Computing and Informatics, Vol. 41, 2022, 1465–1490, doi: 10.31577/cai_2022_6_1465

TEMPERATURE MATRIX-BASED DATA PLACEMENT OPTIMIZATION IN EDGE COMPUTING ENVIRONMENT

Pengwei Wang

School of Computer Science and Technology
Donghua University, Shanghai, China
&
Engineering Research Center of Digitalized Textile and Fashion Technology
Ministry of Education, Shanghai, China
e-mail: `wangpengwei@dhu.edu.cn`

Yuying Zhao, Hengdi Huang, Zhaohui Zhang

School of Computer Science and Technology
Donghua University, Shanghai, China

Abstract. The scale of data shows an explosive growth trend, with wide use of cloud storage. However, there are challenges such as network latency and energy consumption. The emergence of edge computing brings data close to the edge of the network, making it a good supplement to cloud computing. The spatiotemporal characteristics of data have been largely ignored in studies of data placement and storage optimization. To this end, a temperature matrix-based data placement method using an improved Hungarian algorithm (TEMPLIH) is proposed in this work. A temperature matrix is used to reflect the influence of data characteristics on its placement. A data replica matrix selection algorithm based on temperature matrix (RSA-TM) is proposed to meet latency requirements. Then, an improved Hungarian algorithm based on replica matrix (IHA-RM) is proposed, which satisfies the balance among the multiple goals of latency, cost, and load balancing. Compared with other data placement strategies, experiments show that the proposed method can effectively reduce the cost of data placement while meeting user access latency requirements and maintaining a reasonable load balance between edge servers. Further improvement is discussed and the idea of regional value is proposed.

Keywords: Data placement, edge-cloud, data temperature, Hungarian algorithm, regional value

Mathematics Subject Classification 2010: 68-M02

1 INTRODUCTION

Cloud computing has been a hot topic and development trend in recent years. Typically, data is uploaded to a cloud data center, whose powerful storage and computing capabilities have met traditional business needs. However, with the development of artificial intelligence and 5G technologies, powerful applications continue to appear and the amount of data increases dramatically, which brings high requirements for network latency. Therefore, edge computing is in great demand because it places computing at or near the physical location of data source, enabling faster and more reliable service.

From the perspective of application providers, centralized cloud computing adapts with difficulty to accommodate frequent data interaction. It has become increasingly powerless in terms of network latency, broadband load, and data management costs. Although there have been a large amount of studies regarding to data placement and storage, the researchers have focused on improving the optimization algorithm itself in terms of cost, availability and other QoS, and have largely ignored the spatial and temporal characteristics of data. Goodchild [1] proposed that both human and geographic elements have the inherent characteristics of space, time, and attributes. For example, people in different regions have preferences for different types of videos, and the video data can be reasonably stored according to the users' access habits. Most data in the real world have temporal and spatial attributes, and some data properties also have spatiotemporal relevance and variability. These characteristics have a significant impact on data placement in cloud and edge computing environments, but the existing studies lack consideration in this regard.

Therefore, in this study, we consider the temporal and spatial characteristics to model and calculate the data, and combine the temperature matrix to select the data replica that meets the latency. Then, an improved Hungarian algorithm based on the cost matrix is used to reduce the cost of data placement while ensuring reasonable load balancing.

This work makes four main contributions, which are as follows.

- The concept of data temperature and its calculation model is proposed. Then, a data temperature matrix is constructed and used to optimize data placement;
- A data replica matrix selection algorithm based on temperature matrix (RSA-TM) is proposed, which can generate a replica placement solution that meets latency requirements;

- An improved Hungarian algorithm based on data replica matrix (IHA-RM) is proposed, which aims to optimize the cost and load balance of data placement, while satisfying user latency needs;
- Data temperature is discussed and analyzed, and a new idea of regional value based on temperature is proposed, which can further improve the optimization effect.

The rest of the paper is organized as follows. Section 2 gives a review on related work. Section 3 presents the system model and problem formulation. The proposed method is introduced in detail in Section 4, and Section 5 provides experimental evaluation. A further discussion and improvement based on a novel idea "regional value" is provided in Section 6. Finally, Section 7 concludes the paper.

2 RELATED WORK

With the rapid increase in the amount of data and the number of users, the placement of data not only needs to meet the high-quality service requirements of users, but also consider the resource constraints in the real world. Existing researches about data placement and storage focus on cost optimization, latency optimization, and load balancing in the cloud computing environment.

Cloud computing has an on-demand usage model. Service providers hope to reduce operating costs while satisfying user demands. According to the environments required for cloud storage, cost optimization can be studied for data placement in single- and multi-cloud environments. Yuan et al. [2] proposed a cost-aware strategy considering the trade-off between computing and storage costs according to an intermediate data storage strategy. Single-cloud storage has risks such as low data availability, vendor lock-in, and data privacy. Multi-cloud storage has become a new trend. Yuan et al. [3] proposed an algorithm to find the best balance between bandwidth, storage, and computing costs in a multi-cloud environment, and to compute the minimum cost to store and regenerate a dataset. Wang et al. [6] proposed an ant colony algorithm-based method to store data for users. Cost and data availability are considered in their model. Wang et al. [4] defined a multi-objective optimization problem for multi-cloud storage, which aims to optimize the cost and data availability. NSGA-II algorithm was applied to generate a set of non-dominated solutions, and an entropy-based method was proposed to help users select an appropriate solution. Wang et al. [5] proposed an adaptive architecture for multi-cloud data placement, in order to solve the challenge of dynamically store users' data according to time-varying access patterns. The architecture includes an LSTM-based data retrieval frequency prediction module and a Q-learning-based data placement optimization module. Chen et al. [7] proposed a closed-loop method to optimize service quality and cost. An algorithm was proposed to help make choices to reduce costs and maintain a high quality of service, help select cloud data centers in a multi-cloud environment. Wang et al. [8] proposed an immune-based PSO algorithm to schedule workflow in clouds, which aims to optimize the execution cost and make-span under the constraint of user-defined deadline.

To improve competitiveness, latency has become an important optimization goal when researching placement strategies. Many users consider the latency of user access, as either a constraint or an optimization goal. Wang et al. [9] considered the problem of multi-cloud data placement for spatial crowdsourcing scenario. The geographical characteristics of cloud data centers were analyzed by using a density clustering algorithm. The authors proposed a data placement initialization strategy based on the clustering results, and then an improved genetic algorithm is used to further optimize the placement scheme. Rao et al. [10] tried to minimize the total cost while ensuring the QoS for end users. They considered the location and time diversity of price in multiple markets. Luo et al. [11] used a constrained linear programming method to obtain the best results, ensuring user latency while minimizing energy costs. Yao et al. [12] presented a two-time scale decision strategy. They used fast and slow time scales as control decisions, which is used to guarantee the cost and latency.

Load balance is another important factor, which can greatly affect the performance of a system [14]. A severely unbalanced load will waste resources, affect the processing speed, and increase the response time. Pujol et al. [15] proposed a method to locate user data while guaranteeing load balance, so as to maintain a better online social environment. Shao et al. [13] considered transmission latency and formulated a nonlinear programming problem with coupling constraints so as to ensure the best load balancing and energy consumption of a data center while meeting consumer service level agreements. Tran and Zhang [16] presented a method to place data based on an evolutionary algorithm, which aims to optimize the load balance of servers. Chen et al. [17] explored the relationships between users in social networks, and proposed a method to balance the workload among servers.

Cloud computing supports data storage with the computing power of data centers. The emergence of edge computing can provide real-time and low-latency services for users. A large amount of studies on content placement for edge computing have been conducted. Cao et al. [18] proposed a framework combined NSGA-II algorithm with multi-group strategy, in order to help users select appropriate cloud and edge services to place their data. Wang et al. [19] proposed a method to cache data and schedule tasks jointly for mobile edge computing. Li et al. [20] considered the QoE-driven mobile edge buffer placement problem, considering the different rate-distortion (RD) characteristics of video. Optimal buffer placement was performed through the representation of multiple videos. Xu et al. [21] studied the problem of service caching in MEC's cellular network, and proposed an online algorithm to optimize computational latency under the constraint of long-term energy consumption in edge computing environment. Chae et al. [22] determined the trade-off between gains in content diversity and cooperation based on content placement, and proposed probabilistic content placement to optimize it. Wang et al. [23] studied the placement of application entities in the edge computing environment for social virtual reality applications. A fast iterative algorithm was proposed, constructing a graph in each iteration to encode all costs, and converting cost optimization to a graph cutting problem. Golrezaei et al. [24] proposed a video transmission network architecture, expressing the distributed caching problem as the maximization of sub-module functions. A method to deal with the backhaul problem was proposed, replacing it with a small base station with a low-bandwidth link and high storage capacity. Nikolaou et al. [25] studied the problem of cache placement on collaborative cache based on a single client cache in online social network. Jia and Wang [32] studied the problem of data placement and service deployment in cloud-edge environment, in order to minimize response latency. Xia et al. [33] investigated the collaborative data caching problem in edge computing environment with the aim to minimize the system cost.

While there has been much research on data storage in cloud computing environment, and much discussion of the multi-cloud environment, there is a lack of studies on data placement for edge computing environment. Most such research has addressed the optimization of algorithms and different goals, without considering the temporal and spatial characteristics of the data. This paper studies data placement based on a temperature matrix and considering wide-area distribution in the marginal environment. We propose temperature matrix-based data placement using an improved Hungarian algorithm (TEMPLIH), combining temperature, replica, and cost matrices. While ensuring user latency, we can reduce storage costs as much as possible while balancing loads.

3 SYSTEM MODEL AND PROBLEM FORMULATION

3.1 System Framework

Three tasks need to be solved in this framework:

1. modeling and calculation of data temperature;
2. determination of data replica placement strategy;
3. reasonable data placement based on regional servers and corresponding data,

so as to reduce the cost while satisfying conditions of latency and load balance.

We define a dataset $D = \{d_1, d_2, d_3, \dots, d_M\}$ to represent user's requests for data. The user area, $R = \{r_1, r_2, r_3, \dots, r_N\}$, is the access area related to the user set, which can be used for latency calculation. The set of edge servers, $S = \{s_1, s_2, s_3, \dots, s_K\}$, includes a number of edge servers in each area, which provide storage service for data blocks. Each edge server is associated with a set of attributes $\langle P_e^s, P_e^b, P_e^o, l_e \rangle$, where P_e^s is the storage price, P_e^b is the bandwidth price, P_e^o is the Get operation price, and l_e is the storage capacity.

The relationships among edge server, user area and data are shown in Figure 1. The access latency of data blocks D can be calculated when it is placed in different

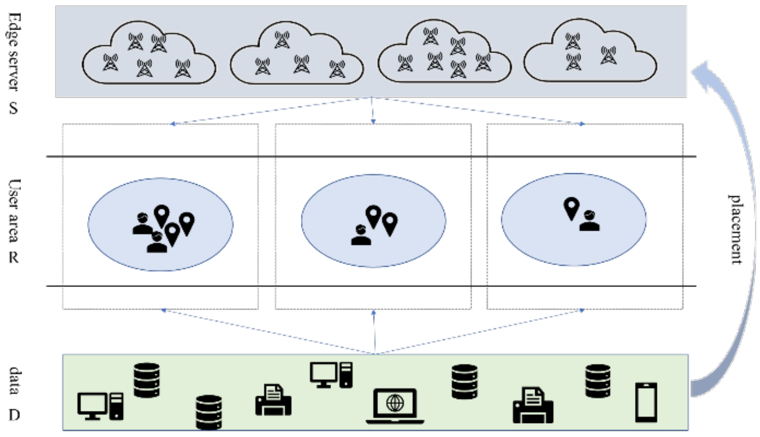


Figure 1. Framework of data placement in edge environment

areas R. Different numbers of edge servers S exist in user access area based on area division. We assume that the latency among the servers in the same area is negligible.

3.2 Data Temperature and Calculation

Data has its own attributes, according to the degree of access to it in different regions, the temperature of data access in different regions is different [9]. We propose the concept of data temperature based on the spatial and temporal characteristics of data. This degree of preference must consider the changes in data attributes and spatial characteristics during a certain period of time. Spatiotemporal data refers to geographic entities whose spatial elements or attributes change over time. Spatial information includes concepts, structures, and spatial relationships with other nearby objects. Temporal information includes concepts, structures, dimensions, and density. For example, let the user's video file collection be $D = \{d_1, d_2, d_3, \ldots, d_m\}$. Indicators from the video data include playback indicators, such as the number of plays and playback quality, and interactive indicators, such as sharing, comments, likes, and favorites. On this basis, we define that each data block contains a set of attributes $\langle d_c, d_t, d_d, d_f \rangle$, where d_c is the number of clicks, d_t is the number of comments, d_d is the number of downloads of the video, and d_f is a user-favorited video. The importance x_i of each data block d_i is evaluated and calculated by the number of clicks and views, comments, downloads, and favorites, which is defined as follows.

$$x_i = 0.8(d_c + d_t + d_d) + 0.2d_f. \tag{1}$$

The relative weight w_i of data block d_i is determined by the ratio of the importance of x_i to that of all other data,

$$w_i = \frac{x_i}{\sum_i^{m-1} x_i}. \tag{2}$$

According to the relative importance of the data and the change characteristics of the data temperature, H is the temperature value of the current data, w is the relative importance, H_0 is the initial temperature, and k is the attenuation coefficient. According to the change characteristics of data temperature

$$H(t) = w * H_0 * e^{-kt}.$$

The data temperature at a certain moment is reasonably quantified by its geographical location and the number of visits, focusing on the initial visit value of the data, relative importance of visits, and number of visits by regional users. Based on the data temperature calculation model, a data temperature matrix T_{mn} can be defined to store the temperature values of data in different regions, i.e., the temperature value $h_{m,n}$ of data m in area n.

$$T_{mn} = \begin{pmatrix} h_{1,1} & \dots & h_{1,n} \\ \vdots & \ddots & \vdots \\ h_{m,1} & \dots & h_{m,n} \end{pmatrix}. \tag{3}$$

For the convenience, we define a regional server matrix to record edge servers by region.

$$R_{nk} = \begin{cases} 1, & \text{Server } k \text{ is in area } n, \\ 0, & \text{Server } k \text{ is not in area } n. \end{cases} \tag{4}$$

3.3 Network Latency

Only by meeting user latency requirements and ensuring user satisfaction and usage can more users be attracted. By taking user access latency as a direct optimization goal or constraint, the goal of satisfying user latency requirements can be achieved. We take time latency as a constraint to ensure that users can access the data they want within an acceptable time. We guarantee that the maximum response time of each request is 200 ms [27] because more than this will seriously affect the user experience.

Define the set $S(t)$ as a collection of data access server at time t. The data access latency is the maximum value of data center in $S(t)$. We use geographic distance as a rough measure of network latency, which we express as a linear function of distance. The correlation between latency and geographic distance can be obtained through network latency data collection, and the round-trip time (RTT) [28, 29] is used to calculate the data access latency.

$$l_m = \max_{d \in S(t)} \{5 + 0.02D(d)\} \tag{5}$$

where $D(d)$ is the distance between the user and the data center, $\hat{D}$ is the maximum acceptable time latency, and the average access latency is

$$\overline{a}\left(\sum_{i=1}^{m} l_i\right) \leq \hat{D}. \tag{6}$$

3.4 System Cost

The goals of cost and average latency may conflict with each other. Placing more copies of data blocks on edge servers may reduce the average latency, but it will increase the cost. We consider the three main parts of resource usage costs, i.e., the costs of data calculation, bandwidth, and storage.

1. Storage Cost

   At time t, the storage cost of data d_i is defined as follows.

   $$P_s = \sum_{e \in S(t)} z_i P_e^s. \tag{7}$$

   The storage cost of data d_i is the sum of the costs of replicas stored on different edge servers. z_i represents the size of data d_i, and P_e^s refers to the storage price of edge server.

2. Network Cost

   Due to the peaks and valleys of user access to video data, we use traffic accounting. There is no restriction on bandwidth, but a fee is charged for passing traffic. The network cost at time t is

   $$P_n = \sum_{e \in S(t)} z_i P_e^b. \tag{8}$$

   P_e^b is the out-bandwidth network price of edge server where data d_i is stored and accessed.

3. Operation Cost

   The cost of the get operation is

   $$P_g = \sum_{e \in S(t)} d_c P_e^o. \tag{9}$$

   P_e^o refers to the price for GET operations in related edge server, and d_c represents the number of operations.

4. Total Cost

The total cost of a data placement scheme is the sum of the above three parts.

$$\hat{P}_C = P_s + P_n + P_g. \tag{10}$$

3.5 Load Balancing

The load of a data placement scheme can be defined as follows.

$$L = \sqrt{\frac{1}{K}\sum_{m=1}^{M}(U_m - U_K)^2} \tag{11}$$

where K is the total number of servers, M is the number of servers where data is placed, U_m is the server utilization, and U_K is the total server utilization. The smaller the value of L, the more balanced the load.

3.6 Problem Formulation

In this work, we aims to find optimal data placement scheme in edge computing environment, so as to minimize the system cost and balance the load among edge servers. Therefore, the optimization problem can be formulated as follows.

$$\min C = \sum_{m=1,n=1,k=1}^{M,N,K} E_{mnk}\hat{P}_C, \tag{12}$$

$$\min L = \sqrt{\frac{1}{k}\sum_{m=1}^{M}(U_m - U_K)^2}, \tag{13}$$

$$\overline{a}\left(\sum_{i=1}^{M} l_i\right) \leq \hat{D}, \tag{14}$$

$$\sum_{m=1,n=1}^{M,N} E_{mnk} z_m < l_e. \tag{15}$$

E_{mnk} is a binary variable, which represents whether data block d_m is placed on edge server s_k in user area r_n. Formulas (14) and (15) respectively require that the data placement scheme must satisfy the user's latency constraint and the server capacity limit. $\hat{D}$ refers to the maximum acceptable time latency. z_m is the size of data d_m, and l_e is the storage capacity of edge server s_k.

4 THE PROPOSED METHOD

The proposed TEMPLIH method consists of a data replica selection algorithm based on temperature matrix (RSA-TM), and an improved Hungarian algorithm based on replica matrix (IHA-RM).

According to the proposed data temperature calculation model, RSA-TM considers the characteristics of data and obtains a data temperature matrix, which can screen suitable data and reduce unnecessary resource consumption. According to the temperature matrix, the areas with high data temperature can be selected to calculate the data access latency for data placement. When the latency constraint is met, placement is stopped and the data placement area is recorded. Otherwise, we select areas to place data in descending order according to the data temperature matrix, and stop the process until the latency requirement is met. The time complexity in calculating the temperature matrix is $O(MN)$.

We then define a data replica matrix based on the temperature matrix. The placement area where data m satisfies the latency in area n is recorded as 1, and otherwise it is 0.

$$L_{mn} = \begin{cases} 1, & \text{Data } m \text{ is placed in area } n, \\ 0, & \text{Data } m \text{ is not placed in area } n. \end{cases} \tag{16}$$

In order to obtain a data placement scheme at the least cost while ensuring load balance, we propose an improved Hungarian algorithm based on the replica matrix (IHA-RM). According to the replica matrix L_{mn} obtained by RSA-TM and the previously defined regional server matrix R_{nk}, the relationship between the data and the regional server can be obtained. The data server placement matrix D_{mk} expresses the placement relationship between data m and area server k,

$$D_{mk} = \begin{cases} 1, & \text{Data } m \text{ is placed in server } k, \\ 0, & \text{Data } m \text{ is not placed in server } k. \end{cases} \tag{17}$$

We combine the regional server matrix R_{nk} and data server placement matrix D_{mk} to get the placement cost of the data on the server in each region according to the cost calculation formula. The cost matrix $P_N = [p_1, p_2, p_3, \dots, p_N]$ represents the placement cost of the data block on the server in each region, i.e., the data placement cost matrix P of data block m and server k under the N areas is collected, the cost of the server storage data block is recorded as $C_{k,m}$, and

$$P_n = \begin{pmatrix} C_{1,1} & \dots & C_{1,m} \\ \vdots & \ddots & \vdots \\ C_{k,1} & \dots & C_{k,M} \end{pmatrix}. \tag{18}$$

Algorithm 1 Data replica matrix selection algorithm based on temperature matrix (RSA-TM)

Input: Data D; User area R; Server S; The capacity of data block stored by the server Cap_m

Output: Data replica matrix L_{mn}

```
1: $L_{mn} \leftarrow \phi$
2: for all data block $d_m \in D$ do
3:     for all area $r_n \in R$ do
4:         $T_{mn} \leftarrow$ Calculate the temperature matrix by Formula (3)
5:         $M \leftarrow$ Regional temperature is sorted by $T_{mn}$
6:     end for
7:     for all area $r_n \in M$ do
8:         $\overline{ave} \leftarrow$ Calculate the latency by Formula (6)
9:         if $\overline{ave} \leq 200$ and $Cap_m < l_e$ then
10:            $R_{index} \leftarrow$ Storage area location
11:        else
12:            $index \leftarrow$ Select the area according to temperature
13:        end if
14:    end for
15: end for
16: Return $L_{mn}$
```

The standard Hungarian algorithm is a one-to-one allocation of computing resources and data, i.e., the number of tasks and computing resources must be equal. In real-world scenario, the numbers of data and servers in each area are often not equal. Therefore, we compare the numbers of data blocks and computing resources in each area. If these are equal, the standard Hungarian algorithm can be used to solve the problem. If they are unequal, we must determine the numbers of servers and data blocks. If the number of servers exceeds the number of data blocks, we add the number of virtual data blocks (add 0) to make it as many dimensions as the number of servers, and then use the standard Hungarian algorithm. If there are more data blocks than servers, according to the dimension of the number of servers, the cost matrix is divided into a small matrix of the number of data blocks divided by the number of servers. If the number of data blocks in the last sub-matrix is less than the number of servers, add the number of virtual data blocks (add 0) to make it consistent with the number of servers. After completing the matrix, we use the traditional Hungarian algorithm to determine the data placement plan.

The time complexity of calculating the data server matrix and cost matrix is $O(NMK)$. The improved Hungarian algorithm (IHA-RM) is used to maintain a low-cost data placement solution under reasonable load balancing conditions between edge servers.

Algorithm 2 Improved Hungarian algorithm based on replica matrix (IHA-RM)

```
Input: Data replica matrix $L_{mn}$; Regional server matrix $R_{nk}$; Number of regional
    servers $R_{num_k}$; Number of area data blocks $D_{num_k}$
Output: Data placement scheme $E_{mnk}$, load $L$, cost $C$
 1: $E_{mnk} \leftarrow \phi$, $L \leftarrow$ inf, $C \leftarrow$ inf, $P_n \leftarrow \phi$
 2: for all area $r_n \in R$ do
 3:   $D_{mk} \leftarrow R_{nk}$ Server dimension and $L_{mn}$ data dimension
 4:   $P_n \leftarrow$ Calculate the cost matrix by Formula (11)
 5:   for all cost matrix $p_n \in P_N$ do
 6:     if $D_{num_k} = R_{num_k}$ then
 7:       $S_{index} \leftarrow$ Use the Hungarian algorithm to select the lowest cost and record
          the placement location
 8:     else if $D_{num_k} < R_{num_k}$ then
 9:       $p_n \leftarrow$ add virtual data block 0 in $R_{num_k}$ dimension
10:       $S_{index} \leftarrow$ Use the Hungarian algorithm to select the lowest cost and record
          the placement location
11:     else if $D_{num_k} > R_{num_k}$ then
12:       $r = D_{num_k}/R_{num_k}$
13:       $R \leftarrow$ Divide $p_n$ into the set of $r$ matrices with $R_{num_k}$ as the dimension
14:       for all matrix $R_i \in R$ do
15:         if $D_{num_k} < R_{num_k}$ then
16:           $R_i \leftarrow$ add virtual data block 0 in $R_{num_k}$ dimension
17:         else
18:           $S_{index} \leftarrow$ Use the Hungarian algorithm to select the lowest cost and
              record the placement location
19:         end if
20:       end for
21:     end if
22:   end for
23: end for
24: $L \leftarrow$ Calculate the load by Formula (12)
25: $C \leftarrow$ Calculate the cost by Formula (11)
26: Return $E_{mnk}, L, C$
```

We define the matrix E_{mnk} to record the data placement plan under the area n, where data m is stored on server k,

$$E_{mnk} = \begin{cases} R_{mk} * 1 = 1,^{m} \text{ is placed in } n, \\ R_{mk} * 1 = 0,^{m} \text{ is not placed in } n. \end{cases} \tag{19}$$

5 EXPERIMENTAL EVALUATION

5.1 Experiment Setup

The dataset is a YouTube popular video dataset with 40 726 items, including the number of views, shares, comments, and likes. Through this dataset, we roughly grasp the degree of user love for videos.

Regional edge server information was obtained from the websites of major edge service providers, including storage price (\$/GB), bandwidth price (\$/GB), get operation price (\$/10k times), and latitude and longitude of the edge server.

The experiment was run on a computer with the Intel Core i7-7500U at 2.7 GHz, with 8 GB memory and Windows 10. The program ran in the Anaconda 3 and Python 3.7 environment.

5.2 Experimental Results and Analysis

We compared the cost and load rate of TEMPLIH with those of several other algorithms for data placement.

Random: The distribution relationship between the data and server is obtained from the replica matrix, and the data block is randomly placed on the regional edge server.

Latency-based [30]: The data are placed on the regional edge server with the lowest total network latency. We calculate the data placement considering cost and load balancing.

Cost-based [10]: According to the replica matrix, we can get the distribution relationship between the data and server. We place the data block on the edge server with the lowest cost.

Load Balance [16]: After the data replica matrix that meets the latency requirement is known, the data blocks are sequentially placed in edge server. Its purpose is to ensure the maximum load balance placement of data.

In the scenario of changing the number of data blocks, the algorithm performance was evaluated by changing the number of data blocks from 6 000 to 13 000. The data block size was fixed at 0.6 GB, the number of servers was 425, and the server capacity was 600 GB. Figures 2 and 3 describe the load rate and cost, respectively, of the data placement schemes obtained by the five algorithms. It can be seen that the load rate of our algorithm is similar to that of the load balance algorithm, but its total average cost is 18.9 % less. As shown in Figure 3, our algorithm sacrifices some cost compared with cost-based method, but has obvious advantages in load balancing.

The data block size was changed to 1.2 GB, with 10 000 fixed data blocks. The number of servers and their capacities were consistent with the above experiment. Figure 4 shows the cost changes of the placement schemes obtained by the five

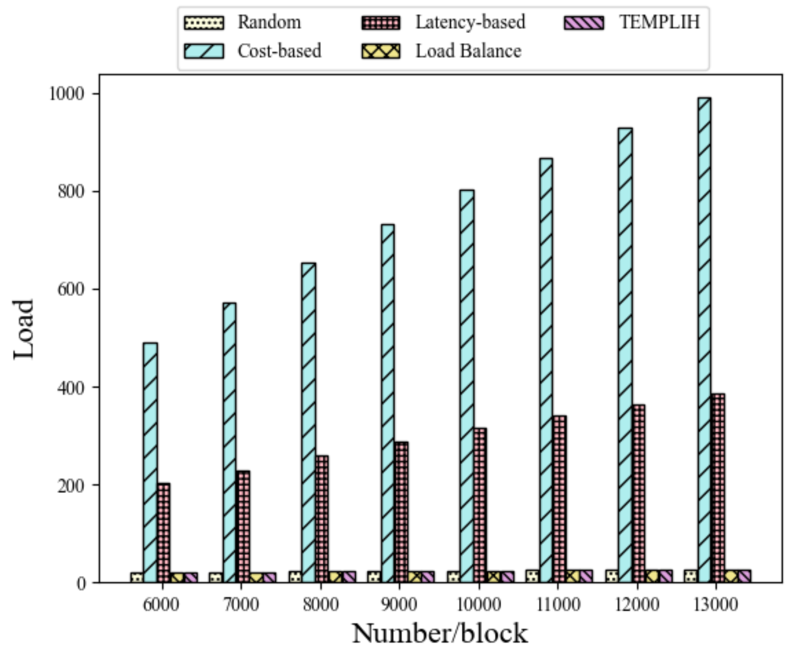


Figure 2. Comparison of load rate with changing of data blocks

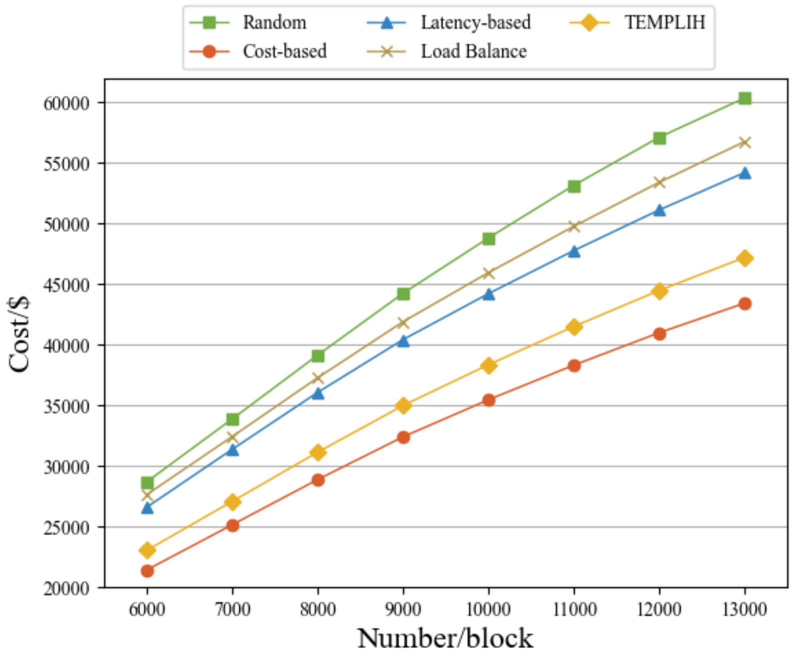


Figure 3. Comparison of cost with changing of data blocks

algorithms. It can be seen that the data block size was too large, the data resources tended to be saturated, and the cost was reduced. Figure 5 compares the load rates of the five algorithms. As the size of the data block increases, the distribution of blocks becomes more dispersed, so the load rate decreases when the number of servers and their capacities are unchanged. The load will be more balanced. Our TEMPLIH algorithm is less effective in cost than the data solution obtained by the cost-based algorithm. However, it can be seen from Figure 5 that the load rate of the cost-based algorithm is 32.8 times that of our proposed algorithm in terms of load conditions. It can be known that the load balancing of the algorithm based on cost is much worse than the TEMPLIH algorithm.

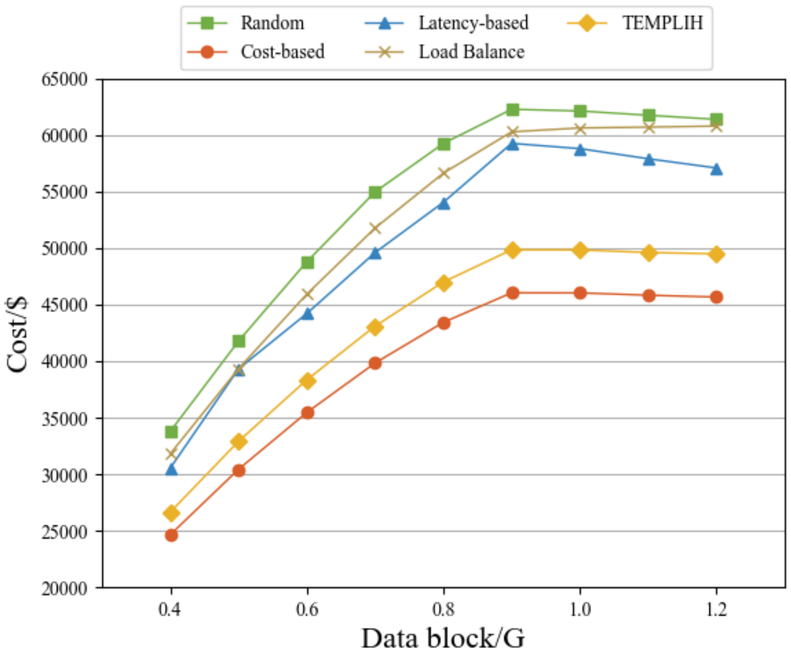


Figure 4. Comparison of cost with changing data block size

Changing the range of server capacity from 400 GB to 650 GB, there were 10 000 data blocks of size 0.6 GB, and the number of servers remained unchanged. Figures 6 and 7 show the changes in load rate and the cost of data placement, respectively, for the five algorithms when the server capacity changed. It can be seen that the load rate increases with the server capacity. This is because the increase in server capacity enables better placement options for data blocks when resources are relatively abundant. With the increase of server capacity, it can be seen from Figure 7 that our proposed TEMPLIH is better than the load balancing algorithm in cost, and our algorithm saves 16.8 % in total average cost.

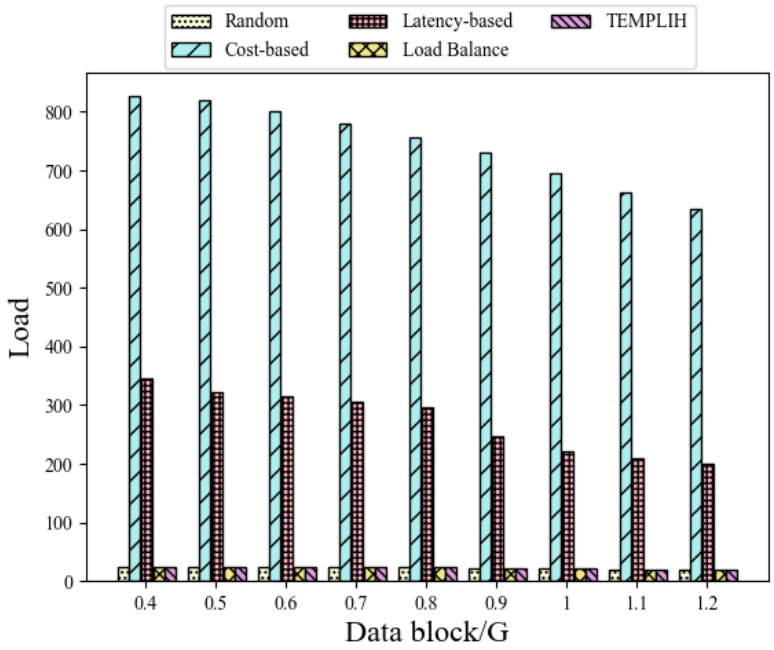


Figure 5. Comparison of load rate with changing data block size

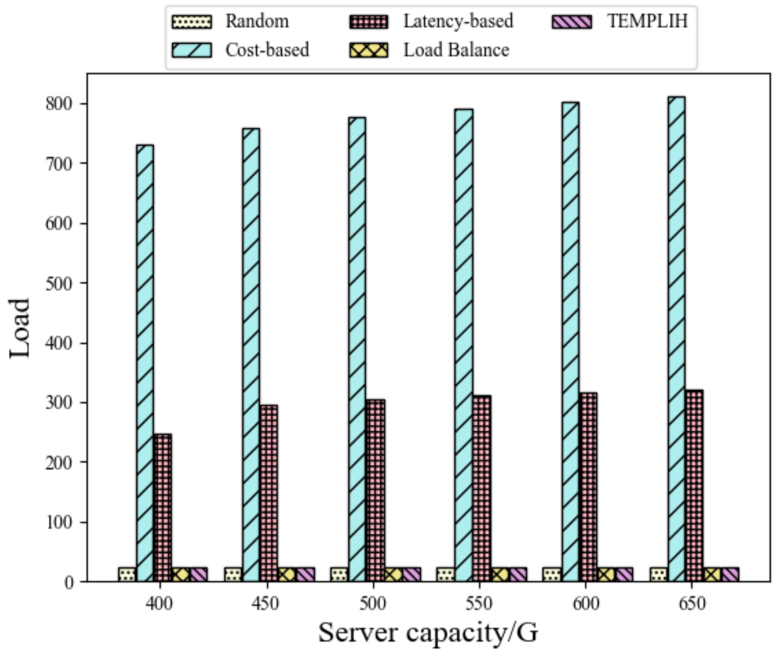


Figure 6. Comparison of load rate with changing server capacity

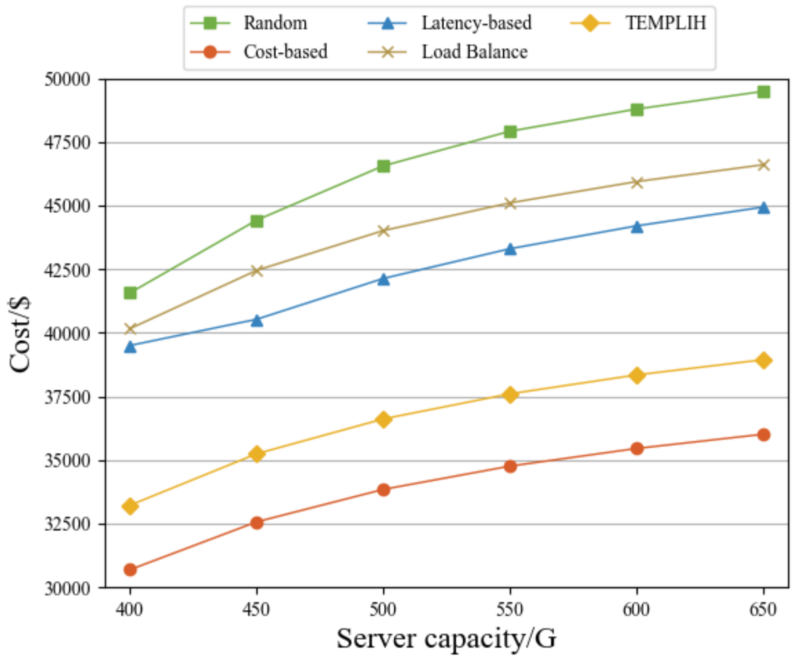


Figure 7. Comparison of cost with changing server capacity

6 DISCUSSION AND IMPROVEMENT

In today's era of big data, cloud-edge computing and AI, a large amount of data increasingly presents many attributes and characteristics related to time and space, and evolve with them, such as the temporal and spatial attributes of data itself, the popularity, access model, value and importance of data. These characteristics have great impact on the scheduling and optimization problem in cloud-edge collaborative environment. The key to solve this is to synchronously connect the characteristics and variability of data in both temporal and spatial dimensions. However, as we have analyzed earlier, this has been largely ignored by the existing methods.

To this end, we present the concept of data temperature to consider such temporal and spatial characteristics jointly, and then propose a temperature matrix-based data placement method using an improved Hungarian algorithm TEMPLIH in this work. However, this is still a "single point" optimization method, since the proposed temperature matrix can only reflect the temperature of a data block in a single area. Whether a data block can be placed and deployed in a certain area, in addition to the attributes of the data block itself and its temperature in this area, we should also consider the influence of neighboring and other relevant areas, so as to obtain a more comprehensive evaluating criterion for data placement.

We take the scenario of online video for example. The demand of hundreds of millions of users around the world for video resources is huge and diverse. The massive video data shows an obvious relevance and variability in two dimensions of space and time. The same video has different popularity and click-to-play times in different regions, and the trend over time is also different. In fact, due to the different nationalities, cultures and preferences of users, there are often significant differences in video requests in different regions. Figure 8 shows an example of video data from YouTube, which has different temperatures in different regions, and distinguished by color. It can be seen that there are obvious regional differences. For example, the data temperature in North Carolina, Virginia and West Virginia is significantly higher than that in surrounding regions.

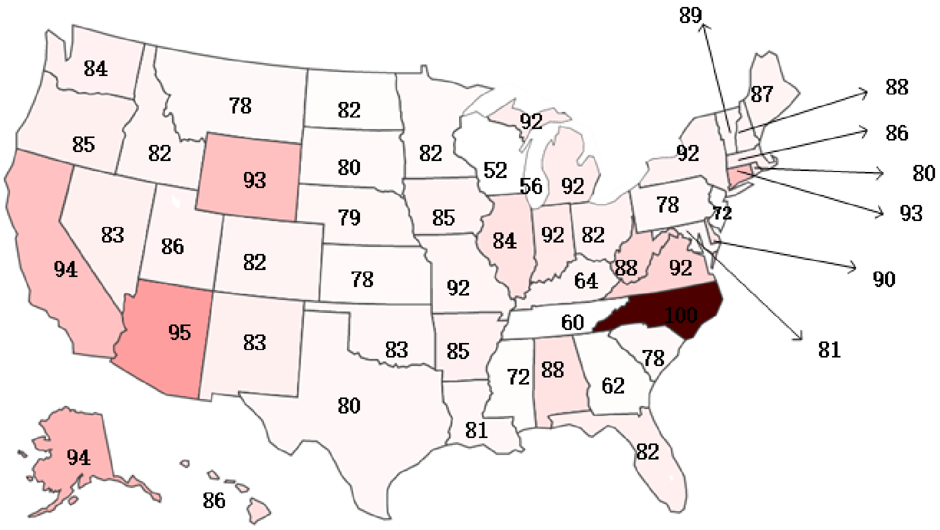


Figure 8. An example of data with different temperatures in different regions

According to the proposed method in this work, compared with Virginia, this video data is more suitable to be placed on edge servers in North Carolina because its temperature value is higher. However, from an overall perspective, placing this video data in Virginia may cover a wider range of user needs, thereby resulting in greater benefits, although the data temperature in this region is lower than that in North Carolina. The placement of a data blocks should not only consider its temperature value in a single region, but also depends on the distribution and adjacency of all regions, as well as the temperature values of this data in relevant regions.

Therefore, considering the correlations among all regions, and also the temperature values of this data in different regions, we can simulate and calculate a value distribution reflecting this aspect (as shown in Figure 9), which can be named as "regional value" of this data in different regions. From the figure, we can find that

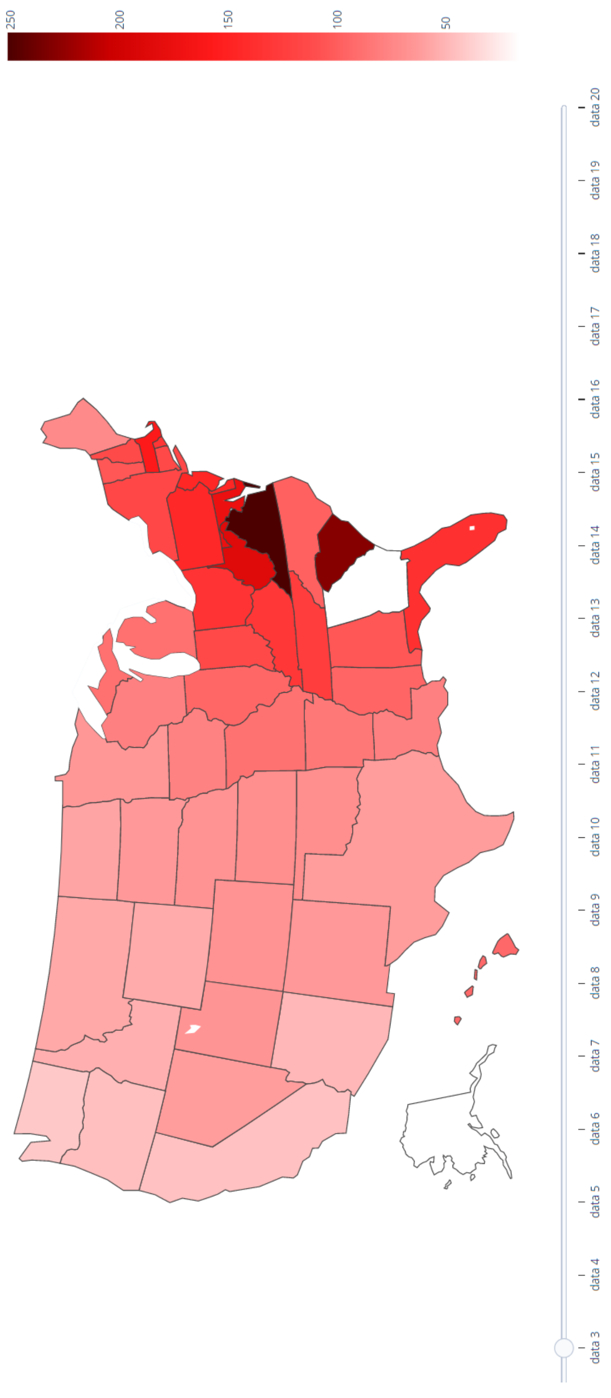


Figure 9. Regional value distribution of the example data considering the correlation between regions

the regional value of this data in Virginia is higher than that in North Carolina, which coincides with the above observation and discussions.

Therefore, it is necessary to further study the regional distribution and correlation characteristics combined with data temperature, which can bring more accurate value orientation to the data placement strategy in edge and cloud computing environment.

Based on the above-mentioned discussions, we present a preliminary study about the idea of regional value in this section. The calculation of regional value of a data block in different regions mainly considers two aspects:

1. the temperature of this data in different regions;
2. the distance between the target region and other relevant regions.

Therefore, the temperature values of this data in relevant areas (including itself) and the distance to the target region together determine the "regional value" of this data block. The higher the temperature value, the greater the contribution to the regional value, and the farther the distance, the smaller the contribution to the regional value.

The idea discussed above coincides with that of PageRank. PageRank's core idea includes quantity hypothesis and quality hypothesis. That is, the more important it is when a web page is linked by more pages, and the more important it is when high-quality pages linking to this page. Inspired by PageRank's idea, we give two principles for regional value.

1. The higher the temperature of the data itself, the greater the value generated, and also the greater the contribution of this region to other regions.
2. The farther the region is from the target region, the smaller the value it will contribute.

Therefore, we can define and calculate the regional value of a data block by a simple model, which is the cumulative sum of the ratio of the temperature value of this data object in a region to the distance between this region and the target region, and add the ratio of temperature value in target region to the average distance among all regions.

In addition, we perform some experiments to verify the above ideas and models. The experimental setup is the same as that in Section 5.1. Figure 10 shows the cost comparison of solutions based on data temperature and regional value when changing data blocks. We can find that the placement scheme resulting from the regional value is lower in cost than that generated only by data temperature.

Figure 11 shows the cost of data placement schemes when the number of data blocks is fixed at 40 000, and the server capacity is set to 50, 100, 150, 200, 250, and 300. It can be seen that the cost obtained by regional value is obviously better than the data placement schemes resulting from data temperature.

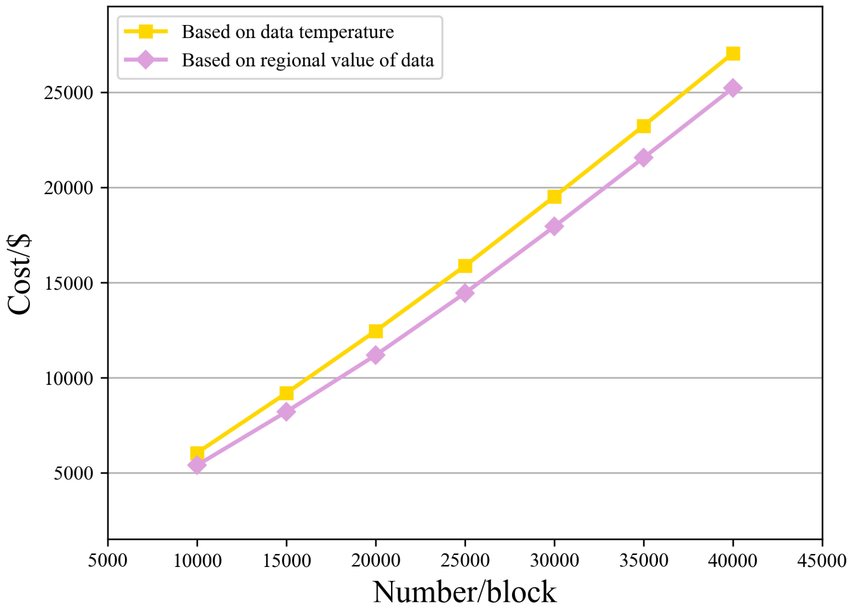


Figure 10. Comparison of cost between two different strategies

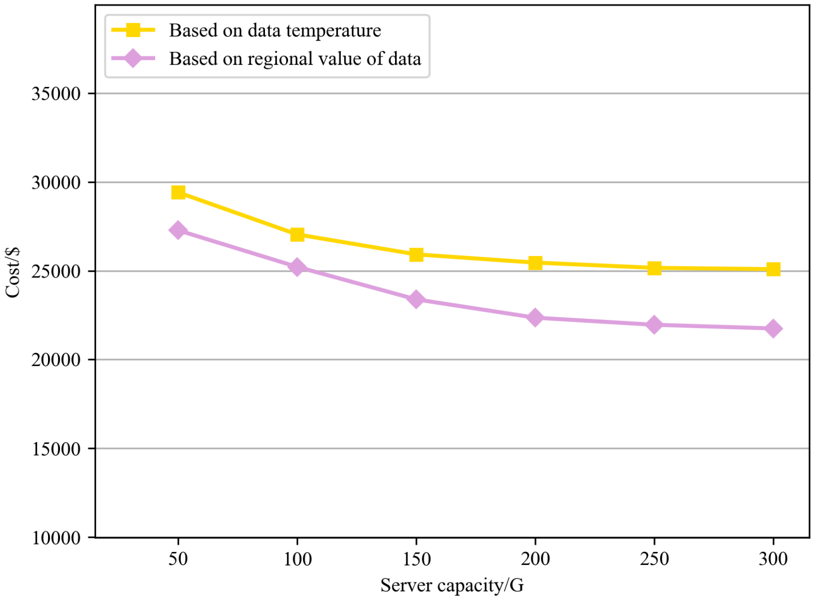


Figure 11. Comparison of load rate between two different strategies with changing server capacity

7 CONCLUSION

The demand for network latency cannot be ignored in the current environment. At present, we cannot any longer meet the needs of users by relying only on the multi-cloud market for data storage. In the edge environment, edge server can take advantage of its own lightweight, real-time computing capabilities and closer proximity to users to place data reasonably, which can effectively improve user experience. However, how to use the characteristics of the data itself and quickly weigh the relationship between various indicators is a problem that remains to be solved in data placement. Our proposed TEMPLIH can optimize the cost and load balance of data in the edge environment under the premise of meeting the latency requirements. Specifically, the RSA-TM and the IHA-RM are adopted. Experiments have proved that in terms of optimization effects, the TEMPLIH strategy that considers the data temperature matrix is better than the traditional multi-cloud data storage strategy. A new idea of regional value based on data temperature is proposed and analyzed, which will be further studied in future work.

Acknowledgement

This work was partially supported by the National Natural Science Foundation of China (NSFC) under Grant 61602109, DHU Distinguished Young Professor Program under Grant LZB2019003, Shanghai Science and Technology Innovation Action Plan under Grants 22511100700, 19511101802, Fundamental Research Funds for the Central Universities. A preliminary conference version [31] of this paper appeared in Proceedings of the 22$^{\text{nd}}$ International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT 2022).

REFERENCES

[1] Goodchild, M. F.: Geographical Data Modeling. Computers and Geosciences. Vol. 18, 1992, No. 4, pp. 401–408, doi: 10.1016/0098-3004(92)90069-4.

[2] Yuan, D.—Yang, Y.—Liu, X.: A Cost-Effective Strategy for Intermediate Data Storage in Scientific Cloud Workflow Systems. 2010 IEEE International Symposium on Parallel and Distributed Processing (IPDPS), 2010, pp. 1–12, doi: 10.1109/IPDPS.2010.5470453.

[3] Yuan, D.—Cui, L.—Li, W.—Liu, X.—Yang, Y.: An Algorithm for Finding the Minimum Cost of Storing and Regenerating Datasets in Multiple Clouds. IEEE Transactions on Cloud Computing, Vol. 6, 2018, No. 2, pp. 519–531, doi: 10.1109/TCC.2015.2491920.

[4] Wang, P.—Zhao, C.—Liu, W.—Chen, Z.—Zhang, Z.: Optimizing Data Placement for Cost Effective and High Available Multi-Cloud Storage. Computing and Informatics, Vol. 39, 2020, No. 1-2, pp. 51–82, doi: 10.31577/cai_2020_1-2_51.

[5] Wang, P.—Zhao, C.—Wei, Y.—Wang, D.—Zhang, Z.: An Adaptive Data Placement Architecture in Multicloud Environments. Scientific Programming, Vol. 2020, 2020, Art. No. 1704258, doi: 10.1155/2020/1704258.

[6] Wang, P.—Zhao, C.—Zhang, Z.: An Ant Colony Algorithm-Based Approach for Cost-Effective Data Hosting with High Availability in Multi-Cloud Environments. 2018 IEEE 15th International Conference on Networking Sensing and Control (ICNSC), 2018, pp. 1–6, doi: 10.1109/ICNSC.2018.8361288.

[7] Chen, W.—Cao, J,—Wan, Y.: QoS-Aware Virtual Machine Scheduling for Video Streaming Services in Multi-Cloud. Tsinghua Science and Technology, Vol. 18, 2013, No. 3, pp. 308–317, doi: 10.1109/TST.2013.6522589.

[8] Wang, P.—Lei, Y.—Agbedanu, P. R.—Zhang, Z.: Makespan-Driven Workflow Scheduling in Clouds Using Immune-Based PSO Algorithm. IEEE Access, Vol. 8, 2020, pp. 29281–29290, doi: 10.1109/ACCESS.2020.2972963.

[9] Wang, P.—Chen, Z.—Zhou, M.—Zhang, Z.—Abusorrah, A.—Ammari, A. C.: Cost-Effective and Latency-Minimized Data Placement Strategy for Spatial Crowdsourcing in Multi-Cloud Environment. IEEE Transactions on Cloud Computing, 2021, pp. 1, doi: 10.1109/tcc.2021.3119862.

[10] Rao, L.—Liu, X.—Xie, L.—Liu, W.: Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment. 2010 Proceedings of IEEE INFOCOM, 2010, pp. 1–9, doi: 10.1109/INFCOM.2010.5461933.

[11] Luo, J.—Rao, L.—Liu, X.: Temporal Load Balancing with Service Delay Guarantees for Data Center Energy Cost Optimization. IEEE Transactions on Parallel and Distributed Systems, Vol. 25, 2014, No. 3, pp. 775–784, doi: 10.1109/TPDS.2013.69.

[12] Yao, Y.—Huang, L.—Sharma, A. B.—Golubchik, L.—Neely, M. J.: Power Cost Reduction in Distributed Data Centers: A Two-Time-Scale Approach for Delay Tolerant Workloads. IEEE Transactions on Parallel and Distributed Systems, Vol. 25, 2014, No. 1, pp. 200–211, doi: 10.1109/TPDS.2012.341.

[13] Shao, H.—Rao, L.—Wang, Z.—Liu, X.—Wang, Z.—Ren, K.: Optimal Load Balancing and Energy Cost Management for Internet Data Centers in Deregulated Electricity Markets. IEEE Transactions on Parallel and Distributed Systems, Vol. 25, 2014, No. 10, pp. 2659–2669, doi: 10.1109/TPDS.2013.227.

[14] Kumar, A.—Kalra, M.: Load Balancing in Cloud Data Center Using Modified Active Monitoring Load Balancer. 2016 International Conference on Advances in Computing, Communication, and Automation (ICACCA), 2016, pp. 1–5, doi: 10.1109/ICACCA.2016.7578903.

[15] Pujol, J. M.—Erramilli, V.—Siganos, G.—Yang, X.—Laoutaris, N.—Chhabra, P.—Rodriguez, P.: The Little Engine(s) That Could: Scaling Online Social Networks. IEEE/ACM Transactions on Networking, Vol. 20, 2012, No. 4, pp. 1162–1175, doi: 10.1109/TNET.2012.2188815.

[16] Tran, D. A.—Zhang, T.: S-PUT: An EA-Based Framework for Socially Aware Data Partitioning. Computer Networks, Vol. 75, 2014, Part B, pp. 504–518, doi: 10.1016/j.comnet.2014.08.026.

[17] Chen, H.—Jin, H.—Wu, S.: Minimizing Inter-Server Communications by Ex-

ploiting Self-Similarity in Online Social Networks. IEEE Transactions on Parallel and Distributed Systems, Vol. 27, 2016, No. 4, pp. 1116–1130, doi: 10.1109/TPDS.2015.2427155.

[18] Cao, E.—Wang, P.—Yan, C.—Jiang, C.: A Cloudedge-Combined Data Placement Strategy Based on User Access Regions. 6th International Conference on Big Data and Information Analytics (BigDIA 2020), 2020, pp. 243–250, doi: 10.1109/BigDIA51454.2020.00046.

[19] Wang, P.—Zhao, Y.—Zhang, Z.: Joint Optimization of Data Caching and Task Scheduling Based on Information Entropy for Mobile Edge Computing. 2021 19th IEEE International Conference on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking (ISPA/BDCloud/SocialCom/SustainCom), 2021, pp. 460–467, doi: 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00070.

[20] Li, C.—Toni, L.—Zou, J.—Xiong, H.—Frossard, P.: QoE-Driven Mobile Edge Caching Placement for Adaptive Video Streaming. IEEE Transactions on Multimedia, Vol. 20, 2018, No. 4, pp. 965–984, doi: 10.1109/TMM.2017.2757761.

[21] Xu, J.—Chen, L.—Zhou, P.: Joint Service Caching and Task Offloading for Mobile Edge Computing in Dense Networks. IEEE INFOCOM 2018 – IEEE Conference on Computer Communications, 2018, pp. 207–215, doi: 10.1109/INFOCOM.2018.8485977.

[22] Chae, S. H.—Quek, T. Q. S.—Choi, W.: Content Placement for Wireless Cooperative Caching Helpers: A Tradeoff Between Cooperative Gain and Content Diversity Gain. IEEE Transactions on Wireless Communications, Vol. 16, 2017, No. 10, pp. 6795–6807, doi: 10.1109/TWC.2017.2731760.

[23] Wang, L.—Jiao, L.—He, T.—Li, J.—Mühlhäuser, M.: Service Entity Placement for Social Virtual Reality Applications in Edge Computing. IEEE INFOCOM 2018 – IEEE Conference on Computer Communications, 2018, pp. 468–476, doi: 10.1109/INFOCOM.2018.8486411.

[24] Golrezaei, N.—Shanmugam, K.—Dimakis, A. G.—Molisch, A. F.—Caire, G.: FemtoCaching: Wireless Video Content Delivery Through Distributed Caching Helpers. 2012 Proceedings IEEE INFOCOM, 2012, pp. 1107–1115, doi: 10.1109/INFCOM.2012.6195469.

[25] Nikolaou, S.—Van Renesse, R.—Schiper, N.: Proactive Cache Placement on Cooperative Client Caches for Online Social Networks. IEEE Transactions on Parallel and Distributed Systems, Vol. 27, 2016, No. 4, pp. 1174–1186, doi: 10.1109/TPDS.2015.2425398.

[26] Wang, P.—Wei, Y.—Zhang, Z.: Optimizing Data Placement in Multi-Cloud Environments Considering Data Temperature. In: Sun, X., Zhang, X., Xia, Z., Bertino, E. (Eds.): Artificial Intelligence and Security (ICAIS 2021). Springer, Cham, Lecture Notes in Computer Science, Vol. 12737, 2021, pp. 167–179, doi: 10.1007/978-3-030-78612-0_14.

[27] Khalajzadeh, H.—Yuan, D.—Grundy, J.—Yang, Y.: Improving Cloud-Based Online Social Network Data Placement and Replication. 2016 IEEE 9th International Conference on Cloud Computing (CLOUD), 2016, pp. 678–685, doi: 10.1109/CLOUD.2016.0095.

[28] Wu, Z.—Butkiewicz, M.—Perkins, D.—Katz-Bassett, E.—Madhyastha, H. V.: SPANStore: Cost-Effective Geo-Replicated Storage Spanning Multiple Cloud Services. Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles (SOSP '13), 2013, pp. 292–308, doi: 10.1145/2517349.2522730.

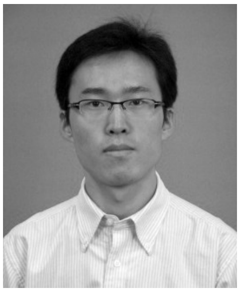
[29] Yu, W.—Wu, C.—Bo, L.—Zhang, L.—Li, Z.—Lau, F. C. M.: Scaling Social Media Applications into Geo-Distributed Clouds. IEEE/ACM Transactions on Networking, Vol. 23, 2015, No. 3, pp. 689–702, doi: 10.1109/TNET.2014.2308254.

[30] Li, X.—Wu, J.—Tang, S.—Lu, S.: Let's Stay Together: Towards Traffic Aware Virtual Machine Placement in Data Centers. IEEE INFOCOM 2014 – IEEE Conference on Computer Communications, 2014, pp. 1842–1850, doi: 10.1109/INFOCOM.2014.6848123.

[31] Zhao, Y.—Wang, P.—Huang, H.—Zhang, Z.: Temperature Matrix-Based Data Placement Using Improved Hungarian Algorithm in Edge Computing Environments. In: Shen, H., Sang, Y., Zhang, Y. et al. (Eds.): Parallel and Distributed Computing, Applications and Technologies (PDCAT 2021). Springer, Cham, Lecture Notes in Computer Science, Vol. 13148, 2021, pp. 237–248, doi: 10.1007/978-3-030-96772-7_22.

[32] Jia, J.—Wang, P.: Low Latency Deployment of Service-Based Data-Intensive Applications in Cloud-Edge Environment. 2022 IEEE International Conference on Web Services (ICWS), 2022, pp. 57–66, doi: 10.1109/ICWS55610.2022.00023.

[33] Xia, X.—Chen, F.—He, Q.—Grundy, J.—Abdelrazek, M.—Jin, H.: Online Collaborative Data Caching in Edge Computing. IEEE Transactions on Parallel and Distributed Systems, Vol. 32, 2021, No. 2, pp. 281–294, doi: 10.1109/TPDS.2020.3016344.



Pengwei Wang received his Ph.D. degree in computer science from the Tongji University, Shanghai, China, in 2013. He finished his Post-Doctoral research work at the Department of Computer Science, University of Pisa, Italy, in 2015. Currently, he is Associate Professor with the School of Computer Science and Technology, Donghua University, Shanghai, China. His research interests include cloud and edge computing, services computing, and big data. He has published more than 80 papers in premier international journals and conferences, including IEEE Transactions on Services Computing, IEEE Transactions on Cloud Computing, IEEE Transactions on Systems, Man, and Cybernetics: Systems, IEEE Transactions on Automation Science and Engineering, IEEE International Conference on Web Services, IEEE International Conference on Services Computing, etc.

Yuying Zhao received her B.Sc. degree in network engineering from the Henan University, Kaifeng, China, in 2019; her M.Sc. degree in the School of Computer Science and Technology from the Donghua University in Shanghai, China, in 2022. Her research interests include cloud computing and edge computing.

Hengdi Huang received her B.Sc. degree in software engineering from the Henan University, Kaifeng, China, in 2021. She is currently Master student studying software engineering at the Donghua University in Shanghai, China. Her research interests include fraud detection and cloud computing.

Zhaohui Zhang received his B.Sc. degree in computer science from the Anhui Normal University, Wuhu, China, in 1994, and became Teacher at the University. He completed his master's courses program of the University of Science and Technology of China from 1999 to 2000. Respectively, he received his Ph.D. degree in computer science from the Tongji University, Shanghai, China, in 2007. He was Professor with the Anhui Normal University before 07/2015. Currently, he is Professor with the School of Computer Science and Technology, Donghua University, Shanghai, China. His research interests include network information services, service computing and cloud computing.