ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

A Light-weight Data Replication for Cloud Data Centers Environment

Mohamed-K HUSSEIN, Mohamed-H MOUSA

Faculty of Computers and Informatics, Suez Canal University, Egypt

ABSTRACT: Unlike traditional high performance computing environment, such as cluster and supercomputers, the cloud computing is a collection of interconnected and virtualized computing resources that are managed to be one unified high-performance computing power. However, the Cloud environment constitutes a heterogeneous and a highly dynamic environment. Failures on the data centers nodes are normal rather because of the large scale of physical resources and data. As a result, the cloud environment requires an efficient adaptive data replication management in order to cope with the inherent characteristic of the Cloud environment. In this paper, we propose a data replication strategy which adaptively selects the data files for replication in order to improve the overall reliability of the system and to meet the required quality of services. Further, the proposed strategy decides dynamically the number of replicas as well as the effective data nodes for replication. The popular data files are selected for replication based on employing a lightweight time-series technique, which analyzes the recent pattern of data files requests, and provides predictions for the future data requests.Experimental results show that the proposed strategy behaves effectively to improve the reliability of the Cloud system under study.

Keywords: system availability, replication, adaptive, cloud computing

I. INTRODUCTION

Cloud computing is a large-scale parallel and distributed computing system. It consists of a collection of inter-connected and virtualized computing resources that are managed to be one unified computing resources. The provided abstract, virtual resources, such as networks, servers, storage, applications and data, can be delivered as a service. Services are delivered on demand to the end-users over high-speed Internet as three types of computing architecture, namely Software as a Service (SAAS), Platforms as a Service (PAAS) and Infrastructure as a service (IAAS). The main goal is to provide users with more flexible services in a transparent manner, cheaper, scalable, highly available and powerful computing resources[1].The Software as a Service (SaaS) architecture provides software applications hosted and managed by a service provider for the end-user replacing locally-run applications with web services applications. In the Infrastructure as a Service (IaaS), Service includes management of hardware and software for processing, data storage, networks and any required infrastructure for deployment of operating systems and applications which would normally be needed in a data center managed by the user. In the Platform as a Service (PaaS), Service includes programming languages and tools and an application delivery platform hosted by the service provider to support development and delivery of end-user applications[2].

In general, the Cloud Computing provides the software and hardware infrastructure as services using large-scale data centers[3]. As a result, Cloud computing moved away the computation and data storage fromthe end user and onto large number of data centers infrastructure. This relieves users of theburdens of hardware and application provisioning and management. Hardware and software are delivered to users as on-demand services over the Internet. The Cloud infrastructure can scale out the hardware capacity to meet the required non-functional quality of services (QoS). However, it is challenging to provide high reliability and efficient access to the cloud data centers because of the large scale and dynamic nature of the Cloud. Replication is the process of providing different replicas of the same service at different nodes[4]. Replication is a used technique in the current different clouds architectures, such as GFS (Google file system) and HDFS (Hadoop Distributed File System) [5, 6]. In the cloud, data replication is achieved through data resource pool and the number of data replicas is statically set based on history and experience[7]. Further, it is not necessary to create replica for all data files, especially for those non frequently or recently used data files. Therefore, it is necessary to adaptively replicate the frequently used data files, determine the number of data replicas and the data nodes locations where to place the new replicas according to the current cloud environments conditions.

In this paper, we propose an adaptive replication strategy in a cloud environment that adaptively copes with the following issues:

- What to replicate to improve the non-functional QoS. The select process is mainly depends on analyzing the history of the data requests using a lightweight time-series prediction algorithm. Using the predicted data request, we can identify what data files need replication to improve the system reliability.
- The number of replicas for each selected data.
- The position of the new replicas on the available data centers.
- The overhead of replication strategy on the Cloud infrastructure. This is the most important factor of the proposed adaptive replication strategy where the Cloud has a large number of data centers as well as a large-scale data. Hence, the adaptive replication strategy should be lightweight strategy.

The proposed adaptive replication strategy is originally motivated by the fact that the recently most accessed data files will be accessed again in the near future according to the collected prediction statistics of the files access pattern[8, 9]. A replication factor is calculated based on a data block and the availability of each existing replica passes a predetermined threshold, the replication operation will be triggered. A new replica will be created on a new node which achieves a better new replication factor. The number of new replicas will be determined adaptively based on enhancing the availability of each file heuristically. This paper depends on the problem formalization described in [9]. However, we employ a lightweight time-series algorithm for predicting the future requests of data files. The replication decision is primarily based on the provided predictions. The heuristic proposed for the dynamic replication strategy is computationally cheap, and can handle large scale resources and data in a reasonable time.

The remainder of this paper is organized as follows. Section 2 presents the related work on data storage and data replication of cloud computing systems. Section 3 presents a formalization of a cloud system model. Section 4 describes the dynamic data replication strategy, including the replication decision, the number of replicas, and the replica placement. Section 5 addresses the simulation environment, parameter setup and performance evaluation of the proposed dynamic data replication strategy. Finally, conclusions and future work are given in Section 6.

II. RELATED WORK

This section presents two broad categories of related work. The first category discusses cloud data center architecture, and the second category presents the related work to the replication in the Cloud environments.

A. Cloud architecture

Cloud computing technology movedcomputation and data storage away fromthe end user onto large number of backend data centers.This relieves users of theburdens of hardware and software provisioning and management. As a result, software and hardwareare delivered as services and consumed over the Internet[10-12]. The Cloud data centers are based on the distributed file system technology, such as Google File System (GFS), the Hadoop distributed file system (HDFS). The GFS architecture consists of three components, namely multiple clients, a single master server and multiple chunk servers[5]. Data files are divided into many fixed size chunks which are stored in the chunk servers. Chunks are stored in plain Linux files which are replicated on multiple nodes to provide high-availability and improve performance. The master server maintains all the metadata of the file system, including the namespace, the access control information, the mapping from files to chunks, and the current locations of chunks. Clients send data requests to the master server, and the master server directs the requests to the chunk servers[13].The HDFS distributed file system also follows a master/slave architecture which consists of a single master server,called the Namenode, that manages the distributed file system namespace and regulates access to files by clients. In addition, there are multiple datanodes, one in each cluster, which manage the disk storage attached to the nodes assigned to Hadoop. The Namenode determines the mapping of blocks to datanodes[6].

B. Replication in the Cloud

Replication technology is one of the useful techniques in distributed systems for improving availability and reliability[9]. In Cloud computing, replication is used for reducing user waiting time, increasing data availability and minimizing cloud system bandwidth consumption by offering the user multiple replicas of a specific service on different nodes. For example, if one node fails, a replica of the failed service will be possibly created on a different node in order to process the requests[12].

Many replication techniques have been proposed in the literature, which are classified into static and dynamic replication. In a static replication, the number of replicas and their locations are initially set in advance [5, 6, 13]. On the other hand, dynamic replication dynamically creates and deletes replicas according to changing environment load conditions [10, 13]. There has been an interesting number of works for data replication in the Cloud computing. For example, in [5], a GFS static cloud data replication algorithm is proposed.The GFS adopts a replication strategy where a single master places the selected data chunk replicas on chunk servers with below-average disk space utilization. Further, the number of replication for each data chuck is specified by the users. Similarly, in [6], an application can specify the number of replicas for each file, and the block size and replication factor are configurable per file. In [13], a static centralized data replication algorithm sets a specific number of replicas based on the minimum weighted distance. In [10], a dynamic distributed cloud data replication algorithm CDRM is based on the HDFS file system where the replication site are selected based on the nodes which have low utilization. In [13], six different dynamic data replication algorithms, Caching-PP, Cascading-PP, Fast Spread-PP, Cascading-Enhanced, and Fast Spread-Enhanced are proposed. In [14], a dynamic centralized data replication algorithm is proposed. The algorithm treats uses a weighting factor for the replication, and utilizes different prediction techniques.

III. PROBLEM FORMALIZATION

A cloud data service system typically consists of the scheduling broker, replica selector, replica broker and data centers[8, 11, 12, 14-18], as shown in Figure 1. The scheduling broker manages the replication process, and contains all the information about the number of replicas as well as their location at the different data centers. The formal model about the Cloud data centers architecture is well described at [9] as follows:

Let $U = \{u_1, u_2, \dots, u_m\}$be m users at the Cloud, $TS = \{TS_1, TS_2, \dots, TS_m\}$ be a set of tasks of the user set U, and $TS_j = \{ts_{j1}, ts_{j2}, \dots, ts_{jm_j}\}$ be a subset of tasks of the j^{th} user u_j, where m_j is the number of subtasks, and $ts_k$is the k^{th} task submitted to the scheduling broker through a Cloud interface. If u_0 has two tasks, then $TS_0 = \{ts_{01}, ts_{02}\}$, and $m_0 = 2$. A task ts_k is characterized by a 4-tuple $ts_k = (tid_k, tr_k, td_k, tfn_k)$, where tid_k, tr_k, td_k *and* $tfn_k$are the task identification, task generation rate, task deadline time and the number of required files of task ts_k, respectively[11, 15, 19]. $DC = \{dnd_1, dnd_2, \dots, dnd_n\}$representsa data center consists of n data nodes on a physical machine PM. Each noderuns a virtual machine, and is characterized by dnd_k that is a 5-tuple$dnd_k = (dnd_k, dr_k, dst_k, df_k, dbw_k)$, where dnd_k, dr_k, dst_k, df_k and $dbw_k$are the data node identification, request arrival rate, average service time, failure probability andnetwork bandwidth of data node dnd_k, respectively.In order to guarantee the service performance of thedata center DC, the task generation rate $tr_k$of user set$U$, the request arrival rate $dr_k$and failure probability$df_k$of DC should meet (1).

$$\sum_{j=0}^{\#subtasks} tr_j \le \sum_{i=0}^{n} dr_i \times (1 - df_i) \quad (1)$$

wheretr_j is the task generation rate of task j, dr_iisthe request arrival rate of task $j$on the node i, df_k is the failure probability of task j.

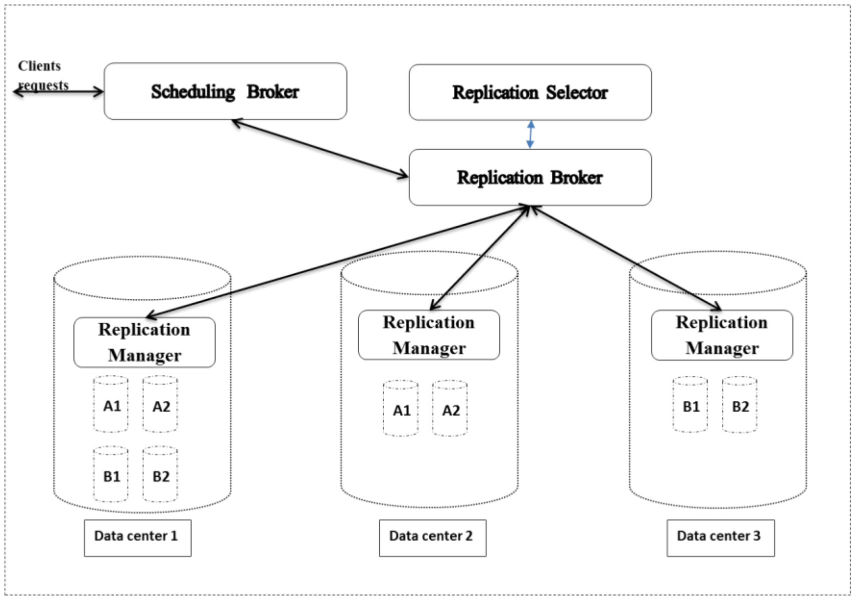

Fig.1. The Cloud data server architecture.

Let $F = \{f_1, f_2, \dots, f_l\}$be a data file set of a datacenter DC. $B = \{B_1, B_2, \dots, B_l\}$be a set of blocksin the data center DC, and $B_i = \{b_{i1}, b_{i1}, \dots, b_{in1}\}$ bethe i-th subset of blocks belonging to the i-th datafile f_i, which is stripped into $n_i$fixed blocks according to its length.A block b_k is characterized by a 5-tuple$b_k = (bid_k ; bp_k ; bs_k ; bn_k ; bt_k)$, where bid_k, bp_k, bs_k, bn_k and bt_k are the block identification, number of requests, block size, the number of replicas and the lastaccess time of block bk, respectively.When user u_j requests a block b_k from a data node$dnd_i$with bandwidth performance guarantee, bandwidth bs_k/dst_i should be assigned to this session. Thetotal bandwidth used to support different requests fromuse set U should be lessthan dbw_i, as shown by (2).

$$\sum_{i=0}^{s_i} \frac{bs_k}{dst_i} \le dbw_i \qquad (2)$$

wheres_i is the maximum number of network sessions ofdata node dnd_ithat can serve concurrently, $bs_k$is theblock size of block b_k, $dst_i$is the average service timeof data node dnd_i, dbw_i is the network bandwidth ofdata node dnd_k.

Block availability is the ability of a data block to provide proper service under given constraints. The block availability of a block b_k is denoted as BA_k. $P(BA_k)$is the probability of block b_k in an available state. $\hat{P}(BA_k)$is the probability of block $b_k$in an unavailable state, and $\hat{P}(BA_k) = 1 - P(BA_k)$.The number of replicas of block $b_k$is bn_k. It is obvious that block $b_k$is considered unavailable only if all the replicas of block $b_k$are not available. So the availability and unavailability of block $b_k$are calculated as 3 and 4.

$$P(BA_k) = 1 - (1 - P(ba_k))^{bn_k} \qquad (3)$$

$$\hat{P}(BA_k) = (1 - P(ba_k))^{bn_k} \qquad (4)$$

File availability is the ability of a data file to provide proper service under given constraints. The file availability of a data file f_i is denoted as FA_i. $P(FA_i)$is theprobability of data file $f_i$an available state. $P(FA_i)$isthe probability of data file $f_i$in an unavailable state,and $\hat{P}(FA_i) = 1 - P(FA_i)$.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

If the data file $f_i$is stripped into $n_i$fixed blocks denoted by $B_i = \{b_{i1}, b_{i2}; \ldots ; b_{in1}\}$, which are distributed on different data nodes. $N_i = \{bn_{i1}, bn_{i2}, \ldots, bn_{ini}\}$is the set of the numbers of replicas of the blocks of B_i. The availability and unavailability of data file $f_i$is given as follows:

$$P(FA_i) = (1 - \left(1 - P(ba_i)\right)^{bn_i})^{n_i} \tag{5}$$

If the data file $f_i$is stripped into $n_i$blocks, there are $n_i$replicas of each block in data file f_i,and all blocks at the same site will have the same available probability as all blocks are stored in data nodes with the same configuration in cloud data centers, the available probability of each replica is $p(ba_i)$ in data file f_i.

IV. DYNAMIC DATA REPLICATION STRATEGY

The proposed adaptive data replication has three important phases: 1) the selection phase: which data files should be selected for replication; 2) setting the number of suitable replicas to meet the specified QoSs.; 3) determining the best location of replicas..

The first step is to decide which data replicate and the replication timing. The selection phase analyzes the history of the data access requests, and employs a lightweight time series technique to predict the future access frequency of the data. If the predicted future requests of data chunks exceeds an adaptive threshold, the data chunks will be selected for replication. Let pd_k be a popularity degree of a block b_k. $pd_k$is defined as the future access frequencybased on the number of access demand,an_k(t)ata time t, the popularity degree $pd_k$of a block b_k can be calculated using Holt's Linear and Exponential Smoothing (HLES).Holt's Linear and Exponential Smoothing (HLES) is a computationally cheap time series prediction technique. HLES is selected for its capability of smoothing and providing short-term predictions for the measured requests arrival rates and service demand rates. Hence, HLES enables the proposed framework to monitor the arrival rates and service rates and to provide a short-term prediction for the future arrival rates and service rates with low computation time. Using these predictions, we can predict the utilization on each server host using equation (2), and predict the future response time of the web service using equation (1).

HLES smoothes the time series and provides a short-term forecast based on the trend which exists over the time series [20]. Suppose an_k(t)is a time series value at time *t*. The linear forecast for the *m* steps ahead is as follows:

$$pd_k = an_k(t+m) = L_t + b_t \times m \tag{6}$$

whereL_t and b_t are exponentially smoothed estimates of the level and linear trend of the series at time *t*:

$$L_t = \alpha\, an_k(\mathrm{t}) + (1 - \alpha)(L_{t-1} + b_{t-1})$$

whereα is a smoothing parameter, $0 < \alpha < 1$,

$$b_t = \beta\,(L_t - L_{t-1}) + (1 - \beta)b_{t-1}$$

whereβ is a trend coefficient, $0 < \beta < 1$.

A large value of α adds more weight to recent values rather than previous values in order to predict the next value. A large value of β adds more weight to the changes in the level than the previous trend.

The replica factor is defined as the average of the ratio of the popularity degree and the averageavailability of replicas on the different data nodes of all blocks *l* of the data filef_i. It is used to determine whether the data file $f_i$should be replicated, denoted as

```
• Initialize available and unavailable probability of each replica of
  block $b_k$,$p(ba_k)$ and $\hat{p}(ba_k)$.
• for each data file $f_i$at all data centers DC do
    o Calculate the popularity degree $pd_k$ of a block $b_k$ of data
      file $f_i$
    o Calculate replica factor $RF_i$ of data file $f_i$.
    o If $RF_i$is less than a threshold λ, trigger the replication for the
      file $f_i$
• end for
• for each triggered replication for data file $f_i$do
    o for each block $b_k$in the file $f_i$
        ▪ Calculate the new $RF_i$ by adding a replication on
          the each data center $dc_k$.
        ▪ apply the replication which gives the highest new
          $RF_i$.
    o end for
• end for
• find the file $f_i$which has the least$\sum_{k=1}^{l} pd_k$.
• for each replica in the file
    o delete the replica which gives the new, without the
      replica,$RF_i$ bigger than a threshold λ
• end for
```

Fig. 2.The proposed adaptive replication strategy.

$$RF_i = \frac{1}{l}\sum_{k=1}^{l}\frac{\sum_{k=1}^{bn_i} pd_k \hat{P}(BA_k)}{bn_i} \tag{7}$$

where$pd_k, \hat{P}(BA_k), l$ and $bn_i$are the popularity degree, the failure probability of a block b_k, number of blocks and numberof replicas of data file f_i,respectively.

In each time interval T, the replication operation of the data file f_i will be triggered if the replication factor $RF_i$is less than a specified threshold. The details of the proposed adaptive strategyare shown in Figure 2.

V. Simulation and Performance Evaluation

This section evaluates the effectiveness of the proposed adaptive replication strategy.The CloudSim framework is a Java based simulation platform for the Cloud environment, it supports modeling and simulation of large scale cloud computing data centers, including users and resources[21-23].In the CloudSim simulation, 64 data centers are created with the corresponding topology shown in Figure 1. The service providers are represented by 1000 virtual machines, and the processing elements (PEs) number of each virtual machine is within the range of 2 to 4. A hundred different data files are placed in the cloud storage environment, with each size in the range of [0.1, 10] GB. Each file is stored in fixed size (bs= 0:2 GB) storage unit called block. Blocks of the same data file are scattered across different virtual machines. 10000 tasks are submitted to the service providers using the Poisson distribution. Each task requires 1 or 2 data files randomly. Initially, the number of replicas of each data file is 1 and placed randomly. For simplicity, it is assumed that the basis element of data storage is block and the element of replication is one total data file.

As shown in Figure 3, with time elapses, the numberof replicas is increasing within a very short period oftime. Then,the number of replicas is maintained at arelatively stable level, which is determined by the adjustable parameter α in the HLES technique. We conclude that the greater theadjustable parameter α and the increasing block request bp_k of a certain file, the more replicas are neededto improve the file availability.

The response time for a data fileis the interval between the submission time of the taskand return time of the result. The average responsetime of a system is the mean value of the response time for all data request tasks of the users, which can be obtained by the following equation.

$$rt_{avg} = \frac{\sum_{j=1}^{m}\sum_{k=1}^{m_j}(ts_{jk}(rt) - ts_{jk}(st))}{\sum_{j=1}^{m} m_j} \qquad (8)$$

wherets_{jk} (st) and ts_{jk} (rt) are the submission time and the return time of the result of task k of the user j, respectively, and m_j is the number of the tasks of user j.

As shownin Figure 4, with the number of tasks increasing and α = 0.7, the response time increases dramatically. The less the blockavailability, the longer the response time will be. It is clear that the proposed adaptive replication strategy enhances the response time and maintains the responsetime at a stable level within a short period of time.

VI. Conclusions and Future Work

This paper proposes an adaptive replication strategy in the cloud environment. The strategy investigates the availability and efficient access of each file in the data center, and studies how to improve the reliability of the data files based on prediction of the user access to the blocks of each file. The proposed adaptive replication strategy redeploys dynamically large-scale different files replicas on different data nodes with minimal cost using heuristic search for each replication. The proposed adaptive strategy is based on a formal description of the problem. The strategy identifies the files which are popular file for replication based on analyzing the recent history of the data access to the files using HLES time series. Once a replication factor based on the popularity of the files is less than a specific threshold, the replication signal will be triggered. Hence, the adaptive strategy identifies the best replication location based on a heuristic search for the best replication factor of each file. Experimental evaluation demonstrates the efficiency of the proposed adaptive replication strategy in the cloud environment.

Future research work will focus on building a database system for 3D models. In fact, high quality 3D modelsare archived in huge files. These files are traditionally stored in distributed databases which suffer from answering visualization queries and traffic overloading on data centers. We aim toprovide a framework for speeding up data access, and further increasing data availability for such databases on a cloud environment. Further, we will study using Genetic algorithms to find the best replication in less time. In addition, the replication strategywill be deployed and tested on a real cloud computing platform. Future work is also planned to provide the adaptive data replicationstrategy as a part of cloud computing services to satisfy the characteristics of cloud computing.

References

1. Buyya, R., et al., *Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility.* Future Gener. Comput. Syst., 2009. **25**(6): p. 599-616.
2. Armbrust, M., et al., *A view of cloud computing.* Commun. ACM, 2010. **53**(4): p. 50-58.
3. Mell, P.M. and T. Grance, *SP 800-145. The NIST Definition of Cloud Computing*. 2011, National Institute of Standards \& Technology.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

4. Bj, M., et al., *Optimizing service replication in clouds*, in *Proceedings of the Winter Simulation Conference*. 2011, Winter Simulation Conference: Phoenix, Arizona. p. 3312-3322.
5. Ghemawat, S., H. Gobioff, and S.-T. Leung, *The Google file system.* SIGOPS Oper. Syst. Rev., 2003. **37**(5): p. 29-43.
6. Shvachko, K., et al., *The Hadoop Distributed File System*, in *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*. 2010, IEEE Computer Society. p. 1-10.
7. Bonvin, N., T.G. Papaioannou, and K. Aberer, *Dynamic cost-efficient replication in data clouds*, in *Proceedings of the 1st workshop on Automated control for datacenters and clouds*. 2009, ACM: Barcelona, Spain. p. 49-56.
8. Chang, R.-S. and H.-P. Chang, *A dynamic data replication strategy using access-weights in data grids.* J. Supercomput., 2008. **45**(3): p. 277-295.
9. Sun, D.-W., et al., *Modeling a Dynamic Data Replication Strategy to Increase System Availability in Cloud Computing Environments.* Journal of Computer Science and Technology, 2012. **27**(2): p. 256-272.
10. Wei, Q., et al., *CDRM: A cost-effective dynamic replication management scheme for cloud storage cluster.*, in *2010 IEEE International on Cluster Computing*. 2010. p. 188 - 196
11. Bonvin, N., T.G. Papaioannou, and K. Aberer, *A self-organized, fault-tolerant and scalable replication scheme for cloud storage*, in *Proceedings of the 1st ACM symposium on Cloud computing*. 2010, ACM: Indianapolis, Indiana, USA. p. 205-216.
12. Nguyen, T., A. Cutway, and W. Shi, *Differentiated replication strategy in data centers*, in *Proceedings of the 2010 IFIP international conference on Network and parallel computing*. 2010, Springer-Verlag: Zhengzhou, China. p. 277-288.
13. Dogan, A., *A study on performance of dynamic file replication algorithms for real-time file access in Data Grids.* Future Gener. Comput. Syst., 2009. **25**(8): p. 829-839.
14. Wang, S.-S., K.-Q. Yan, and S.-C. Wang, *Achieving efficient agreement within a dual-failure cloud-computing environment.* Expert Syst. Appl., 2011. **38**(1): p. 906-915.
15. McKusick, M.K. and S. Quinlan, *GFS: Evolution on Fast-forward.* Queue, 2009. **7**(7): p. 10-20.
16. Lei, M., S.V. Vrbsky, and X. Hong, *An on-line replication strategy to increase availability in Data Grids.* Future Gener. Comput. Syst., 2008. **24**(2): p. 85-98.
17. Jung, D., et al., *An effective job replication technique based on reliability and performance in mobile grids*, in *Proceedings of the 5th international conference on Advances in Grid and Pervasive Computing*. 2010, Springer-Verlag: Hualien, Taiwan. p. 47-58.
18. Yuan, D., et al., *A data placement strategy in scientific cloud workflows.* Future Generation Computer Systems, 2010. **26**(8): p. 1200-1214.
19. Litke, A., et al., *A Task Replication and Fair Resource Management Scheme for Fault Tolerant Grids Advances in Grid Computing - EGC 2005*, P. Sloot, et al., Editors. 2005, Springer Berlin / Heidelberg. p. 482-486.
20. Makridakis, S.G., S.C. Wheelwright, and R.J. Hyndman, eds. *Forecasting: Methods and Applications, 3rd Edition*. 1998.
21. Calheiros, R.N., et al., *CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms.* Softw. Pract. Exper., 2011. **41**(1): p. 23-50.
22. Wickremasinghe, B., R.N. Calheiros, and R. Buyya, *CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications*, in *Proceedings of the 2010 24th IEEE International Conference on Advanced Information Networking and Applications*. 2010, IEEE Computer Society. p. 446-452.
23. Xu, B., et al., *Job scheduling algorithm based on Berger model in cloud environment.* Adv. Eng. Softw., 2011. **42**(7): p. 419-425.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

APPENDIX

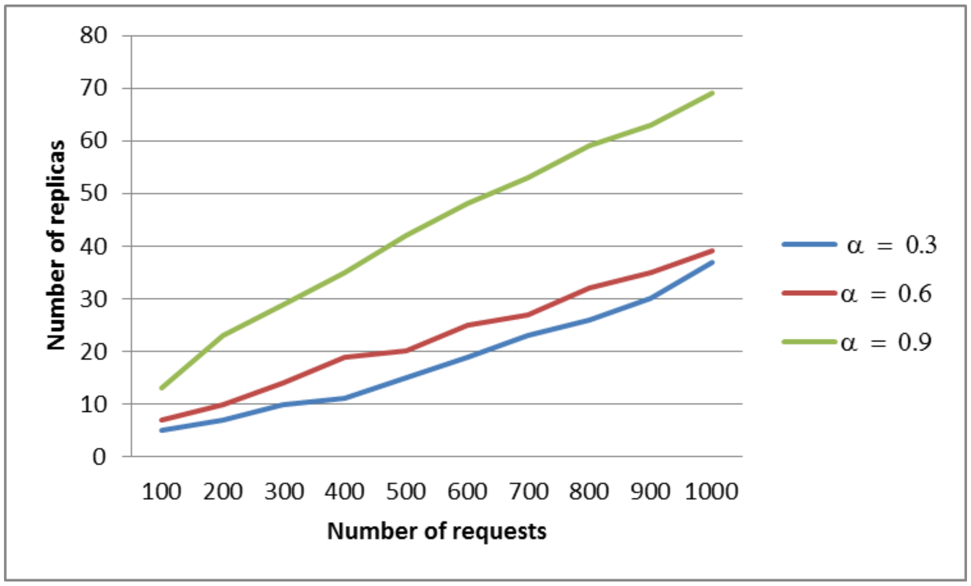

Fig. 3.Number of replicas with increasing parameter α of HLES and increasing requests.

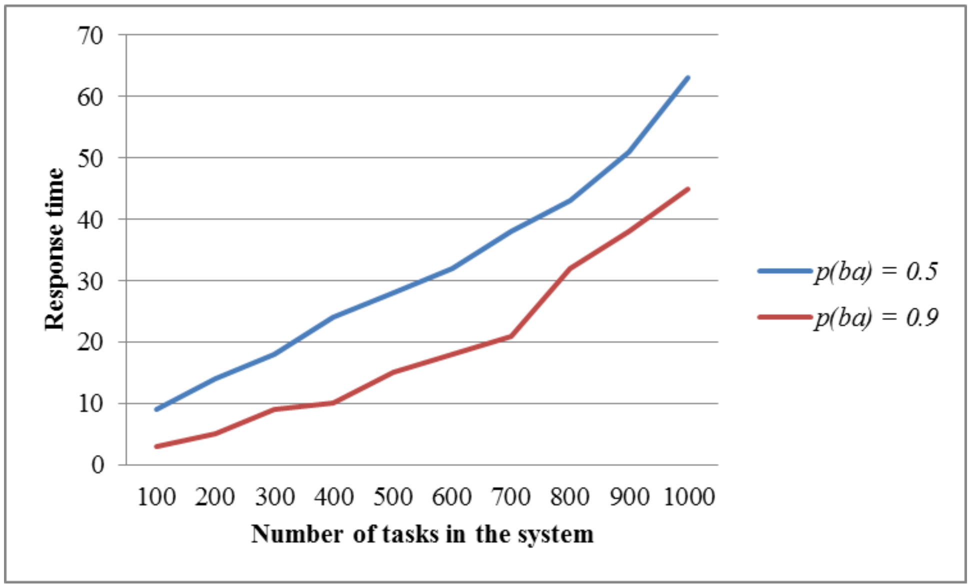

Fig. 4.The response time versus the number of tasks using different probabilities.