International Journal of Grid Distribution Computing
Vol. 8, No.6, (2015), pp.91-104
http://dx.doi.org/10.14257/ijgdc.2015.8.6.10

Dynamic Approach Based on Learning Automata for Data Fault-Tolerance in the Cloud Storage

Seyyed Mansour Hosseini, Mostafa Ghobaei Arani and Abdol Reza Rasouli Kenari

Department of Computer Engineering,
Mahallat Branch, Islamic Azad University, Mahallat, Iran
Department of Computer Engineering,
Parand Branch, Islamic Azad University, Tehran, Iran
Department of Computer Engineering,
Qom Branch, University of Technology Qom, Iran
mansourh64@gmail.com, mostafaghobaei@piau.ac.ir, rasouli@qut.ac.ir

Abstract

Regarding the increasingly expanded utility of Cloud storage, the improvement of resources management in the shortest time to respond upon the users' requests and the geographical constraints is of prime importance to both the Cloud service providers and the users. Since the Cloud storage systems are exposed to failure, fault-tolerance is appraised by Cloud storage systems' capability for responding to unexpected fault through software or hardware. This paper represents an algorithm based on Learning Automata–oriented approach to fault tolerance data in Cloud storage regarding traffic and query loads dispatched on data centers and learning automata that provides the best possible status for scaling up or down of data nodes. Based on appraisal of traffic on nodes, the node with the highest traffic is chosen for coping among physical nodes. The experimental results indicate that the proposed Learning Automata Fault-Tolerant and High-efficient Replication algorithm (LARFH) has utilization high replication, high query efficiency, low cost and high availibility in comparison with other similar approaches.

Keywords: *Cloud Computation, Cloud Storage, Fault-Tolerance, Learning Automata*

1. Introduction

The distributed Cloud storage are increasingly exposed to hardware, power, and network failure, regarding sending request for getting access to resources [1-3]. For this reason, the need to fault-tolerance in Cloud storage is currently turning into a reliable dealing for providers of Cloud storage and leading to the users' trust and tendency towards the Cloud services. Nowadays, the big clod providers, Drop box, Box Net, Google Drive, pay due attention to rendering solutions to enhance Cloud storage [3]. However, not all terms of hosting are given to the third party. For instance, at the time of storage, the private sanctuary is extremely sensitive and influenced by the rules and regulations of the respective country. The users are not concerned with storage infrastructures but they can store and monitor their data on the Cloud storage providers from far distance. Fault-tolerance is one of the most important aspects of Cloud storage having to do with the stability of storage [4]. The data exposed to failure must be replicated and distributed in a couple of machines [2]. One of the wide spread method of fault-tolerance is coping that is increasing applying in the distributed Cloud storage systems. In coping data, to prevent failure, the probability break down of all data in all data centers are employed. However, the majority of the existing Cloud storage systems reproduce any kind of data based on the number of physically distinct nodes in a static way, irrespective of considering the geographical distance and Replication.

ISSN: 2005-4262 IJGDC

More importantly, none of them never pay much attention to design from a traffic-based vantage. For example, hot spot change, flash crowd, lookup skew, and load imbalance are significant factors that influence users' experience and create a problem for replication procedure. The ultimate goal in fault-tolerance of Cloud computation is to retrieve failure, spend less money, and achieve the improved standards in performance. Fault-tolerance via reproduction in Cloud diverse distributed geographical resources is deployed to minimize the rental costs of corresponding storage and communication. If to a partition so many queries dispatched in short-time intervals, that partition is not able to respond the clients properly. On the other hand, if the partition is seldom accessed, it is the waste of resources, causing maintenance overhead. Regarding the previous methods that have rather high costs, this paper renders an algorithm for fault-tolerance using Learning Automata Fault-Tolerant and High-efficient Replication (LARFH henceforth). This algorithm provides a high efficiency rate of replication, high efficacy of query plus rational length of route along with low expenditure and higher availability. For achieving the higher availability, the data are replicated on the nodes with the maximum replication traffic and via replication algorithm and changing the trend of number of copies dynamically, higher availability are made feasible for clients. Therefore, LARFH represents a resilient, fault-tolerant, and high-efficient replication algorithm for Cloud storage systems.

The rest of this paper is organized as follows: Section 2 is allocated to the relevant works. In Section 3, present proposed approach. Finally, the appraisal and findings of the proposed simulation approach and the conclusion as well as further work are presented in Sections 4 and 5 respectively.

2. Related Works

Many researches have been conducted in the domain of fault-tolerance storage. Most of them were based on the replication of data with high availability and some of them based on insure code and some combined the forenamed methods. Following are the details of some of techniques done.

T (1): A. Kumar *et al.* rendered the reliable methods for error diagnosis and fault-tolerance in Cloud storage systems. Their focus was on the different errors types occurred in a system, the diagnostic and recovery techniques applied. To tolerate any error, it is necessary to discern and then separate it with the appropriate unit with the upmost speed. The main mechanisms of discovery are rational monitoring, observational monitoring, protocol error, diagnosis within service, and the counter of transient error of packages. If one unit is defective, many error motives for that unit will be generated. The major aim of error separation is to make the error motives related and recognition of the defective unit [5].

T (2): B. Meroufel *et al.* offered an approach for dynamic replication in a hierarchical network considering failures and errors' system. It rested on the dynamic replication based on two parameters of data management, availability, and generalizability. A manager should determine a specific percentage for data availability in the system and if any replication does not satisfy this percentage, it must be taken into consideration as the minimum degree of repetition. The percent of availability may increase as the data transferability increases. This approach guarantees availability, even in the case of failure [6].

T (3): Qing Song *et al.* proposed cost effective dynamic replication management scheme (CDRM) for large-scale storage systems. This scheme devises a cost model for handling of replication factor. This factor entails the minimum of replication number in the metadata. Based on this model, the low extent of replication source, as necessarily available, can be specified. In addition to CDRM, among the distributed nodes, some

copies for prevention of blockage possibility improve load balance and general performance [7].

T (4): Zhang *et al.* offered Byzantine fault tolerance framework for building reliable system in voluntary resource Cloud infrastructure. The super Binary File Transfer (BFT) input consists of queries continuation with the specific Qualified Obligation Service (QoS) such as price priority, capability, bandwidth, work volume, response time delay, and failure possibility that is sent via the super modulation. The super Binary File Transfer (BFT) output is response continuation corresponding to the queries. The initial responsibilities are the acceptance of query from the super modulation, selection of proper copies for formation of a BFT, resending the query to the entire copies and substitution of the defective copies with the new selected nodes [8].

T (5): Z. Ghilavizadeh *et al.* devised *a new* fuzzy optimal data replication for data grids reducing the total performance of work done upon the recognition and the impact of effective factors on data replication, enhancing the location availability. The proposed method consists of two steps. First, reduce of the time availability to data in order to decrease delay availability, performance, and the occupied bandwidth. Second, it is substitution step. This step reduces the total time of executing grid sites with the aid of the replication factor and upgrades the status of availability and data replication on the network optimally [9].

T (6): Z.Wang et *al.* presented an algorithm for dynamic data replication strategy. This algorithm makes use of the past record of availability that is useful for replication of a file.Then applies a preventive omission method for controlling over the number of copies in order to strike an optimal balance between the updated reading overhead and writing. If amounts exceed from the threshold, the algorithm determines the replication. Different weightings are allocated to the record of availability in different spans of time for finding a common file [10].

T (7): A. D. Meshram *et al.* suggested fault tolerance model for reliable Cloud computing (FTMC) based on the appraisal of reliability from the computing nodes as virtual machines (VMs) inCloud setting and real-time program working fault-tolerance in those computed virtual machines (VMs). A virtual machine is chosen for computation based on its reliability and if does not properly execute the applied real-time programs, it can be deleted. In this model, there are *n* virtual machines (VMs) that execute *n* algorithm. There is an accepter whose function is to confirm final output of any node. Then all outputs are transferred to a timer that checks time span of each result. Next, all results are sent to a decision marker that chooses them based on the outputs' reliability. In addition, the decision marker asks from some responsible modules (resource manager or programmer) to delete a node with the less reliability and build a new node [11].

T (8): Y. Qu *et al.* have offered a resilient, fault-tolerant and high-efficient replication algorithm (RFH) for distributed Cloud storage. This algorithm encounters with flash crowd. Each part of data id rendered via a virtual node. Any virtual node by weighting to the positive and negative aspects determines to be replicated, migrated, or committed suicide. It rests on the traffic load appraisal of all the nodes and for replication or migration among the physical nodes, a node with the most traffic will be selected. In RFH algorithm for per period, any node computes availability. If at least availability for the initial partition is not obtained, it replicates on its upmost forwarding node even if not all the nods are loaded. If minimum availability is achieved, but still a lot traffic exist, the algorithm is forced to reduce the load [12].

3. The Proposed Approach

One of the challenges in the dynamic approach to data fault-tolerance in Cloud storage is the appropriate time for conducting the replication operation on resources in the Cloud datacenters. To this end, for determining the time of conducting the replication operation,

the learning automata is utilized as a decision marker. In this approach, focus is placed on the traffic reduction in the datacenters so that the sent queries to these datacenters are responded swiftly. To decrease traffic and better response to the queries, the data copies on the data centers with the less traffic will be conducted. The general configuration required is traffic reduction, the optimal data replication and the dynamic on the datacenters fault-tolerance. In this section, a brief description of the learning automata is elaborated, and then present proposed LARFH algorithm.

3.1. Learning Automata

A process, during which the living beings learn different things, has attracted experts' attention for a long time. The most important characteristics of a learning system is its capability to enhance efficiency over time. The purpose of such learning system is to optimize a performance that is no possibility to recognize it completely. In this section reviews the learning methods based on the random learning automata. A learning automaton can be defined as an abstract thing that has limited action. The Learning automata acts with selection of an action out of its own action set and execution on the setting. A random setting and automata appraise the said action after receiving response from setting decides upon the next action. During this process, learning automata to select the optimal action. The manner by which the response from setting for selective action lined with the next automata's action is specified via learning automata algorithm. The random environment is defined as in: learning automata α is collection of environment inputs and β is collection of outputs. Automata can be described as a chain of repetitive cycle in which automata an environment interact in to each other.

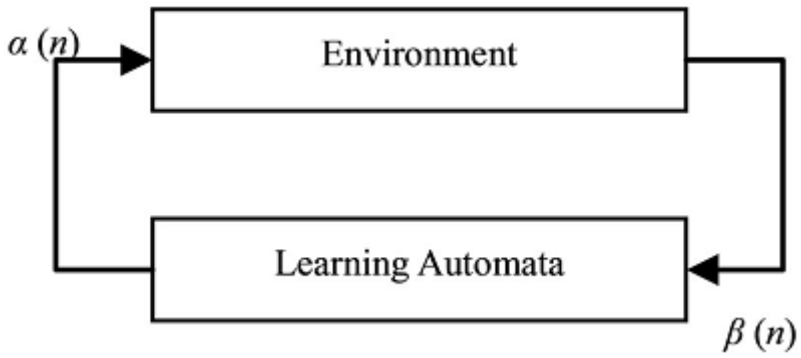


Figure 1. The Relation between Automata and Environment

As it can be seen in Figure 1, the relation between probable environments and automata has been shown.

It can be said that all learning algorithms are as fallow [13-17]; if a query in copy selects the N th of α_i action and receives an optimal replay from the environment, it would increase probabilities $P_i(n)$ related to action and decrease the related probabilities other action. In case of receiving on favorable replay, $P_i(n)$ decreases and the related probabilities related to other action increases. Thus, we have:

For the favorable replay

$$P_i(n+1) = P_i(n) + \sum_{\substack{j=1 \\ j=i}}^{r} f_j[P_j(n)] \tag{1}$$

$$P_j(n+1) = P_j(n) - f_j[P_j(n)];\ \forall j,\ j \neq i$$

For the unfavorable reply:

$$P_j(n+1) = P_j(n) - g_j[P_j(n)];\ \forall j,\ j \neq i$$
$$P_i(n+1) = P_i(n) + \sum_{\substack{j=1 \\ j=i}}^{r} g_j[P_j(n)] \quad (2)$$

3.2. The Proposed LARFH Algorithm

In this section, we offer a dynamic approach to data fault tolerance in Cloud storage based on learning automata and then evaluate the proposed approach based on reviewing the standard query traffic randomly (the amount send request for a partition, the amount of copying operation and migration by virtual node) under the flash crowd setting. In this proposed algorithm, we consider a traffic-monitoring component for computing the traffic on the datacenters that their ranges have short time interval and it will be shown US (usage of traffic). Namely, the availability to datacenters partition is reviewed at every moment by traffic monitoring. Any node, in case of high traffic on the neighboring partition is copied or migrated or in the case of lack of usage and request in other to preserve resources will suicide. If the minimum availability that is shown by $\boldsymbol{A_{expect}}$, for a holder initial partition has not achieved, it replicates on the most forwarding nodes, viz, the node that has the most requests, even though not all nodes are loaded. If the minimum availability is achieved, but there still exists much traffic, the scheme will be forced to reduce its load.

In order fulfill the least expected and required availability for a good experience of a client at the time of usage, the minimum number of copies is called *rmin*. The probability of scale up and down is 0.5 if the least availability in starting up that is required in other to compare the imposed traffic on the partitions if the amount of load threshold equals to 0.2 and we consider the high threshold 0.8. The trend of work will be as follow: in per 10-second epoch and algorithm on N time (100th range) the amount of the send requests for availability to partition is executed in which N consists of the number of virtual nodes. In any execution, the traffic datacenter and a minimum availability of the partition with the load threshold is compared and in case of more than of load threshold, it is rewarded. By any reward, the probability p_i scale up action increases and in case of lower than the low threshold, a penalty is considered, then the reduction of p_i and probability p_j scale down action increases. The probability amounts of (p_i , p_j) per new execution are changed an updated. In continuation and after N time load copy and in the last updated status of (US), one of the scale up and down action with highest probability by virtual node is selected.

This algorithm in case of chosen by scale up compares the traffic of the current datacenter and the virtual node with the higher threshold and if it is more than, the order is issued for copying on the nearest neighboring datacenter, otherwise the selected action is scaled down and it issues the lack of usage from the copy data and suicide order of virtual nodes as well as lack of copy. This section of algorithm is the reason for correct coordination between the type of action and the current status of the virtual node.

The procedure is as follows: the algorithm is executed within a 10-second epoch on N time load, that is, N equals to the number of sent queries to the system. A unit of low threshold with 0.2 and a high threshold with 0.8 exist in order to compare the traffic on the datacenter in which the request for them are dispatched. The availability to datacenter at each moment is consider by monitoring traffic. Per node, in case of high traffic being on its neighboring partition is copied or migrated or in case of lack of usage and request

for preserving its resources commits suicide. The general proposed trend is depicted in Figure 2.

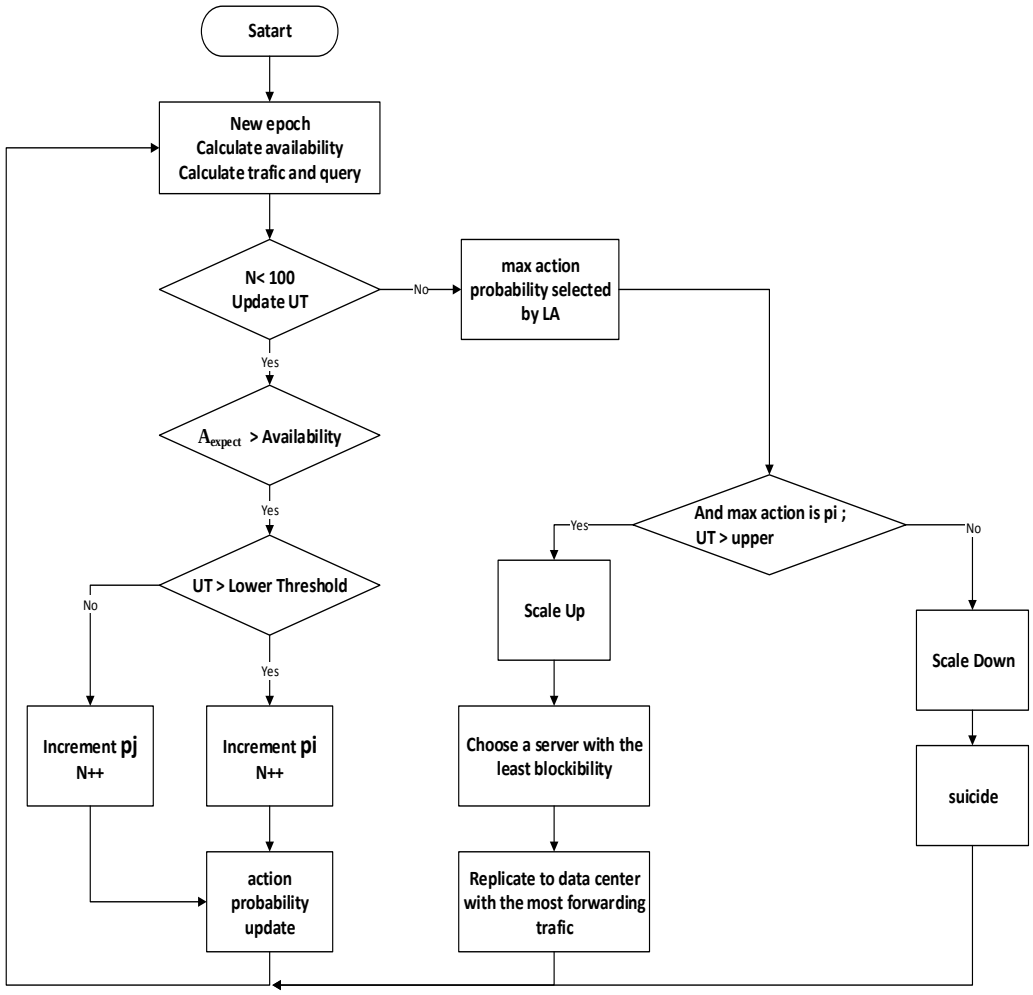


Figure 2. The LARFH Algorithm

4. Performance Evaluation

In this paper for simulation of the proposed approach, a storage environment simulated Cloud consists of 10 datacenters is considered in which geographically distributed in the different continent. First, any datacenter has a room and in any room, there are two racks. In any rack, there are 5 servers or physical nodes. Any server has a fixed storage capability and a bandwidth and fixed processing capability for rendering service to specific numbers of queries in per period. In addition, the bandwidth capability has fixed migration and the copy too. However, any server in accordion with the physical conditions and setting has a different capability from other servers. The Data in partition that are handled by the virtual nodes are kept. The data partition size is 512 KB. In each period, the generated queries from Poisson distribution has average rate of λ followed that may change for fulfillment of different needs. The details of environmental setting and parameters can be seen in Table 1 and also we review our scenarios in Table 2.

Table 1. Environmental Setting and Parameters

Parameter	Default value
Max server storage capacity	10GB
Server storage rate limit	70%
Replication bandwidth	300MB/epoch
Migration bandwidth	100MB/epoch
Epoch	10 seconds
Queries per epoch	Poisson ($\lambda = 300$)
Partitions	64
Partition size	512K
Failure rate	0.1
Minimum availability	0.8
α	0.2
β	2
γ	1.5
δ	0.2
μ	1

Table 2. Type of Scenarios

Scenarios	First target	Second target
Usage rate from copy	Under random query	Under flash crowd
Copy number and cost	Under random query	Under flash crowd

In addition to query random rates, the propose algorithm under the flash crowd is tested and it performance is compared to RFH algorithm [12]. The performance metrics in the proposed algorithm are as follow:

- **Usage Rate from Copy:** usage rate from the copy is a good criterion for measuring the efficiency of the propos algorithm. Whatever the copy is more, the amount of work that a node may do is less and failure probability may likely happen. However, the grate number of copies may lead to lower rate of usage and waste of resources. Another factor that affects the usage rate from the copy is the copy replacement. If copies are local. seldom they can be observed by the quires, consequently the usage rate is definitely lower but if all or majority of copies the key location are executed such as traffic center it is employed completely. The LARFH algorithm places its copies in the required junction routes that definitely leading to more chances of collusion and consequently we have higher usage rate. In migration process, to avoid failure, a threshold for determining whether migration is done or not well. This quantities in the following relation can be shown as:

$$tr_{ij} - tr_{ik} \geq \mu . \bar{tr}_i \tag{3}$$

- **Traffic Computation**: the existence an auto traffic system (middleware) that strikes the load balance sends the received requests to other datacenters with less traffic. This system acts dynamically and monitors over datacenter traffic in our proposed approach. Regarding the proposed algorithm that is traffic-oriented, we should discuss initially. The queries requested information that a client dispatches for request of the specific data partition. In a path finding, the send query produces amount of traffic that in different path finding algorithm this amount is various. Here, only a traffic is valid that travels the path straightly and this sort of traffic is

measurable and not including the sent information to all neighbors. For any virtual node, there exists a limited processing capability and for any physical node, the number of virtual nodes retain the specific sorts of copies. Thus, if the queries load is more than capability, the remainder off queries that cannot be processed will be sent to next physical nodes. All this compotation in a time epoch *t* occurs.
The average of query system along time epoch (t):

$$\bar{l}_{it} = \frac{\sum_{j=1}^{N} l_{ijt}}{N} \tag{4}$$

In order to compensate the rapid changes of queries, the older data transfer to an account
And α is used. Query system rate is represented below:

$$l_{it} = \alpha . \bar{l}_{i(t-1)} + (1-\alpha). \bar{l}_{it} , \quad 0 < \alpha < 1. \tag{5}$$

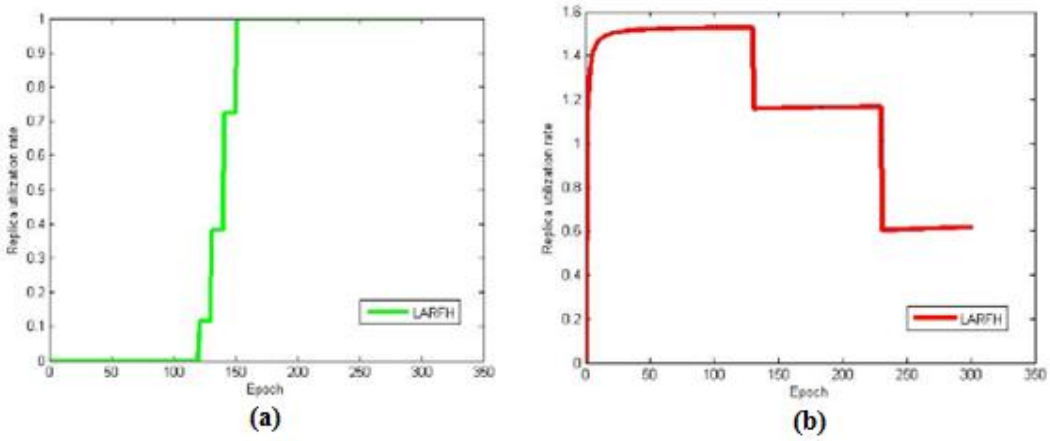
For a sent node, if its traffic is more than γ equals to queries system average, it can be considered as an extra load and is marked as traffic center nodes, namely:

$$tr_{ikt} \geq \gamma . \bar{l}_{it} , \gamma > 1. \tag{6}$$

- **Availability Computation:** quarry rate in under web programs varies. Sometimes concerning query rates maybe low and no crowd is imposed to the server, a minimum availability for taking the possession of resources and the high speed for client's request should be provided. To have the minimum expected and required availability, we can use the minimum of copy numbers via rmin.

$$1 - \sum_{j=1}^{m} (-1)^{j+1} C_m^j (\prod_{i=1}^{r_j})^j \geq A_{\exp ect} . \tag{7}$$

- **First Scenario: Replica Utilization Rate**
This test for checking usage rate from copy under algorithm is offered. The Figure 4(a) displays usage rate average from copy in per epoch with the help of random query. LARFH algorithms has highest rate as compared to RFH algorithm. Since this algorithm is copied in the nodes that have the highest traffic, its copies can be consumed compliantly. The Figure 3 (b) indicates usage rate copy of proposed algorithm under the flash crowd setting. The previous proposed algorithm have the lowest consumption rate under the query setting. When query changes suddenly, after 100^{th} epoch reduces very clearly. The copy consumption rate falls down from 70% to 10%. Apparently, LARFH algorithm performs better in comparison to flash crowd. This algorithm in the first query change will not alter.



(a) under random query (b) under flash crowd.

Figure 3. Replica Utilization Rate

Algorithm of near nodes to those that have the major partition but are not similar in datacenter is selected so that a high availability whit less time and low copy costs can be achieved.

Ideally, if a node replicates from a partition over its neighbors, all queries meet for the partitions before reaching the partition with a node. Therefore, copy consumption rate will be high.

As it can be seen in Figure 4, usage rate from copy for to LARFH and RFH algorithms was shown separately. In all LARFH criteria, bather result in comparison to RFH algorithm was achieved. The copy consumption rate only decreases one time, when the traffic center changes. That is why LARFH copies data in those nodes with the most traffic. Thus, if those nodes change in condition that a great amount of queries in a short time epoch is changing, the consumption rate decreases. However, it regulates rapidly and the rate increases swiftly and leading to a performance approximately good as it started.

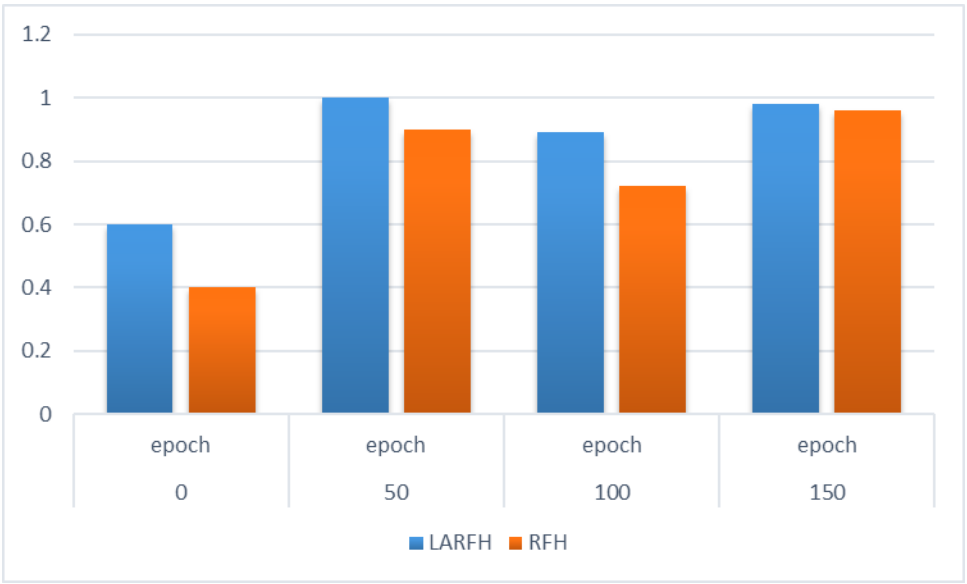
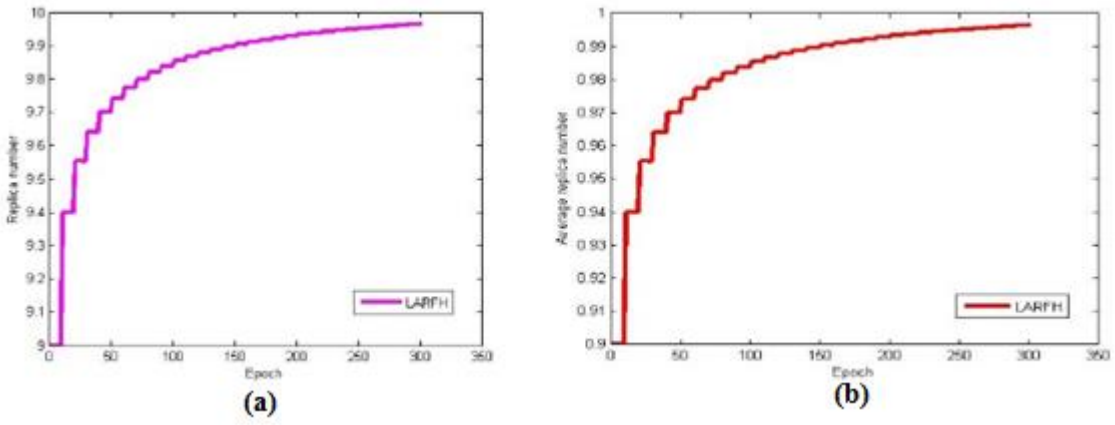


Figure 4. The Comparison of Replica Utilization Rate Consumption Rates

- **Second Scenario: Copy Cost**

Our general purpose in the proposed algorithm is to use the optimal space and proper copy, so with reviewing the amount of threshold we pay attention to the experiment. This experiment for testing the replica number and cost under to random query settings and flash crowd setting are considered. The Figures 5 (a) and (b) respectively depict the number of copy and average of random queries and flash crowd for any partition. In order to fulfill the requirement of working volume of similar queries, more than 500 copies must be copied for any partition 8 times on average. LARFH algorithm has the best performance. This algorithm can be shown the working volume of 250 copies totally and any partition on average has approximately 4 copies. However, under flash crowd its performance approximately is as good as random query, meanwhile RFH algorithm apparently adjusts itself to query changes with more copies.



(a) total replica number under random query and under flash crowd

(b) replication number average for any partition under random query and flash crowd

Figure 5. Copy Number

As can be seen in the Figure 6 can be observed, the rate of total copy for two LARFH and RFH algorithm can be displayed separately. In all LARFH criteria, better result in comparison to RFH has been obtained. This result shows that LARFH algorithm has had a good performance in copy number; especially it has a good compatibility under flash crowd.

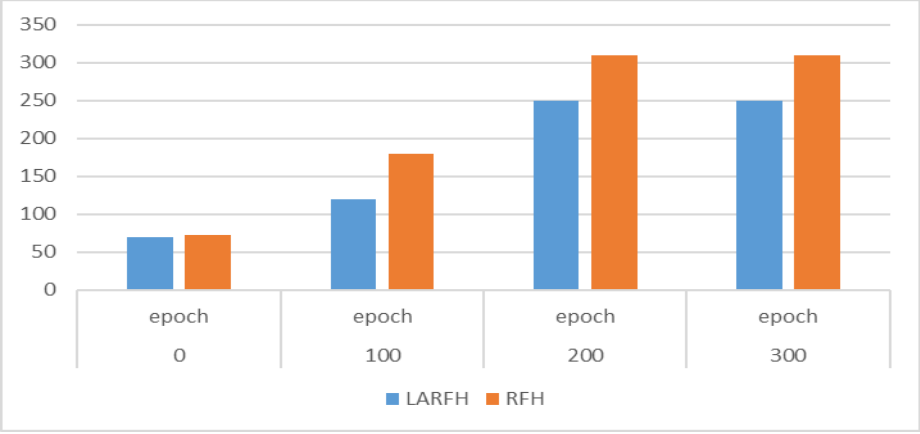
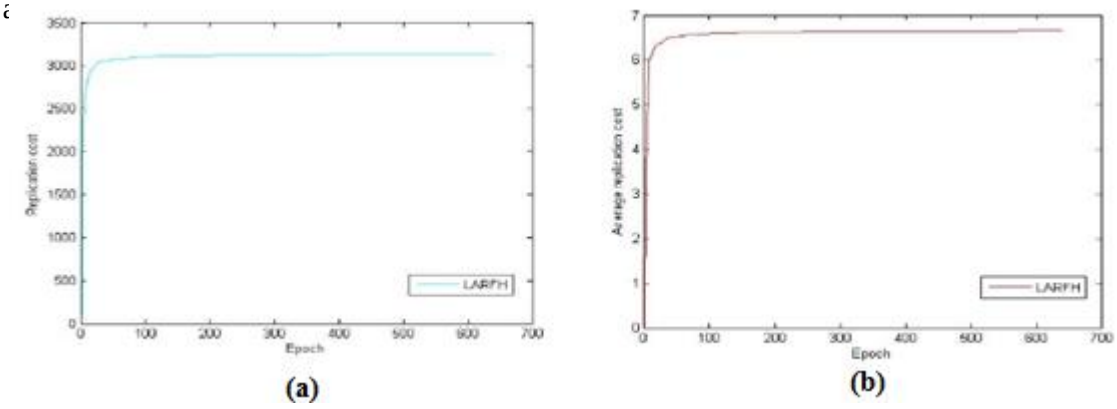


Figure 6. The Comparison of Average Replica Number

The Figures 7 (a) and (b) indicate the copy cost under random query and under flash crowd. The Figur7 (a) has been the total copy cost while the Figure 7 (b) is copy cost average per copy. According to the conducted examinations, the other algorithm have the higher total cost and average. LARFH algorithm has the lowest total copy cost and average in comparison to the rest of and to RFH algorithm. It seems not reasonable that LARFH algorithm can obtain a less copy cost compared to other algorithm since other



(a) Replica cost under random query and under flash crowd

(b) average cost of random query and under flash crowed

Figure 7.

As it can be seen in the Figure 8, the total copy cost rate for to LARFH and RFH algorithm had been given separately. In all LARFH criteria the less cost is achieved compared to RFH. This result indicates that LARFH algorithm has had a good performance in copy cost specially its good compatibility udder flash crowd. It can be claimed that the proposed algorithm of copy in similar datacenter having initial partition but in different servers can be placed. Therefore, even copy cost is lower in comparison to neighbors. This issue takes into account the nodes' locations of traffic center and so on query location based on traffic nodes' form. Other algorithms performance due to grate distance of copy, under random query is very low. The average cost of LARFH algorithm is higher than RFH because different traffic nodes that are under the random query leading to higher cost.

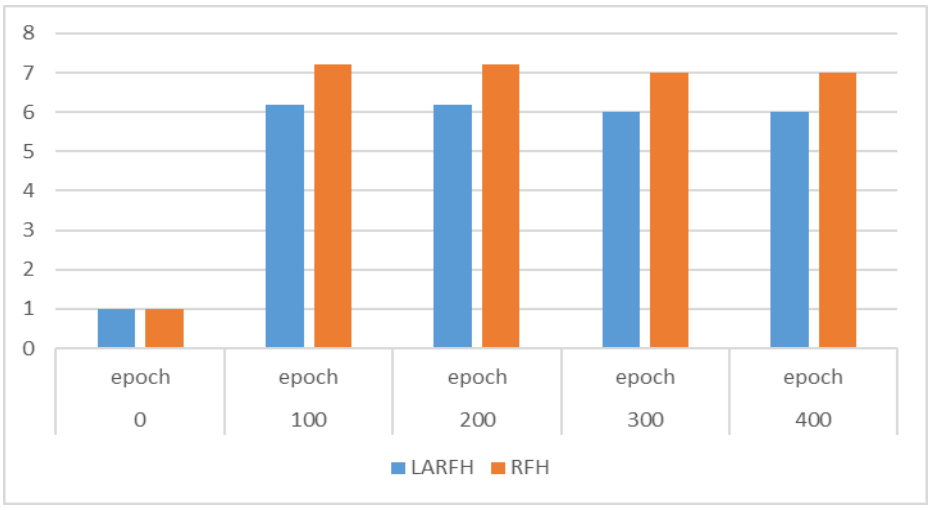


Figure 8. The Comparison of Average Replica Cost

If a partition initial holder does not have a traffic node, copy cost average of LARFH is higher than other algorithms. In spite of this fact, the total copy cost also due to less copy will be lower.

5. Conclusion

With respect to the amazing increase in use of the Cloud storage system, the management resources improvement in the shortest time to respond user's requests and also the geographical constraints and rendering the Cloud services and user are of prime significance. Since the Cloud storage system are usually exposed to failure, fault-tolerance, the capability of the Cloud storage system for responding to unexpected failure are evaluated from the point of software and hardware. In this paper, with offering LARFH algorithm, fault-tolerance increase and data availability enhancement in the Cloud storage system have been provided. The proposed LARFH algorithm, different from the current conventional algorithms that randomly copy the data on the other nodes, node traffic in traffic center are surveyed. The obtained results prove that LARFH algorithm has a better performance compared to other algorithms that can offer higher efficiency, fault-tolerance, and more dynamic coping services. For future researches, we can benefit from genetic algorithm and neural network for improvement of algorithms. Additionally, there is a possibility that the received requests to traffic centers can be defined in terms of categories and classifications and allocating virtual node for data copying based on the demanded activities that can be added or reduce.

References

[1] P.Gill, N.Jain, N.Nagappan. "Understanding Network Failures in Data Centers: Measurement, Analysis, and Implications". In ACM SIGCOMM'11, Canada, **(2011)**.

[2] E. Pinheiro, W.-D. Weber, and L. A. Barroso. "Failure trends in a large disk drive population." In Proceeding of 5th USENIX Conference on File and Storage Technologies, San Jose, USA, Feb. **(2007)**.

[3] Afife Fereydooni, Mostafa Ghobaei Arani and Mahboubeh Shamsi "EDLT: An Extended DLT to Enhance Load Balancing in Cloud Computing." *International Journal of Computer Applications*, vol. 108, no. 7, **(2014)** Dec., pp. 6-11.

[4] M. C. Chan, J. R. Jiang, and S. T. Huang, "Fault-tolerant and secure networked storage," 7th International Conference on Digital Information Management, ICDIM 2012. **(2012)** pp. 186-191.

[5] Arvind Kumar, Rama Shankar Yadav, Ran vijay and Anjali Jain "Fault Tolerance in Real Time DistributedSystem", International Journal on Computer Science and Engineering (IJCSE),Vol. 3,ISSN: 0975-3397, no. 2 ,Feb **(2011)**.

[6] Bakhta Meroufel and Ghalem Belalem, "Dynamic Replication Based on Availability and Popularity in the Presence of Failures," Journal of Information Processing Systems', vol.8, no.2, **(2012)** pp. 263-278.

[7] Wei Q, Veeravalli B, Gong B, Zeng L, Feng D. "CDRM: A cost effective dynamic replication management scheme for cloud storage cluster." , In Proc. 2010 IEEE International Conference on Cluster Computing, Heraklion, Crete, Greece, Sept. 20-24, **(2010)**, pp.188-196.

[8] Yilei Zhang, Zibin Zhengand and Michael R. Lyu, "BFTCloud: A Byzantine Fault Tolerance Framework for Voluntary-Resource Cloud Computing", 4th International Conference on Cloud Computing, IEEE, **(2011)**.

[9] Ghilavizadeh Zeinab, seyed javad mirabedini, Ali Harounabadi, "A new Fuzzy optimal data Replication for data Grid", Management science **(2013)**, pp.927-936.

[10] B.-y.Ooi et al."Dynamic service placement and replication framework to enhance service availability using team formation algorithm," Elsevier Journal of System and Software, March **(2012)**, pp.2048-2062.

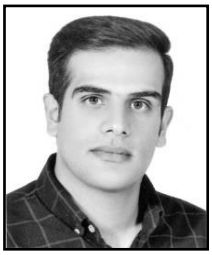
[11] Anjali D. Meshram, A.S.Sambare and S. D. Zade, "Fault Tolerance Model for Reliable Cloud Computing ", International Journal on Recent and Innovation Trends in Computing and Communication, ISSN 2321 – 8169, Vol.

[12] Yanzhen Qu, Naixue Xiong: "RFH: A Resilient, Fault-Tolerant, and High-efficient Replication Algorithm for Distributed Cloud Storage". International Conference on Parallel Processing, 4. **(2012)**.

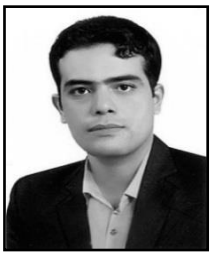
[13] Thathachar, M., and P. Shanti Sastry. "Varieties of learning automata: an overview." Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 32, no. 6 **(2002)**, pp. 711-722.

[14] Najim, Kaddour, and Alexander S. Poznyak. Learning automata: theory and applications. Pergamon Press, Inc., **(1994)**.
[15] Monireh Fallah, Mostafa Ghobaei Arani and Mehrdad Maeen. "NASLA: Novel Auto Scaling Approach based on Learning Automata for Web Application in Cloud Computing Environment" *International Journal of Computer Applications* vol. 113, no. 2, **(2015)** March, pp. 18-23,
[16] Monireh Fallah, Mostafa Ghobaei Arani "ASTAW: Auto-Scaling Threshold-based Approach for Web Application in Cloud Computing Environment." International Journal of u- and e- Service, Science and Technology (IJUNESST),vol.8, no.3,**(2015)** pp. 221-230.
[17] Behnaz Seyed Taheri, Mostafa Ghobaei Arani and Mehrdad Maeen. "ACCFLA: Access Control in Cloud Federation using Learning Automata" International Journal of Computer Applications vol. 107, no. 6, **(2014)** Dec., pp. 30-40.

Authors



Seyyed Mansour Hosseini received the B.S.C degree in Software Engineering from University Azad Kashan, Iran in 2010, and M.S.C degree from Azad University of Mahallat, Iran in 2014, respectively. Her research interests include Cloud Computing, Distributed Systems, Cloud Storage Technology and Software Development.



Mostafa Ghobaei Arani received the B.S.C degree in Software Engineering from IAU Kashan, Iran in 2009, and M.S.C degree from Azad University of Tehran, Iran in 2011, respectively. He's currently a PhD Candidate in Islamic Azad University, Science and Research Branch, Tehran, Iran. His research interests include Grid Computing, Cloud Computing, Pervasive Computing, Distributed Systems and Software Development.

Abdol Reza Rasouli Kenari is Associate Prof. Qom University of Technology. He's received the B.S.C degree in Mathematics from Isfahan University, Iran in 2003, and received the M.S.C degree from Azad University of Isfahan, Iran in 2006, and also received the PhD degree in Software Engineering from UTM in 2011. Her research interests include Distributed Data Security, Big Data and Cloud Computing.