InterCondor: A Prototype High Throughput Computing Middleware for Geocomputation

Yong Xue^{1,2,*}, Yanguang Wang¹, Ying Luo^{1,4}, Jianping Guo¹, Jianqin Wang³,
Yincui Hu¹, and Chaolin Wu¹

¹ State Key Laboratory of Remote Sensing Science, Jointly Sponsored by the Institute of Remote Sensing Applications of Chinese Academy of Sciences and Beijing Normal University, Institute of Remote Sensing Applications, CAS, P.O.Box 9718, Beijing 100101, China

² Department of Computing, London Metropolitan University,
166-220 Holloway Road, London N7 8DB, UK

³ College of Information and EEng., China Agricultural University, Beijing 100083, China

⁴ Graduate University of Chinese Academy of Science,
Yuquan Road, Beijing 100049, China
y.xue@londonmet.ac.uk

Abstract. This paper presents the design, analysis and implementation of InterCondor system. The InterCondor system is an implementation of the concept of InterGrid. It uses Condor as a basic local Grid computing engine. It utilizes a series of Grid services, which including register service, data transfer service, task schedule service, security authentication service and status monitor service, to manage the resources such as remote sensing algorithms, remote sensing data, and computing resource under the management of Condor engine. We aim at integrating Grid Service data management, task schedule, and the computing power of Condor into remote sensing data processing and analysis to reduce the processing time of a huge amount of data and long-processing-time remote sensing task by algorithms issuance, data division, and the utilization of any computing resources unused on Internet.

1 Introduction

The InterCondor system is an implementation of the concept of InterGrid. InterGrid [5][7] is from the comparison of Internet and Grid. With the reference to the idea of Internet, which integrates different local networks into a global network with a certain protocol, InterGrid utilizes a set of Grid services to manage resources such as data, algorithms, and local Grid pools. There are several projects on how to implement an InterGrid. E.g. IVDGL work group devoted to study on the interoperability of US-ATLAS, EDG and NorduGrid[6]. InterCondor is a Grid system that uses Condor as local Grids.

According to the three points checklist of what is the Grid[4], it is no matter how big a Grid is. The key characters of Grid are the coordination of resources, standard protocols and interfaces, and the nontrivial qualities of service. E.g. the famous Cactus project that won 2001 years Gordon Bell Prizes integrated four supercomputers to

* Corresponding author.

solve Grand Challenge problems in physics which require substantially more resources than can be provided by a single machine (<http://www.cactuscode.org>). There could be office Grid, enterprise Grid, certain functional Grids, etc. We can call them local Grid pools. Correspondingly, there is a global Grid, whose components may be local Grid pools. InterGrid is one kind of global Grid, which has at least two local Grids in it.

Although Globus seems to be a defacto standard of Grid, there is no real standard protocol for Grid up to now. There are many different Grids implemented by different technologies, such as Condor, LSF (<http://www.platform.com/products/LSF/>), or SRB system (<http://www.sdsc.edu/srb/>). Condor, LSF, and SRB system are independent Grid systems. It could be a problem to interconnect them into a global Grid. For example, although Condor has the glide mechanism to communicate among different Condor pools, it cannot go through firewalls and its protocol is private (<http://www.cs.wisc.edu/condor/glidein/>).

The Condor Project has performed research in distributed high-throughput computing for the past 18 years, and maintains the Condor High Throughput Computing resource and job management software originally designed to harness idle CPU cycles on heterogeneous pool of computers[1]. In essence a workload management system for compute intensive jobs, it provides means for users to submit jobs to a local scheduler and manage the remote execution of these jobs on suitably selected resources in a pool. Boasting features such as check-pointing (state of a remote job is regularly saved on the client machine), file transfer and I/O redirection (i.e. remote system calls performed by the job can be captured and performed on the client machine, hence ensuring that there is no need for a shared file system), and fair share priority management (users are guaranteed a fair share of the resources according to pre-assigned priorities), Condor proves to be a very complete and sophisticated package. While providing functionality similar to that of any traditional batch queuing system, Condor's architecture allows it to succeed in areas where traditional scheduling systems fail. As a result, Condor can be used to combine seamlessly all the computational power in a community.

In this paper, we present a Grid system (InterCondor system) that interconnects Condor pools as local Grids so that Condor can go through firework and integrate with other Grid systems. It is developed in Institute of Remote Sensing Applications, Chinese Academy of Sciences, China. We aim at integrating Grid Service data management, task schedule, and the computing power of Condor into geocomputation to reduce the processing time of a huge amount of data and long-processing-time remote sensing task by algorithms issuance, data division, and the utilization of any computing resources unused on Internet. The design of InterCondor system is introduced in details in Section 2. The implementation and analysis of it will be demonstrated in Section 3. Finally, the conclusion and further development will be addressed in Section 4.

2 Design of InterCondor

Grid and Internet are both from bottom to top[8]. In fact Grid is a series of standards and protocols in the application layer of Internet. But its inner hierarchy is highly

similar with that of Internet. As to design InterCondor system, we use the idea to divide and rule it too. Figure 1 shows the layer architecture of InterCondor.

InterCondor utilizes a series of Grid services, which including register service, data transfer service, task schedule service, security authentication service and status monitor service, to manage the resources such as remote sensing algorithms, remote sensing data, and computing resource under the management of Condor engine.

Users choose services and submit tasks to InterCondor through the user interface. If InterCondor gets a task from the users, it will trigger the IAST (Intelligent Analyzer & Synthesizer of Tasks). Then the IAST will divide the task into several CPE (Condor Pool Entry) tasks according to the status of the Condor pools currently in InterCondor, and submit the CPE tasks to the Grid Service layer. The Grid Service layer is the key of InterCondor. It stands for dividing the CPE tasks into Condor tasks, submitting the Condor tasks to the right machines which satisfied the computing conditions required by applications, merging the computing results of Condor into CPE results, and return the CPE results to IAST. As a computing engine, Condor stands for no upper management. Finally, the IAST merges the CPE results returned by Grid service layer into a final result corresponding to the division strategy, and returns the final result to user.

Fig. 1. InterCondor layer architecture

2.1 User Interface Layer

The user interface includes a user graphical interface, a console interface, and an API to advance users such as developers. There is no need for users to know the whole system. The user interface makes the users operations transparently. The InterCondor software is ostensibly no difference with traditional software. To achieve this, we need an Intelligent Analyzer & Synthesizer of Tasks (IAST).

2.2 Client Layer

A client is mainly an IAST. An IAST stands for shield the difference between the network attribute of InterCondor and software running on single machine. Figure 2 shows the IAST work flow chart.

2.3 Grid Service Layer

Grid Service layer stands for system management, tasks schedule, and data transfer. It's the key layer of InterCondor. The architecture of InterCondor is clear that grid service stands for management and Condor stands for computing.

Grid service layer includes many kinds of services. They are registration service, status service, transferring service, invoking service stands for receiving the call from users, security authentication service and other special services.

Fig. 2. IAST work flow chart

2.4 Condor Layer

The Condor layer is composed of many machines managed by CPEs in the InterCondor system. As a computing engine, it's no need for the Condor layer to care about the external architecture or status. The only things which need Condor layer to do are: execute the tasks submitted by CPEs, respond to the call to inquire the status of Condor pool and report errors occurred during the execution of the tasks to CPEs. These functions can be accomplished by configuring Condor.

2.5 Resource Layer

The resource layer is a necessary and rock-bottom entity of the InterCondor system. It comes down to the resources of data, algorithm modules and computing power. It's not the focus of this paper.

3 Implementation of InterCondor Version 1.0

As shown in Figure 3, we use a peer-to-peer structure to implement the design mentioned in Section 2. The management of the InterCondor system is in different levels: the InterCondor server doesn't manage each machine in the system directly. It manages the Condor pools instead, and the CPEs manage the machines in the Condor pools. CPE is the most important element in InterCondor system. It has the most functional modules. The InterCondor server has only the registration service on it. Actually it does not manage the whole system. Instead, it only collects the information about the whole system and provides the information to CPEs, so that CPEs can know which CPEs to be cooperate with. It's up to the CPEs to receiving tasks submitted by users, execute the first level division, push CPE tasks to other CPEs, execute the second level division, submit the Condor task to Condor pools, merge the results, and return the final results to users. Each CPE is both a client and a server. Each CPE can start a InterCondor task and also can cooperate on a task which is started by other CPEs.

Up to now InterCondor version 1.0 accomplished using Java. The user interface of InterCondor should shield the Grid computing character of InterCondor in order to make the users feel like the way they are using the traditional remote sensing software that they are familiar to. The Java GUI developed by the Java toolkits of AWT and

SWING means a slow run speed and strange interface. But using Standard Widget Toolkit (SWT) developed by Eclipse organization, people can develop high effect GUI that has standard appearance. Furthermore, the GUI developed by SWT has no matter with platforms.

We use Web service technology such as SOAP, XML, and WSDL to implement the system. Globus is a new technology which progressing fast. From the issuance of Globus toolkits version 1.0 at 1999 to version 4.0 at 2005, the protocol and technology in it changed evidently. The reliability and practicality need textual research. Therefore, it's venturesome to follow it. Web service is a mature technology, which has been tested out in the field of industry for decades of years. Globus is adjusting its steps to web service. So long as our node supports web service, our node will be compatible with Globus. Web service ensures the compatibility and expansibility of our node. It's more complex and difficult to develop using Globus than using web service. There're some convenient tools available for web service. Figure 4 shows the relationship of different kinds of tasks.

3.1 Condor Pool Entry (CPE)

CPE stands for the connection of Condor pool and InterCondor server. It acts as a middleware between Grid services and Condor commands from the point of view of software. The development work of InterCondor mostly aims at CPE. It's up to CPE

Fig. 3. Peer-to-peer structure of InterCondor version 1.0

Fig. 4. The relationship of different kinds of tasks

to accept the commands from clients, turn client's commands to DOS commands which Condor can understand, submit tasks to Condor, manage Condor pool, and return the results. The most of the InterCondor toolkit is installed on CPEs, which including the user interface, the client, and the Grid services. The Grid services currently include status service, transferring service, invoking service and security authentication service.

There are three kinds of machines in a Condor pool: submit machine, manager machine, and execute machine. The manager machine is the manager defined by Condor software itself. It can schedule, monitor, manage, submit, and execute tasks in the scale of a Condor pool. The submit machine stands for submitting tasks and inquiring status of tasks. It can submit and execute tasks. The execute machine only can execute tasks. Only the manager machine or submit machine can be a CPE.

3.2 Module

Module is a collection of executable programs and correlative files owned by users. The executable programs are algorithm codes developed by users to solve certain problems such as NDVI, road extraction, etc. If the users are volunteers to allow other InterCondor users to use their algorithms, they can issue their algorithms to InterCondor. There's no need for the providers to open their source codes or details of their algorithms. They can just provide the executable programs. Each module has a ID to identify itself and some description files to present its requirement to hardware and software, its function, its parameters, its command format, etc.

There is a module manager in each CPE. It stands for the query, add, delete, and update operation to modules. Implementing InterCondor1.0, the default module DB is \$INTERCONDOR_HOME/depot/code. All the information about modules is saved in XML files named Modules.xml (Figure 5). Each module has a corresponding configuration file named module.xml. Only the module manager can operate on modules. Only the module can operate on its module.xml.

Analogously there is a task manager on CPE. It stands for tasks receive, initialization, delete, trigger, stop, division, mergence, and status query. Only the task manager has the right to maintain the configure files (task.xml). Figure 6 presents the relationship between module manager and task manager.

3.3 PUSH Technology

Transferring service is a necessary infrastructure in InterCondor. It must satisfy the following requirements: Active; Can go through firewalls; Safe and reliable and Fast speed.

The nowadays' general transfer protocols are FTP and HTTP. But neither FTP nor HTTP is a passive method for the server. It's up to the client what should be transferred from server to client. But in the InterCondor system, the server makes decision what to be transferred to client. It must in an active way that we called it PUSH. It's for the server (a CPE which start an instance) to make decision which client (other CPEs) to cooperate with according to the inquire result from InterCondor server. Then the server CPE in this instance will divide the task into several CPE tasks, and push each CPE task to each cooperative CPE. Receiving a CPE task, each CPE will divide

it into several Condor tasks, and submit them to Condor. When the CPE task finished, each CPE will push its result to the server CPE in this instance. Finally the server CPE integrates these results and returns the final result to the user.

FTP is prohibited by many firewalls. GridFTP of Globus has added security, parallel transfer, and check-point functions in general FTP. But GridFTP cannot go through many firewalls either. While firewalls won't prohibit HTTP, and HTTP1.1

has the parallel transfer and check-point functions provided by GridFTP, we choose to improve HTTP1.1 to implement our InterCondor system. The core idea to use HTTP download technology to implement our PUSH technology is configure a Grid service on each CPE. When the server CPE wants to transfer something to cooperative CPEs, it will call the Grid service on CPEs. The information about what the server CPE wants to transfer to the cooperative CPE is in the call command. Receiving the call command, the cooperative CPE will analysis it so that the cooperative CPE should know what to be downloaded from the server CPE. And the following things are accomplished by general HTTP download technology.

HTTP is not fast and safe in some cases. So we use some other technology to make up for these. To obtain faster transfer speed, we use:

- Data compression technology
- HTTP multi-thread download technology
- Break point resuming technology

To make the transfer safer, we use the CRC32 accurate checkout codes technology. If the checkout codes wrong, the data should be transferred again.

4 Conclusions and Future Work

In this paper, we introduce our ongoing research on the InterCondor system. It has a peer-to-peer architecture, so that in case that some CPE fails its function the whole InterCondor system also can work. There's no exclusive entry to InterCondor. Each CPE can be an entry. And each CPE can drive other CPEs to cooperate on one task.

Implementing InterCondor version 1.0 we mainly use web service and Java technology. We use Condor as a computing engine. And we developed a PUSH technology

Fig. 5. Module and module manager

Fig. 6. Relations of module and task

to transfer data actively. The function of the InterCondor system like a broker which pushes machines on the Internet as more as possible to cooperate on one task by dividing the task into sub-tasks in different level. We try to make a large virtual supercomputer by integrating cheap personal computers on Internet, and utilize them when they are left unused. Theoretically, InterCondor can provide enough processing ability to anyone at anytime and anywhere if there are enough PCs left unused in it. The InterCondor system seems to work well, but actually it is far away from intactness. There are still many problems to deal with.

One problem is lacking of management to physically distributed data. Remote sensing application is a data- dense application. The long file transfer time between the PCs physically distributed weakens the decrease of the total processing time improved by using more PCs doing one task. Although the data compression technology and multi-thread technology are used, in some case using InterCondor cannot decrease the total processing time of a task.

The up to now version of InterCondor only support Java programs, IDL programs, and programs that can run on the Windows operation system. And the security mechanism of it is SimpleCA. More supports should be developed in the future.

Acknowledgement. This publication is an output from the research projects “Multi-scale Aerosol Optical Thickness Quantitative Retrieval from Remotely Sensing Data at Urban Area” (40671142), “Grid platform based aerosol monitoring modelling using MODIS data and middlewares development” (40471091) funded by NSFC, China, “Multi-sources quantitative remote sensing retrieval and fusion” (KZCX2-YW-313)” funded by CAS, and “973 Project - Active and passive remote sensing of land surface ecological and environmental parameters” (2007CB714407) by MOST, China and “Research Fund for Talent Program” funded by China Agricultural University.

References

1. Basney, J., Livny, M., Tannenbaum, T.: High Throughput Computing with Condor. HPCU news 1(2) (1997)
2. Cactus project, <http://www.cactuscode.org>
3. Condor glide mechanism, <http://www.cs.wisc.edu/condor/glidein/>
4. Foster, I.: What is the Grid? A Three Point Checklist (2002), <http://www.globus.org>
5. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications* 15(3), 200–222 (2001)
6. Gieraltowski, J.: US-ATLAS, EDG, NorduGrid Interoperability: A Focus on ATLAS and GRAPPA (2002), <http://www.hep.anl.gov/gfg/us-edg-interconnect/jerryg-intergrid-comparison.ppt>
7. Perez, J.M., Carretero, J., Garcia, J.D., Sanchez, L.M.: Grid data access architecture based on application I/O phases and I/O communities. In: PDPTA 2004: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, Las Vegas, NV, USA, vol. 1–3, pp. 568–574. CSREA PRESS (2004)
8. Segal, B.: Grid Computing: The European Data Project. In: IEEE Nuclear Science Symposium and Medical Imaging Conference, Lyon, pp. 15–20 (October 2000)