File Heat-based Self-adaptive Replica Consistency Strategy for Cloud Storage

Zhen Zhou
College of Computer Science, Chongqing University, Chongqing, China
Email: zhouzhen1302@163.com

Shuyu Chen*
School of Software Engineering, Chongqing University, Chongqing, China
Email: netmobilab@cqu.edu.cn

Tao Ren and Tianshu Wu
College of Computer Science, Chongqing University, Chongqing, China
Email: {t_ren@163.com, vipwts2@gmail.com}

***Abstract*—In cloud storage systems, replica is a key technology, which reduces access time lag, network bandwidth consumption and system unreliability. However, an inadequate replica consistency management mechanism would cause problems for cloud storage systems in the time lag of file access and network bandwidth consumption. Therefore, this paper proposes a self-adaptive replica consistency strategy which allows the system to automatically select the adequate consistency strategy according to file heat. In the calculating of file heat, this paper takes into account the effect caused by the pattern of file access time sequence in a time period and introduces a new algorithm MRFU for the calculating of file heat, by which the file used most recently and frequently in a time period is assigned the largest heat value. Experiment results show that the file heat calculated by MRFU algorithm can precisely predict file access behavior. On the basis of keeping high file access efficiency, the proposed self-adaptive replica consistency strategy effectively reduces the network bandwidth consumption. Furthermore, it is found that the proposed self-adaptive replica consistency strategy has better performance in file access efficiency and cost of network than other similar solutions by contrast tests.**

***Index Terms*—cloud storage, replica consistency, self-adaptive, file heat, file access pattern**

1. INTRODUCTION

An ever increasing application of electronic devices and fast development of the internet technology have seen a surge in the demand for mega data storage. Cloud storage, outperforming conventional SAN and NAS methods in volume, performance and scalability, provides an ideal solution for future data storage. In cloud storage system, replica technology is a key technology in enhancing performance of the system. With the assistance of replica, the distributed file system is able to reduce access time lag, network bandwidth consumption and system unreliability [1]. However this also leads to a problem in replica consistency management. Namely when data is accessed and read by multiple users, inconsistency occurs in replicas, and as a result the system's consistency and accuracy are directly influenced.

The study on replica consistency aims at achieving synchronism among multiple replicas. There are two replica synchronism strategies [2]: strong consistency and eventual consistency. Strong consistency means that when data is updated, all replicas are updated simultaneously and that all replicas are consistent. Eventual consistency means that unnecessarily all replicas are kept consistent with each other and that a certain replica is updated only when it is accessed, hence no need to transfer updated data to other replicas for each update. Strong consistency helps timely access to the most updated data, but increases network bandwidth consumption due to frequent updates. Eventual consistency, on the other hand, reduces network bandwidth consumption but results in prolonged access time lag and relatively low file access efficiency.

Currently, there are many works focusing on the research of replica consistency strategy. The reference [3] proposes that only a certain replica is chosen to be updated according to the file's reading frequency and that all replicas are updated when reading frequency surpasses the predefined threshold value. Wang, etc., proposes four strategies in achieving consistency [4]. The proposed strategies are based on a file's reading and writing frequencies, so that when a system is running, an appropriate consistency strategy is adopted according to the dynamic running status of system and a balance is struck between overhead and performance. However, above mentioned methods don't adequately fit and predict the access behavior for a file adaptively, which poses adverse impacts on the selecting of appropriate consistency strategy. Because of above situation, the methods in [3] and [4] would not achieve great performance both in file access efficiency and bandwidth consumption.

doi:10.4304/jcp.9.8.1928-1933

Based on the above description it is obvious that the key to achieving adaptive replica consistency lies in predicting future file access behavior and selecting a corresponding consistency strategy according to real-time file access situation. Based on the impact of file access time sequence, this paper proposes a file heat calculation method, the file heat calculated by which can accurately predict the future file access behavior. The fact that the file access behavior is based on file heat means that when heat goes up, the future access goes up, and when heat goes down, the future access goes down. Based on this file heat calculation method, a file system is able to adaptively select the corresponding replica consistency strategy according to the changing file heat and achieve adaptive consistency among replicas in cloud storage, thus guaranteeing file access efficiency and reducing network bandwidth consumption.

II. Related Works

Reference [2] introduces the Replica Consistency Service (RCS), which was developed by the European Data Grid Project. The responsibility of RCS is to maintain consistency among existing replicas. To guarantee local consistency, the Local Consistency Service is performed within each storage unit. RCS forces on the consistency problem considering replica metadata. A file locking mechanism is also proposed in RCS for serialization of file access.

Reference [3] proposes that only a certain replica is chosen to be updated according to the file's reading frequency and that when the reading frequency surpasses the predefined threshold value all replicas are updated.

The strategies, proposed by Wang, etc., in [4], are based on a file's reading and writing frequencies, so that when a system is running, an appropriate consistency strategy is adopted according to the dynamic running status of system and a balance is struck between overhead and performance.

In [5], K.Sashi, etc., provide an algorithm which balances a replica's consistency. This algorithm, a combination of aggressive-copy and lazy-copy algorithms, selects a random replica for consistency update in a grid.

III. The File Heat Calculation Method

In cloud storage the heat of file constantly changes according to real access situation. In this method file heat is calculated for every time interval I, and a corresponding consistency strategy is selected according to the file heat result, which helps achieve self-adaptive replica consistency. The new file heat value is calculated based on current file heat in a time interval I and the historical one with different weights.

$$\begin{cases} Heat_i(f) = 0, \mathrm{i} = 0 \\ Heat_i(f) = aC_i + (1-a)Heat(f)_{i-1}, \mathrm{i} \geq 1 \end{cases} \quad (1)$$

In the above equation, f refers to a file; $Heat_i(f)$ refers to the historical file heat in the ith I time interval; C_i refers to the file heat currently computed in the ith I time interval; the symbol a refers to the weight of C_i. The larger the value of a is, the bigger the influence C_i has on the value of $Heat_i(f)$ and vice versa. The $Heat_i(f)$ is used to predict the file access behavior in the (i+1)th I time interval.

Since file's historical heat is used to calculate the current heat value in this method, the file heat can be kept at a relatively stable level and the impact of heat fluctuation on heat calculation is also kept at a minimal level. Additionally, since current heat takes up a big weight, the impact of which is also big, while historical heat's influence is reduced. This means the current access behavior to the file can be accurately revealed.

However, when we compute the current heat of a file, the pattern of file access time sequence in a time interval should be taken into account. If time interval I is equally divided into 10 parts, the different patterns of file access time sequence are shown in Fig.1.

In Fig.1 the access time of pattern 1 concentrates on the beginning part of I. For pattern 2 the access time concentrates on the middle part of I. And for pattern 3 the access time concentrates on the final part of I. Based on data's locality principle, during time interval I, though the file access frequency is the same, but the possibility of future access to the file of access pattern 2 is higher than pattern 1, and the possibility of future access to the file of access pattern 3 is the highest.

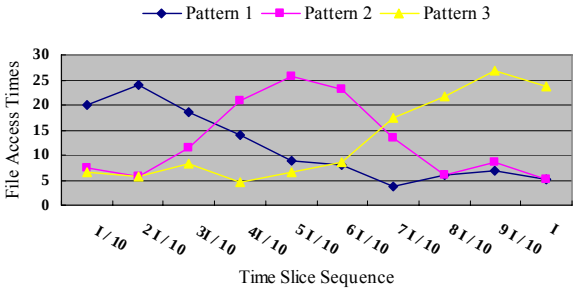


Figure 1. Patterns of file access time sequence

Therefore when we calculate file heat during I, not only access frequency during I, but also the impact of access time sequence on file heat should be taken into consideration. In this paper we adopt the MRFU algorithm to calculate the file heat during a time interval I. The MRFU algorithm is a combination of two principles: MRU (Most Recently Used) and MFU (Most Frequently Used). It gives due consideration to both access time and access frequency. The principle involved is that the weight of the last access is the biggest and it gets smaller in each previous access. Based on the above mentioned discussions, the computational formula for the current file heat C_i is shown below.

$$C_i = \sum_{k=1}^{m} w(T_i - t_i^k) \quad (2)$$

In formula (2), m denotes the file access times during time interval I, T_i represents the final time of time interval

I and $\{t_i^1, t_i^2, \ldots, t_i^m\}$ represents the m time points when the file is accessed. The weight function $w(x)$ for each t_i^k is defined as follows.

$$w(x) = p^{\lambda x}, 0 < p < 1 \qquad (3)$$

IV. The Self-adaptive Replica Consistency Strategy

This paper adopts a replica management model, as show in Fig.2.

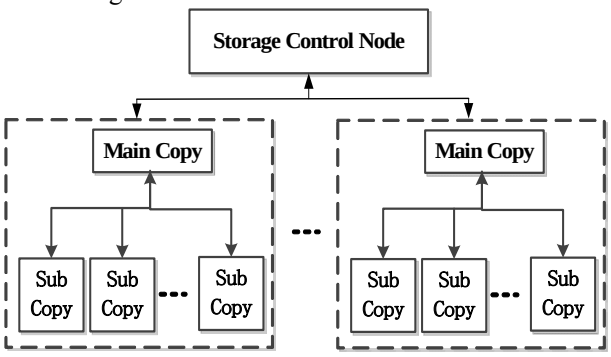


Figure 2. Replica management model

The storage control node collects information of all the storage nodes, the location of nodes, reading frequency and the number of replicas. It is responsible for receiving user's reading and writing requirement and executes actual operation according to the replica consistency strategy.

In the system replicas are divided into two categories: main replicas and sub replicas. Main replicas can be modified by the end user, and the content of sub replicas is updated according to main replicas. In our model, the number of sub replicas can be dynamically adjusted according to system's loading situation.

To guarantee file access efficiency and reduce network bandwidth consumption at the same time, this paper employs a self-adaptive replica consistency strategy which intends to predict the access behavior based on the file access behavior during time interval I and the access behavior is evaluated by the metric, file heat. An upward trend of file heat indicates more future access and vice versa. In order not to increase access delay, the strong consistency method is adopted when file heat is high; and the eventual consistency method is adopted to cut network bandwidth consumption when file heat is low. Therefore the storage control node records the access time sequence of files and the heat values of files are calculated based on these access time sequences. When file heat exceeds the predefined threshold value, the strong consistency strategy will be adopted to update replicas and when file heat is under the threshold value, the eventual consistency strategy would be adopted.

With our replica consistency strategy, the operation procedure for reading requests is shown in the right branch of Fig.3. If the accessed file is a sub replica, it is necessary to identify the consistency strategy currently used. If it's strong consistency strategy, then the sub replica should be directly accessed. If it's eventual consistency strategy, the sub replica is updated through main replica before it is accessed. If the accessed is a main replica, the direct access happens.

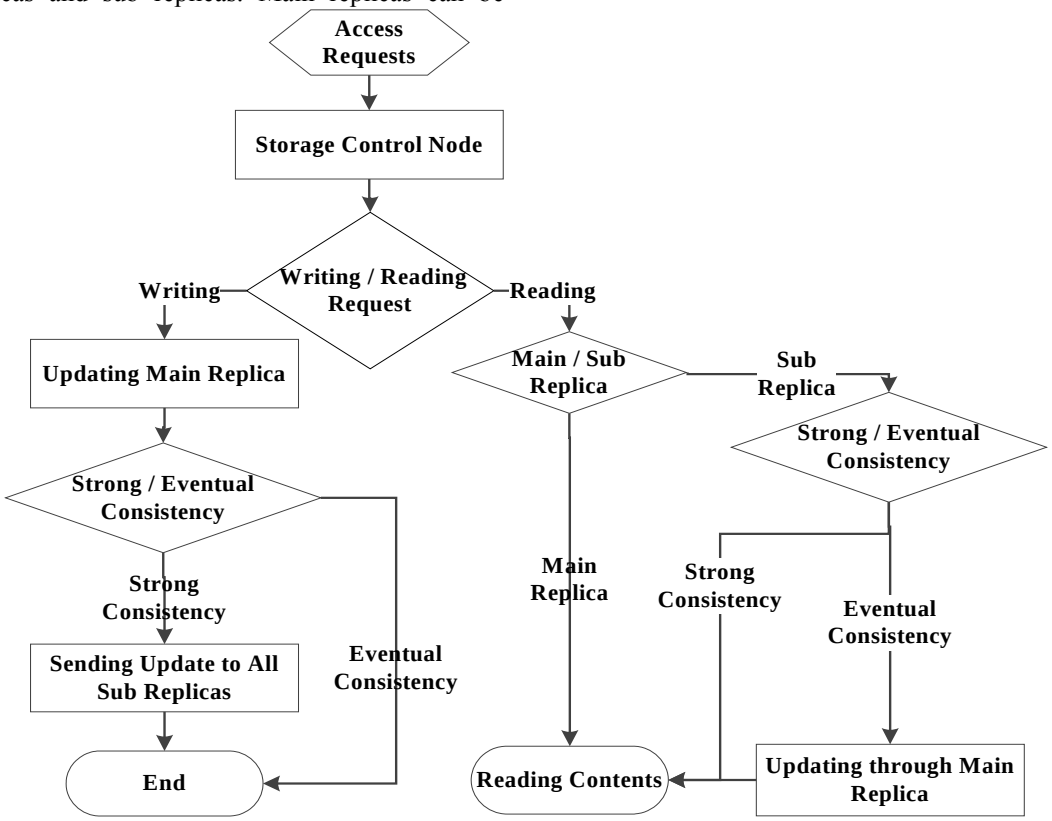


Figure 3. Operation procedure for reading / writing request

The operation procedure for writing requests is indicated in the left branch of Fig.3. When the update request for a file is submitted to the storage control node, the corresponding main replica of the file is updated and the updating of its sub replicas is based on the current consistency strategy adopted for the file. The update is transferred to all sub replicas from their main replica if the adopted strategy is strong consistency, and the sub replicas would not be updated if the strategy is eventual consistency. This self-adaptive consistency not only helps speedy access to the files with high access frequency, but also reduces network bandwidth consumption when updating sub replicas of the files with low access frequency.

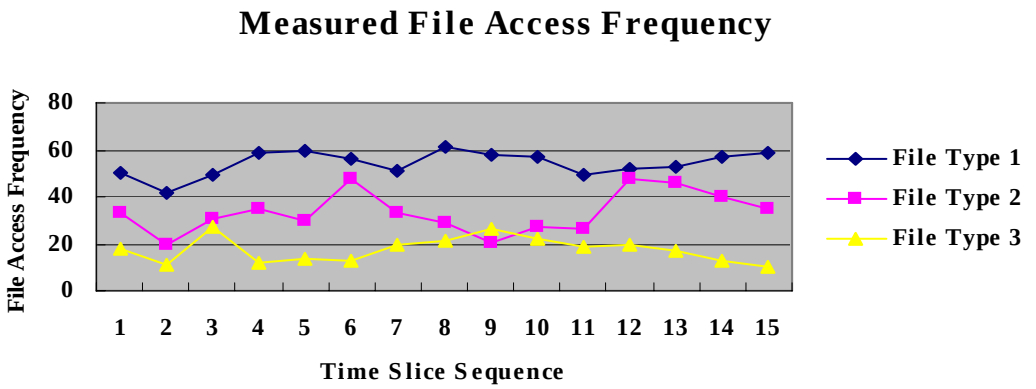
V. The Experiments and Result Analysis

The Optorsim grid mock software is used in the experiments. Optorism is a simulating tool written in java, and is often used to simulate actual grid. Based on above mentioned simulation environment, the copying strategy can be studied. The grid built in Optorsim includes multiple nodes, each of which is able to provide calculation and storage resources for tasks. The running environment of Optorsim is as follows: the storage control node (CPU-Xeon E5606, memory-32GB, hard disk-SATA 500G/7200); ten storage servers (CPU-i5, memory-4GB, hard disk-SATA 500G/7200).

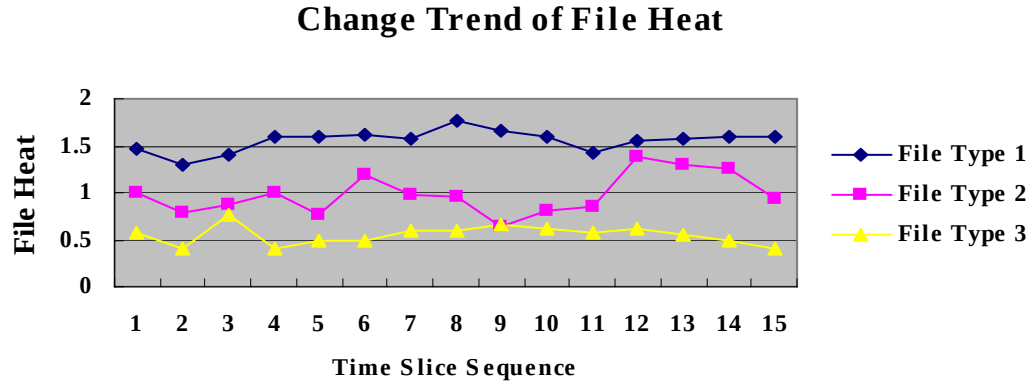
A. Test and Analysis of MRFU Algorithm

In the test the trace data on the access behavior of three different kinds of files is retrieved from an FTP file server in 15 equal time periods. The file type 1 represents the popular files which would be accessed by high frequency, while the file type 3 represents the unpopular ones which would be accessed by low frequency. In addition, the file type 2 simulates the files which be accessed by fluctuating frequency. Then the file heat is calculated for every period. Fig.4 (a) indicates file access frequency practically measured in the 15 periods, while Fig.4 (b) shows the changing trend of the file heat calculated by MRFU algorithm.

As demonstrated by Fig.4 (b), the file heat calculated by MRFU algorithm has an identical changing trend of file's actual access situation. This MRFU algorithm can be used to precisely predict the upward and downward trend of file heat, thus providing a solid foundation for selecting the adequate replica consistency strategy.



(a)



(b)

Figure 4. Validity test for MRFU algorithm

B. Test and Analysis of the Self-adaptive Replica Consistency Strategy

The purposes for proposing this self-adaptive consistency strategy are to enhance file access efficiency on the basis of maintaining relatively low network bandwidth consumption when cloud storage system kicks in for replica management. To test the proposed strategy, metrics the hit rate of accessing new data and the number of updating operations for sub replicas in a set time interval *H* are retrieved and compared with other consistency strategies. It is revealed that the higher is the hit rate of accessing new data, the shorter the access time lag is, and that the lower the hit rate of accessing new data, the longer the access time lag. Furthermore, the smaller the number of updating operations for sub replicas there is, the lower the consumption of network bandwidth, and the larger the number of updating operations for sub replicas there is, the higher the consumption of network bandwidth. Since the test in this paper is conducted in a simple manner, the time interval *H* is only set at 5 minutes and the file heat is calculated for every other minute.

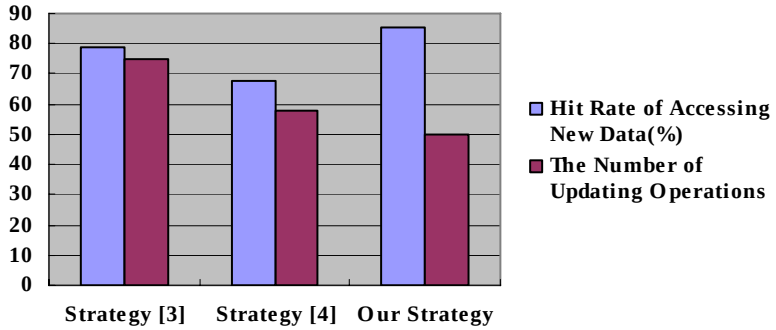


Figure 5. Contrast test between our strategy and ones in reference [3-4]

Fig.5 shows the comparisons between our strategy and the ones proposed by reference [3] and [4]. Fig.5 indicates that with our strategy the hit rate of accessing new data is higher than those of the strategies in [3] and [4], while the number of updating operations for sub replicas is lower than the strategies in [3] and [4]. The reason is that: Compared with other existing strategies, such as these in [3] and [4], our strategy takes into account the effect caused by the pattern of file access time sequence in a time period in the calculating of file heat and uses the algorithm MRFU to effectively quantize this effect. The file heat calculation method in our paper calculates the file heat at finer granularity, which quantizes the different contribution of each access in the access sequence on the file heat, thus being able to more accurately predict future file access behavior. Based on the more accurate file heat, our strategy can adopt more properly consistency strategy, i.e., strong consistency strategy or eventual consistency strategy, to the files with different file heat value, achieving high hit rate of accessing new data and effectively cutting the number of updating operations for sub replicas.

VI. CONCLUSION

This paper probes into a self-adaptive replica consistency strategy which allows the system to automatically select the adequate consistency strategy according to file heat. In the calculating of file heat, MRFU algorithm is used. The experiment results show that the file heat calculated by MRFU algorithm can precisely predict file access behavior. MRFU algorithm provides a solid foundation for the implantation of our self-adaptive replica consistency strategy. The corresponding experiment results prove that on the basis of keeping high file access efficiency, the proposed self-adaptive replica consistency strategy effectively reduces the network bandwidth consumption. Furthermore, by contrast tests, it is found that the proposed self-adaptive replica consistency strategy has better performance in file access efficiency and cost of network than other similar solutions.

ACKNOWLEDGEMENTS

We are grateful to the editor and anonymous reviewers for their valuable comments on this paper. The work of this paper is supported by National Natural Science Foundation of China (Grant No. 61272399), Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20110191110038) and Fundamental Research Funds for the Central Universities (Grant No. CDJXS12180001).

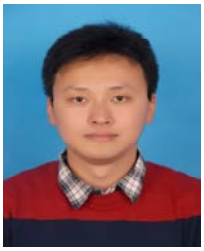
REFERENCES

[1] Heinz Stockinger, Asad Samar, Bill Allcock, Ian Forster, Koen Holtman, Brain Tierney, “File and object Replication in Data Grids”, Journal of Cluster Computing, 2002, pp.305-314.

[2] Andrea Domenici, Flavia Donno, Gianni Pucciani, Heinz Stockinger, Kurt Stockinger, “Replica consistency in a Data Grid”, IX International Workshop on Advanced Computing and Analysis Techniques in Physics Research(ACAT03), Tsukuba, Japan, December 2003.

[3] Ruay-Shiung Chang, Jih-Sheng Chang. Adaptable Replica Consistency Service for Data Grids [C]//Third International Conference on Information Technology: New Generations (ITNG'06). 2006.

[4] Wang Ximei, Yang Shoubao, Wang Shuling, et al. An application-based adaptive replica consistency for cloud storage[C]//Proc of the 9th International Conference on Grid and Cloud Computing. 2010,16:13-17.
[5] K.Sashi, Dr.Antony Selvadoss Thanamani.A New Replica Creation and Placement Algorithm for Data Grid Environment[C]//Proc of 2010 International Conference on Data Storage and Data Engineering. 2010,38:265-269.
[6] Cameron D.G., Carvajal-Schiaffino R, Millar A.P., Nicholson C, Stockinger K, Zini F "Evaluating Scheduling and Replica Optimization Strategies in OptorSim", Forth International Workshop on Grid Computing, Phoenix, USA, November 2003.
[7] W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, K.Stockinger, "Data Management in an International Data Grid Project", IEEE/ACM International Workshop on Grid Computing (Grid' 2000), London, UK, December 2000.
[8] Abdelkrim Beloued, J.-M. G., Maria-Teresa Segarra, Francoise Andre. Dynamic Data Replication and Consistency in Mobile Environments. ACM International Conference Proceeding Series 114. 2009: 1-5.
[9] Ghalem, B. and S. Yahya. Consistency Management for Data Grid in OptorSim Simulator. Proceedings of the 2007 International Conference on Multimedia and Ubiquitous Engineering, IEEE Computer Society. 2007.
[10] Zhou, X., X. Lu, et al. A dynamic distributed replica management mechanism based on accessing frequency detecting. SIGOPS Oper. Syst. Rev. 2004, 38(3): 26-34.
[11] Susarla, S, J. Carter. Flexible consistency for wide area peer replication [A]. In: Proceedings - International Conference on Distributed Computing Systems[C], Columbus, OH, United states: Institute of Electrical and Electronics Engineers Inc, 2005: 199-208.
[12] Haifeng, Y. and V. Amin. The costs and limits of availability for replicated services. ACM Trans. Comput. Syst. 2006, 24(1): 70-113.
[13] Haifeng, Y. and V. Amin. Design and Evaluation of a Conit-Based Continuous Consistency Model for Replicated Services. ACM Transactions on Computer Systems Vol. 20, No. 3, August 2002: 239–282.
[14] Chunxia Zhang, Baoxu Shi, Xudong Li. MFS: A Lightweight Block-level Local Mirror of Remote File System. Journal of Software Vol. 8, No. 6, June 2013: 1459-1470.
[15] Nianmin Yao, Jinzhong Chen, Shaobin Cai. A Non-partitioning File Assignment Scheme with Approximating Average Waiting Time in Parallel I/O System. Journal of Software Vol. 8, No. 2, February 2013: 302-309.
[16] Yihua Lan, Yong Zhang, Haozheng Ren. Size-based Data Placement Strategy in SAN. Journal of Software Vol. 8, No. 2, February 2013: 426-434.



Zhen Zhou received his B.S. and M.S. degrees in Computer Science and Technology repectively from Chongqing University of Technology, China, in July 2006 and Chongqing University, China, in July 2010. Since September 2010, he has been working toward the Ph.D. degree in Computer Science and Technology at Chongqing University. His research interests include cloud computing, anomaly detecting, and virtual machine technique.

Shuyu Chen received his B.S., M.S., and Ph.D. degrees in Computer Software and Theory from Chongqing University, China, in 1984, 1998, and 2001. From 1995 to 2005, he was with the College of Computer Science, Chongqing University. Since 2005, he has been with the School of Software Engineering, Chongqing University, where he is currently a professor. His current research interests include dependable computing, cloud computing, and Linux operating system. He has published more than 100 papers in international journals and conference proceedings.

Tao Ren received his B.S. degree in Chongqing University, P. R. China, at 2010. He is currently a M.S. candidate in College of Computer Science, at Chongqing University. His current interests include cloud computing, and dependable computing.

TianShu Wu received his B.S. degree in Chongqing University of Posts and Telecommunications, P. R. China, at 2011. He is currently a Ph.D. candidate in College of Computer Science, at Chongqing University. His current interests include cloud computing, large-scale data mining and fault detection.