Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.081575

ARTICLE

An Enhanced Genetic Algorithm via an Innovative Elite Retention Strategy for Task Offloading in MEC Scenarios

Chengyu Hou[1,2], **Wenzao Li**[2], **Hanyun Li**[3], **Kui Liu**[1], **Zhuoning Zhao**[1] **and Hongping Shu**[1,*]

[1]School of Software Engineering, Chengdu University of Information Technology, Chengdu, China
[2]School of Communication Engineering, Chengdu University of Information Technology, Chengdu, China
[3]Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, China

*Corresponding Author: Hongping Shu. Email: cqshp@cuit.edu.cn

Received: 07 March 2026; Accepted: 12 May 2026; Published: 15 June 2026

ABSTRACT: The rapid growth of Internet of Things (IoT) and 5G technologies has led to a sharp increase in computing demands from wireless devices, making efficient task offloading a critical challenge. Key issues include reducing application latency, lowering the energy consumption of terminal devices, and improving overall system performance, all of which directly affect user experience. Traditional genetic algorithms (GA), inspired by biological evolution, have been widely used in task offloading, but they often suffer from slow convergence and a tendency to fall into local optima in complex scenarios, limiting their effectiveness. To address these drawbacks, this paper proposes a task offloading strategy based on a refined elite mechanism in a GA. The algorithm introduces multi-point variation in both crossover and mutation operations to enhance population diversity, avoid local optima, and accelerate convergence. This design leverages the GA's strength in multi-objective optimization, which outperforms other bionic heuristic algorithms that excel in single domains. Comparative experiments with GA, ant colony optimization, Deep Q-Network, Greedy algorithms, simulated annealing algorithm and particle swarm optimization, show that the proposed algorithm improves convergence speed by 35%, reduces task completion time by 6%, and optimizes energy consumption by approximately 18%.

KEYWORDS: Task offloading; genetic algorithm; bandwidth constraint; mobile edge computing

1 Introduction

1.1 Background and Motivation

The popularization and rapid development of intelligent applications have imposed increasingly stringent requirements on energy efficiency, real-time response, and data transmission. In parallel, important projects such as Smart Cities (SCs) are being widely discussed and deployed, which further accelerates the growth of data-intensive services and computing demands [1]. Under such circumstances, traditional cloud computing is increasingly challenged by the massive communication and computation pressure generated by large-scale networks. Fortunately, edge cloud computing (ECC) in 5G scenarios provides a promising solution to this problem [2]. By offloading computation tasks to nearby servers, mobile terminals can reduce local computation burden and shorten service response time, while mobile edge computing (MEC) can also alleviate transmission pressure on the backbone network. However, due to the limited bandwidth of wireless channels and the constrained resources of edge servers, the efficiency of task offloading still varies significantly across different schemes. In practical scenarios, mobile terminal devices, such as

smartphones, tablets, and smart vehicles, usually have certain local computing capabilities and can execute part of the workload locally. Therefore, under limited communication and computation resources, designing an effective task offloading strategy is of great importance for improving overall system performance. In multi-server multi-access MEC environments, existing studies on data offloading generally focus on balancing multiple factors, including latency, energy consumption, communication resource allocation, and transmission reliability under dynamic wireless conditions. More specifically, risk-aware data offloading methods can be roughly divided into three categories. The first category is reliability-aware or channel-aware offloading, which mainly considers channel uncertainty, link reliability, bandwidth competition, and transmission failure risk. The second category is queue-aware or workload-aware offloading, which focuses on server workload, task congestion, queuing delay, and resource competition among multiple users. The third category is learning-based risk-aware offloading, where reinforcement learning or deep reinforcement learning methods are used to adaptively learn offloading decisions in dynamic MEC environments. Different from the above studies, this paper does not address risk awareness only from a single aspect, such as link reliability, queueing delay, or dynamic learning policy. Instead, the proposed method integrates transmission distance, maximum transmission rate, local computing time, energy consumption, and task offloading quantity into a unified fitness function. In this way, the potential risk caused by wireless transmission conditions and constrained bandwidth is reflected in the optimization objective. Furthermore, an enhanced multi-point genetic algorithm with elite retention is adopted to search for an efficient offloading strategy under limited communication and computation resources. Therefore, the main difference of this work lies in combining a transmission-condition-aware fitness function with an improved GA-based optimization strategy, which provides a lightweight and efficient solution for task offloading in bandwidth-constrained MEC scenarios. Therefore, risk-aware data offloading has become an important consideration in practical MEC systems, especially in scenarios where service continuity and timely task execution are required.

In MEC, the efficiency of optimization algorithms is critical due to limited computational resources and latency constraints [3]. Traditional genetic algorithms, especially those relying on single-point mutation and crossover operations, are not well-suited for such environments. These methods usually exhibit slow convergence, which may introduce additional delay and thus reduce their applicability to real-time edge services. Single-point mutation, in particular, changes only a small portion of a chromosome, resulting in limited population diversity and insufficient exploration of the solution space. To address this issue, this paper adopts multi-point mutation and crossover operations to improve the search efficiency of the genetic algorithm. Compared with conventional single-point operations, this design introduces richer population variation and accelerates convergence, making it more suitable for task offloading optimization in resource-constrained MEC environments. Accordingly, the difference between our work and existing risk-aware offloading studies lies mainly in the perspective and solution strategy adopted in this paper. Rather than focusing only on the trade-off between computation and communication costs, we emphasize the influence of wireless transmission conditions on offloading efficiency in a multi-server multi-access edge environment, and solve the corresponding optimization problem through an enhanced genetic algorithm with multi-point mutation and crossover. In this way, the proposed method is intended to provide a more efficient optimization process for offloading decision-making under constrained edge resources.

1.2 Challenges and Solutions

Task offloading strategies face three main challenges, in the case of edge cloud computing. The first is to assess the overall value of the computing task. Due to limited wireless channel resources shared among mobile users, the task offloading strategy should not only aim to save the computing resources of mobile devices to the maximum extent but also meet as many computing requests as possible to improve user

experience [4]. The second challenge is ensuring the versatility and practicality of task offloading strategies. Actual task offloading problems need to consider as many optimization indicators as possible, such as energy consumption, time delay, and the number of completed user requests [5]. Integrating multiple optimization goals simultaneously to achieve a good offloading strategy adds complexity to the task. The third challenge is finding optimal solutions to non-convex optimization problems. This optimization strategy is a typical NP-hard problem, and finding a solution that performs better than existing solutions with a small computational cost is a significant challenge [6].

To address the aforementioned challenges, this paper proposes a multi-point genetic algorithm named MP-GA for task offloading in MEC scenarios. In MP-GA, the generalized cost function is composed of transmission delay caused by transmission distance, local computing time, energy consumption, and task offloading volume. It not only considers the value of task offloading processing but also comprehensively takes into account the task offloading rate and user experience. The simulation results verify the effectiveness of MP-GA in task offloading scenarios.

1.3 Contributions and Organization

Existing studies on MEC offloading mostly concentrate on delay and energy metrics, while the effects of transmission distance and user-side service perception are less discussed in bandwidth-limited practical scenarios [7]. To address this issue, this work constructs a new cost function and uses an improved genetic algorithm to search for a better offloading policy. The main contributions are summarized below:

- First, a broader evaluation model is developed by jointly considering local computing energy, offloading-related factors, and the opportunistic network setting, so that the model can be applied to a wider range of scenarios. In addition, the Opportunistic Network Environment simulator is introduced to mimic the spatial distribution of mobile terminals in realistic environments, which makes the simulation setting closer to actual deployment conditions.
- Secondly, to address the task offloading problem characterized by non-convex optimization and NP-hard issues, this study employs four heuristic algorithms to find optimal solutions and proposes a multi-point genetic algorithm with an elite preservation mechanism for MEC task offloading.
- Finally, the proposed MP-GA is evaluated against conventional GA, PSO, Ant Colony Optimization, Deep Q-Network, Greedy algorithms, and Simulated Annealing through simulation experiments. The results are used to examine both the effectiveness of the method and its convergence behavior.

The organization of this paper is as follows. First, the next section summarizes the related work. Next, the subsequent section introduces the system model and computational model for task offloading. Subsequently, the following section proposes an improved genetic algorithm and elaborates on its application in task offloading scenarios. After that, the section that comes next presents the simulation environment setup and discusses the simulation results. Finally, the concluding section provides the research conclusions.

2 Related Work

In the traditional cloud computing model, all computing tasks are usually sent to remote data centers for processing [8]. However, this method has some key problems: first, the delay in data transmission may significantly affect the response time of applications, especially in scenarios with high real-time requirements; second, long-term data transmission and processing can lead to high energy and bandwidth consumption. Additionally, as the number of devices increases, the load on cloud servers also rises, which affects the performance and reliability of the overall system [9]. To address these issues, edge computing has emerged. Edge computing moves computing and storage resources from centralized data centers to the edge of the

network, close to the data source [10]. By performing data processing and calculations closer to users, edge computing can significantly reduce latency, improve application responsiveness, and alleviate the burden on data centers [11,12].

In a study by Wei and Liang, a task offloading strategy for the Internet of Vehicles (IoV) based on a queuing network was designed to optimize delay [13]. Similarly, Alsadie proposed a hybrid task offloading method (HybridTO) that integrates the Grey Wolf Optimizer and particle swarm optimization to optimize energy consumption by considering various factors such as proximity constraints and latency requirements [14]. Both of these solutions focus only on optimizing either energy consumption or time delay, without considering them comprehensively. Shen et al. proposed an algorithm using FDMA golden section search to obtain subsequent optimal time allocation [15]. Their research considers the fact that transmission resources are shared among mobile users. Their mobile edge computing offloading strategies aim not only to maximize overall network performance, but also to balance the optimal solution, to satisfy all mobile users. Subsequently, Chen et al. identified the user's quality of experience (QoE) as the goal for developing an offloading strategy [16]. Peng et al. proposed a multi-UAV-assisted integrated sensing, communication, and computation framework, and formulated the joint optimization problem as a multi-agent Markov decision process, which is solved using a multi-agent proximal policy optimization (MAPPO) algorithm with attention mechanism [17]. Li et al. investigated energy efficiency optimization in UAV-assisted mobile edge computing networks and developed a deep reinforcement learning-based approach, where a proximal policy optimization (PPO) algorithm is employed to jointly optimize task offloading and resource allocation [18]. Wang et al. proposed a two-stage offload decision framework with request holding and dynamics, based on short-term memory (LSTM) task offload request prediction and MEC resource release estimation, to infer the probability that the request will be accepted in subsequent times [19]. Chen et al. proposed the Task Offloading Decision Optimizer, which provides an efficient multi-target offloading scheme that takes into account real-time device workloads and user preferences [20]. Shuai et al. proposed a Q-learning-based reinforcement learning offloading strategy, which considers both the time-sensitive constraints and data requirements of computationally intensive tasks, reduces the overall communication delay and energy consumption, and thus improves the offloading success rate [21]. However, the influence of wireless transmission distance was not fully addressed when mobile terminals were distributed unevenly.

Taken together, previous studies indicate that task offloading in MEC is a multi-factor optimization problem [22], and each method tends to emphasize certain objectives while leaving some issues unresolved. Based on this observation, we build a more inclusive offloading strategy. The fitness function in this study jointly accounts for energy use, delay, the number of offloaded tasks, and the achievable transmission rate of each task. Meanwhile, a multi-point genetic algorithm with an elite retention mechanism is adopted to solve the problem under the considered setting. Simulation results show that, relative to conventional approaches such as standard genetic algorithms and particle swarm optimization, the proposed MP-GA converges more steadily and achieves better overall performance in multi-index optimization as well as bandwidth utilization. Table 1 lists the differences between representative existing studies and our method. In summary, earlier work has not fully handled three key issues at the same time. By comparison, our study provides a more integrated solution. We focus on offloading optimization in intelligent community scenarios, with the aim of reducing backbone-network data transmission, lowering delay, and improving user experience. More specifically, the proposed framework jointly considers latency, energy consumption, offloading quantity, and maximum task transmission rate. It also addresses a single-base-station setting with multiple terminals and multiple tasks under limited bandwidth.

Table 1: Comparison of existing offloading approaches and the proposed strategy.

References	Technique	Objectives	Weakness
Wei and Liang [13]	Queuing Network Model	Delay	Energy consumption has not been considered
Alsadie [14]	HybridTO	Delay/Energy	High complexity
Shen et al. [15]	Golden section search for FDMA	Delay/Complexity	Single resource constraint
Chen et al. [16]	GDTO	QoE/Delay	Low convergence
Wang et al. [19]	TSODF	Delay/Energy	Transmission bandwidth is not considered
Chen et al. [20]	TODO	Energy/Delay/Cost	Restricted mobility
Shuai et al. [21]	Q-learning	Task offloading rate/Energy	Single-user MEC scenario
Proposed model	MP-GA	Energy/Delay/ Bandwidth/Offloading	Adaptivity

3 System Model

The communication framework considered in this paper is composed of mobile terminals, base stations, and edge cloud servers. The terminal side includes smartphones, tablets, and other intelligent devices.

3.1 System Model of Communication

The communication system model consists of mobile terminals, base stations, and edge cloud servers, as shown in Fig. 1. Mobile terminals include smartphones, tablets, and other smart devices. Each edge device is a user.

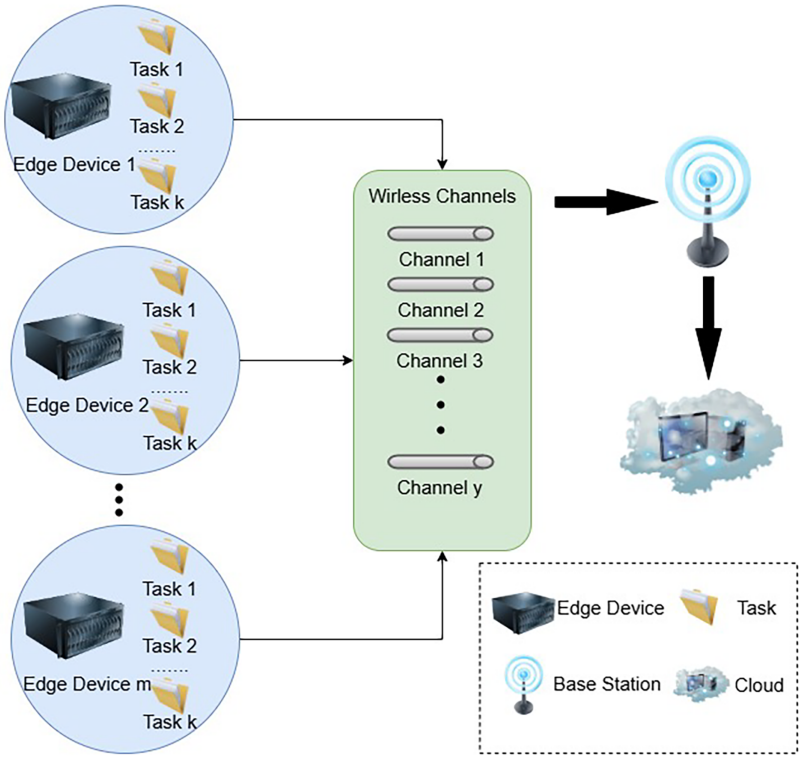

Figure 1: The system model of communication in the MEC scenario.

The figure shows that there are m users, and each user has k independent tasks. These tasks can be uploaded to the cloud via y wireless channels or computed locally. The system model in this paper is built on existing classic communication and computing models, but tailored to the multi-terminal, multi-task offloading scenario under bandwidth constraints in MEC. Although the equations in local computing time/energy and offloading transmission rate are widely used in the existing literature, they are re-derived here to clarify the integration logic with the proposed cost function (Eq. (11)). Specifically, these equations form the basis for quantifying key indicators such as task offloading value, transmission delay, and energy consumption, which are critical for constructing the comprehensive fitness function of MP-GA. Omitting these derivations would break the logical chain between the system model and the algorithm design.

3.2 Local Computing Model

The calculation of each task can be performed locally or offloaded to the base station for cloud computing. This section introduces both the overhead of local computation and offloading to the base station computation. The time and energy consumption required for task computation are key indicators to determine the value of task offloading. We use common calculation methods [23]. Assuming that there are k tasks to be executed on device i, c_i represents the number of cycles per second that the mobile terminal CPU runs and d_j is the number of CPU revolutions required to complete task j. Then, the local computing time of device i can be expressed as Eq. (1):

$$t_i = \sum_{j=1}^{k} \frac{d_j}{c_i} \tag{1}$$

Here, t_i denotes the time required for terminal i to complete all tasks locally. The corresponding energy consumption of terminal i (denoted by e_i) is given in Eq. (2):

$$e_i = \sum_{j=1}^{k} d_j J_i \tag{2}$$

where J_i denotes the energy coefficient consumed by device i per CPU cycle. If there are m mobile terminals in the system, the total local execution time and total local energy consumption of all tasks can be calculated by Eqs. (3) and (4), respectively:

$$t_{\text{tol}} = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_j}{c_i} \tag{3}$$

$$e_{\text{tol}} = \sum_{i=1}^{m} \sum_{j=1}^{k} d_j \cdot J_i \tag{4}$$

According to the above expressions, the total execution time and total energy consumption of mobile terminals increase with the number of tasks processed locally and the CPU cycles required by these tasks. In this paper, the time and energy costs of task execution are used to define the offloading value, as shown in Eq. (5):

$$v_j = \lambda_t \frac{d_j}{c_i} + \lambda_e d_j J_i, \quad \lambda_e + \lambda_t = 1 \tag{5}$$

In Eq. (5), v_j represents the offloading value of task j, and λ_t and λ_e are both weight coefficients that can be modified. Determining how to choose high-value tasks to offload is a matter of opinion. If the user

is more concerned about the terminal's capability resources, the user will set more weights to λ_e. That is, $\lambda_e > \lambda_t$, $\lambda_e \in (0,1)$. The reverse is also true.

3.3 Offloading Computing Model

In actual situations, channel resources are usually shared by multiple users. Resource allocation based on an Orthogonal Frequency-Division Multiple Access (OFDMA) heterogeneous network divides wireless bandwidth into multiple channels, and each sub-channel S provides the maximum transmission rate for users on the premise of satisfying the reliability of channel performance.

Each sub-channel is assigned an independent bandwidth B_s. According to Shannon's theorem, the maximum transmission rate C_s of a channel is determined by its bandwidth B_s and Signal-to-Noise Ratio (SNR). As shown in Eq. (6), C_s denotes the communication capacity, B_s is the bandwidth, and N_a and S_a represent the average noise power and signal power during transmission, respectively. To characterize signal attenuation during transmission, a simplified free-space path loss model is introduced, as shown in Eq. (7):

$$C_s = B_s \cdot \log_2\left(1 + \frac{S_a}{N_a}\right) \tag{6}$$

$$S_{PL}(\mathrm{dB}) = 20\log_{10}(d) + 20\log_{10}(f) + 32.45 \tag{7}$$

Eq. (7) is a simplified free-space path loss model for engineering applications. The constant 32.45 arises from unit conversions and logarithmic operations: when the frequency f is in MHz and the distance d in km, substituting these units into the decibel-form free-space path loss formula

$$\mathrm{FSPL(dB)} = 20\log_{10}\left(\frac{4\pi f d}{c}\right),$$

(where $c = 3 \times 10^8$ m/s is the speed of light)—with unit conversions (MHz to Hz via $\times 10^6$, km to m via $\times 10^3$) and logarithmic simplification—yields

$$20\log_{10}\left(\frac{4\pi \times 10^9}{c}\right) \approx 32.45.$$

This constant ensures that path loss is calculated in decibels (dB), which is consistent with standard practice in wireless communication engineering. For simplicity, this study assumes that each mobile terminal has the same signal transmitting power S_0, and models the attenuation of signal power during transmission as an approximately linear function. Therefore, the average signal power during transmission can be written as Eq. (8):

$$S = 0.5 \cdot [S_0 + (S_0 - S_P)] \tag{8}$$

where S_0 is the initial signal power, and S_P is the power after path loss (derived from $S_{PL}(\mathrm{dB})$). The assumption that all mobile terminals share the same transmit power S_0 is a simplification to reduce model complexity in initial analysis. In practice, terminal transmit power may vary due to hardware differences or adaptive power control. However, this assumption is reasonable for scenarios where terminals are homogeneous (e.g., same smartphone models) or when power differences are negligible compared to path loss. If this condition does not hold, Eq. (8) can be revised to

$$S(d_i) = S_{0,i} - \alpha\, d_i - L_0$$

where $S(d_i)$ denotes the received signal power at distance d_i and depends on the terminal position, $S_{0,i}$ is the individual transmit power of terminal i, α represents the path loss coefficient, i.e., the attenuation factor per unit distance, d_i is the distance between terminal i and the base station, and L_0 accounts for fixed losses such as feeder loss and shadowing fading.

This extension will be considered in future work to improve the realism of the model. Based on the above analysis, the maximum data transfer rate achievable for different offloaded tasks can be obtained. It should be noted that the above assumptions mainly aim to balance modeling realism and analytical tractability. Although identical terminal transmit power and simplified signal attenuation may not fully reflect all practical wireless conditions, they preserve the essential relationship between channel quality and transmission capability, which is sufficient for capturing the main impact of communication factors on task offloading decisions.

3.4 Problem Formulation

In this study, the location of user terminal i is denoted by $P_i = (x_i, y_i)$, and the location of the base station is denoted by $p_v = (x_v, y_v)$. The shortest distance $dist_{iv}$ between terminal i and base station v is calculated by Eq. (9):

$$dist_{iv} = \sqrt{(x_i - x_v)^2 + (y_i - y_v)^2} \tag{9}$$

Because the available bandwidth is limited, only part of the tasks can be offloaded to the base station. We use 0 to denote local computing and 1 to denote offloaded computing, so the state of each task can be written as $s_{iq} \in \{0, 1\}$, $i = 1, 2, \ldots, m$, $q \in 1, 2, \ldots, k$. Therefore, the constraint of the offloading policy is given by Eq. (10):

$$\sum_{i=1}^{m} \sum_{q=1}^{k} s_{iq} \cdot r_{iq} \leq B_{\text{tol}} \tag{10}$$

Here, B_{tol} denotes the total amount of available wireless bandwidth in the considered scenario. Based on this constraint, we define the task offloading cost function F_c by jointly considering task offloading value, user experience, and the maximum offloading rate, as shown in Eq. (11):

$$F_c = \sum_{i=1}^{m} \sum_{q=1}^{k} s_{iq} \cdot \left[\lambda_v \left(\lambda_t t_{iq} + \lambda_e e_{iq}\right) + \lambda_c c_{iq} + \lambda_n\right] \tag{11}$$

Here, λ_v, λ_n, and λ_c correspond to the weights of task offloading value, user experience degree, and maximum transmission rate, respectively, and c_{iq} is the maximum transmission rate of the task obtained by combining Eqs. (6)–(9). The variables s_{iq}, t_{iq}, and e_{iq} correspond to the q-th process of the i-th user. Up to this point, the cost function F_c has been constructed to evaluate the performance of the solutions generated by different algorithms. (If the user wants to prioritize the transmission of high-value tasks, increase λ_v; if the user wants to ensure low latency, increase λ_n; if the user wants to upload the maximum number of tasks, increase λ_c).

3.5 Motivation for Choosing Genetic Algorithms

Meta-heuristic algorithms perform well in solving NP-hard offloading problems in MEC, but each type has unique strengths. Particle swarm optimization (PSO) converges quickly but tends to fall into

local optima [24]; ant colony optimization (ACO) excels in path optimization but has high computational complexity [25]; and simulated annealing (SA) is robust but converges slowly [26].

Genetic algorithms (GAs) are chosen here for three reasons: (1) Strong global search capability: A GA's crossover and mutation operations effectively explore the solution space, critical for multi-objective optimization (latency, energy, bandwidth) in this study. (2) Flexibility in handling constraints: GAs can easily integrate bandwidth constraints (Eq. (10)) into the fitness function, which is challenging for methods like PSO [27]. (3) Adaptability to MEC dynamics: As highlighted in [28], a GA's iterative evolution aligns with the dynamic nature of terminal distribution and task arrival in MEC.

Compared to other meta-heuristics, the GA's balance between exploration and exploitation makes it suitable for the proposed multi-point elite mechanism, which further enhances its performance in MEC task offloading.

4 The Details of MP-GA

Unlike traditional genetic algorithms, the selection operator in MP-GA adopts a hierarchical strategy based on an elite mechanism, effectively selecting and retaining individuals with superior performance and potential. In the crossover operator, MP-GA utilizes a probabilistic method of multiple alleles, enhancing convergence speed while preserving the algorithm's global search capability. The mutation operator employs a method based on symmetric fragments, significantly boosting the algorithm's local search capability. A flow chart comparison is shown in Fig. 2:

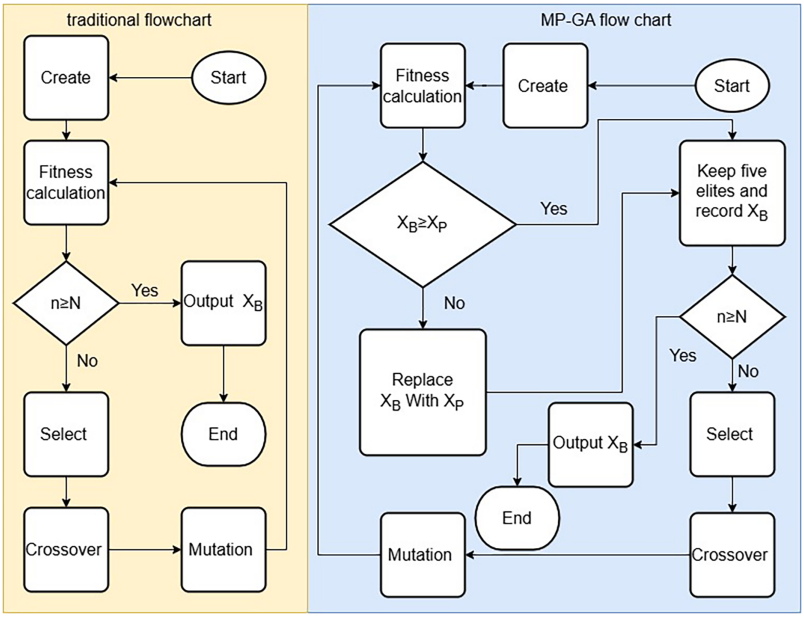

Figure 2: Comparison of traditional flow chart and MP-GA flow chart.

In Fig. 2, n represents the current iteration number, and N represents the set maximum iteration number. X_b represents the optimal solution under the current iteration number, and X_p represents the optimal solution of the previous generation.

4.1 Coding and Initial Resolution

In terms of coding, we use chromosomes to represent all the tasks to be processed at any given moment, with each gene in the chromosome representing a calculation task. Thus, the length of the chromosome will be $m \times k$. In the simulation, 0 indicates local processing, whereas 1 indicates offloading to the edge side. The encoding form adopted in MP-GA is illustrated in Fig. 3:

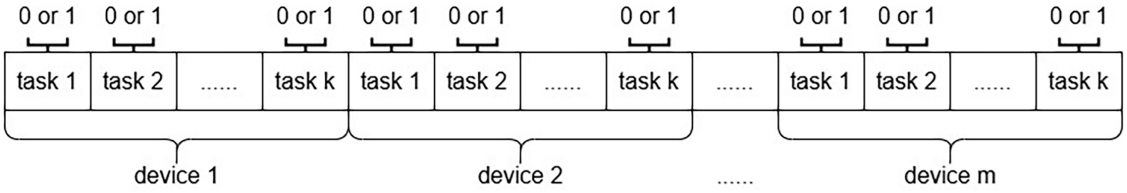

Figure 3: Encoding method of MP-GA.

Using chromosome coding, each chromosome corresponds to an independent solution. Let N denote the number of iterations and M denote the number of individuals in each generation. The matrix A_n ($n \in N$) is used to represent the offloading decision solution set obtained after n iterations. In this study, the first-generation population A_0 is initialized randomly, as shown in Eq. (12):

$$A_0 = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m\times k} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m\times k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M,1} & a_{M,2} & \cdots & a_{M,m\times k} \end{bmatrix} \tag{12}$$

Each row in the matrix represents an independent solution, and each column represents the calculation strategy of a separate task.

4.2 Selection Operation

The traditional genetic algorithm's roulette selection is abandoned and tournament selection is adopted [29]. The specific steps are presented in Algorithm 1:

Algorithm 1: Selection operation based on tournament strategy.

Input: Initial population $A_0 = \{a_1, a_2, \ldots, a_N\}$
Output: Selected population A_s

Initialize $A_s \leftarrow \varnothing$
Sort individuals in A_0 according to fitness in descending order
Select the top 5 individuals and add them into A_s (elitism preservation)
Set tournament size $T = 5$
while $|A_s| < N$ **do**
    Randomly select T individuals from A_0
    Evaluate and compare their fitness values
    Choose the individual with the best fitness as the winner
    Add the winner into A_s
end while
return A_s

From the above selection process, we can see that the number of elites is set to 5, which is due to the fact that through comparative experiments, we found that a convergence speed below 5 was not fast enough; however, a speed higher than 5 could easily lead to high gene similarity, resulting in falling into the local optimal solution. The specific experimental comparison results are shown in Table 2 (ConvergeGen represents the number of iterations).

Table 2: Fitness and convergence generation for different numbers of elites.

Number of Elites	1	2	3	4	5	6	7	8	9	10
Fitness	24.8	25.1	24.9	25.0	25.1	24.7	23.8	22.9	22.2	21.4
ConvergeGen	38	33	28	23	21	21	20	18	16	14

Considering the potential issue of an excessively high elite retention ratio in small population sizes, this paper adds an analysis of population size and elite ratio based on Table 2. It should be noted that the experiments in Table 2 were conducted with a default population size of 50, resulting in an elite ratio of 10% for retaining 5 elite individuals. When the population size is reduced to 20, retaining a fixed number of 5 elite individuals increases the elite ratio to 25%, potentially increasing selection pressure and leading to premature convergence. Therefore, this paper further tests the impact of fixed elite number and different elite ratios on algorithm performance under different population sizes.

As shown in Table 3, under the default population size = 50, retaining five elite individuals corresponds to an elite ratio of 10%, which provides a good balance between convergence speed and solution quality. This is consistent with the result in Table 2, where elite number = 5 achieves high fitness and fast convergence. However, when the population size decreases to 20, retaining five elites increases the elite ratio to 25%, which imposes stronger selection pressure and may reduce population diversity.

Table 3: Influence of population size and elite ratio on MP-GA performance.

Population Size	Elite Number	Elite Ratio	Fitness	ConvergeGen
20	1	5%	20.9	27
20	2	10%	21.4	19
20	5	25%	18.5	8
50	5	10%	25.1	21
100	5	5%	25.1	29

Table 4 compares the influence of different tournament sizes on MP-GA. A smaller T provides weaker selection pressure and helps maintain population diversity, but it may slow down convergence. A larger T accelerates convergence by increasing selection pressure, but it may over-select high-fitness individuals, reduce diversity, and increase the risk of premature convergence. The results show that T 5 achieves a better balance among fitness, convergence speed, and population diversity. Therefore, the tournament size is set to 5 in this study.

Table 4: Sensitivity analysis of tournament size.

Tournament Size	Selection Pressure	Fitness	ConvergeGen
2	Low	24.5	37
3	Moderate	24.7	31
5	Balanced	25.1	21
7	High	23.1	20
10	Very High	22.7	13

How tournament selection works is shown in Fig. 4:

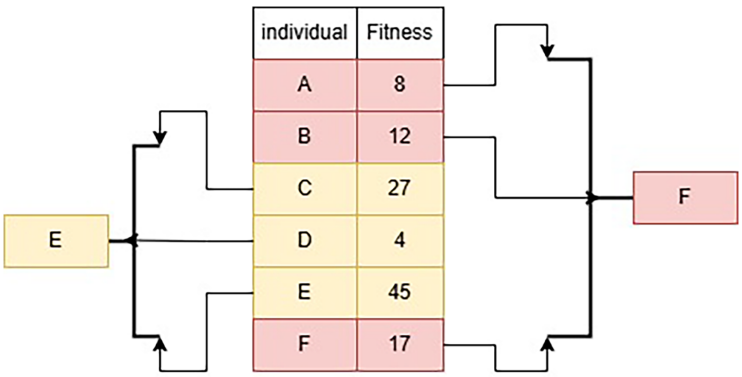

Figure 4: Tournament selection method.

Compared with traditional roulette selection, each round of the tournament only needs to compare a fixed number of individuals. There is no need to calculate the fitness of each individual in the entire population, and the comparisons can be performed in parallel. Therefore, the time efficiency is significantly better than that of roulette selection.

4.3 Crossover Operation

Following the selection process, each individual has a certain probability of undergoing crossover with a randomly chosen chromosome from the population. The crossover strategy employed here is a probabilistic crossover involving multiple alleles. This process generates a new offspring from two parent individuals through the crossover operator. A comparison with traditional crossover operations is illustrated in Fig. 5.

From the figure, we can see that traditional single-point crossover generates two offspring from two parents, while the multi-point crossover used in this article generates one offspring from two parents. This allows for a more flexible combination of different gene segments, thereby increasing genetic diversity. In contrast, single-point crossover can only cross at one point, which may lead to higher similarity among new individuals. At the same time, multi-point crossover allows for a larger exchange of information, not just the crossover of a single gene location. Multi-point crossover helps explore the potential solution space more efficiently.

To further examine whether generating two offspring is necessary, this study compares two offspring generation strategies under the same multi-point crossover mechanism. In the first strategy, two parents generate one offspring through multi-point crossover. In the second strategy, two parents generate two

offspring through multi-point crossover, and the offspring with the higher fitness value is retained. Since both strategies use multi-point crossover, the comparison focuses only on the influence of the number of generated offspring on optimization performance and computational cost.

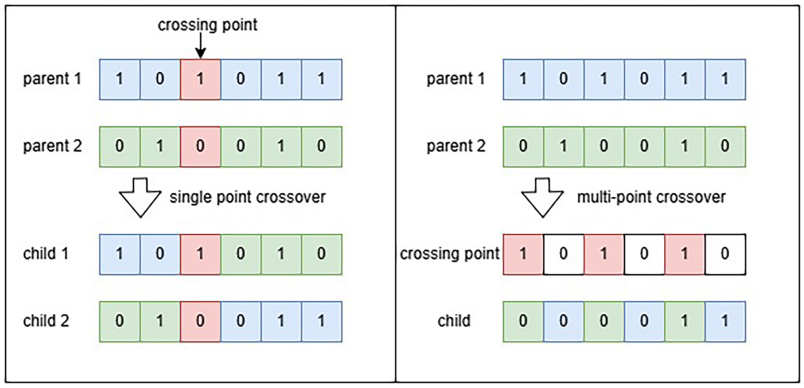

Figure 5: Single-point crossover and multi-point crossover comparison chart.

As shown in Table 5, generating two offspring and selecting the better one does not provide a significant improvement in final fitness or convergence generation compared with generating only one offspring. However, it requires an additional fitness evaluation in each crossover operation, which increases the execution time and computational burden. Therefore, generating one offspring is adopted in this study, as it achieves comparable optimization performance with lower computational cost.

Table 5: Comparison of different offspring generation strategies under multi-point crossover.

Offspring Strategy	Fitness	ConvergeGen	Time (s)	Energy (J)
Generate one offspring	25.1	21	177	15.9
Generate two offspring and select the better one	25.1	23	191	16.8

4.4 Mutation Operation

Similar to the crossover operation, the mutation operation also employs multi-point mutation. Each point on the chromosome is traversed, and when a mutation occurs at a particular point, the corresponding mirror point relative to the center also undergoes mutation simultaneously. The specific differences compared to single-point mutation are illustrated in Fig. 6.

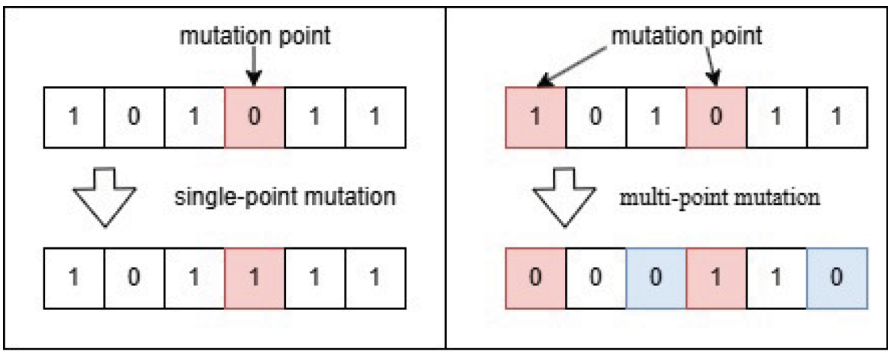

Figure 6: Single-point mutation and multi-point mutation comparison chart.

As shown in the figure above, multi-point mutation can induce mutations at more positions, thereby increasing the impact of mutation on gene combinations. This approach also enables a broader search within the gene space, helping to avoid local optima and enhancing the algorithm's local search capability. (When the chromosome length is odd, the gene point in the middle does not have a symmetric point, and mutations only affect itself. The rest is consistent with when the length is even).

5 Simulation and Discussion

After implementing the MP-GA in the MEC task offloading scenario, this study evaluated the algorithm's performance in the corresponding environment. Heuristic algorithms generally perform well when solving non-convex optimization problems. To comprehensively evaluate the algorithm's performance, several representative methods were introduced for comparison, including four heuristic intelligent algorithms, namely the traditional genetic algorithm, ant colony optimization algorithm, simulated annealing algorithm, and binary particle swarm optimization algorithm, as well as a deep reinforcement learning-based method (Deep Q-learning) and a Greedy strategy. It is worth noting that an elite mechanism was incorporated into the heuristic algorithms, which improved their performance to varying extents. For simplicity, these methods are referred to as GA, ANT, PSO, SA, DQN, and Greedy, respectively. Next, we will describe the experimental scenario parameters and optimization results.

5.1 Experimental Environment

The coverage range of 5G base stations differs depending on the equipment model. To evaluate the performance of the proposed algorithm, a simulation scenario is constructed. In this scenario, experimental parameters reported in previous studies are taken as references, and the communication coverage radius of the 5G base station is set to 350 m [30,31]. To reproduce the non-uniform spatial distribution of mobile terminals in practical environments, the distribution of the base station and mobile terminals is generated using the Opportunistic Network Environment simulator.

Each terminal has k task calculation requests. Task size, local computing energy consumption, and local computing time consumption are randomly generated within a reasonable range and normalized. Other key parameters in the simulation are shown in Table 6:

Table 6: Parameter settings for simulation.

Parameter	Value
Average noise power N	−100 dBm
5G signal transmission frequency f	3400 MHz
Terminal transmit power S_0	23 dBm
Energy weight coefficient λ_e	0.6
Time weight coefficient λ_t	0.4
Task offloading value weight λ_v	0.5
Task quantity weight λ_n	0.3
Maximum transmission rate weight λ_c	0.2
Channel bandwidth B_{tol}	10 MHz
Chromosome crossing probability P_{cos}	0.8
Allele crossing probability P_{swap}	0.5
Chromosome mutation probability P_{mut}	0.03

5.2 Results and Evaluation

In order to verify the effectiveness of the algorithm, key performance indicators of task offloading under both identical and varying task densities are discussed, such as convergence status, energy consumption, execution time efficiency, and bandwidth utilization. To reduce the impact of initialization randomness, each experiment was repeated twenty times under identical conditions. The convergence results of different algorithms with the number of tasks fixed at 48 are presented in Fig. 7.

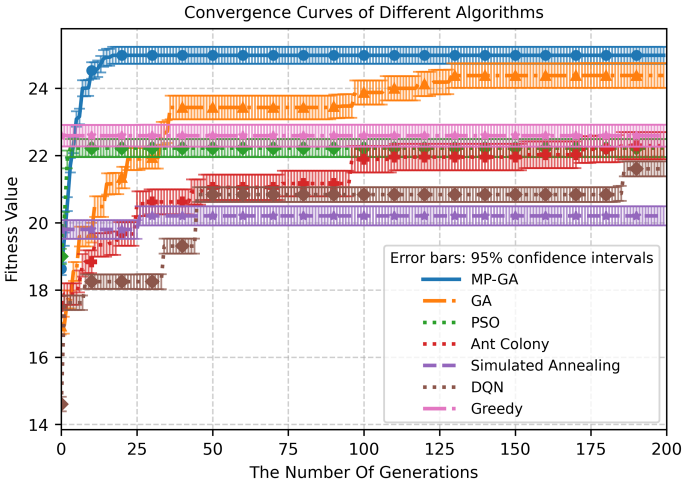

Figure 7: Convergence comparison of different algorithms.

Fig. 7 illustrates the convergence behaviors of different algorithms under the same number of tasks. The results show that, among all compared methods, MP-GA achieves the best solution quality and exhibits a faster convergence rate than the conventional GA. By contrast, PSO tends to fall into a local optimum at an early stage, which leads to relatively inferior performance. The Greedy method converges quickly due to its myopic decision mechanism, but its final solution quality is limited. DQN demonstrates a certain optimization capability, yet its convergence performance remains inferior to that of MP-GA in this scenario. Overall, MP-GA achieves a better balance between convergence speed and solution quality than the other benchmark algorithms. The fitness convergence of various algorithms under different numbers of tasks is shown in Fig. 8.

Fig. 8 shows that, under different numbers of tasks, the performance of the MP-GA remains superior to that of several other heuristic algorithms. As discussed above, this experiment mainly focuses on optimizing energy consumption and time consumption during task offloading. The local computation time consumption, and local computation energy consumption, under different tasks after the same number of iterations, are shown in Fig. 9.

From left to right, the histogram compares the performance of MP-GA, GA, PSO, ANT, SA, DQN, and Greedy. The figure indicates that, when the number of tasks is below 24, the time consumption of MP-GA is close to that of the other baseline algorithms. As the task scale further increases, however, bandwidth limitations gradually become a dominant bottleneck, since the available wireless resources are insufficient to support the offloading of all transmission tasks. Consequently, the optimization capability of the algorithm plays a more critical role, and the performance gaps among different methods become increasingly pronounced. Regarding energy consumption, MP-GA shows competitive performance across all considered task settings. The number of offloaded tasks and the level of bandwidth utilization can both reflect the user experience to some extent, as illustrated in Fig. 10.

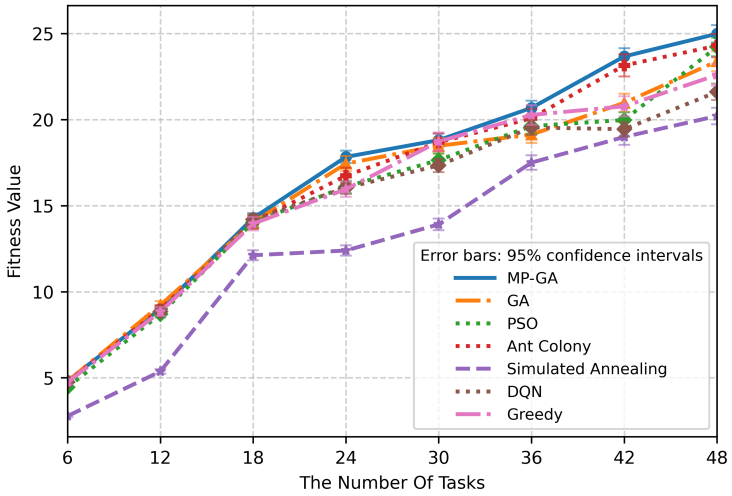

Figure 8: The fitness convergence under different numbers of tasks.

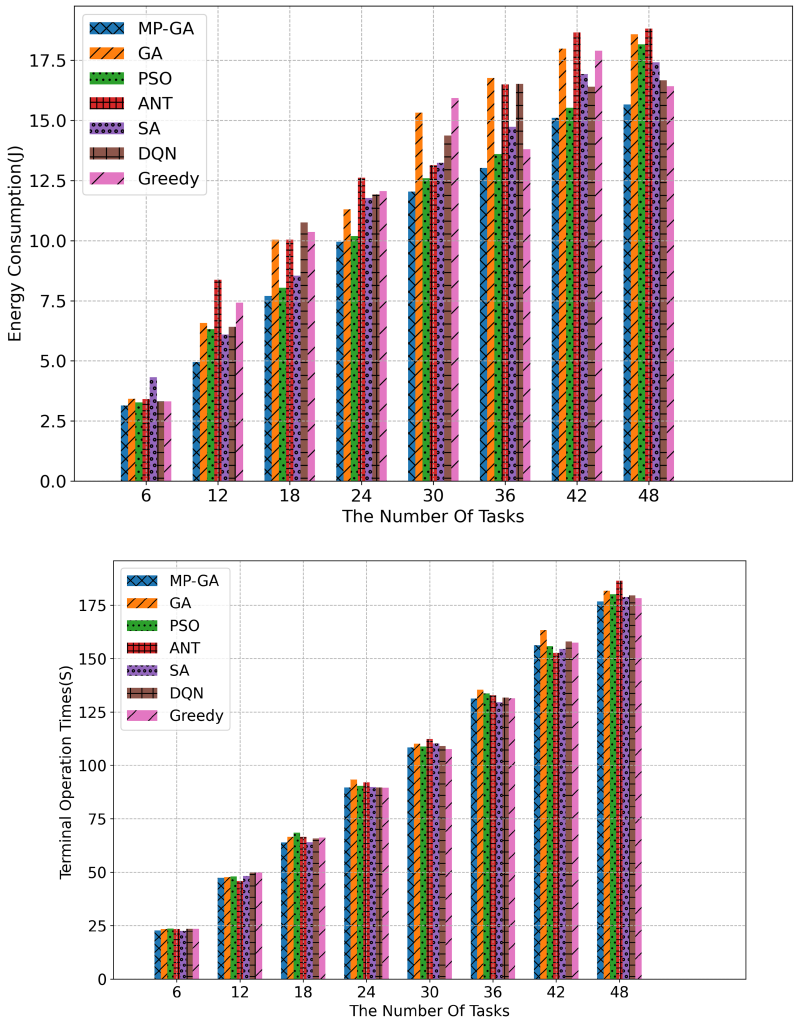

Figure 9: The energy and time consumption under different numbers of tasks.

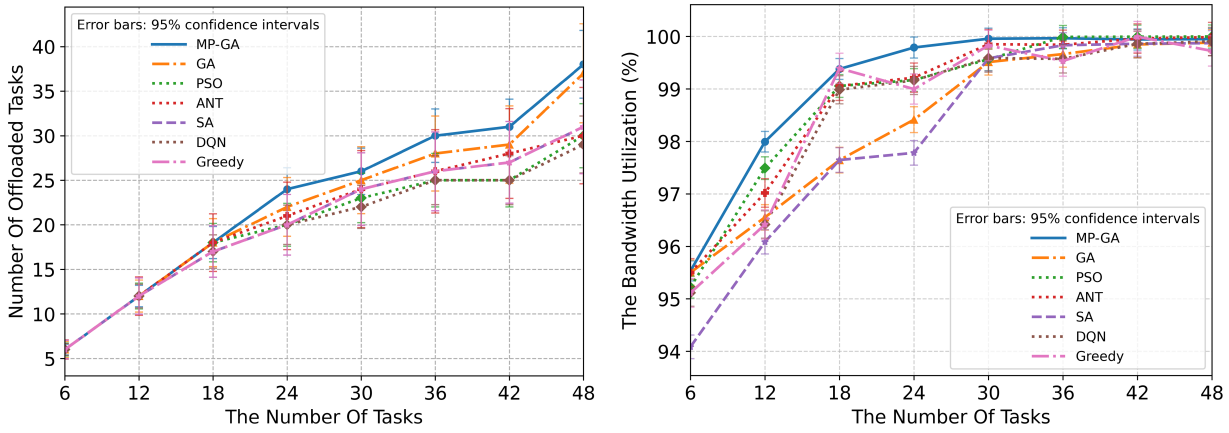

Figure 10: Offloading amount and bandwidth utilization under different task numbers.

As the number of computational tasks increases, the number of tasks successfully offloaded by different algorithms also rises gradually. It can be observed from the figure that when the number of tasks reaches 48, MP-GA offloads 38 tasks, which is the highest among all the compared algorithms. By comparison, GA offloads 37 tasks, SA offloads 32 tasks, Greedy offloads 31 tasks, PSO and ANT each offload 30 tasks, and DQN offloads 29 tasks. These results indicate that MP-GA exhibits a stronger task offloading capability under the considered task scale.

The utilization of available bandwidth is an important indicator for evaluating algorithm performance. In particular, we focus on whether the algorithm can make efficient and reasonable use of limited bandwidth resources. As illustrated in the figure, the pressure on available bandwidth becomes greater as the number of tasks increases. Under such conditions, MP-GA is still able to obtain better decision solutions and maintain a higher level of bandwidth utilization. The experimental results therefore indicate that MP-GA performs effectively in terms of bandwidth utilization.

Overall, the simulation results show that MP-GA can effectively reduce the consumption of local computing resources while enabling more computational tasks to be completed simultaneously. Compared with the conventional genetic algorithm, MP-GA improves convergence speed by 35% and enhances the joint optimization performance of time and energy consumption by about 12%. In addition, MP-GA increases the maximum number of offloaded tasks by 3%. These experimental results further confirm the optimization capability and practical effectiveness of MP-GA.

5.3 Ablation Study

To verify the effectiveness of each method in the proposed MP-GA, we conducted systematic ablation studies, involving the gradual removal or replacement of different components of the model, resulting in 16 ablation configurations. This allowed us to analyze the specific contributions of each method to the task. The quantitative analysis results are presented in Table 7.

Table 7: Ablation study of different genetic operator combinations.

SPC	MPC	SPM	MPM	RS	TS	Elites	Fitness	Energy (J)	Time (s)
✓	×	✓	×	✓	×	×	22.5	16.5	198
✓	×	✓	×	×	✓	×	22.8	17.1	195

(Continued)

Table 7 (continued)

SPC	MPC	SPM	MPM	RS	TS	Elites	Fitness	Energy (J)	Time (s)
✓	×	×	✓	✓	×	×	23.4	15.9	197
✓	×	×	✓	×	✓	×	23.9	16.7	192
×	✓	✓	×	✓	×	×	24.1	16.3	191
×	✓	✓	×	×	✓	×	23.2	15.6	187
×	✓	×	✓	✓	×	×	24.9	16.5	178
×	✓	×	✓	×	✓	×	23.1	16.1	179
✓	×	✓	×	✓	×	✓	23.2	16.5	187
✓	×	✓	×	×	✓	✓	23.4	16.8	187
✓	×	×	✓	✓	×	✓	24.8	15.9	177
✓	×	×	✓	×	✓	✓	25.1	16.3	188
×	✓	✓	×	✓	×	✓	24.5	16.0	183
×	✓	✓	×	×	✓	✓	24.9	15.9	178
×	✓	×	✓	✓	×	✓	25.2	16.4	181
×	✓	×	✓	×	✓	✓	25.3	15.9	177

From Table 7, it can be observed that the configurations incorporating multi-point crossover (MPC) and multi-point mutation (MPM) generally achieve higher fitness values and lower time consumption compared with those using single-point genetic operators. This indicates that multi-point genetic operations can enhance gene-segment recombination and improve the global search capability of the algorithm. In addition, tournament selection (TS) shows more stable performance than roulette selection (RS), particularly in terms of convergence efficiency. The introduction of elitism further improves the overall performance by preserving high-quality individuals across generations, which helps prevent performance degradation during evolution. Among all configurations, the combination of MPC, MPM, TS, and elitism achieves the best overall performance. This suggests that the proposed one-offspring multi-point crossover mechanism does not noticeably weaken the search capability; instead, it improves optimization performance through richer gene recombination and a more stable selection strategy. This result validates that the proposed MP-GA framework can better balance exploration and exploitation, thereby improving optimization performance in the considered task offloading problem.

The abbreviations used in Table 7 are defined as follows: SPC and MPC denote single-point crossover and multi-point crossover, respectively; SPM and MPM represent single-point mutation and multi-point mutation; RS and TS indicate roulette selection and tournament selection; and Elites denotes the use of elitism. It should be noted that SPC represents the conventional single-point crossover mechanism, in which two parents generate two offspring, whereas MPC represents the proposed multi-point crossover mechanism, in which two parents generate one offspring. Therefore, the comparison between SPC-based and MPC-based configurations not only reflects the difference between single-point and multi-point crossover, but also directly evaluates the influence of the one-offspring crossover mechanism on search performance.

The algorithm's exploration ability is indirectly evaluated through convergence behavior, final fitness, and ablation experiments. When the number of elites exceeds 5, although the convergence algebra decreases, the final fitness decreases, indicating that too strong elite retention increases selection pressure and may lead to premature convergence. In addition, ablation experiments show that the combination of multi-point

crossover, multi-point mutation, tournament selection, and elite retention achieves the best performance, indicating that this design can maintain sufficient exploration ability while accelerating convergence.

6 Conclusions

With the continued development of IoT and 5G technologies, the demand for data processing is expected to increase rapidly in the future. Consequently, how to allocate limited computing resources in an efficient and reasonable manner has become an important research issue. In this study, a comprehensive fitness function is designed for the task offloading problem in multi-terminal and multi-task scenarios under bandwidth constraints. The proposed function jointly considers several optimization factors, including energy consumption, time consumption, the number of offloaded tasks, and the maximum transmission rate of tasks. Based on this formulation, an enhanced genetic algorithm is introduced to address the task offloading problem in the considered scenario. Simulation results indicate that, compared with conventional heuristic algorithms such as genetic algorithms and particle swarm optimization, the MP-GA achieves faster and more stable convergence while demonstrating better overall performance in multi-objective optimization and bandwidth utilization.

In practical applications, however, iterative solution processes usually require considerable computing resources and may result in relatively long execution times. Therefore, future work will explore the integration of neural networks with the enhanced genetic algorithm, aiming to reduce computational overhead and further improve overall efficiency.

Acknowledgement: Not applicable.

Funding Statement: This research was supported by National Key Research and Development Program Industrial Software Key Special Project (2022YFB3305100).

Author Contributions: Conceptualization, Chengyu Hou and Hongping Shu; methodology, Chengyu Hou; formal analysis, Chengyu Hou and Hanyun Li; validation, Wenzao Li, Kui Liu and Zhuoning Zhao; investigation, Wenzao Li; data curation, Kui Liu; writing—original draft preparation, Chengyu Hou; writing—review and editing, Zhuoning Zhao and Hongping Shu; supervision, Hanyun Li; funding acquisition, Hongping Shu. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the corresponding author, [Hongping Shu], upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ullah A, Anwar SM, Li J, Nadeem L, Mahmood T, Rehman A, et al. Smart cities: the role of Internet of Things and machine learning in realizing a data-centric smart environment. Complex Intell Syst. 2024;10(1):1607–37. doi:10.1007/s40747-023-01175-4.
2. Abdi S, Ashjaei M, Mubeen S. Cost-aware workflow offloading in edge-cloud computing using a genetic algorithm. J Supercomput. 2024;80(17):24835–70. doi:10.1007/s11227-024-06341-0.
3. Zhu F. Cloud computing load balancing based on improved genetic algorithm. Int J Glob Energy Issues. 2024;46(3/4):191–207. doi:10.1504/IJGEI.2024.137051.
4. Wang K, Tu Z, Ji Z, He S. Faster service with less resource: a resource efficient blockchain framework for edge computing. Comput Commun. 2023;199(2):196–209. doi:10.1016/j.comcom.2022.12.014.

5. Akhunzada A, Zeadally S, ul Islam S. Toward performance and energy-efficient edge-of-things. IT Prof. 2023;25(1):17–23. doi:10.1109/MITP.2022.3230837.
6. Zhu S, Zhao M, Zhang Q. Multi-objective optimal offloading decision for multi-user structured tasks in intelligent transportation edge computing scenario. J Supercomput. 2022;78(16):17797–825. doi:10.1007/s11227-022-04549-6.
7. Asghar S, Karim K, Sohrabi M. Server placement in mobile cloud computing: a comprehensive survey for edge computing, fog computing and cloudlet. Comput Sci Rev. 2024;51(1):100616. doi:10.1016/j.cosrev.2023.100616.
8. Alam M, Shahid M, Mustajab S. Security challenges for workflow allocation model in cloud computing environment: a comprehensive survey, framework, taxonomy, open issues, and future directions. J Supercomput. 2024;80(8):11491–555. doi:10.1007/s11227-023-05873-1.
9. Dong H, Wu W, Song H, Liu Z, Zhang Z. Data and model driven task offloading strategy in the dynamic mobile edge computing system. J Syst Sci Complex. 2024;37(1):351–68. doi:10.1007/s11424-024-4035-x.
10. Sulieman N, Ricciardi C, Li W, Zomaya A, Villari M. Edge-oriented computing: a survey on research and use cases. Energies. 2022;15(2):452. doi:10.3390/en15020452.
11. Xie M, Su H, Sun H, Zhang Q, Sun X. Online task offloading algorithm based on multi-objective optimization caching strategy. Comput Netw. 2024;254(3):348–59. doi:10.1016/j.comnet.2024.110400.
12. Kanellopoulos D, Sharma VK, Panagiotakopoulos T, Kameas A. Networking architectures and protocols for IoT applications in smart cities: recent developments and perspectives. Electronics. 2023;12(11):2490. doi:10.3390/electronics12112490.
13. Wei J, Liang X. Research on task-offloading delay in the IoV based on a queuing network. IEEE Access. 2024;12:31324–33. doi:10.1109/ACCESS.2024.3368793.
14. Alsadie D. Efficient task offloading strategy for energy-constrained edge computing environments: a hybrid optimization approach. IEEE Access. 2024;12(2):85089–102. doi:10.1109/ACCESS.2024.3415756.
15. Shen G, Chi K, Alfarraj O, Tolba A. Task offloading and resource allocation for wireless powered multi-AP mobile edge computing. IEICE Trans Fundam Electron Commun Comput Sci. 2025;E108A(2):129–39. doi:10.1587/transfun.2024EAP1070.
16. Chen Y, Zhao J, Wu Y, Huang J, Shen X. QoE-aware decentralized task offloading and resource allocation for end-edge-cloud systems: a game-theoretical approach. IEEE Trans Mob Comput. 2024;23(1):769–84. doi:10.1109/TMC.2022.3223119.
17. Peng S, Li B, Liu L, Fei L, Niyato Z, Dusit D. Trajectory design and resource allocation for multi-UAV-assisted sensing, communication, and edge computing integration. IEEE Trans Commun. 2025;73(4):2847–61. doi:10.1109/TCOMM.2024.3478115.
18. Li B, Liu W, Xie W, Li X. Energy-efficient task offloading and trajectory planning in UAV-enabled mobile edge computing networks. Comput Netw. 2023;234(1):109940. doi:10.1016/j.comnet.2023.109940.
19. Wang D, Bakar KB, Isyaku B. Two-stage IoT computational task offloading decision-making in MEC with request holding and dynamic eviction. Comput Mater Contin. 2024;80(2):2065–80. doi:10.32604/cmc.2024.051944.
20. Chen S, Wang X, Sun Y. TODO: task offloading decision optimizer for the efficient provision of offloading schemes. Pervasive Mob Comput. 2024;99(3):101892. doi:10.1016/j.pmcj.2024.101892.
21. Shuai J, Xie B, Cui H, Wang J, Wen W. Q-learning-based task offloading strategy for satellite edge computing. Int J Commun Syst. 2023;37(5):e5691. doi:10.1002/dac.5691.
22. Liu J, Ren J, Zhang Y, Peng X, Zhang Y, Yang Y. Efficient dependent task offloading for multiple applications in MEC-cloud system. IEEE Trans Mob Comput. 2021;22(4):2147–62. doi:10.1109/TMC.2021.3119200.
23. Tang J, Wang S, Yang S, Xiang Y, Zhou Z. Federated learning-assisted task offloading based on feature matching and caching in collaborative device-edge-cloud networks. IEEE Trans Mob Comput. 2024;23(12):12061–79. doi:10.1109/TMC.2024.3403851.
24. Zhang Y, Liu Y, Zhou J. Slow-movement particle swarm optimization algorithms for scheduling security-critical tasks in resource-limited mobile edge computing. Future Gener Comput Syst. 2020;112(3):148–61. doi:10.1016/j.future.2020.05.025.
25. Awadallah MA, Makhadmeh SN, Al-Betar MA, Dalbah LM, Al-Redhaei A, Kouka S, et al. Multi-objective ant colony optimization: review. Arch Comput Methods Eng. 2025;32(2):995–1037. doi:10.1007/s11831-024-10178-4.

26. Li Y. Optimization of task offloading problem based on simulated annealing algorithm in MEC. In: Proceedings of the 9th International Conference on Intelligent Computing and Wireless Optical Communications (ICWOC); 2021. p. 47–52. doi:10.1109/ICWOC52624.2021.9530216.
27. Chakraborty S, Mazumdar K. Sustainable task offloading decision using genetic algorithm in sensor mobile edge computing. J King Saud Univ Comput Inf Sci. 2022;34(4):1552–68. doi:10.1016/j.jksuci.2022.02.014.
28. Li Z, Zhu Q. Genetic algorithm-based optimization of offloading and resource allocation in mobile-edge computing. Information. 2020;11(2):83. doi:10.3390/info11020083.
29. Zhou K, Oh SK, Pedrycz W, Qiu J, Seo K. A self-organizing deep network architecture designed based on LSTM network via elitism-driven roulette-wheel selection for time-series forecasting. Knowl Based Syst. 2024;289(1):111567. doi:10.1016/j.knosys.2024.111481.
30. Gui R, Zhang X, Gui X, Han J. A placement method of the 5G edge nodes based on the hotspot distribution of mobile users. Appl Sci. 2024;14(13):5943. doi:10.3390/app14135943.
31. He J. 5G communication resource allocation strategy for mobile edge computing based on deep deterministic policy gradient. J Eng. 2023;3(3):e12250. doi:10.1049/tje2.12250.