Design and Implementation of Intelligent Scheduler for Gaussian Portal on Quantum Chemistry Grid

Takeshi Nishikawa, Umpei Nagashima, and Satoshi Sekiguchi

National Institute of Advanced Industrial Science and Technology
Grid Technology Research Center, Tsukuba Central 2
Tsukuba, Ibaraki 305-8568, Japan
{t.nishikawa,u.nagashima,s.sekiguchi}@aist.go.jp
http://unit.aist.go.jp/grid/

Abstract. We have developed a Quantum Chemistry Grid (QC Grid) for simple and effective use of computer resources for ab initio molecular orbital calculations. Ab initio is a powerful methodology in computational chemistry. Gaussian is one of the codes widely used in quantum chemistry research, and it is used not only by quantum chemistry experts, but also by non-experts. Since the CPU cycles of Gaussian jobs vary significantly with the input parameters, it is difficult for users to choose the most adequate computational resources in a local computing environment. By using grid technology on top of a high-speed network environment, QC Grid enables users to easily share the knowledge of experts and efficiently utilize costly computational resources without having to know the specific system environment. Gaussian Portal is the first implementation of the QC Grid concept. It consists of a Web interface, a meta-scheduler, computing resources, and archival resources on Grid infra-wares.

1 Introduction

Ab initio molecular orbital (MO) calculations are used to design novel materials and molecules in fields such as computer-aided chemistry, pharmacy, and biochemistry. The computation cost of the ab initio MO calculation strongly depends on the molecular size and calculation method. In the case of the Hartree-Fock method, which is the simplest of these methods, the computation cost is proportional to $O(n^4)$, where n is the number of atoms in the molecule, and thus it rapidly increases with system size. Therefore, it is very difficult for users to choose appropriate computer resources for an ab initio MO calculation because of the large variety of molecular sizes and calculation methods. The estimation of computational resource requirements is also difficult without extensive experience with ab initio MO calculations. Usually users choose a large high-performance computing (HPC) resource for execution of their jobs, even if the job is small enough to be executed on a PC. For example, more than 70% of the CPU time of the Tsukuba Advanced Computer Center [1] (TACC), National

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2659, pp. 244–253, 2003.

Institute of Advanced Industrial Science and Technology (AIST), is occupied by Gaussian [2] jobs. Gaussian is one of the codes widely used in quantum chemistry research. Furthermore, a large variety of computer resources is required for the Gaussian jobs. The supercomputers at TACC are always busy with Gaussian jobs, even though more than 60% of these jobs are executable on standard desktop PCs.

Therefore, to improve the efficiency of utilization and to facilitate seamless use of different computer resources (supercomputers, servers and PC's), as well as to shorten the turn around time of Gaussian jobs, we have been developing the Quantum Chemistry Grid (QC Grid). The QC Grid adopts Grid technology for security and job scheduling in which the computational and quantum chemical knowledge of the ab initio MO calculation experts is used for computer resource allocation. Namely, the user doesn't need to be conscious of the specific computing resources on the QC Grid. A look-up function is incorporated in a database of the accumulated results so that already existing ab initio MO calculation results can be quickly accessed. The QC Grid also ensures security when accessing distributed HPC resources.

There are several systems [3,4,5] to enable the use of ab initio molecular orbital calculation programs via a Web-based interface or to enable access to a results database. However, in these systems, users select the computer resource by themselves and the Grid environment is insecure.

The QC Grid has an intelligent meta-scheduler and a computing resource allocation feature that is based on expert knowledge about the calculations and the characteristics and/or the state of its resources. We developed Gaussian Portal as the first implementation of the QC Grid.

The remaining sections of this paper describe the QC Grid's design concept and architecture. Our experience in using Gaussian Portal at TACC is also described.

2 Design Concept of the QC Grid

The design concept of this system is as follows. (1) Apply Grid technology to make the computing resource virtual and the computing service accessible at anytime from anywhere while maintaining security. (2) Build a simple and easy-to-use user interface by using Web technology. (3) Appropriately allocate computing resources according to the content of the calculation with the intelligent meta-scheduler. (4) Facilitate sharing of expert knowledge of quantum chemistry and the user's experience. (5) Archive all input and output (results) files. (6) Throughput must be a priority. The system should be inexpensive and have excellent cost performance.

Because the computing resources are physically distributed among several sites, we don't have the resources at each site that can endure the maximum predicted load. Instead, the maximum load of the entire system is the limitation of resource availability. Even if a situation occurs where a part of the computing

resource becomes unavailable, scheduled operations may continue if they can be allocated to the remaining resources, thus maintaining availability and reliability.

Through the Web-based user interface, the user of the QC Grid can use the physically distributed computer resources as if they were a single system, i.e., without having to consider differences in authentication mechanism, operating system, job management system, hardware, computational chemistry application program, and so on.

The user does not have to worry about which computer to use. The meta-scheduler handles that. If a result for a new request exists in the archives and the retrieval time is shorter than for computing it anew, the user can obtain the result from the database instead of having computing resources assigned to do the calculation. Intelligent resource allocation with the meta-scheduler becomes possible because of expert knowledge sharing and because of the preserving of all input and result files. The meta-scheduler helps to optimize the entire computing resource allocation. The system is scalable to the number of jobs.

3 QC Grid Architecture

The QC Grid logical architecture is depicted in Fig. 3. The QC Grid consists of the Web-based user interface, meta-scheduler, knowledge database, results database, Input/Output(I/O) archives, and computing nodes with Grid infrastructure software (Grid infra-ware). Figure 3 shows the I/O flow and QC Grid components relationships controlled by meta-scheduler. The Grid infra-ware ensures security, management of resource allocation, access to remote data, and monitoring of remote resources. The following subsections describe the components.

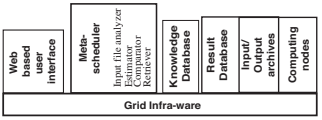

Fig. 1. QC Grid logical architecture

3.1 Web-Based User Interfaces

The QC Grid user has a Web-based interface that does all operations such as uploading thes input file, controlling the job, displaying job status, and getting results. The Web-based interface was created by using a toolkit that can quickly build the portal interfaces. It is served by an HTTP server daemon program. The HTTP server supports Secure Socket Layer (SSL). The Web interface is used to request job resources, control job status, visualize molecular structures, and display results.

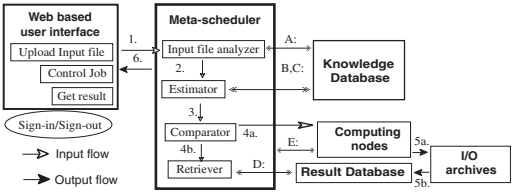

Fig. 2. Flow and relationship of the QC Grid components controlled by the meta-scheduler. 1. Upload input file. 2. Parse input file. 3. Estimate values. 4a. Submit new job. 4b. Retrieve computed data from database and archives. 5. Return computed result from computing node and store result in archives(a) and database(b). 6. Get results via Web interface. Queries are "A: Request system parameters using estimation." "B: Is computing time less than quick searching time ?", "C: Is computing time less than detailed searching time ?", "D: Retrieveing data from Results database.", and "E: Job control".

3.2 Meta-scheduler

The meta-scheduler consists of an input file analyzer, estimator, comparator, and retriever.

Input File Analyzer. The analyzer loads the input file, converts the end- of-line character, and assesses the resource requirements, the calculation method, basis sets, and molecular structure. Resource requirements, the calculation method, basis sets, and molecular structure are described in the input file. An example of the Gaussian input file is shown in Fig. 3.2. The calculation

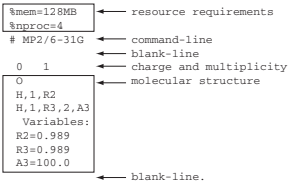

Fig. 3. Gaussian input file

method (HF, MP2, etc.) and basis sets (STO-3G, 3-21G, etc.) are specified on the command-line. The analyzer displays a Web page of its resource choices to the user.

Estimator. The Estimator predicts CPU time, time for retrieving archived results, amount of memory, and disk usage based on the values sent from the Input file analyzer and based on the information in the Knowledge database.

The CPU time estimation is done in accordance with the formula:

$$CPUtime = C_{system} \times C_{JobType} \times C_{Method} \times n^{P_{Method}} \quad (1)$$

C_{system}: system characteristic coefficient, $C_{JobType}$: job type parameter (single point, optimization, frequency, etc.), C_{Method}: method parameter (HF, MP2, etc.), n: Number of basis functions, P_{Method}: power coefficient of method.

Comparator. The Comparator compares the computing time and the retrieval time, and judges whether to retrieve or to compute. It compares the structures of existing results with the structure of the input data.

Retriever. The Retriever has a quick search function and a detailed search function. The former does an exact matching of the computing command-line and the molecule structure. The latter can narrow down results on more than one condition by checking the references and key words.

3.3 Knowledge Database

The Knowledge database contains parameters for computing the resource estimation, performance indices of the computing resources, a retrieval time database, an operation policy, and a list of the organizations facilities.

3.4 Results Database and I/O Archives

The Results database stores all the input files and the location of all output files including checkpoint files. I/O Archives exist on the local file server of each computing resource.

3.5 Computing Nodes

Computing Nodes are installed in the computational chemistry application and the job scheduler. These represent the normal computing environment to which the meta-scheduler can submit the job.

4 Gaussian Portal

Gaussian Portal is the first implementation of the QC Grid. Its system configuration is shown in Fig. 4.

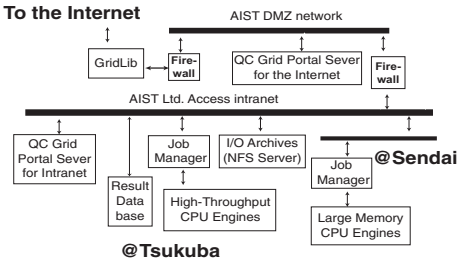

Fig. 4. The QC Grid/Gaussian Portal system configuration.

4.1 System Configuration

We have implemented the QC Grid/Gaussian Portal on geographically distributed servers and clusters in Tsukuba and in Sendai, Japan. These nodes are connected through a Fast-Ethernet switch LAN and uplink to the AIST backbone through Gigabit Ethernet. Two firewalls are installed: one between the Internet and the DMZ network of AIST and the other between the DMZ network of AIST and the AIST intranet. The GridLib server is installed on the Internet. The QC Grid Portal server has the Web server, meta-scheduler, knowledge database server, and Globus [6] Gatekeeper, all in one. There are two QC Grid Portal servers, which are operated under different policies. One is for the Internet, and is installed on the DMZ network of AIST. The other is for the AIST intranet, and is installed on the AIST intranet. The Results database and I/O Archives server (HP AlphaSever GS160), Globus Deploy servers, CPU Engines (IBM x330 cluster and IBM RS6000/SP), and the NFS server are also installed in the AIST intranet region. The GridLib [8] server, two QC Grid Portal servers, the High-Throughput CPU engines and Globus Deploy server for these are located in Tsukuba. Large-Memory CPU engines and Globus Deploy server are located in Sendai. The user can access the Gaussian Portal via the AIST intranet or via the Internet.

In the construction of this system, we gave priority to open source software. The exceptions are Gaussian, the PGI compiler [9] to build Gaussian, the software on the IBM RS6000/SP (AIX, XLF and LoadLeveler) and the software on the HP Alpha Server GS160 (Tru64 UNIX, Oracle). We use Linux for the OS of our machines with Intel architectures. Globus Toolkit, GridPort Toolkit [7], and GridLib prototype are used as the Grid infra-ware. Open-PBS [10] is used as the job manager for the High-Throughput CPU engines.

4.2 Implementations

Portal Interface. GridLib masks hardware and software differences with a common interface. The interface employs a self-signed Certificate Authority (CA) to grant access to the OpenSSL [11]. The CA certificate is installed in the servers and in the client's browsers. Apache [12] is used for the Web server,

and mod_ssl is added for deploying SSL. The GridPort Toolkit and the codes written in Perl or Java were used for the construction of the Portal Interface.

Meta-scheduler. The meta-schedulers modules were written, for the most part, in Perl and Java. Some modules such as the molecule structure comparator and some of the numerical solvers were written in Fortran.

The CPU time estimation is done according to Eq. (1). The inputs of the standard Gaussian test are used as a benchmark to decide the value of the coefficients of Eq. (1). The memory and disk requirement are read directly from the input file. The amount of memory and/or disk space limit is specified in the input file. The meta-scheduler decides what resources to allocate to each job based on this information.

The meta-scheduler accesses the databases by using JDBC [13] technology. It provides cross-DBMS connectivity to a wide range of SQL databases. In this way, we dont have to be aware of the database differences. We use PostgreSQL as the DBMS of the knowledge database and Oracle as the DBMS of the results database.

Knowledge Database. The values extracted from the knowledge database store the meta-scheduler scripts or otherwise the response time would be too slow. For example, the values extracted from the knowledge database such as retrieval time for the database (4 seconds), Csystem for Large-Memory engines (half that of the High-Throughput CPU engines), and the memory requirement threshold (500MB) are embedded in the meta-scheduler scripts.

Results Database and I/O Archives. The Results database stores key words extracted from the input files and the location of all output files and checkpoint files. The I/O Archives exist on the NFS server.

Computing Nodes. Gaussian 98 Rev.A9 is installed in each computing node. The OS on the IBM x330 cluster is RedHat Linux 6.2J and the OS on the RS6000/SP is AIX 4.3.3. The job manager on the IBM x330 cluster is OpenPBS and the job manager on IBM RS6000/SP is LoadLeveler. The CPU Engines do not have an elapsed time limitation for a job. The single scratch file had to be under 2GB on the High-Throughput CPU engine. The scratch file space is limited to under 20GB on the High-Throughput CPU engine, and to under 18GB on the Large-Memory CPU engines. Jobs that require more than 500MB are executed on the Large-Memory CPU engines.

5 Experience

We report on practical operation experience weve gained from operation of the QC Grid/Gaussian Portal at the Tsukuba Advanced Computing Center

(TACC). The QC Grid/Gaussian Portal for the AIST intranet began operation in February 2002. Figure 5 compares the monthly normalized sum of CPU time on the IBM x330 cluster (Pentium III 1.2GHz 2way SMP/node, 108 nodes) and computing nodes of the QC Grid/Gaussian Portal from February to June, 2002 (black bars) with data for the same period of TACCs independently operated IBM RS6000/SP (POWER3 200MHz 2way SMP/node, for which 40 nodes are provided for serial applications such as Gaussian) (gray bars). The single CPU performance of the x330 is about two times greater than that of the RS6000/SP. Thus the sum of the CPU time of IBM x330 cluster was scaled by a factor of two in this @comparison. The monthly values were scaled at 100% of the RS6000/SP in February.

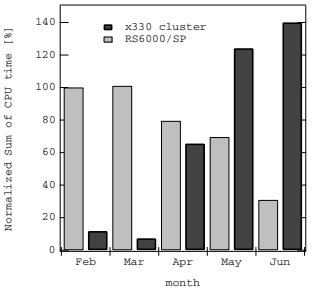

Fig. 5. The monthly normalized sum of CPU time. The ratio is based on the RS/6000SP CPU time in Feb. being equal to 100%. The normalized CPU time of the x330 cluster was scaled to twice that of the RS6000/SP because the single CPU performance of the x330 was two times that of the RS6000/SP

6 Discussion

The experience we gained during this period led us to switch from the IBM RS6000/SP to the IBM x330 cluster. As can be seen in Fig. 5, the series of normalized monthly sum of CPU time data on the IBM x330 cluster rapidly increased, whereas that on the RS6000/SP decreased. The usage ratio (RS6000/SP to IBM x330 cluster) became 1 : 5 in June. The users chose the Gaussian Portal to run their jobs. We couldnt evaluate the usefulness of the Results DB or the I/O Archives during this time, as these functions have just been installed and there still are not enough archived results to be useful to the general user.

7 Conclusion and Future Plans

We designed the QC Grid so that scientists from various fields can access quantum chemical computing services at anytime and from anywhere while maintaining security with Grid technology. We have developed and are operating the

first implementation of the QC Grid as a Gaussian Portal. The meta-scheduler includes job scheduling functions and enables optimal allocation of the entire computing resource for a large number of jobs. It can be used with computers of various classes, although so far we have prepared only the Linux cluster and IBM RS6000/SP as the back-end computing resources for Gaussian. Our experience with Gaussian Portal at TACC shows that it can manage resources more efficiently than the previous system could. Introduction of the QC Grid/Gaussian Portal has made available surplus computing power in the computing resources occupied by Gaussian. It has increased the availability of the RS6000/SP, which has high-speed inter-connects for running parallel jobs.

Although the Gaussian Portal is fully implemented in the QC Grid, its implementation and concept still need tuning. We have to improve its estimation quality and implement automatic reconfiguration of parameters by using these estimations. A feature to control the disclosures of the results DB and I/O archives should be added to the QC Grid.

The Gaussian Portal is the first implementation of the QC Grid, and it is strongly dependent on Gaussian98. We will generalize the logic of the meta-scheduler to extend the portal to work with other computational chemistry programs such as GAMESS [14], NWChem [15], and ABINIT-MP [16]. Our first priority is to settle any licensing issues with Gaussian Inc., who may not have anticipated this implementation. Gaussian is sold under a single node license or site license that only users who belong to an organization with a single address are permitted to access the program.

Regarding the portal, we have to migrate from GridPort to GridLib to unify the user interface with other portals being developed by our colleagues, which include portals not only the computational chemistry calculations but also portals for calculation services of other fields, such as molecule dynamics, bio-infomatics, and weather forecasting.

The proper distribution of computing resources to various fields is necessary because the Linux cluster consisting of thousands of nodes becomes the main stream of the HPC resources. This can be readily seen by noting that a total of 63 PC clusters are in the TOP 500 supercomputer sites.

The general idea of the meta-scheduler developed with the QC Grid can be applied to meet the above resource demands. However, a cleanup of the software and packaging will be needed, before it can be introduced to the public.

Acknowledgements. We would like to thank the other members of the Grid technology research center, AIST, especially Dr. Osamu Tatebe, Dr. Yoshio Tanaka, Mr. Mototaka Hirano, Dr. Hidemoto Nakada, and Mr. Hiroshi Takemiya for their helpful discussions and encouragement. We also would like to thank Dr. Hikaru Samukawa, Japan IBM research center, for his helpful discussions and encouragement. The computing resources (IBM x330-108-node Linux cluster and IBM RS6000/SP cluster) are partly supported by the Tsukuba Advanced Computing Center (TACC). This research is partly (GridLib) supported with a grant for Research and Development Applying Advanced Computational Science and Technology, from the Japan Science and Technology Corporation (ACT-JST).

References

1. The mission of the TACC is to provide computing resources to the research units of the National Institute of Advanced Industrial Science and Technology(AIST). http://unit.aist.go.jp/tacc/
2. Gaussian 98 (Revision A.9), M.J. Frisch, G.W. Trucks, H.B. Schlegel, G.E. Scuseria, M.A. Robb, J.R. Cheeseman, V.G. Zakrzewski, J.A. Montgomery, Jr., R E. Stratmann, J.C. Burant, S. Dapprich, J.M. Millam, A.D. Daniels, K.N. Kudin, M.C. Strain, O. Farkas, J. Tomasi, V. Barone, M. Cossi, R. Cammi, B. Mennucci, C. Pomelli, C. Adamo, S. Clifford, J. Ochterski, G.A. Petersson, P.Y. Ayala, Q. Cui, K. Morokuma, D K. Malick, A D. Rabuck, K. Raghavachari, J.B. Foresman, J. Cioslowski, J V. Ortiz, A.G. Baboul, B.B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. Gomperts, R.L. Martin, D.J. Fox, T. Keith, M.A. Al-Laham, C.Y. Peng, A. Nanayakkara, C. Gonzalez, M. Challacombe, P.M.W. Gill, B.G. Johnson, W. Chen, M.W. Wong, J.L. Andres, M. Head-Gordon, E.S. Replogle and J.A. Pople, Gaussian, Inc., Pittsburgh PA, 1998. Gaussian Inc.: http://www.gaussian.com/
3. CODATA SESSION PC2: Physics and Chemistry M. Aoyagi et al., The Development of a Distributed Computational Environment for Molecular Design on the World Wide Web, http://www.codata.org/newsletters/nl75.html, MOSRV/Linux, http://cresta530.cc.kyushu-u.ac.jp/~planet/
4. Australian Computational Chemistry via the Internet Project, http://www.chem.swin.edu.au/
5. https://gridport.npaci.edu/GAMESS/, Kim Baldridge, Ph.D.: kimb@sdsc.edu, Jerry Greenberg, Ph.D.: jpg@sdsc.edu
6. Globus: A Metacomputing Infrastructure Toolkit. I. Foster, C. Kesselman. Intl J. Supercomputer Applications, 11(2):115–128, 1997. GLOBUS project: http://www.globus.org/
7. M. Thomas, S. Mock, J. Boisseau, M. Dahan, K. Mueller, D. Sutton. The GridPort Toolkit Architecture for Building Grid Portals. Proceedings of the 10th IEEE Intl. Symp. on High Perf. Dist. Comp. Aug 2001
8. Personal Communication. Yoshio Tanaka: yoshio.tanaka@aist.go.jp
9. The Portland Group, Inc.: http://www.pgroup.com/
10. Veridian Systems: http://www.openpbs.org/
11. The OpenSSL Project: http://www.openssl.org/
12. The Apache Software Foundation: http://www.apache.org/
13. http://java.sun.com/products/jdbc/
14. "General Atomic and Molecular Electronic Structure System" M.W. Schmidt, K.K. Baldridge, J.A. Boatz, S.T. Elbert, M.S. Gordon, J.H. Jensen, S. Koseki, N. Matsunaga, K.A. Nguyen, S. Su, T.L. Windus, M. Dupuis, J.A. Montgomery J. Comput. Chem., 14, 1347–63 (1993). http://www.msg.ameslab.gov/GAMESS/GAMESS.html
15. High Performance Computational Chemistry Group, NWChem, A Computational Chemistry Package for Parallel Computers, Version 4.1 (2002), Pacific Northwest National Laboratory, Richland, Washington 99352, USA. http://www.emsl.pnl.gov:2080/docs/nwchem/nwchem.html
16. T. Nakano, T. Kaminuma, T. Sato, K. Fukuzawa, Y. Akiyama, M. Uebayasi, K. Kitaura, Chem. Phys. Lett. 351 (2002) 475–480.