Multi-Objective Resource Allocation in Edge Computing Using Improved Genetic Algorithm with Knowledge-Based Crossover and Segmentation Mutation

Xiaozhi Zhang
College of Artificial Intelligence and Big Data, Zibo Vocational Institute, Zibo 255000, China
E-mail: 10238@zbvc.edu.cn

Keywords: edge computing, mobile edge computing, genetic algorithms, resource allocation, decision-making model

Received: January 6, 2025

With the widespread adoption of edge computing technology, efficient resource allocation becomes an urgent problem to be solved. Aiming at the optimization of resource allocation in the edge computing environment, a multi-objective optimization function with three dimensions of time, energy consumption, and computing resource occupation is constructed. An improved genetic algorithm resource allocation strategy model based on a knowledge-guided cross-segmentation mutation mechanism is proposed. Knowledge crossover enhances the global search capability by prioritizing the retention of gene segments with higher historical fitness contributions. The experimental results showed that the loss function value of the new model could be as low as 0.012 during the training process, indicating that the model convergence on multi-objective optimization was effectively improved. Meanwhile, the improved genetic algorithm model generated an average of 4.8 optimal solutions, which was 1.8 more than the traditional genetic algorithm which generated an average of 3.0 optimal solutions. Compared with multi-objective optimization algorithms such as NSGA-II, SPEA, and MOPSO, the research model reduced the average energy consumption of the device to 112.68 Joules and the average energy consumption of the system to 208.12 Joules in the EdgeDroid dataset. Its utilization of computational resources reached 77.45%, with the processing time of the task shortened to 5.44 seconds. In real application scenarios, the model achieved a 92.1% and 89.2% task completion rate in shopping malls and hospital environments, and the resource utilization rate was improved to 83.49% and 81.67%. In summary, the proposed method effectively improves resource allocation efficiency, reduces system energy consumption, and optimizes task completion performance, which provides strong support for dynamic resource management in edge computing environments.

Povzetek: Predlagan je izboljšan genetski algoritem, ki ima križanje podprto z znanjem in segmentacijsko mutacijo za učinkovito dodeljevanje virov v robnem računalništvu z večciljno optimizacijo časa, energije in zmogljivosti.

1 Introduction

Driven by the Internet of Things, smart devices, and 5G networks, the sharp increase in demand for computing resources has become a significant issue in the field of modern information technology. To meet these challenges, Edge Computing (EC), as a new computing architecture, has gradually been widely used [1]. However, with the rapid growth in the number of access devices and the increasing diversity of application scenarios, traditional EC gradually exposes its limitations in resource scheduling and load balancing. Especially in the context of heterogeneous resources, dynamic network changes, and diverse task requirements, existing resource allocation mechanisms make it difficult to balance system efficiency and service quality. This has led to increasingly prominent issues such as decreased system energy efficiency and increased task delays. Therefore, how to realize efficient and flexible resource allocation in a large-scale dynamic environment has become an important topic to be solved in the field of EC. In recent years, Mobile Edge Computing (MEC) has emerged with the popularization of mobile Internet and the rapid development of wireless communication technology [2]. MEC further promotes the development of EC, emphasizing the close integration of computing and communication infrastructure to support large-scale mobile devices and diverse application requirements. In this context, researchers are faced with two core problems: on the one hand, the EC resource allocation is a complex multi-constraint optimization problem. Although traditional Genetic Algorithms (GAs) have good global search capabilities, they are prone to getting stuck in local optima in dynamic and heterogeneous edge environments, making it difficult to quickly adapt to changes in resources. Therefore, improving the search mechanism of GA and enhancing its global exploration ability becomes an important direction for resource allocation

optimization. On the other hand, multi-objective optimization in EC involves multiple metrics such as computation time, energy consumption, and resource utilization, and these metrics are often contradictory to each other. How to balance device energy consumption control and resource load balancing while ensuring system response speed, and establishing an effective balancing mechanism, is also an urgent problem to be solved. To address the aforementioned issues, many industry researchers have conducted explorations. Mohajer A et al. believed that energy efficiency and fairness assurance were key issues in MEC-based cellular systems. For this purpose, a dynamic optimization model was proposed. Compared to traditional methods, this model had higher energy efficiency in resource allocation and larger throughput [3]. Jiang H et al. built an online joint offloading and resource allocation method under long-term MEC energy constraints to improve the resource utilization efficiency of MEC. The framework could solve the queue optimization problem of resource allocation [4]. Chai F et al. proposed a joint multi-task resource allocation scheme by combining directed acyclic graphs to address the scarcity of satellite communication computing resources. The experimental results showed that this method reduced the cost by 8.87% compared to the baseline method [5]. Liu Q et al. proposed a multi-objective resource allocation method for the Internet of Things to achieve load balancing and lower energy consumption. The experimental results showed that this method was more effective in achieving the goal and had a shorter number of iterations compared to traditional methods [6].

GA is a stochastic optimization strategy that simulates natural selection and genetic mechanisms. Due to its powerful global search capability and good adaptability, GA is widely used in resource allocation optimization problems [7]. Chakraborty S et al. believed that the resource limitation threat of existing mobile devices still existed. Therefore, a resource offloading model in conjunction with GA was proposed. The experimental results showed that the model reduced energy consumption and computational delay compared to traditional methods within the allowed transmission delay range [8]. Apinaya Prethi K N et al. found that due to various tasks, edge nodes had an impact on data accessibility. Therefore, a new optimization scheduling method by combining GA and the multi-objective Krill-Held algorithm was proposed. The experimental results showed that this method was more effective in allocating resources compared to other traditional algorithms [9]. Fan W et al. found that existing research rarely considered the resource utilization of mobile vehicles. Therefore, a joint task offloading and resource allocation model using GA was developed. The model could minimize the total priority-weighted task processing delay in seven different scenarios [10]. Gabi D et al. believed that meta-heuristic technology still faced bottlenecks in achieving sustainable profit advantages and cost reduction in MEC. To this end, a novel resource allocation strategy that combined GA and simulated annealing optimization algorithms was proposed. This strategy outperformed other methods in terms of statistical results within the 95% confidence interval [11]. Chen M et al. found that although several studies have been conducted to investigate the scheduling problem for edge cloud computing. There were still some issues that needed to be addressed in their applications, such as ignoring resource heterogeneity and focusing on only one kind of request. Therefore, this research proposed a heterogeneity-aware task scheduling algorithm to improve the task completion rate and resource utilization of edge clouds with deadline constraints. The results showed that this algorithm outperformed 13 other classical and state-of-the-art scheduling algorithms in terms of task completion rate [12]. Liu Q et al. found that Internet of Things applications still faced severe challenges when applying multi-objective resource allocation to satisfy service demands. Technically, the Pareto archive evolution strategy was used to optimize the time cost, load balancing of MEC servers, and energy consumption of Internet of Things applications. From the results, the tested results were better than before optimization [13]. Chen Q H et al. developed an enhanced framework, namely a multi-objective micro-services allocation algorithm. This algorithm formulated efficient resource management of cloud micro-services resources as a constrained optimization problem guided by resource utilization and network communication overheads to simplify workload monitoring and analysis in different cloud systems. The results showed that the method significantly improved resource utilization, reduced network transmission overhead, and improved reliability [14]. To comprehensively sort out the characteristics and limitations of existing studies in terms of optimization objectives, dataset selection, and performance performance, the study compiles typical representative methods, which are shown in Table 1.

Table 1: Literature review table

Method	Dataset used	Optimization objective	Main performance indicators	Limitations
Dynamic optimization model (Mohajer A et al.)	MEC cellular system simulation dataset	Energy efficiency optimization, fairness assurance	Improved energy efficiency,	Single resource allocation strategy, limited dynamic

			increased throughput	adaptability
Joint offloading and resource allocation framework (Jiang H et al.)	Long-term MEC energy-constrained simulation data	Queue optimization, improved resource utilization	Queue stability, higher resource utilization	Focused on long-term constraints, insufficient real-time support
Multi-task resource allocation scheme (Chai F et al.)	Satellite communication simulation data	Cost reduction, task offloading optimization	Cost reduced by over 8.87%	Limited to satellite IoT scenarios
Multi-objective resource allocation (Liu Q et al.)	IoT application simulation data	Load balancing, energy minimization	Faster convergence, better load balancing	Insufficient multi-objective trade-off handling, weak heterogeneity support
Resource offloading model (Chakraborty S et al.)	Mobile device simulation data	Transmission delay optimization, energy reduction	Lower energy consumption and delay	Lack of task diversity consideration
Optimized scheduling method (Apinaya Prethi K N et al.)	Edge node simulation environment	Improved data accessibility, resource allocation optimization	Higher resource utilization	Focused mainly on data access, limited comprehensiveness
Joint task offloading model (Fan W et al.)	Vehicular network multi-scenario data	Minimization of task processing delay	Reduced task delay	Performance fluctuation under high mobility
Resource allocation strategy (Gabi D et al.)	MEC-cloud continuum simulation data	Cost optimization, profit enhancement	Lower cost, higher profitability	Higher computational complexity
Heterogeneity-aware scheduling algorithm (Chen M et al.)	Edge cloud environment data	Improved task completion rate, better resource utilization	Outperformed 13 classic scheduling algorithms in task completion rate	Focused on single-type request scenarios
Multi-objective microservice allocation algorithm (Chen Q H et al.)	Multi-cloud system environment	Improved resource utilization, reduced communication overhead	Higher resource utilization, lower network overhead	Focused on cloud, insufficient support for edge scenarios

To sum up, the existing research has made significant progress in resource allocation of EC and MEC. However, most existing methods still suffer from a single resource allocation strategy and a balance between computational efficiency and fairness. To solve the above problems, the research innovatively considers MEC's time resources, computing resource occupation, equipment energy consumption, and system energy consumption, and introduces GA to improve the genetic structure, especially crossover and mutation. Finally, a new EC resource allocation strategy model is proposed. To ensure the operability of the proposed method in real MEC scenarios, the study comprehensively considers key factors such as node heterogeneity, resource dynamic changes, task real-time constraints, and energy sensitivity during the design process. Specifically, the knowledge crossover mechanism combines historical task load statistics and node adaptability information to dynamically adjust the gene fragment priority during execution. The segmentation mutation operation adaptively adjusts the mutation probability according to the load state of the edge nodes to avoid excessive perturbation and ensure the stability and effectiveness of resource scheduling. Through this scenario characteristic-oriented mechanism design, the proposed IGA method is able to achieve efficient and reliable

resource allocation optimization in complex and changing real MEC environments. The study further improves the overall performance of the EC system under the premise of ensuring the quality of service, aiming to provide a more robust method to support the diversified development of MEC in the future.

2 Methods and materials

2.1 EC system model and its multi-objective problem

Unlike traditional EC, MEC focuses more on supporting large-scale mobile devices and terminal applications, such as smartphones, autonomous driving, smart cities, etc [15]. Among them, MEC's resource allocation optimization is particularly crucial. Firstly, MEC's resources not only include computing and storage capabilities but also multidimensional resources such as network bandwidth and delay. The allocation of these resources needs to consider multiple factors, such as changes in user demand, device mobility, and uncertainty in the network environment. The network framework of MEC is shown in Figure 1 [16].

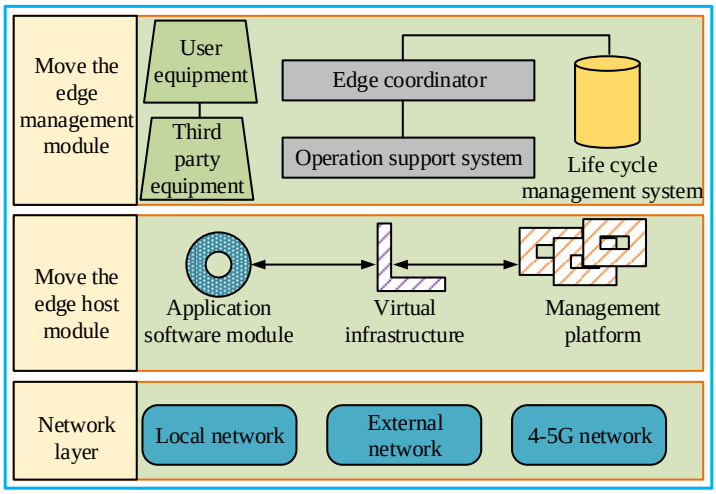


Figure 1: MEC network framework.

In Figure 1, the mobile edge system module involves user devices, third-party devices, coordinators, operating systems, and lifecycle management systems. This module is mainly used for overall control of MEC and visual optimization. In the mobile edge host module, there are built-in application software modules, virtual infrastructure, and multiple management platforms. In the network module, there are mainly three key sub-networks, namely local network, external network, and 4-5G network. The multi-MEC scenario architecture is shown in Figure 2.

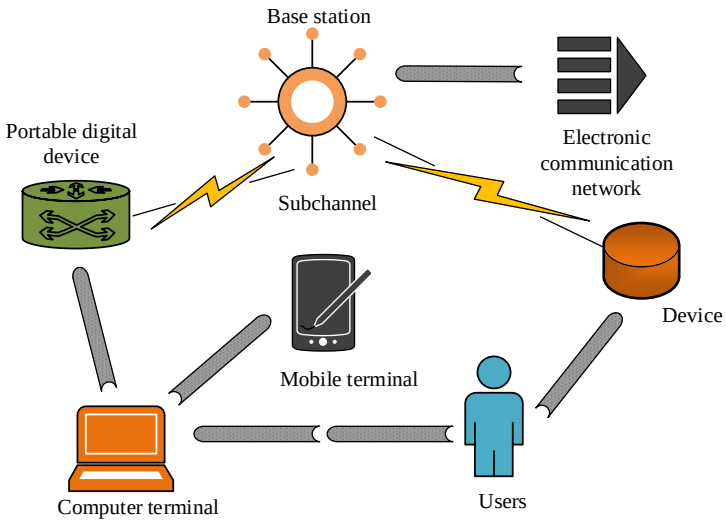


Figure 2: Multi-MEC scenario architecture diagram.

In Figure 2, the multi-MEC scenario consists of multiple similar sub-MECs, each of which is composed of user devices such as mobile phones, computers, portable digital devices, as well as network base stations and electronic communication networks. The devices and networks are connected in a sub-channel manner. If each user starts to perform their own structured tasks, their needs can be effectively disseminated through electronic communication networks. However, the computing resources of each individual electronic communication network are limited, and the increasing demand for user tasks only increases the network load and causes it to collapse. If a crash occurs, the user's task requirements cannot be processed, and timely resource computation and offloading can be carried out to achieve lightweight. The unloading process is shown in Figure 3 [17].

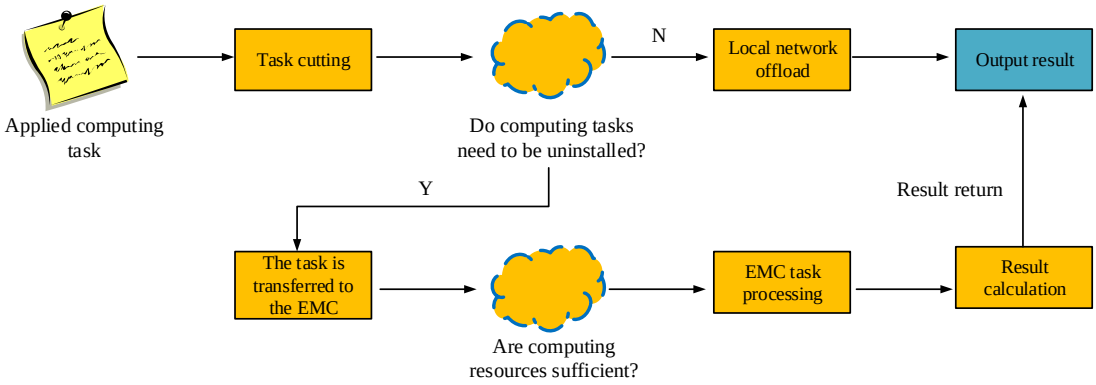


Figure 3: MEC computing uninstall process.

From Figure 3, first, the user device generates computational tasks and evaluates their computational requirements. Based on the computational complexity, data transmission requirements, and latency constraints of the task, the user device decides whether to perform offloading locally, prioritizing tasks with high computation, low latency sensitivity, or sufficient bandwidth conditions. When the task needs to be offloaded, the user device sends the task to a neighboring EC node through the local network and cuts the task data on the node side. The cutting step includes task segmentation, data format standardization, and necessary preprocessing operations for distributed computing. After receiving the task, the edge node will further determine whether to accept the task and enter the formal processing stage according to the current computing resources and load. The processing steps mainly include parallel execution of segmented tasks, resource scheduling, and intermediate result summarization. If the node resources are insufficient, it can carry out offloading or queuing of some sub-tasks again. After completing the computation, the edge nodes transmit the final processing results back to the user device through the network. The returned results include task output data and status feedback. This is different from the initial offloading phase of the local network where only the original task data is transmitted. Assuming that data transmission time and computation time cannot be ignored during task execution, the time resource model is calculated, as shown in equation (1).

$$T_{total} = \frac{D_{task}}{C_{edge}} + \frac{d_{user}}{v_{user}} + T_{propagation} + T_{handover} \quad (1)$$

In equation (1), T_{total} represents the total amount of time resources in seconds. D_{task} signifies the computational requirements of the task. C_{edge} indicates the computing capacity of EC nodes. d_{user} represents the transmission distance from the user device to the edge node. v_{user} represents the transmission rate. $T_{propagation}$ represents the delay caused by signal propagation. $T_{propagation} = \frac{d_{user}}{c_{signal}}$, where c_{signal} represents the speed of signal propagation. $T_{handover}$ denotes the additional delay incurred during device movement or network switching. The basis for selecting this parameter is the propagation theory modeling in wireless communication. This theory refers to the fixed delay caused by physical propagation during the transmission of signals from user devices to edge nodes. It should be noted that secondary influences such as data queuing delays and scheduling delays are ignored in the model, which focuses on the two key overheads of transmission and computation. To further portray the resource occupancy of tasks on the

edge node, it is assumed that the computational capacity of the EC node is C_{edge} and the computational demand of each task is C_{edge}. Then, the computational resource proportion occupied by a single task and multiple tasks is calculated as shown in equation (2).

$$\begin{cases} \alpha_{task} = \dfrac{D_{task}}{C_{edge}} \\ \alpha_{task}^{*} = \sum_{i=1}^{n} \dfrac{D_{task}^{i}}{C_{edge}} \\ \alpha_{total} = \sum_{i=1}^{n} \dfrac{D_{task}^{i} \cdot \beta_i}{C_{edge}} \end{cases} \quad (2)$$

In equation (2), α_{task} and α_{task}^{*} represent the proportion of computing resources occupied by a single task and multiple tasks, respectively. β_i denotes the prioritization factor of task i. The determination of the prioritization factor takes into account the urgency of the task, the computational resource demand, the energy sensitivity, and the business importance level of the application scenario to which the task belongs. In general, tasks with shorter deadlines, higher computational demands, or from high-priority application scenarios have their corresponding β_i values set larger to obtain more computational resource support in the resource allocation process. n denotes the total number of tasks. The introduction of priority factor can adjust the resource allocation strategy according to the importance of tasks. In terms of energy modeling, to reflect the energy consumption of the device during task processing, the energy consumption of the device is divided into two parts: computational energy consumption and transmission energy consumption. The total energy consumption of the device is calculated as shown in equation (3).

$$\begin{cases} E_{compute} = P_{compute} \cdot T_{compute} = P_{compute} \cdot \dfrac{D_{task}}{C_{edge}} \\ E_{transmit} = P_{transmit} \cdot T_{transmit} = P_{transmit} \cdot \dfrac{d_{user}}{v_{user}} \\ E_{device} = E_{compute} + E_{transmit} \end{cases} \quad (3)$$

In equation (3), $E_{compute}$ signifies the computational energy consumption of the user device. $P_{compute}$ represents computational power consumption. $T_{compute}$ represents the calculation time. $E_{transmit}$ represents transmission energy consumption. $P_{transmit}$ represents transmission power consumption. E_{device} represents the total energy consumption of the user device. Standby energy consumption and re-transmission overhead are not introduced in the model, which mainly focuses on the explicit energy consumption during task execution. Assuming that there are m total of EC nodes in the system. The energy consumption of individual nodes and the whole MEC system can be further derived as follows when considering the energy consumption of individual nodes and the whole MEC system, as shown in equation (4).

$$\begin{cases} E_{edge} = P_{edge} \cdot T_{compute} = P_{edge} \cdot \dfrac{D_{task}}{C_{edge}} \\ E_{system} = \sum_{i=1}^{n} E_{device}^{i} + \sum_{j=1}^{m} E_{edge}^{j} \end{cases} \quad (4)$$

In equation (4), E_{edge} signifies the energy consumption of a single node. E_{system} represents the total energy consumption of the system. E_{device}^{i} represents the energy consumption of the i-th user device. E_{edge}^{j} indicates the energy consumption of the j-th EC node. At this point, the resource allocation problem of MEC system is transformed into a multi-objective optimization problem, as displayed in equation (5).

$$E_{total} = \min(T_{total}, \alpha_{total}, E_{device}, E_{system}) \quad (5)$$

In equation (5), the multi-objective function includes time, energy consumption, and utilization of computing resources, while considering resource constraints such as computing power, transmission bandwidth, and device power consumption.

2.2 Edge computing resource allocation model based on improved GA

After constructing a complete function for the multi-objective optimization problem of resource allocation in multi-scenario MEC, this paper explores how to solve the optimization problem through efficient algorithms. Due to the dynamic changes in resources, high mobility of devices, and time-varying nature of tasks in e-commerce scenarios, traditional deterministic optimization methods (such as precise mathematical programming) often struggle to obtain globally optimal solutions [18]. Therefore, in recent years, heuristic optimization methods based on evolutionary computation, especially GA, have been widely applied to solve resource allocation problems in such complex dynamic environments. The process of GA is shown in Figure 4.

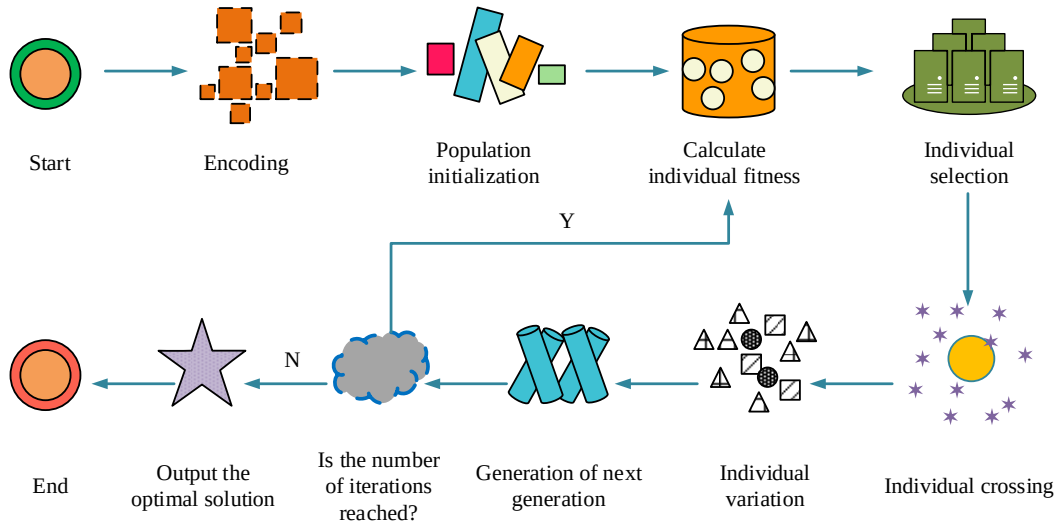


Figure 4: GA flow diagram.

In Figure 4, the initial population is randomly generated during the initialization phase. Each individual represents a resource allocation scheme in the form of an integer vector encoding, used to represent the allocation decisions of different tasks to edge nodes. A population is composed of multiple independent individuals, each of whom is a complete resource allocation plan. As the basic operating unit, individuals ensure the independence and diversity of understanding. Then, in the selection phase, the excellent individuals are selected as parents for crossover and mutation operations based on the fitness function to generate offspring individuals to explore better solutions in the resource allocation space. Next, the individuals are screened by fitness evaluation. The optimal individuals are retained into the next generation to gradually optimize the key performances of the system such as energy consumption, computation time, and resource utilization. Each generation refers to the state update of the population during an evolutionary iteration. The whole evolutionary process is repeated until the preset maximum number of iterations is reached or the population fitness converges, ultimately realizing efficient resource scheduling optimization under multi-objective. However, when allocating resources for multi-scenario MEC, there are still limitations of multiple electronic communication networks. Therefore, to reduce the complexity of the algorithm and optimize its adaptability, the GA is optimized. In the encoding stage, the allocation decision is set to integer encoding, where an individual represents an allocation decision and the decision can be encoded as a vector with dimensions. Among them, the dimensionality represents the number of tasks required by the user. The fitness function of Improved Genetic Algorithm (IGA) is shown in equation (6).

$$F(x) = w_1 \cdot T(x) + w_2 \cdot E(x) + w_3 \cdot C(x) \quad (6)$$

In equation (6), $F(x)$ signifies the fitness value of the resource allocation scheme. $T(x)$ signifies the total task processing time of the system under the allocation plan. $E(x)$ represents the total energy consumption of the system under the allocation scheme. $C(x)$ represents the system computing resource utilization rate under the allocation scheme. w_1, w_2, and w_3 represent the weight coefficients of time, energy consumption, and computational resource utilization, respectively. After completion, the individual with higher f_i-value is selected by means of roulette wheel betting. This means that the parent generation is randomly selected according to the proportional probability of the individual fitness value. The higher the fitness value, the higher the probability that the individual is selected. The roulette wheel betting method can keep the population diversity while retaining the excellent individuals and avoid falling into local optimization. The calculation formula is shown in equation (7).

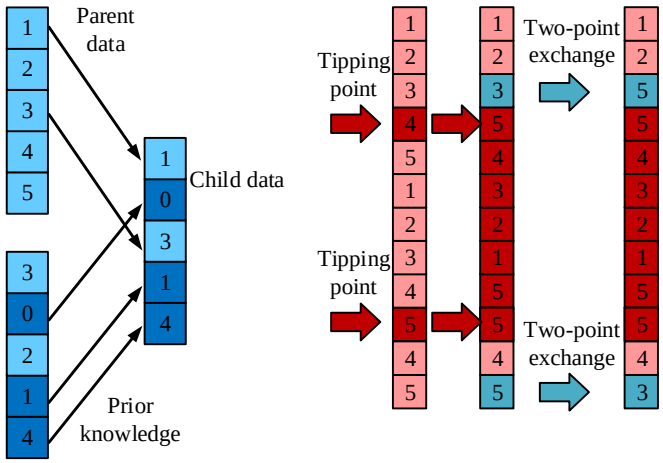
$$P_i = \frac{f_i}{\sum_{i=1}^{N} f_i} \quad (7)$$

In equation (7), f_i represents the fitness value of individual i. N represents the quantity of individuals. P_i represents the probability of selecting the individual i. The f_i is shown in equation (8).

$$f_i = \frac{1}{\xi \cdot F(x)} \quad (8)$$

In equation (8), ξ represents the coefficient for adjusting fitness. In the crossover stage, traditional GA uses single point crossover to combine and select individuals. Specifically, a crossover point is randomly selected on the encoding vector of two parental individuals, and gene fragments are exchanged after the crossover point, resulting in two new offspring individuals. This process can ensure the partial inheritance of the excellent genes of the parent generation

while introducing new combinations to increase the population diversity and promote the further exploration of the solution space. However, in MEC, single point crossover may lead to a lack of sufficient diversity in the population. Therefore, the study introduces knowledge-based crossover operators. In addition, the study aims to avoid excessive concentration of excellent individuals during the crossover process, which may cause the algorithm to fall into local optima prematurely. During the crossover process, the study conducts segmentation and mutation operations in advance to further divide the population, ensuring that the chromosome and mutation probabilities are different. The improved crossover and mutation operations are shown in Figure 5.



(a) Based on the intersection of knowledge

(b) Segmentation mutation

Figure 5: Schematic diagram of knowledge-based crossover and segmentation mutation operations.

Figure 5(a) is a schematic diagram of knowledge-based crossover, and Figure 5(b) is a schematic diagram of segmentation mutation. From Figure 5, knowledge-based crossover operations first evaluate the fitness performance of parental individuals and the contribution of gene fragments to fitness enhancement through prior knowledge or historical data. Prior knowledge includes the average effect of each gene fragment on fitness enhancement during past evolutionary processes. During the crossover process, gene segments with higher historical contributions are preferentially selected for combination. The genes with darker colors after crossover are shown in the figure. The darker genes indicate potential segments [19]. The critical point in the figure indicates the position of gene fragment switching, and the numbered element represents the specific position of the gene fragment in the chromosome code. Then, the segmentation variational operation randomly generates two variational points. Two point switching and inverse sorting are performed on the depicted intervals to expand the search space, increase perturbations, and enhance global search capabilities. Unlike traditional mutation methods with small single point perturbations, split mutation involves randomly selecting two mutation points on the chromosome, exchanging interval genes, and performing reverse sequencing operations. By implementing large-scale structural reorganization in this way, the search space has been significantly expanded [20]. The knowledge crossover is shown in equation (9).

$$P_{crossover} = \frac{e^{\delta \cdot \Delta f_i}}{\sum_{j=1}^{N} e^{\delta \cdot \Delta f_i}} \quad (9)$$

In equation (9), $P_{crossover}$ represents the probability of i being selected for crossover. δ represents the weighting factor. Δf_i signifies the change in fitness of i. To ensure that potential gene fragments are prioritized during the crossover process, a weighted selection mechanism can be used based on the historical performance of gene fragments and their fitness contribution in the population, as shown in equation (10).

$$S_{gene} = \frac{1}{\sum_{k=1}^{M} (\lambda_{k,i} \cdot w_k)} \quad (10)$$

In equation (10), S_{gene} represents the priority of selecting the k-th gene fragment. $\lambda_{k,i}$ represents the contribution of gene fragment k segment in individual i. w_k denotes the historical performance weighting coefficient of gene fragment k, which is used to reflect the contribution of this gene fragment to the enhancement of individual fitness in the previous optimization process. Specifically, w_k is set to be a uniform value for all gene fragments at the initial stage. As the evolutionary process advances, the changes in fitness brought about by new individuals generated by each gene fragment participating in the crossover are recorded and cumulatively updated. M signifies the number of gene fragments. At this point, the multi-level mutation probability control after population segmentation is shown in equation (11) [21].

$$P_{mutation} = \frac{1}{1+e^{-\varphi\cdot(\theta_{subgroup}-\mu)}} \quad (11)$$

In equation (11), $P_{mutation}$ represents the probability of individual i mutating within subpopulation s. φ represents the parameter that controls the rate of change in mutation probability. $\theta_{subgroup}$ denotes the measure of fitness difference between individuals s within a sub-population, used as a measure of population diversity. The sub-populations are divided based on the Euclidean distance of the resource allocation decision vector. Individuals with distances less than a set threshold are grouped into the same sub-population. μ denotes the mean value of fitness in the sub-population. In summary, the study is based on a knowledge-based crossover operation that preferentially combines gene segments that have historically contributed more to fitness enhancement to improve evolutionary directionality. To prevent premature convergence, a split-variation mechanism is used and the probability of variation is adaptively adjusted based on fitness differences within sub-populations to enhance perturbation when population diversity declines. At the same time, an elite retention strategy has been established to ensure that each generation can directly inherit the best individuals, greatly avoiding premature convergence. To sum up, the research uses IGA to solve the multi-objective problem of MEC system in equation (5). A new EC resource allocation strategy is designed. The flow is shown in Figure 6.

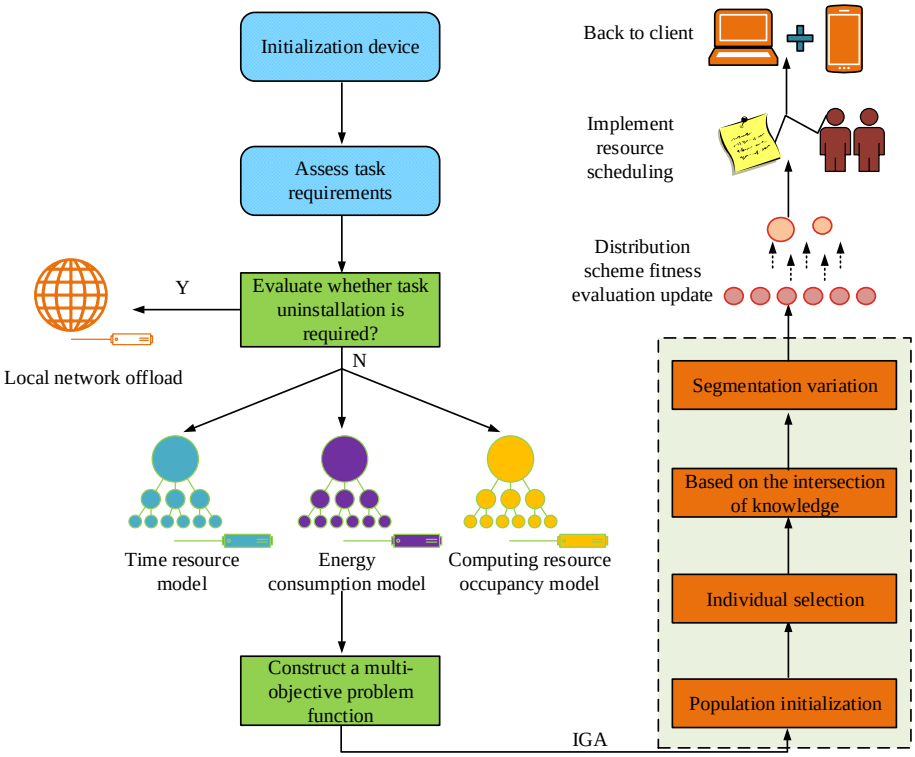


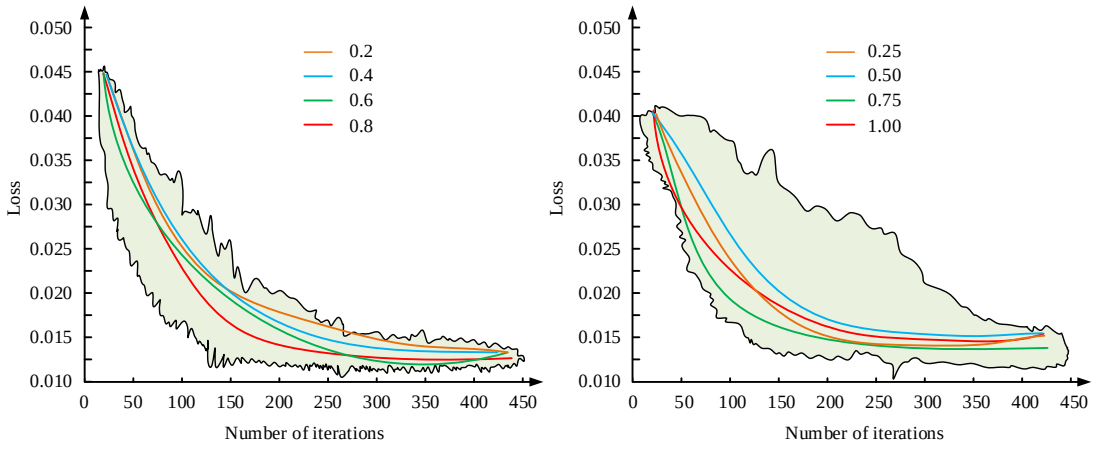
Figure 6: New EC resource allocation strategy model flow.

From Figure 6, first, the user equipment produces a computing task and evaluates the task requirements, and decides whether to unload the task to the EC node according to the computing complexity, data transmission requirements, and delay constraints. Secondly, based on task requirements and edge node load, appropriate edge nodes are selected for resource allocation. Next, considering multidimensional factors such as time resources, computational resource utilization, and energy consumption, a comprehensive evaluation of resources is conducted. Then, the resource allocation problem is transformed into a multi-objective optimization problem, balancing time, energy consumption, and computational resource utilization, and optimized through IGA. Next, IGA generates an initial population, performs crossover and mutation operations, selects individuals with higher fitness, and schedules resources through optimization schemes. Finally, the EC node executes the task and sends the result back to the user equipment to ensure the task is completed successfully.

3 Results

3.1 Performance test of new edge computing resource allocation strategy model

The research sets the CPU to Intel Xeon Gold 6230R and the GPU to NVIDIA Tesla A100 40GB. The operating system is Ubuntu 20.04, with 5 edge nodes configured with 2 GPUs and 16 CPU cores per node, simulating a multi-node distributed computing environment. In addition, the EdgeDroid dataset and the Canadian Institute for Cybersecurity Intrusion Detection System (CICIDS 2017) dataset are used as test data sources. In the experimental setup, for the EdgeDroid and CICIDS 2017 datasets, features such as task size, latency, bandwidth, etc. are first extracted and then uniformly normalized. In addition, the population is initialized with a uniform random distribution by setting the resource requirements in the range of [0.5, 5] MB, latency of [50, 500] ms, bandwidth of [0.5, 10] Mbps, and the random seed is fixed at 42 to ensure reproducibility. In addition, the reason for choosing these two types of datasets is that EdgeDroid covers diverse mobile application scenarios and CICIDS 2017 reflects complex network traffic characteristics. The combination of the two can comprehensively verify the adaptability and generalization of the algorithms under dynamic loads and heterogeneous environments. The adjustment factor for setting the fitness coefficient is to balance selection pressure and population diversity. A smaller adjustment factor is beneficial for maintaining the early explorability of the population. A larger adjustment factor is beneficial for convergence speed. The setting of cross mutation weight factors is to control the strength of the impact of knowledge guided cross mutation and segmentation mutation. It achieves a dynamic balance between exploration and utilization by adjusting the dominance of both at different stages. The study first conducts training and value selection tests on the fitness coefficient adjustment factor ξ and cross mutation weighting factor δ that have the greatest impact on model strategy generation. The result is shown in Figure 7. The figure shows the trend of the Loss function on the training set under different configurations of the fitness adjustment factor and the cross-variance weighting factor. By comparing the convergence speed and final Loss values corresponding to different hyperparameter combinations, the parameter settings that make the model perform optimally are selected to ensure a good balance between convergence and generalization performance during the optimization process.



(a) Fitness coefficient adjustment factor test (b) Cross variation weighting factor test

Figure 7: Hyperparameter selection test result.

Figure 7 (a) displays the selection test results of the fitness coefficient adjustment factor, and Figure 7 (b) displays the selection test results of the cross-mutation weighting factor. As shown in Figure 7 (a), when the fitness coefficient adjustment factor was 0.2, the lowest value of the loss function reached 0.015. When the fitness coefficient adjustment factor gradually increased to 0.8, the lowest value of the loss function was around 0.012, indicating that the model had the strongest generalization ability at this time. As shown in Figure 7 (b), when the cross-mutation weighting factor increased from 0.25 to 0.75, the loss function value obviously decreased to around 0.015. However, when the cross-mutation weighting factor was 1.00, the loss function value actually increased, indicating that factor values that are too large or too small can interfere with the optimization, thereby affecting its performance. Therefore, with a fitness coefficient adjustment factor of 0.8 and a cross mutation weighting factor of 0.75, the model could achieve good performance on both the training and testing sets. The research attempts to conduct ablation tests on the IGA to verify the effectiveness of the improvements made in the study. The results are shown in Figure 8.

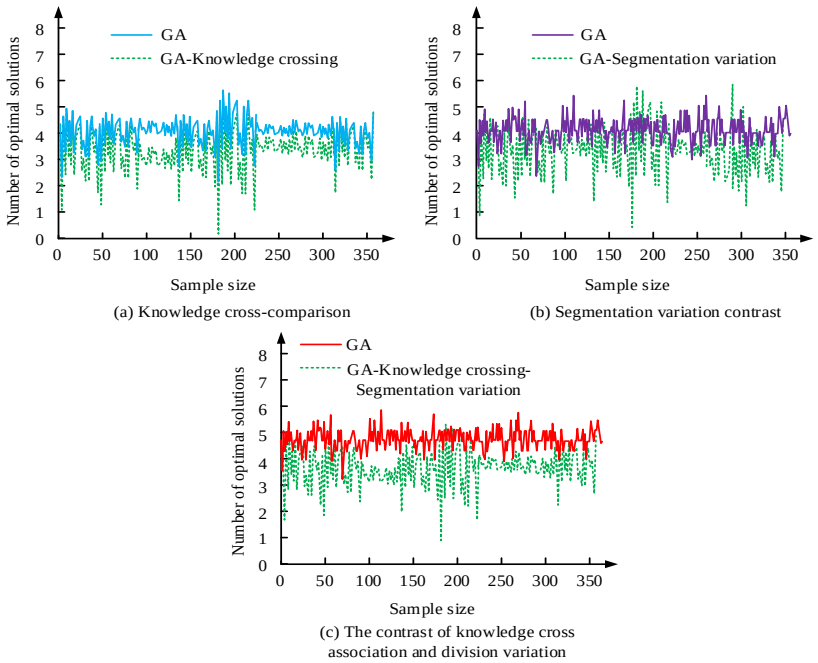


Figure 8: The results of the IGA ablation test.

Figure 8 (a) displays the test results of generating optimal solutions for GA and GA-knowledge crossover. Figure 8 (b) displays the test results of generating optimal solutions for GA and GA-segmentation mutation. Figure 8 (c) displays the test results of generating optimal solutions for GA and GA-knowledge crossover-segmentation mutation. From Figure 8, under different test conditions, the IGA that incorporates knowledge crossover and segmentation variants generates the highest mean value of optimal solutions (4.8). This is significantly better than the version that uses the traditional GA alone or only incorporates a single improvement strategy. This shows that the dual-mechanism synergy effectively improves the algorithm's global search capability and convergence performance. The results under different initial population distributions and task scenario changes show that IGA can still maintain a low fluctuation amplitude, and the number of optimal solutions does not change by more than ± 0.3. This result verifies that the proposed method has good stability and consistency under different initial conditions and dynamic loading environments, and has strong scene adaptability. The study compares similar multi-objective optimization algorithms, such as Non-dominated Sorting Genetic Algorithm II (NSGA-II), Strength Pareto Evolutionary Algorithm (SPEA), and Multi-objective Particle Swarm Optimization (MOPSO), as displayed in Table 2. The average energy consumption of equipment, utilization rate of computing resources, average energy consumption of the system, and average processing time of tasks are used as comparison indicators.

Table 2: Multiple index test results of different models

Data set	Model	Computing resource usage/%	Average energy consumption of equipment/J	Average system energy consumption/J	Average task processing time/s	*p*
Edge Droid	NSGA-II	82.56	115.76	212.34	5.63	0.018
	SPEA	80.21	121.34	220.56	5.78	0.012
	MOPSO	79.84	119.54	217.34	5.93	0.015
	Our model	77.45	112.68	208.12	5.44	0.003

CICIDS 2017	NSGA-II	85.29	145.67	274.65	7.21	0.021
	SPEA	83.12	150.23	280.45	7.35	0.017
	MOPSO	82.76	148.56	276.89	7.45	0.019
	Our model	80.54	141.23	268.42	7.11	0.002

From Table 2, the proposed model has the lowest computational resource occupancy of 77.45% in both datasets, which is 5.11%, 2.76%, and 2.39% lower compared to NSGA-II, SPEA, and MOPSO, respectively. In addition, the proposed model also shows better energy efficiency in terms of average device energy consumption (112.68 J) and average system energy consumption (208.12 J). The system energy consumption is reduced by 4.22% compared to NSGA-II and MOPSO. In terms of average task processing time, the proposed model is optimal with a result of 5.44 seconds, which is 0.19 seconds less than NSGA-II and 0.49 seconds less than MOPSO. Further statistical analysis shows that all comparison items reach a significant difference level (p<0.05). This indicates that the proposed method has statistical advantages in resource utilization, energy optimization, and improving processing efficiency. Among them, the weight coefficients of energy consumption, task completion time and resource utilization are determined by preliminary hyperparameter search. The parameter combination that can balance each index and optimize the overall adaptability is selected. When the system detects significant changes in task load or node status, the research model can dynamically adjust the weight ratio of energy consumption target and delay target through a weight adaptive adjustment mechanism to maintain the stability and optimization effect of system performance.

3.2 Simulation test of new edge computing resource allocation strategy model

The study selects two real scenarios for testing, namely shopping malls and hospitals. The bandwidth between edge nodes and servers is set to 1Gbps, and between servers and cloud computing centers to 10Gbps. About 2,000 smart devices are installed inside the mall. The hospital's internal equipment is relatively complex, involving equipment monitoring, real-time diagnosis, and other tasks, simulating 40 EC nodes. In the energy consumption modeling process, the study comprehensively considers the hardware heterogeneity among edge nodes. The node computing power, energy consumption characteristics, and transmission bandwidth are modeled as input parameters, respectively. Node heterogeneity directly affects task offloading and resource allocation decisions. That is, low-computing-capacity or high-energy-consumption nodes are inclined to be assigned with lighter or latency-tolerant tasks during the optimization process. This further optimizes the energy consumption distribution and resource utilization while ensuring the overall system performance. The completion rate of resource allocation tasks for four types of models in different scenarios is shown in Figure 9.

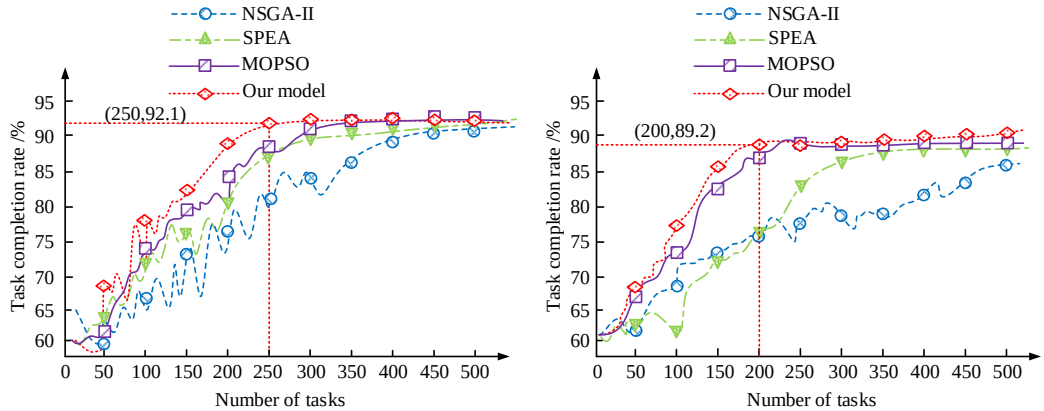


Figure 9: Success test of model resource allocation in shopping mall and hospital scenarios.

Figure 9 (a) shows the task completion rate results of four types of models in the shopping mall scenario. Figure 9 (b) displays the task completion rate results of four types of models in the hospital scenario. According

to Figure 9 (a), the task completion rate of the proposed model is significantly better than the other three models, reaching 92.1%, while the task completion rates of NSGA-II, SPEA, and MOPSO are 88.42%, 87.15%, and 90.34%, respectively. This result indicates that in the shopping mall scenario, the proposed model can better meet the real-time and processing requirements of the task, maximizing the task completion rate. From Figure 9 (b), the task completion rate of the proposed model is 89.2%, which is far higher than the other three models. It is proved again that the new model can better adapt to complex equipment monitoring and real-time diagnosis tasks in the hospital scene, and optimize the allocation of EC resources. During the resource allocation process, the model sets feasibility filtering criteria based on the memory capacity, processing capability, and bandwidth constraints of edge nodes. It allocates tasks only when the node resource constraints are met, ensuring the feasibility of the allocation scheme and the stability of the system. The MOPSO with better performance is selected and compared it with the proposed method. Taking the network load and delay as indicators, the results are shown in Figure 10. In this case, the network load is measured by recording the data transfer rate (in Mbps) between nodes in real-time, and the latency is calculated by recording the end-to-end delay (in ms) from the initiation of the task to the reception of the returned result.

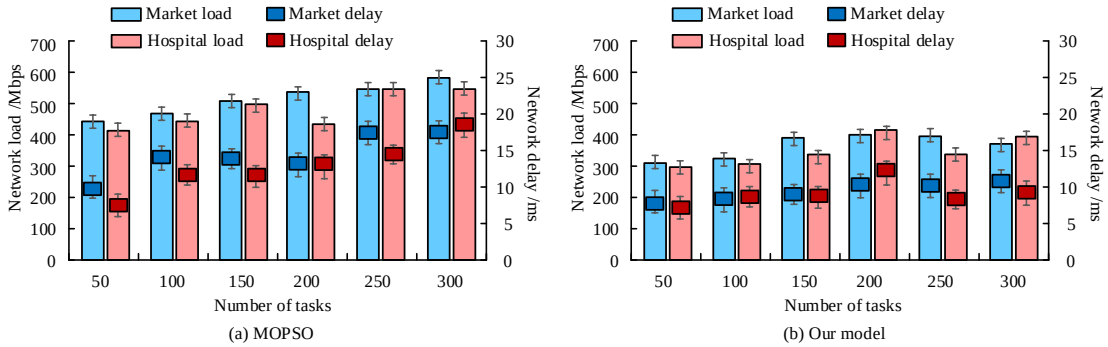


Figure 10: Load and delay comparison results of MOPSO and the designed method.

Figure 10 (a) shows the load and delay test results of shopping malls and hospitals in the MOPSO model. Figure 10 (b) shows the load and delay test results of shopping malls and hospitals in the proposed model. In Figure 10 (a), in the shopping mall scenario, with the increase of network tasks, the network load of MOPSO reached up to 600 Mbps. This indicated that the resource scheduling efficiency of the model was low under high user density, resulting in heavy network load. Meanwhile, the delay also significantly increased with the increase of network load, reaching a maximum of 18.4 ms. In the hospital scenario, although the network load was always lower than in the shopping mall scenario, the delay fluctuated between 10 ms and 15 ms, indicating higher user density and task quantity. From Figure 10 (b), the proposed model had significantly lower network loads in shopping malls and hospitals compared to MOPSO when handling multitasking, with average loads of 318 Mbps and average delays of 9.1 ms, respectively. This demonstrated that the proposed method could effectively control network load and delay in high load environments, with better resource allocation capabilities and optimization effects. Finally, the study conducted tests using resource utilization rate, allocation execution time, and total system energy consumption as indicators, as displayed in Table 3.

Table 3: Index test results of different models in shopping mall and hospital scenarios

Environment	Model	Resource utilization/%	Allocation execution time/s	Total system energy consumption/J
Mall	NSGA-II	75.34	15.87	230.57
	SPEA	72.18	16.5	240.63
	MOPSO	80.21	14.32	225.11
	Our model	83.49	12.94	215.32
Hospital	NSGA-II	68.92	18.25	280.47
	SPEA	70.14	17.88	290.25
	MOPSO	78.11	16.17	270.83
	Our model	81.67	14.68	260.02

According to Table 3, the proposed model in shopping mall and hospital scenarios was superior to other comparative models. In the shopping mall scenario, the resource utilization rate of the proposed model was 83.49%, the allocation execution time was 12.94 seconds, and the total energy consumption was 215.32 J. In the hospital scenario, the resource utilization rate was 81.67%, the allocation execution time was 14.68 seconds, and the total energy consumption was 260.02 J. Compared with NSGA-II, SPEA, and MOPSO models, the proposed model showed the highest resource utilization, the shortest allocation execution time, and the lowest system energy consumption in these two scenarios. This verifies its optimization effect in EC resource allocation, which can more effectively balance performance and energy efficiency, and improve the overall system performance. In addition, the results in Table 2 were based on the simulation experiments of EdgeDroid with CICIDS 2017 dataset. The statistics were the average values of system energy consumption under different dataset tests. On the other hand, the results in Table 3 were from tests conducted in actual application scenarios (shopping centers and hospitals), reflecting the average system energy consumption measured in specific deployment environments, hence there were differences in values. The study continued to validate the task offloading success rate, computational complexity, and task re-scheduling rate of the different approaches under different task priority fluctuation ranges. The results are shown in Table 4.

Table 4: Model performance results under fluctuating task priorities

Fluctuation level	Method	Task offloading success rate (%)	Computational complexity (ms)	Task rescheduling rate (%)
Small fluctuation (±1 level)	NSGA-II	90.7	240	11.2
	SPEA	89.3	235	12.5
	MOPSO	91.5	220	10.1
	Our model	95.8	185	6.3
Large fluctuation (±3 levels)	NSGA-II	86.4	255	18.9
	SPEA	84.9	250	20.3
	MOPSO	87.8	235	16.2
	Our model	92.4	190	10.5

From Table 4, under the condition of small priority fluctuation, the proposed model outperforms NSGA-II, SPEA, and MOPSO in task offloading success rate (95.8%), computational complexity (185 ms), and task re-scheduling rate (6.3%), and exhibits higher scheduling efficiency and stability. Under the condition of large priority fluctuation, although the overall performance of each method decreases, the proposed model still maintains the highest task offloading success rate (92.4%) and the lowest task re-scheduling rate (10.5%). The computational complexity grows less, which further verifies the adaptability and robustness of the proposed method under dynamic task priority change scenarios. Compared with traditional GA, the proposed IGA effectively improves global search capability and dynamic scheduling flexibility by introducing knowledge guided crossover and splitting variants, while increasing a small amount of computational overhead. This has good scalability for multi node heterogeneous device environments in large MEC systems. The optimization model can dynamically adjust resource allocation based on node algorithm power, prioritize low complexity tasks to low-power devices, and achieve overall energy efficiency optimization.

4 Discussion

Aiming at the problem of resource allocation efficiency and energy consumption optimization in EC environment, this paper proposed an IGA model incorporating knowledge-guided crossover and segmentation variants. The proposed model was comprehensively validated in the EdgeDroid and CICIDS 2017 datasets, as well as in real-world application scenarios in shopping malls and hospitals. The experimental results showed that the proposed model outperformed mainstream multi-objective optimization methods such as NSGA-II, SPEA, and MOPSO in terms of computational resource occupancy (minimum 77.45%), average energy consumption of the device (minimum 112.68J), and task processing time (minimum 5.44s) (p<0.05). The improvement in this performance was mainly due to the collaborative mechanism of knowledge-guided intersection. In the early stage, global search was accelerated, and in the later stage, variant segmentation was accelerated, enhancing population diversity. This effectively avoided the local optimization

problems that traditional GA is prone to. By dynamically balancing energy consumption, latency, and resource utilization targets through the resource aware fitness function, the overall flexibility and robustness of scheduling were enhanced.

Compared with NSGA-II's lack of efficiency in Pareto frontier exploration, SPEA's slow convergence of energy consumption under high load conditions, and MOPSO's large fluctuation of task completion rate, the proposed IGA was able to maintain a higher level of task success and energy efficiency under different task priority changes and server dynamics. This result verified its adaptive advantages in dynamic MEC environments. However, there are still some limitations to this study. The main reason is that IGA introduces additional cross and variance computation overhead in complex scenarios, which may lead to an overall increase in scheduling latency for large-scale node systems (such as thousands of edge nodes), although the time cost increase for a single iteration is small (about 8.5%). Moreover, although the current model has taken node heterogeneity into account, it has not yet been optimized for the extreme resource-limited devices. In addition, although the current model has considered node heterogeneity, there is still room for further optimization for the migration and deployment of extreme resource-constrained devices. Future work can combine the lightweight evolution strategy with the layered parallelism mechanism to further improve the scalability and real-time response capability of the algorithm in the ultra-large-scale distributed MEC environment.

5 Conclusion

To solve the optimization problem of MEC resource allocation in EC environment, the study constructed a multi-objective optimization function and improved the crossover and variance structure of the GA. A new EC resource allocation strategy model was proposed. The results showed that the training Loss reached a minimum of about 0.012 when the fitness adjustment factor was 0.8 and the cross-variance weighting factor was 0.75. The number of optimal solutions generated by the IGA that incorporated the knowledge crossover and segmentation variance was improved to nearly 4.8 in this setting. This was significantly optimized compared to both the traditional GA and the single-improved strategy. Compared with NSGA-II, SPEA, and MOPSO methods, the proposed model achieved the optimum in key metrics such as computational resource occupancy (minimum 77.45%), average energy consumption of the device (112.68 J), average energy consumption of the system (208.12 J), and average processing time (5.44 s). The task completion rates for the shopping mall and hospital scenarios were 92.1% and 89.2%, respectively. The average network latency was 9.1ms, the resource utilization rate was 83.49%, and the allocated execution time was 12.94s. It had scheduling advantages in dynamic multi-scenario environments. Nevertheless, considering the challenge of highly dynamic changes of edge nodes, dynamic scheduling methods based on real-time reinforcement learning can be explored in the future. Additionally, IGA can be fused with other evolutionary algorithms to further improve the model's real-time responsiveness and self-adaptive optimization in ultra-large-scale MEC systems.

References

[1] Tan T, Zhao M, Zeng Z. (2022). Joint offloading and resource allocation based on UAV-assisted mobile edge computing. *ACM Transactions on Sensor Networks (TOSN)*, vol. 18, no. 3, pp. 1-21. https://doi.org/10.1145/3476512

[2] Deng C, Fang X, Wang X. (2022). UAV-enabled mobile-edge computing for AI applications: Joint model decision, resource allocation, and trajectory optimization. *IEEE Internet of Things Journal*, vol. 10, no. 7, pp. 5662-5675. https://doi.org/10.1109/JIOT.2022.3151619

[3] Mohajer A, Daliri M S, Mirzaei A, Ziaeddini A, Nabipour M, Bavaghar M. (2022). Heterogeneous computational resource allocation for NOMA: Toward green mobile edge-computing systems. *IEEE Transactions on Services Computing*, vol. 16, no. 2, pp. 1225-1238. https://doi.org/10.1109/TSC.2022.3186099

[4] Jiang H, Dai X, Xiao Z, Lyengar A. (2022). Joint task offloading and resource allocation for energy-constrained mobile edge computing. *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 4000-4015. https://doi.org/10.1109/TMC.2022.3150432

[5] Chai F, Zhang Q, Yao H, Xin X, Gao R, Guizani M. (2023). Joint multi-task offloading and resource allocation for mobile edge computing systems in satellite IoT. *IEEE Transactions on Vehicular Technology*, vol. 72, no. 6, pp. 7783-7795. https://doi.org/10.1109/TVT.2023.3238771

[6] Liu Q, Mo R, Xu X, Ma X. (2024). Multi-objective resource allocation in mobile edge computing using PAES for Internet of Things. *Wireless Networks*, vol. 30, no. 5, pp. 3533-3545. https://doi.org/10.1007/s11276-020-02409-w

[7] Li Z, Li G, Bilal M, Liu D, Huang T, Xu X. (2023). Blockchain-assisted server placement with elitist preserved genetic algorithm in edge computing. *IEEE Internet of Things Journal,* vol. 10, no. 24, pp. 21401-21409. https://doi.org/10.1109/JIOT.2023.3290568

[8] Chakraborty S, Mazumdar K. (2022). Sustainable task offloading decision using genetic algorithm in sensor mobile edge computing. *Journal of King Saud University-Computer and Information Sciences*,

vol. 34, no. 4, pp. 1552-1568. https://doi.org/10.1016/j.jksuci.2022.02.014

[9] Apinaya Prethi K N, Sangeetha M. (2022). A multi-objective optimization of resource management and minimum batch VM migration for prioritized task allocation in fog-edge-cloud computing. *Journal of Intelligent & Fuzzy Systems*, vol. 43, no. 5, pp. 5985-5995. https://doi.org/10.3233/jifs-213520

[10] Fan W, Liu J, Hua M, Wu F, Liu Y. (2022). Joint task offloading and resource allocation for multi-access edge computing assisted by parked and moving vehicles. *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 5314-5330. https://doi.org/10.1109/TVT.2022.3149937

[11] Gabi D, Dankolo N M, Muslim A A. (2022). Dynamic scheduling of heterogeneous resources across mobile edge-cloud continuum using fruit fly-based simulated annealing optimization scheme. *Neural Computing and Applications*, vol. 34, no. 16, pp. 14085-14105. https://doi.org/10.1007/s00521-022-07260-y

[12] Chen M, Qi P, Chu Y, Wang B, Wang F, Cao J. (2024). Genetic algorithm with skew mutation for heterogeneous resource-aware task offloading in edge-cloud computing. *Heliyon*, vol. 10, no. 12. https://doi.org/10.1016/j.heliyon.2024.e32399

[13] Liu Q, Mo R, Xu X, Ma X. (2024). Multi-objective resource allocation in mobile edge computing using PAES for Internet of Things. *Wireless Networks*, vol. 30, no. 5, pp. 3533-3545. https://doi.org/10.1007/s11276-020-02409-w

[14] Chen Q H, Wen C Y. (2024). Optimal resource allocation using genetic algorithm in container-based heterogeneous cloud. *IEEE Access*, vol. 12, pp. 7413-7429. https://doi.org/10.1109/ACCESS.2024.3351944

[15] Jia M, Zhang L, Wu J, Guo Q, Gu X. (2022). Joint computing and communication resource allocation for edge computing towards Huge LEO networks. *China Communications*, vol. 19, no. 8, pp. 73-84. https://doi.org/10.23919/JCC.2022.08.006

[16] Li Y, Yang B, Wu H, Han Q. (2022). Joint offloading decision and resource allocation for vehicular fog-edge computing networks: A contract-stackelberg approach. *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 15969-15982. https://doi.org/10.1109/JIOT.2022.3150955

[17] Zhou H, Wu T, Chen X, He S. (2022). Reverse auction-based computation offloading and resource allocation in mobile cloud-edge computing. IEEE *Transactions on Mobile Computing*, vol. 22, no. 10, pp. 6144-6159. https://doi.org/10.1109/TMC.2022.3189050

[18] Singh G, Chaturvedi A K. (2024). Hybrid modified particle swarm optimization with genetic algorithm (GA) based workflow scheduling in cloud-fog environment for multi-objective optimization. *Cluster Computing*, vol. 27, no. 2, pp. 1947-1964. https://doi.org/10.1007/s10586-023-04071-1

[19] Wen J, Yang J, Wang T. (2023). Energy-efficient task allocation for reliable parallel computation of cluster-based wireless sensor network in edge computing. *Digital Communications and Networks*, vol. 9, no. 2, pp. 473-482. https://doi.org/10.1016/j.dcan.2022.06.014

[20] Kabadurmus O, Erdogan M S. (2023). A green vehicle routing problem with multi-depot, multi-tour, heterogeneous fleet and split deliveries: a mathematical model and heuristic approach. *Journal of Combinatorial Optimization*, vol. 45, no. 3, pp. 89-93. https://doi.org/10.1007/s10878-023-01016-7

[21] Gheisari M, Hamidpour H, Liu Y, Saedi P, Raza A, Jalili A, Rokhsati H, Amin R. (2023). Data mining techniques for web mining: A survey. *Artificial Intelligence and Applications*, vol. 1, no. 1, pp. 3-10. https://doi.org/10.47852/bonviewAIA2202290