



Replication management in peer-to-peer cloud storage systems

Ali Majed¹ · Fatemeh Raji¹ · Ali Miri²

Received: 9 July 2020 / Revised: 28 June 2021 / Accepted: 17 August 2021 / Published online: 14 September 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

Data availability represents one of the primary functionalities of any cloud storage system since it ensures uninterrupted access to data. A common solution used by service providers that increase data availability and improve cloud performance is data replication. In this paper, we present a dynamic data replication strategy that is based on a hybrid peer-to-peer cloud architecture. Our proposed strategy selects the most popular data for replication. To determine the proper nodes for storing popular data, we employ not only the feature specifications of storage nodes, but also the relevant structural positions in the cloud network. Our simulation results show the impact of using features such as data popularity, and structural characteristics in improving network performance and balancing the storage nodes, and reducing user response time.

Keywords P2P cloud storage systems · Data replication · Data availability · Load balancing · Response time

1 Introduction

The massive increase in data generation in recent years has led to the need for cost-effective solutions for data storage systems [1]. An overriding requirement of such systems is that timely access to data is guaranteed [2]. Lack of access to data can have severe consequences. For example, in medical emergency applications, patients' vital information is to be available to physicians whenever required [3]. In other activities such as research collaboration, it is desired that researchers share and use their findings wherever needed [4].

Cloud computing offers on-demand virtualized resources, such as computing nodes and storage spaces, to users who are billed on a fee-for-service model with negotiated service quality. In general, service quality criteria are expressed in Service Level Agreements (SLAs) between cloud providers and users. A typical highest priority item in such SLAs is data availability. This item deals with data access assurance, and is measured as the probability of data

availability [5]. The second most common item in typical SLAs is related to system response time.

Cloud providers may have to pay penalties if they fail to provide users with the negotiated data availability. They may also face penalties for poor network performance such as high response times to users' requests. As a result, it is important to improve the quality of cloud services to promote both user satisfaction and cloud profitability.

In distributed storage systems, each node can handle at most a limited number of requests simultaneously. If a request is sent to a node with a high workload, the request may be delayed until the previous requests are completed [6]. Well-designed strategies distribute users' requests among storage nodes to decrease users' response times. In this paper, we consider the response time to be the interval between the request initiation and the time the node receives the response to that request [8]. One such strategy is to use well-balanced placement of replicated data on the storage nodes. This usual results in balanced network node workloads, which in turn improves network performance [7].

Data replication strategies aim to improve data availability by creating identical copies of data, and placing them in distributed geographical locations. In this way, requested data can be accessed from any data replica [1, 2].

Data replication strategies have to address four main challenges [9]:

✉ Fatemeh Raji
f.raji@eng.ui.ac.ir

¹ Department of Computer Science, University of Isfahan, Isfahan, Iran

² Department of Computer Science, Ryerson University, Toronto, Ontario, Canada

1. When is the appropriate time to replicate data? If the number of requests for data access increases prior to the creation of data replicas, this can result in increased response times. Thus, the replication process must be done before network performance degrades.
2. Which data need to be replicated? Replicating data which are not frequently accessed has no positive impact on network performance and will waste storage space. Although it is recommended that all data have at least one replica as backup to avoid data loss due to failure of storage nodes, data having high access rates should be targeted for the replication process.

Selecting data for replication is mostly based on prediction and popularity techniques. In the prediction-based methods, the data that are forecast to be accessed by a majority of users in the near future using locality properties are replicated. In the popularity-based methods, data that had the highest access rates in the past are replicated [10].

3. How many replicas should be created? The number of copies used to reduce response time should be balanced with the need to minimize the required storage space.
4. Which nodes should be selected to store the replicated data? The nodes used to store replicated data are mostly chosen based on various characteristics such as available storage space and distance from users.

Depending on how they address the challenges listed, data replication strategies can be grouped into two main categories: (1) *Static* replication strategies are those that executed the same way regardless of changes in network conditions. Despite their ease of implementation, static strategies are rarely used in dynamic architectures such as cloud storage systems; (2) *Dynamic* replication strategies consider network conditions such as the status of storage nodes. Despite their higher computational costs, dynamic strategies increase network performance more than static ones [11].

Note that replication strategies can replicate data fully or partially. While all the data are replicated in full-replication strategies, partial-replication strategies only focus on high demand data to reduce related storage costs [10].

Recently, Peer-to-Peer (P2P) architecture has been considered an attractive alternative for cloud storage systems. In this type of architecture, nodes of the system are made up of independent nodes connected via the Internet. P2P cloud storage systems can offer advantages over those of traditional client–server architectures. For example, if a node fails to respond to a user’s request, other nodes holding the same data can respond to that request. However, a well-positioned node in a P2P structure may fail to serve any requests due to such features as low processing

capacity. On the other hand, a node may be hard to access if it is not located in a proper structural position in the network, even if it has optimal feature characteristic. As a result, a replication strategy in a P2P storage system must consider both the features and structural status of nodes.

In this paper, we propose a full dynamic popularity-based replication strategy for P2P cloud storage systems that increases data availability and improves network performance in load balancing and response time. Our strategy aims to select the most popular data for replication and determine the proper nodes for storing the popular data. Additionally, the proposed strategy employs both feature and structural status to select the nodes for storing the popular data.

The outline of this paper is as follows: A literature survey is given in Sect. 2, followed by our proposed methodology in Sect. 3. Simulation results are presented in Sect. 4, and conclusions and future work are discussed in Sect. 5.

2 Literature Review

In P2P architecture, the nodes i.e. peers, may act as independent nodes if the peer nodes are built from users’ own computer systems in such a way that each node can act as a server or client [12, 13].

To model the P2P architecture, undirected connected graphs are considered. We denote the P2P graph as $G(P, E)$, where P is a non-empty set of graph nodes and E is a set of links connecting the nodes [14].

In contrast to client–server architecture, there is no single point of failure in P2P structures. This is because all nodes can act as a server. A P2P network can also be easily scaled up by the addition of a new peer node with low cost [15, 16]. Additionally, the small-world phenomenon applies to P2P networks, which means the data can be transferred across the network in a reasonable time. In general, in the networks having small-world behavior, the average of the highest distance between two nodes (d) can be estimated using the following equation:

$$\langle d \rangle \approx \frac{\ln N}{\ln \langle k \rangle} \quad (1)$$

where, N is the number of nodes in the network and k is the average node degree.

Despite the advantages mentioned, P2P architecture suffers from some drawbacks. It mostly requires a complex management system, since the network is made up of different computer systems. Furthermore, each node can freely and randomly join or leave the network at any time, which negatively impacts data availability [17].

P2P structure can be classified into four categories [18] which are unstructured, structured, super-peer and hybrid P2P networks. In the following, the replication strategies of different P2P architectures are discussed in detail.

2.1 Replication strategies in unstructured P2P networks

In unstructured P2P networks, the nodes do not have the information related to the geographical locations of other nodes. In this way, searching for a piece of data is done by continuous forwarding of search messages from one node to its neighbors. As a result, successful hits for specific data are not guaranteed since predefined mappings between data and their locations are not provided [3, 11].

In [19], a zone-based data replication is introduced while the whole network is divided into sub-zones based on the number of nodes in the network. It is assumed that each node has the information about the number of access requests for its local data. In this way, the popularity of the local data is calculated and broadcasted to the network. Nevertheless, the access time and the size of the data are not considered. Furthermore, as the nodes have no global information about the network, data may be replicated in different sub-zones redundantly. Finally, neither the feature nor the structural status of nodes is considered for storing the replicated data.

In [20], a data replication strategy for utilization of unused storage space is discussed. When the replication process starts, every node gathers the status of each node from the network to find the history of data transferred to other nodes. In this way, data are not replicated only in a group of well-configured nodes. The nodes having enough space to store the replicated data are selected. However, the structural score of each node is not taken into account in this strategy.

2.2 Replication strategies in structured P2P networks

In the structured P2P networks, the mappings between the data and their locations are controlled using key-value pairs, kept in Distributed Hash Tables (DHT). DHTs can then be utilized to locate the data faster.

In [21], a replication strategy called CORP is suggested to decrease the load on busy nodes. CORP replicates the data that have high access rates. On the other hand, the data that have frequent updates are replicated as low as possible to minimize the cost of ensuring consistency among replicas. In the creation phase, CORP considers both the data popularity and frequency of update on the data. In the placement phase, the created replicated data are placed on nodes with highest load capacities and closest distance to

replicas that store the same data. As a result, CORP strategy reduces the cost of frequent updates. Nevertheless, storing the replicated data in the same geographical regions makes the data vulnerable to failures such as natural disasters. This shortcoming reduces the data availability and influences network performance especially for the requests originated from different geographical areas.

In [22], a distributed data replication model is presented. The replicated data is placed on the nodes, where the lowest delay for data access is obtained. Additionally, this model uses balanced-tree data structure to search for replicated data. Nevertheless, it is assumed that no changes are happened in the network which is not realistic due to the churning nature of cloud network.

2.3 Replication strategies in super-peer P2P networks

In the super-peer P2P architecture, the nodes are categorized into groups which are managed by super-peer. The super-peers are the nodes having more processing power, bandwidth, and storage spaces compared to others. Each super-peer keeps the group information such as the nodes characteristics and the data stored in them. In this way, the super-peers are able to manage search requests initiated by the nodes within the groups [3].

In [23], an interest-based data replication strategy named SWARM is introduced. In SWARM, the nodes are grouped into colonies using users' common interests to share data among them. Each group of nodes are assigned to a super-node to deal with the replication process. However, SWARM does not consider the dynamic characteristic of cloud storage system due to its low dynamicity.

In [24], a data replication strategy is introduced to prepare a predefined level for data availability and minimize the response time. The network nodes are grouped into virtual clusters. Stronger nodes, in terms of processing power and storage capacity, called CH, are selected to manage each cluster. Initially, each CH computes the local data availability which means the ratio of the number of replicas in the cluster to the cluster size. Afterwards, global data availability is calculated using communications among CHs. Then, the data with lower availability are considered as rare data, and selected for replication. However, this model assumes the nodes have no limitations in their storage spaces. Moreover, the structural status of the nodes is not considered in this model.

2.4 Replication strategies in hybrid P2P networks

Unstructured P2P networks are rarely implemented in a fully-distributed manner. In contrast, a combination of peer-to-peer and client–server models called the hybrid P2P networks are used. Hybrid models employ centralized servers to monitor accessing to data. For example, in P2P structure of Skype, when a node searches data, it sends message to the centralized servers [12]. Then, the servers identify the nodes stored the requested data. Finally, the initial nodes receive the requested data by direct communication with the storage nodes [3, 18].

In [25], a model named DROPS is studied. DROPS aims to increase the network performance and improve the security of users' data. In DROPS, the users' data are divided into blocks to increase data security. Nevertheless, fragmentation reduces data availability since all storage nodes holding any blocks of data must be available when needed. To select the proper node for storing the popular data, the centrality characteristic related to the structural status of node is applied. However, only one replicated data for each fragmentation is created which makes data availability at risk.

In [26], a distributed strategy named DARS, is suggested for data replication. Each node keeps the access rate of its stored data. Before the node workload surpasses the threshold, the data with the highest requests is replicated using fuzzy logic. Then, the replicated data is placed on the neighbor nodes with a lower workload and higher degree. However, the replicas aggregation in one region decreases the data availability. This is because, the localization strategy of DARS increases the access time for users in different geographical areas.

Table 1 illustrates a comparison of the discussed data replication strategies in the P2P architecture based on the main criteria.

3 The proposed replication strategy

In this section, we propose a dynamic data replication strategy using a hybrid P2P architecture for cloud storage system. In the proposed strategy, any request for accessing a piece of data will be carried out by super-peer nodes. In this way, the specification of the replicas is sent to the user. The super-peers are aware of the information about each data such as its size, number of replicas, and the access rate in each period of time. As a result, the super-peers are able to calculate the popularity changes for each data. Note that, any changes on data popularity and the scores related to the candidate nodes are performed at the start of a fixed period in the network activity.

Our proposed strategy consists of two main parts:

1. *Determining the most popular data for replication:* The most popular data are replicated to reduce the response time to access the data and distribute the workload among storage nodes. Additionally, any replicated data that are no longer in interest of different users is removed to free the storage spaces and use it efficiently.
2. *Determining the proper nodes for storing the most popular data:* The best nodes are selected according to the feature specifications and the positional status of all storage nodes in the network.

In the following, each part of the proposed strategy is discussed in detail.

Table 1 Comparison of data replication strategies in P2P architecture, that and indicate the parameter is considered and not considered respectively

Reference	P2P architecture type	Load balancing	Response time	Availability	Node status	
					Feature status	Structural status
[19]	Unstructured	X	X	✓	X	X
[20]	Unstructured	✓	X	✓	✓	X
[21]	Structured	X	✓	✓	X	✓
[22]	Structured	X	✓	✓	✓	X
[23]	Super-peer	✓	X	✓	✓	X
[24]	Super-peer	✓	✓	✓	✓	X
[25]	Hybrid	X	✓	✓	X	✓
[26]	Hybrid	✓	X	✓	✓	✓

3.1 Determining the most popular data for replication

Generally, different data do not have the same interest to users. In fact, in practice only a small subset of data is of interest to a large subset of users. Thus, it is important to calculate the popularity of each data and replicate only the data that are of more interest to the users. On the other hand, the pattern of access requests for a particular data will not remain constant over the time. For example, analyzing Google searches on two different programming languages, the Perl and the Python, from 2004 to 2020 shows that these two programming languages have different access ratios. Perl programming language search was a dominant trend in 2004–2006, but this trend \ lost its attraction afterwards. In this way, we can consider the data related to the Perl programming language as “cool” data. This means these data were only attractive in the past but no longer. Given that most of the recent searches are now focused on Python (compared to Perl), we can consider the data related to Python programming language as “hot” data.

It is important to identify both the hot and cool data in the replication strategy. This is because, we can replicate the hot data to increase the network performance, and remove the replicated data related to the cool data to free the storage space without downgrading the performance.

The number of access requests for a piece of data affects its popularity directly. However, the access rate is also related to the total number of requests sent in the network. Using the access rate, a more accurate perspective can be obtained for identifying the popularity of data. On the other hand, the quantity of used storage space is vital in the P2P networks. Thus, we must determine the size of each data before replication. In this way, we consider the smallest stored data in the network.

Figure 1 is the algorithms of the proposed strategy for determining the most popular for replication. The popularity change of the i^{th} data in the n^{th} period is calculated in Eq. (2).

$$\Delta P^n(f_i) = \begin{cases} + \left(\frac{NR^n(f_i)}{TNR^n} * \frac{\min(\text{Size}(f))}{\text{Size}(f_i)} \right), \\ - \left(\alpha * \left(\frac{TNR^n - 1}{TNR^n} \right) \right), \\ \text{Otherwise} \end{cases} \quad (2)$$

where,

- $NR_n(f_i)$ is the number of access requests for the data file i in the n^{th} period;
- TNR_n is the total number of access requests in the n^{th} period for all the data in the network;

- $\text{Size}(f_i)$ is the size of the data file i .
- $\min(\text{Size}(f))$ is the smallest data file in the network;
- Parameter α , which ranges from 0 to 1.

In our proposed strategy, if the popularity change of a data file exceeds 1, the data file is considered as “hot”. Note that, the number of replications for a piece of data is determined based on the integer part of the popularity changes. On the other hand, when the popularity of a data file is less than -1 , the data file is determined as “cool”. In this way, one of its replicated data is eliminated randomly provided that the availability of a cool data file does not become less than the desired threshold for data availability.

A minimum replica number for each data file must be determined and kept unchanged. This number has a direct correlation with data availability and users’ desired availability for the data files. In this way, the minimum required replicas for each data are calculated in Eq. (3):

$$R_{min}(f_i) = \lceil \frac{1}{1 - Av_{th}} * (P(F) + H_{off}) \rceil \quad (3)$$

where,

- Av_{th} is the desired threshold for data availability which is a decimal number;
- H_{off} is the offline hours of the storage nodes which is related to the churn behavior of peers in the P2P structure;
- $P(F)$ is the average failure probability of the storage nodes, calculated in Eq. (4):

$$P(F) = \frac{\sum_{i=1}^N (P(F_i))}{N} \quad (4)$$

where,

- o $P(F_i)$ is the failure probability of the storage node i ;
- o N is the number of nodes in the network.

Note that, the average lifetime of storage nodes or any hardware is determined by their factories. Thus, the failure probability for the i^{th} storage node in a year is calculated in Eq. (5):

$$P(F_i) = AFR_i = 1 - \exp\left(\frac{-8760}{MTTF_i}\right) \quad (5)$$

where,

- AFR_i is the annual failure rate.
- $MTTF_i$ is the average failure time.
- 8760 is the number of hours per year [2].

The offline hours, expressed as H_{off} in Eq. (3), is the offline hours of storage nodes in 24 h. H_{off} is calculated for the node i in Eq. (6):

Fig. 1 Determining the most popular data for replication

```

1- In each time interval  $T$  do
2- set  $P \leftarrow$  data popularity set initialized with 0s
3-   for each data file  $f_j$  do
4-      $P[j] \leftarrow P[j] +$  calculate  $\Delta P^T(f_j)$ 
5-   end
6-   sort  $P$  ascendingly
7-   for each data file  $j$  in  $P$  do
8-     if  $P[j] > 0$ 
9-       while  $P[j] > 0$ 
10-        replicate the data file  $f_j$ 
11-         $P[j] \leftarrow P[j] - 1$ 
12-      end
13-    end
14-    else
15-      if the number of replicated data file  $f_j > R_{min}(f_j)$ 
16-        delete data file  $f_j$  from Node  $N_i$ 
17-         $P[j] \leftarrow P[j] + 1$ 
18-      else
19-         $P[j] \leftarrow 0$ 
20-      end
21-    end
22-  end

```

$$H_{off} = \frac{\sum_{i=1}^N \binom{H_{off}^i}{24}}{N} \quad (6)$$

If the number of replicas for a cool data becomes less than the number calculated in Eq. (3), the replicated data related to the cool data is not deleted. In this condition, the popularity of the cool data is set to 0.

3.2 Determining the proper nodes for storing the most popular data

It is important to place the created replicated data of the most popular data on proper nodes. In our proposed strategy, the proper nodes are determined using scores calculations for each node which are based on two different scores as the following:

1. The first score is calculated using the nodes feature specifications called *feature score*. The feature

specifications include nodes workload and their available storage space.

2. The second score called *structural score*, consider the position of storage node in the network. The structural score also informs other storage nodes in the network and the links between them. In other words, the positional status of all storage nodes effects the structural score of each node.

The total score of storage nodes is computed based on the feature and structural scores. This calculation is demonstrated in Eq. (7):

$$ScTotal(n) = [\theta * ScStructural(n)] + [(1 - \theta) * ScFeature(n)] \quad (7)$$

where,

- Parameter θ , called impact factor, determines the significance of the feature and structural scores. If its value gets close to 0, it indicates the priority of the

feature score versus the structural score. On the other hand, the structural score of the storage nodes can be prioritized over the feature score by assigning its value close to 1.

Figure 2 is the algorithms of the proposed strategy for determining the proper nodes for storing the most popular data. In every time interval of the network activity, all storage nodes calculate their scores. Each node has all the information needed for the computation of the feature score. Moreover, the structural score of each storage node is computed using super-peer(s) support. Note that, super-peer(s) has the global information of the P2P network. Finally, when the feature and.

structural scores of the storage nodes are calculated, the total scores can be obtained. In the following, each score is elaborated in detail.

3.2.1 Feature score

In every time interval, the feature score of the storage nodes is calculated using the feature specifications which is given in Eq. (8):

$$ScFeature(n) = \beta * \left(\frac{c_n * a_n}{\lambda_n * \tau_n} \right) \quad (8)$$

where,

- β indicates the importance of feature score versus structural score;
- λ_n presents the access ratio received by the node;
- τ_n shows the average response time of the node for the received access requests;
- c_n specifies the number of requests that the node can respond simultaneously. In general, each node can respond to a limited number of simultaneous requests;
- a_n presents the ratio of the node activities in the current time interval. a_n is important in the P2P architecture

since a storage node can get offline and inaccessible at some points.

3.2.2 Structural score

The position of the storage nodes and how they connect with other nodes determine the structural scores. Note that, storage nodes are connected via an undirected link in an internet-based network which indicates the communications of the nodes in both directions. Moreover, the distance between two nodes affects their communications quality. Since shorter connection between two nodes improves network performance, we consider the reverse of link length for the weight of each link.

On the other hand, the centrality metric demonstrates the importance of nodes according to their structure status in the network graph [27, 28]. If a storage node has a higher centrality, it has prominent impact on the network. However, the type of centrality metric is based on the network topology. In this way, the evidential centrality is one of the centrality metrics that can be applied to weighted undirected graphs [29] which is accordance to our P2P graph. The evidential centrality is based on Dempster-Shafer theory. In this theory, each problem consists of a set of hypotheses as shown in Eq. (9):

$$U = \{H_1, H_2\} \quad (9)$$

The power set of the problem set, stated in Eq. (9), includes all the possible states of the hypotheses which is named as frame of discernment [29] which are as the following:

1. No hypothesis occurs.
2. If H_1 occurs
3. If H_2 occurs
4. Both the H_1 and H_2 occur

Fig. 2 Determining the proper nodes for storing the most popular data

-
1. In each time interval I
 2. set $T \leftarrow$ total score of all nodes, initialized with 0s
 3. **for each** node n in the network
 4. calculate $ScStructural(n)$
 5. calculate $ScFeature(n)$
 6. calculate $ScTotal(n)$
 7. **if** $T[n] - ScTotal(n) > Score_{th}$
 8. $T[n] \leftarrow ScTotal(n)$
 9. **end**
 10. sort T ascendingly
-

The possible hypotheses are demonstrated in Eq. (10):

$$2^U = \{\emptyset, \{H_1\}, \{H_2\}, \{H_1, H_2\}\} \quad (10)$$

Generally, the probability mass function assigns the value from 0 to 1 for each member of the power set. The assigned value indicates the occurrence probability of each member [29]. Furthermore, the mass value for the null member is zero. Finally, the sum of the assigned mass values is equal to 1. The characteristics of mass function are presented in Eq. (11):

$$m : 2^U \rightarrow [0, 1], m(\emptyset) = 0, \sum_{A \in 2^U} m(A) = 1 \quad (11)$$

In the proposed strategy, we employed centrality metric to calculate the structural score. Based on the Dempster-Shafer theory, the structural score in our strategy is defined using the two problems related to degree and weight. The number of links connected to each node is used in the degree problem. For the weight problem, the sum of the weights of links connected to each node is considered.

Similar to Eq. (9), the hypotheses l and h for the degree problem is illustrated in Eq. (12):

$$di = \{l, h\} \quad (12)$$

where,

- The hypothesis l indicates the probability that the node i is low-effective in the network based on its degree;
- The hypothesis h show the probability that the node i is high-effective in the network based on its degree.

Similar to Eq. (10), the power set of the degree problem set for the node i is shown in Eq. (13):

$$2^{di} = \{\emptyset, \{l\}, \{h\}, \{l, h\}\} \quad (13)$$

The probability mass function assigns a value between 0 and 1 (inclusively) to each member of the power set. In this way, the max–min normalization is used which is displayed in Eq. (14) [30]:

$$A_{normal} = \frac{|A - \min(S)|}{\max(S) - \min(S)} * (U - L) + L \quad (14)$$

where,

- S is the set of numbers;
- A is a member of the set S ;
- U is the upper bound of the range;
- L is the lower bound of the range;
- $\max(S)$ is the maximum value in the set S ;
- $\min(S)$ is the minimum value in the set S .

The lower and upper bounds of the mass function are 0 and 1, respectively. When all members have equal value, the denominator of min–max normalization gets 0. To

avoid this condition, a small positive value as ε is added to denominator. Hence, the modified form of normalization function is illustrated in Eq. (15):

$$A_{normal} = \frac{|A - \min(S)|}{\max(S) - \min(S) + \varepsilon} \quad (15)$$

To determine the probability that member A is a high-effective member in the set, the value of member A is compared to the minimum value of the set members. In this way, the probability that storage node i with the hypothesis h is a high-effective node according to its degree can be calculated using Eq. (16):

$$m_{di}(h) = \frac{|k_i - k_m|}{k_M - k_m + \varepsilon} \quad (16)$$

where,

- k_i is the degree of node i ;
- k_m and k_M are the lowest and the highest degrees of the nodes in the network respectively.

This is also shown in Eq. (17) and Eq. (18):

$$k_m = \min\{k_1, k_2, \dots, k_N\} \quad (17)$$

$$k_M = \max\{k_1, k_2, \dots, k_N\} \quad (18)$$

A member is compared to the maximum value in the set to detect the probability of its low-effectiveness which is illustrated in Eq. (19):

$$A_{normal} = \frac{|A - \max(S)|}{\max(S) - \min(S) + \varepsilon} \quad (19)$$

The probability that a storage node is a low-effective node using its degree is calculated similarly. The value of the hypothesis l in the degree problem is obtained in Eq. (20):

$$m_{di}(l) = \frac{|k_i - k_M|}{k_M - k_m + \varepsilon} \quad (20)$$

Since the total value of the mass function for a power set should be equal to 1, the value of $m_{di}(l, h)$ can be calculated using Eq. (21):

$$m_{di}(l, h) = 1 - [m_{di}(l) + m_{di}(h)] \quad (21)$$

Additionally, the weight problem of the node i can be defined using Eq. (22):

$$wi = \{l, h\} \quad (22)$$

where,

- The hypothesis l shows the probability that node i is low-effective in the network based on its weight;
- The hypothesis h indicates the probability that the node i is high-effective in the network based on its weight.

Moreover, the problem power set for the weight problem is presented in Eq. (23):

$$2^{w_i} = \{\emptyset, \{l\}, \{h\}, \{l, h\}\} \tag{23}$$

The probability that node i is a high-impact node according to its weight is calculated according to Eq. (15). Specifically, the value of hypothesis h for the weight problem is obtained in Eq. (24):

$$m_{w_i}(h) = \frac{|w_i - w_m|}{w_M - w_m + \varepsilon} \tag{24}$$

where,

- w_i is the weight of the node i ;

w_m and w_M are the minimum and maximum weights of the nodes in the network as shown in Eq. (25) and (26):

$$w_m = \min\{w_1, w_2, \dots, w_N\} \tag{25}$$

$$w_M = \max\{w_1, w_2, \dots, w_N\} \tag{26}$$

The probability that a storage node such as node i has a low-effect in the network based on its weigh is calculated in Eq. (27):

$$m_{w_i}(l) = \frac{|w_i - w_M|}{w_M - w_m + \varepsilon} \tag{27}$$

Using Eq. (11), the total value of the mass function in each power set problem should be equal to 1. Hence, the value of $m_{w_i}(l, h)$ is calculated in Eq. (28):

$$m_{w_i}(l, h) = 1 - [m_{w_i}(l) + m_{w_i}(h)] \tag{28}$$

The probability that a node is high-effective in the network is obtained by combining the hypotheses h for both degree and weight problems. This is done by applying the Dempster rule of combination theory given in Eq. (29). The Dempster rule of combination for our hypotheses is also shown in Table 2.

$$m_i(h) = \frac{1}{1 - k} \sum_{A=m_i(h)} m_{d_i}(h).m_{w_i}(h), k = \sum_{A=\emptyset} m_{d_i}(h).m_{w_i}(h), A = m_{d_i}(h) \cap m_{w_i}(h) \tag{29}$$

The probability that the node i is low-effective in the network, is calculated by unifying the hypotheses l for both degree and weight problems as is shown in Eq. (30):

$$m_i(l) = \frac{1}{1 - k} \sum_{B=m_i(l)} m_{d_i}(l).m_{w_i}(l), k = \sum_{B=\emptyset} m_{d_i}(l).m_{w_i}(l), B = m_{d_i}(l) \cap m_{w_i}(l) \tag{30}$$

Finally, the structural score of the storage node i is calculated based on the evidential centrality given in Eq. (31):

$$ScStructural(i) = m_i(h) - m_i(l) \tag{31}$$

In the following, the structural score in a sample network, illustrated in Fig. 3) is calculated using the proposed strategy.

As discussed, the minimum and maximum degree of this sample network are achieved using Eq. (17) and Eq. (18) respectively:

$$k_m = \min\{1, 2, 3\} = 1$$

$$k_M = \max\{1, 2, 3\} = 3$$

The probability that the storage node A is a high-effective node in the network based on its degree is calculated using Eq. (16).

$$m_{d_A}(h) = \frac{|1 - 1|}{3 - 1 + 0.01} = 0$$

Using Eq. (20), the probability that the node A is a valuable storage node according to its degree is shown in the following:

$$m_{d_A}(l) = \frac{|1 - 3|}{3 - 1 + 0.01} = \frac{2}{2.01}$$

Hence, the value of $m_{d_A}(l, h)$ according to Eq. (21) is illustrated below:

$$m_{d_A}(l, h) = 1 - \left[0 + \frac{2}{2.01}\right] = \frac{0.01}{2.01}$$

Table 2 The probability mass functions based on the Dempster rule of combination

m_{d_i} m_{w_i}	$\{l, h\} = 1 - [\{l\} + \{h\}]$	$\{h\}$	$\{l\}$
$\{l\}$	$\{l\} \cap \{l\} = \{l\}$ $m_{d_i}(l) * m_{w_i}(l)$	$\{h\} \cap \{l\} = \emptyset$ $m_{d_i}(h) * m_{w_i}(l)$	$\{l, h\} \cap \{l\} = \{l\}$ $m_{d_i}(l, h) * m_{w_i}(l)$
$\{h\}$	$\{l\} \cap \{h\} = \emptyset$ $m_{d_i}(l) * m_{w_i}(h)$	$\{l\} \cap \{h\} = \emptyset$ $m_{d_i}(l) * m_{w_i}(h)$	$\{l, h\} \cap \{h\} = \{h\}$ $m_{d_i}(l, h) * m_{w_i}(h)$
$\{l, h\}$	$\{l\} \cap \{l, h\} = \{l\}$ $m_d(l) * m_{w_i}(l, h)$	$\{h\} \cap \{l, h\} = \{h\}$ $m_{d_i}(h) * m_{w_i}(l, h)$	$\{l, h\} \cap \{l, h\} = \{l, h\}$ $m_{d_i}(l, h) * m_{w_i}(l, h)$

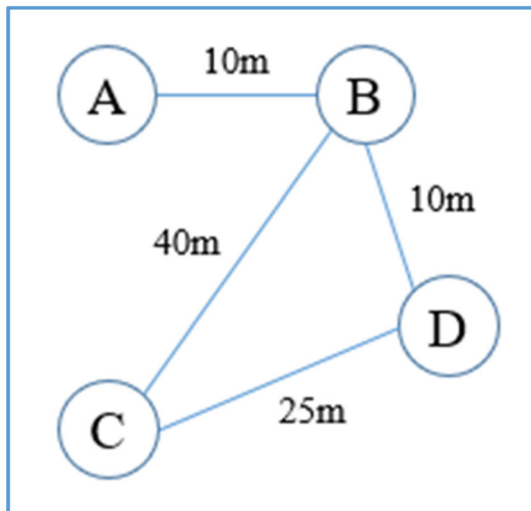


Fig. 3 A sample network

In the proposed strategy, the weights of nodes are specified as: $AB = \frac{1}{10}$, $BC = \frac{1}{40}$, $BD = \frac{1}{10}$, and $CD = \frac{1}{25}$.

Furthermore, the normalized links weights are $AB = 4$, $BC = 1$, $BD = 4$, and $CD = 1.6$. Using Eq. (25) and Eq. (26), the minimum and maximum weights are obtained as the following:

$$w_m = \min\{2.6, 4, 5.6, 9\} = 2.6$$

$$w_M = \max\{2.6, 4, 5.6, 9\} = 9$$

Using Eq. (24), the probability that node A is a high-effective node in the network based on its weight is computed as what is below:

$$m_{wA}(h) = \frac{|4 - 2.6|}{9 - 2.6 + 0.01} = \frac{1.4}{6.41}$$

Using Eq. (27), the probability that the storage node A does not have structural impact according to its weight is calculated as the following:

$$m_{wA}(l) = \frac{|4 - 9|}{9 - 2.6 + 0.01} = \frac{5}{6.41}$$

$$m_{wA}(l, h) = 1 - \left[\frac{1.4}{6.41} + \frac{5}{6.41} \right] = \frac{0.01}{6.41}$$

Using Dempster rule of combination given the Eq. (29), Eq. (30), and Table 2, we obtain $m_A(h) = 0.0014$ and $m_A(l) = 0.9983$. Consequently, the structural score of node A is obtained as what is below:

$$ScStructural(A) = 0.0014 - 0.9983 = -0.9969$$

This negative value shows that the structural status of storage node A is unsuitable in the network. In other words, accessing the data stored in node A is not affordable for users. Similarly, the structural score of node B is 1.0015. Thus, the node B is an important node in the network

which is aligned with Fig. 3 that the node B acts as a hub in the network. Using Eq. (7) and Eq. (8), the total score of node A and B is demonstrated in Table 3.

As shown in Table 3, if the parameter θ gets close to 0, the total score of node A increases to more than node B score. This is because, the feature score of node A is higher than node B. On the other hand, by increasing the parameter θ , the importance of node B compared to node A is proved.

In our proposed strategy, the parameter θ controls the impact of feature or structural scores of storage nodes. Hence, the best value for the parameter θ can be adjusted flexibly according to the network conditions.

4 Model evaluation

To evaluate the proposed strategy, we have used the CloudSim simulation toolkit. Table 4 shows the configuration specifications of the virtual machines used in the simulation.

The configuration specifications of the storage nodes are given in Table 5.

In CloudSim, users' access requests are sent as cloudlets in the network. The configuration specifications of cloudlets are shown in Table 6.

The number of data files is randomly applied in this simulation with its size ranging from 1 to 1024 MB. To simulate the users' access rate for each data, the normal distribution function is used.

We compared our proposed strategy with DROPS [25], since DROPS is based on hybrid P2P architecture. Additionally, the random placement strategy is used for comparison. This strategy places the created replicated data randomly on the storage nodes [22].

Figure 4 compares the average number of replicas versus the time interval for the proposed strategy, DROPS [25]

Table 3 The scores of nodes A and B related to the network shown in Fig. 3

Storage node	A	B
Feature score	1.7	1.1
Structural score	- 0.9969	1.0015
Total score ($\theta = 0.1$)	1.4303	1.0901
Total score ($\theta = 0.2$)	1.1606	1.0803
Total score ($\theta = 0.3$)	0.8909	1.0704
Total score ($\theta = 0.5$)	0.3515	1.0507
Total score ($\theta = 0.9$)	- 0.7272	1.0113

Table 4 The specifications of virtual machines

Processing elements (PE)	1–2
PE capacity (MIPS)	1000–1500
RAM (MB)	512–1024
Bandwidth (bit/s)	1000–4000
Virtual machine monitor	Xen

Table 5 The specifications of storage nodes

Storage capacity (GB)	15–25
Maximum transfer rate (MB/s)	100–150
Latency (s)	3–5
Annualized failure rate (%)	3–5

Table 6 The specifications of cloudlets

Cloudlet length (million instructions)	1000–5000
Cloudlet size (byte)	100–400
Processing elements count	1–2

As shown in Fig. 4, the consumption of the storage spaces in the proposed strategy are more than the other models.

This is because, the proposed strategy considers the users’ offline hours and the nodes failure probability. This accommodates with the real condition since it is related to the churn behavior of the P2P networks’ users. Additionally, DROPS [25] and random placement strategy generate a fixed number of replicas for all the data which is not appropriate for the dynamic nature of cloud network.

For the proposed strategy, the average number of replicas compared to the file removal rate i.e., the parameter α in Eq. (2), is illustrated in Fig. 5.

Note that, PF notation denotes the failure probability of storage nodes in Eq. (4) and H_{off} is the average ratio of the node offline hours per day as described in Eq. (6). The sum of these two parameters shows the unavailability probability of the storage nodes. In other words, the number of replicas can be examined compared to the unavailability probability of nodes using Fig. 5.

As shown in Fig. 5, using a constant value for parameter α , when the unavailability of storage nodes increases, the average number of replicas increases.

For example, if the parameter α is 0.2 and the sum of PF and H_{off} are 0.1 and 0.5, the average number of replicas of the data files are 4.4 and 11.9, respectively. Moreover, by increasing the parameter α and keeping a constant value for

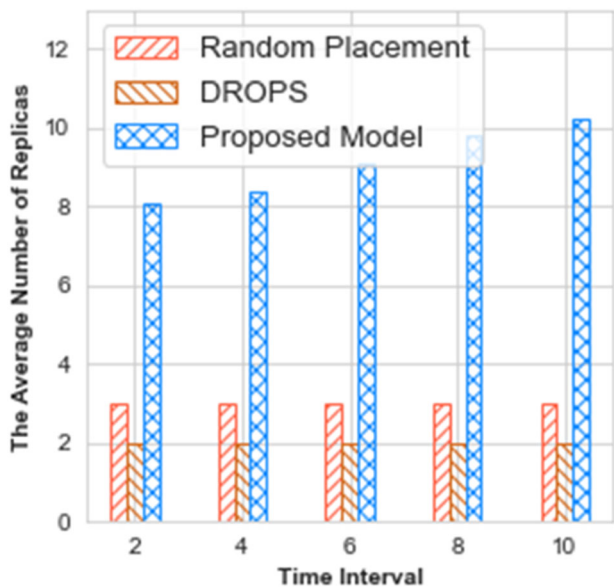


Fig. 4 The average number of replicas versus the network activity in the proposed strategy compared to DROPS [25] and random placement strategy

and random placement strategy. Note that, the unavailability probability of the storage nodes in the proposed strategy is set to 0.3.

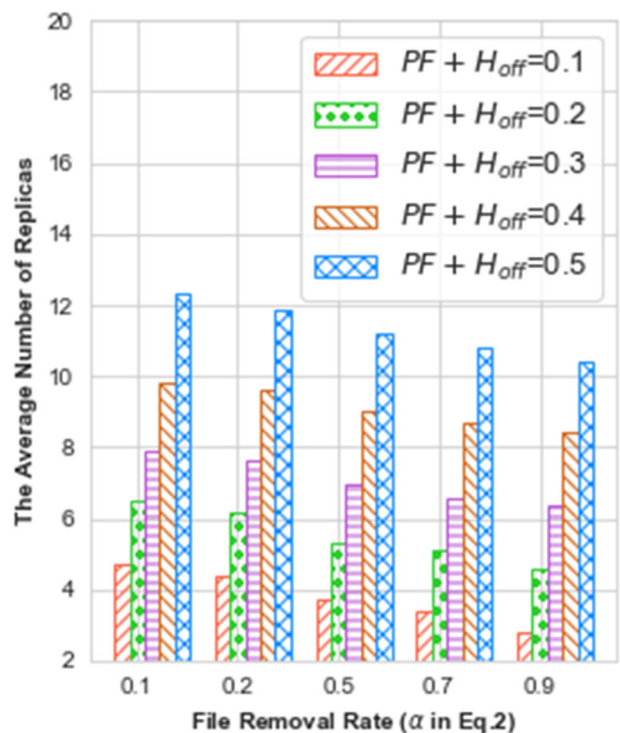


Fig. 5 The average number of replicas versus the file removal rate (α in Eq. (2)) in the proposed strategy

the node’s unavailability probability, the number of replicas reduces. For instance, if the parameter α is 0.1 and 0.9 and the sum of PF and H_{off} are equal to 0.1, the average number of replicas is 4.3 and 2.9, respectively.

The load balancing of storage nodes versus the parameter θ (in Eq. (7)) using the proposed strategy is shown in Fig. 6. Parameter θ is the impact factor related to the impact of feature and structural scores. When the parameter θ gets close to 1, the structural score of nodes is prioritized over the feature score, and vice versa.

Note that, Std notation indicates the standard deviation of data access in a normal distribution. The lower values for Std denotes that many accesses are requested for a small group of data. On the other hand, a more distributed access pattern for the data results in higher values for Std .

As presented in Fig. 6, the load balancing of nodes is increased by considering both the feature and structural scores of storage nodes in the proposed strategy. This is because, regardless of the parameter Std , by setting up the parameter θ between 0 and 1, a more balanced workload is achieved than setting the parameter θ to 0 or 1. For instance, if the parameter θ gets close to 0, the load balancing of the nodes increases since more distributed access pattern for the data is happened. On the other hand, when a

small group of data are more popular to many users, this shows that the parameter θ is close to 1.

As illustrated in Fig. 6, the feature score has more influence than the structural score for higher values of Std . Alternatively, for lower values of Std , the structural scores of storage nodes have greater impact than the feature one. For example, assigning 0.3 and 10 for the parameter θ and Std respectively, the nodes workload is at their highest balanced states.

If we decrease Std , the workload mostly gets higher for a constant value of the parameter θ . For example, if the parameter θ is assigned to 0.4, the lowest load balancing of the network occurs for $Std = 0.1$.

Note that, if the parameter θ is set to 0, the feature score is only considered for storage nodes. On the other hand, by assigning the value of 1 for the parameter θ , the feature score is not used. Clearly, considering both the feature and structural scores has more advantages over using only one score and is taken into account for selecting the nodes to store the replicated data.

Our proposed strategy is flexible in balancing the nodes workload. In this way, according to the data access rates and the network workload, we can change the parameter θ to reach our desired goals.

For the proposed strategy, the response time versus the parameter θ in Eq. (7) is shown in Fig. 7 for a variety of std values.

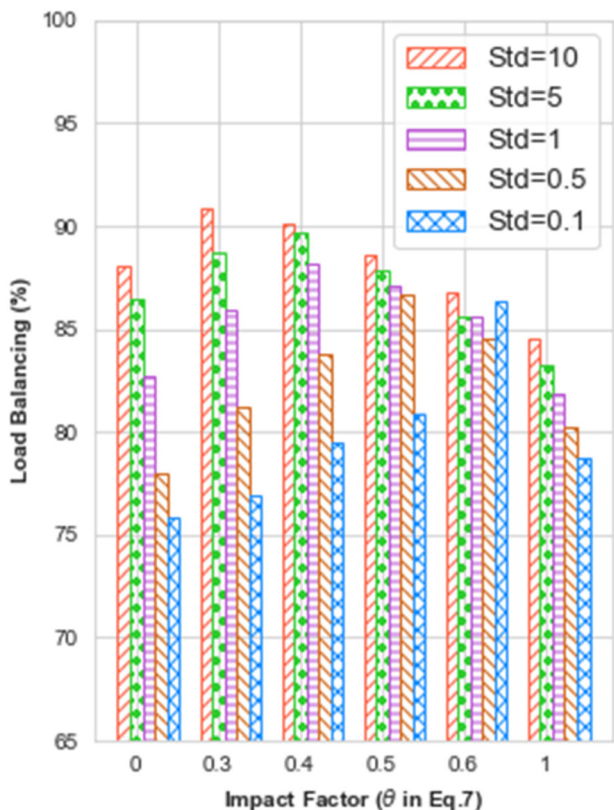


Fig. 6 The load balancing of nodes versus the impact factor (θ in Eq. (7)) in the proposed strategy

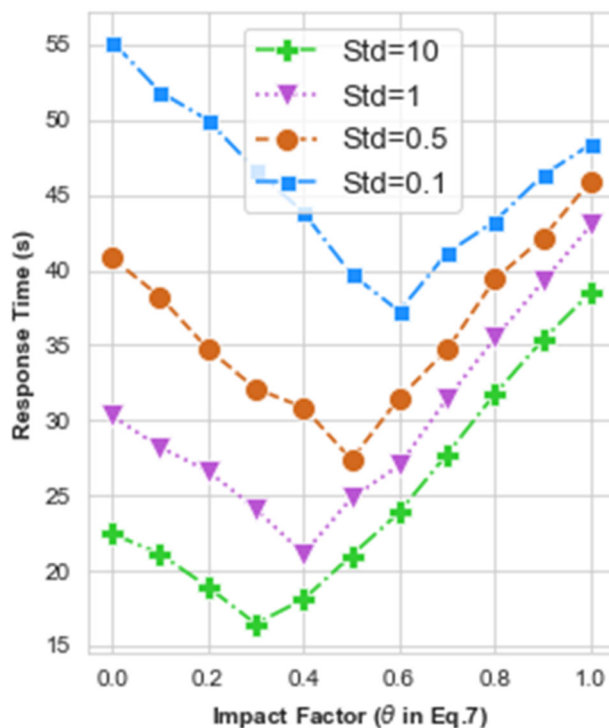


Fig. 7 The response time versus the impact factor (parameter θ in Eq. (7)) in the proposed strategy

As presented in Fig. 7, when Std is equal to 10, the lowest response time occurs value the parameter θ is equal to 0.3. Similarly, for different values for Std, the lowest response time is achieved when the parameter θ is between 0 to 1 exclusively.

Having a more distributed access rate for the data results in lower response time. Furthermore, access to a small group of data increases the response time. As a result, the idea of considering both the feature and the structural score in the proposed strategy improves the network performance in terms of the response time. This is attained using the parameter θ in the proposed strategy. Thus, the proposed strategy can flexibly adjust the parameter θ according to the data access rate.

In Fig. 8, the nodes' load balancing versus the number of cloudlets in the proposed strategy, DROPS [25] and random placement strategy are depicted and compared.

As shown in Fig. 8, when the number of cloudlets sent to the network is 1500, the percentage of load balancing in the proposed strategy, DROPS [25] and random placement strategy are 87.4, 81.8 and 78.4 respectively.

The proposed strategy improves the network performance by increasing the nodes load balancing, compared to DROPS [25] and the random placement strategy. This is because, only the structural score is considered in DROPS [25]. On the other hand, the random placement strategy does not use any feature or structural status of nodes for selecting the storage nodes.

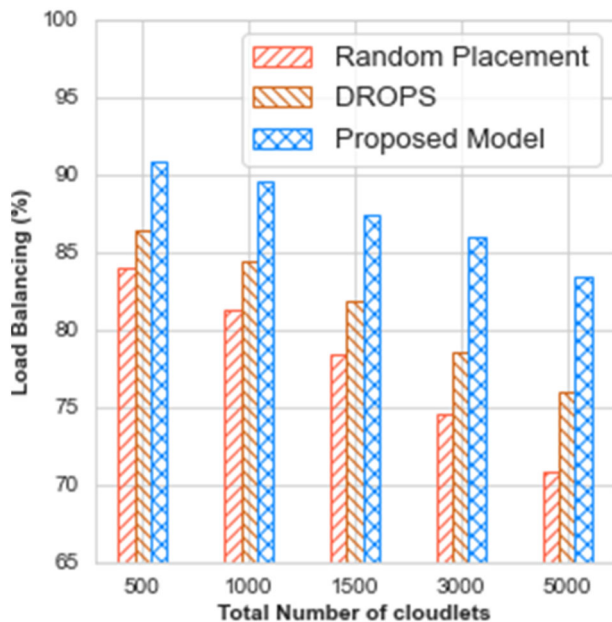


Fig. 8 The nodes load balancing versus the number of cloudlets in the proposed strategy compared to DROPS [25] and random placement strategy

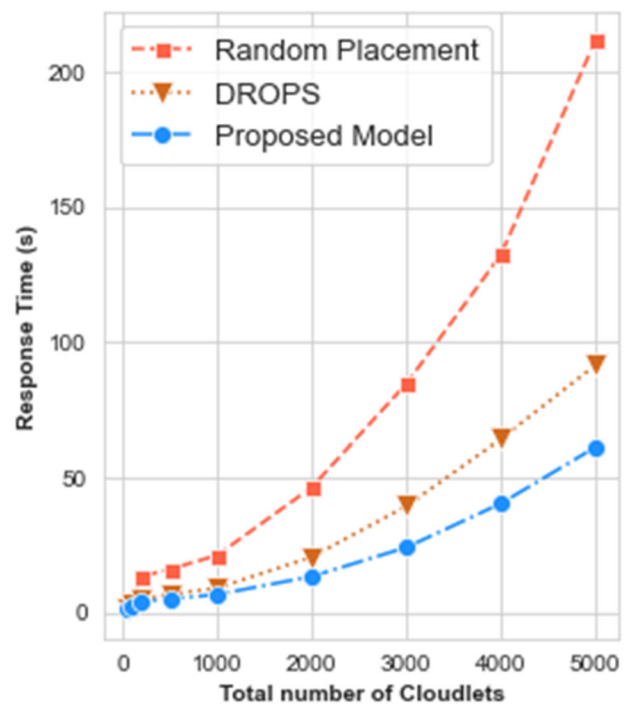


Fig. 9 The response time versus the number of cloudlets in the proposed strategy compared to DROPS [25] and random placement strategy

In Fig. 9, the response time versus the number of cloudlets is illustrated in the proposed strategy, DROPS [25] and random placement strategy.

The proposed strategy benefits from lower response time in comparison with DROPS [25] and random placement strategy. For instance, when the number of cloudlets is 3000, the average response time in the proposed strategy, DROPS [25] and the random placement strategy is 24.050, 39.360 and 84.550 s respectively.

In the following, the time complexity of the proposed strategy is evaluated. The first part of the proposed strategy is related to the data popularity determination. If m indicates the number of data in the network, the smallest data is initially searched in linear time, i.e., $O(m)$.

Afterwards, the data popularity for each data is calculated according to Eq. (2) which is done at linear time i.e., $O(m)$. Finally, all the data are sorted based on their popularity which is accomplished in time $O(m * \log m)$. Consequently, the total time complexity for calculating the data popularity is equal to $O(m * \log m)$.

The second part of the proposed strategy is related to the proper nodes determination for storing the most popular data. In this process, the lowest and highest degrees of the storage nodes are found using divide and conquer approach. This is performed in $(\frac{3}{2}n - 2)$ steps and has a linear time complexity equal to $O(n)$ with n showing the number of storage nodes in the network.

As stated in [15], the average node degree in the Internet network graph is 6.34. Hence, the total number of network links is equal to $(\frac{6.34}{2}n)$. In this way the lowest and highest weights links are detected in $(\frac{3}{2}(\frac{6.34}{2}n) - 2)$ steps.

The feature and structural scores of each storage node are computed in constant time C . Additionally, the storage nodes are sorted according to their total score in the time $O(n * \log n)$. Consequently, the total time complexity for determining the proper nodes for storing the most popular data is equal to $O(n * \log n)$.

Figure 10 illustrates the number of execution cycles versus the number of nodes in the network when determining the proper nodes for storing the most popular data in the proposed strategy.

When only the feature score is considered, i.e., $\theta = 0$, the time complexity decreases significantly. Thus, calculating the structural score of storage nodes takes considerable amount of time in the proposed strategy.

Figure 11 depicts the execution cycles versus the number of storage nodes and data files in the network of the proposed strategy.

As depicted in Fig. 11, when the number of the storage nodes is close to 500, the execution time is independent of parameter θ . However, by increasing the number of data files, the execution time of the proposed strategy increases linearly and are affected more by the parameter θ . This is because, a high portion of the time cost is related to the computation of structural scores.

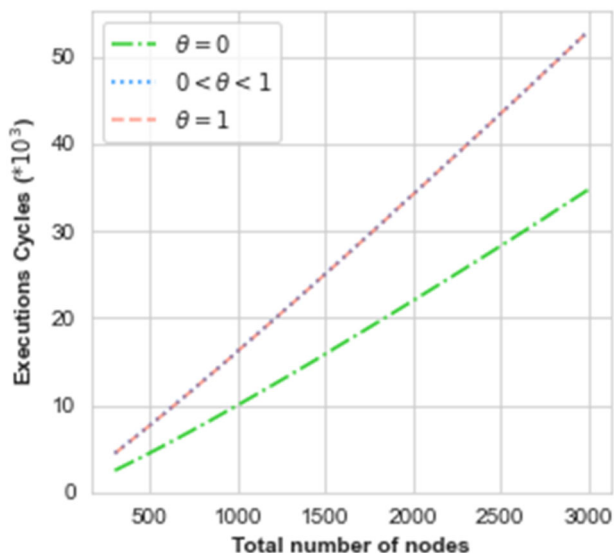


Fig. 10 The number of execution cycles versus the number of nodes in the network when determining the proper nodes for storing the most popular data in the proposed strategy

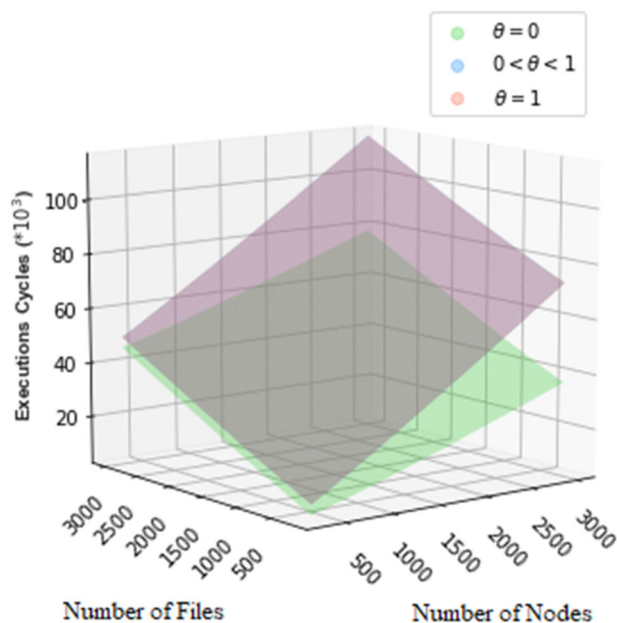


Fig. 11 The number of execution cycles versus the number of nodes and data files in the network of the proposed strategy

5 Conclusions and Future Work

In this paper, we have proposed a novel dynamic data replication strategy to improve data availability and network performance. In our proposed strategy, data files with the most frequent access rates are selected for replication. These popular data are then placed on the storage nodes with higher structural and feature scores. The feature score considers the configuration specifications of storage nodes. The structural score uses the position of storage nodes in the network.

Our simulation results shows that the introduced scores have significant impacts on the network performance. This is due to the fact that when we consider both the feature and structural characteristics of storage node, the data access rate distributes more uniformly in the network. As a result, more balanced workload in the storage nodes and lower response time are achieved. Moreover, the proposed strategy has the flexibility to adjust the impact of feature and structure scores according to the network conditions.

As part of a future work, one can consider the nodes proximity to determine the proper nodes for storing the most popular data. This can provide the ability to capture the structural status of nodes more precisely. Additional improvement to the proposed strategy would be also to balance between the communication and computation overhead in the storage nodes.

References

- Nachiappan, R., Javadi, B., Calheiros, R.N., Matawie, K.M.: Cloud storage reliability for Big Data applications: A state of the art survey. *J. Netw. Comput. Appl.* **97**, 35–47 (2017)
- W. Li, Y. Yang and D. Yuan, “Literature Review,” in *Reliability Assurance of Big Data in the Cloud: Cost-Effective Replication-based Storage*, Melbourne, Australia, School of Software and Electrical Engineering Swinburne University of Technology Hawthorn, 2015, pp. 9–17.
- A. S. Tanenbaum and M. Van Steen, *Distributed Systems: Principles and Paradigms*, Prentice-Hall, 2007.
- Hanan, J., Kechaou, Z., Ayed, M.B.: An enhanced healthcare system in mobile cloud computing environment. *Vietnam Journal of Computer Science* **3**(4), 267–277 (2016)
- M. R. Mesbahi, A. M. Rahmani and M. Hosseinzadeh, “Reliability and high availability in cloud computing environments: a reference roadmap,” *Human-centric Computing and Information Sciences*, vol. 8, no. 20, 2018.
- Q. Wei, B. Veeravalli, B. Gong, L. Zeng and D. Feng, “CDRM: A Cost-effective dynamic replication management scheme for cloud,” in *IEEE International Conference on Cluster Computing*, Crete, Greece, 2010.
- Yang, J.-P.: Efficient load balancing using active replica management in a storage system. *Math. Probl. Eng.* **2016**(1), 1–9 (2016)
- J. A. Hoxmeier and C. Dicesare, “System Response Time and User Satisfaction: An Experimental Study of Browser-based Applications,” *AMCIS 2000 Proceedings*, 2000.
- Tos, U., Mokadem, R., Hameurlain, A., Ayav, T., Bora, S.: Ensuring performance and provider profit through data replication. *Clust. Comput.* **21**(3), 1479–1492 (2018)
- A. Shakarami, M. Ghobaei-Arani, A. Shahidinejad, M. Masdari and H. Shakarami, “Data replication schemes in cloud computing: a survey,” *Cluster Comput.*, 2021.
- B. Alami Milani and N. Jafari Navimipour, “A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions,” *Journal of Network and Computer Applications*, **64**, 229–238, 2016.
- B. Tomas and B. Vuksic, “Peer to peer distributed storage and computing cloud system,” in *34th International Conference on Information Technology Interface*, Cavtat, 2012.
- T. Amrit, “Platform architecture,” in *Platform ecosystems: aligning architecture, governance, and strategy*, Newnes, 2013, pp. 73–117.
- Jafari Navimipour, N., Sharifi Milani, F.: A comprehensive study of the resource discovery techniques in Peer-to-Peer networks. *Peer-to-Peer Netw Appl* **8**, 474–492 (2015)
- Barabási, A.L.: *Network Science*. Cambridge University Press, Cambridge (2016)
- Newman, M.: The large-scale structure of networks. In: Newman, M. (ed.) *Networks: An Introduction*, pp. 235–270. Oxford University Press, Oxford (2010)
- Meng, X.: A churn-aware durable data storage scheme in hybrid P2P networks. *J. Supercomput.* **74**, 183–204 (2018)
- Mohammadi, B., Jafari Navimipour, N.: Data replication mechanisms in the peer-to-peer networks. *Int. J. Commun Syst* **32**, e3996 (2019)
- Cherbal, S., Barouchi, I.: ZRR-P2P: zone-based mechanism for data replication and research optimization in unstructured P2P Systems. *International Information and Engineering Technology Association* **26**, 23–32 (2021)
- R. Upra, R. Chaisricharoen and M. Yaibuates, “Personal Cloud P2P,” *Wireless Personal Communications*, 2021.
- Shen, H., Liu, G.: A lightweight and cooperative multifactor considered file replication method in structured P2P systems. *IEEE Trans. Comput.* **62**(11), 2115–2130 (2013)
- Hassanzadeh-Nazarabadi, Y., Küpçü, A., Özkasap, Ö.: Decentralized and locality aware replication method for DHT-based P2P storage systems. *Futur. Gener. Comput. Syst.* **84**, 32–46 (2018)
- Shen, H., Liu, G., Chandler, H.: Swarm intelligence based file replication and consistency maintenance in structured P2P file sharing systems. *IEEE Trans. Comput.* **64**(10), 2953–2967 (2015)
- M. Rahmani and M. Benchaïba, “An efficient replication scheme to increase file availability in mobile P2P systems,” in *International Symposium on Networks, Computers and Communications (ISNCC)*, 2017.
- M. Ali, K. Bilal, S. U. Khan, B. Veeravalli, K. Li and A. Y. Zomaya, “DROPS: Division and Replication of data in cloud for Optimal Performance and Security,” in *IEEE Transactions on Cloud Computing*, 2018.
- Sun, S.Y., Yao, W.B., Li, X.Y.: DARS: A dynamic adaptive replica strategy under high load Cloud-P2P. *Futur. Gener. Comput. Syst.* **78**, 31–40 (2018)
- L. Metcalf and W. Casey, “Graph Theory,” in *Cybersecurity and Applied Mathematics*, Rockland, Massachusetts, Syngress, 2016, pp. 67–94.
- Parau, P., Lemnar, C., Dinsoreanu, M., Potolea, R.: “Opinion Leader Detection,” in *Sentiment Analysis in Social Networks*, pp. 157–170. Morgan Kaufmann, Romania (2017)
- Wei, D., Deng, X., Zhang, X., Deng, Y., Mahadevan, S.: Identifying influential nodes in weighted networks based on evidence theory. *Phys. A* **392**(10), 2564–2575 (2013)
- S. K. Patro and K. K. Sahu, “Normalization: {A} Preprocessing Stage,” *Computer Research Repository*, 2015.
- Chen, G., Wang, X., Li, X.: “Preliminaries,” in *Fundamentals of Complex Networks: Models*, pp. 15–90. Wiley, Structures and Dynamics (2015)
- X. Bai, H. Jin, X. Liao, X. Shi and Z. Shao, “RTRM: A response time-based replica management strategy for cloud storage system,” in *International Conference on Grid and Pervasive Computing*, Berlin, 2013.
- Gao, G., Li, R., Wen, K., Gu, X.: Proactive replication for rare objects in unstructured peer-to-peer networks. *J. Netw. Comput. Appl.* **35**(1), 85–96 (2012)

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



Ali Majed is a M.Sc. student in the department of Computer Science at University of Isfahan from 2018. He received his B.Sc. in Computer Science from SRTTU in 2018. His research interest is in the field of cloud computing specially the replication methods to increase performance in cloud environment systems.



Fatemeh Raji is an assistant professor in the department of Computer Science at University of Isfahan, Iran from 2015. She obtained her Ph.D. in computer science from Isfahan University of Technology, Iran in 2014. She also received her M.S. and B.S. degrees in computer science from the University of Isfahan, Iran in 2007 and from Isfahan University of Technology in 2003, respectively. Her research interests included availability and privacy issues in the cloud computing, online social networks and distributed systems.



Ali Miri is currently a Full Professor with the School of Computer Science, Ryerson University, Toronto. His research interests include cloud computing and big data, computer networks, and security and privacy technologies and their applications. He has authored and coauthored more than 200 refereed articles, six books, and eight patents in these fields. He has chaired over a dozen international conference and workshops, and had served on more than 100 technical program committees. Dr. Miri is a member of the Professional Engineers Ontario.