

Developing Lightweight Application Execution Mechanisms in Grids

Ligang He, Martin Dove, Mark Hayes, Peter Murray-Rust, Mark Calleja, Xiaoyu Yang, Victor Milman*

University of Cambridge, Cambridge, UK

*Accelrys Inc., Cambridge, UK

lh340@cam.ac.uk

Abstract

CASTEP is an application which uses the density functional theory to calculate atomistic structure and physical properties of materials and molecules. This paper investigates the execution mechanisms for running CASTEP applications using grid resources. First, a lightweight execution environment is developed for CASTEP, so that the CASTEP applications can be run on any grid computing resource with the supported platform even without the CASTEP server being installed. Then, two execution mechanisms are developed and implemented in this paper. One is based on Condor's file transfer mechanism, and the other based on independent data servers. Both mechanisms are complete with the GridSAM job submission and monitoring service. Finally, the performance of these mechanisms is evaluated by deploying them on CamGrid, a university-wide condor-based grid system at the University of Cambridge. The experimental results suggest that the execution mechanism based on data servers is able to deliver higher performance; moreover, if multiple data replicas are provided, even higher performance can be achieved. Although these mechanisms are motivated by running CASTEP applications in grids, they also offer excellent opportunities to be applied to other e-science applications as long as the corresponding lightweight execution environments can be developed.

1. Introduction

Grid systems are becoming a popular platform for processing e-science applications [2][3]. CASTEP is a software package used for atomistic quantum-mechanical modeling of materials and molecules [7]. In order to run CASTEP applications, the CASTEP server is required to provide the necessary execution environment.

Currently, a popular approach to running CASTEP applications is to employ the Materials Studio Modeling from Accelrys Inc [12]. Materials Studio Modeling 4.0 is a Client/Server software environment [12]. At the client side, the users can create models of molecular structures and materials. The model information is transferred to the

server side, where the calculations are performed and the results are returned to the users. The CASTEP server component is included in the server side. In the Grid environments, however, it cannot be assumed that Materials Studio is installed on all computational resources. This presents a challenge of harvesting the computational grid resources that have no pre-installed CASTEP server environments. This paper tackles this problem by developing a lightweight execution environment for CASTEP applications. The lightweight environment includes the CASTEP executables, shared libraries, relevant configuration files, license checking-out executables and license data, etc.

The execution mechanisms developed in this paper are implemented and deployed on CamGrid, a university-wide grid system at the University of Cambridge [1]. CamGrid is based on the Condor system [8], in which a federated service is provided to flock the condor pools located in different departments. The jobs submitted to a local Condor pool may be dispatched to another flocked remote pool depending on the resource availability [9].

GridSAM is a job submission and monitoring web service [6]. One of advantages of GridSAM is that the GridSAM client and server communicate with each other through the standard SOAP protocol. Moreover, GridSAM can be connected to different distributed resource managers such as Condor [4]. In this paper, GridSAM is deployed in the CamGrid and configured to distribute the jobs for execution through Condor. In the configuration GridSAM web service is available to the Internet, and the users can submit jobs over the Internet to the GridSAM service, where the jobs are further submitted by GridSAM to a condor job submission host. Then the job submission host further schedules the jobs to the remote execute machines in CamGrid.

This paper investigates two approaches to transferring a lightweight CASTEP execution environment. One approach is to make use of the Condor's file transfer mechanism. In the Condor submission description file, it can be specified what files need to be transferred to the execute nodes. The job submission host will copy these files over once the execute nodes have been determined.

Another approach is to setup independent data servers for a lightweight CASTEP execution environment. When a job has been launched on execute nodes, it first stages in the necessary CASTEP execution environments and then starts running.

These CASTEP execution mechanisms are presented in detail in this paper. Their performance is evaluated on CamGrid in terms of various metrics, including makespan, average turnaround time, overhead and speedup. The performance is also compared against that achieved by the Materials Studio mechanism.

The work developed in this paper is motivated by the MaterialsGrid project being conducted at the University of Cambridge [13]. The MaterialsGrid project aims to create a dynamic database of materials properties based on quantum mechanical simulations run in grid environments. CASTEP is a typical example of a computational application involved in this project.

2. Lightweight Execution Environment

After installing Materials Studio Modeling 4.0 on a machine, the CASTEP execution environment is built. By tracing the calling path of the CASTEP executables, the necessary files and settings for running CASTEP applications are identified in this paper.

All files required to run a CASTEP application can be divided into the following three categories.

1. User-defined files: cell file and parameter file

CASTEP application needs to take as input the information about atomistic system users want to study. The information is summarised in two text files: *model.cell* and *model.param* (cell and param are the file extensions). The cell file defines the molecular system of interest, and the parameter file (.param) defines the type of calculation to be performed. These two files can be generated using Materials Studio Modeling 4.0 [12] or by the CML transformation toolkit [11][10].

2. Application-dependent system files: potential files

When calculating the property of a given model, the pseudopotential files corresponding to the elements composing the structure will be used. A pseudopotential file for an element describes the screened electrostatic interaction between valence electrons and ionic core for that element. The pseudopotential files for all elements in the periodic table are provided in the Materials Studio Modeling package.

3. Application-independent system files:

The files in this category are required by all calculations. These include:

- CASTEP executables and shell scripts – CASTEP executables are the programs for performing the actual calculations. These executables are wrapped in shell scripts, in which the environmental variables as

well as other system configurations are defined and different CASTEP executables are invoked in different scenarios.

- Shared libraries – these libraries are required at run time by the CASTEP executables.
- License checking-out executables and relevant license data – the commercial version of Materials Studio Modeling 4.0 needs to check out the licence from the pre-installed license server before being able to run the CASTEP executables. These license-relevant files are included in a license pack, which is located in a directory separated from the Material Studio package. (Please note that a simpler license mechanism is applied to academic versions of CASTEP for the UK scientists; in this case the CASTEP executables do not need to check out the license files).

When installing the Materials Studio and the license Pack, the system configurations need be specified, such as the home directory of the installation, the location of the license server and license files. Under the Condor job manager, however, all files are placed in a single temporary working directory. Therefore, in order to successfully access the files in a execute node, these configurations need to be changed to adapt to the Condor working location.

After all necessary changes are made, the required files for running CASTEP applications are packed into tar files. The lightweight execution environment can be constructed in a execute node by unpacking these tar files. The total size of the Materials Studio and the License Pack package is 884 Megabytes. The size of the tar files for the lightweight CASTEP execution environment is reduced to 105 Megabytes.

3. Execution Mechanisms in Grids

In this section, different CASTEP execution mechanisms are presented in detail. The execution mechanisms differentiate from each other mainly by their approaches to dealing with the transferring of the CASTEP lightweight environment.

3.1 Execution Mechanism Based on Condor's File Transfer Mechanism

When connecting GridSAM with the Condor job manager, two major configuration files need to be edited. One is the *jobmanager.xml* file, which specifies the job manager and its settings used to distribute the jobs received by GridSAM. The other is the file *classad.groovy*, in which the values of the ClassAd attributes of a Condor job can be specified.

In the *classad.groovy* file, the transferred input files are specified, which include the user submitted cell file

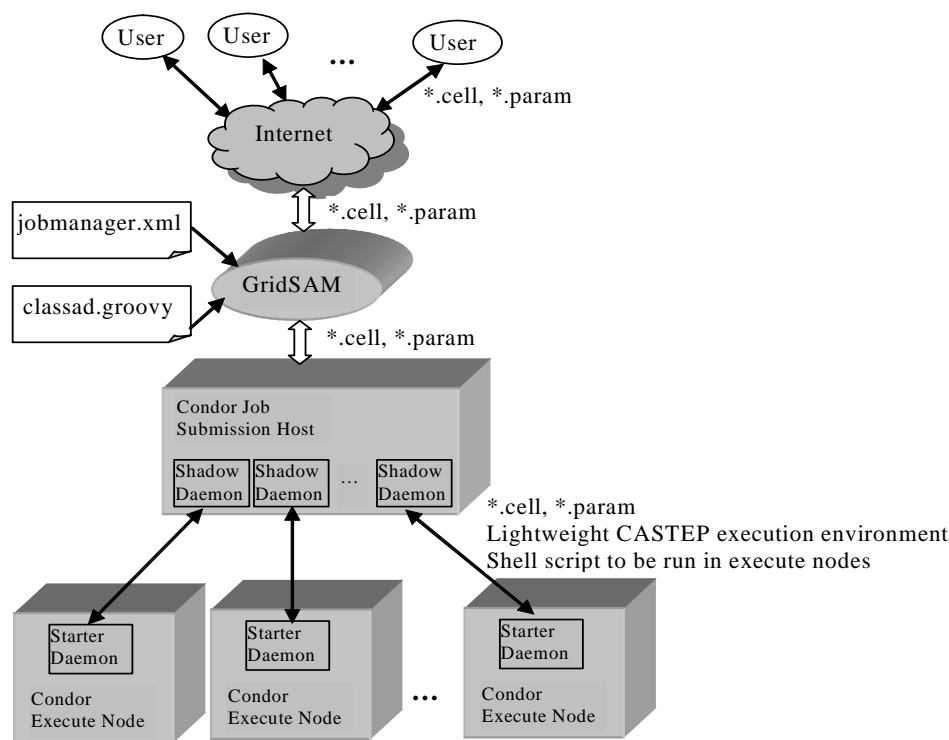


Figure 1. Application execution mechanism based on Condor's file transfer mechanism

```

1. FILE1='MaterialsStudio.tar'
2. FILE2='LicensePack.tar'
3. FILE3='Al_00.usp'
4. FILE4='O_00.usp'
5. flag=`expr $RANDOM % 2`
6. if [ $flag -lt 1 ]
7. then
8.     HOST=$HOST1
9.     PORT=$PORT1
10. else
11.     HOST=$HOST2
12.     PORT=$PORT2
13. fi
14. ftp -n $HOST $PORT << END_SCRIPT
15.     quote USER $USER
16.     quote PASS $PASSWD
17.     get $FILE1
18.     get $FILE2
19.     get $FILE3
20.     get $FILE4
21.     quit
22. END_SCRIPT
23. tar -xf $FILE1
24. tar -xf $FILE2
25. Call the script RunCASTEP.sh, which is included in
    the MaterialsStudio.tar, to calculate the specified
    model, Al2O3
    
```

Figure 2. Execution flow of the data server-based execution mechanism in an execution node

and the parameter file as well as the tar files for lightweight CASTEP execution environment. This classad.groovy file is used by GridSAM to construct the job submission description file for Condor. When the constructed Condor submission description file is submitted in the job submission host (where the tar files for the lightweight CASTEP execution environment are also stored), the Sched daemon sends the job advert to matchmaker and then the matchmaker looks for the execute node whose resource advert can match the job advert. After the suitable node is found, the Sched daemon starts a Shadow daemon acting on behalf of the submitted job, while the Start daemon at the execute node spawns a starter daemon to represent the process which performs the actual calculation. After the communication between the Shadow daemon and Starter daemon has been established, the specified input files are transferred from the job submission host to the execute node. The steps of constructing the CASTEP environment (i.e., unpacking the transferred tar files) and performing the actual CASTEP calculations are wrapped in a shell script, which is the executable sent to the execute nodes.

This execution mechanism is illustrated in Fig.1.

3.2 Execution mechanism based on independent data servers

In this mechanism, the independent data servers are set up to accommodate the tar files for the lightweight CASTEP

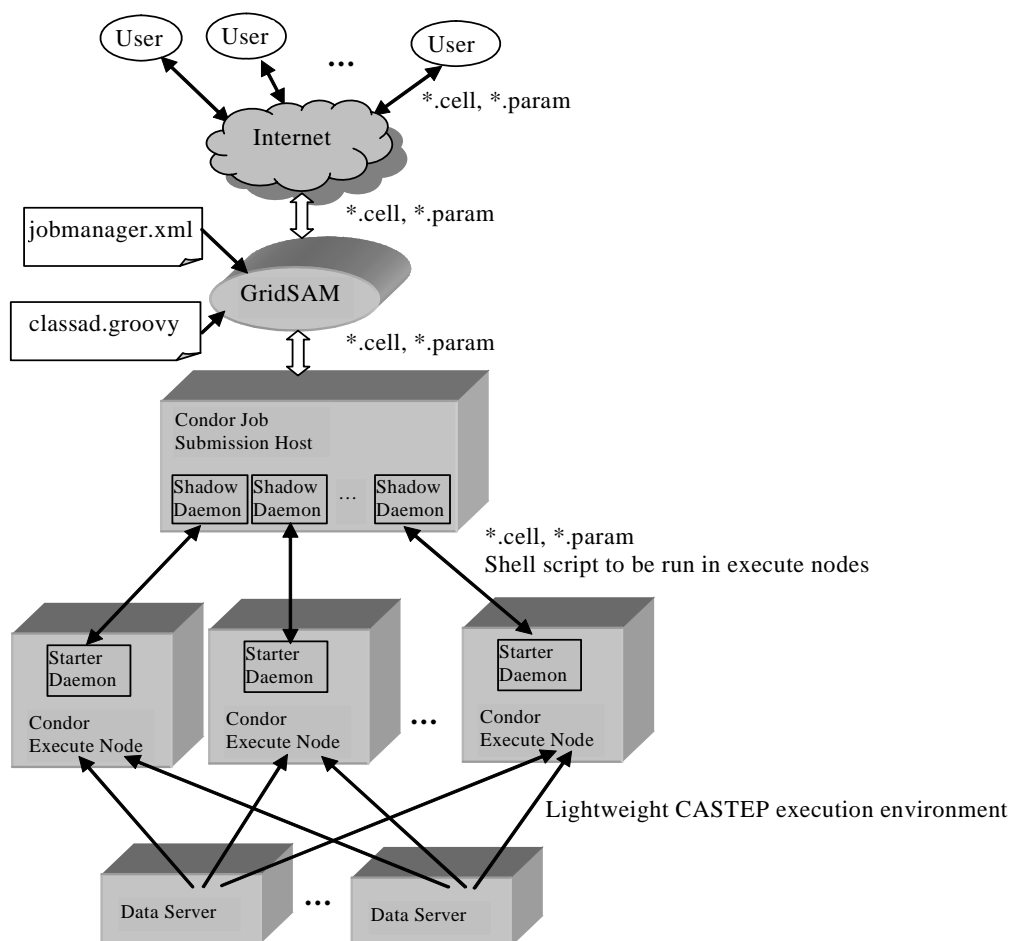


Figure 3. Application execution mechanism based on independent data servers

execution environment. These data servers can be any publicly accessible servers, such as FTP server, HTTP server, or more securely SFTP server. In the performance evaluation in Section 4 the FTP servers are utilised to offer file downloading. Multiple data servers can be set to provide the data replicas and the lightweight CASTEP environment can be downloaded from any one of them.

The access to data servers and execution of a CASTEP application are wrapped in an executable shell script. When the shell script is scheduled onto an execute node, it will first stage in the necessary files from one of the data servers and then call the downloaded CASTEP executable to perform the calculations. The Condor's file transfer mechanism is only used to transfer user-defined input files and this executable shell script. The shell script for the execution mechanism with 2 FTP servers is outlined in Fig.2. The submitted job request is to calculate the total energy of the Al_2O_3 crystal [5]. In the shell script, the files in step 1-2 contain the lightweight environment for running the CASTEP application, and the files in step 3-4 are the potential files needed for calculating the

user-specified Al_2O_3 model. Step 5 generates a random number, which is 0 or 1 in this example. In step 6-13, a FTP server is specified according to the result of step 5. In step 14-22, the specified files are downloaded. Step 23-24 build the execution environment for running the CASTEP application. Step 25 invokes the shell script contained in the constructed execution environment to calculate the model.

The execution mechanism based on data servers is illustrated in Figure 3.

4. Performance Evaluation

This section evaluates the performance of the execution mechanism presented in this paper. Multiple users are simulated to simultaneously submit the CASTEP execution requests via GridSAM (each user submits one request) to calculate the total energy of the Al_2O_3 molecular model [5]. GridSAM then launches the batch of jobs through Condor onto CamGrid. GridSAM 1.1 [14] is deployed in the experiments. Please refer to [1] and [15]

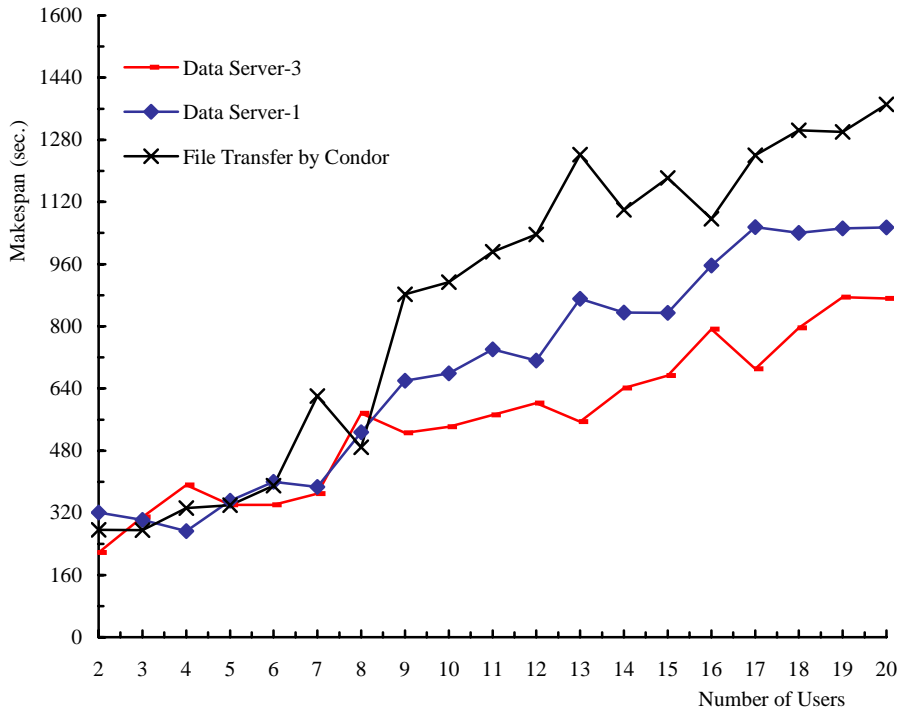


Figure 4. Performance comparison of different execution mechanisms in terms of makespan

for the detailed settings of CamGrid. In this paper, the experimental results for three mechanism settings are demonstrated. These three mechanism settings are:

- 1) Lightweight CASTEP execution environment is transferred through Condor's file transfer mechanism;
- 2) One FTP server is used to provide downloading of the lightweight CASTEP execution environment;
- 3) Three FTP servers are employed, i.e., three replicas are provided for the downloading. Each FTP sever offers the same network bandwidth.

4.1. Makespan

In this subsection, the experiments have been conducted to compare the performance of different execution mechanisms in terms of makespan as the number of the users submitting CASTEP jobs increases. Makespan is defined as the time duration between the time when GridSAM submits the batch of jobs to CamGrid and the time all these jobs are completed. The shorter makespan, the better performance is achieved. The experimental results are demonstrated in Fig.4.

It can be observed from this figure that when the number of users is small (less than 9) the performance curves under these three mechanisms are intertwined with each other in terms of makespan. As the number of the users increases to more than 9, the performance of these

three cases differentiates from each other. The mechanism with 3 data servers outperforms the other two mechanisms while the mechanism which makes use of Condor's file transfer mechanism shows the worst performance.

These results can be explained as follows. When using Condor's file transfer mechanism, the execute nodes obtain the files from the job submission host. In the Condor system, the job submission host is a very busy node. It has to generate a Shadow daemon for every execute node it sends the job to. The Shadow daemon is responsible for communicating with the Starter daemon in each execute node, which includes transferring input files specified in the Condor submission description file. Therefore, although each execute node has a separate Shadow daemon in the job submission host for transferring files, these Shadow daemons work in the same node and share the same network card and a stretch of the network link, and as the result, these Shadow daemons may not be able to transfer the files in parallel, especially when their number becomes high.

Setting up an independent data server to offer data downloading can shift the burden of the job submission host. This can explain why the execution mechanism based on the data server outperforms that based on the Condor's file transfer mechanism in Fig.4. When there are more than one data servers to offer multiple data replicas, the file transfers can be processed in the higher

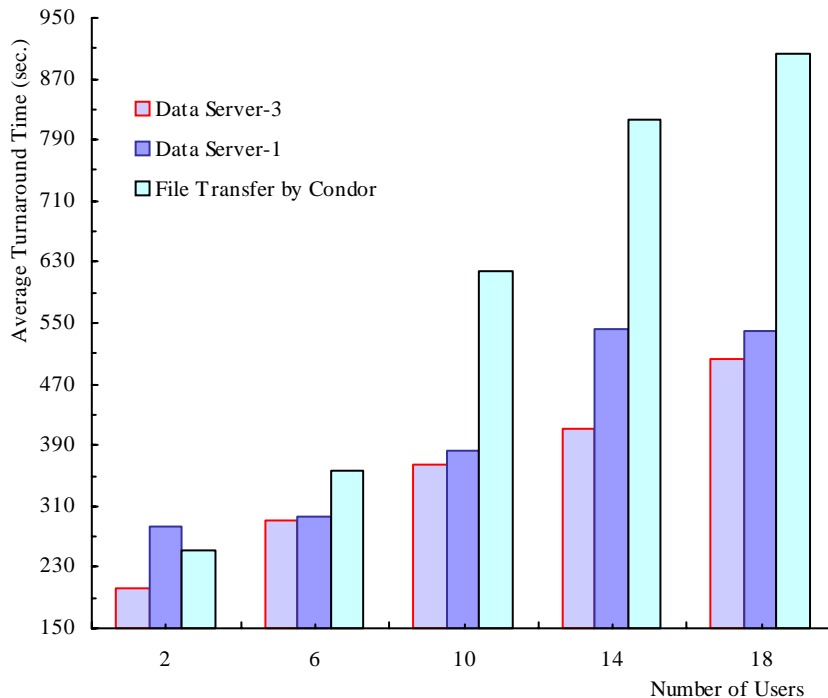


Figure 5. Performance comparison of different execution mechanisms in terms of average turnaround time

degree of parallelism. Hence, the mechanism with 3 data servers performs better than that with one data server. This result suggests that providing multiple data replicas for access are beneficial to improving performance in terms of makespan.

4.2. Average turnaround time

Fig.5 demonstrates the performance of different execution mechanism in terms of average turnaround time under different number of users. A job’s turnaround time is defined as the time duration between the time when it is submitted and the time when the job is completed.

As can be observed from Fig.5, the execution mechanisms based on data servers significantly outperform the one based on the Condor’s file transfer mechanism except the case of 2 users. This may be due to a smoother file transfer with independent data servers. It can also be observed from this figure that the mechanism with 3 data servers achieves better performance than the one with a single data server. Again, this can be explained by the fact that the files can be transferred in higher degree of parallelism when multiple data servers are accessible.

4.3 Overhead of the mechanisms

Fig.6 demonstrates the overhead of the mechanisms presented in this paper. The overhead imposed by using

the mechanism to submit, schedule and run a CASTEP job is defined as all the time that is not spent on executing the job. The overhead includes the time for file transfer, queuing time, the time used to dispatch jobs as well as the time used to collect the calculated results. The average overhead of a batch of jobs is defined as the average of the overhead of all jobs. As can be seen from Fig.6, the data server-based execution mechanisms have much lower overhead than the one based on the Condor’s file transfer mechanism when the number of users is greater than 2. This may be because that in the mechanism using Condor to transfer files, the job submission host becomes a bottleneck since the job management functionalities including file transfer are all performed on this node. In the mechanism based on data servers, a large portion of file transfer functionalities are shifted to data servers. Therefore, the job management can be carried out in the higher degree of parallelism so that the overhead endured by each job is likely to be reduced.

4.4 Speedup

The experiments conducted in this section investigate the speedup achieved by the CASTEP execution mechanisms presented in this paper. It is assumed that the CASTEP execution requests processed by the presented mechanisms are also sent to and run in sequence in the Materials Studio Server where the CASTEP execution

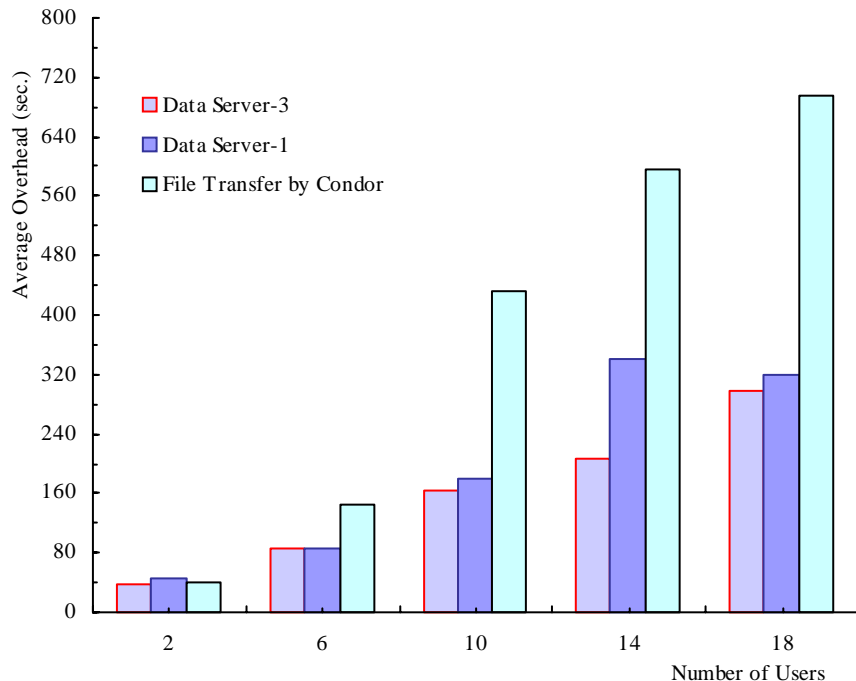


Figure 6. Average overhead of different mechanisms under different number of users

Table 1. Speedup of the CASTEP execution mechanism with 3 data servers in terms of makespan (time unit: second)

The number of users	2	6	10	14	18
Data Server-3	217.8	340.8	541.7	641.4	796.2
Materials Studio	331.4	1230.2	2017.0	2282.7	3590.2
Speedup	1.5	3.6	3.7	3.6	4.5

Table 2. Speedup of the CASTEP execution mechanism with a single data server in terms of makespan (time unit: second)

The number of users	2	6	10	14	18
Data server-1	320.9	399.7	679.0	835.0	1040.6
Materials Studio	472.4	1257.3	2030.3	2802.1	3908.4
Speedup	1.5	3.1	3.0	3.4	3.8

Table 3. Speedup of the CASTEP execution mechanism based on Condor’s file transfer in terms of makespan (time unit: second)

The number of users	2	6	10	14	18
File transfer by Condor	276.2	390.2	913.3	1099.2	1304.3
Materials Studio	423.8	1273.0	1863.7	3093.5	3920.5
Speedup	1.5	3.2	2.4	2.8	3.0

environment is available (therefore, there is no need to transfer the lightweight execution environment).

The speedup of the execution mechanism in this paper over the Materials Studio can be defined as the ratio of the makespan achieved by the Materials Server to the makespan by the execution mechanism.

Table 1 shows the makespan achieved by the execution mechanism with 3 data servers, the makespan when the same batch of jobs is run in the Materials Studio,

and the resulting speedup. Table 2 and Table 3 demonstrate the experimental results of the other two mechanisms in terms of speedup.

It can be seen from these tables, the speedup achieved by all these three mechanisms is greater than 1 even if there exist the overhead of transferring the lightweight CASTEP execution environment and the overhead imposed by using GridSAM job submission service and the Condor job manager. These overheads are relatively

constant for the given execution platform and execution mechanism. Therefore, if the molecular models with the longer calculation time are sent for execution, the speedup to be achieved is expected to be higher (the order of the time for calculating the Al₂O₃ molecular model in these experiments is about 200 seconds).

It can also be observed from these tables that the data server-based execution mechanisms achieve the higher speedup when the number of users is high (greater than 6). This result once again suggests that the data server-based execution mechanism is superior to the one based on the Condor's file transfer mechanism when the number of jobs submitted to the system is high.

5. Conclusions

CASTEP is an application using the density functional theory to calculate the structure and properties of materials and molecules. Currently, the Client/Server architecture in the Materials Studio is the popular approach to running CASTEP applications. This paper investigates the execution mechanisms for running CASTEP in grid resources, where CASTEP execution environments may not be available. In this paper, a lightweight CASTEP execution environment is developed and it can be transferred to any grid computational resource. Based on the lightweight environment, two execution mechanisms have been developed in this work. One makes use of the Condor's file transfer mechanism to transfer the lightweight environment and the other employs the independent data servers. The performance of these execution mechanisms is evaluated on CamGrid in terms of various metrics. The results show that the proposed execution mechanisms can speed up the batch processing of the CASTEP applications; moreover, the mechanisms based on data servers can achieve higher performance than the one based on Condor's file transfer mechanism, especially when the workload level is high. These execution mechanisms also have excellent potentials to be applied to other computational applications as long as the corresponding lightweight execution environments can be developed.

Acknowledgement

The authors would like to thank the DTI for sponsoring MaterialsGrid project.

6. References

[1] M Calleja, B Beckles, M Keegan, M A Hayes, A Parker, M T Dove, "CamGrid: Experiences in constructing a university-wide, Condor-based, grid at

the University of Cambridge", *Proceedings of the 2004 UK e-Science All Hands Meeting*

- [2] M Dove, E Artacho, T White, R Bruin, et al, "The eMinerals project: developing the concept of the virtual organisation to support collaborative work on molecular-scale environmental simulations", *Proceedings of the 2005 UK e-Science All Hands Meeting*
- [3] I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 2003, 2nd edition. ISBN: 1-55860-933-4
- [4] C. Goonatilake, C. Chapman, W. Emmerich, M. Farrellee, T. Tannenbaum, M. Livny, M. Calleja and M. Dove, "Condor Birdbath - web service interface to Condor", *Proceedings of the 2005 UK e-Science All Hands Meeting*.
- [5] V Hoang and S. Oh Molecular, "dynamics study of aging effects in supercooled Al₂O₃", *Phys. Rev. E*, 70, 061203, 2004
- [6] W. Lee, A. McGough, and J. Darlington, "Performance Evaluation of the GridSAM Job Submission and Monitoring System", *Proceedings of the 2005 UK e-Science All Hands Meeting*, Nottingham, UK, 2005
- [7] M.D. Segall, P.J.D. Lindan, M.J. Probert, C.J. Pickard, P.J. Hasnip, S.J. Clark and M.C. Payne, "First-principles simulation: ideas, illustrations and the CASTEP code", *J. Phys. Condensed Matter* 14 (2002) 2117
- [8] D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid", in Fran Berman, Anthony J.G. Hey, Geoffrey Fox, editors, *Grid Computing: Making The Global Infrastructure a Reality*, John Wiley, 2003. ISBN: 0-470-85319-0
- [9] D. Thain, T. Tannenbaum, and M. Livny, "Distributed Computing in Practice: The Condor Experience" *Concurrency and Computation: Practice and Experience*, Vol. 17, No. 2-4, pages 323-356, February-April, 2005.
- [10] J. Wakelin, A. García, P. Murray-Rust, "The use of XML and CML in Computational Chemistry and Physics Applications", *Proceedings of the 2004 UK e-Science All Hands Meeting*, 2004
- [11] Yong Zhang, Peter Murray-Rust, Martin T Dove, Robert C Glen, Henry S Rzepa, Joa A Townsend, Simon Tyrell, Jon Wakelin, Egon L Willighagen, "JUMBO - An XML infrastructure for eScience", *Proceedings of the 2004 UK e-Science All Hands Meeting*.
- [12] <http://www.accelrys.com/products/mstudio/>
- [13] <http://www.materialsgrid.org/>
- [14] <http://gridsam.sourceforge.net/1.1/>
- [15] <http://www.escience.cam.ac.uk/projects/camgrid/>