UDC 621.391

RESOURCE ALLOCATION SCHEME BASED ON GRAY WOLF OPTIMIZATION ALGORITHM IN MOBILE EDGE COMPUTING

LIU YU

Henan University, The People's Republic of China

Received April 11, 2024

Abstract. Aiming at the problem of server resource allocation in Mobile Edge Computing (*MEC*) task migration, considering the limitation and heterogeneity of server resources, a resource allocation model aiming at minimizing the system server energy consumption is constructed. And proposed a Dynamic Weighted Grey Wolf Optimization (*DWGWO*) algorithm to solve the optimal resource allocation scheme. The initialization, convergence factor and update strategy of Gray Wolf Optimization (*GWO*) algorithm are improved to improve the convergence performance of the algorithm and avoid the algorithm falling into local optimum. Simulation results show that *DWGWO* algorithm can find the optimal solution of resource allocation scheme quickly, reduce the energy consumption of server operation, and is superior to Random strategy, Genetic algorithm and *GWO* in convergence and stability.

Keywords: mobile edge computing, resource allocation, grey wolf optimizer, dynamic weight.

Introduction

Mobile Edge Computing (*MEC*) is a new type of wireless network architecture that deploys edge servers close to users to provide task migration services [1]. Due to the limited resources of edge servers, a reasonable resource allocation scheme is the key to improving task migration effectiveness and effectively reducing task execution delay and energy consumption [2-3].

Aiming at the resource allocation problem, Yu et al. proposed an optimization problem aiming at minimizing the execution delay of all mobile devices and the weighted sum of *UAV* energy consumption, and solved the problem based on an effective algorithm of continuous convex approximation [4]. In order to minimize energy consumption, You et al. transformed the server resource allocation problem into an optimal scheduling sequence problem, and conducted joint optimization of data partitioning and time division [5]. Mao et al. modeled the resource allocation problem under multiple users as a stochastic optimization problem and proposed an online algorithm to solve the problem [6]. In order to maximize the total revenue of the system, Kim et al. modeled the resource allocation problem as a single-leader - multi-user Stackelberg game model to achieve a balance between mobile users and server utility [7]. However, the above researches focus on optimizing the execution delay and energy consumption of task migration, and do not consider the heterogeneity of edge servers, such as the computing, storage and communication capabilities of *MEC*.

To solve the above problems, this paper first considers the heterogeneity of server resources, and constructs a server task migration resource allocation model. Then, the Dynamic Weighted Grey Wolf Optimization (*DWGWO*) algorithm is proposed to solve the optimal server resource allocation scheme, utilize the effective resources of the server effectively, and minimize the server energy consumption. *DWGWO* algorithm makes the following improvements to Grey Wolf Optimization (*GWO*) algorithm: Use Tent chaotic mapping initialization instead of random initialization to ensure the diversity of initial resource allocation schemes. And cosine change is used to replace the convergence factor of linear change. In the iterative updating process, the dynamic weight method is used to improve the linear convergence factor of *GWO* algorithm to change with the cosine of the number of iterations, so as to fit the trend of resource allocation scheme first global search and then fast convergence. In order to avoid

GWO algorithm falling into local optimal solution, dynamic update weights are set to increase the randomness of update.

Server resource allocation model

Each edge server provides services to each mobile device, and each mobile device has one and only one task. Edge servers are heterogeneous with different computing resources.

The base station collects the *MEC* system information (including the task information on the device side and the server resource information on the server side). In the case of limited server resources, considering the delay constraints of the task, the resource allocation scheme is optimized, and the server resources are reasonably allocated to users to complete the task.

Let $I=\{I_n | n=1,2,\cdots,N\}$ denotes the set of tasks in the *MEC* network. And $I_n=\{B_n, D_n, Z_n\}$, where I_n denotes the task of the mobile device n, B_n denotes the size of task, D_n denotes the amount of computation required to complete the task, and Z_n denotes the delay limit of the task. Let $E=\{E_m | m=1,2,\cdots,M\}$ denotes represent the set of edge servers, where server m is represented by $E_m=\{B_m, S_m, V_m, R_m, C_m\}$, B_m is the maximum communication bandwidth, S_m is the maximum storage space, V_m is the number of virtual machine included (representing the computing capacity of the server), R_m is the computing speed of the virtual machine, and C_m is the power consumption per unit of the virtual machine. The paper optimizes the resource allocation scheme to minimize the energy consumption of edge servers. The problem can be expressed as $C_{total}=\sum_{m=1}^{M}\sum_{n=1}^{N} x_{nm} C_m t^s_{nm,p}$ indicates whether task I_n selects a server E_m. $x_{nm}=1$ indicates that task I is executed on server E_m, $x_{nm}=0$ indicates that no server E_m is selected. The task completion delay includes task upload delay $t^s_{nm,t}$, task calculation delay $t^s_{nm,p}$. Specific as follows $t^s_{nm,t}=\frac{B_n}{r_n}$, $t^s_{nm,p}=\frac{D_n}{R_{nm}}$, where r_n indicates the wireless channel upload rate, and R_{nm} indicates the computing speed of the server m selected by mobile device.

According to the above objective functions and constraints, this paper designs a resource allocation scheme based on dynamic weighted Gray Wolf Optimization (*DWGWO*). The Grey Wolf optimization algorithm (*GWO*) is applied to the design of resource allocation scheme, so that the optimal solution of resource allocation scheme can be found quickly and the energy consumption of server can be reduced.

Design of resource allocation scheme based on Dynamic Weight Grey Wolf Optimization

Grey Wolf Optimization (*GWO*) algorithm compares the solution space of the problem to the hunting space of wolves based on the hunting behavior of wolves [8]. Each wolf as a search individual to represent the search for target food, and the optimal solution required for optimization is the prey to be sought by wolves. It has the advantages of simple structure, few parameters to be adjusted, easy implementation, and good performance in terms of problem solving accuracy and convergence speed [9]. The *GWO* algorithm divides resource allocation schemes into four levels: a, b, c and d, where a is the leader, which denotes the optimal allocation scheme for the current iteration. b denotes the second best choice, c denotes the third best choice. The rest are defined as d, and they are guided and updated by a, b, c in the iterative process. As shown in equation

$$\begin{cases} X_1 = X_a - A_1 \cdot D_a \\ X_2 = X_b - A_2 \cdot D_b, \\ X_3 = X_c - A_3 \cdot D_c \end{cases} \tag{1}$$

where X_a, X_b, X_c respectively denote the current position of a, b, c. And D_a, D_b, D_c as show in equation

$$\begin{cases} D_a = |C_1 X_a - X| \\ D_b = |C_2 X_b - X|, \\ D_c = |C_3 X_c - X| \end{cases} \tag{2}$$

where D_a, D_b, D_c respectively denote the distance between d and a, b and c. In which, X denotes the position d. A and C is the vector of cooperation coefficient, calculated by the following equation

$$A = 2er_1 - e, \tag{3}$$

$$e = 2 - \frac{2t}{T}, \tag{4}$$

$$C = 2r_2, \tag{5}$$

where e is the convergence factor, decreasing linearly from 2 to 0 as the number of iterations. T is the maximum number of iterations. r_1 and r_2 are random numbers between 0 to 1.

Algorithm Improvement

The quality of the initial population can affect the convergence speed of the algorithm and the quality of the solution. In *GWO* algorithm, the initial population is generated randomly, and population diversity cannot be guaranteed. In the optimization process of resource allocation scheme, the algorithm should conduct sufficient global search in the early stage and accelerate the convergence in the later stage, but the linear change of *GWO* algorithm convergence factor cannot fit this trend. Moreover, *GWO* algorithm is easy to fall into local optimal solution during the updating process.

Based on the above analysis, on the basis of *GWO*, Tent chaotic mapping is used to initialize the population to ensure the diversity of the initial population. The convergence factor value changes dynamically with the cosine of the number of iterations to better fit the trend of fast convergence after global exploration. By setting dynamic update weights, can increase the randomness of update and avoid the situation of unfair resource allocation.

Tent mapping is used to initialize the population to improve the ergodicity and diversity of the population. The chaotic sequence generation formula for Tent mapping is as follows

$$y_{t+1} = \begin{cases} 2y_t & 0 \le y_t \le 0,5 \\ 2(1-y_t) & 0,5 \le y_t \le 1 \end{cases}. \tag{6}$$

In this section, formula (6) is improved by setting cosine factor cosine change, expressed as

$$e = 2 \cdot \cos\left(\frac{3,14}{2} \cdot \frac{t}{T}\right). \tag{7}$$

In order to enhance the flexibility and adaptability of the algorithm, the key element of randomness is introduced into the updating process

$$X(t+1) = \frac{X_1 + \left(F_a / F_b\right) X_2 + (F_a / F_c) X_3}{3}, \tag{8}$$

where F_a denotes the fitness value of a, F_b denotes the fitness value of b, and F_c denotes the fitness value of c.

Simulation experiment

In order to verify the performance of *DWGWO* algorithm, it is compared with Random strategy, Genetic algorithm and *GWO* algorithm. All experimental results are the average of 30 times running of the algorithm.

Figure 1 shows the convergence trend of the four algorithms with the number of iterations when the number of servers is 6, the number of tasks is 30, and the size of tasks is 30. As is shown in the picture. In the 300 iterations of the Random strategy, the result does not show any regularities, and the total energy consumption of the server obtained by the obtained resource allocation strategy is basically higher than that of the strategy sought by the algorithm. The *DWGWO* algorithm converges rapidly in the first 50 iterations, and becomes basically stable when the iteration reaches 200. Compared with Genetic algorithm and *GWO*, the convergence is faster in the first 100 iterations, and it also tends to be stable in the 200 iterations. And in terms of convergence accuracy, *DWGWO* can obtain lower total server energy consumption.

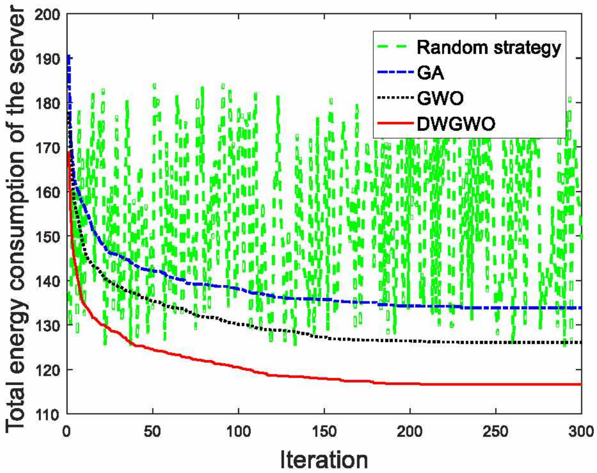

Figure 1. Convergence trend of the four algorithms with the number of iterations

Genetic algorithm, *GWO* algorithm and *DWGWO* algorithm are all population-based heuristic algorithms, and the number of of the three algorithms under different population numbers was tested. Figure 2 shows the population has a direct impact on the performance of the algorithm. Therefore, the comparison of energy consumption parison of convergence results of the three algorithms under different populations when the number of tasks is set to 30, the number of servers is set to 6, the size of tasks is set to 30, and the iteration period is set to 300.

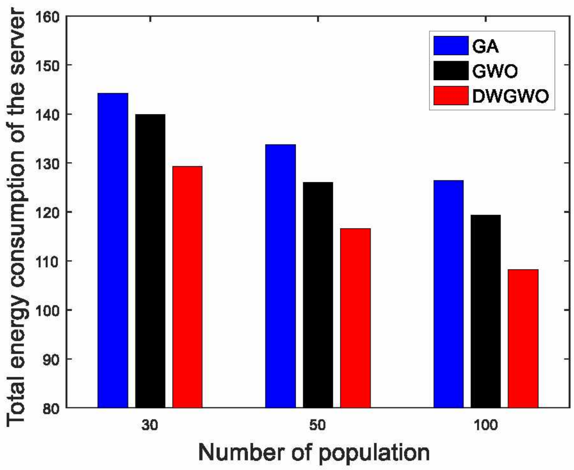

Figure 2. Convergence results of the three algorithms under different population numbers

As can be seen from the figure, when the population size is 30, 50 and 100, the *DWGWO* algorithm can obtain lower server energy consumption. When the population size is 30, the energy consumption obtained by *DWGWO* algorithm is reduced by 0,1031 compared with Genetic algorithm and 0,07 compared with *GWO*. When the population size is 50, the energy consumption obtained by *DWGWO* is reduced by 0,1287 compared with Genetic algorithm and 0,075 compared with *GWO*. When the population is 100, the energy consumption of *DWGWO* algorithm is 0,1435 less than that of Genetic algorithm and 0,0927 less than that of *GWO*.

Under the condition that the number of servers is 6, the number of tasks is 30, and the size of tasks is 30. The three algorithms were simulated for 30 times, and the optimal solution, the worst solution, the average solution and the variance of the obtained total server energy consumption are compared. As can be seen from Table 1, the convergence accuracy of Genetic algorithm is the worst, but the variance is small. That is, the algorithm has good stability. The convergence accuracy and stability of ray wolf optimization algorithm are general. *DWGWO* algorithm is superior to *GWO* in convergence accuracy and stability. Stability is slightly worse than genetic algorithm. It can be seen that when *DWGWO* algorithm solves the problem of multi-user resource allocation, its convergence speed and convergence accuracy are higher than the other three algorithms, and it also has good algorithm stability.

Table 1. **Stability comparison of the three algorithms**

Methods	Genetic algorithm	*GWO*	*DWGWO*
Worst solution	140,5796	134,3470	123,3593
Optimal solution	122,6979	115,6979	106,3851
Mean value	133,8140	126,0505	116,5948
Variance	17,26039	23,75953	18,53467

Conclusion

Aiming at the problem of resource allocation in mobile edge computing, this paper proposes a resource allocation scheme based on *DWGWO*. Tent chaotic mapping is used to initialize the population to ensure the diversity of initial resource allocation schemes. Secondly, the adjustment value changes dynamically with the cosine of the number of iterations to better fit the trend of resource allocation first global exploration and then rapid convergence. Finally, the randomness of the update is increased by assigning different weights to the fitness values. The simulation results show that the resource allocation scheme based on *DWGWO* can quickly find the optimal allocation solution and reduce the energy consumption of the server.

References

1. Abbas N., Zhang Y., Taherkordi A., et al. // IEEE Internet of Things Journal. 2018. Vol. 5. P. 450–465.
2. Zhang Kaiyuan., GUI Xiaolin., Ren DeWang., et al. // Journal of Software. 2019. Vol. 30(08). P. 2491–2516.
3. Xu Y., Gui G., Gacanin H., et al. // IEEE Communications Surveys & Tutorials. 2021. Vol. 23(2). P. 668–695.
4. Yu Z., Gong Y., Gong S., et al. // IEEE Internet of Things Journal. 2020. Vol. 7(4). P. 3147–3159.
5. You C., Zeng Y., Zhang R., et al. // 2018 IEEE International Conference on Communications Workshops (ICC Workshops). 2018. P. 20–24.
6. Li H., He X., Ding S. // 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA). IEEE. 2019. P. 258–263.
7. Kim S., Park S., Chen M., et al. // IEEE Communications Letters. 2018. Vol. 22(9). P. 1922–1925.
8. Mirjalili S., Mirjalili S.M., Lewis A. // Advances in Engineering Software. 2014. Vol. 69. P. 46–61.
9. Zhang Xiaofeng, Wang Xiuying. // Computer Science. 2019. Vol. 46(03). P. 30–38.