Hindawi Publishing Corporation
Scientific Programming
Volume 2016, Article ID 1496714, 13 pages
http://dx.doi.org/10.1155/2016/1496714

Research Article

Data Sets Replicas Placements Strategy from Cost-Effective View in the Cloud

Xiuguo Wu

School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan 250014, China

Correspondence should be addressed to Xiuguo Wu; xiuguosd@163.com

Received 29 December 2015; Accepted 24 April 2016

Academic Editor: Ligang He

Replication technology is commonly used to improve data availability and reduce data access latency in the cloud storage system by providing users with different replicas of the same service. Most current approaches largely focus on system performance improvement, neglecting management cost in deciding replicas number and their store places, which cause great financial burden for cloud users because the cost for replicas storage and consistency maintenance may lead to high overhead with the number of new replicas increased in a pay-as-you-go paradigm. In this paper, towards achieving the approximate minimum data sets management cost benchmark in a practical manner, we propose a replicas placements strategy from cost-effective view with the premise that system performance meets requirements. Firstly, we design data sets management cost models, including storage cost and transfer cost. Secondly, we use the access frequency and the average response time to decide which data set should be replicated. Then, the method of calculating replicas' number and their store places with minimum management cost is proposed based on location problem graph. Both the theoretical analysis and simulations have shown that the proposed strategy offers the benefits of lower management cost with fewer replicas.

1. Introduction

Today, several cloud providers offer storage as a service, such as Amazon S3 [1], Google Cloud Storage (GCS) [2], and Microsoft Azure [3]. All of these services provide storage in several data centers distributed around the world. Clients can store and retrieve data sets without buying and maintaining their own expensive IT (Information Technology) infrastructures. Ideally, CSPs (Cloud Service Providers) should be able to provide low-latency service to their clients by leveraging the distributed locations for storage offered by these services. However, today's Internet still cannot guarantee quality of services and potential congestions can result in prolonged delays. Replication technology has been commonly used to minimize the communication latency by bringing the copies of data sets close to the clients [4]. Moreover, they also provide data availability, increased fault tolerance, improved scalability, and reduced response time and bandwidth consumption. Amjad et al. [5] have presented various dynamic replication strategies.

Compared with the definitions of conventional computing paradigms such as cluster [6], grid [7], and peer-to-peer (p2p) [8], "economics" is a noticeable keyword in cloud computing which has been neglected in data sets replicas placements. "Economics" denotes that cloud computing adopts a pay-as-you-go model, where clients are charged for consuming cloud services such as computing, storage, and network services like conventional utilities in everyday life (e.g., water, electricity, gas, and telephony) [9]. However, most current replicas approaches largely focus on improving reliability and availability [10, 11] by providing users with different replicas of the same service, ignoring the management cost spending on replicas, which cause great financial burden (storage cost, transfer cost, etc.) not only for cloud users, but also for CSPs.

It is obvious that the client access latency can be reduced with the number of replicas increased. And every client demands to access its data set from a replica that is as close as possible in order to minimize its delay. However, there are at least two challenges while replicating all data sets to all data centers can ensure low-latency access [12]. First, the system can not offer unlimited storage resources, and that approach is costly and may be inefficient for the extra storage resource consumption. Second, the problem is more complicated when the data set may be updated, and the more the replicas

are in the system, the higher the update cost will be. Thus, data set replicas need to be carefully placed to avoid unnecessary expense. And the replicas number and their store placements may have a profound impact on the optimal replicas distribution in a balance way from cost-effective view. The resulting tradeoff between the number of replicas and the data set delay maps precisely to the replicas placement problem.

In practice, CSPs supply a pool of resources, such as hardware (storage, network), development platforms, and service at the expense of cost. And data set storage and transfer costs are the two most important components in data management, which are caused by storage resource and bandwidth consumption, respectively. Also, with the number of new replicas increased, the transfer cost will be declined because the data set can transfer more effectively; but the storage cost is getting bigger because of new replicas' existence. That is to say, too many replicas in the cloud may lead to high storage cost, and the increased storage cost for replicas may be greater than the reduced transfer cost, if there is no suitable replicas placements strategy. Therefore, it is urgent to find a balance selectively to replicate the popular data or not.

Based on the analysis above, there are at least three important issues that must be solved in order to achieve the minimum-cost data set replicas placements scheme: (1) whether or not to create a replica in cloud computing environment; (2) how many data set replicas should be created in the cloud; (3) where the new replicas should be placed to meet the system task successful execution rate and bandwidth consumption requirements.

Therefore, in this paper, towards achieving the minimum-cost replicas distribution benchmark in a practical manner, we propose a replicas placements strategy model, including the way to identify the necessity of creating replica, and design an algorithm for replicas placements that can easily reduce the total cost in the cloud.

The main contributions of this paper include (1) proposing data sets management cost models, involving storage cost and transfer cost; (2) presenting a novel global data set replicas placements strategy from cost-effective view named MCRP, which is an approximate minimum-cost solution; (3) evaluating replicas placements algorithms using analysis and simulations.

The remainder of this paper is organized as follows: Section 2 presents related works in data set replicas placement and management cost. Then, in Section 3, data sets cost models are proposed, and a replicas scarce resource test algorithm is shown in Section 4. Section 5 describes the data sets replicas number and store places from cost-effective view based on Steiner Graph. Section 6 addresses the simulation environments, parameters setup, and performance evaluations of proposed replicas solutions. Finally, conclusions and future works are given in Section 7.

2. Related Works

We will present in this section the background related to data sets replicas placements and management cost models in the cloud.

2.1. Data Sets Placements Strategies. Data sets replication is considered to be an important technique used in cloud computing environment to speed up data access, reduce bandwidth consumption and user's waiting time, and increase data availability [13]. Data sets replicas placement is the problem of placing duplicate copies of data set in the most appropriate node in the cloud, which can be logically divided into two stages, namely, replication decision and replicas placements. The replication decision stage decides whether to create the replica. If the decision is not to replicate, the data set will be read remotely. The second stage is to select the best sites to store the new replicas. There are two types of replicas placements techniques: centralized and distributed. Distributed replicas placements can be further classified according to different implementation: free-scale topology [14, 15]; graph topology [16]; multitier architecture [17]; hierarchical architecture [18, 19]; peer-peer architecture [20]; tree architecture [21]; and so on. As a typical centralized replication placements technology, Andronikou et al. have proposed a dynamic QoS-aware data replication technique which is based on data importance in [22]. Kalpakis et al. considered the minimum cost of servicing read and write requests in a distributed system; however it is in a tree network [23].

Conclusions as a result, the above-mentioned replication technologies have not involved data set storage and transfer cost, which are the most important elements for the clients in deciding whether or not to use cloud storage system, especially for small business. Therefore, we will consider the data sets management cost as a basis for replicas placements in order to minimize the storage and transfer costs on the premise that system performance satisfies data set availability requirements in this paper.

2.2. Data Set Management Cost. In a pay-as-you-go paradigm, all the resources in the cloud carry certain costs, so the more the replicas the more we have to pay for the corresponding resources used. Some of them may often be reused while the others may not be. So, once we decide to create a replica, we need to evaluate its access frequency as well as management cost, especially when large data sets—or "big data"—are usually common in the cloud. In [23], toward practically achieving the minimum data set storage cost in the cloud, a runtime local-optimization based storage strategy has been developed. The strategy is based on the enhanced linear CTT-SP algorithm used for the minimum-cost benchmarking. Theoretical analysis and random simulations have shown its validity and reliability. Auction protocol is used by Bell et al. to select the best replica of a data file, where the total cost is computed as the sum of file transfer cost and estimated queuing time for all jobs in the queue [24].

Base on the analysis, it is very necessary to design the data sets replicas placements strategy from cost-effective view. And this research is very significant for business, especially for small businesses, which usually use big data on cloud computing platforms.

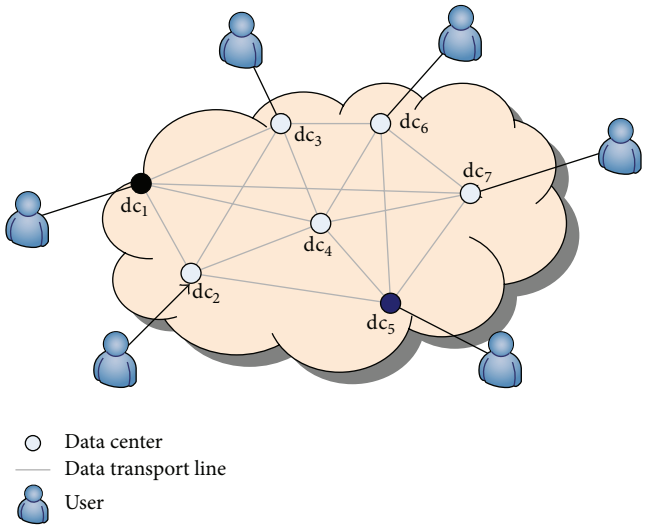

Figure 1: Architecture of cloud environment.

3. Data Sets Cost Models in the Cloud

In this section, we will first present some concepts in cloud environment; then we propose storage cost model and single transfer cost model, respectively. At last, we present data set management cost model in the cloud.

3.1. Some Concepts in the Cloud. In cloud storage system, there are some distributed data centers in cloud environment for data sets storage. And each data center has some properties, such as storage capacity, CPU speed, and network bandwidth, and read/write speed. Similarly, different configurations of data center lead to different quality of service (QoS).

Definition 1 (cloud computing environment, CCE). Cloud computing environment (CCE) can be regarded as a set of distributed data centers, written as CCE $= \bigcup_{i=1,2,\ldots,|\mathrm{DC}|}\{\mathrm{dc}_i\}$, where dc_i denotes the ith data center.

Figure 1 describes basic architecture of cloud computing environment constituted by seven data centers.

Definition 2 (data center, DC). In CCE, each data center dc_i can be described as a 5-tuple: $(\mathrm{dc}_i, s_i, \mathrm{vs}_i, \mathrm{sp}_i, \mathrm{tp}_i)$, where dc_i is the identifier of data center, which is a unique identification in CCE; s_i is the total space of data center dc_i, whose unit is TB; vs_i is the size of vacant space on data center, which means the extra storage capacity of dc_i; sp_i means the storage price of data, determined by the service provider; and tp_i is transfer cost ratio with each unit size data set.

Definition 3 (data set, d). Data set d_m in CCE can be described as a 3-tuple: (d_m, s, p), where d_m is the identifier of data set, and it is unique in the whole cloud environment; s is the size of data set; and p is its store place, $p \in \mathrm{DC}$.

In the following section, we assume that the architecture of CCE and data set d_m are relatively fixed during a period of time for simplicity.

3.2. Data Sets Storage Cost Model. In a commercial cloud computing environment, service providers have their cost models to charge users. For example, Amazon cloud service's prices are as follows: \$0.15 per gigabyte per month for the storage resource [1]. Storage cost depends on parameters such as the CSP's price policy, the size of the data set (original data set and inserted data set), and the storage time.

Definition 4 (storage cost, c_s). Data set d_m's storage cost is a function of its data size $d_m \cdot s$, storage time t, and its deployed data center dc_i's storage cost ratio sp_i, and can be represented as follows: $c_s = \mathrm{sp}_i \times d_m \cdot s \times t$.

That is to say, the total storage cost is the CSP's storage cost ratio function multiplied by the size of the data set and its storage time, for example, using Amazon S3 for storage pricing and considering that 0.5 T (512 G) data set has been stored for 6 months. The storage cost is \$0.15 $*$ 512 $*$ 6 = \$460.8.

3.3. Data Sets Transfer Cost Model. In the cloud, the data sets transfers are absolutely necessary once a request arrives, in which process the transfer cost will be generated inevitably for the reason for network consumption. In this model, the input data sets transfers are free, whereas output transfer cost varies with respect to data set volume and the CSP's atomic transfer cost ratio function.

Definition 5 (transfer cost, c_t). Data set d_m's transfer cost is the product of its data sets transfer time t_t and data center dc_i's atomic transfer cost tp_i, which can be described as follows: $c_t = \mathrm{tp}_i \times t_t$.

It is noted that the transfer time t_t depends heavily on the data set size and network bandwidth. And in practice, the bandwidth may fluctuate from time to time according to peak and off-peak data access time. In this paper, we simplify the problem and regard the bandwidth as a static value, ignoring the volatility over time. For example, for a 10 G data set, the single transfer cost is \$0.12 $*$ 10 = \$1.2, if the transfer cost ratio is \$0.12 per GB data set.

3.4. Data Sets Management Cost Model. In this paper, we facilitate a data set d_m's management cost during a period depending on several parameters: the size of the data sets $d_m \cdot s$, the time t, and the requests times n during t. And d_m's total management cost can be defined as follows.

Definition 6 (data set d_m's total management cost, tc). Data set d_m's total management cost during a period of t is the sum of its storage cost c_s and transfer cost c_t: $\mathrm{tc} = c_s + n \times c_t = \mathrm{sp}_i \times d_m \cdot s \times t + \mathrm{tp}_i \times t_t \times n$, where n is the requested times.

Let us introduce a simple example: a 500 G data set is stored in the cloud for a month, and the storage cost is \$0.1 per GB and per month. The transfer cost is \$0.12 per GB data set, and the number of requests is 10. Then, storage cost is \$50, and transfer cost is \$600, for a total of \$650 in a month.

Input: data set d_m, ε_{af}, ε_{rt}, t;
Output: if d_m is a replica scarce resource, return true; else return false;
(01) count the data set d_m's requested access times n using logs files;
(02) set $af_{d_m} = n/t$;
(03) sum the data set d_m's response time rt_i using logs files;
(04) set $art_{d_m} = (\sum_{k=1}^{i} rt_k)/n$;
(05) set $rsr_{d_m} = art_{d_m}/d_m \cdot s$;
(06) if (($af_{d_m} > \varepsilon_{af}$) and ($rsr_{d_m} > \varepsilon_{rt}$))
(07) return true;
(08) else
(09) return false;
(10) End.

Algorithm 1: Replica scarce resource testing.

4. Replicas Scarce Resource Model in the Cloud

In this section, we will present the replicas scarce resource model in order to determine whether or not to create a new replica in the cloud.

4.1. Replicas Scarce Resource. There are many data sets stored on the data center in the cloud environment. And it is not necessary to replicate the data set on all the data centers. It is intelligent to replicate the popular data sets with high user frequencies for reducing data set transfer delay. In this way, we will define replica scarce degree as a criterion of adding replicas.

Definition 7 (access frequency, af). For a data set d_m, its access frequency is the requested access times per unit of time and can be represented as follows: $af = \sum_{i=0}^{n} times_i/n$, where $times_i$ indicates the number of accesses to the replicas on data center dc_i of unit time interval and n is the total number of replicas.

If a data set d_m's access frequency is greater than a preset threshold ε_{af}, then data set d_m is hot data.

Definition 8 (response time, rt). Response time rt is the time that elapses when a service requests a data set until the user receives the complete data set.

It is obvious that average response time can be calculated starting from the initial time when the request is submitted till the final response if received with the image from the target node.

Definition 9 (average response time, art). Average response time art is the ratio of total response time and the requested times per unit of time and can be represented as follows: $art = (\sum_{k=1}^{m} rt_k)/m$, where m is the requested times during a period of time t.

Average response time (art) is a basic parameter to determine the replicas numbers and stored places for the reason that the awt can be reduced by placing replicas on data centers. However, awt is not the only valid parameter to create replicas. The reason is the average response time depends on a number of factors, such as bandwidth and data set size al. And the bigger the data set size, the longer the average response time. In this way, we will present replica scarce resource by introducing data set size.

Definition 10 (replica scarce resource, rsr). A data set d_m is a replica scarce resource if the ratio of a data set d_m's average response time art_{d_m} and its size are less than a preset threshold ε_{rt}, which can be described as $art_{d_m}/d_m \cdot s > \varepsilon_{rt}$.

To sum up, it is necessary to create replicas for replica scarce resource. And there are two important factors to be considered before creating new replica: (1) longer average response time and (2) higher requests frequency.

For those data sets with low requested frequency, and those with high requested frequency but short response time, there is no need to place replicas from cost-effective view. Algorithm 1 describes whether it is necessary to create replicas for data set d_m.

Algorithm 1 presents the way to determine the possibility of creating replicas. And its time complexity is $O(n)$, for the reason that line (04) sums up all the response time n times.

4.2. Replica Placements from Cost-Effective View. Once the decision to create replicas has been made, the most urgent problem needed to solve is where to place it. In this subsection, we will present a replica placement algorithm from cost-effective view.

It is obvious that the replica's candidate store places are not unique, but a set of data centers. Then the most economic way is to select the data centers with lower storage cost and transfer cost to other data centers on the basis of shorter average response time. And Algorithm 2 presents the suitable stored places choosing schema by comparing the data set management cost during a period of time t.

In Algorithm 2, line (01) defines the total cost cc as a largest value, which will be modified immediately after the first cycle. And the function min() in line (10) and line (25) will return the smaller response time and transfer cost, respectively. Line (14) is mainly used to compare average

Input: data set d_m, data centers set DC = $\{dc_0, dc_1, dc_2, \ldots, dc_n\}$, dc_0 stores the primitive data set d_m, testing time t; pre-set average response time ε_{rt};
Output: data center dc_m with lowest cost;
(01) set cc = MAX_COST; //Assuming the cost is largest
(02) set $m = 0$; //Initialize return data center index
(03) for each data center dc_j (except data center dc_0)
(04) begin
(05) //Assuming the replica stores on dc_j;
(06) set $rs_j = 0$; //record total response time
(07) set $n_j = 0$; //record access times;
(08) for data center dc_k (except dc_0 and dc_j)
(09) begin
(10) set $rs_j = rs_j + n_k \times \min(rs_{k\_dc_0}, rs_{k\_dc_j})$;
(11) set $n_j = n_j + n_k$;
(12) end
(13) set $ars_j = rs_j / n_j$;
(14) if ($ars_j > \varepsilon_{rt}$)
(15) continue;
(16) else
(17) begin
(18) calculate the storage cost sc_j using Definition 4;
(19) set tc = 0; //transfer cost is initialized to zero
(20) set $n_j = 0$;
(21) for data center dc_k (except dc_0 and dc_j)
(22) begin
(23) calculate transfer cost $c_{tdc_k - dc_j}$ using Definition 5;
(24) calculate transfer cost $c_{tdc_k - dc_0}$ using Definition 5;
(25) set $tc = tc + n_k \times \min(c_{tdc_k - dc_j}, c_{tdc_k - dc_0})$;
(26) set $n_j = n_j + n_k$;
(27) end
(28) if $tc < cc$
(29) $m = j$;
(30) end
(31) end
(32) return m.

Algorithm 2: Select replica's economic stored placements.

response time before and after a replica is stored on data center dc_j. And the process will automatically break and turn into next cycle once average response time is still greater than preset threshold ε_{rt}.

Here, we will analyze the time complexity. Suppose there are n data centers, and in line (08), the loop times for transfer cost are $(n - 1)$, the same in line (21). So, the total time complexity is $O(n^2)$.

5. Data Sets Replicas Placements from Cost-Effective View

In the previous sections, we have tentatively placed one replica on a data center and obtained high system performance with lowest data sets management cost. However, replicas number and their storage places are still urgent problems to be solved from cost-effective view in practice. In this section, we will present an approximate minimum-cost replicas placements algorithm based on location problem (LP).

5.1. Minimum-Cost Replicas Placements Model. In order to formulate the minimum-cost replicas placements problem, we make the following assumptions: (1) The cloud computing environment is a customer-to-server system, in which the data sets themselves travel to the facilities to be served. On the other hand, the user requests the data set for further analysis, not a result by computing or querying from the data set. (2) Each data center represents a candidate replica location as well as a data set demand point, for the reason that client requests data via data center. (3) Only one replica may be located per data center. (4) The replicas service is uncapacitated; that is, they may serve an unlimited amount of data sets requests.

In its simplest form, the minimum-cost replicas placements problem is as follows: given a set of data centers, which represent demand points as well as candidate replicas placements, and a set of connections between each pair of data centers. Each connection has a transport cost per unit data set and each data center is associated with a charge for data set storage. Also, all demands must be routed over

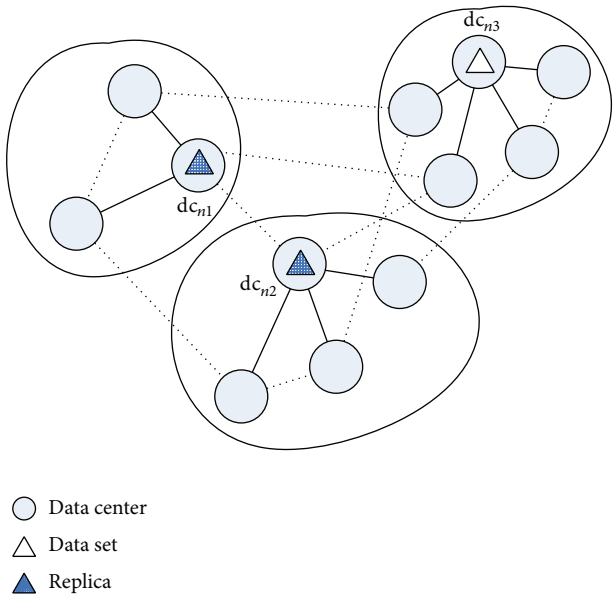

Figure 2: Data set access domain with minimum management cost.

the connection to the nearest replica. The problem is to find the set of replicas placements that minimize the total management cost: the sum of data sets storage and transport cost.

Ideally, the optimal minimum-cost replicas placement is shown in Figure 2, where each replica is stored in a domain D, and the rest of the data centers retrieve data set from it. In this way, the transfer lines and the data centers constitute a tree.

Next, we model the minimum-cost replicas placements problem. Conventionally, a cloud environment is represented by a graph where two nodes have an edge if and only if two corresponding nodes can communicate with each other. In order to describe such a circumstance, we will transform them into a graph G. The transformation rules are as follows:

(1) Each data center dc_i is mapped to a node dc_i in the graph, and all nods constitute the node set V.

(2) Each connection line between two data centers should be classified according to the type of network and used for implementation: (i) lines in one domain from a node with a replica to others can be transformed into edges with weight 0, respectively; (ii) lines in one domain between nodes without replica can be transformed into edges, respectively, and their weight is minimum transfer cost between corresponding data centers; (iii) lines cross-domain can be transformed into edges between corresponding nodes, and its weight is the minimum transfer cost.

(3) The storage cost of each data center should be mapped to the first property of corresponding node.

(4) The product of access frequency and time of period t is mapped to second property of corresponding node.

In order to describe the nodes, we define a 3-tuple (dc_x, c_x, p_x) representing the identifier, storage cost, and transfer times, respectively.

In this way, we wish to find optimal locations at which we place replicas to serve a given set of n clients; we are also given a set of locations at which replicas may be stored, where store replica at data center incurs a cost of $f_1(v_i)$, and each client j must be assigned to one replica (or source data), thereby incurring a cost of c_{ij}, proportional to the distance between data centers dc_i and dc_j; the objective is to find a solution of minimum total cost.

The minimum-cost replicas placements problem is defined as follows.

Definition 11 (minimum-cost replicas placements, MCRP). Given a connected undirected and weighted graph $G = (V, E, w, f_1, f_2)$, (1) V is the set of nodes; (2) E is the set of edges; (3) w is a function $w : e(v_i, v_j) \to R^+$; (4) f_1 is a function: $v_i \to R^+$, $v_i \in V$, and $f_1(v_i)$ is a nonnegative real value; (5) f_2 is a function $V \to Z^+ : v_i \to Z^+$, and $f_2(v_i)$ is a nonnegative integer value associated with each node. The goal is to divide V into two subsets V_1 and V_2, and V_1 is nodes set with replicas, while V_2 is nodes set without replicas and any $v_i \in V_2$ need to request data set from ($v_j \in V_2$). The task is to construct a forest constituted by trees. Each subtree consists of a source node (with replica) and multiple destination nodes such that the sum of transfer cost and storage cost is minimized. That is, the value from each node v_i to its root multiplied by $f_2(v)$ is minimized, represented as

$$\text{Minimize}\left(\sum_{v_i \in V_1(T)} f_1(v_i)\right) + \sum_{v_i \in V_2(T)} \sum d(T, v_i) f_2(v_i). \quad (1)$$

Several aspects of this formulation are worth noting. First, we observe that if we set the transfer cost to the same, then the result is simple, which is an alternate formulation of the uncapacitated facility location problem (UFLP) in which link additions are disallowed. Thus, the UFLP is a special case of the UFLNDP in which link additions are disallowed. Since the UFLP is NP-hard (in the parlance of computational complexity) [25, 26], so is the more general MCRP. Therefore, we can conclude the following theorem.

Theorem 12. *MCRP is NP-hard.*

In the original problem, we need to decide d_m's replicas number and their places to put. Once we select a data center dc_k, then the sum of the rest of the data centers transfer cost and the storage cost must be minimal compared to selecting other data centers. In the same way, the question that how many replicas need to be placed in the cloud platform has turned into how to select a subset of nodes in graph G that can minimize the sum of nodes weight and edges weight.

Then, it can be regarded as a UFLP using mapping rules shown in Table 1.

Note that the storage cost for the node and cost of the edges should be expressed in comparable units; for example, the storage cost for each node can be expressed in dollars for

Table 1: Mapping rules from minimum-cost replica placements strategy to UFLP.

Index	Element in minimum-cost replica placements strategy	Mapping	Element in UFLP
(1)	The transfer cost ratio between data centers c_t	→	$w(v_i, v_j)$
(2)	The storage cost of each data center c_s	→	$f_1(v_i)$
(3)	The user requested access times	→	$f_2(v_i)$
(4)	$\min\left(\text{Cost}_{\text{storage}}(\text{dc}_k) + \sum \text{Cost}_{\text{transfer}}\right)$	→	$\min(\text{Cost}(T))$

Input: Graph $G = (V, E, w, f_1, f_2)$ with edge-weighed and node-weighted;
Output: Graph $G' = (V', E', w')$ with edge weighed;
(01) Initialize an edge-weighed graph G'', by setting $V'' = V$, and $w'' = w$;
(02) For each $(v_i, v_j) \in E$, do
(03) Assign the weight of this edge as
(04) $w''(v_i'', v_j'') = w(v_i, v_j) + \max(f_1(v_i) + f_2(v_j), f_2(v_i) + f_1(v_j), f_2(v_i) + f_2(v_j))$;
(05) EndFor
(06) Initialize an edge-weighed graph G' by setting $V' = V''$, and $f_1' = f_1''$;
(07) For each $v_i \in V'$, do
(08) Assign the weight of edge from v_i to v_j
(09) $w''(v_i'', v_j'') = \text{Dijkstra}(v_i, v_j)$;
(10) Output G'.

Algorithm 3: Construct graph with edge weighted.

each replica, while the cost of each edge can be represented in dollars per request.

Theorem 13. *An optimal solution to the MCRP consists of p replicas and $|N|$-p connections, where N is the total number of data centers.*

This property quantifies our intuition about the tradeoff between constructing facilities and links; that is, as we build more facilities, fewer links are needed. The property also has implications in the identification of polynomial solvable cases, as has been discussed in [27].

In this way, the minimum-cost replicas placements problem is NP-hard. Therefore, no polynomial time algorithms of solving the problem are likely to exist for minimum-cost replica placements. Hence, it is of practical importance to obtain approximation methods whose costs are close to optimal.

5.2. Approximate Algorithm for Replicas Placements with Minimum Management Cost. In this subsection, we introduce an approximate algorithm for MCRP. The idea is first decomposing the transfer ratio to edge weight and then finding the candidate replicas placements data centers using graph G. Finally, the approximate solution for the graph G gives an approximate solution for the original problem.

5.2.1. Transformation from Edge and Node-Weighted Graph to Edge-Weighted Graph. First, we will construct a graph G and then move the user access frequency and storage cost to the edge as weight, constructing an edge-weighted graph. The basic idea can be summarized as follows: decomposing the storage cost on adjacent edges according to degree of data centers. Algorithm 3 describes the setting of edge weight.

In Algorithm 3, function Dijkstra(v_i, v_j) returns a minimum transfer cost from v_i to v_j. And the time complexity of Dijkstra is $O(n^2)$; we know that the complexity of Algorithm 3 is $O(n^3)$ from line (07).

5.2.2. Approximate Algorithm for MCRP. Next, we will propose approximate minimum management cost replicas placement algorithms based on the graph G'. The basic idea is to obtain possible replicas placements according to the minimum spanning tree, as is shown in Algorithm 4.

In Algorithm 4, function $\deg(v_i)$ means the number of edges linked to node v_i. And Algorithm 4 requires the number of nodes greater than two; if not, the nodes with minimum storage cost are the better ones.

Here, we will analyze the time complexity of Algorithm 4. A simple implementation using an adjacency matrix graph representation and searching an array of weights to find the minimum weight edge to add requires $O(|V|^2)$ running time. Kruskal Algorithm is greedy algorithm that runs in polynomial time, whose time complexity is $O(n^2)$; n is the number of vertexes. In the other steps, such as in line (03), its time complexity is $O(n^2)$. So, the total time complexity of MCRP is $O(n^2)$. A detailed example will be shown in Section 6.2.

The data center that deployed the original data set is responsible for the data set and its replicas' management, including when to create replicas and where to place them. With the data set requests increased (e.g., the number of requests amounts to 5000 in a month), the data set replicas placements algorithm with minimum management cost data

Input: Graph $G' = (V', E', w', f_1', f_2')$ with edge-nodes weighted as (dc_i, c_i);
Output: The nodes set for replicas placements.
(01) Generate a minimum-cost spanning tree from G' using Kruskal Algorithm;
(02) v_i = node with maximum degree;
(03) While $\deg(v_i) \geq 2$ do
(04) Begin
(05) v_i = node with maximum degree;
(06) Print v_i;
(07) For $v_j \in V$ and $i \neq j$ do
(08) If $e(v_i, v_j) \in E$
(09) Begin
(10) Delete $e(v_i, v_j)$;
(11) If $\deg(v_j) = 0$ Delete v_j;
(12) EndIF
(13) Delete v_i;
(14) v_i = node with maximum degree;
(15) EndWhile
(16) End.

Algorithm 4: Replicas placements with approximate minimum management cost.

center will start up, and the data set replicas will be transferred to other data centers according to the computation results. Also, a replicas distribution table including some important information such as original data set and its replicas' location will be placed, and its size, most recent update time, and so forth al, will send to each data center, in order to let the others know where the data sets are placed. In this model, a user that connected the data center accesses a data set as follows: First he/she tries to locate the data set replica locally. If the object replica is not present, he/she goes to check the data placement directory residing on each data center, which stored a data set replicas distribution structure. After that, the user's request goes to the nearest data center and then will transfer the data set to user via near data center.

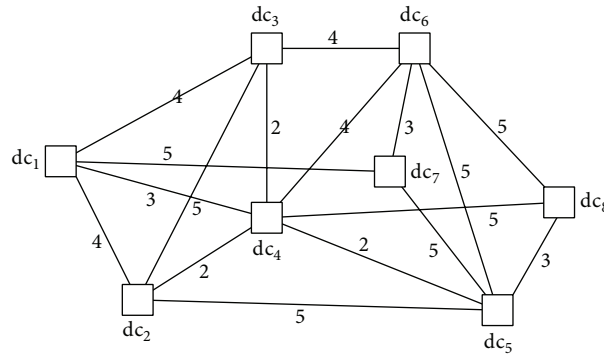

Figure 3: Architecture of cloud environment.

6. Analysis and Evaluation

In this section, we first present the experimental setups and then discuss the tradeoff between storage cost and transfer cost. Next, we describe the whole procedures of MCRP approximate algorithm using an example step by step. At last, we compare the result among different circumstances to demonstrate how our replicas placements strategy works.

6.1. Experimental Setup. The experiments were conducted on a cloud computing simulation environment built on the computing facilitates at Network & Information Security Lab, Shandong University of Finance and Economics (SDUFE), China, which is constructed based on SwinDeW [28] and SwinDeW-G [29–31]. The cloud system contains 10 data centers (servers) and 50 high-end PCs (clients), where we install VMWare (http://www.vmware.com), so that it can offer unified computing and storage resources.

Figure 3 describes the simulation architecture model of cloud storage platform, and the prices of cloud services follow the well-known Amazon's cost model: (1) \$0.15 per gigabyte per month for storage; (2) \$0.1 per gigabyte for data sets transfer process.

Table 2: Data sets access configurations.

Data centers	dc_1	dc_2	dc_3	dc_4	dc_5	dc_6	dc_7	dc_8
Users number m_i (×100)	1	0.6	0.8	1.1	0.6	0.7	0.9	1
Access frequency π (×10)	2	3	3	5	4	6	5	4

And in the analysis, we observe and study the running conditions for a period of one month. The usage frequency is according to Poisson distribution. Table 2 describes the data centers users' number m, access frequency π, and so forth.

6.2. Tradeoff between Storage and Transfer Costs. We define the cost of a solution in MCRP as the sum of storage cost and transfer cost. In order to compare the total costs with different replicas numbers, we have computed the storage and transfer costs, respectively. The result is shown in Figure 4. Every point on the black thick line in this diagram represents a value of sum cost.

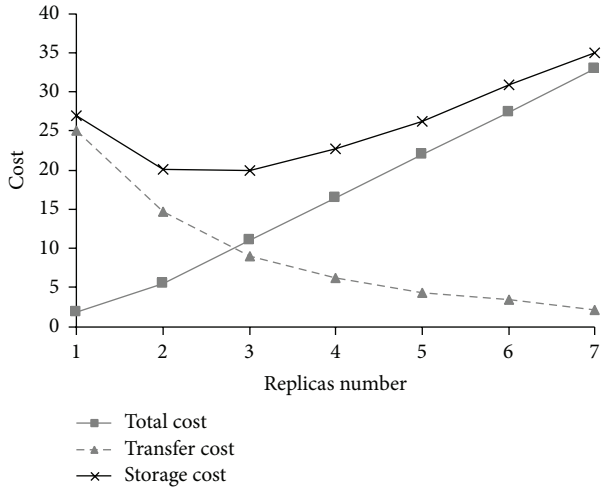

Figure 4: Tradeoff between storage and transfer costs.

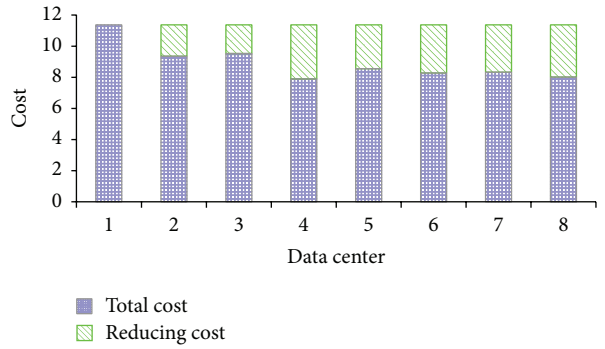

Figure 5: Reduced cost comparisons of data centers.

From Figure 4, we can see that, without replica, the storage cost is less than transfer cost for the reason that many data sets are transported in the system. However, with replicas number increased, the storage cost is gradually getting bigger for the reason that more replicas need to be stored on data centers, while the transfer cost becomes smaller for the nearby replicas access. Furthermore, the total cost has a minimum value with three replicas, which means the optimal solution.

Similarly, Figure 5 describes the reduced cost with one replica. And the green part is the value that can be saved. Also, it is obvious that the cost will be reduced with the replicas added. Also, we can see that the data center dc_4 is the replica storage place with maximum saved cost.

6.3. Approximate Algorithm for MCRP. In this section, we will analyze the MCRP algorithms proposed in Section 5.

First, we need to generate a minimum-cost spanning tree from Figure 3 using Kruskal Algorithm. And Figure 6 shows the result, including seven edges.

Then, we will select the node with maximum linked edges, for example, dc_4, and delete its adjacent edges. Also, if the degree of adjacent node is zero, then remove it at the same time, for example, nodes dc_1 and dc_2. Figure 7 shows the

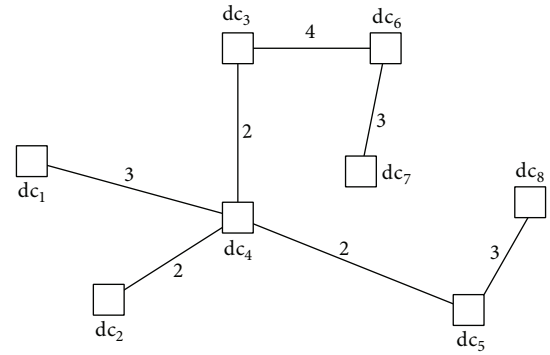

Figure 6: Minimum-cost spanning tree.

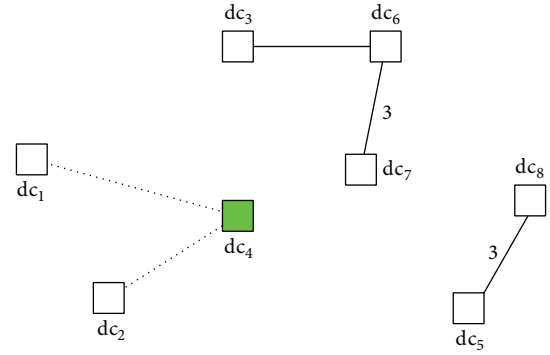

Figure 7: Result after first deletion.

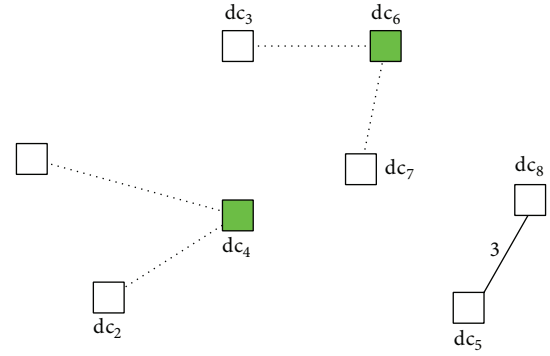

Figure 8: Result after second deletion.

result after first deletion. And in this step, dc_4 is the first selected replica store location.

It is obvious that there still exist nodes with degree greater than two, for example, dc_6, and then the node should be deleted and also its linked edges. In this way, dc_6 is the second selected replica store location. Figure 8 shows the result after second deletion.

In this case, the maximum degree of all nodes is only one, and then the node with small storage cost value is the suitable place to store replica, for the reason that they have the same transfer cost despite where the replica store place is. Figure 9 shows the result of replica store place, that is, a node set of $\{dc_4, dc_6, dc_8\}$.

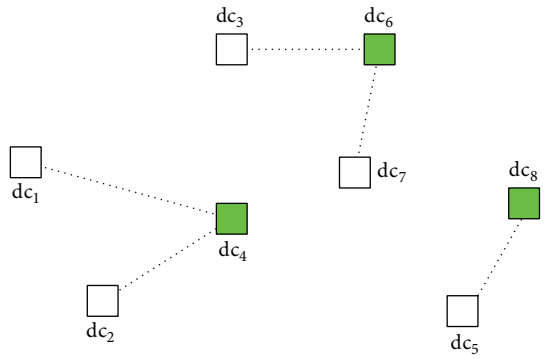

Figure 9: Result of replicas store places $\{dc_4, dc_6, dc_8\}$.

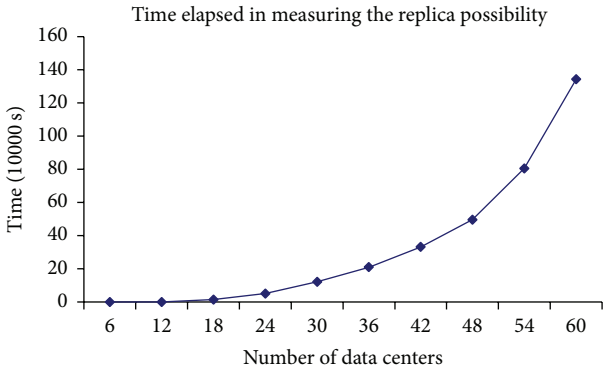

Figure 10: Elapsed time in measuring the replica possibility.

6.4. System Performance Comparisons of Replicas and No Replicas. The random simulations are conducted on randomly generated data sets of different sizes, generation times, and usage frequencies. In the simulations, we use a number of 50 data sets, each with a random size from 100 GB to 1 TB. And the usage frequency is also random from 1 to 10 times.

Simulation 1 (time complexity with one replica strategy). In the previous section, we have simulated the possibility of creating replicas in cloud platform. However, the algorithm used is exhaustive method, whose time complexity is very high. Figure 10 describes the elapsed time with increased number of nodes. From Figure 10, we can see that the time has a sharp increase. It is reasonable that we have to consider several data centers as the candidate place for replica.

Simulation 2 (comparisons of replicas numbers between MCRP and optimal strategy). Next, we evaluate the efficiency of MCRP strategy with the optimal solution in different number of usage frequencies. In the simulation, we assume that there is different number of data centers in cloud environment. And also data set usage frequency via each data center is random.

Figure 11 describes comparisons of the replicas numbers with different usage frequency, where we can see that the numbers beyond the optimal values are in tolerable limitations. Also, it is obvious that, with the increase of usage frequency, the amount of replica numbers increased quickly. The reason is that too much of data access leads to a big burden for network transmission in current conditions, such as network latency and bandwidth.

Simulation 3 (comparison of total cost between MCRP, optimal strategy, "each data center with one replica (ARS)," and "only original data set with no replica (NRS)" strategies). MCRP is a minimum-cost replicas placements strategy that can meet the system requirements under the condition of minimum overall data set cost. In this simulation, we will put the emphasis on the reduced cost between MCRP.

Figure 12 describes comparisons of total cost among different replicas placements strategies, where we can see that the total cost of MCRP is greatly decreased compared to other two straightforward strategies NRS and ARS. However, the total cost of MCRP is higher than the optimal strategy though the time complexity is lower.

From the above experimental and simulation results, the following conclusions can be drawn: (1) the proposed data sets replicas placements strategy effectively reduces the cost of application data set; (2) the proposed data replica strategy reduces the number of replicas; also (3) the proposed data replica strategy can effectively achieve system load balance by placing the popular data files according to the cost and user access history.

7. Conclusions and Future Works

In this paper we have investigated a model that simultaneously optimizes replicas placements from cost-effective view in the cloud. This model has a number of important applications in replication technology. Also, our current research, including experiments and simulations, is based on Amazon cloud's cost model, which can be reused by replacing the corresponding cost ratio. The data sets replicas placements strategy proposed in this paper is generic and dynamic, which can be used in any data intensive applications with different price models of cloud service. As presented in Section 5 and demonstrated in Section 6, minimum-cost replica placements strategy is close to the optimal cost benchmark, though it may not achieve the minimum cost. Therefore, it has great significance in theory and application explained as follows:

(1) The strategy proposed in this paper mainly focused on only one data set d_m, not all the data sets in a dynamic cloud computing system, which is simple and clear in practice. In this way, the popular data sets with high user frequencies are of particular concern, for the reason that they are important factors determining the system performance. However, for those data sets with low access frequency, even never used since generated, we have no need to consider their replicas.

(2) Considering the dynamic nature, which is the main metric of the cloud computing system, such as cost of transfer and storage and the user access frequency, the minimum-cost replica strategy is still available and

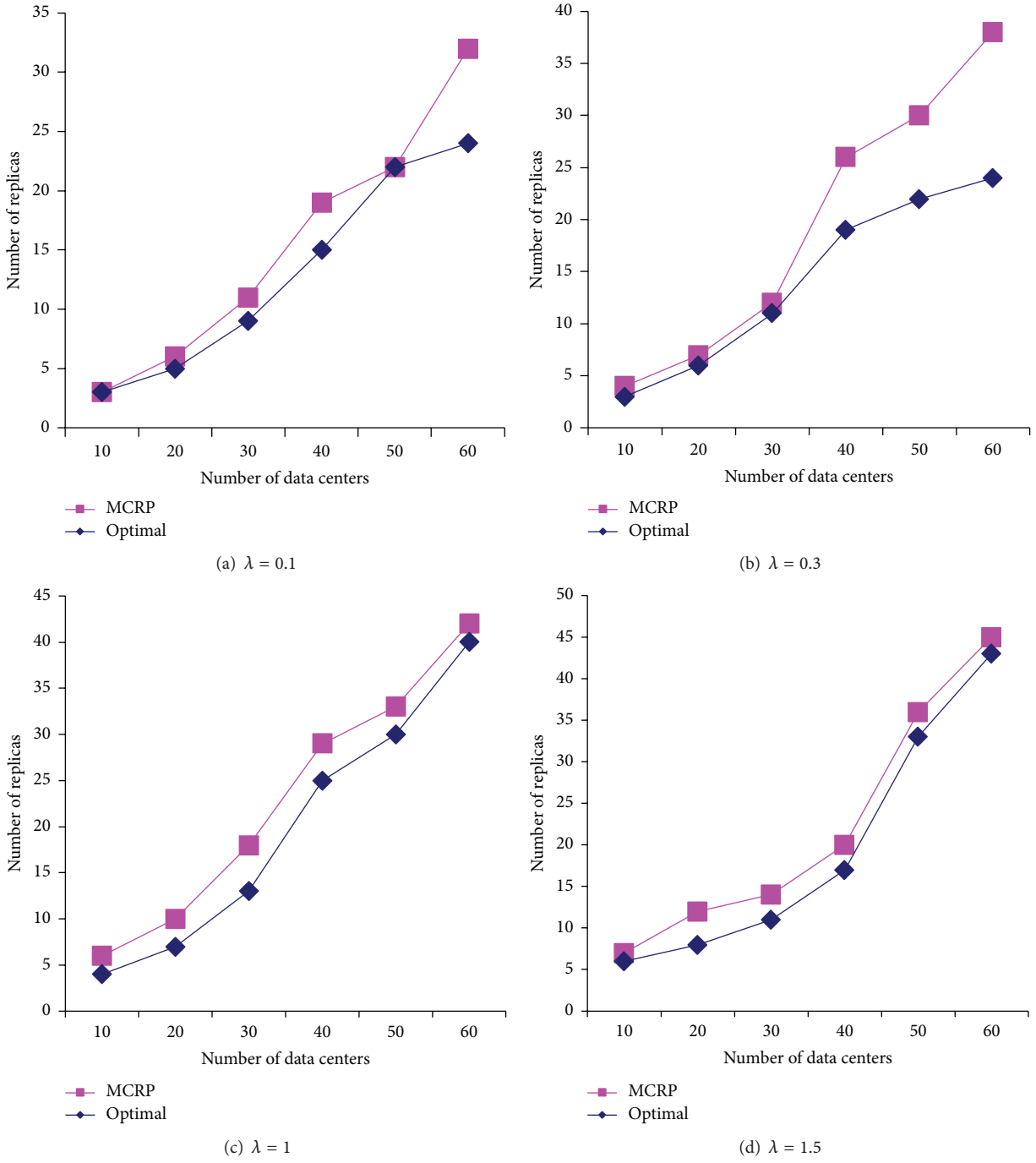

Figure 11: Comparisons of replicas numbers with different usage frequency.

effective, since we focus on a period of time T, not a time point t. In this way, the strategy is still applicable as long as we know the total cost in the past time.

(3) The replica deletion and maintenance strategies are also easy to obtain from the minimum-cost replica strategy. The basic idea is that when comparing the total cost with replica and the cost of no replica, then delete replicas once the cost with replicas is greater than no replica. On the other hand, we can update the replicas stored places according to the minimum-replicas strategy at scheduled time intervals.

Furthermore, experimental results and analysis show that the proposed strategy in cloud environment is feasible, effective, and universal. Hence, we deem that it is highly practical as a replica strategy. However, this paper presents the first attempt to apply the technique to solve the problem as how to place data sets replicas in the most appropriate data centers in the cloud from the minimum-cost view. It must

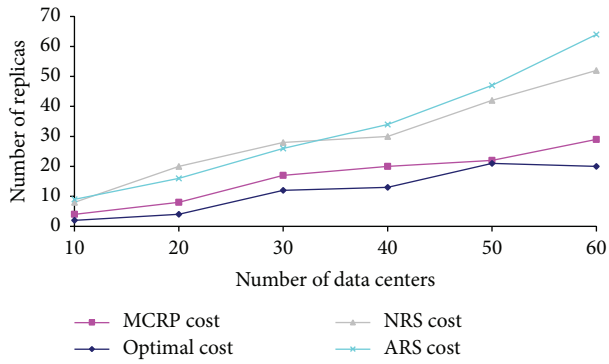

Figure 12: Comparisons of total cost among different strategies.

be kept in mind that these findings are the results of a preliminary study. To be more useful in practice, future works can be conducted from the following aspects:

(1) The current work in this paper has an assumption that the data set's usage frequencies are obtained from the system log files. Models of forecasting data set usage frequency can be further studied, with which our benchmarking approaches and replicas strategies can be adapted more widely to different types of applications.

(2) The replicas placements strategy should incorporate the data set generation, and deduplication technology, especially data content based deduplication technology, which is a strong and growing demand for business to be able to more cost-effectively manage big data while using cloud computing platforms.

Competing Interests

The author declares having no competing interests.

Acknowledgments

Some experiments in this paper were done on the cloud platform of SwinDeW-C in Swinburne University of Technology during Xiuguo WU's visiting period, which is sponsored by Shandong Provincial Education Department, China. Moreover, we want to show thanks to Doctor Dong Yuan and Professor Yun Yang, Swinburne University of Technology, Australia, for their valuable feedback on earlier drafts of this paper. In addition, this work presented in this paper is partly supported by Project of Shandong Province Higher Educational Science and Technology Program (no. J12LN33), China; the Doctor Foundation of Shandong University of Finance and Economics under Grant no. 2010034; and the Project of Jinan High-Tech Independent and Innovation (no. 201303015), China.

References

[1] Amazon S3, http://aws.amazon.com/s3.

[2] Google cloud storage, https://cloud.google.com/storage/.

[3] Windows Azure, https://www.azure.cn/.

[4] C.-H. Zuo, Z.-D. Lu, and R.-X. Li, "Load balancing in peer-to-peer systems using dynamic replication policy," *Journal of Chinese Computer Systems*, vol. 28, no. 11, pp. 2020–2024, 2007.

[5] T. Amjad, M. Sher, and A. Daud, "A survey of dynamic replication strategies for improving data availability in data grids," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 337–349, 2012.

[6] A. Martínez, F. J. Alfaro, J. L. Sánchez, F. J. Quiles, and J. Duato, "A new cost-effective technique for QoS support in clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 12, pp. 1714–1726, 2007.

[7] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proceedings of the Grid Computing Environments Workshop (GCE '08)*, pp. 1–10, IEEE, Austin, Tex, USA, November 2008.

[8] Y. Yang, K. Liu, J. Chen et al., "Peer-to-peer based grid workflow runtime environment of SwinDeW-G," in *Proceedings of the IEEE International Conference on e-Science and Grid Computing*, pp. 51–58, Bangalore, India, December 2007.

[9] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[10] S. Bansal, S. Sharma, I. Trivedi et al., "Improved self fused check pointing replication for handling multiple faults in cloud computing," *International Journal on Computer Science & Engineering*, vol. 4, no. 6, pp. 1146–1152, 2012.

[11] I. Arrieta-Salinas, J. E. Armendáriz-Iñigo, and J. Navarro, "Classic replication techniques on the cloud," in *Proceedings of the 7th International Conference on Availability, Reliability and Security (ARES '12)*, pp. 268–273, IEEE, Prague, Czech Republic, August 2012.

[12] W. Lloyd, M. J. Freedman, M. Kaminsky et al., "Don't settle for eventual: scalable causal consistency for wide-area storage with COPS," in *Proceedings of the 23rd ACM Symposium on Operating Systems Principles*, pp. 401–416, ACM, Cascais, Portugal, October 2011.

[13] W. Hoschek, J. Jaen-Martinez, A. Samar et al., "Data management in an international data grid project," in *Grid Computing—GRID 2000: First IEEE/ACM International Workshop Bangalore, India, December 17, 2000 Proceedings*, vol. 1971 of *Lecture Notes in Computer Science*, pp. 77–90, Springer, Berlin, Germany, 2000.

[14] D.-W. Chen, S.-T. Zhou, X.-Y. Ren, and Q. Kong, "Method for replica creation in data grids based on complex networks," *The Journal of China Universities of Posts and Telecommunications*, vol. 17, no. 4, pp. 110–115, 2010.

[15] X.-Y. Ren, R.-C. Wang, and Q. Kong, "Using optorsim to efficiently simulate replica placement strategies," *The Journal of China Universities of Posts and Telecommunications*, vol. 17, no. 1, pp. 111–119, 2010.

[16] M. Tu, P. Li, I.-L. Yen, B. Thuraisingham, and L. Khan, "Secure data objects replication in data grid," *IEEE Transactions on Dependable and Secure Computing*, vol. 7, no. 1, pp. 50–64, 2010.

[17] K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid," in *Grid Computing—GRID 2001*, C. A. Lee, Ed., vol. 2242 of *Lecture Notes in Computer Science*, pp. 75–86, Springer, Berlin, Germany, 2001.

[18] R.-S. Chang, J.-S. Chang, and S.-Y. Lin, "Job scheduling and data replication on data grids," *Future Generation Computer Systems*, vol. 23, no. 7, pp. 846–860, 2007.

[19] K. Sashi and A. S. Thanamani, "Dynamic replication in a data grid using a modified BHR region based algorithm," *Future Generation Computer Systems*, vol. 27, no. 2, pp. 202–210, 2011.

[20] K. Ranganathan, A. Iamnitchi, and I. Foster, "Improving data availability through dynamic model-driven replication in large peer-to-peer communities," in *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, p. 376, IEEE, May 2002.

[21] R. M. Rahman, K. Barker, and R. Alhajj, "A predictive technique for replica selection in grid environment," in *Proceedings of the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGRID '07)*, pp. 163–170, Rio de Janeiro, Brazil, May 2007.

[22] V. Andronikou, K. Mamouras, K. Tserpes, D. Kyriazis, and T. Varvarigou, "Dynamic QoS-aware data replication in grid environments based on data 'importance'," *Future Generation Computer Systems*, vol. 28, no. 3, pp. 544–553, 2012.

[23] K. Kalpakis, K. Dasgupta, and O. Wolfson, "Steiner-optimal data replication in tree networks with storage costs," in *Proceedings of the International Symposium on Database Engineering and Applications*, pp. 285–293, IEEE Computer Society, Grenoble, France, July 2001.

[24] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini, "Evaluation of an economy-based file replication strategy for a data grid," in *Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid '03)*, pp. 661–668, IEEE, May 2003.

[25] S. Melkote and M. S. Daskin, "An integrated model of facility location and transportation network design," *Transportation Research Part A: Policy and Practice*, vol. 35, no. 6, pp. 515–538, 2001.

[26] D. B. Shmoys, É. Tardos, and K. Aardal, "Approximation algorithms for facility location problems," in *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC '97)*, pp. 265–274, ACM, May 1997.

[27] S. Melkote and M. S. Daskin, "Capacitated facility location/network design problems," *European Journal of Operational Research*, vol. 129, no. 3, pp. 481–495, 2001.

[28] Y. Yang, K. Liu, J. Chen, X. Liu, D. Yuan, and H. Jin, "An algorithm in SwinDeW-C for scheduling transaction-intensive cost-constrained cloud workflows," in *Proceedings of the IEEE Fourth International Conference on eScience (eScience '08)*, pp. 374–375, IEEE, Indianapolis, Ind, USA, December 2008.

[29] J. Yan, Y. Yang, and G. K. Raikundalia, "SwinDeW-a p2p-based decentralized workflow management system," *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, vol. 36, no. 5, pp. 922–935, 2006.

[30] Y. Yang, K. Liu, J. Chen, J. Lignier, and H. Jin, "Peer-to-peer based grid workflow runtime environment of SwinDeW-G," in *Proceedings of the IEEE International Conference on e-Science and Grid Computin*, pp. 51–58, Bangalore, India, December 2007.

[31] D. Yuan, Y. Yang, X. Liu, and J. Chen, "A local-optimisation based strategy for cost-effective datasets storage of scientific applications in the cloud," in *Proceedings of the IEEE 4th International Conference on Cloud Computing (CLOUD '11)*, pp. 179–186, Washington, DC, USA, July 2011.