INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING, *SERIES B*
Volume 5, Number 1-2, Pages 97–112

ROBINIA: SCALABLE FRAMEWORK FOR DATA-INTENSIVE SCIENTIFIC COMPUTING ON WIDE AREA NETWORK

YANG GU, GUOQING LI, QUAN ZOU, AND ZHENCHUN HUANG

Abstract. With the continuously growing data from scientific devices and models, data exploration becomes one of four kinds of scientific research paradigms. It leads to faster, larger-scale and more complex processing requirements, and parallelism is being more and more important for scientific data analyzing applications. But, because of troubles such as unstable wide-area network and heterogeneity among computing platforms, it is difficult to create scalable parallel scientific applications, especially wide-area parallel applications which have to process big data from geographically distributed research institutes to enable complex data analysis for "great challenge problems". In this paper, a data intensive computing framework named Robinia is proposed for exploiting parallelism among processing nodes over wide area network for data-intensive analysis on scientific big data. Robinia integrates distributed resources such as scientific data, processing algorithms, and storage services by a platform-independent framework; provides a unified execution environment for wide-area network based distributed spatial applications; and helps them exploit parallelism by a well-defined web-based programming interface. Experiments on prototype system and demo applications show that scientific analysis applications based on Robinia can achieve higher performance and better scalability by analyzing distributive stored big data over wide-area network such as Internet simultaneously.

Key words. parallel processing, wide area network, scientific computing, big data.

1. Introduction

In the last decades, more scientific devices are built and more scientific data are captured and stored. For example, devices such as the Square Kilometre Array of radio telescopes project, CERNs Large Hadron Collider, and astronomys Pan-STARRS array of celestial telescopes are capable of generating several petabytes (PB) of data per day, but present plans limit them to more manageable data collection rates. To enable the fourth paradigm for science based on data-intensive computing, tremendous capability and scalability are required for scientific data processing and visualization infrastructure to store, process, analyze and visualize the data so that information, knowledge and theories in the big data can be discovered and mined. In order to make the data processing faster, simpler and more scalable, parallelism is enabled for the scientific data analyzing applications, and one of the best ways to achieve scalable performance is exploiting parallelism of applications. It decomposes a data processing job on mass data into a lot of subtasks on distributed nodes, which will deal with a piece of data independently.

There are many ways for scientific applications to achieve parallelism. For example, High Performance Fortran (HPF) [1] and OpenMP [2] exploits parallelism by employing many processors or processor cores to process different parts of a single array, MPMD programming with MPI [3] extends the same idea to a distributed setting such as cluster, and a data-intensive distributed application may achieve it by running coarse-grain subtasks on geographically distributed computing nodes concurrently.

Received by the editors January 11, 2014 and, in revised form, March 21, 2014.
2000 *Mathematics Subject Classification.* 35R35, 49J40, 60G40.

Otherwise, scientific big data are often massive and geographically distributed among research institutes around the world. To analyze these data for some scientific results, applications must execute on heterogeneous computing nodes provided and managed by different agents. For example, a spatial application will query and analyze remote sensing data from several space agencies such as NASA, ESA and JAXA so that knowledge hidden in the big data set can be discovered. In this scenario, a coarse-grain parallel application can minimize the data transfer cost by scheduling subtasks to process data closer to where it is stored, and achieve better performance by allocating subtasks on more geographically distributed computing nodes. It is one of the best ways to achieve parallelism, especially on wide area networks (WAN) such as Internet.

But, it is challenging to create a scalable and efficient wide-area parallel application for scientific data processing, especially for beginners who have not much knowledge and experiment in parallelization for distributed context. Codes must be written carefully to decompose the processing job into subtasks, distribute data among nodes, transfer commands and messages through WAN, schedule subtasks for load balancing, monitor computing nodes, etc. So, programming models and frameworks which can help data-intensive application development and execution will be very valuable for scientific data processing.

Due to the lack of wide-area parallel processing framework for scientific data-intensive applications, we propose such a framework named Robinia, in order to enable parallel processing on wide area network for scientific data-intensive computing. First of all, distributed execution environment of Robinia exchanges commands and messages by standard protocols such as HTTP, so that firewalls can be passed through. Then, toolkits provided by Robinia such as dynamical node discovery, smart data distribution, description-based algorithm migration, and adaptive task scheduling, integrates increasing processing and storage resources all over the WAN together for distributed spatial applications. Furthermore, existing codes and algorithms for scientific data processing can be reused easily and deployed dynamically, and new codes and models can be developed simply by many programming languages and script languages. (e.g. Java, Beanshell, Groovy and Scala) As the result, much higher amount of scientific data can be processed by more computing nodes around the world without much performance loss and extra work when the problem size increases.

The paper is organized as follows. After the discussion about related work in section 2, section 3 proposes the architecture overview of Robinia, and designs components such as Executor, Engine, Node Discovery, Global Weather Focus and Web-based User Interaction for Robinia. Section 4 studies how to store mass data on a number of nodes by storage cluster, the atomic element for data storage in Robinia. Section 5 describes parallel processing implementation based on master-worker executors, and implements MapReduce model on Robinia as an example. Section 6 details the implementation such as event loop, distributed node discovery, and existing codes integration, and end-user interaction. Finally, section 7 discusses experiences about typical scientific data processing applications such as remote sensing parallel processing and distributed biological sequence comparing; and conclusions are summarized in section 8.