Chapter 1
Data-Aware Distributed Computing

Esma Yildirim
State University of New York at Buffalo (SUNY), USA

Mehmet Balman
Lawrence Berkeley National Laboratory, USA

Tevfik Kosar
State University of New York at Buffalo (SUNY), USA

ABSTRACT

With the continuous increase in the data requirements of scientific and commercial applications, access to remote and distributed data has become a major bottleneck for end-to-end application performance. Traditional distributed computing systems closely couple data access and computation, and generally, data access is considered a side effect of computation. The limitations of traditional distributed computing systems and CPU-oriented scheduling and workflow management tools in managing complex data handling have motivated a newly emerging era: data-aware distributed computing. In this chapter, the authors elaborate on how the most crucial distributed computing components, such as scheduling, workflow management, and end-to-end throughput optimization, can become "data-aware." In this new computing paradigm, called data-aware distributed computing, data placement activities are represented as full-featured jobs in the end-to-end workflow, and they are queued, managed, scheduled, and optimized via a specialized data-aware scheduler. As part of this new paradigm, the authors present a set of tools for mitigating the data bottleneck in distributed computing systems, which consists of three main components: a data-aware scheduler, which provides capabilities such as planning, scheduling, resource reservation, job execution, and error recovery for data movement tasks; integration of these capabilities to the other layers in distributed computing, such as workflow planning; and further optimization of data movement tasks via dynamic tuning of underlying protocol transfer parameters.

DOI: 10.4018/978-1-61520-971-2.ch001

INTRODUCTION

Scientific applications and experiments are becoming increasingly complex and more demanding in terms of computational and data requirements. Large experiments, such as high-energy physics simulations (ATLAS, 2010; CMS, 2010), genome mapping (Altshul et al, 1990), and climate modeling (Kiehl et al, 1998) generate data volumes reaching hundreds of terabytes per year (Hey & Trefethen, 2003). Data collected from remote sensors and satellites, dynamic data-driven applications, digital libraries and preservations are also producing extremely large datasets for real-time or offline processing (Ceyhan & Kosar, 2007; Tummala & Kosar, 2007). To organize and analyze these data, scientists are turning to distributed resources owned by collaborating parties or national facilities to provide the computing power and storage capacity needed. But the use of distributed resources imposes new challenges (Kosar, 2006). Even simply sharing and disseminating subsets of the data to the scientists' home institutions is difficult and not yet routine — the systems managing these resources must provide robust scheduling and allocation of storage resources, as well as efficient and reliable management of data movement.

Although through the use of distributed resources the institutions and organizations gain access to the resources needed for their large-scale applications, complex middleware is required to orchestrate the use of these compute, storage, and network resources between collaborating parties, and to manage the end-to-end processing of data. The majority of existing research has been on the management of compute tasks and resources, as they are widely considered to be the most expensive. As scientific applications become more data intensive, however, the management of data resources and data flow between the storage and compute resources is becoming the main bottleneck. Many jobs executing in distributed environments are failed or are inhibited by overloaded storage servers, congested network links, or incomplete data transfers. These failures prevent scientists from making progress in their research.

According to the 'Strategic Plan for the US Climate Change Science Program (CCSP)', one of the main objectives of the future research programs should be *"Enhancing the data management infrastructure",* since *"The users should be able to focus their attention on the information content of the data, rather than how to discover, access, and use it."* (CCSP, 2003). This statement by CCSP summarizes the goal of many cyberinfrastructure efforts initiated by NSF, DOE and other federal agencies, as well the research direction of several leading academic institutions. This is also the main motivation for our work presented in this chapter.

Traditional distributed computing systems closely couple data handling and computation. They consider data resources as second class entities, and access to data as a side effect of computation. Data placement (i.e., access, retrieval, and/or movement of data) is either embedded in the computation and causes the computation to delay, or is performed by simple scripts which do not have the same privileges as compute jobs. The inadequacy of traditional distributed computing systems in dealing with complex data handling problems in our new data-rich world has motivated a new paradigm called *data-aware distributed computing* (Kosar et al, 2009).

We have previously introduced the concept that the data placement activities in a distributed computing environment need to be first class entities just like computational jobs, and presented the first batch scheduler specialized in data placement and data movement: Stork (Kosar & Livny, 2004). This scheduler implements techniques specific to queuing, scheduling, and optimization of data placement jobs, and provides a level of abstraction between the user applications and the underlying data transfer and storage resources. Stork is considered one of the very first examples of "data-aware scheduling" and has been very actively used in many e-Science application areas