Available online at www.sciencedirect.com

ScienceDirect

Future Generation Computer Systems 23 (2007) 846–860

FGCS
FUTURE GENERATION COMPUTER SYSTEMS

www.elsevier.com/locate/fgcs

Job scheduling and data replication on data grids

Ruay-Shiung Chang*, Jih-Sheng Chang, Shin-Yi Lin

Department of Computer Science and Information Engineering, National Dong Hwa University, Shoufeng, Hualien 974, Taiwan

Received 25 July 2006; received in revised form 23 February 2007; accepted 27 February 2007
Available online 16 March 2007

Abstract

In data grids, many distributed scientific and engineering applications often require access to a large amount of data (terabytes or petabytes). Data access time depends on bandwidth, especially in a cluster grid. Network bandwidth within the same cluster is larger than across clusters. In a communication environment, the major bottleneck to supporting fast data access in Grids is the high latencies of Wide Area Networks (WANs) and Internet. Effective scheduling in such network architecture can reduce the amount of data transferred across the Internet by dispatching a job to where the needed data are present. Another solution is to use a data replication mechanism to generate multiple copies of the existing data to reduce access opportunities from a remote site. To utilize the above two concepts, in this paper we develop a job scheduling policy, called HCS (Hierarchical Cluster Scheduling), and a dynamic data replication strategy, called HRS (Hierarchical Replication Strategy), to improve the data access efficiencies in a cluster grid. We simulate our algorithm to evaluate various combinations of data access patterns. We also implement HCS and HRS in the Taiwan Unigrid environment. The simulation and experiment results show that HCS and HRS successfully reduces data access time and the amount of inter-cluster-communications in comparison with other strategies in a cluster grid.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Data replication; Data grid; Job scheduling

1. Introduction

In data grids [1,2], distributed scientific and engineering applications often require access to a large amount of data (terabytes or petabytes). Managing this large amount of data in a centralized way is ineffective due to extensive access latency and load on the central server. Hence, such huge dataset must be separated and stored in several physical locations.

In a communication environment, the performance of accessing a distributed and huge amount of data depends on the availability of network bandwidth. Namely, slow data access can throttle the performance of data-intensive applications running on grid computers. In Fig. 1, a simple hierarchical form of a grid system, called cluster grid, is shown. A cluster represents an organization unit which is a group of sites that are geographically close. We define two kinds of communications between sites in a cluster grid. Intra-communication is the communication between sites within the same cluster. On the other hand, inter-communication is the communication between sites across clusters. Network bandwidth between sites within a cluster will be larger than across clusters. Therefore, to reduce access latency and to avoid WAN bandwidth bottleneck in a cluster grid, it is important to reduce the number of inter-communications.

To address this problem, we consider two aspects of inter-communication: job scheduling and replication mechanism. Consider a case that many of the authorized users submit jobs to solve data-intensive problems. We want that jobs be executed as fast as possible. The size of the data used on Data Grid is from terabytes to petabytes. Scheduling jobs to suitable grid sites is necessary because data movement between different grid sites is time consuming. The scheduling decisions should be based on the appropriate resources a grid site has. Other factors to be considered include CPU workload, features of computational capability, location of data and network load. If a job is scheduled to a site where the required data are present, the job can process data in this site without any transmission delay for getting data from a remote site.

Data replication is another important optimization step to manage large data by replicating data in geographically distributed data stores. Previous replication strategies show

* Corresponding author. Tel.: +886 3 8632031; fax: +886 3 8632030.
E-mail address: rschang@mail.ndhu.edu.tw (R.-S. Chang).

0167-739X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2007.02.008

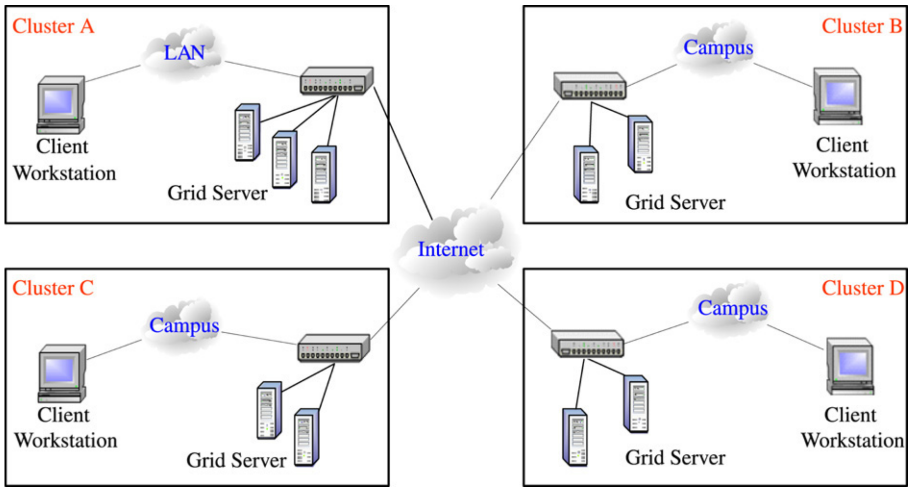

Fig. 1. A cluster grid architecture.

that replicating data can offer high data availability and low bandwidth consumption. When users' jobs access a large amount of data from remote sites, dynamic replica optimizer running in the site tries to store replicas on local storage for future possible repeated requests. If most data resides on the same site where they are needed, the frequency of remote data access is going to decrease. This can reduce job execution time and increase the robustness of grid application. Inter-cluster communications also can be avoided, if data within the same cluster are always accessed first.

Our new scheduling policy considers the locations of required data, the access cost and the job queue length of a computing node. It is called HCS (Hierarchical Cluster-based Scheduling). HCS uses hierarchical scheduling that takes cluster information into account to reduce the search time for an appropriate computing node. When data have to be replicated, we develop a replication strategy, called HRS (Hierarchical Replication Strategy). It integrates previous replication strategy and increases the chances of accessing data at a nearby node.

In order to study the complex nature of a typical grid environment and evaluate various replica optimization algorithms, a grid simulator, called OptorSim [4], was developed by EU Data Grid project [3]. Our job scheduling policy and replica strategy are simulated in OptorSim and compared with various scheduling policies and replica strategies. HCS and HRS successfully reduces data access time and the amount of inter-cluster communications in comparing to other combinations in a cluster grid. In addition, we have implemented HCS and HRS in the Taiwan UniGrid Platform [23] for observing the system performance on a real grid platform that contains many clusters distributed across the Internet.

This rest of this paper is organized as follows: Section 2 gives an overview of previous work on grid replication and job scheduling. Section 3 introduces our HCS policy and HRS replication strategy. We show results from simulations of HCS and HRS in Section 4. The implementation and performance evaluation is given in Section 5. Finally, Section 6 concludes the paper and outlines some future research work.

2. Related work

As jobs are data intensive, scheduling issues often involve effective computation and data management in the data grids. The replication of data sets is not really a new technique. Data replication has been around for decades and it is now adapted to the grid environment. In [5,6], Ranganathan and Foster present six different replica strategies: (1) No replication or caching, (2) Best Client: a replica is created at the best client that has the largest number of requests for the file, (3) Cascading replication: once popularity exceeds the threshold for a file at a given time interval, a replica is created at next level which is on the path to the best client, (4) Plain caching: the client that requests the file stores a copy of the file locally, (5) Caching plus Cascading Replication: this combines Plain caching and Cascading replication strategy, and (6) Fast Spread: replicas of the file are created at each node along its path to the client.

These strategies are evaluated with three different data patterns: (1) Random access: there is no locality in access patterns, (2) data access with a small degree of temporal locality (recently accessed file are likely to be accessed again), (3) Data access with a small degree of temporal and geographical locality. (Files recently accessed by a site are likely to be accessed by nearby site.)

The results of simulations indicate that different access pattern needs different replica strategies. With suitable strategies, they can dramatically improved the performances in bandwidth savings or access latency. Two strategies performed the best in the simulations: Cascading and Fast Spread when compared to traditional strategies.

Ranganathan and Foster also propose a variety of techniques to intelligently replicate data across sites and assign jobs to sites in data grid [7]. They have conducted a study of the performances of various scheduling algorithms by using simulator. A scheduler selects a remote site to dispatch a job based on one of the four algorithms: (1) JobRadom: schedule a job randomly, (2) JobLeastLoaded: schedule a job to where there is the least number of jobs waiting to run, (3) JobDataPresent: schedule a job to where it has the least load and requested data, and (4) JobLocally: always run jobs locally.