BIOINFORMATICS APPLICATIONS NOTE Vol. 23 no. 9 2007, pages 1175–1177
doi:10.1093/bioinformatics/btm086

Systems biology

ABCGrid: Application for Bioinformatics Computing Grid

Ying Sun[1], Shuqi Zhao[1], Huashan Yu[2], Ge Gao[1,*] and Jingchu Luo[1,*]

[1]Center for Bioinformatics, National Laboratory of Plant Genetic Engineering and Protein Engineering and College of Life Sciences and [2]School of Electronic Engineering and Computer Science, Peking University, Beijing 100871, P.R. China

Received on January 16, 2007; revised on February 23, 2007; accepted on February 28, 2007

Advance Access publication March 7, 2007

Associate Editor: Golan Yona

ABSTRACT

Summary: We have developed a package named Application for Bioinformatics Computing Grid (ABCGrid). ABCGrid was designed for biology laboratories to use heterogeneous computing resources and access bioinformatics applications from one master node. ABCGrid is very easy to install and maintain at the premise of robustness and high performance. We implement a mechanism to install and update all applications and databases in worker nodes automatically to reduce the workload of manual maintenance. We use a backup task method and self-adaptive job dispatch approach to improve performance. Currently, ABCGrid integrates NCBI_BLAST, Hmmpfam and CE, running on a number of computing platforms including UNIX/Linux, Windows and Mac OS X.

Availability: The source code, executables and documents can be downloaded from http://abcgrid.cbi.pku.edu.cn

Contact: abcgrid@mail.cbi.pku.edu.cn

1 INTRODUCTION

Massive computing resources are often required for large-scale bioinformatics analyses. Many bioinformatics tools such as BLAST (Altschul *et al.*, 1997) and HMMER (Eddy, 1998) require powerful computers for better performance. However, small or mid-scale biological laboratories cannot afford powerful servers or dedicated clusters. Instead, they usually have dozens of personal computers and workstations which are often relatively idle. Grid technology provides an alternative approach for these laboratories to utilize distributed idle resources to meet the needs of computational capability. Location transparency is another important feature of grid technology. Users can access applications without being aware of where these packages are installed.

A number of grid-enabled applications have been developed including BeoBlast (Grant *et al.*, 2002), Soap-HT-BLAST (Wang and Mu, 2003), mpiBLAST (Darling *et al.*, 2003), GridBLAST (Krishnan, 2005), W.ND BLAST (Dowd *et al.*, 2005) and Squid (Carvalho *et al.*, 2005). These applications have been proven useful. However, they are mainly designed to parallel the execution of BLAST rather than to support general-purpose bioinformatics applications. They also lack the ability to synchronize with continuously updated sequence databases automatically in worker nodes. Moreover, installation and management of most grid applications are rather complicated, depending on specific computational frameworks, e.g. the Globus Toolkit which is not very handy for most biology laboratories.

We developed a grid package named Application for Bioinformatics Computing Grid (ABCGrid). ABCGrid is easy to install, easy to use and easy to maintain, especially for small and mid-scale biology laboratories which lack professional computer system administrators. ABCGrid can seamlessly run on a number of heterogeneous computing platforms to provide users an integrated environment for accessing different bioinformatics applications. Currently, NCBI BLAST, Hmmpfam and CE (Shindyalov and Bourne, 1998) were implemented as ABCGrid plug-ins, and deployed in our local environment which consists of worker nodes with varying platforms, such as UNIX/Linux, Mac OS X and MS Windows.

2 MAIN FEATURES

ABCGrid is a Client/Server application package which consists of three independent programs: ABCMaster, ABCWorker and ABCUser. ABCUser accepts commands from a user and sends commands to ABCMaster. ABCMaster translates commands, spawns a job and decomposes the job into several tasks then dispatches tasks to ABCWorkers to process. After the task is completed, ABCWorkers send the results back to ABCMaster. If the result of the last task of one job is returned, ABCMaster will collect all results with the same job identity, sort them by their task identity and write the sorted data into the result file.

2.1 Easy to install and easy to maintain

Given that most biological laboratories lack professional computer system administrators, it is critical to make the software installation and management as easy as possible. ABCGrid does not require any pre-installed third-party software such as a complex grid tool-kit which is usually hard to install and configure. We provide universal binary distribution packages in ABCGrid for all popular platforms with Java Runtime Environment. To make the code work, users are only required to download and uncompress the package.

*To whom correspondence should be addressed.

A big challenge for all bioinformatics computing grid implementations is to keep the data and applications consistent among all worker nodes, which is time-consuming and error-prone if it is carried out manually. To solve this problem, we implemented an online updating mechanism in ABCGrid to install and update databases and applications in all worker nodes automatically. A configuration file on the master node is used to describe all available data sources and supported services. Each ABCWorker compares the local configuration file with that of ABCMaster when it connects to, or receives an update command from ABCMaster. If it is necessary, ABCWorker will connect to specified data sources and download all updated files automatically. This approach greatly reduces the workload of manual maintenance. Moreover, to facilitate the user management and security, ABCGrid also supports user authentication and user priority by assigning different priorities to different users.

Adding support to new bioinformatics applications in ABCGrid is easy for developers. Only one parser class inherited from a public interface of ABCMaster is needed. The parser class is used to parse the command, then spawn the job and decompose it into tasks. No modifications are required for ABCWorker and ABCuser. A developer guide can be found at the ABCGrid website with an example showing how to write a plug-in step by step.

2.2 Fault tolerance

Since one job is decomposed into tasks and tasks are dispatched to several worker nodes, a submitted job will not be completed until all tasks are completed. However, unexpected exceptions such as abnormal termination of a task execution, hardware failure of worker nodes, or breakage of network connection may occur from time to time in a distributed computing environment. We use a method called backup task inspired by Google's MapReduce Model (Dean and Ghemawat, 2004) to solve this problem and improve the performance. The primary idea of backup task is that if one worker node cannot finish a task on time, other worker nodes will take over and process that task. A task will be marked as 'in-progress' when it is taken and processed by a worker node. When this task is completed and the result is returned, it will be marked as 'complete'. This execution is called primary execution. If one worker node cannnot complete a task within an unusual long time due to various reasons such as low performance or machine failure, other idle worker nodes will take the same task to execute. The task completes whenever either the primary or the backup execution completes. When a job is completed, all backup executions will be terminated immediately to reduce unnecessary computations.

2.3 High performance

A grid system should dispatch appropriate workload to worker nodes with various computing resources to reach high performance. A common dispatching method used by many grid implementations is that the master actively sends tasks to the workers. However, different applications may require different computing resources such as fast CPU or large memory or fast disk I/O. It is a challenge to determine appropriate workload in a heterogeneous and distributed computing environment in practical bioinformatics applications. Inappropriate workload dispatching will lead to prolonged running time, hence, waste some computing resources. We implemented a self-adaptive job dispatch approach in ABCGrid to solve this problem. When a new job is submitted, ABCMaster decomposes the job into several tasks and notifies all connected ABCWorkers. Each awaked ABCWorker fetches one task to process. It will not fetch the next task until the current task is successfully completed. This approach overcomes the complicated load balance problem, since each worker node processes tasks 'as best as it can' within its ability. Other methods are also implemented to improve performance. For example, to save bandwidth, data can be compressed before transferring across the network.

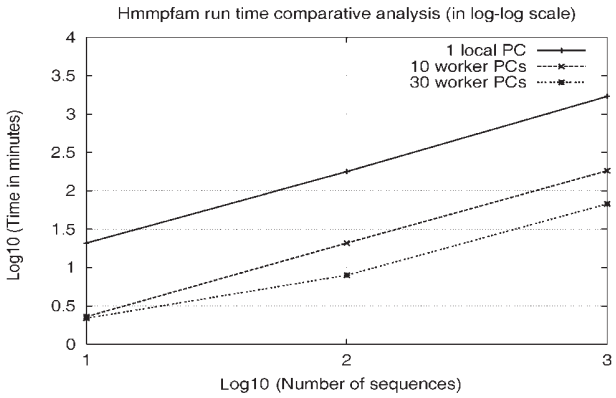

Fig. 1. A test case to show the performance of ABCGrid. The *X*-axis denotes the three datasets with 10, 100 and 1000 sequences randomly retrieved from PDB. The *Y*-axis shows the running time using one local PC (solid line), 10 (dash line) and 30 (dot line) work nodes in ABCGrid.

3 EVALUATION

To evaluate the performance of ABCGrid, we made two test cases using Hmmpfam and Blastp as demo applications. The test environment consists of 30 worker nodes connected with a local area network. Each node has a single CPU (AMD Sempron 2200+), 512MB RAM, 80 GB IDE Disk, running Gentoo-Linux. The master node has the same hardware configuration running Windows XP. Three FASTA files with 10, 100 and 1000 amino acid sequences, respectively, were randomly retrieved from the PDB database. An Hmmpfam search against the PFAM database was performed on 1 machine using local Hmmpfam, as well as 10 and 30 worker nodes using ABCGrid, respectively. The performance increased approximately linearly when the number of worker nodes increases (Fig. 1). A second test also showed a similar linear relationship using Blastp to search the NCBI non-redundant database with the same input sequence files.

ACKNOWLEDGEMENTS

We thank X Wu, L Kong and WZ Zhang for helpful discussions. This study was supported by grants of the

863 Program (2006AA02Z334) and the High-Tech Platform Program of China.

Conflict of Interest: none declared.

REFERENCES

Altschul,S.F. *et al.* (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, **25**, 3389–3402.

Carvalho,P.C. *et al.* (2005) Squid - a simple bioinformatics grid. *BMC Bioinformatics*, **6**, 197.

Darling,A., *et al.* (2003) The Design, Implementation, and Evaluation of mpiBLAST. 4th International Conference on Linux Clusters. San Jose, CA, June 2003.

Dean,J. and Ghemawat,S. (2004) MapReduce: Simplified Data Processing on Large Clusters. OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December, 2004.

Dowd,S.E. *et al.* (2005) Windows. NET Network Distributed Basic Local Alignment Search Toolkit (W.ND-BLAST),. *BMC Bioinformatics*, **6**, 93.

Eddy,S.R. (1998) Profile hidden Markov models,. *Bioinformatics*, **14**, 755–763.

Grant,J.D. *et al.* (2002) BeoBLAST: distributed BLAST and PSI-BLAST on a Beowulf cluster. *Bioinformatics*, **18**, 765–766.

Krishnan,A. (2005) GridBLAST: a Globus-based high-throughput implementation of BLAST in a Grid computing framework. *Concurrency and Computation: Practice and Experience*, **17**, 1607–1623.

Shindyalov,I.N. and Bourne,P.E. (1998) Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng*, **11**, 739–747.

Wang,J. and Mu,Q. (2003) Soap-HT-BLAST: high throughput BLAST based on Web services. *Bioinformatics*, **19**, 1863–1864.