Improve the Performance of Data Grids by Value-Based Replication Strategy

Wuqing Zhao[#1], Xianbin Xu[#2], Zhuowei Wang[#3], Yuping Zhang[#4], Shuibing He[#5]

[#]*School of Computer, Wuhan University*
Wuhan, China

[1]whuzhwq@163.com
[2]xbxu@whu.edu.cn
[3]wangzhuowei0710@163.com
[4]yuping.whu@gmail.com
[5]hesbingxq@163.com

***Abstract*— Data grid, which provides the capability to store the mass data, has become one of the hottest topics in distributed storage research domains. However, the limited capacity of local storage devices and the high latency of network have become the bottlenecks for efficiently accessing the files in the data grids. These problems can be solved through introducing a replication strategy by which the overall performance of the system can be guaranteed. The replication strategy tries to creat multiple replicas in the distributed memory of data grid. In this paper, we propose a value-based replication strategy (VBRS) to decrease the network latency and meanwhile to improve the performance of the whole system. VBRS can be described as two steps. First, the threshold to decide whether a file should be replicated in the local storage device is introduced according to the access history and the storage capacity. Second, a measure, based on the values of the local replicas, is devised to choose the replica that should be replaced. The experiment results confirm the effectiveness of our proposed algorithm.**

I. INTRODUCTION

Data Grid is one of the major topics in the grid community, which provides geographically distributed storage resources to the computation tasks. In tasks such as distributed scientific computation, we often need to access massive data or generate a large amount of data. For example, scientific areas such as high energy physics, molecular modeling and earth sciences involve the production of large datasets from simulations or large-scale experiments. These datasets have tremendous amount of data, and evaluating and managing such large amount of data in each storage resource is a hot topic in the data grid community.

In order to manage the huge amount of data and share these data and resources in an efficient way, the data should be replicated and placed in several storage resources to satisfy the access need, which can decrease the access latency. Data replication, as an important technique to speed up data access, is to replicate the data in storage resources, so that a user can access the data in the local (nearest) site. Data replication not only reduces access costs, but also increases data availability in many applications.

In data grids, replica management is to create or delete replicas at a storage resource. To create a replica, we have to answer some questions, for example, which file should be replicated? Where the file should be placed? And when should the replicas be created? According to the above questions, Some strategies have been proposed. In VBRS, we suppose the storage capacity is limited, so another question can't be avoided, which is a well-designed replication replacement algorithm.

In the recent works, the economy model has been used rather broadly and successfully in the data grid research [1, 2], and the strategy - latest access largest weight (LALW) is based on the access-weight from calculating each file's value [4]. In [7], they propose a mechanism to keep the data most likely to be used from deploying retention value functions. In this paper, we propose a value-based replication strategy, which is based on the access history, i.e., file's size and the network condition. From calculating file's value, it is adaptive to dynamic data grid environment and can enhance the efficiency of data access. The better performance is illustrated by the experiment.

We show the remainder of the paper in the following sections. Section 2 interprets the related work. Section 3 presents the design rationale and VBRS in detail. In order to show the performance of our proposed strategy, the simulation results in OptorSim are presented in Section 4. The last section summarizes the paper and proposes the future works.

II. RELATED WORK

This section summarizes the existing works for replication strategy in data grids, which includes some traditional strategies, economy-based predictive strategies and weight-based strategies.

In [3], Ranganathan and Foster introduced six different replication strategies: (1) No replication and caching, (2) Best client, (3) Cascading replication, (4) Plain caching, (5) Caching plus cascading replication, (6) Fast spread. These strategies are evaluated with different data patterns in the simulation tool. The results show different access pattern needs different replication strategy. When data grids selects the proper strategy, the performances in bandwidth savings and access latency can be improved. Cascading and fast Spread are the best strategies with the lowest response time and bandwidth consumption.

978-0-7695-4189-1/10 $26.00 © 2010 IEEE
DOI 10.1109/SKG.2010.48

In [1, 2], they presented the resource management techniques, based on an economic model, proposed policies for data replication broker and discussed its use in Data Grids to optimize both replica selection and dynamic replication. The economic model-based data resource management system shows improvements when using a variable pricing scheme with considering both data locality and network. They also discussed the prediction function the agents use for making replication decisions, which uses historical data about file access pattern.

In [4], they propose a weight-based dynamic replication strategy. They collect the data access history, which contains file name, the number of requests for file, and the sources that each request came from, and then calculates the different file's weight according to their ages. According to the metric, a popular file is found and replicated to suitable sites to improve the system's performance, including the effective network usage and storage usage. But the total job execution time of this strategy is similar to other strategies.

III. Replication Strategy

A. Design Rationale

In the process of job computing, when computing sites request massive files from different storage sites. If the files are stored in the local storage sites, the computing sites can get it directly with short latency. If the file isn't in the local site, in order to fetch the ones from other remote sites, it is better to make a replica in the local site. A good replication strategy can reduce file access time, access latency and overall computing time, and increase the network usage.

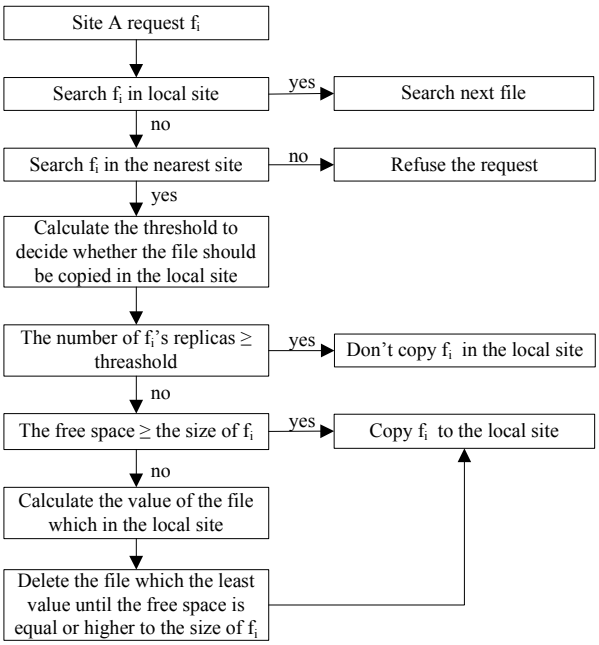

Fig. 1 Value-Based Replication Strategy

In order to effectively utilize the storage capacity, we need to devise proper strategy to decide, which files should be copied, and where the file should be placed. Because the storage capacity is limited, the replica replacement strategy is also useful, when there is not enough space left to copy the requested new files, which is not in the local storage site. To solve the above questions, we propose the value-based replication strategy (VBRS).

In VBRS, we make a threshold to decide whether copy the requested file, and then solve the replica replacement problem. VBRS is executed from two steps. In the first step, according to the access history and system's storage capacity, we calculate the threshold to decide whether the requested file should be copied in the local (nearest) storage site. In the second step, we devise a novel replication replacement algorithm. When the requested file needs to be copied, the local (nearest) storage site has no enough space, the replacement algorithm is triggered. It firstly calculates the files' values in the local storage site. Then, the replacement algorithm chooses the files, which has the least value, to be deleted. The files' value mostly concerns with three factors: network bandwidth, file's size, and the access history. The files with higher value should be retained, so the files with lower value should be deleted. We present the value-based replication strategy in Fig.1, and describe it in detail in last part.

B. Replication Strategy

In this part, we first calculate the number of the file which should be copied in the system. After that, if the number of file's replicas is less than the threshold, the file can be copied to the local site. But the local site's storage capacity is limited, it is necessary to delete some files. Which file should be deleted? We design an effective algorithm to solve this question.

Then, we calculate the file's value, noted as $Value(f)$, based on the value $V(f)$ of the file f, access cost $C(f)$ and the maximum bandwidth $B(f)$ with the neighbors in the system.

1) Calculate the replica's threshold

First, we can get the sum of all nodes' storage capacity (C_{sum}) from the resource manager in the system, and every file accessed times (F_{f_i}) from the access history.

And then we can calculate the access frequency (AF_{f_i}) of file f_i, which is defined as below.

$$AF_{f_i} = \frac{F_{f_i}}{\sum_{i=1}^{n} F_{f_i}} \tag{1}$$

Last, we can get the number of every file's replicas (N_{f_i}), which is calculated from the below function, based on the above factors.

$$N_{f_i} = \frac{AF_{f_i} * C_{sum}}{S_f} \tag{2}$$

S_f : the size of file f .

N_{f_i} is the threshold to decide whether the file will be copied. When the number of the file's replicas is greater than N_{f_i} , the file will not be copied into the local node, but when the number of the file's replicas is less than N_{f_i} , the file can be copied.

2) Calculate the replica's value

Because the storage capacity is limited, in order to make enough local space for replication, we propose an effective replacement strategy to delete the file, which has the least value.

We devise the value function that can be roughly characterized as having three phases. The value is high but declining slowly in the first phase, in the second phase, the value is declining sharply, and in the last, the value is low and declining slowly. In all the phases, the value function is monotonic decreasing. Setting different values is used to evaluate the importance for history records. Older history records have smaller values, and the recent history tables are worthier than the previous access. The function can be indicted below.

$$V(f) = \sum_{k=1}^{n} arc\cot(\frac{t_k - \theta}{\sigma}) \qquad (3)$$

t_k : the time of which file f had been accessed in the k_{th} time.

θ and σ are variables.

In the above function, we only take into the factors, the file's accessed time and frequency. The file's size $S(f)$ and network bandwidth also are important factors during the whole replica replacement process. Too much bandwidth consumption may affect the network condition and increase the possibility of fault appearance during the transfer process. Consequently, the lower the bandwidth consumption is, the better the performance of replacement algorithm is.

So we chose the best one--the one with the largest bandwidth $B(f)$, to transfer the related replica in the beginning of the transfer process, when the replica can't be found in the local site. We can get the file at the least access cost, which can be calculated in the function as below

$$C(f) = \frac{S(f)}{B(f)} \qquad (4)$$

Based on the above definitions, the value of the file f is defined as:

$$Value(f) = V(f) * C(f) = \frac{V(f) * S(f)}{B(f)} \qquad (5)$$

According to the above functions, first, we calculate the value of every file in the local site. Then, we sort the files by the values in ascending order, select the file, which has the least value, from the sorted file list and add it into the deleted list until the accumulative size of the selected files is larger than or equal to the requested file. The last step is to delete the selected files, and copy the new file.

IV. Performance Evaluation

In order to demonstrate the performance improvements of VBRS, which is proposed in this paper, we use the Grid simulator OptorSim, which is developed in Java, to evaluate the performance. OptorSim can simulate the real Data Grid environment. We can easily compare the effectiveness of VBRS with other strategies in the simulator.

A. Simulator and Parameters

We evaluate VBRS and other four strategies using OptorSim. It contains several elements including Computing Element components (CEs), Storage Element components (SEs), Resource Broker (RB), Replica Manager (RM) and Replica Optimiser (RO) [5]. The OptorSim tool works based on three configuration files, which are grid configuration file, job configuration file and bandwidth configuration file. Grid configuration file specifies the Grid topology and the contents of each site. Job configuration file contains information for the simulated jobs, and specifies the jobs each site will accept. Bandwidth configuration file contains information for the simulated bandwidth. In addition, there is a parameter file in the OptorSim system, which is used to define some simulation parameters.

At the beginning of simulation, all the files are stored at the center site, which has a huge SE but no CE. When the simulation starts, the local site's replica manager decides whether and how to make the replications based on the specific policy, and select files to delete based on the proper replacement algorithm when the storage capacity is full.

The parameter values in the simulation appear as below. The topology of our simulated platform includes 18 sites comprised of 10CEs and 11SEs. Each site has different storage capacity. There are 97 different files, which are stored in the Grid and 2000 jobs. Each job accesses 2–50 files and each file size is 1 G. The storage available at an SE ranges from 30 G to 100000 G. θ and σ in formula (3) are set proper values. In order to simplify the requirements, we do not consider the write and consistency of replicas. When evaluating the performance of the data replication strategies, we usually assume that the data is read-only in the Data Grid environment.

B. Simulation Results and Comparisons

In our simulation, we compare VBRS with other four replication strategies.

1) No replication: It means that there is no replica in all the computing process.

2) Least Recently Used (LRU): In the computing process, it always makes replication until there is not enough space. Then according to LRU, it deletes the file, which is the oldest file, to make enough space for the new replication.

3) *Least Frequency Used (LFU):* It always makes replication as the above strategy until there is not enough

space. Then according to LFU, it deletes the file, which has the least frequency of request, to make enough space for the new replication.

4) *Latest Access Largest Weight (LALW):* It calculates the files' values, and then chooses the most popular file for replication. At last, it calculates a suitable number of copies and selects proper grid sites for replication.

Fig. 2 shows the mean job time of the five replication strategies. The job execution time has about 20% difference in the five different strategies. It indicates that VBRS leads to its better performance than other four strategies.

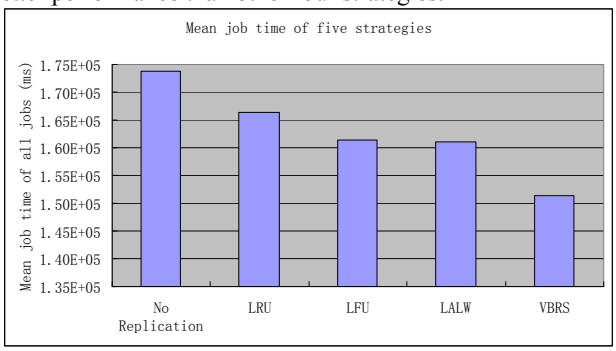

Fig. 2 Mean job time of three strategies.

An evaluation metric Effective Network Usage in OptorSim is used to estimate the efficiency the network resource usage, noted as E_{enu} [6]. It is defined as:

$$E_{enu} = \frac{N_{rfa} + N_{fr}}{N_{lfa} + N_{rfa}} \quad (6)$$

Where N_{rfa} is the number of access times that CE reads a file from a remote site, N_{fr} is the total number of files' replicas, and N_{lfa} is the number of times that CEs read files locally. A lower value indicates that the network bandwidth is used more efficiently.

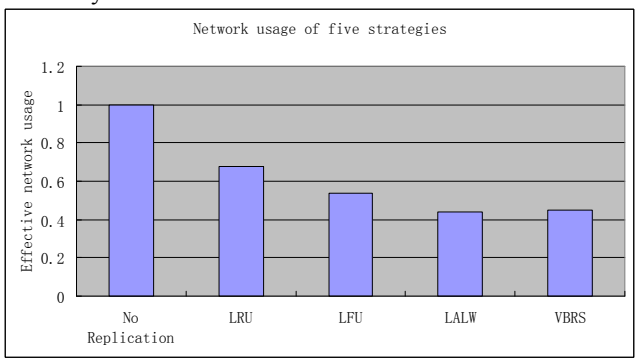

Fig. 3 Effective network usage of three strategies.

Fig. 3 shows the comparison of ENU (effective network usage) of the five replication strategies. The ENU of our algorithm is lower about 30% compared to others. The reason is that LFU and LRU always replicate, so the large value of N_{lfa} will increase the ENU value, whereas No Replication strategy don't make any replication, so the remote access will increase the ENU, too.

V. Conclusion and Future Work

In this paper, we propose a value-based replication strategy. The VBRS consists of two steps. In the first step, we calculate the ideal threshold to decide whether the file should be copied or not, according to the system's access history and storage capacity. Then, we design an efficient replica replacement strategy to solve the problem of limited storage space in each node. The replica replacement policy is developed by considering the replica's value which is based on the file's access frequency and access time. Finally, we compared the VBRS with other four common replica selection strategies. Experimental results demonstrate that our VBRS strategy can achieve a significant improvement of over former similar work. Currently, our work only focuses on replication strategy for creating/deleting replicas. To improve the current approach further, there are some aspects that can be considered. First, we will employ the replica placement strategy as an important part of the future work. Second, given the significant improvement of the current approach, we can still develop the better target function to attain better accuracy.

References

[1] William H.Bell, David G.Cameron, Ruben Carvajal-Schiaffino, A.Paul Millar, Kurt Stockinger, and Floriano Zini, "Evaluation of an Economy-Based File Replication Strategy for a Data Grid", Proceedings of the 3rd International Workshop on Agent based Cluster and Grid Computing at International Symposium on Cluster Computing and the Grid, IEEE Computer Society, 2003, pp. 661-668.

[2] Mark Carman, Floriano Zini, Luciano Serafini, and Kurt Stockinger, "Towards an Economy-Based Optimization of File Access and Replication on a Data Grid", Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, IEEE Computer Society, 2002, pp.340-345.

[3] Kavitha Ranganathan, and Ian T. Foster, "Identifying Dynamic Replication Strategies for a High Performance Data Grid", Proceedings of the Second International Workshop on Grid Computing, Springer-Verlag, 2001, pp. 75-86.

[4] Ruay-Shiung Chang, Hui-Ping Chang, "A dynamic data replication strategy using access-weights in data grids", Journal of Supercompute, 2008.

[5] David G. Cameron, Rubén Carvajal-schiaffino, A. Paul Millar, Caitriana Nicholson, Kurt Stockinger, and Floriano Zini, "OptorSim: A Grid Simulator for Replica Optimization", UK e-Science All Hands Meeting, 2003.

[6] David G. Cameron, Rubén Carvajal-schiaffino, A. Paul Millar, Caitriana Nicholson, Kurt Stockinger, and Floriano Zini, "UK Grid Simulation with OptorSim", UK e-Science All Hands Meeting, 2003.

[7] KIRSTEN HILDRUM, FRED DOUGLIS, JOEL L. WOLF, and PHILIP S. YU. "Storage Optimization for Large-Scale Distributed Stream-Processing Systems". ACM Transactions on Storage, Vol. 3, No. 4, Article 18, February 2008.