Grid-BGC: A Grid-Enabled Terrestrial Carbon Cycle Modeling System

Jason Cope*, Craig Hartsough†, Peter Thornton†, Henry M. Tufo*†,
Nathan Wilhelmi† and Matthew Woitaszek*

* University of Colorado, Boulder
† National Center for Atmospheric Research
matthew.woitaszek@colorado.edu

Abstract. Grid-BGC is a Grid-enabled terrestrial biogeochemical cycle simulator collaboratively developed by the National Center for Atmospheric Research (NCAR) and the University of Colorado (CU) with funding from NASA. The primary objective of the project is to utilize Globus Grid technology to integrate inexpensive commodity cluster computational resources at CU with the mass storage system at NCAR while hiding the logistics of data transfer and job submission from the scientists. We describe a typical process for simulating the terrestrial carbon cycle, present our solution architecture and software design, and describe our implementation experiences with Grid technology on our systems. By design the Grid-BGC software framework is extensible in that it can utilize other grid-accessible computational resources and can be readily applied to other climate simulation problems which have similar workflows. Overall, this project demonstrates an end-to-end system which leverages Grid technologies to harness distributed resources across organizational boundaries to achieve a cost-effective solution to a compute-intensive problem.

Introduction

Setting up and running high-resolution simulations of terrestrial biogeochemical (BGC) processes is currently an involved process for scientists. Performing a complete simulation consists of gathering required environmental data from various storage systems onto one platform, running preprocessing software to prepare meteorological data for the target model, executing the simulation itself, and then moving the data to other systems for post-processing, visualization, and analysis. The process must then be repeated for multiple simulation tiles constituting the desired geographical region requiring mundane repetition and attention to detail. As such, the overhead to running terrestrial biogeochemical simulations is quite high, and scientists must perform many manual tasks and possess adequate data storage, computational resources, and substantial platform-specific computer expertise.

The objective of the Grid-BGC project is to create a cost effective end-to-end solution for terrestrial ecosystem modeling. Grid-BGC allows scientists to easily configure and run high-resolution terrestrial carbon cycle simulations without having

to worry about the individual components of the simulation or the underlying computational and data storage systems. In order to run a simulation, the user interacts with a web-based portal to control the various stages of processing. The portal then functions as a grid client, submitting the simulation to a Grid-BGC tile processing grid service that gathers the required data from the storage systems and performs the simulation.

The development of Grid-BGC is a collaborative effort between the National Center for Atmospheric Research (NCAR) and the University of Colorado (CU). The Grid-BGC project uses computational and data grid technology [3] to leverage the resources available at both organizations in order to provide a cost effective and high performance solution. In particular, Grid-BGC is designed to run on commodity cluster systems such as those available at the university instead of production supercomputer systems at NCAR. Large model runs, however, produce multi-terabyte output in excess of the capacity available on the university clusters, so the system utilizes the NCAR mass storage system (MSS) for its storage requirements. Our software solution is also designed to provide reliable model execution tolerant of the transients present in distributed grid systems, support NCAR's operational security requirements, and be extensible enough to support running other similar scientific models.

As we engineer the Grid-BGC software, our overall goal is to develop an extensible set of grid-enabled tools that solve this problem and will be useful for subsequent similar Grid-based projects. The software infrastructure developed for Grid-BGC enables application-oriented data accessibility. Instead of requiring users to manually locate data by searching through a reference interface, entire applications can be configured to locate and download required data. In the past, these simulations would have to be performed at NCAR in order to gain access to the mass storage system. This is no longer the case, as data grid technologies allow the data to be accessed from anywhere.

The remainder of this paper is organized as follows: Section 2 describes relevant related projects in the grid community and Section 3 presents the NCAR software workflow required for terrestrial this ecosystem model forming the basis for our system requirements. Section 3 presents our solution architecture and design, and section 4 relates the current state of our prototype implementation and test grid. Section 5 describes our experiences with cluster-based grid computing. The final sections present future work and conclusions.

Related Work

Many other organizations are developing projects similar to the Grid-BGC execution platform. These projects, which range from holistic graphical workflow manipulation tools to client-server distributed processing systems, differ in approach and magnitude. All of these tools are service-based and allow computational platforms to expose computational resources as a commodity for the use of a community. While we are presenting our solution in respect to our targeted terrestrial climate model, our

software environment is completely general and usable by applications with similar characteristics.

One example of running legacy applications in a Grid environment is the Grid Execution Management for Legacy Code Architecture (GEMCLA) project [4]. The goal of GEMCLA is to provide a framework designed to make any legacy code executable as an Globus Toolkit (GT) 3.0 [3] compliant Grid service without manually turning each application into a Grid service, access to the legacy source code, or requiring custom Java executable wrapping. GEMCLA functions as a Grid service with a front-end that interacts with the client to pass parameters and a back-end to run jobs using the Globus master managed job factory service. The interface to the legacy code is described in an XML file. GEMCLA also provides a robust graphical workflow editor, uses traditional Globus Toolkit components for job execution, and also provides workflow management and portal services.

While GEMCLA focuses on running single applications in a grid environment, other projects provide managed computing services. For example, the Distributed Infrarastructure with Remote Agent Control (DIRAC) project developed by CERN coordinates computational resources for large physics simulations [9]. DIRAC is a high-throughput service oriented computational grid middleware application. In the DIRAC architecture, a user submits a series of computational jobs to a central server, much like a queue on a traditional cluster. Software running on each compute site determines its free computing resources, and then polls the central server to retrieve jobs for processing. The server runs a "Matchmaker" service to select the best jobs for the available resources. The authors assert that this pull methodology is less complex and more scalable than a traditional server-based scheduling system that must maintain the state of every compute node at all times. DIRAC implements its own data management system, including replica catalogs and a reliable data transfer service. Job execution is flexible, as each job simply installs software required for to its execution.

Similarly, NorduGrid was developed to handle large physics simulations [2]. The authors considered the use of previous grid computing tools, such as Globus and software developed by the European Data Grid, but found that they were as a whole inadequate and other components were required. NorduGrid augments the Globus Toolkit with a user interface, grid manager, replica catalog, and information dissemination service. The user interface, installed on client machines, provides the ability for users to submit job requests and obtain system information. The information service provides information on storage and computing resources on a grid using the Globus Monitoring and Discovery System (MDS). Finally, the grid manager provides an interface layer between the grid and the system software, such as a batch scheduling system. NorduGrid utilizes the Globus Replica Catalog to locate data sources and GridFTP to transfer files, but is intended for operation on cluster computer systems with locally shared file systems. Job requests are flexible and are submitted to the Grid Manager using Resource Specification Language.

Our approach to service-oriented computing is different. Instead of introducing complexity to support future arbitrary software execution, we impose *a priori* administration overhead to ensure that specific applications may be executed on resources that advertise their availability. Other solutions utilize a job description language, such as the Globus Resource Specification Language (RSL), to describe

jobs in their most basic terms, such as requested architecture, requested number of nodes and processors, requested data capacity. Then, when the job is actually scheduled on a computer, the user's application and data must be transferred to that platform and executed. Many things can go wrong, ranging from compilation problems, long-distance storage system access problems, and a host of issues related to client environment configuration management.

We approach service-oriented computing from a contract perspective. In our architecture, a computational resource broadcasts that it can provide, for example, the Grid-BGC tile processing service. This broadcast availability demonstrates a commitment to provide the service with minimal details. The executable has been installed and tested, required security relationships have been established, and paths to remote storage systems have been tested. Instead of a job description language like RSL, the client submits a processing job using a generic specification format suitable for many types of executables but with additional stanzas specific to the advertised service. Our software approach provides a fault tolerant computational offloading grid service for specifically configured applications.

Terrestrial Ecosystem Modeling

Our software system uses two NCAR software applications, Daymet and Biome-BGC, to simulate the terrestrial ecosystem in a three step workflow (see Fig. 1). The Biome-BGC model is point-based; that is, it simulates the ecosystem at a single point on a spatial grid representing an area of planetary surface. The model itself acts on only one point at a time, but multiple points within a region are aggregated into tiles that become the unit of work for the Biome-BGC simulation. For a simulation, the area of land under analysis is broken up into manageable tiles and each tile is simulated independently.

The first stage in the workflow is preprocessing to convert raw single-site meteorological data into the spatially gridded format required by the simulator. The data ingest program Daymet [7] interpolates ground-based weather observations to produce high-resolution grids of historical surface weather data. These tile weather fields are then stored for possible later re-use. The Daymet output is then piped into the Biome-BGC model [8] in conjunction with soil and plant data. The model simulates the terrestrial carbon, water, and nitrogen cycles. The soil and plant data specification essentially defines forests and deserts, and the Daymet output describes where it rains, so the model grows trees in the forests and saguaro cacti in the deserts. The output is post-processed to display map overlays of variables of interest to climatologists such as gross primary production of carbon by photosynthesis.

The point- and tile-based nature of the Daymet and Biome-BGC models require that a scientist pay careful attention to parameter setup and spatial tile decomposition. Before Grid-BGC, the scientist running the simulation was required to manually organize the preprocessing and model execution for every tile independently. The Grid-BGC system architecture is designed to automate the process and eliminate this overhead. User interaction is constrained to a web-based portal interface, and the

system automatically generates spatial tiles necessary to run the simulation over a desired area.

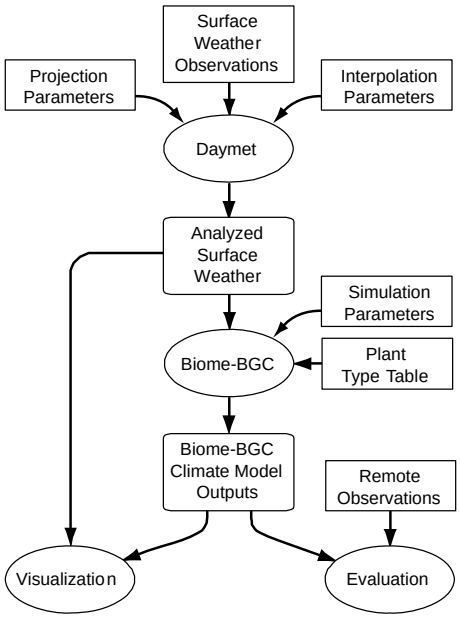

Fig. 1. Carbon cycle modeling workflow

Architecture

Our software architecture is composed of several components that work together in a coordinated manner using Grid technology. GT 3.2 is used to provide grid enabled web services, authentication protocols, and data transfer facilities (see Fig. 2). These features are utilized in the four primary components of Grid-BGC: the user interface portal, the grid service, the JobManager daemon, and the DataMover file transfer utility.

The user interface portal provides the front-end for the Grid-BGC system. The interface exposes mechanisms to define simulation parameters and support collaboration between users. A grid-enabled client is integrated into the portal, which interfaces with the remotely executing grid service. The client can interpret data received from the grid service and present it to the user and can communicate user requests to start simulations and query simulation status to the grid service. The portal essentially provides a thin web-based client that can accommodate a distributed user base with heterogeneous systems easily and efficiently.

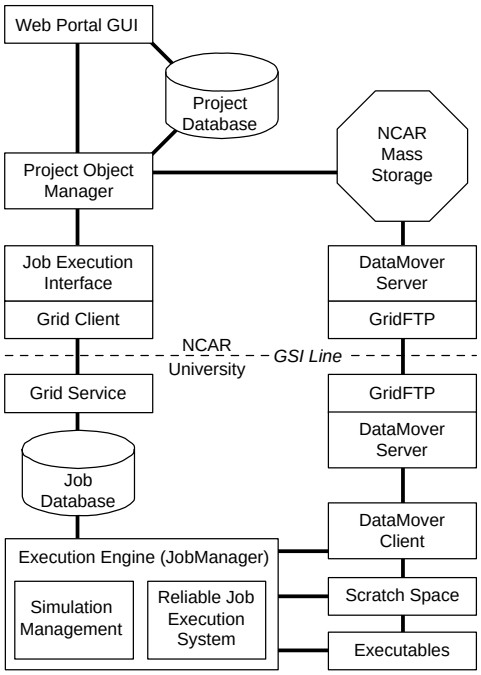

Fig. 2. System architecture

A single grid service, the Grid-BGC Grid Service, is exposed by our execution framework. The primary responsibility of the service is to act as a link between the user interface portal and the execution environment. The service allows clients to invoke methods that start, stop, and analyze the state of Daymet and Biome-BGC jobs executing in the Grid-BGC environment. Communication between the service and the user interface portal is accomplished through the transmission of an XML based specification language. This language is not specific to Grid-BGC. Instead, the language is intended to support running any executable requiring input files, initialization files, and command line arguments. Upon receiving a message from the portal the service method parses the message, stores the parsed data into a persistent database, and then executes the appropriate action specified by the message. Once the action has been completed, the method composes a response message and sends it to the client in the portal. Typical actions include starting, stopping, and querying the state of a simulation.

The JobManager daemon is responsible for managing the high-level execution tasks of the Biome-BGC jobs and controlling the simulation of their constituent tiles. This includes preparing and priming a simulation by fetching needed input data sets from remote storage, starting, stopping, and monitoring the tile Daymet and Biome-BGC simulations, monitoring the execution environment, and performing cleanup operations when a job completes. A persistent database is used to store all the actions

the server must execute, the state of all active processes in the Grid-BGC execution environment, and the state of the JobManager itself.

The final component of our solution required for Grid-BGC simulations is DataMover [6]. This file transfer facility, developed by Lawrence Berkeley National Laboratories for the Earth System Grid (ESG) project, is designed to replicate large sets of files between tape- and disk-based mass storage systems using GridFTP [1] and is currently in use at national laboratories in the United States. We use DataMover to transfer files between NCAR and CU. DataMover provides GSI authentication [10], reliable file transfer guarantees, and the required interface to the NCAR mass storage system. File staging is straightforward. Input files used by the Biome-BGC model are downloaded from the DataMover server at NCAR, and as the simulations finish, the generated output files are moved from CU to NCAR.

Implementation Experiences and Discussion

The Grid-BGC prototype implementation currently provides a fully functional grid service, grid client, JobManager daemon, and data transfers through the use of DataMover. At the present time, the prototype only executes Grid-BGC jobs, but we are working to increase the robustness of the software for application to similar projects. To date, this project has provided us with several valuable experiences while attempting to produce a working system that appeals to scientists, administrators, and software developers. In particular, we have ensured that the Grid-BGC framework fits in a managed security environment, provides reliability features to support execution on commodity cluster equipment, and is extensible so that its components are useful for future projects.

NCAR Security and Auditing Requirements

As a large government computing facility, NCAR provides computing resources to both internal projects and community users. Maintaining data security and auditing for charging purposes is required of all systems implemented at NCAR. The Grid-BGC solution is designed to meet these NCAR security and auditing requirements while functioning in a Globus Grid-based environment.

In a traditional grid environment, users authenticate with servers only using public key certificates. Because NCAR limits access to the mass storage system to users possessing a NCAR-issued “gatekeeper” account, our users must establish an account with NCAR and then use this username and password to authenticate with our portal. We internally generate a Grid-BGC certificate for all of our users. When a user logs in to the portal, their Grid-BGC certificate is used to instantiate a proxy that is uploaded into a MyProxy [5] server for later retrieval and used to contact tile processing grid services.

The authentication scheme on the cluster providing the grid service is intentionally simple to reduce administration overhead. All Grid-BGC user certificates are mapped to one UNIX user account. When a user submits a simulation request, the portal authenticates with the grid service using the user’s certificate. The grid service merely

stores the simulation request in a database, and the JobManager daemon then runs the simulations under the auspices of the service account. At no time does the user actually have possession of their internal Grid-BGC certificate, so they may not connect to a compute cluster directly but must use the portal interface.

In addition to the portal contacting the cluster running the grid service, it is also necessary for the JobManager daemon on the cluster to contact the NCAR mass storage system to download and upload data. Because NCAR requires complete user-based accountability, the daemon running under a service account must impersonate the user who submitted the request. To do this, we have a job request contain information about the portal's MyProxy server and the user's current stored proxy certificate. When the daemon must authenticate with the mass storage system, it first retrieves a copy of the user's proxy from the MyProxy server and uses these credentials for the data transfer.

Reliability and Fault Tolerance

Engineering fault tolerance into the Grid-BGC system is essential in our distributed grid environment. While the users of the CU cluster enjoy an uptime usually measured in months, during the course of Grid operations the end-to-end system is surprisingly prone to problems. We must cope with scheduled downtime – NCAR actually shuts down their entire facility once or twice a year for physical plant maintenance – as well as anticipated transients and genuine errors. Our software distinguishes between transients and errors so that users are not bothered with cryptic messages when a solution must be postponed due to the temporary unavailability of a required resource.

To facilitate fault tolerance, all cluster-side components including the grid service, the JobManager daemon, the data transfer system, and the models are arbitrarily restartable. The grid service is stateless and only performs atomic database transactions, so it may be restarted at any time. The remainder of the fault tolerance is built into the JobManager daemon.

Despite our best efforts, the CU cluster is still occasionally subject to node reboots with little or no warning, power failures, and students who ignore system administrator threats and circumvent the job scheduler to run code that is capable of causing kernel panics. The JobManager daemon monitors and controls the data transfer processes and simulation batch queue jobs. If, for any reason, a data transfer of a simulation job fails without completing successfully, the JobManager can restart it. If a job fails repeatedly, the system is presumed to be operating in a failure mode, and jobs are held for administrator intervention. Finally, the daemon itself maintains persistent state information in a database and all management system iterations are atomic. In the event of a system problem, the daemon may be stopped immediately. When it is restarted, the queue history is analyzed to determine if running jobs completed successfully, jobs are finalized or restarted as appropriate, and everything resumes normally. The Grid-BGC grid service provides "submit and forget" tile processing capabilities.

Expandability to Other Projects

The Grid-BGC software is designed to reliably execute the data transfers and models under its control within a completely flexible framework. Thus, the model it is running may be changed at any time. In addition to running the terrestrial ecosystem model integral to Grid-BGC, the Grid-BGC JobManager can be configured to run unrelated software applications among cooperating grid sites. We are also examining the possibility of extracting the Reliable Job Execution Service, a software component developed as part of the Grid-BGC Job Manager, and making it available by itself as part of a Grid middleware initiative.

Future Work and Conclusions

Work is underway to turn our Grid-BGC prototype into a fully functional system capable of running end-to-end carbon cycle simulations for the BGC user community. We believe that the entire system, including the user interface portal, will be fully operational by June 2005. At that point the system will be sufficiently developed to introduce climatologists as beta users.

After demonstrating full integrated functionality with our development cluster, we intend to expand the system to involve other clusters available via our grid. The first step is to provide the Grid-BGC tile processing service on other clusters under our control at the university and NCAR. A grid metadata publication and discovery service will be used to maintain clusters that are available to run tile simulation jobs and new simulations will be dispatched to clusters with the shortest anticipated turnaround time. The second step is to allow other collaborative institutes to instantiate their own tile processing services. In this case, users with their own clusters will be allowed to specify that their simulation jobs be run on their dedicated hardware instead of our shared resources.

One substantial component of Grid-BGC is data storage and transfer. We presently use DataMover to transmit data from one site to another. While DataMover maintains its own caching capabilities, it may be useful to analyze the operation of the Grid-BGC system in a production mode to determine if certain files should be replicated instead of transferred. This analysis will not be possible until the system is being used to run real science-based simulations instead of our test job collection.

Grid-BGC demonstrates an end-to-end system prototype leveraging Grid technologies to distribute a scientific application seamlessly across organizational boundaries. Our use of the Globus toolkit allows us to access NCAR datasets while running the computationally-intensive software on remotely administered commodity clusters. Overall, Grid-BGC provides a cost-effective, end-to-end solution for terrestrial ecosystem modeling through a straightforward and simple interface. As we have engineered the Grid-BGC execution framework to be as extensible as possible, we hope to apply our software solution for use in other similar applications.

Acknowledgements

University of Colorado computer time was provided by equipment purchased under DOE SciDAC Grant #DE-FG02-04ER63870, NSF ARI Grant #CDA-9601817, NSF sponsorship of the National Center for Atmospheric Research, and a grant from the IBM Shared University Research (SUR) program. NASA has provided funding for the Grid-BGC project through the Advanced Information Systems Technology Office (NASA AIST Grant #NAG2-1646) and the Terrestrial Ecology Program.

References

1. Allcock, B., Bester J., Bresnahan, J., Chervenak, A. L., Foster, I., Kesselman, C., Meder, S., Nefedova, V., Quesnal, D., Tuecke, S. Data Management and Transfer in High Performance Computational Grid Environments. Parallel Computing Journal, Vol. 28 (5), May 2002.
2. Eerola, P., Kónya, B., Smirnova, O., Ekelöf, T., Ellert, M., Hansen, J. R., Nielsen, J. L., Wäänänen, A., Konstantinov, A., Ould-Saada, F. The NorduGrid Architecture and Tools. Proceedings of Computing in High-Energy and Nuclear Physics (CHEP 03), La Jolla, California, March 2003.
3. Globus. The Globus Project, 2004, http://www.globus.org/A
4. Kacsuk, P., Goyeneche, A., Delaitre, T., Kiss, T., Farkas, Z., and Boczko, T. High-level Grid Application Environment to Use Legacy Codes as OGSA Grid Services. Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID 2004), Pittsburgh, USA, 8 November 2004.
5. Novotny, J., Tuecke, S., Welch, V. An Online Credential Repository for the Grid: MyProxy. Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
6. Sim, A. J. Gu, A. Shoshani, V. Natarajan. DataMover: Robust Terabyte-Scale Multi-File Replication over Wide-Area Networks. Proceedings of the 16th International Conference on Scientific and Statistical Database Management, 403, 21 June 2004.
7. Thornton, P.E., S.W. Running, and M.A. White. Generating surfaces of daily meteorological variables over large regions of complex terrain. Journal of Hydrology, 190: 214-251, 1997.
8. Thornton, P.E., S.W. Running. An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. Agricultural and Forest Meteorology, 93: 211-228, 1999.
9. Tsaregorodtsev, A., Garonne, V., and Stokes-Rees, I. DIRAC: A Scalable Lightweight Architecture for High Throughput Computing. Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID 2004), Pittsburgh, USA, 8 November 2004.
10. Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L., Tuecke, S. Security for Grid Services. The Twelfth IEEE International Symposium on High-Performance Distributed Computing, June, 2003.