Future Generation Computer Systems 78 (2018) 31–40

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

DARS: A dynamic adaptive replica strategy under high load Cloud-P2P

ShengYao Sun [a,c,*], WenBin Yao [a], XiaoYong Li [b]

[a] Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing, China
[b] The Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China
[c] College of Software, Henan University, Kaifeng 475004, China

HIGHLIGHTS

- The paper presents a dynamic adaptive replica strategy under high load Cloud-P2P.
- The strategy obtains the replica creation opportune moment based on the node's overheating similarity.
- The strategy applies the fuzzy clustering analysis method to find optimal placement node.
- The node creates replicas by a decentralized self-adaptive manner.
- The strategy has been simulated and evaluated to demonstrate its effectiveness.

ARTICLE INFO

Article history:
Received 27 September 2016
Received in revised form 9 July 2017
Accepted 19 July 2017
Available online 6 August 2017

Keywords:
Cloud Storage
P2P
Replica strategy
Access latency
Load balance

ABSTRACT

In Cloud-P2P system, replica strategy is utilized for obtaining low access delay and high load balance, which requires paying more attention to the time of replica creation. Most of current replica strategies usually utilize the method of afterwards adjustment fixed threshold. However, these strategies may increase a large number of overload nodes and then lead to aggregate effect of being overloaded, especially under high load condition. To deal with the problem, this paper proposes a Dynamic Adaptive Replica Strategy (DARS) based on the node's overheating similarity. DARS addresses the problem of replica creation time, including obtaining the replica creation opportune moment and finding the optimal replica placement node, by a decentralized self-adaptive manner. Based on the overheating similarity, DARS applies fuzzy membership function to obtain the replica creation opportune moment, which enables the node to create replica before its overload. Meanwhile, DARS adopts the fuzzy clustering analysis method to find out the node with high node degree and low load which is used as optimal placement node to store replicas. Extensive experiments demonstrate that DARS obtains superior performances in access latency around 15%∼20% on average and better load balance than other similar methods under high load condition.

1. Introduction

Cloud storage is a new extension of cloud computing, providing data storage and transaction access service [1,2]. With the proliferation of cloud storage, as an intentional supplement, the P2P technology has been applied in cloud storage [3–10]. This new framework is called Cloud-P2P by a lot of scholars. The Cloud-P2P has attracted much attention, and has been used in many applications, e.g., IPTV [4], Video-on-Demand [11], etc.

In Cloud-P2P system, overloaded conditions are common during flash crowds. If a node receives many requests at a time, it can become overload and consequently cannot respond to the requests quickly, leading to the node overheating. Replica strategy is an effective method to deal with the problem of node overload by distributing load over replica nodes [12], which helps to achieve lower access delay and better load balance by reducing node response latency and accessing pressure [12,13]. Since the highly popular files could exhaust the bandwidth capacity of the node, leading to the service node overload. In order to reduce the access pressure of node, replica strategy can create replicas for these popular files, and store them on the specified nodes.

Nowadays, numerous replica strategies have been proposed [12–20]. A lot of researches show that the replica strategy generally needs to focus on the time of replica creation. The time

* Corresponding author at: Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing, China.
E-mail address: snowssy123@163.com (S. Sun).

http://dx.doi.org/10.1016/j.future.2017.07.046
0167-739X/© 2017 Published by Elsevier B.V.

of replica creation is a process, including obtaining the replica creation opportune moment and finding the replica placement node to store replica. The traditional strategies usually utilize method of **afterwards adjustment fixed threshold** to deal with this process. These strategies usually preset a fixed threshold which are based on a certain performance (e.g., bandwidth [13], node load [12], etc.). When the node reaches predefined conditions, a particular file stored on the node creates a replica and stores the replica to other node, which can reduce the node access pressure or release rate of bandwidth occupancy. Therefore, these strategies achieve low access delay and high load balance.

However, under high load condition, these traditional strategies, which still utilize the idea of traditional fixed threshold, may cause the following risks. First, they may lead to overload nodes increasing. In order to reduce the access pressure, these methods begin to create replica when the node reaches or exceeds the predefined threshold. But at this point, the node has been an overload node, which definitively increases the number of overload nodes. Consequently, the problem of obtaining the replica creation opportune moment is very important. Second, these strategies may cause **aggregate effect of being overloaded**. The node needs to create a replica and store the replica to other node to release its access pressure when it overloads. But under high load condition, most of nodes face the risk of overload at any time. In such an extreme case: the placement node is also about to reach the predefined threshold. It may overload when a replica is stored on it, leading to aggregate effect of being overloaded. Consequently, the problem of how to find the optimal node to store the replica also needs to be carefully considered.

To deal with these risks, this paper presents a Dynamic Adaptive Replica Strategy(DARS) based on the **node's overheating similarity**. This paper defines the overheating similarity, which is a probability that a node changes into an overloaded node. DARS realizes the ultimate goal of low access delay and high load balance under high load condition. Specifically, it has the following novel features to overcome the drawback of the traditional methods.

One novel feature of DARS is that it achieves the goal of reducing the number of overload nodes and relieving the aggregate effect of being overloaded under the high load condition. In order to achieve the goal, DARS focuses on the replica creation opportune moment and the selection of placement node. Based on the overheating similarity, it applies fuzzy membership function to obtain the opportune moment. DARS defines the membership function of overheating similarity according to node load, which designs a similarity range for relieving node overload. The similarity range is a probability range that a node changes into an overload node. When the node's overheating similarity reaches or exceeds the similarity range, it begins to create replica. In other words, before the node overload, it begins to create replica. In order to find optimal node, DARS combines the overheating similarity with the node degree as reference index, and uses the fuzzy clustering analysis method to select the node with high node degree and low overheating similarity as placement node, which can effectively decrease the probability of the placement node overload.

Another novel feature of DARS is that it conducts the operations in a decentralized self-adaptive manner without compromising access delay. Although the data center of Cloud-P2P can periodically monitors the files on the nodes, a centralized fashion cannot handle overload in time, especially under high load condition. Rather than depending on such a centralized method, DARS enables nodes themselves to decide whether to create or store replicas based on their load, which can effectively reduce risk of overload. The self-adaptive manner can handle overload in time and reduces the access delay, enhancing DARS scalability. Meanwhile, the manner relieves the data center burden.

The rest of the paper is structured as follows: Section 2 presents the related works. Section 3 presents DARS replica strategy with theoretical analysis. In Section 4, extensive experiments show that DARS obtains better performances than similar approaches within a variety of metrics, and analyzes of various factors are conducted. Finally, conclusion is summarized in Section 5.

2. Related work

The replica strategy become a hot research topic for many years, and has been widely used in many applications, e.g., P2P Network [21–23], Cloud storage [1,2,20,24], large data storage [25,26], the system of disaster recovery [14], fault tolerant technique [27,28], Data Grid [15] and distributed system [29], etc.

Nowadays, a large number of replica strategies have been proposed. Different replica strategies have different evaluation criterions, e.g., session state of data block [30], access frequency of replica [12,13], the security of data [17,27], etc.

In general, the replica strategies can be divided into static strategy and dynamic strategy. In static strategy, the number of replicas is fixed (e.g., GFS [31,32] and HDFS [25]). This strategy is so easy, and manages conveniently, but lacks flexibility. At present, most of the research focus on the dynamic strategy [12,13,15–20,30,33]. The number of replicas is variable in these strategies, and can be dynamically adjusted according to the requirements of the request. The way of replica placement is usually the random placement and order placement. Based on the system operation condition, these strategies store the replica on the query path by random way or order way when a node creates a new replica.

Paper [12] proposes an efficient and adaptive decentralized file replication algorithm(EAD). EAD selects the query traffic hubs and frequent requesters of a file as its replica nodes to guarantee hit rate while the minimum number of replicas. In EAD, base on request number of file, when the node's request number achieves node's access threshold of pre-set, the node begins to create new replica.

Paper [16] constructs a reliability model of replica service for cloud storage system. The model presents the method of data service reliability, trigger of replica creation and the storage node section according to the relationships among access reliability. This paper aims to the reliability of data service and the number of redundant replicas further decrease.

Paper [18] presents a file replication mechanism SWARM based on swarm intelligence. SWARM determines the placement of a file replica based on the accumulated query rates of nodes in a swarm. The nodes in a swarm, achieving fewer replicas and high querying efficiency, share replicas.

Paper [19] proposes the Selective Data replication mechanism in Distributed Data centers (SD^3). SD^3 aims to reduce inter-data center communications while still achieving low service latency. It considers update rates and visit rates to select user data for replication. Furthermore, SD^3 atomizes users' different types of data for replication, ensuring that a replica always reduces inter-data center communication.

There are other studies for file replication in the Cloud-P2P [4,5,7–11]. These works study the system performance such as successful queries and bandwidth consumption, focus on the node or file achieving a predefined condition. However, these works pay less attention to the problem that the node may have been overloaded when node or file achieves the predefined condition.

3. DARS replica strategy

In this section, we describe the various aspects of the DARS replica strategy in detail. We present DARS from the following aspects:

1. The relationship between the node load and the node state is discussed (Section 3.1).

2. The problem of how to obtain replica creation opportune moment is explained (Section 3.2).
3. The problem of how to get files set that needs to create new replica is discussed (Section 3.3).
4. The problem of how to find the optimal replica placement node is discussed. The problem includes how to reduce the probability of the placement node overload, and how to improve the probability of a replica being accessed (Section 3.4).
5. The various aspects are combined, forming the DARS replica strategy. These aspects include how to obtain the replica creation opportune moment, how to conduct the replication of popular files and how to find the optimal replica placement node (Section 3.5).

3.1. The relationship between the node load and the node state

3.1.1. The node load

Since a node becomes overload when it receives many requests during unit time [12]. We assume that the request is main reason that causes the node load increasing. Meanwhile, we assume that the request amount, received by the node during unit time, includes the access amount of local files and the forwarding amount of the node.

We assume that there are N storage layer nodes and M original files in Cloud-P2P. We use $\{HC_1, HC_2, \ldots, HC_N\}$ to denote these storage layer nodes, and use $\{D_1, D_2, \ldots, D_M\}$ to denote these files. We assume that total of m files are stored on the HC_i, denoted by $\{f_1, f_2, \ldots, f_m\}$, and define the access amount of f_i as the number of requests the f_i receives during T, denoted by f_i^c. The f_i^c expresses as $f_i^c = \sum f_i$. The access amount of each file can be measured at any time by node that the file stores.

Since the access amount of local is the access amount of all of its files, denoted by V_i^L. Therefore, during T, on HC_i, the access amount of local files can express as $V_i^L = \sum_{j=0}^{m} \sum f_j$.

We define the forwarding amount of the node as the number of requests the node forwarding during T, denoted by V_i^{Fw}. Since the node's request amount includes the access amount of local files and the forwarding amount of the node. The request amount of HC_i, denoted by V_i^{Now}, is sum of V_i^{Fw} and V_i^L during T, expressing as (1).

$$V_i^{Now} = V_i^{Fw} + V_i^L = V_i^{Fw} + \sum_{j=0}^{m} \sum f_j. \tag{1}$$

We use $\{V_1^{Max}, V_2^{Max}, \ldots, V_N^{Max}\}$ to denote the node's capacity represented by the number of queries it can respond during T, and use the ratio of node load to denote the fraction of HC_i capacity that is used, denoted by q_i. The ratio of node load can also represent the node load, and express as (2).

$$q_i = \frac{V_i^{Now}}{V_i^{Max}} = \frac{V_i^{Fw} + V_i^L}{V_i^{Max}} = \frac{V_i^{Fw} + \sum_{j=0}^{m} \sum f_j}{V_i^{Max}}. \tag{2}$$

3.1.2. The relationship between the node load and the node state

We assume that there exists thresholds θ and ϕ $(\theta, \phi \in [0, 1],\ \theta < \phi)$. When $q_i \geq \phi$, the node will stop service. When $q_i < \theta$, the node can provide normal service. At here, the ϕ is called overload threshold, which usually is a fixed threshold in traditional methods. The θ is called pro-hot threshold in this paper. Based on the node load, we use fuzzy membership function to describe the relationship between the node load and the node state. The membership function, is shown as (3), is determined by the assignment method [34].

$$B(q_i) = \begin{cases} 0, & q_i < \theta \\ \dfrac{q_i - \theta}{\phi - \theta}, & \theta \leq q_i \leq \phi \quad \theta, \phi \in [0, 1] \\ 1, & q_i > \phi. \end{cases} \tag{3}$$

We assume the discourse domain is the set of storage layer nodes. $B(q_i)$ is map function. Based on the fuzzy λ-cut set, we have the following proposition about the set of storage layer nodes.

Proposition 1. *Under the mapping of $B(q_i)$, the set of storage layer nodes can form fuzzy set $B(HC_i)$, which can be divided into three fuzzy subsets by B_θ and $B_\phi (\theta, \phi \in [0, 1])$. There are fuzzy subset of overheating $B(HC_i)_\phi$, fuzzy subset of pre-hot $B(HC_i)_\theta \bigcap B(HC_i)_\phi$, and fuzzy subsets of normal $\overline{B(HC_i)_\theta}$.*

Proof. According to the membership function of node state (3), the fuzzy set $B(HC_N)$ is one of classic sets for fuzzy λ-cut set. The characteristic function of members is

$$X_{B(HC_i)_\lambda} = \begin{cases} 1, & B(q_i) \geq \lambda \\ 0, & B(q_i) < \lambda. \end{cases} \tag{4}$$

When the output of membership function is ϕ, it means getting the node set of $B(q_i) > \phi$. According to the fuzzy λ-cut set, the node set is a fuzzy subset of overheating $B(HC_i)_\phi$. When the value is θ, it means getting the node set of $B(q_i) \geq \theta$. The node set is fuzzy subset $B(HC_i)_\theta$ made up of overheating fuzzy subset and pre-hot fuzzy subset. As a result, the fuzzy subset pre-hot is $B(HC_i)_\theta \bigcap B(HC_i)_\phi$. The complementary set of fuzzy subset $B(HC_i)_\theta$ is normal set, which is $\overline{B(HC_i)_\theta}$. The proposition is proved. □

Since the node's state is changing with the increasing of the request during unit time. Therefore, the node can be divided into three states: normal state, pre-hot state and overheating state according to the node load and Proposition 1. We aim to reduce the overload nodes amount. Therefore, if node's state is pro-hot state, it begins to create replica, which can effectively reduce the overload probability.

3.2. Obtaining the replica creation opportune moment

3.2.1. The overheating similarity and its membership function

If a node creates replica before overload, this can effectively reduce the probability of the node overload. However, a node cannot arbitrarily create replica when its state is pro-hot, which may cause the number of redundant replicas increasing. The node should create replica when its load closes to the overload. We use the overheating similarity to describe the proximity.

The node state from pre-hot to overheating is a gradual process. When the ratio of node load (i.e., q_i) is closer and closer to the overload threshold ϕ, the overload probability is gradually increased. In this paper, it (i.e., the proximity of q_i and ϕ) is called the node's overheating similarity, which describes the probability that a node becomes overload.

The node's overheating similarity can be described by the membership function of overheating similarity, which defines the proximity of the current state to the overload state. We determine it according to the node load, expressed as (5)

$$A(q_i) = \frac{1}{1 + \frac{1}{\beta}(q_i - \phi)^2} \times 100\% \quad \beta \in N, \theta \leq q_i \leq \phi. \tag{5}$$

The $A(q_i)$ $(0\% \leq A(q_i) \leq 100\%)$ is a probability that a node becomes overload, and is a percentage factor. The β can be dynamically adjusted according to the practical similarity.

3.2.2. Obtaining the replica creation opportune moment

In order to relieve overload, we design a similarity range for overheating similarity, expressed as

$$\alpha \leq A(q_i) \leq \varepsilon \quad (\alpha, \varepsilon \in [0\%, 100\%] \text{ and } \alpha < \varepsilon). \tag{6}$$

The inequality (6) represents the proximity range of $A(q_i)$ and ϕ. When the node's load reaches or exceeds the pre-defined similarity range, the node should create replica to reduce the node's overload probability. At here, the α and ε are percentage factors, and are the overheating similarity's floor and ceiling. For instance, $\alpha = 70\%$ *and* $\varepsilon = 90\%$. The node begins to create replica when the $A(q_i)$ reaches or exceeds [70%, 90%]. At this point, the node state is pro-hot state. The way of creating replicas before overload can effectively reduce the number of overload node, especially under high load condition.

We summarize the relationship among the request, the node load, the overheating similarity and the membership function of overheating similarity. A node receiving requests causes its load increasing. When receiving more requests, it may change into an overload node. In order to alleviate the occurrence of the situation, we define the overheating similarity. It is a probability that a node changes into an overload node, and is closely related to the node load. The overheating similarity can be described by the membership function of overheating similarity. Meanwhile, in order to obtain the replica creation opportune moment and reduce the probability of overload, we design a similarity range for overheating similarity. The similarity range is a probability range that a node changes into an overload node. It is designed to create replica when the overheating similarity reaches the predefined similarity range.

Each node can measure its overheating similarity $A(q_i)$ by itself at any time, and checks whether it reaches or exceeds (6) by the factor of $A(q_i)$. When $A(q_i)$ reaches the similarity range, it is the replica creation opportune moment.

3.3. Getting files set that needs to create new replica

Since the high popular files can attract more requests than low popular files, and can exhaust the node bandwidth capacity, resulting in the node overload. Consequently, the node release its access pressure by creating replica for popular files and storing the replica to other node when $A(q_i)$ reaches the pre-defined similarity range. Specifically, HC_i firstly orders its files in a descending order based on the access amount (i.e., f_i^c) when it begins to create replica, aiming to find the popular files, and get the descending array $Array_{Dec}^{HC_i}$, which represents the popularity rate of the files on the node HC_i. Then, the HC_i gets files one by one from the $Array_{Dec}^{HC_i}$, until

$$\sum f_i^c \geq V_i^{Now} \times \alpha - V_{i,T_i}^{Fw}. \tag{7}$$

The inequality (7) represents the amount of queries to be released by node. The node state is normal when it satisfies the inequality (7). These files, getting from $Array_{Dec}^{HC_i}$, are required to create replicas and form a set.

Furthermore, we need to explain the relationship among the node load, the similarity range and the amount of request. According to the (5) and (6), we have the inequality (8), which represents the relationship between the node load and the similarity range.

$$\left|\sqrt{\beta\left(\frac{1}{\alpha}-1\right)}\right| + \phi \geq q_i \geq \left|\sqrt{\beta\left(\frac{1}{\varepsilon}-1\right)}\right| + \phi. \tag{8}$$

According to (2) and (8), we have the inequality (9), which represents the relationship between the amount of request and the similarity range.

$$\left(\left|\sqrt{\beta\left(\frac{1}{\alpha}-1\right)}\right| + \phi\right) \times V_i^{Max} \geq V_i^{Fw} + \sum_{j=0}^{m} \sum f_j. \tag{9}$$

3.4. Finding the optimal placement node

3.4.1. The selection analysis of optimal node

When a node creates replicas, it needs to store these replicas on the most appropriate node. If the replica is stored on the most appropriate node, it can help to reduce the probability of node overload, and relieves the aggregate effect of being overloaded under the high load condition, and meanwhile can obtain low access delay and high load balance. The node is called the optimal placement node in this paper. It has many features. We should find it according to these features.

Under high load condition, since the node with high overheating similarity faces the risk of overload at any time. If a replica is stored on the node, it may become overload, resulting in high delay and low load balance. Consequently, we should select the node with low overheating similarity as placement node, try the best efforts to prevent the high overheating similarity nodes from being selected. Hence, the low overheating similarity is one of the important features of the optimal placement node.

However, the low overheating similarity node may also indicate the low probability of being accessed. Storing a replica on the node with low overheating similarity may cause high access delay. In other words, a replica should be easily accessed when it is stored on the optimal placement node. Consequently, another feature of the optimal placement node is how to increase the probability of a replica to be accessed.

In Cloud-P2P, a node may have many neighbor nodes. The different nodes have different node degrees. The node degree is the number of links that connect the node with their neighbor nodes in this paper. We assume all of nodes have the same service capacity. The following proposition can be established according to the node degree.

Proposition 2. *A neighbor node with the max node degree is selected as placement node when node stores its replicas to its neighbor nodes. If a replica is stored on this node, the probability of being accessed is higher than other neighbor nodes.*

Proof. We use Q^k to denote the request number for replica of $Array_{re}^{HC_i}[k]$ during T, and use q to denote the probability that the file request passes node Nd_k during T. The probability of successful access to the $Array_{re}^{HC_i}[k]$ is

$$P_n(r) = \binom{Q^k}{r} q^r (1-q)^{Q^k - r}. \tag{10}$$

The r indicates the times of successful access.

We assume that the neighbor nodes are constant when node stores its replicas to its neighbor nodes. Since all of the neighbor nodes have the same service capacity. Therefore, the q increases with the node degree increasing. According to (10), if a replica is stored on the neighbor node with high node degree, the probability of being accessed is higher than other replicas that are stored on the neighbor node with low node degree. The proposition is proved. □

According to Proposition 2, the probability of being accessed is higher than that on the node with low node degree when a replica is stored on the node with high node degree. Consequently, the second feature of the optimal placement node is high node degree.

We can select a node with low overheating similarity and high node degree as optimal placement node according to analysis of finding optimal node. If replica stores on the optimal node, the probability that the replica is accessed is relatively high, and the probability of node overload is low. In other words, the manner of finding the optimal placement node can reduce the access delay and improve load balance under high load condition.

Table 1
The evaluation index of initial candidate nodes.

Index of evaluation	The set of candidate nodes			
	HC_1	HC_2	...	HC_{cp}
The node degree	De_1	De_2	...	De_{cp}
The overheating similarity	$A(q_1)$	$A(q_2)$	...	$A(q_{cp})$

Table 2
The evaluation index of candidate nodes.

Index of evaluation	The set of candidate nodes				
	HC_1	HC_2	...	HC_{cp}	HC_{vr}
The node degree	De_1	De_2	...	De_{cp}	De_{vr}
The overheating similarity	$A(q_1)$	$A(q_2)$	...	$A(q_{cp})$	$A(q_{vr})$

3.4.2. The fuzzy set model of the optimal node

We combine the overheating similarity with the node degree, and use the fuzzy clustering analysis method to obtain the optimal placement node.

In order to obtain the optimal node, we build a fuzzy set model for nodes. The domain of discourse is the neighbor nodes of a node, denoted by $U = \{U_1, U_2, \dots, U_{cp}\}$. The node degree and the node's overheating similarity are combined as the index of evaluation. We use $\{(De_i, A(q_i))\}$ to denote the evaluation index of U_i. Consequently, the initial evaluation index follows as Table 1.

We assume that there is a virtual node, which has max node degree and min overheating similarity on the U, denoted by HC_{vr}. The evaluation index of HC_{vr} is (11):

$$\begin{aligned}&\{(De_{vr}, A(q_{vr}))|(Max\{De_1, De_2, \dots, De_{cp}\},\\&\quad Min\{A(q_1), A(q_2), \dots, A(q_{cp})\})\}.\end{aligned} \tag{11}$$

At this point, the domain of discourse is $U_{All} = U \bigcup \{HC_{vr}\} = \{U_1, U_2, \dots, U_{c_p}, HC_{vr}\}$. We have the following evaluation index Table 2.

We use $x_i = \{(x_{i1}, x_{i2})|(De_i, A(q_i))\}$ to denote these evaluation indexes. The similarity r_{ij} of U_i and U_j denotes to $r_{ij} = R\{U_i, U_j\}$. The r_{ij} is determined by following (12).

$$r_{ij} = \begin{cases} 1, & i = j \\ \dfrac{1}{M}\displaystyle\sum_{k=1}^{2} x_{ik} \times x_{jk}, & i \neq j. \end{cases} \tag{12}$$

At here, the $M = Max_{i \neq j}(\sum_{k=1}^{2} x_{ik} \times x_{jk})$. According to the fuzzy clustering analysis method and (12), we build the fuzzy similar matrix R for Table 2. The fuzzy similar matrix R is a $(cp+1)\times(cp+1)$ data matrix, which can be expressed as

$$\begin{pmatrix} 1 & r_{1,2} & \dots & r_{1,(cp+1)} \\ r_{2,1} & 1 & \dots & r_{2,(cp+1)} \\ \dots & \dots & 1 & \dots \\ r_{(cp+1),1} & r_{(cp+1),2} & \dots & 1 \end{pmatrix}.$$

3.4.3. The method of obtaining the optimal placement node

The HC_{vr} is undoubtedly the optimal placement node. However, it is a virtual node. In order to find the optimal placement node, we have the following proposition according to concept of fuzzy λ-cut set and the fuzzy similar matrix R.

Proposition 3. *Given a μ ($\mu \in (0, 1]$), the U_{All} can be classified many classifications by μ according to fuzzy similar matrix R. These classifications are determined by R_μ and belong to U_{All}. Given a $\lambda(\lambda \in (0, 1]$, and $\lambda < \mu)$, when getting a classification from these classifications, it must be a subclass that is determined by R_λ and belongs to U_{All} too. This means that R_λ is the classification by R_μ re-classification to U_{All} when $\lambda < \mu$.*

Proof. We assume that the $R_\lambda = (r_{ij}^{\lambda})_{c_p \times c_p}$, $R_\mu = (r_{ij}^{\mu})_{(c_p+1)\times(c_p+1)}$. Because of $\forall\lambda, \mu \in [0, 1], \lambda < \mu$, consequently, $r_{ij}^{\mu} = 1 \Leftrightarrow r_{ij} \geq \mu \Rightarrow r_{ij} \geq \lambda \Leftrightarrow (r_{ij}^{\lambda}) = 1$. Then, every classification which is determined by R_μ is one of subclassifications, and is determined by R_λ to U_{All}. □

The Proposition 4 can be established according to Proposition 3 and the U_{All}.

Proposition 4. *In the domain of discourse U_{All}, We divide the cluster by the method of fuzzy clustering analysis. When a node and the HC_{vr} are firstly partitioned into the same set, the node is the optimal placement node.*

Proof. Based on the theory of fuzzy set, if a node and HC_{vr} have the highest similarity, the node is real node, and is optimal replica placement node. Since the matrix R is a similarity matrix of U_{All}. According to fuzzy clustering analysis method and conclusion 3, the similarity of PR_x and HC_{vr} is the highest when a node PR_x and virtual node HC_{vr} are first partitioned into the same set. The PR_x is the optimal placement node. □

According to Proposition 4, we can find the optimal placement node from the domain of discourse U to store these newly created replicas.

3.5. The dynamic adaptive replica strategy

We combine various aspects, including obtaining the replica creation opportune moment, getting files set that needs to create new replica and finding the optimal replica placement node, forming the DARS.

In previous strategies, the data center maintains information of storage layer nodes to manage the replicas, and disseminates information about new replica sets. Considering that the nodes of Cloud-P2P face the risk of overload at any time under high load condition, DARS creates and stores replica in a decentralized manner. DARS enables nodes themselves to determine whether they should create or store replica according to their actual overheating similarity. Such decentralized adaptation helps to reduce the probability of overload under the high load condition. If a node has high overheating similarity, and has reached or exceeded the pre-defined similarity range (i.e., the probability of node overload reaches the predefined range), it begins to create replicas for popular files that are stored on the node. In other words, rather than creating a replica after the node overload, it begins to create replicas before node overload, which can effectively reduce the probability of overload. In order to store these replicas, DARS finds the optimal placement node from the neighbor nodes of the node. The optimal placement node has low node load and high node degree. The replica stores on the optimal placement node, which can effectively reduce the access delay and relieves the aggregate effect of being overloaded, meanwhile improves the probability of replica to be accessed and gets better load balance.

Specifically, DARS arranges each node to update its load by itself at any time, and gets its overheating similarity $A(q_i)$ according to its load. If node's $A(q_i)$ reaches (or exceeds) pre-defined similarity range, it begins to create replicas. The node begins to find the optimal node from the neighbor nodes, and stores these replicas on the optimal node. The algorithm 1 shows the pseudo code of DARS replica strategy.

4. Experiment simulation and analysis

This paper designs and implements a simulator for evaluating the DARS strategy based on *OptorSim* [35]. We compare the DARS's

Algorithm 1: DARS replica strategy

Input: HC_i
$//HC_i$ calculates its node load q_i by itself.
if $q_i \geq \theta$ **then**
  HC_i measures its similarity $A(q_i)$.
  if $A(q_i)$ *reaches* $[\alpha, \varepsilon)$ **then**
    Getting the descending order array $Array_{Dec}^{HC_i}$ by f_i^c.
    Getting files set $Array_{Cr}^{HC_i}$ that needs to create new replicas.
    if $Array_{Cr}^{HC_i}.Length > 0$ **then**
      $U_{cp} \leftarrow$ getting the neighbor node set.
      $A(q_i), De_i \leftarrow$ Getting overheating similarity and node degree.
      $A(q_{cp}) \leftarrow$ Getting U_{cp} similarity function value.
      $U_{All} \leftarrow$ Constructing virtual nodes HC_{vr}, merging U_{cp} and HC_{vr}.
      Establishing similar matrix $R \leftarrow$ based on (12).
      Partition R into sub matrix according to proposition (3).
      Getting the optimal placement node PR_x according to proposition (4).
      The replicas from the set $Array_{Cr}^{HC_i}$ are stored on PR_x.

Table 3
The simulated environment and parameters.

Parameters	Default values
File distribution	Random distribution
The number of nodes	256
The node bandwidth	Random value, 50M–150M
The max session	Random value, 100–500
The request amount	2000
The number of source files	1500, randomly assigned

performance with other two replica strategy in dynamic environments: SWARM [18] and DARS (Without Replicas). In following, we use WR to demote DARS (Without Replicas). WR does not provide the function of replica creation.

To be comparable, we use the same number of request operations in all of replication algorithms. In the experimental environment, the nodes are organized by the manner of metropolitan area network, which can reflect the real world. Meanwhile, we construct a data center to record all replicas and periodically monitor all of files, including the original files and their replicas. The file distribution utilizes a random manner. The requested files are randomly chosen. Table 3 lists the parameters of the simulation and their default values.

To verify the superior performances of DARS, we compare the ratio of overload node, the access delay and the load balance in different strategies. Low access delay is verified by operation time and the ratio of request response. High load balance is verified by the ratio of average load.

Since the performance, that we will compare, is closely related to the node overload threshold, the request amount and the similar range. Consequently, in each group of the performance comparing, we have compared two sets of experiments, which are under the same similar range and the different similar range, respectively. We set the request amount from 800 to 2000. Under the same similar range, we set the similarity range from 90% to 100%, the overload threshold is from 0.5 to 0.9. Under the different similar range, we set the overload threshold to 0.9, and respectively set the similarity is 50%–100%, 70%–100%, and 90%–100%.

4.1. *The ratio of overload node*

This experiment demonstrates that DARS can effectively reduce the number of overload nodes, and relieves the aggregate effect of being overloaded under high load condition. We use the ratio of overload nodes to represent the overload fraction of the system's total node number.

Fig. 1 demonstrates the ratio of overload nodes in different strategies under the same similar range. From Fig. 1, we can know that the ratio increases with the number of request increasing, and increases when the pre-defined overload threshold decreases. In addition, we can also observe that DARS generates the least ratio, SWARM has lower ratio than WR, and WR has the highest ratio. Compared with SWARM, DARS can reduce the ratio around 15% on average, and reduces around 20% on average than WR. These imply that DARS can effectively reduce the number of overload nodes than other similar strategies under the same similar range.

Fig. 2 shows the ratio of overload in different strategies under the different similar ranges. We can observe that the ratio of overload nodes increases with the pre-defined overload threshold reducing. When the similar range is fixed, the ratio increases with the request amount increasing. When the similar range is 70%–100%, the ratio of DARS reduces around 10% on average comparing with SWARM, and can reduce around 15% on average comparing with WR. We can also observe that the larger similarity range can realize a better replica distribution from Fig. 2.

Because of utilizing the method of afterwards adjustment fixed threshold, SWARM begins to create replica to relieve the access pressure when the nodes have overloaded, which resulting in more overload nodes. Unlike SWARM, DARS utilizes the advance adjustment method. It begins to create replica to reduce the node load when the node is about to overload. Consequently, DARS has the lowest ratio than other two strategies under the same similar range. In addition, the lower overload threshold represents the lower communication capacity. The node will be easily overload when the overload threshold is relatively small. As a result, the ratio of overload nodes increases with reducing of the overload threshold. DARS still has the lowest ratio when the overload threshold reduces. This implies that distribution of DARS is more reasonable under high load condition.

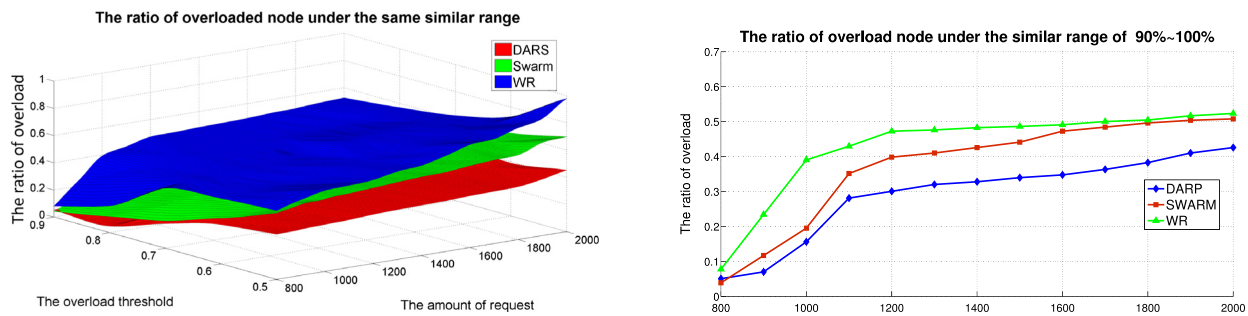
4.2. *The time of average operation*

This experiment is conducted to verify that DARS has lower average operation time under high load condition, which explains DARS can reduce access delay. For the purpose of comparison, we use $\sum_{i=0}^{m} RT_i/m$ to denote the average operation time, in which $\sum_{i=0}^{m} RT_i$ is sum of operation time for total m requests. If a request is not responded, we take the timeout as operation time.

Fig. 3 illustrates the changing of operation time in different strategies under the same similar range. From Fig. 3, we can know that the average operation time increases with the number of request operations increasing, and increases with the reducing of the overload threshold. DARS can decrease operation time around 10% on average than SWARM and reduce around 15% on average than WR. The lower average operation time represents the lower access delay, which implies that DARS can effectively decrease access delay.

Fig. 4 demonstrates the average operation time in different strategies under the different similar ranges. We can know that the average time of decreasing is closely related to similar range. When the similar range is 90%–100%, the operation time of DARS can decline around 15% on average than SWARM, and decline around 20% on average than WR. When the similar range is 70%–100%, it can decrease around 22% on average than SWARM, and decrease around 27% on average than WR. Consequently, the low similarity range can be more effective in reducing the operation time.

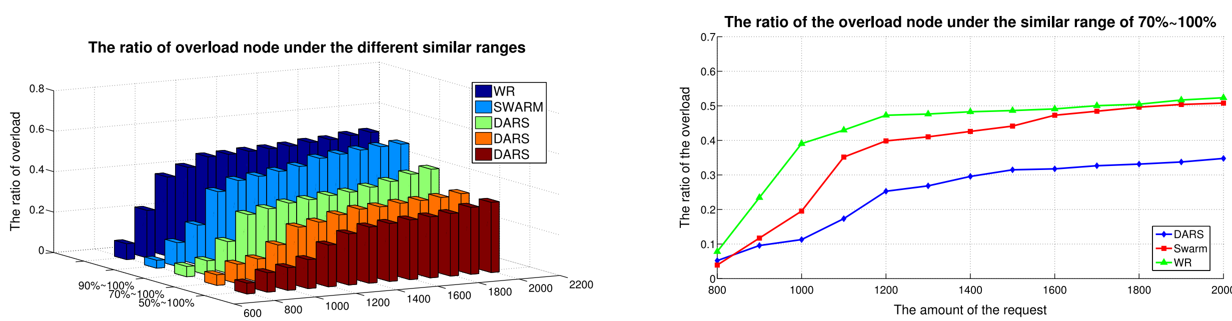
In conclusion, whether under the same or different similar range, DARS has the lowest operation time. It is because the



(a) Under the different overload threshold.

(b) The overload threshold is 90%.

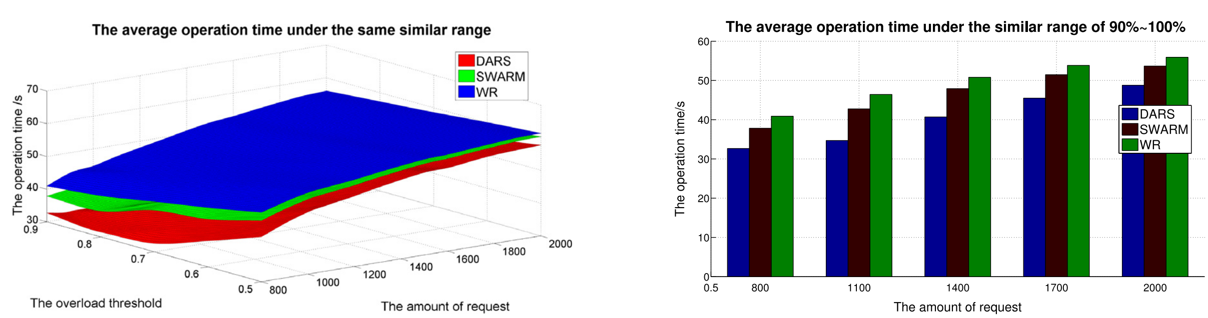
Fig. 1. The ratio of overload node. (The similar range is 90%–100%.)



(a) Under the different similar ranges.

(b) The overload threshold is 90%.

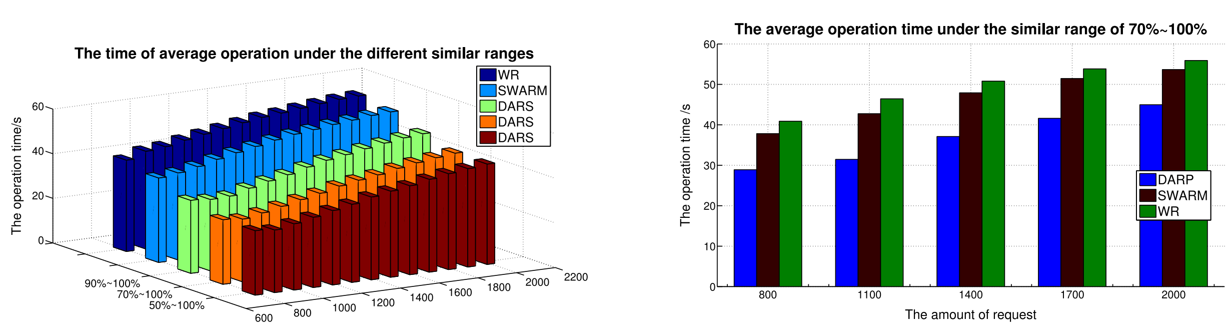
Fig. 2. The ratio of overload node. (The overload threshold is 90%.)



(a) Under the different overload threshold.

(b) The overload threshold is 90%.

Fig. 3. The average operation time. (The similar range is from 90% to 100%.)

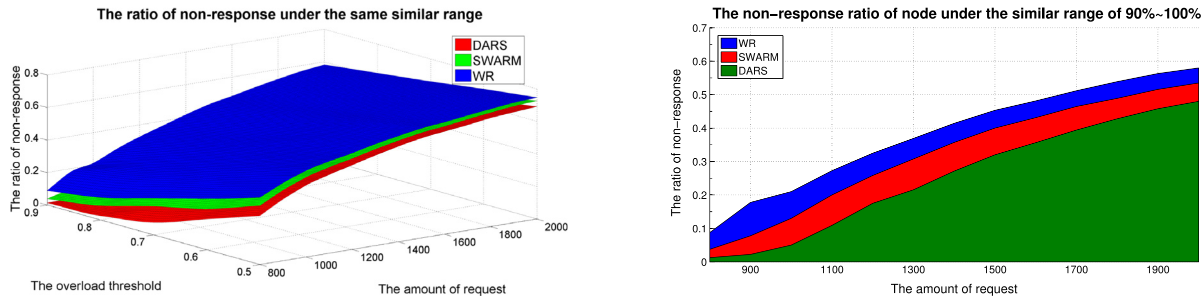


(a) Under the different similar ranges.

(b) The overload threshold is 90%.

Fig. 4. The average operation time. (The overload threshold is 90%.)

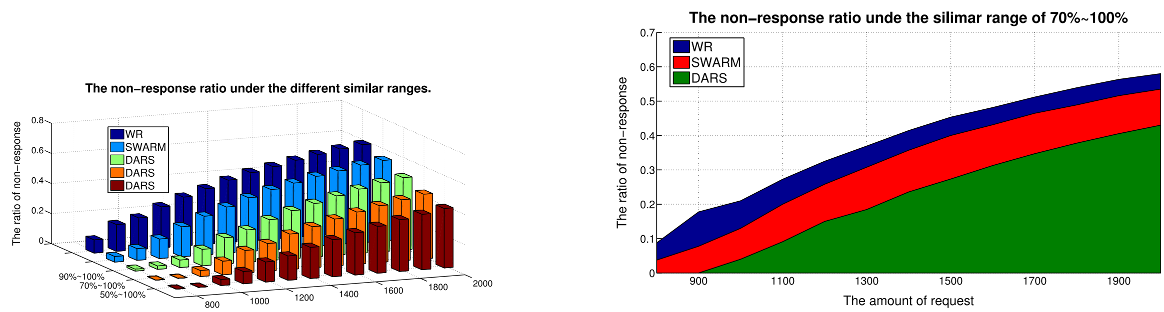
obtaining replica creation opportune moment and the finding optimal replica placement node are all trying to reduce the operation time in DARS. Moreover, the placement node has relatively high node degree, which can improve the probability of replica being accessed and reduce the access path. Consequently, the operation time of DARS is reduced, achieving lower access delay.



(a) Under the different overload threshold.

(b) The overload threshold is 90%.

Fig. 5. The non-response ratio of node. (The similar range is from 90% to 100%.)



(a) Under the different similar ranges.

(b) The overload threshold is 90%.

Fig. 6. The non-response ratio of node. (The overloaded threshold is 90%.)

4.3. The ratio of request response

This experiment demonstrates the ratio of request response to verify that DARS has lowest access delay than other two strategies. The ratio of response request equals Ob_c/Num_{sum}. At here, the Ob_c is the amount of the request, and the Num_{sum} is the amount of total request. The non-response ratio equals $1 - Ob_c/Num_{sum}$.

Fig. 5 demonstrates the non-response ratio of nodes in each replication strategies under the same similar range. We can observe that the non-response ratio increases with increasing of the amount of request, and increases with reducing of the overload threshold. The non-response ratio of DARS declines around 5% on average than SWARM, and reduces around 10% on average than WR.

Fig. 6 shows the non-response ratio of nodes in different replication strategies under the different similar ranges. We can know that the non-response ratio is closely related to similar range. When the similar range is 90%–100%, DARS can reduce around 9% on average than SWARM, and decline around 13% on average than WR. Compared with SWARM and WR, DARS respectively decreased around 10% and 15% on average when the similar range is 70%–100%. This implies that the low similarity range can be more effective in reducing the non-response ratio.

Since SWARM utilizes a centralized method, it periodically monitors the node. When a node has been overload, the system begins to adjust the replicas to reduce the node's access pressure. Unlike SWARM, DARS uses a decentralized self-adaptive manner to adjust the replicas, which enables nodes themselves to determine whether they should create replicas according to approximation relationship of node load. Consequently, DARS has the lowest non-response ratio than other two strategies. This implies that DARS can decrease the access delay under high load condition.

4.4. The ratio of average node load

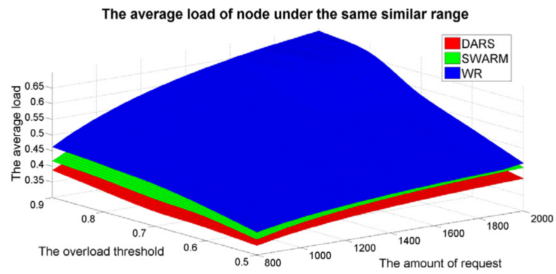
This experiment demonstrates the load balance among replica nodes in each replication strategy. In order to compare the load balance, we define a ratio of average node load, which equals $\sum_{j=1}^{N} q_i/N$, in which the $\sum_{j=1}^{N} q_i$ is sum of the all of nodes' load ratio. Fig. 7 illustrates the changing of average load ratio in different strategies under the same similar range. It shows that the average load ratio increases with the increasing of the amount of request, and increases with the increasing of the overload threshold. Compared with other two strategies, DARS has the least ratio. It decreases around 8% on average than WR, and declines around 5% on average than SWARM.

Fig. 8 shows the average load ratio in different replication strategies under the different similar ranges. We can know that DARS has the least ratio, SWARM has lower ratio than WR, and WR has the highest ratio. Meanwhile, the average load ratio is also closely related to similar range. When the similar range is 90%–100%, compared with SWARM and WR, DARS can reduce the average load around 6% and 10%, respectively. It can reduce around 12% and 19% than SWARM and WR when the similar range is 50%–100%.

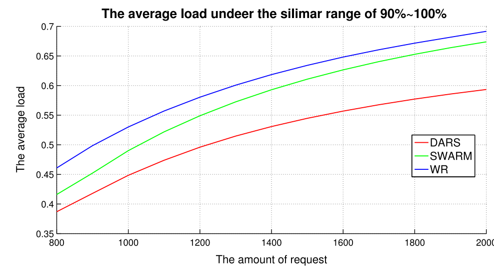
Recall that higher overload threshold causes the fewer replicas, and can cause longer access path. Consequently, the ratio increases with increasing of the overload threshold. Meanwhile, DARS selects the node with high node degree and low load as placement node to store replicas. It takes into account node available load during file replication, and takes into account the probability of nodes being accessed. Unlike SWARM, it bases on swarm intelligence, to identify node swarms with common node interests and close proximity, and utilizes collective node's behaviors to find placement node store replicas. Under the high load condition, the swarm node may be a high load node, which may lead to load imbalance. Consequently, DARS has the least ratio than other two strategies. This implies that DARS can keep node with light load under high load condition.

5. Conclusion

In Cloud-P2P system, replica strategy can significantly reduce the access delay and obtains better load balance. The traditional

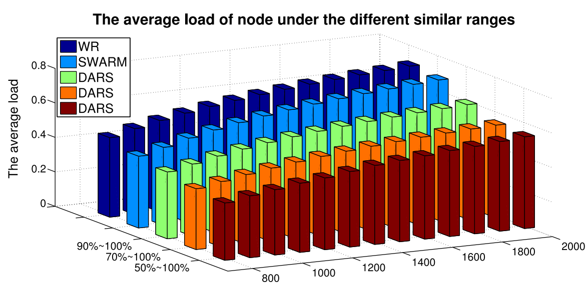


(a) Under the different overload threshold.

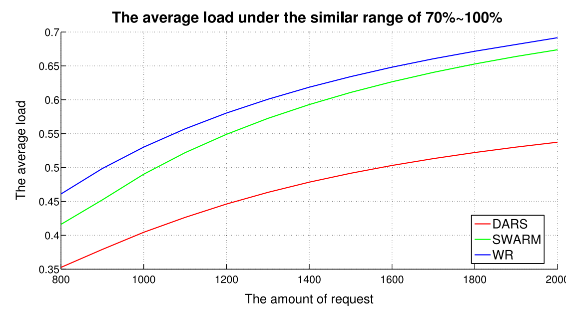


(b) The overload threshold is 90%.

Fig. 7. The average load of node. (The similar range is from 90% to 100%.)



(a) Under the different similar ranges.



(b) The overload threshold is 90%.

Fig. 8. The average load of node. (The overloaded threshold is 90%.)

strategies usually utilize the method of afterwards adjustment fixed threshold to obtain low access delay and high load balance. However, these strategies may increase a large number of overload nodes, leading to bad service performance, especially under high load condition.

This paper proposes a DARS replica strategy that pays attention to the time of replica creation to guarantee the reducing of the number of overload nodes. In DARS, the obtaining replica creation opportune moment and the finding optimal replica placement node are disposed by a decentralized self-adaptive manner. DARS defines the membership function of overheating similarity based on node's overheating similarity, in which a similar range is designed for relieving overload. When the overheating similarity reaches or exceeds the pre-defined similar range, the node begins to create replica. DARS combines the overheating similarity and the node degree as reference index, and uses the fuzzy clustering analysis method to select the node with high node degree and low overheating similarity as placement node.

Extensive experiments demonstrate the superiority of DARS in comparison with other replica strategies. It dramatically reduces the number of overload nodes and produces significant improvements in probability of replica access, achieving low access delay and high load balance.

In the future work, we will further study how to delete the redundant replica. In Cloud-P2P system, more replicas can lead to lower access delay but more maintenance overhead and vice versa. A challenge for a replica strategy is how to minimize replicas while low access latency and high load balance. We also plan to further explore adaptive methods to fully exploit node load for efficient replica consistency maintenance.

Acknowledgments

The author is grateful to the anonymous reviewers for their valuable comments and suggestions. This work was partly supported by the National Nature Science Foundation of China (NSFC) (61370069, 61672111), the NSFC-Guangdong Joint Found (U1501254) and the Co-construction Program with the Beijing Municipal Commission of Education, the Fundamental Research Funds for the Central Universities (BUPT2011RCZJ16) and China Information Security Special Fund (NDRC).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.future.2017.07.046.

References

[1] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, Privacy-preserving public auditing for secure cloud storage, IEEE Trans. Comput. 2009 (2) (2013) 579.
[2] Y. Zhang, S.Q. Ren, S.B. Chen, B. Tan, Differcloudstor: Differentiated quality of service for cloud storage, 2012, pp. 1–9.
[3] L. Savu, Cloud computing: Deployment models, delivery models, risks and research challenges, 2011, pp. 1–4.
[4] J. Song, H.J. Deng, J.L. You, Nova:a p2p-cloud vod system for iptv with collaborative pre-deployment module based on recommendation scheme, Adv. Mater. Res. 756–759 (2013) 1566–1570.
[5] J. Chakareski, Cost and profit driven cloud-p2p interaction, Peer-to-Peer Netw. Appl. 8 (2) (2015) 244–259.
[6] O. Babaoglu, M. Marzolla, M. Tamburini, Design and implementation of a p2p cloud system, 2012, pp. 412–417.
[7] S.F. Jin, C.F. Wang, L.L. Chen, et al., Modeling and analysis of Cloud-P2P storage architecture, J. Commun. 36 (3) (2015).
[8] H. Kavalionak, A. Montresor, P2p and cloud: a marriage of convenience for replica management, 2012, pp. 60–71.
[9] Z. Li, T. Zhang, Y. Huang, Z.L. Zhang, Y. Dai, Maximizing the bandwidth multiplier effect for hybrid cloud-p2p content distribution, 2012, pp. 1–9.
[10] C. Wu, B. Li, S. Zhao, Multi-channel live p2p streaming: Refocusing on servers, in: The Conference on Computer Communications, INFOCOM 2008, IEEE, 2008, pp. 1355–1363.
[11] V. Rocha, F. Kon, R. Cobe, R. Wassermann, A hybrid cloud-p2p architecture for multimedia information retrieval on vod services, Computing 98 (1) (2016) 73–92.
[12] H. Shen, Ead: An efficient and adaptive decentralized file replication algorithm in p2p file sharing systems, IEEE Trans. Parallel Distrib. Syst. 21 (6) (2010) 827–840.

[13] X. Sun, Q.Z. Li, P. Zhao, K.X. Wang, F. Pan, An optimized replica distribution method for peer-to-peer network, Chinese J. Comput. 37 (6) (2014).
[14] Q. Lian, W. Chen, Z. Zhang, On the impact of replica placement to the reliability of distributed brick storage systems, 2005, pp. 187–196.
[15] D. Nukarapu, B. Tang, L. Wang, S. Lu, Data replication in data intensive scientific applications with performance guarantee, IEEE Trans. Parallel Distrib. Syst. 22 (8) (2010) 1299–1306.
[16] C.Q. Huang, Y. Li, H.Y. Wu, Y. Tang, X. Luo, Modeling and maintaining the reliability of data replica service in cloud storage system, J. Commun. 36 (3) (2014).
[17] M. Tu, P. Li, I.L. Yen, B.M. Thuraisingham, L. Khan, Secure data objects replication in data grid, IEEE Trans. Dependable Secure Comput. 7 (1) (2008) 50–64.
[18] H. Shen, G.P. Liu, H. Chandler, Swarm intelligence based file replication and consistency maintenance in structured p2p file sharing systems, IEEE Trans. Comput. 64 (10) (2015) 2953–2967.
[19] G. Liu, H. Shen, H. Chandler, Selective data replication for online social networks with distributed datacenters, IEEE Trans. Parallel Distrib. Syst. 27 (8) (2016) 2377–2393.
[20] N.K. Gill, S. Singh, A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers, Future Gener. Comput. Syst. 65 (2016) 10–32.
[21] F. Furfaro, G.M. Mazzeo, A. Pugliese, Managing multidimensional historical aggregate data in unstructured p2p networks, IEEE Trans. Knowl. Data Eng. 22 (9) (2010) 1313–1330.
[22] E. Cohen, S. Shenker, Replication strategies in unstructured peer-to-peer networks, ACM SIGCOMM Comput. Commun. Rev. 32 (4) (2002) 177–190.
[23] M. Roussopoulos, M. Baker, Cup: Controlled update propagation in peer-to-peer networks, Comput. Sci. 21 (6) (2002) 1–6.
[24] G. Ananthanarayanan, S. Agarwal, S. Kandula, A. Greenberg, I. Stoica, D. Harlan, E. Harris, Scarlett:coping with skewed content popularity in mapreduce clusters, 2011, pp. 287–300.
[25] The apache software foundation. hadoop. http://hadoop.apache.org/core/2009.
[26] Amazon-s3. amazon simple storage service (amazon s3), http://www.amazon.com/s.2009.
[27] L. Wu, B. Liu, W. Lin, A dynamic data fault-tolerance mechanism for cloud storage, 2013, pp. 95–99.
[28] J.P. Walters, V. Chaudhary, Replication-based fault tolerance for mpi applications, IEEE Trans. Parallel Distrib. Syst. 20 (7) (2009) 997–1010.
[29] B. Wah, File placement on distributed computer systems, Computer 17 (1) (1984) 23–32.
[30] Q. Wei, B. Veeravalli, B. Gong, L. Zeng, D. Feng, Cdrm: A cost-effective dynamic replication management scheme for cloud storage cluster, 2010, pp. 188–196.
[31] Google's colossus makes search real-time by dumping mapreduce. http://highscalability.com/blog/2010/9/11/googles-colossus-makessearch-real-time-by-dumping-mapreduce.html.
[32] S. Ghemawat, H. Gobioff, S. Leung, File and storage systems: The google file system, in: Acm Symposium on Operating Systems Principles Bolton Landing, Vol. 37, 2003, pp. 29–43.
[33] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, P. Keleher, Adaptive replication in peer-to-peer systems, 2004, pp. 360–369.
[34] G.J. Klir, B. Yuan, Fuzzy sets and fuzzy logic - theory and applications, 1995, pp. 283–287.
[35] G. Belalem, Y. Slimani, Consistency management for data grid in optorsim simulator, 2007, pp. 554–560.

ShengYao Sun male, received the B.S. degree in computer science and technology 2005 and the M.S. degree in network engineering at Henan University 2009, he is a lecturer in Henan University, and is currently working toward the Ph.D. degree in the department of Computer of Beijing University of Posts and Telecommunications, his research interests includes disaster recovery, cloud storage, distributed computing, etc.

WenBin Yao male, Ph.D., he is currently a professor in Beijing University of Posts and Telecommunications, is a member of Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, his research interests includes disaster recovery, fault-tolerant computing, trusted computing, system reliability evaluation, etc.

XiaoYong Li male, Ph.D., he is currently an associate professor in Beijing University of Posts and Telecommunications, is a member of the Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, his research interests includes Distributed computing, pretreatment of Network data analysis, trusted computing, network security, etc.