th generation. Moreover, according to the analysis of C_P and M_P, we can make sure that the algorithm has converged to the global optimum.

Evaluation summary

Among all the simulation benchmarks, the proposed GABDOS has the best performance in terms of delay and

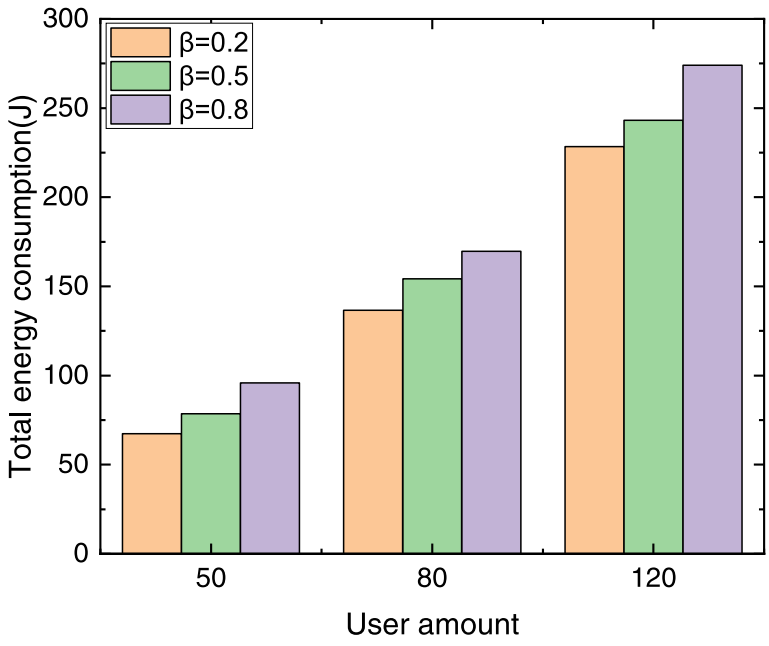

Fig. 11 Impact of β on total energy consumption

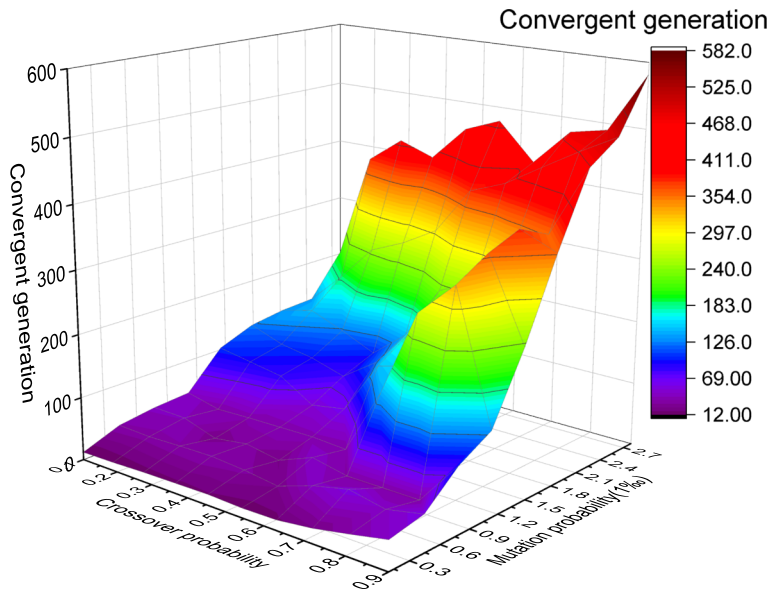

Fig. 12 Impact of C_P and M_P on convergent generation

energy consumption. In a 120-users network, compared with PROMOT, GABDOS reduces latency by 78% and energy consumption by 11%.

Conclusion

In this paper, a multi-user-to-multi-servers offloading decision (MUMS) problem is proposed for edge computing in ultra-dense 5G cellular networks. The MUMS problem is divided into two phases and conquered one by one. The first phase is to group users to BS according to a preference metric that considers both proximity and workload of the BS. After that, the original problem is decomposed into several sub-problems with 0-1 selectivity, and then a genetic algorithm named as GABDOS is designed to solve these sub-problems in parallel. It is verified that the proposed scheme can give a balanced offloading decision for low latency and user energy consumption.

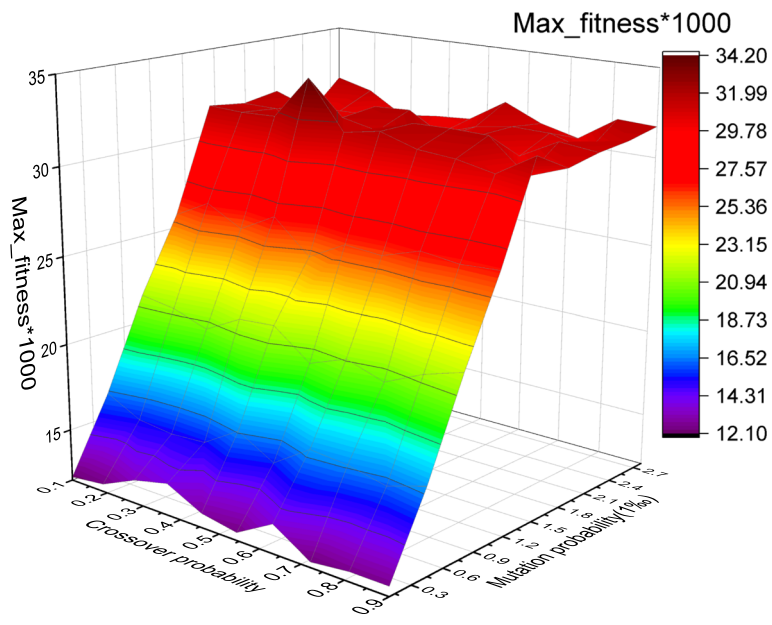

Fig. 13 Impact of C_P and M_P on maximum fitness

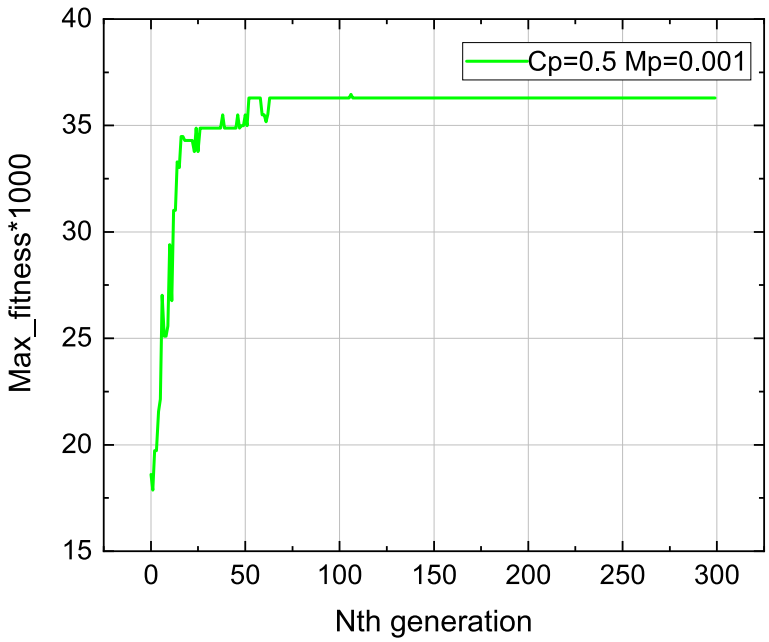

Fig. 14 Convergence curve of GA

Acknowledgements
We thank Central South University and Changsha University of Science and Technology for funding this work by the Degree & Postgraduate Education Reform Project of Hunan Province under Grant 2019JGZD057, and the National Science Foundation of China under Grant 61872387, 61972055.

Authors' contributions
Z. Liao carried out the experimental design, data analysis, interpretation, mathematical model design, and drafted the manuscript. J. Peng carried out the mathematical model design, B. Xiong participated in the experimental design, and J. Huang participated in conceptualization, implementation and approved the final manuscript.

Authors' information
Zhuofan Liao received the Ph.D degree in computer science from Central South University, China, in 2012. From 2017 to 2018, she worked as a Visiting Scholar in University of Victoria in Canada. She is currently an Assistant Professor in the School of Computer and Communication Engineering at Changsha University of Science and Technology, China. Her research interests includes wireless networks optimization, big data and edge computing for 5G. Email: zfliao@csust.edu.cn.
Jingsheng Peng is currently pursuing the master's degree with the Changsha University of Science and Technology. His research interests include mobile edge computing and big data placement optimization. He is good at C/C++ programming under Linux, and also familiar with Python. Email: jasonpeng@stu.csust.edu.cn.
Bing Xiong received the Ph.D. degree in Computer Science by master-doctorate program from Huazhong University of Science and Technology (HUST), China, in 2009. Supported by China Scholarship Council, he worked as a visiting scholar in University of Temple, USA, from 2018 to 2019. He is currently an associate professor in the School of Computer and Communication Engineering, Changsha University of Science and Technology, China. His main research interests include software-defined networking, network measurements, and network security. He is a member of IEEE and China Computer Federation. Email: xiongbing@csust.edu.cn.
Jiawei Huang received the bachelor's degree from the School of Computer Science, Hunan University, in 1999, and the master's and Ph.D. degrees from the School of Computer Science and Engineering, Central South University, China, in 2004 and 2008, respectively. He is currently a Professor with the School of Computer Science and Engineering, Central South University. His research interests include performance modeling, analysis, and optimization for wireless networks, and data center networks. Email: jiaweihuang@csu.edu.cn

Funding
This work has been institutionally supported by the Degree & Postgraduate Education Reform Project of Hunan Province under Grant 2019JGZD057, and the National Science Foundation of China under Grant 61872387, 61972055.

Availability of data and materials
There is no supporting data available.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]School of Computer and Communication Engineering, Changsha University of Science and Technology, Wanjiali South Road, 410114 Changsha, China. [2]School of Computer Science and Engineering, Central South University, Lushan South Road, 410083 Changsha, China.

Received: 1 September 2020 Accepted: 27 January 2021
Published online: 17 February 2021

References

1. Mach P, Becvar Z (2017) Mobile Edge Computing: A Survey on Architecture and Computation Offloading. IEEE Commun Surv Tutorials 19(3):1628–1656. https://doi.org/10.1109/COMST.2017.2682318
2. Hsieh H-C, Chen J-L, Benslimane A (2018) 5g virtualized multi-access edge computing platform for iot applications. J Netw Comput Appl 115:94–102. https://doi.org/10.1016/j.jnca.2018.05.001
3. Tomkos I, Klonidis D, Pikasis E, Theodoridis S (2020) Toward the 6g network era: Opportunities and challenges. IT Prof 22(1):34–38
4. Bhushan N, Li J, Malladi D, Gilmore R, Brenner D, Damnjanovic A, Sukhavasi RT, Patel C, Geirhofer S (2014) Network densification: the dominant theme for wireless evolution into 5g. IEEE Commun Mag 52(2):82–89
5. Du J, Yu FR, Chu X, Feng J, Lu G (2019) Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization. IEEE Trans Veh Technol 68(2):1079–1092. https://doi.org/10.1109/TVT.2018.2883156

6. Sun H, Zhou F, Hu RQ (2019) Joint Offloading and Computation Energy Efficiency Maximization in a Mobile Edge Computing System. IEEE Trans Veh Technol 68(3):3052–3056. https://doi.org/10.1109/TVT.2019.2893094
7. Qi Q, Wang J, Ma Z, Sun H, Cao Y, Zhang L, Liao J (2019) Knowledge-Driven Service Offloading Decision for Vehicular Edge Computing: A Deep Reinforcement Learning Approach. IEEE Trans Veh Technol 68(5):4192–4203. https://doi.org/10.1109/TVT.2019.2894437
8. Poularakis K, Llorca J, Tulino AM, Taylor I, Tassiulas L (2019) Joint Service Placement and Request Routing in Multi-cell Mobile Edge Computing Networks. Proc IEEE INFOCOM 2019-April:10–18. https://doi.org/10.1109/INFOCOM.2019.8737385
9. Tran TX, Pompili D (2019) Joint task offloading and resource allocation for multi-server mobile-edge computing networks. IEEE Trans Veh Technol 68(1):856–868. https://doi.org/10.1109/TVT.2018.2881191
10. Li G, Cai J (2020) An online incentive mechanism for collaborative task offloading in mobile edge computing. IEEE Trans Wirel Commun 19(1):624–636. https://doi.org/10.1109/TWC.2019.2947046
11. Li C, Tang J, Luo Y (2019) Dynamic multi-user computation offloading for wireless powered mobile edge computing. J Netw Comput Appl 131:1–15. https://doi.org/10.1016/j.jnca.2019.01.020
12. Bai T, Wang J, Ren Y, Hanzo L (2019) Energy-efficient computation offloading for secure uav-edge-computing systems. IEEE Trans Veh Technol 68(6):6074–6087. https://doi.org/10.1109/TVT.2019.2912227
13. Alghamdi I, Anagnostopoulos C, Pezaros DP (2019) On the optimality of task offloading in mobile edge computing environments. In: 2019 IEEE Global Communications Conference (GLOBECOM), Hawaii. pp 1–6. ISBN 9781728109626. https://doi.org/10.1109/GLOBECOM38437.2019.9014081
14. Guo K, Yang M, Zhang Y, Jia X (2019) Efficient resource assignment in mobile edge computing: A dynamic congestion-aware offloading approach. J Netw Comput Appl 134:40–51. https://doi.org/10.1016/j.jnca.2019.02.017
15. Huang M, Liu W, Wang T, Liu A, Zhang S (2020) A cloud–mec collaborative task offloading scheme with service orchestration. IEEE Internet Things J 7(7):5792–5805. https://doi.org/10.1109/JIOT.2019.2952767
16. Huang S, Liu A, Zhang S, Wang T, Xiong N (2020) Bd-vte: A novel baseline data based verifiable trust evaluation scheme for smart network systems. Trans Netw Sci Eng I:1–1. https://doi.org/10.1109/TNSE.2020.3014455
17. Qiu X, Liu L, Chen W, Hong Z, Zheng Z (2019) Online Deep Reinforcement Learning for Computation Offloading in Blockchain-Empowered Mobile Edge Computing. IEEE Trans Veh Technol 68(8):8050–8062. https://doi.org/10.1109/tvt.2019.2924015
18. Lu H, Gu C, Luo F, Ding W, Liu X (2020) Optimization of lightweight task offloading strategy for mobile edge computing based on deep reinforcement learning. Futur Gener Comput Syst 102:847–861. https://doi.org/10.1016/j.future.2019.07.019
19. Gao M, Cui W, Gao D, Shen R, Li J, Zhou Y (2019) Deep neural network task partitioning and offloading for mobile edge computing. In: 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa. pp 1–6. https://doi.org/10.1109/GLOBECOM38437.2019.9013404
20. Ge J, Liu B, Wang T, Yang Q, Liu A, Li A Q-learning based flexible task scheduling in a global view for the internet of things. Trans Emerg Telecommun Technol:4111. https://doi.org/10.1002/ett.4111
21. Miao Y, Wu G, Li M, Ghoneim A, Al-Rakhami M, Hossain MS (2020) Intelligent task prediction and computation offloading based on mobile-edge cloud computing. Futur Gener Comput Syst 102:925–931. https://doi.org/10.1016/j.future.2019.09.035
22. Liu X, Song H, Liu A (2020) Intelligent uavs trajectory optimization from space-time for data collection in social networks. Trans Netw Sci Eng I:1–1. https://doi.org/10.1109/TNSE.2020.3017556
23. Xu X, Li Y, Huang T, Xue Y, Peng K, Qi L, Dou W (2019) An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks. J Netw Comput Appl 133:75–85. https://doi.org/10.1016/j.jnca.2019.02.008
24. Goudarzi M, Zamani M, Toroghi?Haghighat A (2017) A genetic-based decision algorithm for multisite computation offloading in mobile cloud computing. Int J Commun Syst 30(10):1–13. https://doi.org/10.1002/dac.3241
25. Guo Y, Mi Z, Yang Y, Obaidat MS (2019) An energy sensitive computation offloading strategy in cloud robotic network based on ga. IEEE Syst J 13(3):3513–3523
26. Du C, Chen Y, Li Z, Rudolph G (2019) Joint optimization of offloading and communication resources in mobile edge computing. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), Xiamen. pp 2729–2734. https://doi.org/10.1109/SSCI44817.2019.9003099
27. Kuang L, Gong T, OuYang S, Gao H, Deng S (2020) Offloading decision methods for multiple users with structured tasks in edge computing for smart cities. Futur Gener Comput Syst 105:717–729. https://doi.org/10.1016/j.future.2019.12.039
28. Wang J, Wu W, Liao Z, Simon Sherratt R, Kim G, Alfarraj O, Alzubi A, Tolba A (2020) A probability preferred priori offloading mechanism in mobile edge computing. IEEE Access 8:39758–39767
29. Mao Y, Zhang J, Letaief KB (2016) Dynamic computation offloading for mobile-edge computing with energy harvesting devices. IEEE J Sel Areas Commun 34(12):3590–3605
30. Tran TX, Pompili D (2019) Joint task offloading and resource allocation for multi-server mobile-edge computing networks. IEEE Trans Veh Technol 68(1):856–868. https://doi.org/10.1109/TVT.2018.2881191
31. Goldberg DE (1989) Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley Professional
32. Yang Y, Ma Y, Xiang W, Gu X, Zhao H (2018) Joint optimization of energy consumption and packet scheduling for mobile edge computing in cyber-physical networks. IEEE Access 6:15576–15586. https://doi.org/10.1109/ACCESS.2018.2810115
33. Chu X, Lopez-Perez D, Yang Y, Gunnarsson F (2013) Heterogeneous Cellular Networks Theory, Simulation and Deployment. Cambridge University Press. ISBN-13: 9781107023093

Publisher's Note