Paper ID: 298

Deploying Scientific Applications to the PRAGMA Grid Testbed: Strategies and Lessons

David Abramson, Amanda Lynch, Hiroshi Takemiya, Yusuke Tanimura, Susumu Date, Haruki Nakamura,
Karpjoo Jeong, Suntae Hwang, Ji Zhu, Zhong-hua Lu, Celine Amoreira, Kim Baldridge, Hurng-Chun Lee,
Chi-Wei Wang, Horng-Liang Shih, Tomas Molina, Wilfred W. Li, Peter W. Arzberger

***Abstract* — Recent advances in grid infrastructure and middleware development have enabled various types of applications in science and engineering to be deployed on the grid. The characteristics of these applications and the diverse infrastructure and middleware solutions developed, utilized or adapted by PRAGMA member institutes are summarized. The applications include those for climate modeling, computational chemistry, bioinformatics and computational genomics, remote control of instruments, and distributed databases. Many of the applications are deployed to the PRAGMA grid testbed in routine basis experiments. Strategies for deploying applications without modifications, and those taking advantage of new programming models on the grid are explored and valuable lessons learned are reported. Comprehensive end to end solutions from PRAGMA member institutes that provide important grid middleware components and generalized models of integrating applications and instruments on the grid are also described.**

***Index Terms*—PRAGMA, grid, applications, deployment, strategies, lessons**

I. Introduction

The grid environment has evolved rapidly over the past few years, with a number of scientific applications serving as drivers for development of middleware and grid infrastructure [1]. While the exponential growth in processor power, storage, bandwidth and fiber lays the foundation for the new computing infrastructure, the grid will only succeed if it attracts users, and meets their needs [2]. Scientists and engineers often are the first to plunge into the new technology and come out with valuable lessons learned for posterity.

Over the past 10 years, development effort in grid middleware in projects such as Globus [3] has enabled a middleware layer that handles essential functionalities such as user authentication, authorization, data storage and transfer. However, in the last 3 years, a new layer of tools have emerged or matured that build on top the “lower middleware” and further enhance the ease of use for scientific and engineering applications (Figure 1).

In this paper, we describe applications that are deployed on the PRAGMA (Pacific Rim Application and Middleware Assembly [4]) testbed [5] with or without modifications. We also describe some comprehensive grid application environments being developed by PRAGMA member institutes that provide end to end solutions for grid computing.

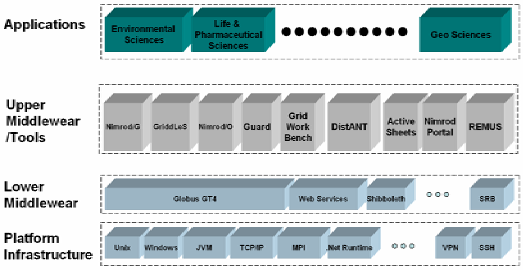

Figure 1. User applications and experiences on the grid are enhanced by upper middleware and tools

II. Deployment Strategies

A. *Applications without code modifications*

Parameter sweep applications are those that iterate over certain sets of parameters for a particular application as required by the research problem at hand. Here we’ll discuss the climate modeling and computational chemistry complex workflow, as well as bioinformatics applications using Nimrod [6] and Gfarm [7]. These middleware support the deployment of applications without requiring changes in existing applications. The no change or minimal changes on “legacy” applications significantly increases the appeal of the grid to a wider audience.

1) Paleoclimate experiments with Nimrod

The Australian monsoon is critical to the environment and

Manuscript received November 30, 2005. The author names are sorted in alphabetical order by name of the country, institute, and last name where appropriate. PRAGMA member institutes are funded by funding agencies of their respective governments. CNIC wishes to acknowledge NSFC-60533020 and MSTC-2005DKA64000-2. PRAGMA is supported by NSF Grant No. INT-0216895 and INT-0314015. WWL, PWA also wishes to acknowledge NIH NCRR P41 RR08605 to NBCR. WWL wishes to acknowledge Fang-Pang Lin, Habibah Wahab, and the participants at the 1st South Asia International Joint Research and Training workshop hosted by NCHC, Taiwan, for their valuable input.

D. Abramson and A. Lynch are with the University of Monash, Australia, Victoria 3800, Australia. (e-mail: {david.abramson, amanda.lynch}@infotech.monash.edu.au). H. Takemiya and Y. Tanimura are with AIST, Japan (e-mail: {h-takemiya, yusuke.tanimura}@aist.go.jp). S. Date and H. Nakamura are with CMC, and Institute for Protein Research at Osaka University, Japan, repectively. ({date@ais.cmc,harukin@protein}.osaka-u.ac.jp). K. Jeong and S. Hwang are with {Konkuk, Kookmin} University, Korea. (e-mail: {jeongk@konkuk, sthwang@kookmin}.ac.kr). J. Zhu and Z. Lu are with CNIC, CAS, China. (e-mail: {zhuji, zhlu}@sccas.cn). C. Amoreira and K.Baldridge are with University of Zurich, Switzerland (email {amoreira,kimb}@oci.unizh.ch). H. Lee, C. Wang, H. Shih are with Academia Sinica, Grid Computing, Taiwan. (e-mail: {hclee,chiwei,hlshih@ccweb.sinica.edu.tw). T. Molina, K. Baldridge, W. W. Li, and P. A. Arzberger are with University of California, San Diego, La Jolla 92093, USA. (phone: 858-822-0974; fax: 858-822-0861; e-mail: molina@ncmir.ucsd.edu; *wilfred@sdsc.edu*; *parzberg@ucsd.edu*).

economy of the north Australian region, which produces wealth for Australia out of proportion to its population. It delivers life-sustaining moisture to a dry continent, although lightning associated also causes devastating fires. During the past, the monsoon has varied both spatially and in intensity. Contemporary ecosystems have adapted to these extremes, but how tightly linked are burning, vegetation, and rainfall? What might these linkages mean for future Australian water resources?

Interactions between atmosphere, ocean and land in the context of the Australian monsoon are complex, as they result from feedbacks that operate on a variety of spatial and temporal scales. Until now, computational and data volume limitations have hindered efforts to reach a full understanding of the biophysical processes and the mechanisms for long term variation in the natural monsoon system. Performing simulations of the type and extent needed to understand monsoon dynamics require more computational resources than one is likely to obtain from any single high performance computing centre (HPC) in Australia. Moreover, the databases required to drive the simulations are distributed and replicated at different centers. As a result, there are "few secure facts concerning when and why the Australian summer monsoon developed or how it has varied" [8]. In this context, an ongoing initiative seeks to develop the capability for the simulation and analysis of the natural variability of the Australian monsoon.

-- Use of Nimrod: Over the past 10 years, Abramson *et al* have developed a software tool called Nimrod/G, which allows a user to migrate a particular class of applications to the Grid [6, 9]. Specifically, it automates the execution of parameter sweep and search applications (parameter studies) over global computational grids. Nimrod is particularly novel because it supports user-defined deadline and budget constraints for scheduling computations and manages the supply and demand of resources in the Grid using an experimental computational economy. Nimrod/G supports the type of parameter study required in the paleoclimate modeling experiment to understand the Australian monsoon. It allows the scientists to vary the initial conditions and various other parameters of the climate models, as well as performing a large number of Monte Carlo style simulations.

-- Pilot Study over PRAGMA testbed: The pilot study utilizes an existing Australian atmosphere-land model, the Cubic Conformal Atmospheric Model (C-CAM) developed at the CSIRO Division of Atmospheric Research. C-CAM has 18 vertical levels, hydrostatic dynamics, and a single-layer canopy with 44 vegetation types. The sea surface temperatures are prescribed as well as the wind components using a far field nudging technique. This configuration reproduces the Australian Monsoon climatologically in a reasonable way.

A one month simulation using the single CPU version of C-CAM takes 150 minutes to run on a 2.8 GHz INTEL XEON architecture using the INTEL FORTRAN 95 v8.0 compiler with modest optimization enabled. Typical paleoclimate experiments are on the order of 30 years for a single realization – such an experiment then requires 38 days of wall clock time and 100 GB of forcing data to complete. However, because the earth's climate is a chaotic system with thousands, if not millions, of degrees of freedom, an appropriate experimental design requires ensembles of realizations and ranges of sensitivity experiments. These may be considered to be highly parallel in nature even when the parallelized version of C-CAM is not used.

The core component for process control and the distribution of the runs is Nimrod/G, and is described in more detail in an accompanying paper [5]. The use of nimrod/G enabled successful executions of the C-CAM on the PRAGMA testbed.

2) *QM/APBS in computational chemistry*

Rational drug design relies on computational models that determine whether and how small drug ligands interact with large molecules like proteins. This is done by calculating the ligand-protein configuration that minimizes the binding energy. The methods are often complex because the code performs both energy calculations as well as nonlinear optimizations. Baldridge *et al* has proposed an alternative framework, enabling user-specific exploration using a QM (quantum mechanics)/APBS (adaptive Poisson-Boltzmann Solver)/MD (molecular dynamics) hybrid method. The ligand is treated using QM methods, and the full complex is treated using classical electrostatics (APBS) [10] and empirical force field (MD) methodology. The first phase involves only QM/APBS, with the ultimate goal being to envelop the classical electrostatics directly into the QM procedure, and combine with MD methods.

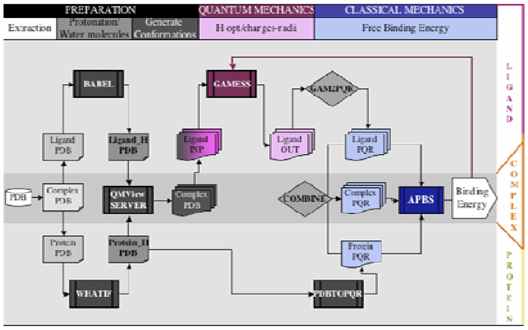

Figure 2. Various steps involved in the hybrid method for study of ligand-protein interactions.

Computationally, the hybrid methodology involves many individual steps, many of which we have now eliminated through creative middleware and web services implementations, most noticeably on the PRAGMA testbed, through the use of the Nimrod/G. The electrostatic effects are numerically solved using APBS. The atomic charges and radii for the ligand are determined using QM. Additionally, QM provides more accurate structural and energetic information for the ligand, an important factor in the determination of which structure is preferred in the pocket of the protein or enzyme. All charges and radii must also be specified for the electrostatics computation.

The overall goal is to understand the binding energy and mechanism associated with the complex formation of the ligand and protein, and how they vary with position, substitution (residue mutation), and environmental conditions. In particular, it is desirable to define a large set of parameters required in defining the binding energy, parameters having to do with ligand positioning and environmental conditions, with the added flexibility to run investigations over a wide set of possible mutations in structure. In addition, it is also important to determine more accurate energy functions for predicting the total binding energy for a protein-ligand complex. The total theoretical method involves multiple tools and resources, including molecular modeling software, databases, and auxiliary tools, all managed by the grid middleware tool, Nimrod, discussed above.

3) *Protein annotation studies using iGAP*

The international genome sequencing effort has steadily produced a large number of complete genomes. Since 1995, more than 180 complete, and over 1000 partial proteomes have been made publicly available. Progress is being made towards high quality proteome annotation through a combination of high throughput computation as well as manual curation. Increasingly the new knowledge is translated into tangible diagnostic and remedial procedures for the benefits of public health and education. Grid technology promises to meet the increasing demand in large scale computation and simulation in the fields of computational biology, chemistry and bioinformatics. The integrative Genome Annotation Pipeline, iGAP [11], provides functional annotation using known protein structural information. The computational requirement of iGAP and the initial experience in using AppLeS Parameter Sweep Template (APST) [12] to deploy it on the grid has been previously described by Li *et al* [13].

While the previous system provides many rich features and works well for the dedicated workflow, it requires detailed knowledge to operate and requires significant effort to generalize to other applications. Ease of use is fundamental for the widespread adoption of grid technologies in life sciences, where application scientists do not have the time to keep up with the ever changing grid computation standards and models. As Moore's law continues to hold true, commodity computing clusters is becoming a reality on university campuses across the globe. While this trend is fundamental to the maturation of the grid, it also poses a new challenge. Many users prefer to run bioinformatics applications within a cluster environment, which is the most reliable production environment to date, despite recent advances in grid middleware technology.

One problem often experienced in both the cluster and grid environment is the limitation of file I/O. In a cluster environment, if an application generates a lot of intermediary and output files, the load on the NFS server may become quite high, as the number of compute nodes increases, and I/O becomes a rate limiting step. Code modification is required to move the file I/O to local disks on compute nodes. Additional code is required to reliably transfer the end results from the compute nodes back to the NFS server. Even with a dedicated cluster where local disk may be used for data storage, it becomes difficult to find the results on various compute nodes if one wishes to revisit the data after a certain period of time.

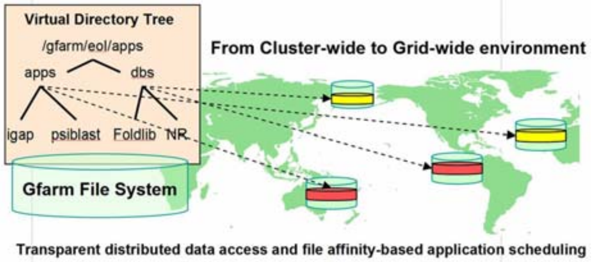
Virtual Directory Tree
/gfarm/eol/apps
apps
dbs
igap
psiblast
Foldlib
NR
Gfarm File System
From Cluster-wide to Grid-wide environment
Transparent distributed data access and file affinity-based application scheduling

Figure 3. Gfarm file system enables centralized distribution of biological databases and applications.

In a grid environment, network latency, bandwidth shortage, and overhead in file transfer are often prohibitive to the effective grid deployment of legacy applications which produce many output files with sizes ranging from tens of kilobytes to tens of megabytes. Gfarm^{TM} provides a scalable and transparent solution, which enables effective use of not only the distributed computing power, but also the disk storage space, through a familiar virtual file system view (Figure 3) [5, 7]. The capability of Gfarm to support "legacy" applications without code modification has proven valuable for lowering the cost of entry to the grid. Additional metascheduling using the Community Scheduler Framework 4 (CSF4) [14] provides the cross site scheduling abilities offered by Nimrod/G or APST, with additional features being developed for a full fledge metascheduler.

B. *Applications with code modifications*

While running applications without code modifications allows great flexibility and shields the application scientists from the overhead of learning about grid technology and programming models, new programming models may enhance the performance and fault tolerance of applications re/designed for the grid. These cases are explored below using bioinformatics applications, and QM/MD/MM (molecular mechanics) studies.

1) *Building high throughput BLAST service using MPICH-G2*

Following the successes in genomic sequence analysis, BLAST has been widely used as a fundamental tool in majority of bioinformatics applications. Thus the efficiency of BLAST becomes an essential metric of application performance. The mpiBLAST, developed by LANL, is a parallel BLAST implemented by using the NCBI toolbox libraries. The database splitting scheme improves the BLAST performance without requiring high-end computers [15].

From the service perspective, the fast growing database and user demands of BLAST searches require resources that cannot be offered by traditional PC clusters. Considering the limited resources in computing center and the fact that several life science institutes are running BLAST on their own computing resources, the idea of building high throughput BLAST services is to link these resources as a single computing pool by using the Grid technology so that one can leverage the idle computing slots to facilitate on time consuming and on-demand BLAST searches.

-- Modifications to mpiBLAST: Lee *et al* aims to enable cross-cluster CPU and database sharing features in mpiBLAST on the GT2 based PRAGMA testbed. Figure 4 shows the schematic diagram as well as the workflow of the GT2-enabled mpiBLAST. Porting mpiBLAST on PRAGMA testbed can be done simply by re-compiling the source code with the GT2-enabled MPI implementation, MPICH-g2. With the use of GT2 DUROC (Dynamic-Update Request Online Coallocator), mpiBLAST-g2 can handle MPI processes running in different clusters; however, the job request needs to be consistent with the mpiBLAST-g2 configuration at each site. Instead of using mpirun, an mpiBLAST-g2 job submitter was developed to produce a site-dependent RSL (Resource Specification Language).

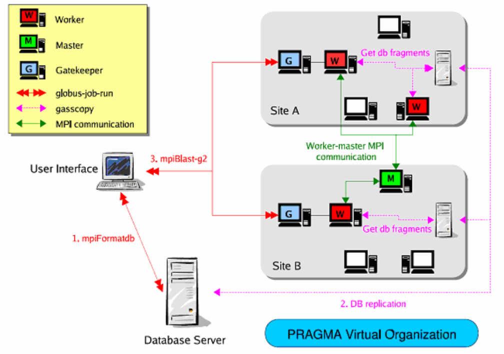

Figure 4. Using MPICH-G2 for MPI-BLAST

To reduce the I/O overhead of BLAST search, simple copy is used in mpiBLAST to fetch the database fragments from a central database repository to local disk before the searching. This limits the location of the repository to an internal network in order to minimize security risks. In mpiBLAST-g2, the copy method was replaced by GASS (Global Access to Secondary Storage) to adopt the GSI (Grid Security Infrastructure) based data access protocol for global database sharing.

-- Observations: The test runs on PRAGMA testbed show that mpiBLAST-g2 can leverage the idle computing resources from multiple clusters to improve the throughput of BLAST service; however, the MPI job in cross-cluster mode has two major issues:

(1) Inbound connectivity is required on each CPU. In the PRAGMA testbed, not all resources can offer public IPs for cross-cluster MPI applications due to site administration policies. Nevertheless, resources without inbound connectivity can still be used to run small jobs in single-cluster mode. For instance, the mpiBLAST-g2 job submitter can select resources in an intelligent way with the use of a static site information table and switch jobs in between single and multiple cluster modes based on the characteristics of the selected resources.

(2) Job hangs due to single CPU failure. Since the grid is a dynamic environment, some unexpected transient network problem might result in a single CPU failure thus causing the whole MPI job to hang while waiting for a response from the failed process. This can lead to inefficient use of grid resources. Although one can periodically check and cleanup the unhealthy jobs in the backend, a better way to address this issue is to introduce a more resilient framework to handle distributed jobs on the grid. Several possibilities, such as fault tolerance MPI implementations and master-worker framework like DIANE [16], are under investigation.

2) *QM/MD simulations in material science & engineering*

The hybrid QM/MD technique for atomistic simulations is becoming more and more important in the field of modern material science and engineering, such as designing future electronic devices or micro-machines. In order to design delicate and robust devices, designers need a microscopic analysis such as the stress distribution of the material or the deformation process. The hybrid QM/MD simulation [17] is a combination of a classical MD simulation with a Density Functional Theory (DFT)-based QM simulation. The simulation calculates the behavior of atoms in the entire region based on the MD scheme, while QM simulation is performed in interesting regions to improve the MD result.

Typically this type of simulation is difficult to execute on a single cluster, because thousands of CPUs must be used over several months to calculate the result on the real problem even using the hybrid method. Takemiya *et al* redesigned the QM/MD simulation code to use Ninf-G for grid deployment and job scheduling using GridRPC (Grid Remote Procedure Call).

a) QM/MD simulation code redesign considerations

In implementing the code, the following three requirements are deemed to be very important when executing a large scale simulation on the grid for a long time.

(1) Flexibility: When considering the long run simulation, it is unrealistic to expect exclusive access to computing resources over the entire simulation time, because these resources on the grid are generally shared by many users. One must assume that each cluster will be available only for a part of the simulation time. The code should, therefore, be flexible enough to continue simulation on different target clusters dynamically.

(2) Robustness: The grid is inherently unstable and heterogeneous. To the matter worse, the more resources used for the simulation, the higher the probability of trouble events. The code should, therefore, be robust against the network/cluster trouble events, including long queuing time.

(3) Efficiency: The code should be highly parallelized to reduce the computation time.

Although several grid programming models have been proposed [18], it is difficult for these models to satisfy all the requirements at the same time. For example, grid-enabled MPI enables the code to execute efficiently, but it does not provide the mechanism for dynamic resource switching or for failure recovery.

b) Combined GridRPC and MPI strategy

As a result of the above considerations, two programming models, GridRPC [19] and MPI are used. GridRPC is designed to support RPC on the Grid based on the client-server paradigm. It has functions for dynamic execution of server programs, for detection of network/server errors, and for time-outing to avoid waiting for a long time. These functions may be used to satisfy above first two requirements. On the other hand, MPI is used to realize efficient execution of both MD and QM simulation. In implementing the code, we used Ninf-G [20] and MPICH [21] as reference implementations of GridRPC and MPI, respectively.

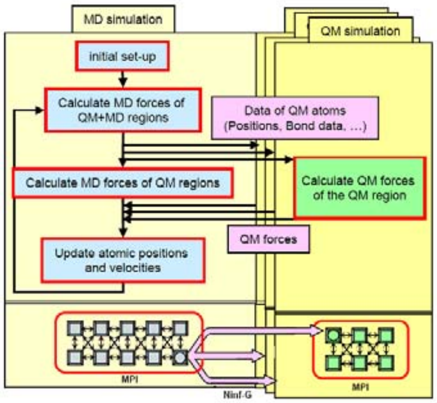
MD simulation
QM simulation
initial set-up
Calculate MD forces of QM+MD regions
Data of QM atoms (Positions, Bond data, ...)
Calculate MD forces of QM regions
Calculate QM forces of the QM region
QM forces
Update atomic positions and velocities
MPI
Ninf-G
MPI

Figure 5. Execution flow of the QM/MM simulation

The execution flow of the code, divided into two parts, MD part and QM part, is depicted above (Figure 5). First of all, MD part calculates the behavior of atoms in the entire region in parallel using MPI. Then, it sends the data of QM atoms to each QM part using GridRPC. Each part calculates the force in the QM regions in parallel using MPI again. After finishing all the simulation in the QM region, MD part gathers the force data to update atomic positions and velocities. By repeating the cycle, the simulation precedes the time step. When some trouble on servers takes place, the client stops execution of the target QM simulation and try to allocate it on another cluster to restart the simulation.

c) Preliminary Results of the long run experiment

The target experiment is a simulation of the diffusion problem in a box-shaped Si system. One cluster was used to execute the MD simulation and seven clusters for QM simulation. The total number of atoms is 1728. Five QM regions were defined totally, each of which has only one QM atom. The QM simulation of each region is allocated dynamically on five clusters among seven.

The experiment continued for two weeks, in which the code tried to execute QM simulations 47593 times. Over this period, 524 trouble events occurred, such as connection failure, batch system down, and queuing time-out. Most of them (80 %) occurred during allocating QM simulation on the target cluster. But even after the allocation, there are trouble events such as exceeding CPU time limit which results in the termination of a program by the batch system.

In spite of these troubles during the experiment, the code succeeded in continuing simulation by changing the target cluster. The result clearly shows that the approach, combining GridRPC and MPI, is valid for long run simulations on the grid.

3) Time-Dependent Density Functional Theory (TDDFT) experiment

TDDFT [22] is used for molecular simulations in computational quantum chemistry. TDDFT was parallelized with Ninf-G.

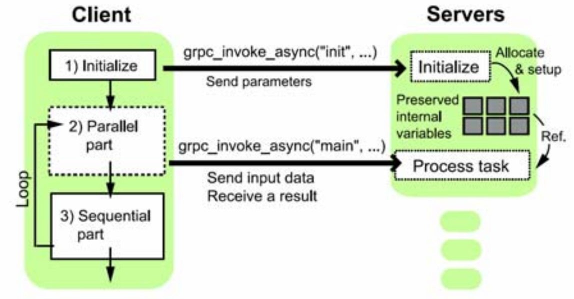

Figure 6. Implementation of TDDFT using Ninf-G

-- Implementation of TDDFT: In general, the key equation in TDDFT is converted to a linear equation by the linear approximation of the reflectance function so that it is calculated on a computer as a problem of the frequency domain. The redesign adopted a method for numerical integration of the equation because it is suitable to parallel computation. As shown in above (Figure 6), steps 1) arrays initialization, 2) time evolution of the orbital $(\varphi_i(t) \rightarrow \varphi_i(t+\delta t))$ and 3) calculation of n(r), V_H and V_{xc} with φ_i are iterated with incrementing the time step. Mostly 2) is a hotspot of TDDFT calculation which can be divided into multiple tasks to perform numerical integration for each orbital.

Implementation of TDDFT using Ninf-G calculation requires the least modification of original code. In Figure 6, φ_i and potential data are sent to the server in the asynchronous RPC for main calculation step 2) and renewed φ_i' is returned to the client. Step 3) is executed with φ_i', subsequent to completion of asynchronous calls.

-- Lessons learned: Typically, a hotspot would be more than 50% of total calculation to simulate medium size molecules. For example, the simulation of the ligand-protected Au_{13} molecule, 122 RPCs are invoked at every time step and more than 5,000 iterations are required to achieve significant or practical results. The total number of RPCs will be 610,000. Because one RPC costs a few seconds on the Pentium III 1.0 GHz processor, the estimated calculation time could be one week. In addition, each RPC requires large data transmission: 4.87MB to the server and 3.25MB from the server.

The TDDFT program was parallelized within a short time and achieved reasonable performance on a single cluster system. However, some RPCs failed when the program is run in the testbed. The failure was caused by poor communication performance on the PRAGMA testbed between some sites, such as at most 300 KB/sec, less than 40 KB/sec in the worst, and varies significantly over time. Consequently, retry and timeout mechanisms of the RPC session, elimination and recovery methods of a down server were added to the client program to overcome the instability, using optional functions of Ninf-G. In the end, the modified program achieved the estimated one-week execution time. Further changes to the algorithm are necessary to reduce the communication overhead or to keep down the calculation cost for scalability on the grid.

In terms of system support and middleware requirement, the hardest part was to setup and test all client and server programs on all sites. The server programs were rarely updated after the distribution. Sometimes, manual logins to remote sites are necessary to kill zombie processes that were left by the unhandled faults and to check an error log for each server. These problems are shared by most applications and should be automated by administrators' and middleware developers' efforts.

III. End to end grid computing environment solutions

The end to end grid computing environments described here provide more than user interfaces such as web sites, portals or desktop clients, which improves user experience of a particular deployment strategy. They consist of components designed for specific hardware environments, instrumentation, or provide significant new enhancements to Globus toolkit, or offer new standards towards grid service interoperability.

A. Development of BioPfuga (Biosimulation Platform United on Grid Architecture)

The usual usage of the grid architecture is to run one computation on many distributed CPUs through a high-speed network. However, in order to analyze much more complicated biological systems, composed of simulations at different levels, on a new paradigm for biological science, more integrated computational approaches are required. BioPfuga is the computing grid component of the BioGrid project headed by Shinji Shimojo at the Cybermedia Center of Osaka University. The BioPfuga group developed their own biological simulation programs, which cover a wide range of fields in biological science from electronic analyses of biological macromolecules to cellome and Physiome research. In particular, AMOSS (*Ab initio* Molecular Orbital System for Supercomputer) for electronic state simulation [23] and pretsoX-basic for protein molecular dynamics simulations [24, 25] have been driver applications for the BioGrid architecture.

QM/MM simulation is highly suitable for the Grid. Most of biological simulation programs have been separately developed by distinct organizations. This means that it is difficult to integrate these programs into a single program. Rather, it is easy to integrate and manage these programs on a wide-area computational environment, or the grid. Also, some simulation programs require special hardware devices. This means that it is difficult to integrate these traditional programs into a single program at a single site.

BioPfuga aims to establish a research infrastructure where scientists and researchers can flexibly combine a variety of simulation programs to a multi-scale simulation program for their research purpose. To this end, grid services or web services for each simulation are developed based on the OGSI standards, and are now migrated to WSRF. In addition, a standard XML format to describe data of bio-molecular simulation, BMSML, has been developed, and abstraction of data transfer interface is achieved by utilizing the inheritance of web service interface. Wrapping each simulation with grid services has achieved integration of simulations interfaces with keeping each simulation's independence.

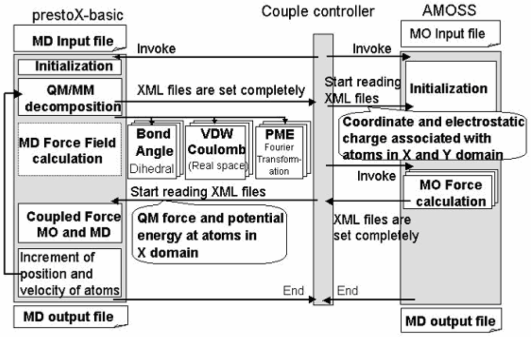

Figure 7. An example of BMSML in Base64 form for the data exchange between different computational programs for actual execution of Grid computations.

BioPfuga requires that (1) application programs be divided into a set of many pieces, each of which corresponds to a unit simulation procedure, and that (2) data communication be made between the program pieces by a standard description. For the former requirement, the simulation unit should not be too small for rapid computation so that the data communication time among different machines is minimal. For the latter problem, a simple, standard description has been developed using XML, BMSML for the data exchange between different computational programs (Figure 7).

Three types of XML description are used: a text form, a hexadecimal form, and a Base64 form. When the Base64 form is used, the size is only about 1.3 times larger than that used in the binary form. The advantage of the XML form for intermediate and output data is that any meta-data can be easily added as an attribute or as tagged information in addition to the actual computed data to be exchanged among the different application programs. It should be emphasized that the unit of data can always be provided in BMSML, so that the different application programs recognize and confirm the unit system for computation and analysis.

B. Telescience

Scientific imaging instruments are used in a variety of disciplines to gather vital data for research and study. Specifically, in the biomedical field, various types of biological imaging instruments, such as electron microscopes and light microscopes, are used everyday to acquire 2D and 3D datasets for further understanding of biological structures. Remote operation or "tele-operation" of instruments has become a popular solution for research scientists to acquire and share data across research domains separated by geographical barriers.

Many of these scientific imaging instruments such as electron microscopes, light-microscopes, and synchrotrons share similar properties and hardware functionality features. The Telescience architecture [26] uses web services and grid services to achieve interoperability, scalability, and performance within the architecture. It is multi-tiered and uses the notion of architectural fragments, which are reusable components that describe a design pattern or a framework. A single architectural fragment describes the structure of architecture in terms of its components, also known as *roles*. Each of these components is represented as a web service, each having its own distinctive interface for interaction. These components can be composed with each other and with other reusable components to build up the framework.

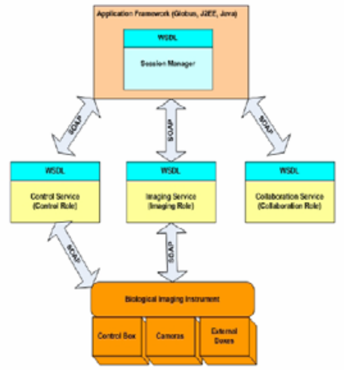
Application Framework (Globus, J2EE, Java)
WSDL
Session Manager
SOAP
SOAP
SOAP
WSDL
Control Service (Control Role)
WSDL
Imaging Service (Imaging Role)
WSDL
Collaboration Service (Collaboration Role)
SOAP
SOAP
Biological Imaging Instrument
Control Box
Cameras
External Doors

Figure 8. Multi-tier service based software architecture for telemicroscopy

Grid services are incorporated to achieve another level of integration and increased performance not possible with just web services themselves. Using grid services allows the software components to take full advantage of all the distributed resources on a Grid to effectively query, process, and store data from instruments. Finally, a set of software plug-in specification libraries is presented to describe how developers can interface with this software framework. These libraries abstract the complexities of the Grid, web services, security, and other underlying logic to the software developer in order to focus on creating a usable client interface for a particular instrument (Figure 8).

C. *MGrid and* e-*Glycoconjugates*

Molecular simulation is considered to be a promising research technique for many current and future bio/chemical research areas since the experimental methods such as X-ray and NMR spectroscopy are not efficient enough to obtain full structural information of bio-macromolecules. However, the simulations for the bio-conjugates of protein, DNA, lipid, and carbohydrates often needs much more than the computing capacity of large scale clusters or supercomputers. Simulation results on those molecules whose three-dimensional structures or appropriate simulation settings are not well-known are difficult to validate without aid of real experiments. These two critical problems, *challenging requirements of computation power and simulation results validation*, of the technique have prevented the popular and reliable use of the molecular simulations.

The MGrid system is designed to address these two issues. The system provides a shared and integrated molecular simulation grid environment for computing, databases, and analyses which consists of computational grids, data grids, and semantic grids. In addition to sufficient computing power due to grid computing, it also allows scientists to verify simulation results *in a collaborative way*, by sharing simulation jobs (e.g., input files) and results, by comparing simulation results on similar biological or chemical molecules, and by creating new simulation jobs by modifying previous ones (e.g., with different parameter values). The MGrid system is currently running on the grid testbed at the Applied Grid Computing Center of the Konkuk University (Figure 9).

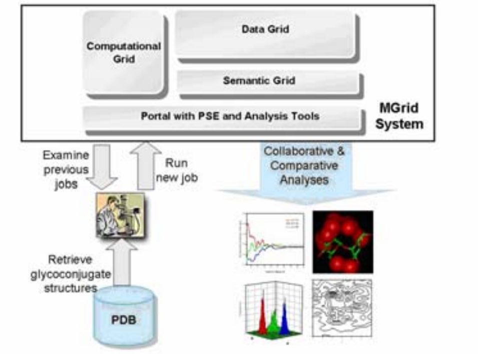
Computational Grid
Data Grid
Semantic Grid
Portal with PSE and Analysis Tools
MGrid System
Examine previous jobs
Run new job
Collaborative & Comparative Analyses
Retrieve glycoconjugate structures
PDB

Figure 9. e-glyconjugate environment in M*Grid

e-Glycoconjugates, is a grid portal for molecular simulation of Glycoconjugates, which is constructed by the MGrid system and designed to provide database services and analysis environments for simulation results. It is a novel alternative computational solution to overcome aforementioned two critical problems of molecular simulation for the advanced computational research on the structure and function of glycoconjugates. Services for simulation results on more than 2,000 glycan chains and 100 glycoproteins (which are so far available in structural databases such as Protein Data Bank or GlycoScience DataBase) are expected in three years.

D. *Bioinformatics grid in CNGrid and ChinaGrid*

CNGrid is supported by Chinese National Hi-tech Program (“863 Project”) and ChinaGrid is supported by “211 Project” of Ministry of Education in China. CNGrid has 8 nodes distributed in CAS (Chinese Academy of Sciences) and universities. The two fastest supercomputers in China are equipped in the CNGrid. ChinaGrid integrates all of the “211 Project” universities in China is therefore much larger in scale. It uses CERNET (China Education and Research Network) to connect the computational resource in the universities in China.

The middleware of CNGrid is GOS (Grid Operating System) v2 and the middleware of ChinaGrid is CGSP (ChinaGrid Support Platform). GOS v2 is based on OGSA and CGSP is developed according to the latest grid specification WSRF. They support multiple service types, such as super (virtual) services, common web services and WSRF services. CGSP has more powerful local selection as it has many nodes consisting of CNGrid nodes and other Universities across the country.

More than 30 bioinformatics applications have been deployed to both grids, some without modifications and others with optimizations for parallel computing. For further details, please refer to [27]. Briefly, a given application is accessible from a web portal and an appropriate cluster or supercomputer, which may belong to different VO's, is selected automatically after a given request is received.

IV. Summary and Discussion

There are a number of applications deployed on the grid on a routine basis within the PRAGMA testbed. The PRAGMA workshops held twice yearly and the collaborations among PRAGMA working groups in Biosciences, Telescience, Resources and Data Computing have been instrumental in the successful deployment of these applications. While the paper by no means covers all the exciting grid activities in member institutes, the spirit of scientific collaboration and training permeating PRAGMA workshops means that knowledge is continuously produced based on our collective experiences.

Some key lessons learned from these routine basis experiments are as follows: 1) Lower middleware level that deals with local cluster authentication, computation, data management has reached a level where some stability is achieved for routine basis experiments. 2) Upper middleware that deals with cross-cluster computation, virtual organizations, and meta-scheduling still require improvement. For example, it's difficult to schedule a large number of nodes simultaneously without manual intervention. 3) Some applications may be deployed without modifications though changes may be required if better performance and fault tolerance is desired.

In addition, experiences from PRAGMA members based on experiments on their own testbed also have valuable lessons. Progress is under way to deploy reusable components of the comprehensive end to end solutions within the PRAGMA testbed or at other member institutes. Some of these applications require resources beyond those available in the current PRAGMA testbed. In order for more applications to run on the grid, there are several approaches: 1) Develop new applications with the grid in mind. 2) Execute existing applications without modifications. 3) Develop upper middleware which enable the legacy applications. 4) Realize that some applications should not be run on the grid. 5) Develop new programming models for grid access.

Many different approaches have been tried, and the grid may stay heterogeneous due to human nature. However, interoperability may be possible with availability of more open source software, and adoption of common integration technologies including but not limited to web services.

References

[1] I. Foster and C. Kesselman, "The Grid 2: Blueprint for a New Computing Infrastructure," 2 ed. San Francisco: Morgan Kaufmann Publishers, Inc., 2004.

[2] L. Smarr, "Grids in Context," in *The Grids 2: Blueprint for a new computing infrastructure*, I. Foster and C. Kesselman, Eds. San Francisco: Elsevier, 2004, pp. 3-12.

[3] Globus, "The Globus Alliance," 2004, http://www.globus.org.

[4] P. W. Arzberger and P. Papadopoulos, "PRAGMA: Example of Grass-Roots Grid Promoting Collaborative e-Science Teams," in *CTWatch Quarterly*, vol. Feb 2006, 2006.

[5] C. Zheng, D. Abramson, P. W. Arzberger, S. Ayuub, C. Enticott, S. Garic, M. Katz, J.-H. Kwak, B. S. Lee, P. M. Papadopoulos, S. Phatanapherom, S. Sriprayoonsakul, Y. Tanaka, Y. Tanimura, O. Tatebe, and P. Uthayopas, "The PRAGMA Testbed: building a multi-application international grid," presented at CCGrid, Singapore, 2006.

[6] D. Abramson, J. Giddy, and L. Kotler, "High performance parametric modeling with Nimrod/G: Killer application for the global grid?," presented at IPDPS, 2000.

[7] O. Tatebe, N. Soda, Y. Morita, S. Matsuoka, and S. Sekiguchi, "Gfarm v2: A Grid file system that supports high-performance distributed and parallel data computing," presented at 2004 Computing in High Energy and Nuclear Physics, Interlaken, Switzerland, 2004.

[8] D. M. J. Bowman, "The Australian summer monsoon: a biogeographic perspective," *Austr. Geog. Studies*, vol. 40, pp. 261-277, 2002.

[9] D. Abramson, R. Sosic, J. Giddy, and B. Hall, "Nimrod: A tool for performing parameterised simulations using distributed workstations," presented at HPDC, 1995.

[10] N. A. Baker, D. Sept, S. Joseph, M. J. Holst, and J. A. McCammon, "Electrostatics of nanosystems: application to microtubules and the ribosome," *Proc Natl Acad Sci U S A*, vol. 98, pp. 10037-41, 2001.

[11] W. W. Li, G. B. Quinn, N. N. Alexandrov, P. E. Bourne, and I. N. Shindyalov, "A comparative proteomics resource: proteins of Arabidopsis thaliana," *Genome Biol*, vol. 4, pp. R51, 2003.

[12] H. Casanova and F. Berman, "Parameter sweeps on the Grid with APST," in *Grid Computing: Making the Global Infrastructure a Reality*, F. Berman, G. C. Fox, and A. J. G. Hey, Eds. West Sussex: Wiley Publishers, Inc., 2003.

[13] W. W. Li, R. W. Byrnes, J. Hayes, A. Birnbaum, V. M. Reyes, A. Shahab, C. Mosley, D. Pekurovsky, G. B. Quinn, I. N. Shindyalov, H. Casanova, L. Ang, F. Berman, P. W. Arzberger, M. A. Miller, and P. E. Bourne, "The Encyclopedia of Life Project: Grid Software and Deployment," *New Generation Computing*, vol. In Press, 2004.

[14] X. Wei, W. W. Li, O. Tatebe, G. Xu, L. Hu, and J. Ju, "Integrating Local Job Scheduler - LSF^{TM} with Gfarm^{TM}," *Lecture Notes In Computer Science*, vol. 3758, pp. 197, 2005.

[15] A. Darling, "The Design, Implementation, and Evaluation of mpiBLAST,," presented at ClusterWorld, San Jose, 2003.

[16] DIANE, "DIANE framework," 2005, http://cern.ch/diane.

[17] S. Ogata, R. K. Shimojo, A. Nakano, and P. Vashishta, "Hybrid Quantum Mechanical/Molecular Dynamics Simulations on Parallel Computers: Density Functional Theory on Real-space Multigrids," *Computer Physics Communications*, vol. 149, pp. 30-38, 2002.

[18] C. Lee, S. Matsuoka, D. Talia, A. Sussman, M. Mueller, G. Allen, and J. Saltz, "A Grid Programming Primer.," presented at GWD-I, GGF Advanced Programming Models Research Group, 2001.

[19] K. Seymour, H. Nakada, S. Matsuoka, J. Dongarra, C. Lee, and H. Casanova, "Overview of GridRPC: A Remote Procedure Call API for Grid Computing.," *Lecture Notes In Computer Science*, vol. 2536, pp. 274-278, 2002.

[20] Y. Tanaka, H. Nakada, S. Sekiguchi, T. Suzumura, and S. Matsuoka, "Ninf-G: A Reference Implementation of RPC-based Programming Middleware for Grid Computing.," *J. of Grid Computing*, vol. 1, pp. 41-51, 2003.

[21] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "A High-Performance, Portable Implementation of the MPI Message Passing Interface Standard.," *Parallel Computing*, vol. 22, pp. 789-828, 1996.

[22] K. Yabana and G. F. Bertsch, "Time-Dependent Local-Density Approximation in Real Time: Application to Conjugated Molecules," *Quantum Chemistry*, vol. 75, pp. 55-66, 1999.

[23] T. Sakuma, H. Kashiwagi, T. Takada, and H. Nakamura, "Ab initio MO study of the cholorphyll dimer in the photosynthetic reaction center I. A theoretical treatment of the electrostatic field created by the surrounding proteins," *Int. J. Quant. Chem.*, vol. 61, pp. 137-151, 1997.

[24] Y. Fukunishi, Y. Mikami, and H. Nakamura, "The filling potential method: a method for estimating the free energy surface for protein-ligand docking," *J. Phys. Chem B.*, vol. In Press, 2005.

[25] N. Nakajima, J. Higo, A. Kidera, and H. Nakamura, "Free energy landscapes of peptides by enhanced conformational sampling," *J. Mol. Biol.*, vol. 296, pp. 197-216, 2000.

[26] T. E. Molina, G. Yang, A. W. Lin, S. T. Peltier, and M. H. Ellisman, "A Generalized Service-Oriented Architecture for Remote Control of Scientific Imaging Instruments," presented at CCGrid, Singapore, 2005.

[27] Z. Ji, Z.-h. Lu, Y.-w. Wu, A. Guo, B. Shen, and X.-b. Chi, "Superficial analysis of the bioinformatics grid technique application in China," presented at CCGrid 2006, Singapore, 2006.