PAPER

Placement of Virtual Storages for Distributed Robust Cloud Storage

Yuya TARUTANI[†a)], **Yuichi OHSITA**[††b)], ***Members***, ***and*** **Masayuki MURATA**[††c)], ***Fellow***

SUMMARY Cloud storage has become popular and is being used to hold important data. As a result, availability to become important; cloud storage providers should allow users to upload or download data even if some part of the system has failed. In this paper, we discuss distributed cloud storage that is robust against failures. In distributed cloud storage, multiple replicas of each data chunk are stored in the virtual storage at geographically different locations. Thus, even if one of the virtual storage systems becomes unavailable, users can access the data chunk from another virtual storage system. In distributed cloud storage, the placement of the virtual storage system is important; if the placement of the virtual cloud storage system means that a large number of virtual storages are possible could become unavailable from a failure, a large number of replicas of each data chunk should be prepared to maintain availability. In this paper, we propose a virtual storage placement method that assures availability with a small number of replicas. We evaluated our method by comparing it with three other methods. The evaluation shows that our method can maintain availability while requiring only with 60% of the network costs required by the compared methods.

key words: *data center, cloud storage system, fault tolerance, redundancy*

1. Introduction

Cloud storage services have become popular, and a large amount of data is stored in them [1]–[4]. Cloud storage services are provided via datacenters, with a part of the storage space of the data center is allocated to each user. Users can upload or download their data by accessing the data center. The cloud storage service enables the users to access their data regardless of what devices, tools, and areas are used. In addition, by using the cloud storage services, the users can save the cost otherwise required to manage storage devices. Due to the above advantages, cloud storage services have become used both by personal users and companies.

Availability is important for cloud storage [5]–[7] especially that used by companies; the data should always be able to be accessed by the user. However, the storage in a data center may fail. In addition, a data center may become unreachable by the users due to network failure or congestion. Therefore, providers of cloud storage service should ensure availability even in such cases.

Manuscript received July 10, 2015.
Manuscript revised November 18, 2015.

[†]The author is with the Cybermedia Center, Osaka University, Toyonaka-shi, 560-0043 Japan.
[††]The authors are with the Graduate School of Information Science and Technology, Osaka University, Suita-shi, 565-0871 Japan.
a) E-mail: y-tarutn@cmc.osaka-u.ac.jp
b) E-mail: y-ohsita@ist.osaka-u.ac.jp
c) E-mail: murata@ist.osaka-u.ac.jp

DOI: 10.1587/transcom.2015EBP3292

One approach to ensuring the accessibility of data even in such cases is to prepare the replicas of data. Distributed file systems such as Google file system (GFS) [8] and the Hadoop file system [9] use this approach to keep the availability of the data. The distributed file systems divide the data into chunks, and store them in one of the storages. At the same time, replicas of each chunk are stored by the different storages. By doing so, users can access the data even when several storages become unavailable. By using this approach to store the replicas in the different data centers, availability can be maintained even if several data centers fail.

Though the approach of preparing replicas ensuring availability of the data so long as a sufficient number of replicas are prepared, more storages are needed and more bandwidth is consumed as the number of replicas increases. The required number of replicas depends on the locations of the data centers storing the replicas; a small number of replicas are sufficient if no possible failure in the network never can causes multiple data centers storing the replicas to become unreachable.

There are several studies that solves for placement of the data and the backups in a network including multiple data centers [5], [10]–[12]. This such approaches place a complete backup of the original data at one of the data centers, so that the backup can be accessed by the user if the data center storing the original data becomes unreachable. However, this approach incurs a large overhead: a large storage resource at a data center may be required to store a complete backup of the data. If one of the data centers fails and the additional data center storing the backup becomes required, all of the original data should have been sent to the additional data center.

In this paper, we discuss distributed cloud storage that maintains availability of the data, even in case of failures, without a large number of replicas. This method deploys *virtual storages* at multiple data centers for each user who requires availability even in case of failure. In the distributed cloud storage, data are divided into chunks, similar to existing distributed file systems. Then, by storing each chunk at multiple virtual storages, we ensure availability in case of failures. This approach makes the placement of the virtual storage flexible; each virtual storage holds only a part of the data. This also reduces the overhead required to place the additional virtual storage, because the size required for the virtual storage is not large.

In this paper, we also propose a method to determine

Copyright © 2016 The Institute of Electronics, Information and Communication Engineers

which data centers should host the virtual storages so as to ensure availability even in the case of any possible failures with a small number of replicas. In this method, we use the *split groups*, which are defined as sets of groups including the nodes belonging to the same connected subgraph, if failure occurs. The nodes belonging to the same split group can communicate with each other even if failure has occurred. Thus, our method guarantees the availability of the data by placing the virtual storages so that the split groups including the node connected to the user include more than $(N-k)$ virtual storages, where N is the number of deployed virtual storages and k is the number of replicas.

Our method determines the data center for hosting the virtual storages and the number of replicas by searching the suitable sets of the data centers for a small number of replicas. When we cannot find a suitable set of the data centers, we increment the number of replicas, and search the suitable sets again. By continuing these steps, we determine the data centers to host the virtual storages and ensure availability with a small number of replicas.

The rest of this paper is organized as follows. In Sect. 2, we explain the cloud storage system discussed in this paper. Section 3 presents a heuristic method for deciding which data centers should host the virtual storages. In Sect. 4, we evaluate our method by comparing it with other a methods without considering the required number of replicas. Finally, Sect. 5 provides our conclusions.

2. Distributed Cloud Storage

Figure 1 shows an overview of the distributed cloud storage discussed in this paper. The distributed cloud storage discussed in this paper is constructed of multiple data centers and the network between them. The provider of the storage service places multiple *virtual storages* for each user. Each virtual storage is hosted by one of the data centers using a part of the storage of the data center. The set of the virtual storages stores each user's data.

The data stored in the distributed cloud storage are divided into multiple small chunks, and each chunk is stored in k_u virtual storages. By doing so, we keep all fractions of data available unless all of the k_u virtual storages storing the chunk become unavailable simultaneously. Each chunk has an ID from which the virtual storage storing it is determined. By using the k_u hash functions, we can determine the set of virtual storages used to store the chunk.

To reduce communication delay from users and the amount of traffic on networks, the virtual storages are stored in the data centers that are close to users. However, doing so means that the number of unavailable virtual storage becomes large in the case of failure, because most of the virtual storages are concentrated in the data centers that are close to users. This causes an increase in recovery time and network overhead for recovering the robustness. In this paper, the data centers hosting the virtual storages are determined on the basis of the impact of failures to reduce the recovery time and the network overhead for recovering the robustness.

In the rest of this section, we explain the operation on the distributed cloud storage.

2.1 Access by Users

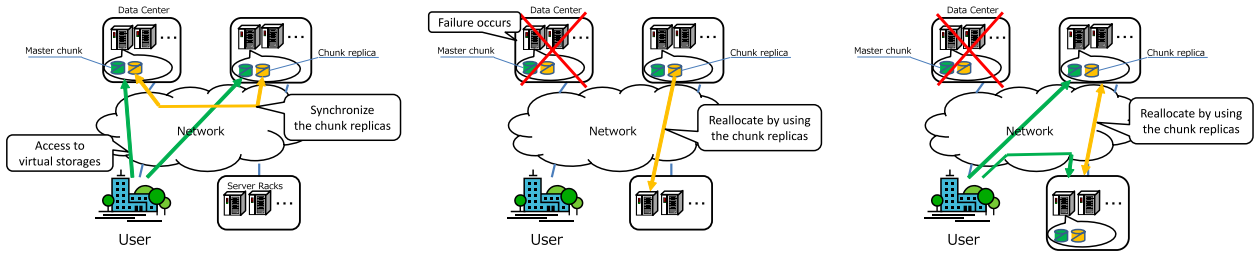
User access the their data on the distributed cloud storage through client software. The client software knows the hash functions, allowing it to obtain the virtual storage storing the chunk.

Data items are downloaded by downloading all the chunks required to construct the data. For each chunk, the client software calculates the hash functions to obtain the lists of the virtual storages where the chunk is stored. Then, the client software selects one of them, and downloads the chunk from the selected one.

Similarly, when uploading data, the client software uploads all the chunks included in the data. Each chunk is uploaded to one of the virtual storages corresponding to the ID of the chunk, as calculated by the hash functions. Then the virtual storage copies the chunk to the other virtual storages corresponding to the ID of the chunk.

2.2 Management of the Set of Virtual Storages

The set of virtual storages for each user is managed by the central manager. The central manager knows the location of the users and data centers in the network, the remaining resources of the data centers, the available bandwidth of the links in the network, and the set of nodes or links that are



(a) Normal condition (b) Failure occurs (c) Repairing the Cloud Storage

Fig. 1 Distributed robust cloud storage.

could fail simultaneously. Using this information, the central manager controls the locations of the virtual storages so as to keep the availability even in case of the failures.

After placing the virtual storages, the central controller monitors the set of virtual storages. If the set of virtual storage does not satisfy the requirements due to one or more virtual storages becoming unavailable, the central controller detects this, and new virtual storages are placed. The chunks stored in the new virtual storages are obtained from the other active virtual storages; the virtual storages storing the chunks required to be stored by the new virtual storages can be obtained by the hash functions, and then the new virtual storages send a request to the active virtual storages.

3. Placement of the Virtual Storages

In this paper, the central controller decides the data centers to host the virtual storages for each user. The placement of the virtual storages has an impact on the number of required replicas. Because the number of required storages and the bandwidth between data centers both increase as the number of replicas becomes large, the central controller should determine the data centers hosting the virtual storages so as to keep ensure availability without requiring a large number of replicas, and then determine the number of replicas needed. In this section, we formulate the problem to decide the data centers to host the virtual storages, and propose a heuristic method to solve the problem.

3.1 Problem Formulation

3.1.1 Input

The central controller knows the network topology. The network is represented as a graph G. We denote the set of nodes by N and the set of edges by L. Each edge l has available bandwidth b_l^{avail}. There are multiple data centers that can host the virtual storages, and each data center is connected to the network. We denote the set of data centers by D. We denote the storage space provided by the data center $d \in D$ by p_d. Each user is also connected to at least one of the nodes in the network. We denote the set of users by U, and the set of the nodes connected to the user u by N_u.

A path on the graph G can be represented as the set of links in the path. There may be multiple paths between any two nodes. We denote the set of paths between the nodes a and node b by $r_{a,b}$, and the set of all paths on the graph G by R. Among the paths included in $r_{a,b}$, we denote the shortest one by $r_{a,b}^{\text{short}}$.

The central controller also knows the possible patterns of failures. A set of links that could fail simultaneously is called a shared risk group (SRG). We denote the set of SRGs by F, and the set of the nodes included in the SRG f by N_f^{fail}. The central controller receives requests from users. Each request includes the number of required virtual storages C_u, the traffic rate from the user uploading data b_u^{upload} and the traffic rate to the user downloading data b_u^{download}.

3.1.2 Variables

We determine the data centers hosting the virtual storages for the user u by setting a variable $M_{n,u}$ that indicates the number of virtual storages for the user u hosted by the node n. In this problem, in addition to determining the data centers to host the virtual storages, we also determine the number of replicas. We denote the number of replicas for the user u by k_u.

In the rest of this paper, we set the size of each virtual storage to 1 in order to simplify discussion. As a result, the total number of virtual storages for the user u is $(1 + k_u)C_u$. In addition to the above variables, we also define a variable b_l indicating the traffic amount from the distributed cloud storage over the link l.

3.1.3 Objective

In this paper, we minimize the network cost defined as the total bandwidth used by the distributed cloud storage.

$$\text{minimize} \sum_{l \in L} b_l. \tag{1}$$

3.1.4 Constraints

- The total number of virtual storages hosted by data centers should be $(1 + k_u)C_u$

$$\forall u \in U\colon \sum_{n \in D} M_{n,u} = (1 + k_u)C_u.$$

- Each data center must have sufficient resources to host all of the assigned chunks, including chunk replicas.

$$\forall n \in D\colon \sum_{u \in U} M_{n,u} \le U_n,$$

  where U_n is the maximum number of virtual storages in data center n.
- All data must be available even if any failure occurs. Because each chunk is stored by $k_u + 1$ virtual storages, all data is available unless more than k_u virtual storages become unreachable by the user u. That is,

$$\forall u \in U, \forall f \in F\colon N_{u,f}^{\text{unreach}} \le k_u,$$

  where $N_{u,f}^{\text{unreach}}$ is the number of virtual storages that become unreachable from the user u when the set of nodes is included in N_f^{fail}, which is calculated by

$$N_{u,f}^{\text{unreach}} = \sum_{n \in \{n | n \in D, \forall r \in r_{N_u,n}, \exists m \in N_f^{\text{fail}}\colon m \in r\}} M_{n,u}$$

- b_l is the sum of the traffic passing the link l. In this distributed cloud storage, data are divided into multiple small chunks, and chunks are stored in separate virtual storages. Therefore, we assume that users access all

virtual storages at the same rate. The traffic rate from the user u to each data center with virtual storage is denoted by $\frac{b_u^{\text{upload}}}{C_u}$ where b_u^{upload} is amount of traffic in the worst case. Similarly, the traffic rate from each data center with virtual storage to the user u is denoted by $\frac{b_u^{\text{download}}}{C_u}$ where b_u^{download} is the amount traffic in the worst case. Each uploaded chunk is copied to k_u virtual storages. In this paper, we assume that the virtual storage hosting a replica is selected uniformly randomly from the possibilities, because the calculation overhead for solving the placements of replicas is avoided by assuming this. This problem is considered in future work. Thus, the traffic rate between the data center with the virtual storages of the user u is $\frac{b_u^{\text{upload}}}{C_u}$. That is,

$$b_l = \sum_{u \in U, n \in D, l \in r^{\text{short}}_{N_u, n}} \left(\frac{M_{n,u}(b_u^{\text{upload}} + b_u^{\text{download}})}{C_u} \right) + \sum_{u \in U, n_1, n_2 \in D, l \in r^{\text{short}}_{n_1, n_2}} \left(\frac{M_{n_1,u} M_{n_2,u} b_u^{\text{upload}}}{C_u} \right)$$

- b_l should be less than the available bandwidth of the link l

$$\forall l \in L\text{: } b_l \leq b_l^{\text{avail}}.$$

3.2 Heuristic Method to Place the Virtual Storages

Solving the optimization problem formulated in Sect. 3.1 requires a large amount of time, because the problem involves 6 integer variables. Therefore, we propose a heuristic method to determine the data centers to host the virtual storages.

To determine the data centers to host the virtual storages, we use the *split groups*, with a split group defined as the set of groups including the nodes belonging to the same connected subgraph if failure occurs. Nodes belonging to the same split group can communicate with each other even if failure occurs. Thus, we can guarantee the availability of the data by placing the virtual storages so that the split group including the node connected to the user includes more than $|N| - k$ virtual storages.

In our method, the data centers to host the virtual storages are determined by the following steps. We first calculate the split group for each SRG in advance. Then, we decide the data center to host the virtual storages, deciding one by one by using the split groups. The rest of this section explains the details of the above steps.

3.2.1 Calculation of the Split Groups

The split groups for SRG f are obtained by following steps.

1. Obtain the set of nodes $S_{N_f^{\text{fail}}}$ that is connected to one of the links in N_f^{fail}
2. Construct the graph G', in which the links in N_f^{fail} are removed from the network G.
3. Calculate the route from the node $s_{f,1}$ to the node $s_{f,2}$ on the graph G' by using the Dijkstra algorithm, for all node pairs $s_{f,1}$ and $s_{f,2}$ included in $S_{N_f^{fail}}$
4. Construct groups so that the nodes $s_{f,1}$ and $s_{f,2}$ belongs to the same group only when a route between $s_{f,1}$ and $s_{f,2}$ is found in the previous step.

The calculation time to obtain the split group for each SRG is $O(|N|^2)$. In a large-scale network, this calculation time becomes large, because the number of SRG increases. Therefore, our future work is to reduce the calculation time to obtain the split group for each SRG by combining several SRG into one SRG.

3.2.2 Placement of the Virtual Storages

Figure 2 shows the flowchart to determine the data centers to host the virtual storages for the user u. In these steps, we first set k_u to 1 and search for a suitable set of data centers to host the virtual storages such that the set ensures availability of the data for all SRGs even when k_u is 1, to minimize the number of required replicas. Then, if we cannot ensure availability of the data for all SRGs, we increment k_u, and search again for a suitable set.

When searching for a suitable set of data centers, we determine the data centers to host virtual storages one by one. The candidate data centers for hosting a virtual storage are checked in ascending order of the network cost $B_{M,u}(d)$

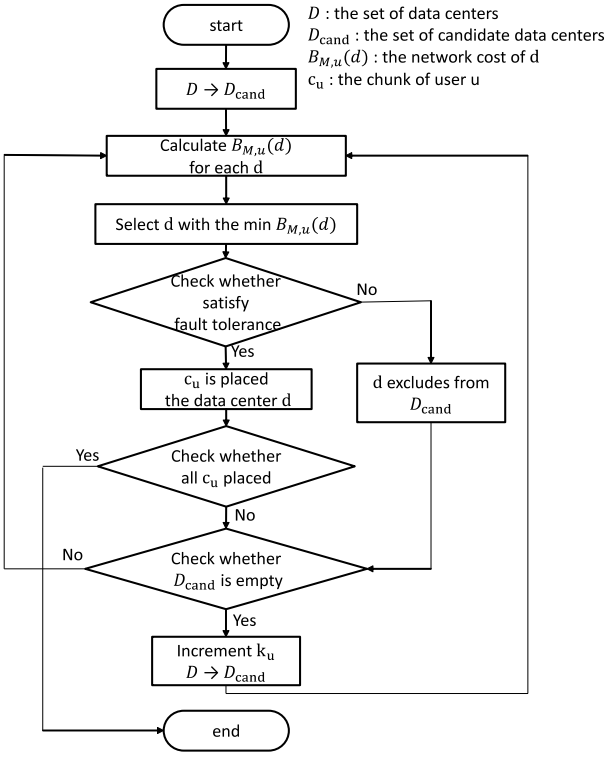


Fig. 2 Flowchart of our method.

caused by the traffic from/to the data center and characterized as

$$B_{M,u}(d) = (b_u^{\text{upload}} + b_u^{\text{download}})|r_{n_u,d}| + \sum_{n \in D} M_{n,u} k_u b_u^{\text{upload}} |r_{n,d}|. \quad (2)$$

where $M_{n,u}$ is the number of virtual storages whose locations are already decided as the data center n, and n_u is the node corresponding to the user u in the graph G. By selecting the data center with the smallest $B_{M,u}(d)$, we avoid a large network cost.

When checking whether a data center is suitable for hosting the virtual storage, we check that the constraints on availability are not violated. We can ensure availability unless more than k_u virtual storages become unreachable by the user. That is, for all of the SRGs in F, the number of virtual storages hosted by the data centers belonging to split groups not belonged to by the user should be less than k_u, so as to ensure availability in case of failure.

Therefore, in our method, we count the number of virtual storages hosted by data centers in a different split group from the user for each of the SRGs. Then, if the number exceeds k_u, we regard the selected data center as unsuitable for hosting the virtual storage, and eliminate it from the candidate data centers.

The computational complexity for selecting the data centers to host the virtual storage of user u is $O(|C_u|)$.

4. Evaluation

In this section, we evaluate our method, and demonstrate the advantage of the placement of the virtual storages considering the impact of failure.

4.1 Evaluation Environment

4.1.1 Network Topology

In this evaluation, we use the Japan Photonic Network Model (JPNM), which is a model of the network in Japan [13] shown in Fig. 3. This network topology is likely to form

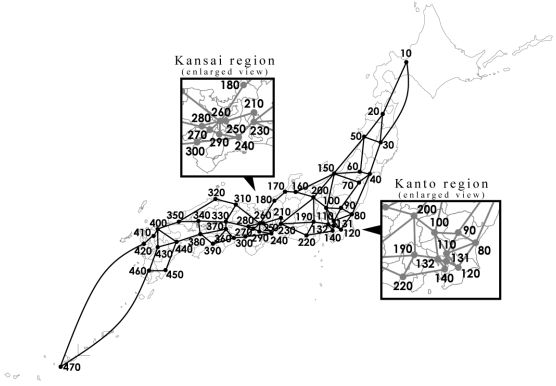


Fig. 3 Japane photonic network model.

subgraphs on failure. In this case, the impact of failure depends on the location of the data centers. Therefore, in this evaluation, we place 8 data centers whose locations are selected to be close to users, and to be far from users. The number of virtual storages of possible at each data center is set to 1000, and the bandwidth of each link is set to 10 Gbits per second. The size of virtual storage is set to 500 GB.

In this evaluation, the SRG is created so that the failure divides the network into two subgraphs. The SRG is given by the following steps for each data center.

1. Select the closest data center d to a data center s.
2. Add the links on the shortest path between s and d to the SRG.
3. Remove the links on the shortest path between s and d.
4. Check the whether the path exists. If yes, go back to Step 2, otherwise end.

4.1.2 Request from Users

We assume that users are in Tokyo or Osaka, and with equal number of users in Tokyo and Osaka. Each user is connected to two nodes in the network, because users can access their virtual storages in the case of failure. We connect the users in Osaka to nodes 260 and 270, and connect users in Tokyo to nodes 131 and 132. The number of users at each location is set to 25. The size of the storages requested by each user is set to an integer value from 5 to 20 chosen uniformly randomly. Each user uploads and downloads 10 Mbit per second.

4.1.3 Evaluation Metrics

In our evaluation, we investigated the resource required by the distributed cloud storages. The network resources required as the distributed cloud storages are estimated by the network cost defined by Eq. (2). The resources of the data centers are estimated as the size of the storages allocated for use in the distributed cloud storage, which is calculated as

$$C^{\text{all}} = \sum_{u \in U} C_u(k_u + 1). \quad (3)$$

Moreover, we evaluate the availability and restoration time for recovering robustness. The availability of data to user u is guaranteed by placing the virtual storages so that the split groups that include the node connected to the user have more than $|C_u| - k_u$ virtual storages. Thus, the availability of data to user u is denoted by the following;

$$\text{Availability} = \begin{cases} 100 & (C_u - k_u \le N_{u,f}^{\text{reach}}) \\ 100 \frac{N_{u,f}^{\text{reach}}}{C_u - k_u} & (Otherwise) \end{cases} \quad (4)$$

where $N_{u,f}^{\text{reach}}$ is the number of virtual storages that remain reachable by user u when the set of nodes are included in N_f^{fail}.

The data centers hosting the virtual storages for recovery are also determined by using the method to determine

the data centers to host the initial virtual storages. In the case of restoration, the virtual storages are sent to data centers by using the available bandwidth of links. In this evaluation, we assume that the delay depends on the amount of data sent, because the transmission delay is much larger than the delay from distance. Thus, the restoration time t_{rest} is denoted by the following;

$$t_{\text{rest}} = \frac{V_d}{\min_{l \in r_{s,d}^{\text{short}}} b_l^{\text{avail}}} \tag{5}$$

where V_d is the amount of data sent and $r_{s,d}^{\text{short}}$ is the shortest path from the source data center s to the restoring data center d.

4.1.4 Compared Methods

In this evaluation, we compare the proposed method with three other methods that guarantee the availability of the data by placing virtual storages. In this paper, we call these methods *mirroring method*, *random method*, and *last determining method*.

The mirroring method is similar to a method described in Ref. [5]. This method guarantees the availability of the data by preparing the backup virtual storages. In this method, the service provider selects the data center hosting the virtual storages within the shortest distance from a user. To ensure the availability of the data, another data center is selected to host backup virtual storages. In this paper, the mirroring method selects a data center that users can access their virtual storages or their backup virtual storages in the case of failure.

The random method is similar to the GFS method [8]. In this method, we determine the data centers to host the virtual storages randomly. Moreover, the number of replicas is a constant value independent of the placement of virtual storages. In this paper, we set the number of replicas to one quarter the number of required virtual storages.

Finally, the last determining method determines the number of replicas after determining the data centers to host the virtual storages. In this method, we first determine the data centers to host the virtual storages so that the network cost is minimized. This can be calculated by using the method described in Sect. 3.2.2 with setting b_u to a sufficiently large value C_u. Then, b_u is set as the minimum value that does not require violating the availability in any case of the SRG. Finally, the number of virtual storages in each data center is adjusted so as to minimize the used storage size under the constraint that b_u replicas per each chunk can be stored.

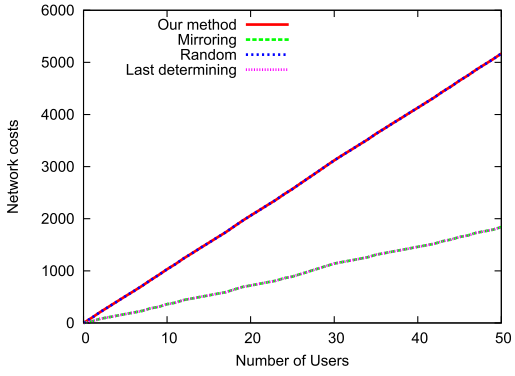
The purpose of the last determining method is to minimize the network cost for the user to access the data. However, this method does not consider that the network cost is increased by synchronizing the replicas. Therefore, our evaluation shows the advantage of determining the placement of virtual storage with considering the number of replicas.

4.2 Results

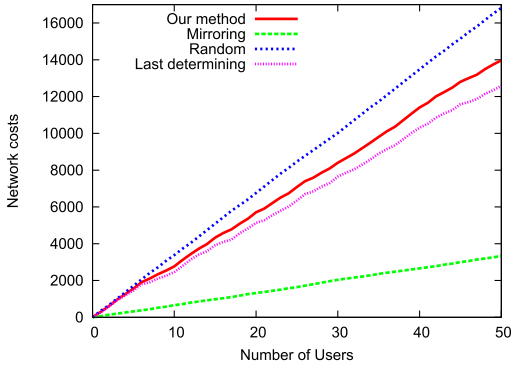
4.2.1 Cost of Networks

Figure 4 shows the network cost of our method and the compared methods in the case where the data centers are placed far from users. In these figures, the horizontal axis is the number of users, and the vertical axis is the network cost calculated by Eq. (2). Figure 4(a) shows the network cost caused by the communication between users and data centers. This figure indicates that the mirroring method and the last determining method achieve lower network cost than our method and the random method. This is because that the mirroring method and the last determining method deploy virtual storages near the users, while our method and the random method deploy virtual storages at the data center far from the users, too, to ensure availability in case of failure. As a result, the number of hops from the user to the data centers is large in our method and the random method, compared with the mirroring method and the last determining method.

However, Fig. 4(b) shows that the total cost of our method is similar to that of the last determining method.

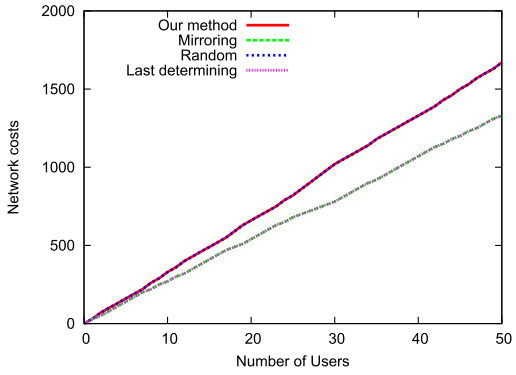


(a) Users

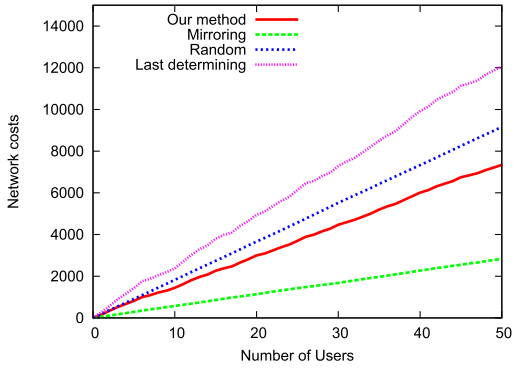


(b) Total

Fig. 4 Network cost for data center placed far from users.



(a) Users



(b) Total

Fig. 5 Network cost for data center placed near users.

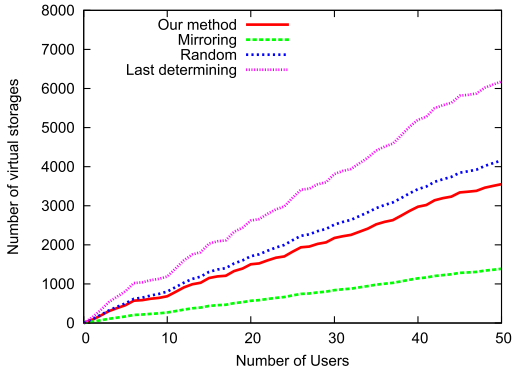


Fig. 6 Number of virtual storages.

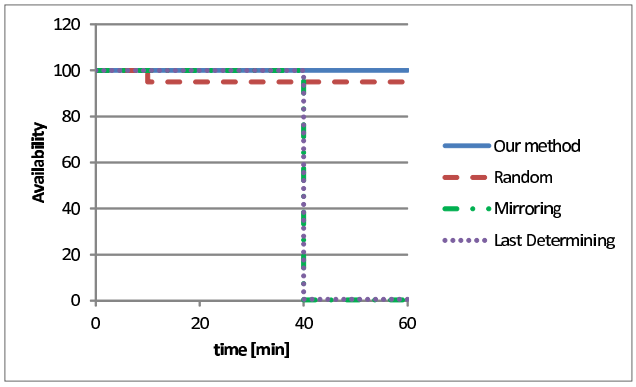


Fig. 7 Availability of data.

This is because the last determining method requires more replicas to ensure availability. As a result, the data centers send a large amount of data to keep the replicas updated when the last determining method is used, which causes a high network cost. In contrast, Fig. 4(b) also shows that the total cost of the mirroring method is smaller than that of other methods. This is because, in the mirroring method, the number of backup virtual storages is smaller than in other methods. As a result, the amount of traffic used in guaranteeing availability is small.

Figure 5 shows the network cost of our method and the compared methods in the case where the data centers are placed near users. In these figures, the horizontal axis is the number of users, and the vertical axis is the network cost calculated by Eq. (2). Figure 5(a) indicates that the mirroring method and the last determining method achieve lower network cost than our method and the random method, but the difference is smaller than in the case shown in Fig. 4. This is because data centers are chosen near users. As a result, the number of hops from users to data centers is small in our method. Therefore, our method is more suitable in this case. Figure 5(b) also shows that the total cost of our method is lower than that of the last determining method. This is because, the number of replicas used in our method is smaller than in the last determining method, and the number of hops between data centers is small. As a result, our method can ensure availability while requiring only 60% of the network costs required used in the last determining method.

Figure 6 shows the total number of virtual storages. In this figure, the horizontal axis is the number of users, and the vertical axis is the total number of virtual storages. Figure 6 shows that the total number of virtual storages used by our method is much smaller than that used by the last determining method. This is because our method decides the data centers to host the virtual storages so that a large number of virtual storages never become unavailable simultaneously. As a result, in our method and the random method, a small number of replicas is sufficient to ensure availability in the case of failures, which leads to a reduction in required resources.

4.2.2 Availability

We evaluate the availability in the case of failure. In this evaluation, a randomly selected SRG is set as failed. The time of failure is 10 minutes and 40 minutes. In this evaluation, the number of cases is set to 20, and the result shows the worst case. Figure 7 shows the result. In this figure, the horizontal axis is the time, and the vertical axis is the availability defined as Eq. (4). Figure 7 indicates that our method, the mirroring method and the last determining method en-

Table 1 Restoration time.

Method	maximum restoration time
Proposed method	27 minutes
Mirroring	133 minutes
Random	30 minutes
Last Determining	67 minutes

sure availability of data in the case of one failure. In contrast, the random method dose not guarantee availability of data, because the number of replicas is too small for to guarantee availability.

However, the mirroring method and the last determining method cannot guarantee the availability of data in the case of a second failure. Table 1 shows the time for restoring the robustness. The restoration time of these methods is large because size of the data to be sent is large in these methods. Therefore, the mirroring method and the last determining method are not suitable when multiple failures are possible. On the other hands in contrast, our method can also ensure availability of data in this case. This is because the number of virtual storages not available to users is small. As a result, the restoration time is small.

5. Conclusion

In this paper, we discuss a distributed cloud storage system that ensure availability of the data even in case of failure without a large number of replicas. We also propose a method to determine which data centers should host the virtual storages so as to ensure availability even in the case of any possible failures using only a small number of replicas. In our method we use split-groups, which are defined as groups including the nodes belonging to the same connected subgraph after a failure occurs. In this method, we calculate the split-group for each node, which is the group that the nodes are grouped into according to connectivity when the node fails. We evaluated our method by comparing it with three other methods. The evaluation showed that our method ensures availability while requiring only 60% of the network costs required by other methods that do not consider the required number of replicas.

In this paper, the placement of chunk replicas is performed randomly. We expect that the number of chunk replicas and the network costs can be reduced by considering the placement of chunk replicas. Moreover, in our method, the split-group must be recalculated after a failure. Therefore, reducing the calculation time is task for future work in the large-scale networks.

Acknowledgment

This work was supported in part by the National Institute of Information and Communications Technology (NICT) of Japan.

References

[1] “Google Drive,” https://www.google.com/intl/en/drive/
[2] “Dropbox,” https://www.dropbox.com/
[3] “Microsoft OneDrive,” https://onedrive.live.com/
[4] “box,” https://www.box.com/
[5] A. Xiao, Y. Wang, L. Meng, X. Qiu, and W. Li, “Topology-aware virtual network embedding to survive multiple node failures,” Proc. 2014 IEEE Global Communications Conference, pp.1823–1828, Dec. 2014.
[6] K.D. Bowers, A. Juels, and A. Oprea, “HAIL: a high-availability and integrity layer for cloud storage,” Proc. 16th ACM Conference on Computer and Communications Security, CCS’09, pp.187–198, 2009.
[7] B. Calder, J. Wang, A. Ogus, N. Nilakantan, A. Skjolsvold, S. McKelvie, Y. Xu, S. Srivastav, J. Wu, H. Simitci, J. Haridas, C. Uddaraju, H. Khatri, A. Edwards, V. Bedekar, S. Mainali, R. Abbasi, A. Agarwal, M.F. ul Haq, M.I. ul Haq, D. Bhardwaj, S. Dayanand, A. Adusumilli, M. McNett, S. Sankaran, K. Manivannan, and L. Rigas, “Windows azure storage: A highly available cloud storage service with strong consistency,” Proc. Twenty-Third ACM Symposium on Operating Systems Principles, SOSP’11, pp.143–157, 2011.
[8] S. Ghemawat, H. Gobioff, and S.-T. Leung, “The Google file system,” SIGOPS Oper. Syst. Rev., vol.37, no.5, pp.29–43, Dec. 2003.
[9] “Apache hadoop project,” http://hadoop.apache.org/
[10] H. Yu, C. Qiao, V. Anand, X. Liu, H. Di, and G. Sun, “Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures,” Proc. 2010 IEEE Global Telecommunications Conference GLOBECOM 2010, pp.1–6, 2010.
[11] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, “Adaptive virtual network provisioning,” Proc. Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures, VISA’10, pp.41–48, 2010.
[12] P. Bodí k, I. Menache, M. Chowdhury, P. Mani, D.A. Maltz, and I. Stoica, “Surviving failures in bandwidth-constrained datacenters,” ACM SIGCOMM Computer Communication Review, vol.42, no.4, pp.431–442, 2012.
[13] “Japan Photonic Network Model,” http://www.ieice.org/˜pn/jpn/jpnm.html

Yuya Tarutani received an M.E. and a Ph.D. in Information Science and Technology from Osaka University in 2012 and 2014, respectively. He is currently an Assistant Professor at the Cybermedia Center, Osaka University. His research interests include traffic matrix estimation, data center networks and network reconfiguration. He is a Member of IEICE and IEEE.

Yuichi Ohsita received an M.E. and a Ph.D. in Information Science and Technology from Osaka University in 2005 and 2008, respectively. He is currently an Assistant Professor at the Graduate School of Information Science and Technology, Osaka University. His research interests include traffic matrix estimation and countermeasures against DDoS attacks. He is a Member of IEICE, IEEE, and the Association for Computing Machinery.

Masayuki Murata received an M.E. and a D.E. in Information and Computer Sciences from Osaka University in 1984 and 1988, respectively. In April 1984, he joined the Tokyo Research Laboratory of IBM Japan as a researcher. From September 1987 to January 1989, he was an Assistant Professor at the Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. From 1992 to 1999, he was an Associate Professor at the Graduate School of Engineering Science, Osaka University, and became a Professor at the same school in April 1999. He moved to the Graduate School of Information Science and Technology, Osaka University in April 2004. He has published more than 300 papers in international and domestic journals, and has given presentations at numerous conferences. His research interests include computer communication networks, as well as performance modeling and evaluation. He is a Fellow of IEICE and a Member of IEEE, the Association for Computing Machinery (ACM), The Internet Society, and IPSJ.