Resource Allocation Optimization Study of Cloud-Edge-End Collaboration Based on QUBO Model

Liangyu Qin
School of Electronic and Information Engineering
Beihang University
Beijing, China
qly2422603874@outlook.com

2024 IEEE 6th International Conference on Power, Intelligent Computing and Systems (ICPICS) | 979-8-3503-7431-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICPICS62053.2024.10796207

***Abstract*—The core of this article is to construct an effective QUBO model to solve the problem of cloud edge end computing resource allocation, including two key scenarios: optimization of server maximum computing demand coverage and optimization of cloud edge server computing network layout. Each scenario provides a detailed description of the model's process from analysis, establishment to solution, overcoming several major difficulties in establishing a QUBO model, including using binary variables to express the objective function, transforming the objective function into QUBO expressions, and transforming inequality constraints. A scheme for the number of constraint relaxation factors is introduced, which effectively reduces the number of quantum bits and computational complexity. Finally, statistical analysis of the solution results, non parametric testing based on data characteristics, and detailed analysis are conducted, demonstrating the potential application of the QUBO model and quantum computing in dealing with complex combinatorial optimization problems. During the solving process, China's first optical quantum computing power cloud platform and corresponding quantum computing library Kaiwu SDK were utilized for research. Multiple solvers were used and their performance was compared to select the best performing taboo search solver, and then detailed results were calculated. The final results were fully visualized to make them more intuitive. The results indicate that quantum computing can provide an efficient, accurate, and stable solution for cloud edge end computing resource allocation. Through these studies, this article not only confirms the efficiency of quantum computing in solving specific optimization problems, but also provides some experience and reference for exploring the application of quantum computing in a wider field.**

Keywords—QUBO Model, Cloud-Edge-End, Kaiwu SDK, Quantum Computing

I. Introduction

With the rapid development of information technology, there is a large-scale demand for computing resources in multiple fields. Especially in the training of artificial intelligence large models, data processing for autonomous driving, and cloud computing services for government and business, the demand for computing resources is growing day by day. According to research, the traffic of global data centers is expected to triple in the next five years[1], while the complexity of artificial intelligence models is increasing at a rate of 10 times per year . The computational complexity of accurately solving combinatorial optimization problems in application scenarios is also increasing exponentially. This not only poses higher requirements for computing speed, but also challenges the efficiency of computing resource allocation. Existing solving techniques often struggle to complete these solutions within a reasonable time frame. In addition, as traditional computing power approaches the limit defined by Moore's Law, quantum computing has become a promising technology that breaks through the computational bottleneck of traditional computers. In the distributed resource aggregation optimization method of virtual power plants, the use of optical quantum machines for solving takes 1/10 of Gurobi's computational time, and is much smaller than classical algorithm solvers such as Cplex, which proves that quantum computers have broad prospects [2]. This article will focus on the combination optimization scenario of cloud edge end collaboration, establish a QUBO model that can be solved by quantum computers, and promote the application fields of quantum computing[3](The data used in this paper are all reference data for experimental simulation purposes).

II. Optimization of Server Maximum Computing Demand Coverage

A. Research on the Installation Location of Edge Servers

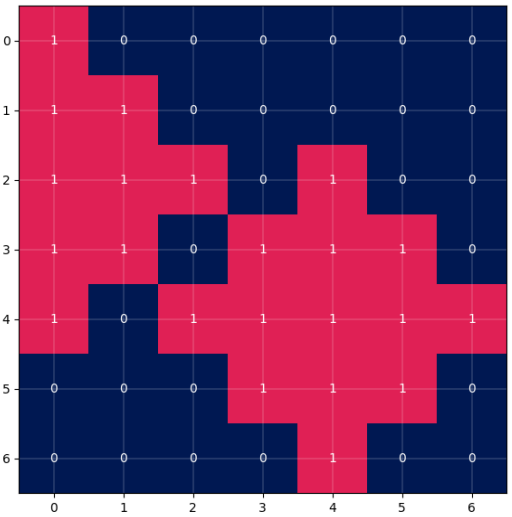


Figure 1 Schematic diagram of edge server coverage effect

The computing needs of users within a certain range can be represented by a two-dimensional plane, which is segmented using a square grid. To simplify the problem, the center of each grid is the computing needs of users in that area, which are generated by user terminals, including mobile devices, industrial robots, sensors, etc. These computing needs need to be calculated using servers. If it is in a situation where edge servers are limited, then it should be considered to meet the computing needs of users as much as possible. For this purpose, this chapter establishes the QUBO model, which not only effectively solves combinatorial optimization

problems, but also can be solved using coherent Ising machines (CIM) without being limited by traditional computing power. For a given number of edge servers, install them in appropriate locations to maximize the coverage of user computation, and the calculation range of each edge server is limited. In the grid model, Euclidean distance can be used to represent the relationship between users and edge servers. If this Euclidean distance is greater than the coverage radius of the server, it cannot be accepted and calculated by the server. Fig. 1 shows the schematic diagram of the coverage range of two edge servers with a coverage radius of 2 in the range of 7 * 7, located at (2,0) and (4,4).

This article will deploy a given number of edge servers within the grid area based on the given distribution of computing demands, in order to maximize the coverage of the total computing value. Assuming that the distribution of user computing demands is a 4 * 4 grid, it will be represented by a matrix as follows:

$$demandmatrix = \begin{pmatrix} 36 & 20 & 63 & 54 \\ 51 & 46 & 74 & 44 \\ 54 & 34 & 5 & 27 \\ 48 & 35 & 46 & 38 \end{pmatrix} \quad (1)$$

Where di and j represent the total computational demand of users located at the center of the (i, j) grid.

B. Establishment and Solution of the Model

(1) objective function

The total amount of computation covered by edge servers is taken as the opposite number, as the maximum solution is required:

$$\text{SumDemand} = -\sum_{i,j} y_{i,j} \cdot d_{i,j} \quad (2)$$

Due to the fact that in binary optimization problems, decision variables can only take 0 and 1, i.e. $y_{ij}=y_{ij}^2$, equation 6 can be written in QUBO form:

$$\text{SumDemand} = -\sum_{i,J} y_{i,j}^2 \cdot d_{i,j} = -\mathbf{Y}^T \mathbf{D}\mathbf{Y} \quad (3)$$

Among,

$$Y = \left[y_{11}\ y_{12}\ y_{13} \cdots y_{IJ}\right]^T\Big|_{I=J=4} = \left[y_1\ y_2\ y_3 \cdots y_{4I+J-4}\right]^T \quad (4)$$

$$D = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & d_{4i+j-4} \end{bmatrix} \quad (5)$$

(2) constraint condition

The number of deployed edge servers must be equal to the specified number:

$$y_{i,j} \le \sum_{k,m} x_{k,m}, \sqrt{(i-k)^2 + (j-m)^2} \le R \quad (6)$$

As long as any server is deployed within the radius range of i and j, it will make yi, j=1, which is equivalent to the grid being covered.

(3) QUBO modeling

Constraints on the number of servers:

$$\begin{aligned} \text{Constraint1} &= \left(\sum_{i,j} x_{i,j} - \text{EdgeNum}\right)^2 \quad (7) \\ &= \sum_j \sum_k x_j x_k - 2 \cdot EN \sum_i x_i + EN^2 \\ &= \sum_j \sum_{k(k\neq j)} x_j x_k + \sum_i x_i^2 - 2 \cdot EN \sum_i x_i + EN^2 \\ &= \sum_j \sum_{k(k\neq j)} x_j x_k + (1 - 2 \cdot EN) \sum_i x_i^2 + EN^2 \end{aligned}$$

Write it in the matrix form of the QUBO model[4], with

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{4I+J-4} \end{bmatrix}^T \begin{bmatrix} 1-2EN & 1 & \cdots & 1 \\ 1 & 1-2EN & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 1-2EN \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{4I+J-4} \end{bmatrix} \quad (8)$$

If there are edge servers deployed within the coverage radius around the grid, the grid must be covered, and a relaxation factor is introduced here ξ, Convert inequality constraints into equality constraints:

$$Constraint2 = \sum_{i,j} \left(\sum_{k,m} x_{k,m} - y_{i,j} - \xi_{i,j}\right)^2 \quad (9)$$

In the formula, $\sqrt{\left((i-k)^2 + (j-m)^2\right)} \le R$ need to be met, while $\varsigma_{i,j} = \sum_N z_{i,j,k}$. Theoretically, when N is large enough, $\varsigma_{i,j}$ can be expressed as any integer greater than or equal to zero. In order to find the optimal solution more accurately and efficiently, it is necessary to make N as small as possible, but at the same time, it is necessary to make the constraint equation hold, so it is necessary to analyze $\sum_{k,m} x_{k,m} - y_{i,j}$. The range of values for m xk, m − yi, and j. This value is limited by both the coverage range of edge servers and the expected number: For the coverage range of edge servers: By searching for patterns, the maximum number of grids covered by edge servers with a coverage radius of R is (2R-1) 2+4.For the number of edge servers: the number of edge servers that can cover a certain grid does not exceed the total number of servers it is expected to deploy.

$$N = \mathrm{Min}\left(2R-1\right)^2 + 4,\mathrm{EdgeNum}(10)$$

Further research has found that the value of N can be further reduced based on Max {(2R-1) 2+4, EdgeNum}, and ξ can be represented as any real number, i.e

$$\xi_{i,j} = \sum_{k=1}^{N} 2^{k-\gamma} \cdot z_{i,j,k}(11)$$

In the formula, γIt is a binary numerical control variable, where zi, j, and k represent the k-th quantum bit constrained by (i, j) position ; If here γIf a number greater than 1 is taken, then ϵ i.J can be expressed as decimals, so the exact value of N can be directly expressed as:

$$N = ceil\left(\log_2\left(\mathrm{Min}\left(2R-1\right)^2 + 4,\mathrm{EdgeNum}\right)\right)(12)$$

In the formula, ceil() represents rounding up.

Taking the originally set parameters as an example (with a server coverage radius of 1), a maximum of 5 can be deployed around each grid An edge server can be represented as N=3 by substituting into equation 13. According to equation 13, an integer not greater than 5 can also be expressed as: $z_{i,j,1} + 2z_{i,j,2} + 4z_{i,j,3}$, their values are shown in Table I.

TABLE I DISCUSSION ON THE RANGE OF RELAXATION FACTOR VALUES

$z_{i,j,1}$	$z_{i,j,2}$	$z_{i,j,3}$	$\xi_{i,j}$
0	0	0	0
1	0	0	1
0	1	0	2
1	1	0	3
0	0	1	4
1	0	1	5

The use of Equation 14 can increase the spatial complexity of the solution from O (n) in Formula 10 to O (log2n), and significantly reduce the computational cost. In this paper, only three relaxed quantum bits can fully represent integers in the range of 0-5, which is more suitable for existing quantum bit limited devices.

C. Model Solving

Write a Python program to convert the QUBO model into a CIM model, and use the taboo search solver of the Kaiwu library to calculate[4]. Set the parameters as shown in Table II.

TABLE II OPTIMIZATION RESULTS FOR DIFFERENT NUMBERS OF SERVERS

EdgeNum	Maximum calculated coverage value	Edge1 coordinates	Edge2 coordinates	Edge1 coordinates
1	232	(2,3)	——	——
2	419	(2,3)	(3,1)	——
3	561	(1,3)	(2,1)	(4,2)
4	675	(1,2)	(2,4)	(3,1)
5	675	(1,2)	(2,1)	(2,4)
...	...	...	...	...
8	675	(1,2)	(1,3)	(2,1)

In order to present the results more intuitively, the visualization is shown in Fig. 2, where the blue area represents the uncovered grid, the red area represents the covered grid, and the yellow circle represents the grid where the server is deployed.

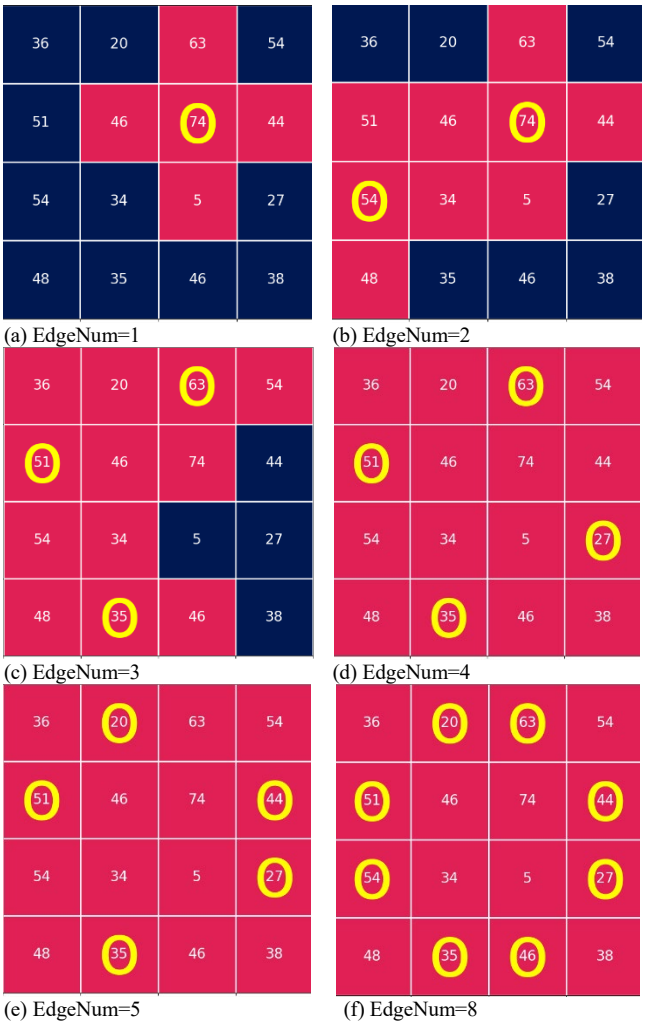
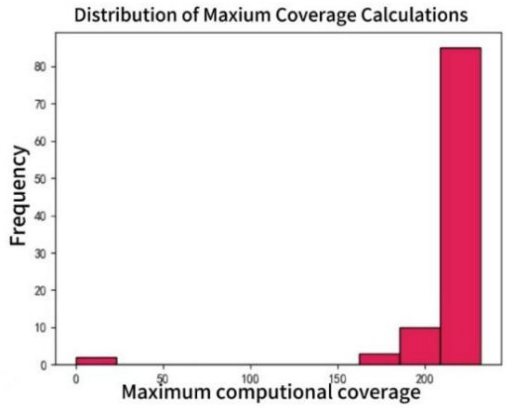


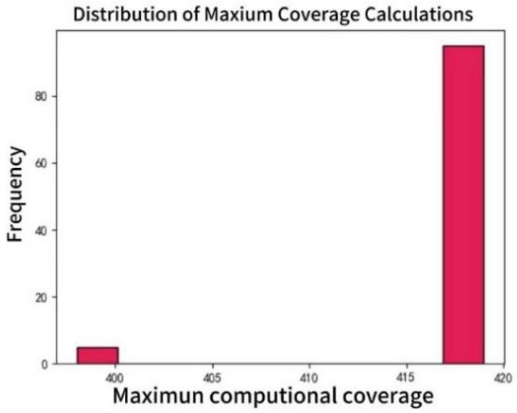
Figure 2 Optimal Deployment Visualization

D. Result Verification Analysis

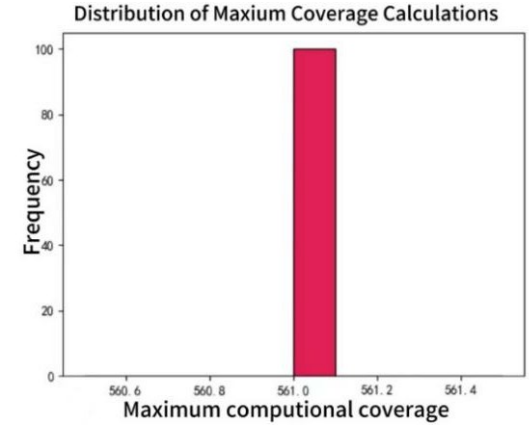
In order to make the results more reliable, the maximum values of the above scenarios with the number of edge servers were tested as follows: repeat the experiment and conduct statistical analysis: run the code solution multiple times, calculate the average and standard deviation of the solution results to evaluate the stability and reliability of the algorithm. Non parametric testing: Statistical testing can determine whether performance differences are statistically significant, providing more rigorous comparative results. Due to the fact that the experimental data clearly does not follow a normal distribution and the experimental results do not exceed the expected maximum (i.e., the maximum value), the original hypothesis H0 is defined using Bootstrap based non parametric testing and random permutation based non parametric testing, respectively. It is assumed that there is no significant difference between the average value of the maximum coverage calculation obtained from the experiment and the expected maximum value[5].Partial test analysis results are shown in Fig. 3.



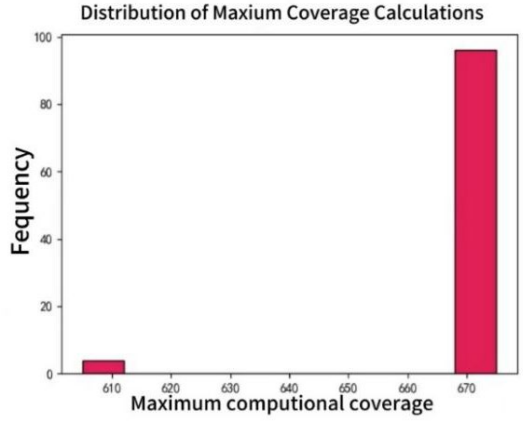
(a) EdgeNum=1



(b) EdgeNum=2



(c) EdgeNum=3



(d) EdgeNum=4

Figure 3 Visualization of Maximum Coverage Calculation Quantity Distribution

Analyzing the results to test the data, it was found that the standard deviation was very small, indicating that the algorithm has good stability and reliability; All 95% confidence intervals contain 0 or the upper limit is very close to 0, indicating that the difference between the actual observed value and the expected value is very small. The P-values of both test methods are far greater than the significance level of 0.05, indicating that there is not enough evidence to reject the null hypothesis, that is, there is no significant difference between the experimental value and the expected value[6].

III. Optimization of Cloud Edge Server Computing Network Layout

Data Definition: Assume that the user computation requirement is a 6*6 grid, represented by a matrix as:

$$\text{demandmatrix} = \begin{pmatrix} 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \\ 3 & 0 & 4 & 0 & 8 & 0 \\ 0 & 4 & 0 & 11 & 0 & 0 \\ 0 & 6 & 0 & 0 & 3 & 0 \\ 0 & 0 & 7 & 0 & 0 & 5 \end{pmatrix} \tag{13}$$

where demandmatrix(i.j) denotes the total user computing demand located at the center of the (i,j) grid. And the goal is to minimize the total cost of using edge servers and cloud servers for end-user computing needs.

A. Establishment and Solution of the Model

(1) Objective function

The objective function consists of three parts: fixed cost, computational cost, and transmission cost.

fixed cost:

$$FixedCost = \sum_{k,m} f_{k,m} \cdot y_{k,m} \tag{14}$$

Since $y_{k,m}$ and m are binary variables, $y_{k,m} = y_{k,m}^2$, therefore the fixed cost is converted into the QUBO model expression:

$$FC = Y^T FY = \begin{bmatrix} y_1 y_2 \cdots y_J \end{bmatrix} \begin{bmatrix} f_1 & 0 & \cdots & 0 \\ 0 & f_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_J \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_J \end{bmatrix}_{J=5} \tag{15}$$

Edge server computing costs:

$$2\sum_{k,m}\left[\sum_{i,j} d_{i,j}\, x_{i,j,k,m}\left(1 - c_{k,m}\right) + 2 \cdot maxsize \cdot c_{k,m}\right] \tag{16}$$

In the formula, if $c_{k,m} = 0$, it indicates that the edge server is not connected to the cloud server, and the total computing power is the total computing power of all connected edge servers. If $c_{k,m} = 1$, that is, the edge server is

connected to the cloud server, it indicates that the total computing power of the edge server is greater than its maximum capacity, and its computing power will be the maximum computing power. The rest will be transmitted to the cloud server.

Cloud server computing cost(CompCostEdge) is:

$$\sum_{i,j} d_{i,j} \cdot z_{i,j} + \sum_{k,m} \left(\sum_{i,j} d_{i,j} \cdot x_{i,j,k,m} - \text{maxsize} \right) \cdot c_{k,m} (17)$$

Transmission cost from users to edge server:

$$\text{TranCostUserEdge} = \sum_{k,m} \sum_{i,j} d_{i,j} \cdot \text{S}_{i,j,k,m} \cdot x_{i,j,k,m} (18)$$

Transmission cost from edge servers to cloud server, which is:

$$\sum_{k,m} \left(\sum_{i,j} d_{i,j} \cdot x_{i,j,k,m} - \text{maxsize} \right) \cdot \text{S1}_{k,m} \cdot c_{k,m} (19)$$

And the transmission cost from users to cloud server:

$$\text{TransCostUserCloud} = 2 \cdot \sum_{i,j} d_{i,j} \cdot \text{S0}_{i,j} \cdot z_{i,j} (20)$$

(2) Constraint condition

The calculation radius of the edge server is 3. As the Kaiwu quantum computing library currently only supports QUBO matrices of size 100 bits, in order to reduce QUBO variables and meet the constraint conditions, the following approach is adopted: first calculate the Euclidean distance from each user (i, j) to each edge server (k.m), and if it is less than or equal to 3, then design the corresponding QUBO variable $x\_\{i,j,k,m\}$.

Each user can only connect to one server:

$$\sum_{k,m} x_{i,j,k,m} + z_{i,j} = 1 \quad \forall i,j (21)$$

The server that the user is connected to must have already been deployed:

$$x\_\{i,j,k,m\} - y\_\{k,m\} \leq 0 \quad (22)$$

The edge server that connects to the cloud server must have already been deployed:

$$c\_\{k,m\} - y\_\{k,m\} \leq 0 \quad (23)$$

Edge server capacity limit:

$$\left(\sum_{i,j} d_{i,j} \cdot x_{i,j,k,m} - \text{maxsize} \right) < c_{k,m} \cdot maxsize (24)$$

If the total computational load of edge servers k and m is greater than their maximum load, which is $\sum_{i,j} d_{i,j} \cdot x_{i,j,k,m} - \text{maxsize} > 0$, $c_\{k,m\}$will take a value of 1, which means connecting to the cloud server for computation.

(3)QUBO model

To convert the objective function and constraints into a QUBO model, a penalty term needs to be introduced to ensure that the constraints are met. Set a large positive number P as the penalty coefficient.

Penalty items for user connection constraints:

$$UCP = P \cdot \sum_{i,j} \left(\sum_{k,m} x_{i,j,k,m} + z_{i,j} - 1 \right)^2 (25)$$

Penalty items that must have been deployed on the server that the user is connected to[7][8]:

$$UCT = P \cdot \sum_{k,m} \sum_{i,j} \left(x_{i,j,k,m} - y_{k,m} \cdot x_{i,j,k,m} \right) (26)$$

Penalty items that must have been deployed for edge servers connected to cloud servers:

$$CCT = P \cdot \sum_{k,m} \left(c_{k,m} - y_{k,m} \cdot c_{k,m} \right) (27)$$

If the maximum computing load of the edge server is exceeded, it is necessary to connect to the cloud server. Here, the inequality constraint needs to be transformed into an equality constraint, and corresponding relaxation factors need to be introduced for each edge server :

$$EC = P \cdot \sum_{k,m} \left(\sum_{i,j} d_{i,j} \cdot x_{i,j,k,m} - \text{maxsize} - c_{k,m} \cdot \text{maxsize} + \xi_{k,m} \right)^2 (28)$$

According to Section 2.2.3 of Chapter 2 and Equation [eq11] and Equation [eq12], it can be concluded that:

$$\xi_{k,m} = \sum_{l=1}^{l_{max}} 2^{l-\gamma} \cdot a_{k,m,l} (29)$$

$$l_{max} = ceil \left(log_2 \left(\sum_{i,j} d_{i,j} \cdot x_{i,j,k,m} \right) \right) (30)$$

B. Complete QUBO Objective Function and Solution

Combine all the above cost and penalty terms to obtain the complete QUBO objective function:

$$TotalCost = FC + CCE + CCC + TCUE + TCEC + OEC + TCUC + UCP + CCT + UCT + EC (31)$$

Write a Python program and convert it into a CIM model. Use three solvers in the Kaiwu SDK (taboo search solver, CIM class simulation solver, and simulated annealing solver) to perform calculations, and select the best performing algorithm based on the solution results. The visualization results are shown in Figs. 4-7.

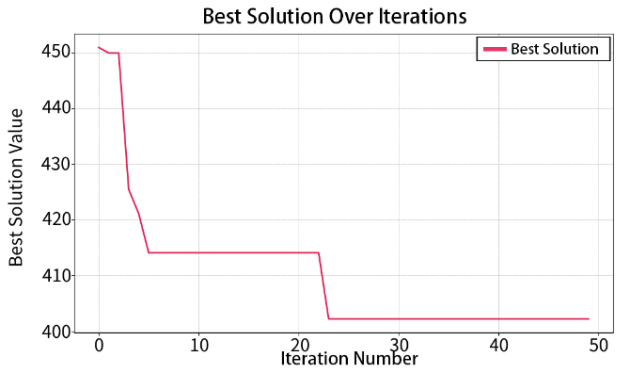


Figure 4 Taboo Search Solver Optimal Solution Iteration Results

Fig. 4 shows the variation curve of the optimal solution of the taboo search solver in 50 iterations, and the final optimization result obtained is 403.972.

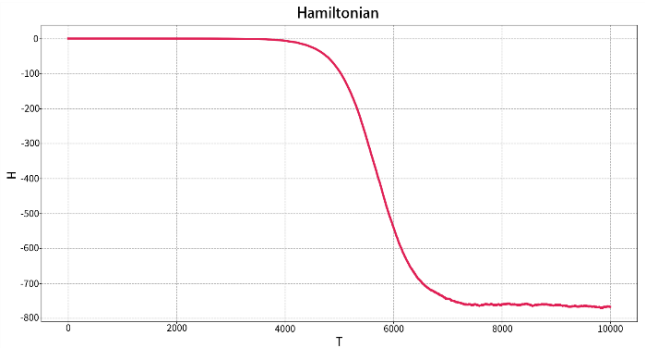


Figure 5 CIM Hamiltonian variation curve

Figs. 5 to 6 show the visualization of the results using a CIM simulation solver, including the variation of the Hamiltonian during iteration and the evolution of quantum bits. The solver gradually evolves the Hamiltonian of the system towards lower energy and converges. During the evolution of quantum bits, the value of the variable is calculated based on its positive and negative values, and it is binarized. Positive values are binarized to 1, while negative values are binarized to -1, corresponding to values of 1 and 0.

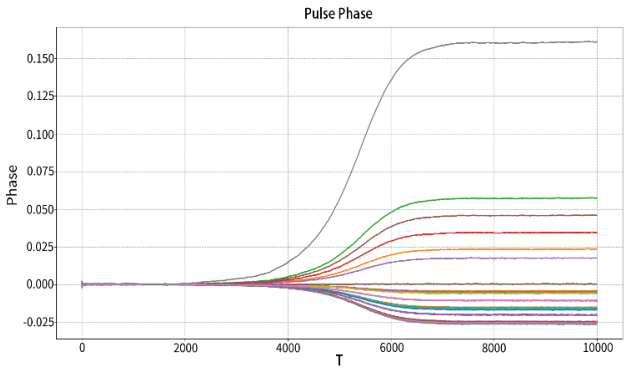


Figure 6 CIM Quantum Bit Evolution Diagram

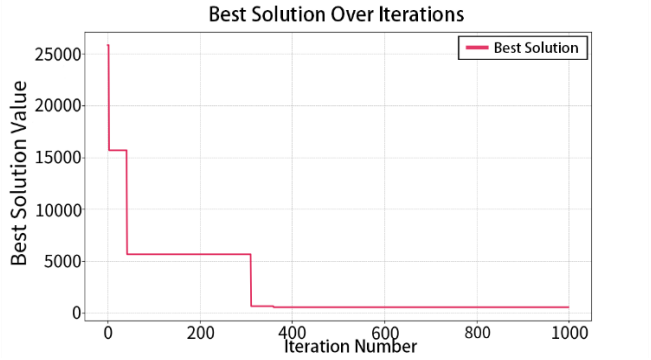


Figure 7 Iteration results of the optimal solution of the simulated annealing solver

Fig. 7 shows the optimal solution variation curve of the simulated annealing solver after 1000 iterations, with a final optimization result of 449.221. Comparing the three algorithms, it can be seen that the taboo search algorithm has the smallest solution result, the least number of iterations, and the best performance. Furthermore, based on the values of binary decision variables in the taboo search algorithm, the specific costs of each item are calculated, as shown in Table III and Table IV.

TABLE III USER SERVER CONNECTION STATUS

User	Server connection status	Transmission costs	Calculate costs
User1(0,3)	Edge2(2,1)	8.000	8
User2(1,5)	Edge2(2,3)	11.180	10
User3(2,0)	Edge5(2,1)	3.000	6
User4(2,2)	Edge5(2,1)	4.000	8
User5(2,4)	Edge3(4,5)	17.889	16
User6(3,1)	Edge5(2,1)	4.000	8
User7(3,3)	Edge2(2,3)	11.000	22
User8(4,1)	Cloud(4,0)	12.000	6
User9(4,4)	Edge3(4,5)	3.000	6
User10(5,2)	Cloud(4,0)	31.305	7
User11(5,5)	Edge3(4,5)	5.000	10

TABLE IV EDGE SERVER DEPLOYMENT AND CLOUD SERVER CONNECTION

Server	Deployment situation	Calculate total load	Cloud server connection	Transmission costs
Edge1(6,1)	0	——	——	——
Edge2(2,3)	1	20	1	28.844
Edge3(4,5)	1	16	1	20.000
Edge4(6,5)	0	——	——	——
Edge5(2,1)	1	11	0	——
Cloud(4,0)	1	25	——	——

In order to present the results more intuitively, the data in Tables III and IV were visualized, and the results are shown in Fig. 8.

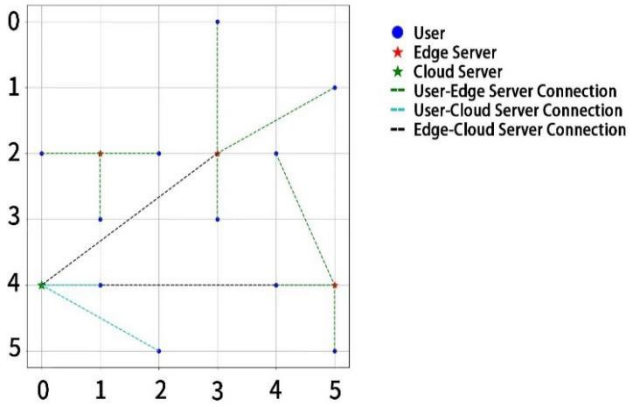


Figure 8 Visualization of Optimization Results

C. Result Verification Analysis

In order to make the results more credible, the optimal cost deployment solution solution obtained in the cloud edge server collaboration scenario, 403.98, was tested through repeated experiments and statistical analysis. The code was run multiple times to solve the problem, and the average and standard deviation of the solution results were calculated to evaluate the stability and reliability of the algorithm. Non parametric testing: Statistical testing can determine whether performance differences are statistically significant, providing more rigorous comparative results. Due to the fact that the experimental data clearly does not follow a normal distribution and the experimental results do not exceed the expected maximum (i.e. maximum), the original hypothesis H0 is defined using Bootstrap based non parametric testing and random permutation based non parametric testing, respectively. It is assumed that there is no significant difference between the average of the maximum coverage calculation obtained from the experiment and the expected maximum[9]. The test results are shown in Table V.

TABLE V STATISTICAL ANALYSIS AND BOOTSTRAP TEST RESULTS

average value	standard deviation	Original difference
410.21	11.97	6.218
95% confidence interval	Bootstrap-P value	Random permutation - P value
(3.21, 9.78)	0.520	0.0001

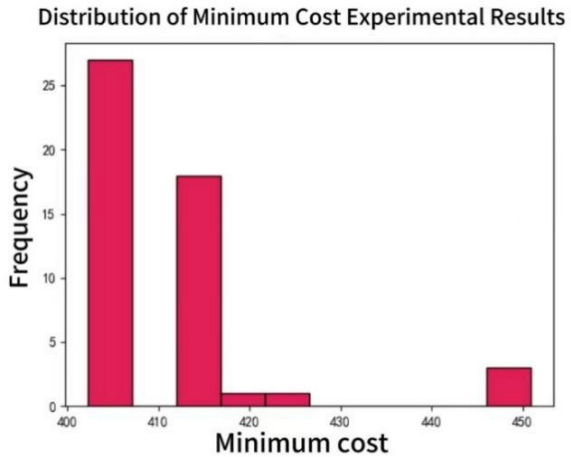


Figure 9 Visualization of experimental result distribution

According to the analysis of the test results in Fig. 9, the standard deviation is small, indicating that the algorithm has good stability and reliability; The P-value of the Bootstrap test is greater than the significance level, and the null hypothesis cannot be rejected. However, the P-value of the random permutation test is lower than the significance level, and the 95% confidence interval does not include 0 and there is a certain distance from 0, indicating that the deviation between the algorithm results and the expected value is not caused by accidental factors. The reason analysis is as follows: there are too many decision variables in the model, and the complexity of the constructed QUBO matrix is high. The relaxation factor for linear inequality constraint transformation occupies a higher proportion of decision variables, leading to further increase in algorithm complexity and difficulty in finding the optimal solution[10].

IV. CONCLUSIONS

The background and creative source of this study are the dramatic increase in computing scale and the birth of China's first quantum computing platform. This article explores the application of quantum computing in cloud edge end computing optimization based on the QUBO model, aiming to solve the complexity problem of resource allocation that traditional computing methods are unable to cope with. This article overcomes several major difficulties in constructing the QUBO model, including using binary variables to construct the objective function and equivalent transformation of constraint conditions. At the same time, it proposes a way to calculate the relaxation factor, which reduces the computational complexity from O(n) to O(log2n); Finally, the optimization of server computing requirements coverage and cloud edge network layout was successfully achieved. Through multiple experiments and statistical analysis of the results, as well as non parametric tests, it was found that the standard deviation is about 2% of the mean value. At the same time, the P-value of the test results from multiple experiments is greater than the significance level, which verifies the efficiency, practicality, and stability of quantum computing technology and algorithms in dealing with combinatorial optimization problems. In terms of programming and computation, this study utilized China's self-developed quantum computing library Kaiwu SDK instead of Pyqubo in Python, which solved the problem more efficiently.

REFERENCES

[1] H. Kuwahara, Y. -F. Hsu, K. Matsuda and M. Matsuoka, "Dynamic Power Consumption Prediction and Optimization of Data Center by Using Deep Learning and Computational Fluid Dynamics," 2018 IEEE 7th International Conference on Cloud Networking (CloudNet), Tokyo, Japan, 2018, pp. 1-7.

[2] Guo, Mengjie, et al. "Optimization method of distributed resource deaggregation for virtual power plant based on optical quantum computer." Chinese Journal of Electrical Engineering. 2024, pp.1-12.

[3] T. Koch, D. Rehfeldt and Y. Shinano, "APDCM 2022 Keynote Talk: Solving QUBOs on Digital and Quantum Computers," 2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Lyon, France, 2022.

[4] Mahasinghe, A., Fernando, V. and Samarawickrama, P. (2021). 'QUBO formulations of three NP problems', Journal of Information and Optimization Sciences, 42(7), pp. 1625–1648.

[5] S. Li, "A Permutation Test for High-Dimensional Covariance Matrix," Journal of Physics: Conference Series, vol. 1865, (4), 2021.

[6] F. Glover et al, "Quantum bridge analytics I: a tutorial on formulating and using QUBO models," Annals of Operations

Research, vol. 314, (1), pp. 141-183, 2022.

[7] T. Zheng, J. Chen, Z. Zhang, Z. Gong and Y. Chen, "Bank Credit Score Card Selection and Threshold Determination Based on Quantum Annealing Algorithm and Genetic Algorithm," 2023 IEEE 5th International Conference on Power, Intelligent Computing and Systems (ICPICS), Shenyang, China, 2023, pp. 588-594.

[8] W. Chen, M. Zhou and L. Yue, "Research on the optimization of credit score card for solving the QUBO model based on annealing algorithm and quantum computer," 2023 International Conference on Computers, Information Processing and Advanced Education (CIPAE), Ottawa, ON, Canada, 2023, pp. 617-621.

[9] M. Zaman, K. Tanahashi and S. Tanaka, "PyQUBO: Python Library for Mapping Combinatorial Optimization Problems to QUBO Form," in IEEE Transactions on Computers, vol. 71, no. 4, pp. 838-850, April 2022

[10] He Bingze, Ji Xingyi, Lv Jiaxin. Optimization of credit scorecard combinations based on quantum annealing algorithm[J]. Highlights in Science, Engineering and Technology, 2023, 61: 57‑62.