Available online at www.sciencedirect.com
SCIENCE DIRECT®
Future Generation Computer Systems 21 (2005) 271–280
FGCS FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/future

Heuristic scheduling for bag-of-tasks applications in combination with QoS in the computational grid

Chuliang Weng*, Xinda Lu

Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200030, PR China

Available online 26 November 2003

Abstract

The computational grid provides a promising platform for the deployment of various high-performance computing applications. One issue in implementing computational grid environments is how to effectively utilize various resources in the system, such as CPU cycle, memory, communication network, and data storage. A heuristic Qsufferage is presented to schedule the bag-of-tasks application in the grid environment. The algorithm considers the location of each task's input data, while makespan and response ratio are chosen as metrics for performance evaluation. The result of the experiment shows that Qsufferage algorithm can obtain better performance compared to the other four existing algorithms.

Keywords: Computational grid; Mapping strategy; Heuristic algorithm

1. Introduction

Advances in networking technology and computational infrastructure make it possible to construct large-scale high-performance distributed computing environments, or computational grids [1]. As a new infrastructure for next generation computing, computational grid enables the sharing, selection, and aggregation of geographically distributed heterogeneous resources for solving large-scale problems in science, engineering and commerce [2]. Many studies have focused on various aspects of grid computing [3], and much recent effort has been concentrated on providing middleware and software programming layers to facilitate grid computing. There are a number of projects such as Globus, Legion, Polder, and EcoGRID that deal with a variety of problems such as resource specification, information service, allocation and security issues in a grid computing environment involving different administrative domains. A crucial issue for the efficient deployment of distributed applications on the grid is that of scheduling [3].

Although not all applications are equally suitable to run in a computational grid, an ideal class of applications for grid is the bag-of-tasks application. This type of application is composed by independent tasks, which existing tasks waiting to be scheduled can be executed in any order and do not need inter-task communication (i.e. embarrassingly parallel) [4,5]. Note that there are many important bag-of-tasks applications, including data mining, massive searches (such as key breaking), parameter sweeps, Monte Carlo simulations [6], fractals calculations (such as Mandelbrot), and image manipulation applications (such as tomographic reconstruction [7]).

Despite the better suitability of bag-of-tasks applications for computational grids, ensuring that such applications will achieve good performance on

* Corresponding author.
E-mail addresses: clweng@sjtu.edu.cn (C. Weng), lu-xd@cs.sjtu.edu.cn (X. Lu).

0167-739X/$ – see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2003.10.004

computational grids is not a trivial task, and a number of issues make scheduling such applications challenging. Resources on the grid are typically shared so that the contention created by multiple applications results in dynamically fluctuating delays and qualities of service. Moreover, due to dynamic nature of grids, information about the whole system is typically changing with time, which further complicates scheduling. In addition, resources in a grid environment are heterogeneous and may not perform similarly for the same application.

Due to these difficulties, the scheduling of applications on computational grids has been the target of considerable efforts in recent years [6,7,10,14–20]. In this paper, we present a Qsufferage algorithm for scheduling bag-of-tasks applications that considers the location of each task's input data, while makespan and response ratio are chosen as metrics for performance evaluation.

This paper is organized as follows. In Section 2, we introduce the application mode and the grid environment. In Section 3, we describe the mapping strategy in grid environment. The Qsufferage heuristic is presented in Section 4. Simulation results are discussed in Section 5. In Section 6, a brief overview is given for the research efforts related to our work. At last, we conclude the paper in Section 7.

2. Assumption and problem statement

2.1. Grid model

We assume that the grid environment consists of *resource domains* that are individual and autonomous administrative domains, and contain certain number of computational hosts. A computational grid can be denoted by $\mathrm{RD} = \{\mathrm{rd}_1, \mathrm{rd}_2, \ldots, \mathrm{rd}_j, \ldots\}$, in which rd_j denotes the jth resource domain. In resource domain $\mathrm{rd}_j = \{\mathrm{rd}_{j,1}, \mathrm{rd}_{j,2}, \ldots, \mathrm{rd}_{j,k}, \ldots, s_j\}$, $\mathrm{rd}_{j,k}$ denotes the kth host and s_j denotes a data repository in it. Hosts inside one resource domain have different relative CPU speed corresponding to heterogeneity.

It is metascheduler that receives requirements of users in the grid environment and maps users' tasks into geographically distributed resources. We assume that there is one metascheduler in a certain resource domain that dedicates to schedule tasks of bag-of-tasks applications (illustrated in Fig. 1).

The topology of the network is shown in Fig. 1. Hosts inside the same resource domain are connected with LAN, which bandwidth is higher compared to the bandwidth of WAN through which different resource domains are connected. In addition, network bandwidth varies corresponding to realistic network. A task assigned on some host not only can access to the data repository located in the same resource domain, also can access to the remote data repositories.

2.2. Task model

For bag-of-tasks applications such as massive searches (such as key breaking), MCell simulations [6], fractals calculations (such as Mandelbrot), tomographic reconstructions [7], one application consists of many tasks among which existing tasks waiting to be scheduled are independent each other, do not need inter-task communication, which resembles the SPMD (Single Program Multiple Data) model. In addition, before execution tasks need an input data

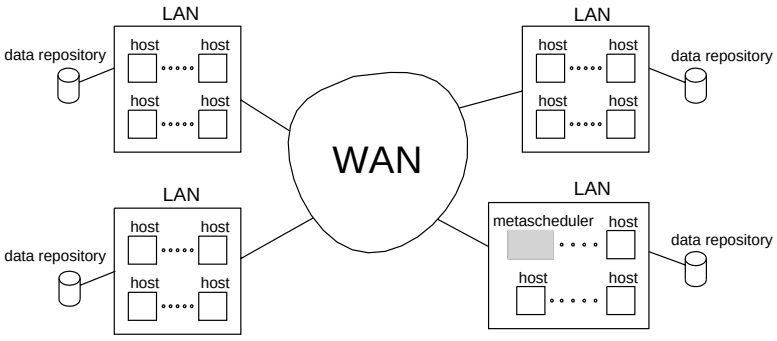

Fig. 1. Network topology.