Cluster Computing (2024) 27:457–476
https://doi.org/10.1007/s10586-022-03952-1

File block multi-replica management technology in cloud storage

Qinlu He[1] · Fan Zhang[1] · Genqing Bian[1] · Weiqi Zhang[1] · Zhen Li[2] · Zhimin Yu[1] · Hao Feng[3]

Received: 10 August 2022 / Revised: 1 December 2022 / Accepted: 20 December 2022 / Published online: 10 January 2023

Abstract

With the rapid development of cloud computing technology and the broad application of the Internet, more and more user data are stored in the cloud storage system online. Data copy technology is an effective technical means to manage various data in cloud storage systems, which has the advantages of improving data access speed and enhancing data availability in cloud storage systems. Since a data copy management strategy has an important impact on the performance of cloud storage systems, how to design a data copy management strategy is of great significance to achieving high-quality cloud storage management. Firstly, this paper analyzes the HDFS random copy placement strategy and its shortcomings. Secondly, proposes a copy placement strategy based on evaluation value and load balancing. Thirdly, gives the evaluation function of rack and node by considering multiple evaluation indicators such as node disk space load capacity, node memory utilization, node CPU utilization, rack load rate and network distance between racks. Fourthly, finds suitable nodes through computer shelf evaluation values and node evaluation values when placing replicas. Finally, through simulation experiments, the experimental results of the HDFS random replica placement strategy and the evaluation results and load balancing-based replica placement strategy are compared and analyzed.

Keywords Cloud storage · HDFS · Replica management strategy · Replica placement

1 Introduction

In recent years, the rapid rise of cloud computing technology has changed the traditional data storage mode and is favored by more and more enterprises and individuals. Cloud computing is distributed computing, which connects and integrates storage and computing resources across regions through the Internet to provide users with various required services. Cloud computing includes four essential parts: cloud platform, cloud storage, cloud terminal, and cloud security [1].

Cloud storage has many advantages over traditional storage systems, but it still faces considerable challenges [2–6]. Bata copy management technology is proposed to improve the performance of cloud storage systems. Data copy management technology replicates multiple copies of data blocks and places them on different nodes. When the node to be accessed fails, users can access other copies of the data block to obtain data, thereby improving system availability. The emergence of data copy management technology, a way to effectively manage the data lifecycle and conserve storage resources, has brought new opportunities to cloud storage systems [7–14].

Data copy technology is a data management strategy that improves system performance through data redundancy policies. Because data copy technology has excellent performance, such as reducing data response time and increasing system reliability, it has been widely used in cloud storage systems to solve the problems and optimizations they face. To implement data copy technology in cloud storage systems, solve two key issues: the determination and adjustment of the number of copies and the determination and optimization of the location of the replicas. Because the cost of maintaining a system

✉ Qinlu He
luluhe8848@Hotmail.com

✉ Hao Feng
18992800660@189.cn

[1] School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710054, China

[2] Shaan Xi Institute of Metrology Science, Xi'an 710043, China

[3] Shaan Xi Big DATA Group Co., Ltd, Xi'an 710078, China

increases as the number of replicas increases, it is essential to minimize operational costs while ensuring that system availability requirements are met. In addition, how to make the location layout of replicas can quickly respond to user access needs, and balance the load balancing requirements and other performance indicators of the system, is a critical problem that data replica technology needs to solve. In the existing data copy management solution, a typical system adopts the default number of data copies, such as Google File System, Hadoop Distributed File System, etc. This method of defaulting to the number of data copies is convenient to operate, but it will cause a particular waste of resources. Moreover, most of the existing copy placement strategies are designed according to the needs of data node performance or file corresponding rate. There few data copy layout strategies consider multiple performance metrics as a total optimization function and solve them using heuristics. A heuristic algorithm is obtained by simulating the activities of swarm intelligence in nature, and the obtain optimal solution in a limited time by using it to solve optimization problems [15–18]. Therefore, it is of great practical significance to study suitable data copy technology to solve the issues in data copy technology.

In addition, with the large-scale construction of data centers around the world to meet the demand for storage systems, the energy consumption problem has become increasingly prominent and has received much attention from industry and academia. As early as 2013, the power consumption of data centers in the United States reached 9.1×10^{10}KWh. Annual energy consumption is expected to reach 1.4×10^{11}KWh by 2025 [19]. In other words, the cost of electricity in the data center has far exceeded the price of hardware configuration. According to ENP's analysis of data center energy consumption, server energy consumption accounts for more than half of the total energy consumption.

In addition, when some servers are overloaded for a long time, it is easy to cause rapid equipment aging and increase hardware purchase expenses, while some servers are not overloaded for a long time, which will lead to the reduction of effective energy utilization and increase power expenditure. The overload and no-load of data nodes are partly caused by the unreasonable placement of data copies. Therefore, optimizing the layout of the data copy stored in the data center is essential.

In summary, the data copy management strategy has an important impact on the performance of the cloud storage system, and the design of a suitable data copy management strategy is of great significance to achieve high-quality cloud storage management, and optimizing the layout of data copy to integrate the cloud storage resources of the system and improve the effective utilization of energy in the cloud storage system is also of great practical significance to solve the energy efficiency problem in the cloud storage system.

In this paper, we will focus on data copy placement strategies. This paper first introduces the HDFS distributed file system, then introduces and analyzes the HDFS random replica placement strategy and points out its shortcomings, then studies the way of file partitioning, and then proposes a replica based on evaluation value and load balancing placement strategy.

By considering five evaluation indicators, including node disk space load capacity, node memory utilization, node CPU utilization, rack load rate, and network distance between racks, the appropriate node is selected for placement through the computer shelf evaluation value and node evaluation value when placing replicas. Finally, through the simulation experiment, the experimental results of the evaluation factors, such as rack load balancing, node load balancing time overhead under the HDFS random replica placement strategy, and the replica placement strategy based on evaluation value and load balancing, are compared and analyzed. Experimental results show that compared with the HDFS random copy placement strategy, the copy placement strategy based on the evaluation value and load balancing can better achieve inter-rack load balancing and inter-node load balancing and improve data security, but its time cost is large.

This paper makes the following contributions:

(1) Analyzes the shortcomings of existing HDFS random replica placement strategies.

(2) A replica placement strategy based on evaluation value and load balancing is proposed. When selecting the replica placement location, it is analyzed and evaluated from both the rack and the node.

(3) Through implementation analysis, the replica placement strategy based on evaluation value and load balancing proposed in this paper has more significant advantages in rack load balancing, node load balancing, and data security.

2 Related work

Regarding data replica placement, Armani et al. [20] have studied the problems of increased bandwidth consumption and transmission delay and reduced quality of service and reliability by the consistency overhead between replicas. In the paper, by analyzing the latency and reliability of different data consistency operations to study the copy placement problem with consistency overhead, propose a replication algorithm, namely write-aware copy placement (WARP). WARP consists of two algorithms: leader selection and copy placement. Both algorithms are executed based on the latency of the operation in question. The leader selection algorithm produces a data center with the minimum average latency with which the leader allocates all operations. The replica placement algorithm allocates master and slave replicas based on average latency and the number of requests to improve the performance of different operations. Experimental results show that WARP can shorten the response time, improve the quality of service, and provide reliability comparable to the previous algorithm. Li Pengden et al. [21] proposed a load-balanced replica placement method to solve the problem of selecting a suitable data center for data replication to improve the access performance of cloud storage systems effectively. The strategy determines the new data center node based on all data center loads for the entire system, based on the distance the service request information is forwarded between the two nodes. The parameters for the standby data center are then sent to the data center. The central node calculates the capital value for each data center, selecting the data center with the highest capital value as the new replica placement node. This strategy can adapt to the changes in the system and reasonably select the data center to place the replica, which can not only meet the availability and reliability of the data but also improve the overall stability of the system and ensure the load balance between the data centers. However, this strategy has some drawbacks, as the number of new replica nodes increases, management costs will increase rapidly, and energy consumption issues will become a new challenge [22, 23]. Mansouri et al. [24] have studied the waste of time and resources caused by the current HDFS relying on load-balancing utilities to balance replica distribution. To solve this problem, the authors propose an innovative HDFS replica placement strategy for data-intensive computation on massive data sets of cloud systems. This strategy not only meets all the requirements of the original HDFS copy placement policy but also distributes the replicas evenly to the cluster nodes without allowing balancing utilities. In addition, use this strategy in both homogeneous and heterogeneous environments. It can achieve perfect load balancing based on the difference in processing power of the cluster nodes. Experimental results show that the proposed strategy has better data node utilization than the default replica placement strategy of HDFS. Wang Yan et al. [25] proposed a decentralized copy placement (D-ReP) algorithm that is based on the facility location problem (FLP) and requires only local topology information. The two versions of the algorithm (source and edge) are triggered at equal intervals and iteratively push the replica from the source node (central storage) to the request edge node. Duplicate a set of closely adjacent nodes that require the same data object causes replicas to be created and migrated to those nodes. When demand drops, replicas are discarded or migrated to a different location. The main goal of the D-ReP algorithm is to place replicas between storage nodes (based on cloud storage providers and edge entities) in a way that minimizes the cost function. Experimental results show that D-ReP has significant advantages in terms of latency and cost for non-replicated data sources and client-side caching. Park et al. [26] constructed a model based on a cryptogram to express the data copy placement problem and proposed a data copy placement strategy based on genetic algorithms. In the three-point graph model, three vertices represent copies of data and data nodes. The edge between a task vertex and a data replica vertex is a mapping from task to data replica representing a scheduling scheme. The edge between the vertex of a data copy and the vertex of a data node is a mapping from a copy of the data to a data node representing the location of the copy. The authors use a strategy based on genetic algorithms to search for mappings from tasks to data replicas and data replicas to data nodes. Using genetic algorithms, the best-performing mapping is obtained as the best solution to the problem of data copy placement. This strategy reduces the size, data movement time, and several moves. The copy placement strategy based on genetic algorithms performs better in cloud science applications than the random placement strategy used by HDFS. Huang et al. [19] proposed an approach focused on virtualizing Hadoop to coordinate the management of virtual machines and file copies simultaneously. This method minimizes power consumption, waste of physical resources, and file unavailability by determining virtual machine allocation, template selection, and file copy placement. The non-dominant sorting genetic algorithm-II (NSGA-II) is used to implement this solution, using dual chromosomes and cross-operators to solve multi-objective optimization problems. Experimental results show that the proposed method has a significant effect on file unavailability and resource waste and has little improvement in power consumption. Huang et al. [27] devised a novel data placement strategy based on genetic algorithms to manage data on large social networks and reduce storage costs while meeting access latency requirements. This method can find the best data placement location and minimize inter-server traffic based on each user's location and social graph. Experiments using a real Facebook dataset show that the strategy can significantly reduce data storage costs and inter-server traffic. Unlike other heuristics that require the addition of replicas to reduce inter-server traffic, the proposed algorithm does not need to increase the number of replicas when optimizing inter-server traffic, ensuring optimal storage costs.

In terms of replica placement strategy, by learning the default replica placement strategy and drawing on previous experience, it is concluded that the replica dynamic management strategy needs to consider the practical compromise between load balancing and computing efficiency, so this paper uses multi-factor evaluation of node performance to optimize the replica placement strategy with computing efficiency as the primary and inter-node load as the supplement. The relative importance of node evaluation factors is described by the weight of each factor, the weight of the node determinant is determined by analytic hierarchy, and the sum of the importance of each determinant is 1. By setting the primary and secondary relationship between each factor and calculating the impotrance of each element according to probability theory, this method can avoid the influence of subjective human factors and objectively assign the important of each judgment factor at the node. In this paper, we select five factors to evaluate node performance and calculate the node evaluation value by linearly weighting the three load factors under the condition of ensuring node availability to represent node performance and select nodes with high performance for replica placement.

3 Replica placement strategy based on evaluation values and load balancing

The default copy placement strategy of HDFS meets the reliability and fish-load balance of the system to a certain extent. Still, it cannot solve the read-and-write efficiency and storage space utilization after the change of file access heat. The longer the file system runs, the heat of system Chinese shows different levels, and the high-popularity files cannot meet the sudden increase in user access due to the limitation of the number of copies, resulting in a decrease in system processing efficiency and affecting the user experience. Low-heat files are accessed by users below average, and the default 3-copy policy will undoubtedly cause a waste of system storage space.

To solve the above problems existing in the default copy placement strategy, this paper designs a dynamic copy placement strategy that increases the copy factor of hot files and selects nodes with high node evaluation coefficients as target nodes when reducing the copy factor of low-heat files, comprehensive file heat decision and availability decision two methods determine the number of reductions. They should also follow the system reliability, ensure that each data block minimum two copies, and distributed in different racks, under the above rules to evaluate the node where the replica is located. Select a node with a low rating to delete the representation. This part uses a multi-objective optimization algorithm to estimate the node where the model is situated, comprehensively assesses the node performance in three aspects: CPU load capacity, memory load capacity, and disk space load capacity, and uses the node performance evaluation level to select.

4 Evaluation indicators

This paper proposes the following five evaluation indicators:

(1) *Node disk space load capacity* The lower the disk space occupancy, the greater the disk space remaining, and the stronger the load capacity of the disk, and the node with low disk occupancy should be preferred when placing the secondary. The node disk occupancy P_{disk} in this paper is expressed as the ratio of the amount of disk used to the total disk space.

(2) *Node memory utilization* The lower the node memory share, the stronger the memory load capacity, and the node with a standard memory share should be preferred when placing the copy. The node memory utilization P_{RAM} in this paper is expressed as the ratio of the memory used by the node running process to the total memory of the node.

(3) *Node CPU usage* The lower the node CPU usage, the stronger the CPU load capacity, and the node with common node CPU utilization is preferred when placing the replica. In this paper, the node CPU utilization of the P_{CPU} is expressed as the ratio of the CPU execution time of non-idle processes to the total execution time of the CPU.

(4) *Rack load rate* Rack load rate is an important indicator to measure the load balance between racks in the cluster, and racks with a low rack load rate should be preferred when selecting racks. In this paper, the rack load rate P_{rack} is expressed by the ratio of the total used space of all host nodes in the rack to the entire disk space of all host nodes in the rack.

(5) *Network distance between racks* The size of the network distance has a direct impact on the data transmission bandwidth. The smaller the network distance means that the bandwidth of data transmission is more significant, which will make the data transmission rate higher and speed up the read and write speed of data.

We use a simple method to define network distance: think of the network as a tree, the distance between two nodes is the sum of their lengths to the nearest common ancestor, and the hierarchy in the tree is not predetermined but relative to the data center, racks, and nodes, it is usually possible to set the level [28]. Generally, there are the following common situations, and the network distance gradually increases:

(1) process located on the same node;
(2) different nodes situated in the same rack;
(3) Nodes situated in different racks in the same data center;
(4) nodes located in other data centers.

For example, as shown in Fig. 1, there are data centers *D1* and *D2*, racks *R1, R2, R3*, and *R4*, nodes *N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, N11, N12*. As follows, we give descriptions of distances in four different situations, where *dis* represents distance:

The distance between the same nodes is: *dis(N1, N1) = 2.*

The distance between the other nodes located on the same rack is: *dis(N1, N3) = 2.*

The distance between the other nodes located in the same data center is: *dis (N1, N5) = 4.*

The distance between nodes located in other data centers is: *dis (N1, N8) = 6.*

Assuming that the network distance between node i and the client where the data copy is located is d_i, and the network distance from the node furthest to the client is d_{max}, we can use the formula (1) to describe the network distance.

$$D_i = \frac{d_i}{d_{\max}} \tag{1}$$

Among the above five evaluation indicators, node disk occupancy, node memory utilization, and node CPU utilization are the evaluation indicators of the node, and the rack load rate and inter-rack network distance are the evaluation indicators for the rack. Since these five indicators contribute to the evaluation values to different degrees, they need to be normalized separately. Section Review Value V_{node} Calculation Formula is shown in formula (2), rack evaluation value V_{rack} Calculation Formula is shown in formula (4).

$$V_{node} = \alpha * P_{disk} + \beta * P_{RAM} + \gamma * P_{CPU} \tag{2}$$

$$\alpha + \beta + \gamma = 1 \tag{3}$$

$$V_{rack} = \varphi * P_{rack} + (1 - \varphi) * D_i \tag{4}$$

This paper transforms the replica management problem to be solved into a multi-objective optimization problem. It looks for the relationship between multiple performance indicators and the number and location of replicas based on the plant root algorithm. Use the full objective function to represent the replica management problem to be solved by target(p), and consider five indicators comprehensively, as expressed by the formula.

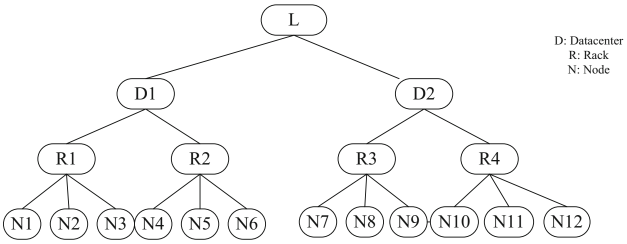


Fig. 1 Network topology diagram

Among them, 5 indicators correspond to the scale factor. Adjust different scale factors to increase the adaptability of the strategy. In the data copy management strategy based on the cuckoo algorithm, the total objective function value obtained by the individual is called the fitness value of the individual. In this section, a smaller fitness value indicates better individual fitness, which is the better the replica management solution sought.

The algorithm steps of strategy optimization in this paper are as follows:

1 Time interval *T*, through the CFRS algorithm [29] to calculate the number of copies that need to be added or deleted by the file;
2 Based on the current copy layout in the system, generate *Q* points, that is, *Q* feasible layout schemes *Q*{*q1, q2,… qn*}, and calculate the fitness for each feasible layout and record the most suitable *fitness*;
3 The current iteration is less than or equal to GC, execute (4)–(5), otherwise transfer (6);
4 Levy flight: randomly select a solution q_a from Q for Levy flight, obtain a new feasible solution q'_a, calculate its *fitness*', and then randomly select a solution q_b from *Q*, if *fitness*' > *fitness*$_b$, replace q_b with solution q'_a, otherwise discard q'_a;
5 Discard other solutions that are not optimal according to probability, and randomly generate feasible solutions, turn (3).
6 Select the most suitable layout scheme from the *Q* set, and the system will layout according to this scheme.

The pseudocode that produces the new solution in the cuckoo algorithm is shown in Algorithm 1.

The implementation pseudocode for Levy's flight in the cuckoo algorithm is shown in Algorithm 2:

The pseudocode for the cuckoo algorithm used in this paper is shown in Algorithm 3 [34]:

Algorithm 1: New layout generation method

```
Input: $n$ Number of copies added or deleted
       Current layout $q_{now}$
Output: Lower cycle layout $q_{next}$
          For  $i$=0 to $n$ do
             While $num_{change}(i)>0$  do
          // Randomly select one of the nodes that does not contain the file f_i and meets the constraints, and stores a copy of f_i
          $d_j$ ← $f_i$
          manage(i,j) ← 1
          $num_{change}(i)$ ← $num_{change}(i)$-1
    End While
    While $num_{change}(i)<0$  do
// Randomly select one of the nodes containing the file f_i ith and delete the copy of f_i
             $d_j$ ← $f_i$
             manage(i,j) ← 0
           $num_{change}(i)$ ← $num_{change}(i)$+1
    End While
 End For
    $q_{next}$ ← $manage$
return  $q_{next}$
```

Algorithm 2: Levy Fly

```
Input: The optimal solution in the population is $q_{best}$
       Levy Flight's solution $q_a$
Output: Levy flight was solved $q'_a$
        $Leay(\lambda)$ ← Calculate the Levy flight nonce number
        α ← $0$
          For  $i$=1 to $n$
             If  $manage_{best}[i][]!=manage_q[i][]$   then
          α ← α+1
  End If
 End For
If  $Leay(\lambda)>0$  then
  $q'_a$ ← Levy flight is performed in a file in layout $q_a$ that is different from layout $q_{best}$
End If
If  $Leay(\lambda)<0$  then
  $q'_a$ ← The same file in layout $q_a$ as layout $q_{best}$ is performed for the levy flight
  End If
Return  $q'_a$
```

Algorithm 3: Multi-factor comprehensive evaluation method

```
Input: $n$ Number of copies added or deleted
        The current layout $q_{now}$
        The maximum number of iterations $GN$
        Number of $Q$ layout
Output: Optimal layout $q_{best}$
        t←0, k←0
        While K<Q   do
          q← Generate new solutions using algorithm 1
  End While
While $t<GN$   do
        $maxFitness$←The fitness of the solution with the highest fitness of all solutions
        The most adaptable of all solutions
        $q_a$←Randomly one non-optimal solution among all solutions
        $q'_a$← Using algorithm 2 for the Levi flight yields a new solution
        $q'_aFitness$←Calculate the fitness of $q'_a$
        $q_b$ ←Randomly one non-optimal solution among all solutions
        If   $q'_aFitness > q_bFitness$   then
        $q_b$←$q'_a$
      // $q'_a$   replaces $q_b$
    End if
  $i$←0
While   $i<Q$ do
    If   $rand<p$   then
      // Algorithm 1 produces a new solution and puts it into the solution set
  End If
End While
 $Sort(q)$
// Keep the solution ranked in Q and keep it for the next generation

End While
return   $q_{best}$
```

Formula (2) in the equilibrium factors α, β, and γ can be adjusted by the administrator according to actual needs.

In this article, we set the value of α to 0.55, the value of βto 0.18, and the value of γ to 0.27 according to the degree of influence of the three indicators on node selection. Therefore, the calculation formula of the node evaluation value is:

$$V_{node} = 0.55 * P_{disk} + 0.18 * P_{RAM} + 0.27 * P_{CPU} \tag{5}$$

The balance factor φ in formula (5) can be adjusted by the administrator according to the actual needs.

In this paper, we set the value of φ to 0.7 based on the degree of influence of two indicators on rack selection. According to the replica placement strategy proposed in this paper, the rack evaluation value calculation formula is divided into two formulas: formula (6) and formula (7).

$$V_{rack} = 0.7 * P_{rack} + 0.3 * D_i \tag{6}$$

$$V_{rack} = 0.7 * P_{rack} + 0.3 * \frac{1}{D_i} \tag{7}$$

The critical point of the multi-factor comprehensive evaluation method is determining the appropriate weight of each factor. The literature [28] is used to read the relevant literature and define the weight of each factor according to the different importance of different factors. The methods in this literature rely on expert experience and can only roughly reflect the relative importance of factors, and this paper needs to optimize the weight allocation further.

Multivariate evaluation methods are divided into two categories: objective empowerment evaluation method and subjective empowerment evaluation method [30]. The subjective empowerment evaluation method is based on the relevant experience of the expert's subjective judgment,

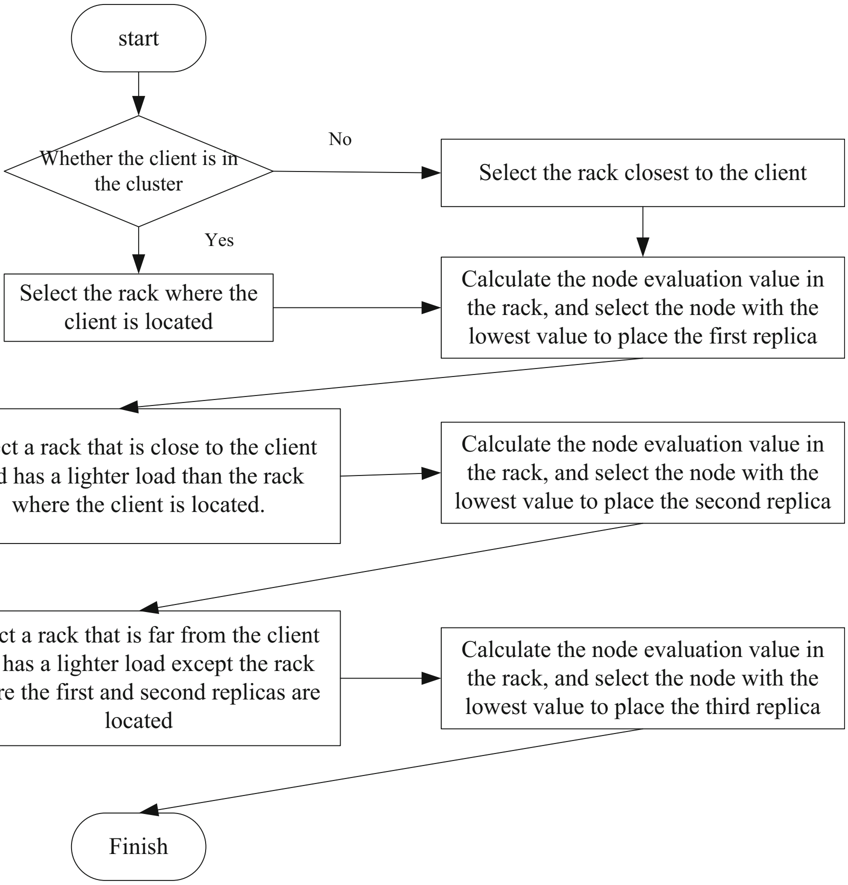

start
Whether the client is in the cluster
No
Select the rack closest to the client
Yes
Select the rack where the client is located
Calculate the node evaluation value in the rack, and select the node with the lowest value to place the first replica
Select a rack that is close to the client and has a lighter load than the rack where the client is located.
Calculate the node evaluation value in the rack, and select the node with the lowest value to place the second replica
Select a rack that is far from the client and has a lighter load except the rack where the first and second replicas are located
Calculate the node evaluation value in the rack, and select the node with the lowest value to place the third replica
Finish

Fig. 2 Flowchart of replica placement strategy based on evaluation values and load balancing

artificially defined the importance of factors to determine the weight coefficient, and then on the node performance comprehensive evaluation. Common subjective empowerment evaluation methods have analytic hierarchy method, fuzzy evaluation method, index weighting method, etc., although this kind of method will be affected by human factors, according to the relevant expert experience to determine the importance between the factors, through the above methods for weight distribution can still achieve good results. The objective weighting evaluation method is to determine the weight coefficient according to the coefficient of variation of each factor or the correlation relationship between factors and then comprehensively evaluate the node's performance. Standard objective weighting evaluation methods are the coefficient of variation method, entropy value method, neural network analysis method, etc., this kind of method needs to consider the interrelationship between the factors, and the weight is determined according to the amount of initialization information provided by each factor. The comprehensive evaluation can be carried out more accurately.

According to the above methods, we use the analytic hierarchy method in the subjective weighting evaluation method to determine the influence of each evaluation factor on the performance of data nodes [31]. The method belongs to the theory of operations research, which was first developed by Thomas in the United States. Set *i* proposed that the technique is mainly applied to complex decision-making problems. The difficult issue of this method is decomposed into several influencing factors, and the selection and judgment are made through quantitative calculation by comparing the importance of each evaluation factor.

5 Replica placement policy based on evaluation values and load balancing

The default number of data replicas proposed in this paper is 3 for the replica placement strategy based on evaluation values and load balancing, and the replica placement is as follows:

5.1 First copy

If the client is in the cluster, place the first copy on different nodes in the rack where the client node is located; If the client is not in the cluster, place the first copy on the node in the closest rack to the client. The selection of nodes on the rack is determined by calculating the section review

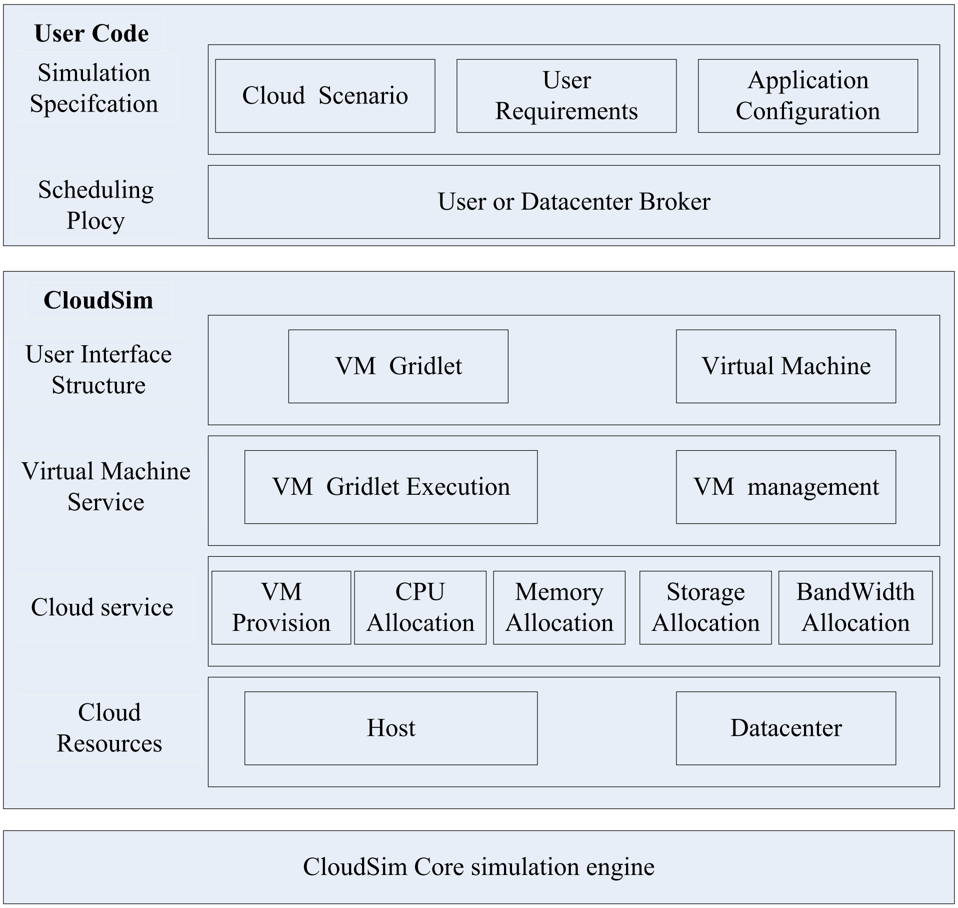


Fig. 3 CloudSim architecture diagram

Table 1 Host Type Configuration

host type	Hard disk capacity (GB)	Memory (MB)	CPU(MIPS/CORE)
Type 1	1024	8192	3600/32
Type 2	1024	8192	3200/28
Type 3	1024	8192	2800/24
Type 4	512	4096	2400/20
Type 5	512	4096	2000/16
Type 6	512	4096	1600/12

value V_{node}. The node with the lowest evaluation value is selected to place the first copy. Placing replicas on different nodes in the same rack can ensure that when the node where the client is located fails, data can be quickly retrieved from other nodes in the same rack for recovery.

5.2 Second copy

The second copy should be placed in a different rack than the first copy. First, according to formula (6), calculate the evaluation value of the racks except for the rack where the first replica is located, select the rack with the smallest rack evaluation value, and select the nodes on the rack by calculating the node evaluation value V_{node}. It is decided that the node with the smallest evaluation value is determined according to formula (5) to place the second replica. The second replica is placed on a rack with a low rack load rate and a close network distance. The second replica is placed on a rack with a low rack load rate and a tight network distance, and if the client's rack fails, data can be quickly retrieved from the rack closer to the client's rack for recovery. In addition, the rack load ratio can balance the load conditions between racks.

5.3 Third copy

The third copy is placed in a different rack than the first and second replicas. First of all, according to the formula

Table 2 Distribution of host nodes in a rack

Rack	host node	host type	Storage space/GB
Rack 1	NameNode	Type 1	7680
	DataNode 1	Type 1	
	DataNode 2	Type 1	
	DataNode 3	Type 1	
	DataNode 4	Type 3	
	DataNode 5	Type 3	
	DataNode 6	Type 3	
	DataNode 7	Type 5	
	DataNode 8	Type 5	
	DataNode 9	Type 5	
Rack 2	DataNode 1	Type 2	6144
	DataNode 2	Type 2	
	DataNode 3	Type 2	
	DataNode 4	Type 4	
	DataNode 5	Type 4	
	DataNode 6	Type 4	
	DataNode 7	Type 6	
	DataNode 8	Type 6	
	DataNode 9	Type 6	
Rack 3	DataNode 1	Type 1	7680
	DataNode 2	Type 1	
	DataNode 3	Type 1	
	DataNode 4	Type 3	
	DataNode 5	Type 3	
	DataNode 6	Type 3	
	DataNode 7	Type 5	
	DataNode 8	Type 5	
	DataNode 9	Type 5	
Rack 4	DataNode 1	Type 2	6144
	DataNode 2	Type 2	
	DataNode 3	Type 2	
	DataNode 4	Type 4	
	DataNode 5	Type 4	
	DataNode 6	Type 4	
	DataNode 7	Type 6	
	DataNode 8	Type 6	
	DataNode 9	Type 6	
Rack 5	DataNode 1	Type 1	7680
	DataNode 2	Type 1	
	DataNode 3	Type 1	
	DataNode 4	Type 3	
	DataNode 5	Type 3	
	DataNode 6	Type 3	
	DataNode 7	Type 5	
	DataNode 8	Type 5	
	DataNode 9	Type 5	

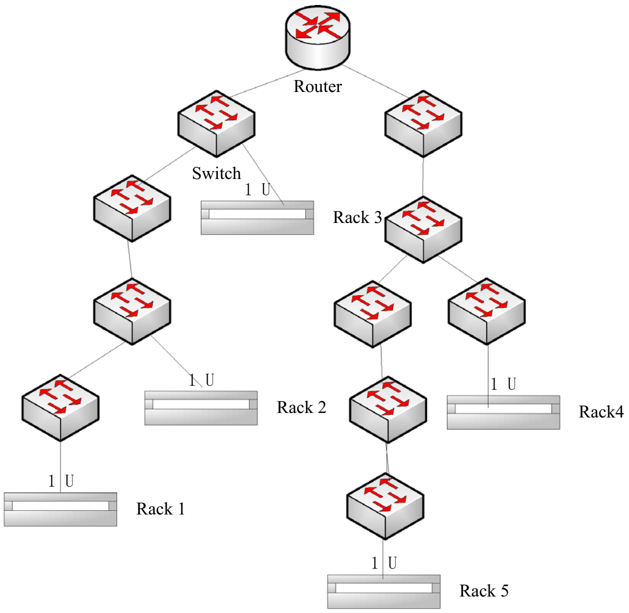


Fig. 4 Cluster network topology diagram

Table 4 Network distances between racks in a cluster

Rack	Rack 1	Rack 2	Rack 3	Rack 4	Rack 5
Rack 1	–	3	5	9	11
Rack 2	3	–	4	8	10
Rack 3	5	4	–	6	8
Rack 4	9	8	6	–	6
Rack 5	11	10	8	6	–

(7), the rack evaluation value except for the first copy and the second copy is located, select the rack with the smallest evaluation value, the selection of nodes on the rack should be determined by calculating the node evaluation value V_{node}, and select the node with the smallest evaluation value according to formula (5) to place the third copy. The third copy is placed on a rack with a low rack load rate and a long network distance, and the choice of a rack farther away from the client's rack is to consider that in practical applications, the probability of simultaneous failure of the closer rack is higher, and the likelihood of the distant rack to fail at the same time is small, so choose to store one of the copies on the rack far away, although this practice consumes specific resources, but to a large extent ensures data security and availability.

The specific copy placement flowchart is shown in Fig. 2.

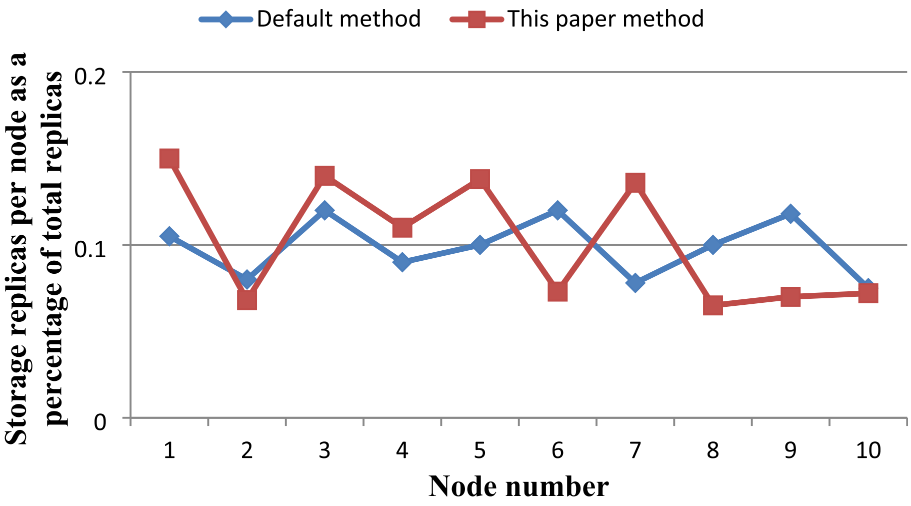


Fig. 5 Percentage of total replicas stored by node

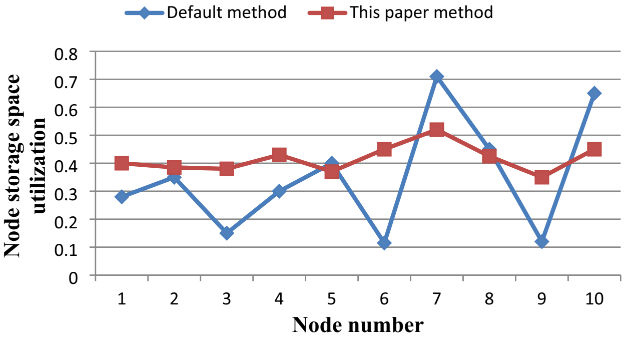


Fig. 6 Storage space usage of each node

6 Simulation experiments and analysis of results

In cloud computing research, new algorithms or strategies are constantly emerging. They usually need to be constantly tested and validated before they can be officially applied in a natural cloud environment. However, initial testing and validation in a natural cloud environment are hazardous, and it is likely to accidentally destroy existing services and data in the natural cloud environment, and the cost is exceptionally high. CloudSim is therefore a very suitable alternative to seamlessly modeling the real cloud environment to fully replicate a real cloud environment in the framework, and then add the algorithm or strategy to be verified to the modeled virtual cloud environment in an appropriate way, which can easily and cost-effectively test and validate the related work.

Before extending CloudSim, you need to give a brief introduction to the system structure of the platform. The platform adopts a hierarchical structure, with four levels of SimJava, GridSim, CloudSim, and user code from the bottom up. CloudSim version 2.0 was previously the SimJava discrete event simulation engine, which was responsible for performing the core functions of the upper layer, building system components, communicating between different system components, and analog clock management. With the system update, SimJava has been removed from the emulation platform architecture to improve the system's functionality. The CloudSim layer extends the core functions provided by the system. It is the core layer of the platform, which can perfectly support experiments in the cloud computing environment. For example, according to the expansion of bandwidth and storage performance, determine whether the task scheduling algorithm meets the target requirements [32]. The top layer is the user code layer, which describes information such as task requests, the number, and the characteristics of resources, etc. If you want to verify the method or theory, you should extend the platform according to your research theory and call the extended classes and practices in the simulation platform. The architecture and main components are shown in Fig. 3:

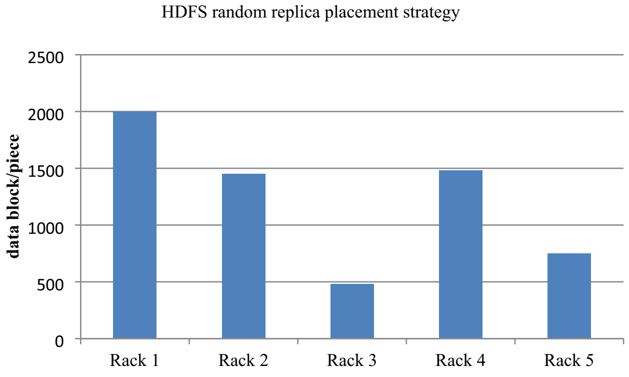


Fig. 7 Data copy distribution in HDFS random copy placement strategy

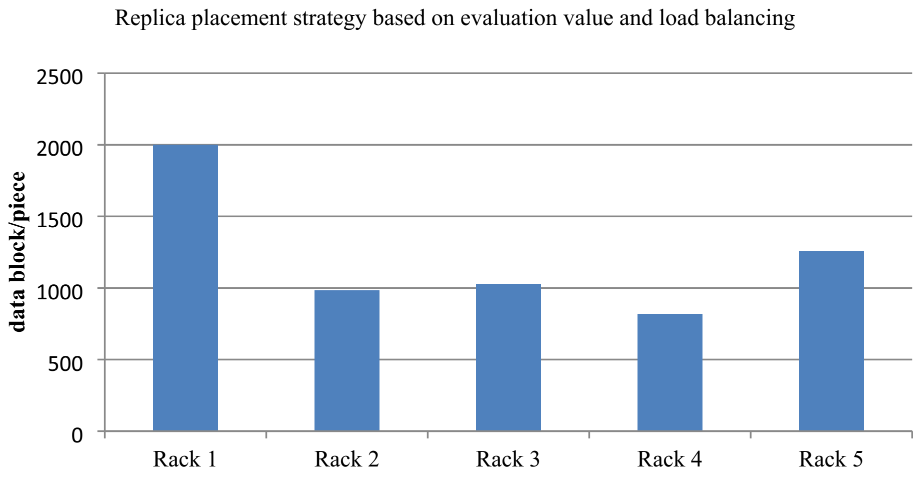


Fig. 8 Data replica distribution in a replica placement strategy based on evaluation values and load balancing

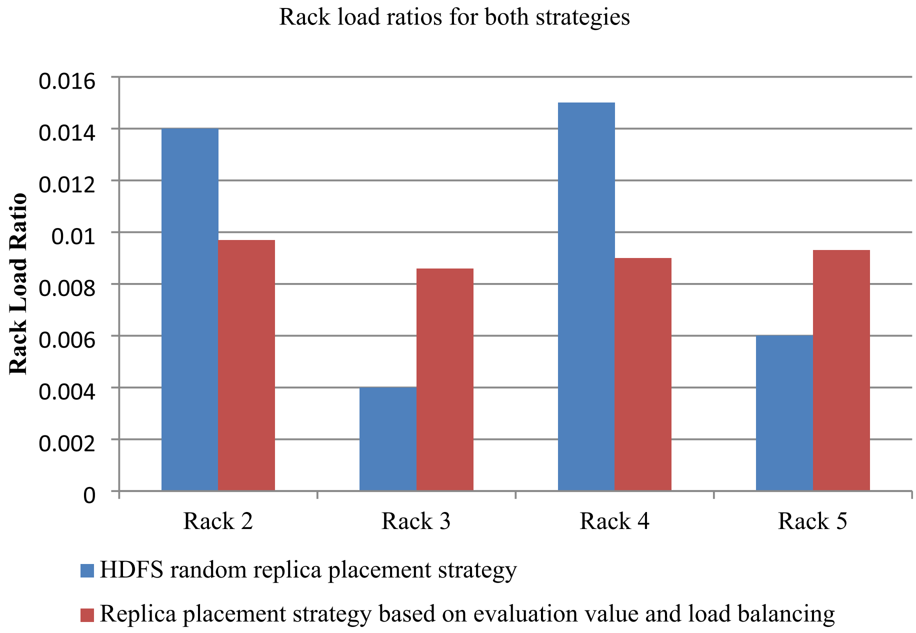


Fig. 9 Rack load rate for two strategies

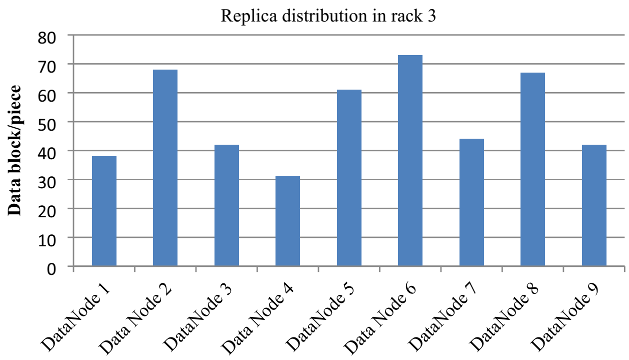


Fig. 10 Data replica distribution on rack 3 in the HDFS random copy placement strategy

7 Experimental setup

This experiment uses CloudSim-3.0 as a simulation tool. The hardware environment is *Intel(R) Core(TM) i7-10,700*, the memory is 16 GB, the operating system is Windows 10, and the compilation environment is JDK 1.8.0.

In the experiment, we will use CloudSim simulation software to simulate a heterogeneous cluster with 5 racks, each containing 9 DataNode nodes and rack 1 containing 1 NameNode node. In this experiment, we are targeting a heterogeneous node network, so the host nodes are configured to 6 different types, the specific configuration of the host type is shown in Table 1, and the layout of the host nodes on each rack is shown in Table 2.

Table 5 Number of blocks in a rack

Rack	Rack 2	Rack 3	Rack 4	Rack 5
Data blocks/blocks	950	1050	867	1133

In a Hadoop cluster, the network structure is typically a tree topology. To verify the impact of network distance on replica placement racks, we set up a network topology shown in Fig. 4 and the network topology distance shown in Table 4.

This experiment needs to calculate the disk space utilization, memory utilization, and CPU utilization of the node, where the node CPU utilization can be obtained by the *getUtilizationOfCpu()* function in *org.cloudbus.cloudsim.HostDynamicWorkload.java*. There is an error in the *getUtilizationOfRam()* function that comes with *sources.org.cloudbus.cloudsim.HostDynamicWorkload.java*, so it is modified. Add the node disk space utilization function getUtilizationOfStorage() in org.cloudbus.cloudsim.Host.java and node memory utilization function *getUtilizationOfRam()*. The rack load rate is calculated based on the usage of all host nodes on the rack. In this experiment, we added the rack load rate calculation function *getUtilizationOfDCStorage()*.

8 Experiment results and analysis

In this experiment, we used the Data Node 1 node in rack 1 as the client and submitted 2000 data blocks to the server through the client, each with a data block size of 64 MB. The number of replicas defaulted to 3. That is, the system needed to process a total of 6000 data blocks.

The experiment first tests whether the replica placement strategy proposed in this paper can be more balanced and reasonable than the distribution of replicas in each node in the system default replica placement strategy. In the simulation experiment, multiple clients write 600 files to the cluster at the same time, that is, 1800 block copies need to be written, and the default copy placement strategy is used for the initial write of files. Now we start to simulate the scenario of the system after running for a period of time, the access heat of the file changes, the copy factor dynamic adjustment strategy proposed in this paper begins to take effect, and the factor analysis method proposed in this paper is used to place the copy of the increased data copy. At this point, observe the replica storage status and storage resource usage of each node in the cluster, as shown in Fig. 5 and Fig. 6.

Figure 5 shows the ratio of the number of stored replicas to the total number of replicas in the system, and under the default placement strategy, node 7 with high performance stores fewer replicas, and node 6, 8, and 9 with poor performance stores more replicas, resulting in unbalanced load between nodes. After the model optimization in this paper, nodes 6,7,8,9 comprehensively consider the storage resources and computing resources, and place an appropriate number of replicas on the node. In Fig. 6, there are three sets of data, and the “default method” is the default replica placement policy in the distributed file system; “This paper method” is the proposed dynamic copy placement strategy. From the figure, it is found that the disk storage space usage of each node in the default replica placement strategy varies greatly, with the highest node utilization rate reaching 71%, and some nodes only about 12%, which is relatively idle and has a serious uneven distribution of replicas. The default replica placement strategy also has a balancer tool to adjust the replica distribution, which consumes a lot of system resources and has a great impact on computing efficiency if the cluster size is large. The paper method optimized by analytic hierarchy method can make the storage space utilization of each node in the cluster relatively balanced, and there are no nodes with too much difference in utilization, and the standard deviation of storage space utilization of each node in the default method is 0.206, and the standard deviation of the method in this paper is 0.058. Experiments show that the node data distribution of the improved method is adapted to the overall performance, which reduces the impact of different node performance on the system execution time.

8.1 The starting load is no load

When the data load is empty at the start of the system, we use the HDFS random copy placement strategy and the copy placement strategy based on evaluation values and load balancing to conduct the copy placement experiment.

8.1.1 Rack-to-rack load balancing

When the HDFS random copy placement strategy is used experimentally, the specific distribution of data copies on rack 1, rack 2, rack 3, rack 4, and rack 5 is shown in Fig. 4. When using a copy placement strategy based on evaluation values and load balancing, the specific distribution of data copies on racks 1, rack 2, rack 3, rack 4, and rack 5 is shown in Fig. 7.

Figure 7 reflects the distribution of data copies across racks when HDFS random copy placement strategy is adopted (Fig. 8). We can see from Fig. 9 that there are 2000 data blocks in rack 1 because the client is on rack 1, and according to the HDFS random copy placement strategy, 2000 copies are placed on the node where the client is located. Another 4000 copies are randomly distributed on rack 2, rack 3, rack 4, and rack 5. We can see that the number of data blocks on rack 2 and rack 4 is more than the number of data blocks on rack 3 and rack 5, but the storage space of rack 2 and rack 4 is smaller than that of rack 3 and rack 5. The reason for this result is that the copy uses a random selection strategy when selecting nodes. This shows that the HDFS random copy placement strategy does not guarantee load balancing between racks.

Figure 8 shows the specific distribution of data replicas on different racks using a replica placement strategy based on evaluation values and load balancing. We can see from Fig. 10 that there are 2000 blocks of data on rack 1 because the client is on rack 1, and the replica placement strategy based on the evaluation value and load balancing requires the first vice to be placed on the node in the rack where the client is located. In addition, 4000 copies are distributed on rack 2, rack 3, rack 4, and rack 5. We can see that the number of data blocks on rack 2 and rack 4 is less than the number of data blocks on rack 3 and rack 5. This is because the storage space of rack 2 and rack 4 is smaller than that of rack 3 and rack 4. Using the evaluation value and load balancing copy placement strategy to consider the rack load situation, racks with ample storage space can place more copies, and racks with small storage space can place fewer copies. This reduces load variability between racks and allows for better load balancing.

The number of data blocks on rack 2, rack 3, rack 4, and rack 5 is shown in Table 5. Based on the different storage space sizes of rack 2, rack 3, rack 4, and rack 5, to better reflect the replica placement strategy based on evaluation values and load balancing to ensure good load balancing between racks, we calculated the rack load rate. The rack load rate of these four racks is shown in Table 6. From Table 6, it can be seen that the difference in rack load rate between these four racks is slight, and the load between racks is relatively balanced. Among them, the rack load rate of rack 2 is more significant because rack 2 is closest to the client, and the second copy is preferred to rack 2 for placement when the rack load rate is similar; The rack load rate of rack 5 is higher because rack 5 is the farthest away from the client, and the third replica is preferred to rack 5 for placement when the rack load rate is similar.

Table 6 Rack Load Rates

Rack	Rack 2	Rack 3	Rack 4	Rack 5
Rack load rate	0.00966	0.00854	0.00882	0.00922

Table 7 Initial rack loads

Rack	Rack 1	Rack 2	Rack 3	Rack 4	Rack 5
Initial load rate/pc	100	50	500	300	100

In addition, we can compare the rack load rate of the HDFS random copy placement strategy with the secondary placement strategy based on evaluation values and load balancing, as shown in Fig. 9.

Figure 9 shows the comparison of the rack load ratio between the HDFS random replica placement strategy and the replica placement strategy based on evaluation values and load balancing. As shown in Fig. 9, the rack load rates of the four racks in the HDFS random copy placement strategy vary greatly. In contrast, the rack load rates differ less across racks in the replica placement strategy based on evaluation values and load balancing. Compared with the HDFS random copy placement strategy, the copy placement strategy based on evaluation values and load balancing can reduce the rack load difference between each rack and better achieve inter-rack load balancing.

8.1.2 Load balancing between nodes

The replica placement strategy based on evaluation values and load balancing not only considers the load balancing between racks but also considers the real-time performance of the nodes, which we use as an example to illustrate the distribution of replicas on each node in rack 3. When the HDFS random copy placement strategy is used in experimental statistics, the specific distribution of replicas in rack 3 is shown in Fig. 10. When using the replica placement strategy based on evaluation value and load balancing proposed in this article, the specific distribution of replicas in rack 3 is shown in Fig. 11.

Figure 10 shows the specifics of the data copy distribution on shelf 3 when the HDFS random copy placement

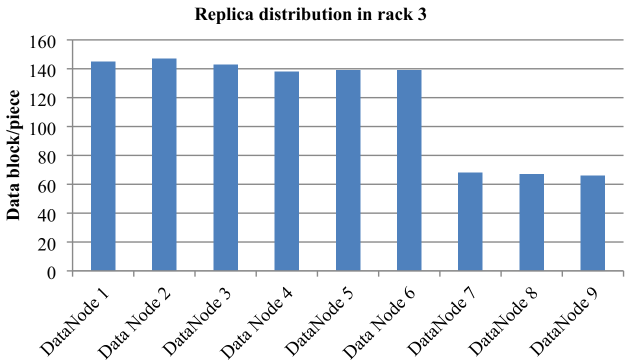


Fig. 11 Distribution of data copies on rack 3 based on evaluation value and load balancing copy placement strategy

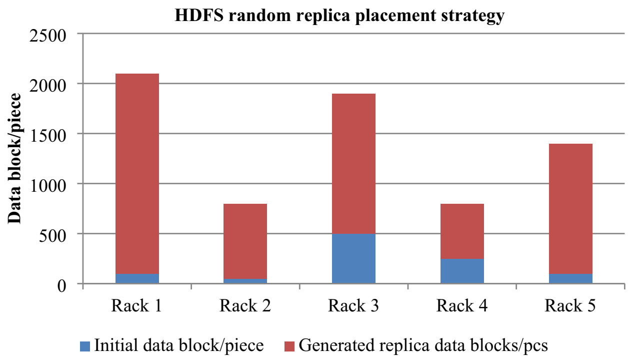


Fig. 12 Data copy distribution in HDFS random copy placement strategy

strategy is used. We can see that the number of replicas distributed on each node in rack 3 is unbalanced. Some nodes have a large number of replicas, and some nodes have a small number of replicas. This is because the HDFS random replica placement strategy is used when placing replicas. Node performance is not considered, just random placement.

Figure 11 shows the distribution of data replicas on shelf 3 with a replica placement strategy based on evaluation values and load balancing. From Fig. 11, we can see that the number of data blocks on DataNode 1, DataNode 2, and DataNode 3 is the largest and the number of data blocks on these three DataNodes is the same, the number of data blocks on DataNode 4, DataNode 5, DataNode 6 is less than the first three DataNodes. The number of data blocks on these three DataNodes is the same, DataNode 7, and DataNode 8. The number of data blocks on DataNode 9 is the smallest, but the number of data blocks on the three DataNodes is the same. The reason for this distribution is because the first three DataNodes have the same and best performance, the middle three DataNodes are worse than the first three DataNodes in terms of CPU performance, and the last three DataNodes are worse than the first six DataNodes in terms of hard disk capacity, memory and CPU performance. Therefore, nodes with poor performance have fewer data blocks. From this, we can see that the replica placement strategy based on the evaluation value and load balancing can achieve load balancing between nodes in the same rack.

8.2 The starting load is not empty

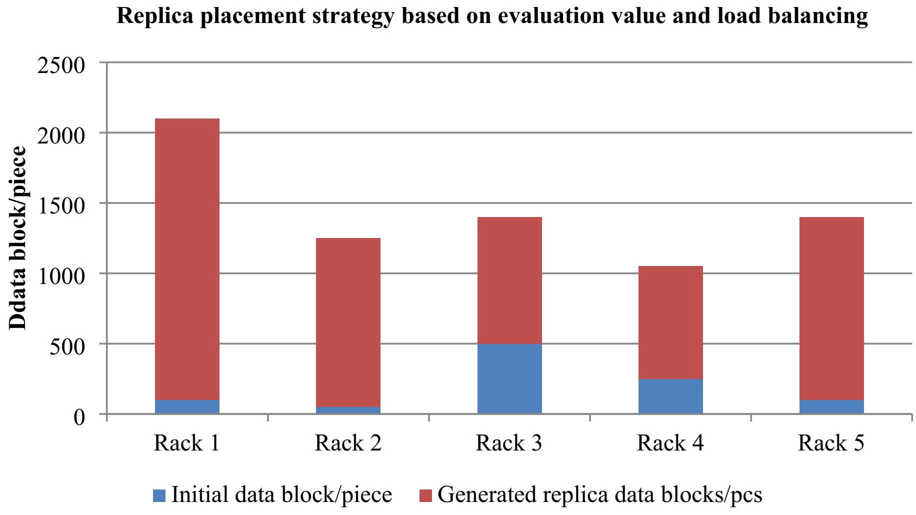
The data load is not empty during the system start state, and the initial load of each rack is shown in Table 7. We use the HDFS random replica placement strategy and the replica placement strategy based on evaluation value and load balancing to conduct replica placement experiments.

8.2.1 Rack-to-rack load balancing

When the HDFS random copy placement strategy is used in experimental statistics, the distribution of data copies on rack 1, rack 2, rack 3, rack 4, and rack 5 is shown in Fig. 12. When the replica placement strategy based on evaluation value and load balancing is adopted, Fig. 13 shows the distribution of data copies across racks 1, rack 2, rack 3, rack 4, and rack 5.

Figure 12 reflects the distribution of data blocks on each rack after the HDFS random copy placement strategy is used to place the copy. We can see from Fig. 11 that there are 2100 blocks in rack 1, including 100 initial blocks and 2000 generated copy blocks. On rack 1, there are 2000 copies placed on the node where the client resides according to the HDFS random copy placement strategy. In addition, you can see that the number of data blocks on rack 2, rack 3, rack 4, and rack 5 is quite different before and after replica placement because the HDFS random copy placement strategy does not consider the rack load

Fig. 13 Data replica distribution in a replica placement strategy based on evaluation values and load balancing



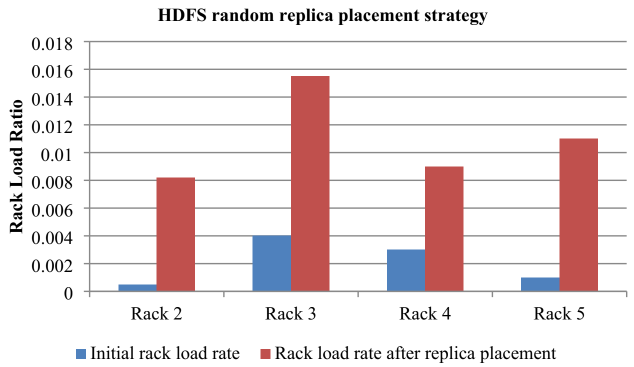


Fig. 14 HDFS Random Copy Placement Strategy Rack Load Rate Before and After Replica Placement

when placing replicas, but uses random node selection to identify replicas.

Figure 13 reflects the distribution of data replicas on rack 1, rack 2, rack 3, rack 4, and rack 5 when using a copy placement strategy based on evaluation values and load balancing. We can see from Fig. 13 that there are 2100 blocks in rack 1, including 100 initial blocks and 2000 generated copy blocks, because on rack 1, there are 2000 copies placed on the node where the client resides, depending on the replica placement strategy based on evaluation values and load balancing. Another 4000 copies are distributed in rack 2, rack 3, rack 4, and rack 5. You can see that the number of data blocks on rack 2 and rack 4 is less than on rack 3 and rack 5. This is because the storage space of rack 2 and rack 4 is smaller than that of rack 3 and rack 5. Using the evaluation value and load balancing copy placement strategy to consider the rack load situation, racks with ample storage space can place more copies, and racks with small storage space can place fewer copies. This reduces load variability between racks and allows for better load balancing.

To better reflect that the replica placement strategy based on evaluation value and load balancing can ensure good load balance between racks, we calculate the rack load rate of four racks in the HDFS random copy placement strategy and the replica placement strategy based on evaluation value and load balancing, respectively, the HDFS random copy placement strategy rack load rate is shown in Fig. 14. The replica placement strategy based on evaluation value and load balance is shown in Fig. 15.

Figure 14 shows the HDFS random replica placement strategy compared to the rack load before and after replica placement. As can be seen from Fig. 14, the initial rack load rate difference between these four racks before the replica placement is significant. The difference in the rack load rate of the four racks is even more significant after the replica placement is completed, so we can see that the HDFS random copy placement strategy cannot reduce the rack load difference between racks and cannot guarantee the load balance between racks.

Figure 15 compares rack load before and after replica placement based on evaluation value and load balancing strategy. From Fig. 15, we can see that the initial rack load rate difference between these four racks before the replica placement is significant. The difference in the rack load rate of these four racks becomes smaller after the replica placement is completed, so we can see that the replica placement strategy based on the evaluation value and load balancing can reduce the load difference between racks and better achieve inter-rack load balancing.

8.2.2 Load balancing between nodes

When the HDFS random copy placement strategy is used experimentally, the specific distribution of data copies in

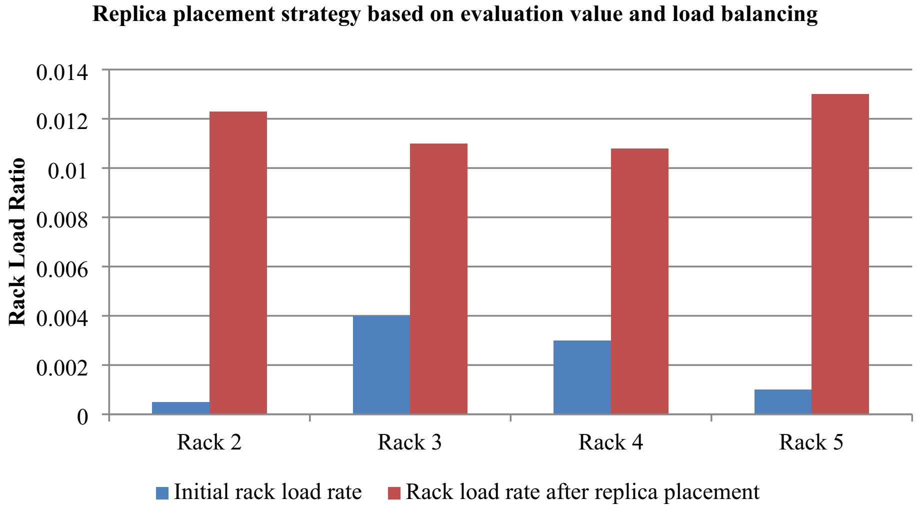


Fig. 15 Replica placement strategy based on evaluation values and load balancing The rack load rate before and after copy placement

each node in rack 3 is shown in Fig. 16. When using a replica placement strategy based on evaluation values and load balancing, the specific distribution of data replicas across the nodes in rack 3 is shown in Fig. 16.

Figure 16 reflects the specific situation where data copies are distributed on rack 3 using the HDFS random copy placement strategy. We can see that the number of replicas distributed across the nodes in Rack 3 is uneven because the HDFS random copy placement strategy only places replicas randomly when placing replicas without regard to node performance.

Figure 17 shows the distribution of data replicas on rack 3 with a replica placement strategy based on evaluation values and load balancing. From Fig. 17, we can see that the number of data blocks on DataNode 1, DataNode 2, and DataNode 3 is the largest. The number of data blocks on these three DataNodes is the same, the number of data blocks on DataNode 4, DataNode 5, and DataNode 6 are less than the first three DataNodes. The number of data blocks on these three DataNodes is the same. DataNode 7, DataNode 8, and DataNode 9 have the fewest blocks, but the number of blocks on the three Data Nodes is essentially the same. The reason for this distribution is because the first three DataNodes have the same and best performance, the middle three DataNodes are worse than the first three DataNodes in terms of CPU performance, and the last three DataNodes are worse than the first six DataNodes in terms of hard disk capacity, memory and CPU performance. Therefore, nodes with poor performance have fewer data blocks. From this, we can see that the replica placement strategy based on the evaluation value and load balancing can achieve load balancing between nodes in the same rack.

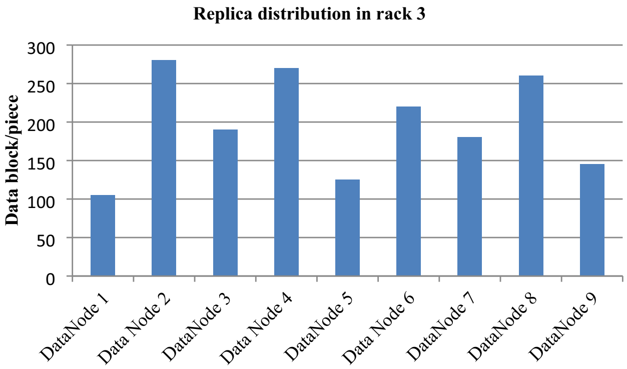


Fig. 16 HDFS Random Copy Placement Strategy Replicas distribution in rack 3

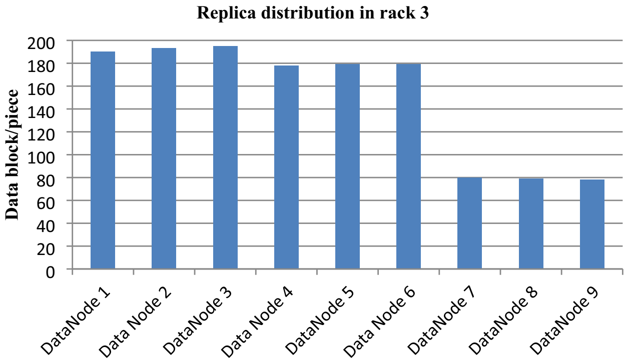


Fig. 17 Distribution of replica placement strategies in rack 3 based on evaluation values and load balancing

Next, we'll analyze the time overhead of both strategies. We define the time overhead as the time overhead of submitting a 64 MB block of data to a node, recording the time taken from the time a block commit request is made to the completion of the data block storage.

In this experiment, we will use the HDFS random copy placement strategy and the copy placement strategy proposed in this paper to submit 10 data blocks to the same node 10 times in the same environment, recording the time cost of each data block separately. The time cost of each data block is selected in the experimental node, and the time cost of the two strategies is shown in Table 8.

To compare the two data sets in Table 8 more intuitively, we use a line chart for analysis, as shown in Fig. 18

Figure 18 reflects the time overhead of submitting a data block using the HDFS random copy placement strategy and the copy placement strategy based on evaluation values and load balancing. As can be seen from the figure, the time overhead generated by the HDFS random copy placement strategy is less than that of the replica placement strategy based on evaluation value and load balancing, which occurs because the HDFS random copy placement strategy adopts a random placement strategy when placing data block copies, while the copy placement strategy based on evaluation value and load balancing needs to computer the

Table 8 The time overhead of the two strategies

Data blocks	HDFS random copy placement strategy	Replica placement policy based on evaluation values and load balancing
Data blocks 1	5.91	7.59
Data blocks 2	5.97	7.65
Data blocks 3	5.67	7.37
Data blocks 4	5.98	7.26
Data blocks 5	5.92	7.84
Data blocks 6	5.89	7.45
Data blocks 7	5.76	7.69
Data blocks 8	5.69	7.85
Data blocks 9	6.02	7.53
Data blocks 10	5.85	7.92

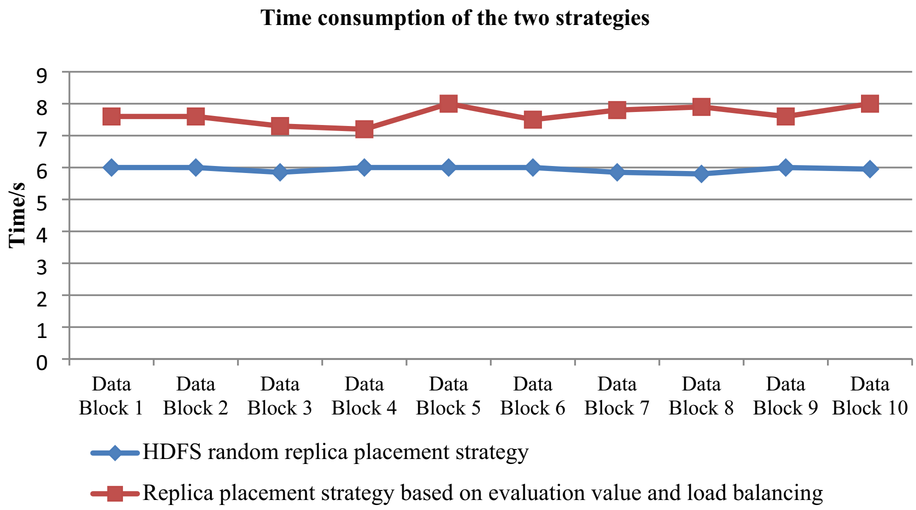


Fig. 18 The time overhead of the two strategies

shelf evaluation value and node evaluation value before setting the copy, which will consume a certain amount of time. In addition, based on the evaluation value and load balancing replica placement strategy, a replica is chosen to be placed on a rack far away from the network, so it also consumes some time.

From the above comparison, we can see that the replica placement strategy based on evaluation values and load balancing proposed in this paper not only ensures load balancing between racks but also achieves load balancing between nodes in the rack. In addition, the copy placement strategy proposed in this paper also considers the network distance between racks, placing the second copy and the third copy on the rack closer and farther away from the client, respectively. Although this will consume specific network resources, but significant improve data security and reliability. Of course, there are some drawbacks to the replica placement strategy based on evaluation value and load balancing. For example, compared with the HDFS random copy placement strategy, the replica placement strategy based on evaluation value and load balancing has a more significant time cost when placing data blocks.

Through the analysis and research of the default replica creation and placement strategy of the distributed file system, aiming at its shortcomings, the proportion of load capacity of each factor of cluster nodes is quantitatively described, the evaluation value of each node is calculated through the comprehensive evaluation model of cluster node performance given, and the best selection is carried out according to the node evaluation value. An improved copy placement strategy is proposed. Experimental results show that the optimized replica placement strategy in this paper can improve the overall computing efficiency and disk space utilization of the system and help the replica distribution in the cluster to be more balanced and reasonable.

9 Conclusion

This paper first briefly introduces the HDFS distributed file system, then introduces and analyzes the HDFS random copy placement algorithm, points out the shortcomings, then studies file chunking and HDFS data reading and writing process, and finally proposes a copy placement strategy based on evaluation value and load balancing. Using five evaluation indicators, including node disk space load capacity, node memory utilization, node CPU utilization, rack load rate, and inter-rack network distance, both racks and nodes are analyzed, and evaluation functions are given. CloudSim simulation software is used to simulate the HDFS random copy placement strategy and the copy placement strategy based on evaluation value and load balancing and analyze and compare the experimental results. From the experimental results, it can be seen that the copy placement strategy based on evaluation value and load balancing proposed in this paper is better than the HDFS random copy placement strategy in terms of node load balancing, rack load balancing, and data security.

The replica placement strategy based on evaluation value and load balancing proposed in this paper focuses on the performance of rack load balancing, node load balancing, and data security. Still, it does but does not take much account of resource consumption. In addition, other factors are ignored in the current evaluation indicators, which need to be improved and improved in future work.

Acknowledgements This work is supported by the National Natural Science Foundation of China (61872284); Industrial field of general projects of science and Technology Department of Shaanxi Province (2020GY-012); Industrialization Project of Shaanxi Provincial Department of Education (21JC017); “Thirteenth Five-Year” National Key R&D Program Project (Project Number: 2019YFD1100901); Yulin science and technology project (2019-175); Natural Science Foundation of Shannxi Province, China (2014JM2-6127); The project sponsored by the scientific research

Foundation for the returned overseas Chinese scholars, SEM No.[2014] 1685.

Author contributions Qinlu He, Fan Zhang, Hao Fengand Zhen li conceived the study. Genqing Bian, Weiqi Zhang and Zhimin Yu assisted with computational and laboratory processing pipelines. Qinlu He directed the study and wrote the manuscript with input from co-authors. All authors reviewed and approved the manuscript.

Funding Industrial field of general projects of science and Technology Department of Shaanxi Province, 2020GY-012, Industrialization Project of Shaanxi Provincial Department of Education, 21JC017, Natural Science Foundation of Shannxi Province, China, 2014JM2-6127, The project sponsored by the scientific research Foundation for the returned overseas Chinese scholars, SEM No.[2014] 1685, National Natural Science Foundation of China, 61872284, “Thirteenth Five-Year” National Key R&D Program Project, 2019YFD1100901, Yulin science and technology project, 2019-175

Data availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflicts of interest.

References

1. Tao, M., Ota, K., Dong, M.: DSARP: dependable scheduling with active replica placement for workflow applications in cloud computing. IEEE Trans. Cloud Comput. **4**, 8 (2020)
2. Mansouri, N., Javidi, N., Zade, N.: Hierarchical data replication strategy to improve performance in cloud computing. Front. Comput. Sci. **15**(2), 17 (2021)
3. Al-Ramahi, N.M., Odeh, M., Alrabie, Z., et al.: The TOEQCC framework for sustainable adoption of cloud computing at higher education institutions in the kingdom of Jordan. Sustainability **14**(19), 12744 (2022)
4. Bhatta, D., Mashayekhy, L.: A bifactor approximation algorithm for cloudlet placement in edge computing. IEEE Trans. Parallel Distrib. Syst: A Publ. IEEE Comput. Soc. **33**(8), 1787–1798 (2022)
5. Cui, L., Zhang, J., Yue, L., et al.: A genetic algorithm based data replica placement strategy for scientific applications in clouds. Serv. Comput. IEEE Trans. **11**(4), 727–739 (2018)
6. Jiaojiao, Wu., Yanping, Li., Tianyin, W., et al.: CPDA: a confidentiality-preserving deduplication cloud storage with public cloud auditing. IEEE Access **7**, 160482–160497 (2019)
7. Guangyan, Z., Guiyong, Wu., Shupeng, W., et al.: Ca Co: an efficient Cauchy coding approach for cloud storage systems. IEEE Trans. Comput. **65**(2), 435–477 (2016)
8. He, L., Qian, Z. Shang, F.: A novel predicted replication strategy in cloud storage. J. Supercomput. **76** 4838–4856 (2020)
9. H Xu G Wang L Luo M Lei 2018 “The Design of Reliability Simulation of Cloud System in the Cloudsim,” 2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China 215–219.
10. Lizhen, C., Junhua, Z., Lingxi, Y., et al.: A genetic algorithm based data replica placement strategy for scientific applications in clouds. IEEE Trans. Serv. Comput. **11**(4), 727–739 (2015)
11. Souravlas, S., Sifaleras, A.: Binary-tree based estimation of file requests for efficient data replication. IEEE Trans. Parallel Distrib. Syst. **28**(7), 1839–1852 (2017)
12. Yuhua, L., Haiying, S.: EAFR: an energy-efficient adaptive file replication system in data-intensive clusters. IEEE Trans. Parallel Distrib. Syst. **28**(4), 1017–1030 (2017)
13. He, Q., Bian, G., Zhang, W., et al.: Research on routing strategy in cluster deduplication system. IEEE Access **9**, 135485–135495 (2021)
14. Ting, Y., Haibo, P., Wei, Li., et al.: An energy-efficient storage strategy for cloud datacenters based on variable K-coverage of a hypergraph. IEEE Trans. Parallel Distrib. Syst. **28**(12), 3344–3355 (2017)
15. He, Q., Zhimin, Yu., Bian, G., Zhang, W., Liu, K., Li, Z.: Research on key technologies of NBD storage service system based on load classification. AIP Adv. **11**, 125124 (2021). https://doi.org/10.1063/5.0071929
16. Juipin, Y.: Elastic load balancing using self-adaptive replication management[J]. IEEE Access **5**(99), 7495–7504 (2017)
17. Chitra, D.D., Rhymend, U.V.: Load balancing in cloud computing environment using improved weighted round robin algorithm for nonpreemptive dependent tasks. Scientific World J. **2016**, 1–14 (2016)
18. Junfeng, T., Weiping, Li.: Pheromone-based genetic algorithm adaptive selection algorithm in cloud storage. Int. J. Grid Distrib. Comput. **9**(6), 269–278 (2016)
19. Huang, C.: Analysis on application of wavelet neural network in wind electricity power prediction. Appl. Mech. Mater. **7**, 3764–3769 (2022)
20. Armani, V., Faticanti, F., Cretti, S., et al.: A Cost-Effective Workload Allocation Strategy for Cloud-Native Edge Services. (2021). https://doi.org/10.48550/arXiv.2110.12788
21. Pengden, Li., Xiaofan, Y.: On dynamic recovery of cloud storage system under advanced persistent threats. IEEE Access **7**, 103556–103569 (2019)
22. Wang, B., Lv, B., Song, Y.: A hybrid genetic algorithm with integer coding for task offloading in edge-cloud cooperative computing. IAENG Int. J. Comput. Sci. **49**(2), 503–510 (2022)
23. Jing, W., Zhiyuan, Y., Kuanching, Li., et al.: Local codes with cooperative repair in distributed storage of cyber-physical-social systems[J]. IEEE Access **8**, 38622–38632 (2020)
24. Mansouri, N., Javidi, M.M.: A review of data replication based on meta-heuristics approach in cloud computing and data grid. Soft Comput. **24**(11), 1–28 (2020)
25. Yan, W., Yuankai, G.: Forecasting method of stock market volatility in time series data based on mixed model of ARIMA and XGBoost. China Commun. **17**(3), 205–221 (2020)
26. Park, H., Lee, D., Moon, J.: LDPC code design for distributed storage: balancing repair bandwidth, reliability and storage overhead. IEEE Trans. Commun. **66**(2), 507–520 (2018)
27. Scheid, E.J., Rodrigues, B.B., Granville, L.Z., Enabling Dynamic, S.L.A., Smart, C.-B., Contracts, et al.: IFIP/IEEE symposium on integrated network and service management (IM). Arlington **2019**, 53–61 (2019)
28. Xu, X., Yang, C., Shao, J.: Data replica placement mechanism for open heterogeneous storage systems. Procedia Comput. Sci. **109**, 18–25 (2017)
29. Song, P.C., Pan, J.S., Chu, S.C.: A parallel compact cuckoo search algorithm for three-dimensional path planning. Appl. Soft Comput. **94**, 106443 (2022)
30. Saaty, T.L.: Decision making with the analytic hierarchy process. Int. J. Serv. Sci. **1**(1), 83–98 (2008)
31. Calheiros, R.N., Ranjan, R., Beloglazov, A., et al.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. Pract. Exp. **41**(1), 23–50 (2011)

32. Wang, R., Peng, L.: Topology-aware congestion control algorithm in data center network. J. Comput. Appl. **36**(9), 2357 (2016)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Qinlu He is an Associate Professor at Xi'an University of Architecture and Technology. He received Ph.D. degree in computer science from Northwestern Polytechnical University. His current research interests include data deduplication, cloud storage, and distributed file systems. He has more than 20 publications in journals and international conferences. He is a member of the IEEE and China Computer Federation.

Fan Zhang Department of Computer Sciences, Xi'an University of Architecture and Technology, Xi'an. Fan Zhang obtained her Ph.D. Degree from the Department of Computing, University of Surrey, in 2008, UK, and BS degree from the Northwestern Polytechnical University in 2001. Her current research interests include Machine learning, Data minng and Brain-computer interface.

Genqing Bian is currently pursuing the Ph.D. degree with the School of Management, Xi'an University of Architecture and Technology (XAUAT), Shaanxi, China. He is currently an Associate Professor with XAUAT. His research includes information security, cloud computing security, massive information storage technology, and VANETS security. He is a member of the China Computer Federation and the Association for Computing Machinery.

Weiqi Zhang received her Master. degree in Xi'an University of Architecture and Technology (XAUAT). Now he is an Associate professor with XAUAT. His current research interests include computer storage and cloud computing. he has more than 30 publications in journals and conferences in these areas and is a member of the China Computer Federation.

Zhen Li Senior Engineer of Shaanxi Institute of Metrology. Research direction: reliability test of electronic and electrical products, EMC electromagnetic compatibility, computer information network inspection and testing. More than 10 papers of various types have been published in journals and international conferences. Member of Intelligent Building and Building Automation Professional Committee of Shaanxi Institute of Automation.

Zhimin Yu now studying in the School of Information and Control Engineering of Xi'an University of Architecture and Technology with a master's degree, focuses on graphic neural networks.

Hao Feng Senior Engineer of SHAAN XI Big Data Group Co., Ltd. his main research interests include network resilience, complex system reliability, and big data techonolgy.