Chapter 1.5
Resource Co-Allocation in Grid Computing Environments

Marco A. S. Netto
The University of Melbourne, Australia

Rajkumar Buyya
The University of Melbourne, Australia

ABSTRACT

One of the promises of Grid Computing is to enable the execution of applications across multiple sites. Several multi-site applications require simultaneous access to resources hosted on autonomous domains; this problem is known as resource co-allocation. Projects working on resource co-allocation face four major problems: distributed transactions, fault tolerance, inter-site network overhead, and schedule optimization. Although resource co-allocation is fundamental for Grid Computing, no survey has covered the current projects, solutions, and open challenges on this topic. Therefore, in this chapter, the authors describe the challenges on resource co-allocation, present the projects developed over the last decade, and classify them according to their similar characteristics. In addition, the authors discuss open research issues and trends such as negotiation, advance reservations, and rescheduling of multi-site applications.

INTRODUCTION

One of the promises of Grid Computing is to enable the execution of applications across multiple sites. Some of these applications require coordinated access to resources managed by autonomous entities. This coordinated access is known as *resource co-allocation.* There are two main classes of applications that require resource co-allocation: parallel applications with inter-process communication, and workflow applications. Parallel applications with inter-process communication require all resources to be available at the same time, whereas workflows constitute the execution of tasks with precedence constraints, i.e. resources have to be available in a certain order. Although both application classes require co-allocation, in the Grid computing community, the term *co-*

DOI: 10.4018/978-1-4666-0879-5.ch1.5

allocation usually refers to the *simultaneous access to resources hosted by autonomous providers* (Czajkowski, Foster, & Kesselman, 1999). The coordinated access to resources by tasks with precedence constraints is referred as *workflow scheduling* (Yu & Buyya, 2005). In this work, we follow the Grid computing community definition.

The two main reasons for executing applications on multiple sites are: (i) the lack of special resources in a single administrative domain, such as devices for generating data, visualization tools, and supercomputers; and (ii) reduce response time of parallel applications by increasing the number of resources (Czajkowski et al., 1998). However, there are other applications that require co-allocation. Conference and multimedia users engaged in activities, such as scientific research, education, commerce, and entertainment, require co-allocation of multiparty real-time communication channels (Ferrari, Gupta, & Ventre, 1997; Xu, Nahrstedt, & Wichadakul, 2001). Data-intensive applications use co-allocation to collect data from multiple sources in parallel (Vazhkudai, 2003; Yang, Yang, Wang, Hsu, & Li, 2007). In addition, increasing the number of resources is a requirement of large-scale applications demanding considerable amounts of memory, storage, and processing power. Examples of these applications are semiconductor processing (Takemiya et al., 2006) and computational fluid dynamics (Dong, Karniadakis, & Karonis, 2005).

Various projects have developed software systems with resource co-allocation support for large-scale computing environments, such as TeraGrid, Distributed ASCI Supercomputer (DAS), and Grid'5000. TeraGrid has deployed Generic Universal Remote (GUR) (Yoshimoto, Kovatch, & Andrews, 2005) and Highly-Available Resource Co-allocator (HARC) (Maclaren, Keown, & Pickles, 2006), the DAS project has developed KOALA (Mohamed & Epema, 2005), and Grid'5000[1] has relied on the OAR(Grid) scheduler (Capit et al., 2005) to allow the execution of applications requiring co-allocation. There are also projects dedicated to the management of network links, such as G-lambda (Takefusa et al., 2006).

Although resource co-allocation is fundamental for Grid Computing, no survey has covered the four major challenges in this field: distributed transactions, fault tolerance, inter-site network overhead, and schedule optimization. Therefore, in this chapter, we describe the challenges on resource co-allocation, present some of the efforts and projects developed over the last decade, and classify them according to their similar characteristics. In addition, we discuss open research issues and trends such as negotiation, advance reservations, and rescheduling of multi-site applications.

BACKGROUND

Existing work on resource co-allocation have focused on four research problems: distributed transactions, fault tolerance, inter-site network overhead, and schedule optimization. Most of the projects we present in this chapter have considered at least two of these problems. Resource co-allocation involves the interaction of multiple entities, namely clients and resource providers. Multiple clients may ask for resources at the same time from the same providers. This situation may generate deadlocks if the resource providers use a locking procedure; or livelock if there is a timeout associated with the locks. Therefore, there has been research on protocols to handle distributed transactions in order to avoid deadlocks and livelocks, and minimize the number of messages during these transactions.

Another common problem in the resource co-allocation field is that a failure in a single resource compromises the entire execution of an application that requires multiple resources at the same time. One approach to minimize this problem is defining a fault tolerance strategy that notifies applications of a problem with a resource.