SCIENTIFIC COMPUTING ON THE GRID

Tomorrow's dynamic applications will require computational grids

GABRIELLE ALLEN, EDWARD SEIDEL, AND JOHN SHALF

Computer simulations are becoming increasingly important as the only means for studying and interpreting many different processes, from engineering design such as modeling the Space Shuttle's heat shields during reentry into the Earth's atmosphere, to complex processes in nature such as the collision of two black holes described in the accompanying text box entitled "Colliding Black Holes."

Yet the scope and accuracy of these simulations are severely limited by available computational power, even using today's most powerful supercomputers. As we endeavor to simulate the true complexity of nature, we will require much larger scale calculations than are possible at present. We can break through these limits by simultaneously harnessing multiple networked supercomputers running a single massively parallel simulation to carry out more complex and high-fidelity simulations. Such endeavors require development of new latency-tolerant algorithms, as well as new code frameworks like Cactus (see the accompanying text box entitled "Cactus Code"). The code framework must be capable of fault tolerance and dynamic reconfiguration in response to rapidly changing computational resource conditions.

In addition, the modeled phenomena themselves may be highly dynamic, with computational demands increasing—or decreasing—-by orders of magnitude over the lifetime of a simulation. One numerical technique, adaptive-mesh refinement, automatically places higher resolution computational grids in areas requiring additional resolution, dramatically changing the resource requirements of a simulation by many orders of magnitude. The simulation program must interact closely with its computational resources to be able to reconfigure to match the changing computational load, or even seek out new resources if the current resources prove insufficient.

Now imagine intelligent computer programs with the ability to seek out and acquire/release computational resources around the world, based on their current needs and the status of the resources and networks connecting them. In such a world, a simulation may begin on one machine, redistribute itself across multiple machines as required (see the accompanying text box entitled "Distributed Computing"), and even move itself or spawn subtasks from one group of machines to another, as described in the accompanying text box entitled "Migration: The Cactus Worm." Computer processes will be able to autonomously adapt, taking advantage of the different resources they need. When computational processes are endowed with the ability to be

Gabrielle and Edward are researchers at the Max-Planck-Institut füer Gravitationsphysik in Germany. John is a researcher at the Lawrence Berkeley National Laboratory in California. They can be contacted at allen@aei-potsdam.mpg.de, eseidel@aei-potsdam.mpg.de, *and* jshalf@lbl.gov, *respectively.*

NEW!

Software Protection Electronic Licensing

Wrap and Secure Applications
WITHOUT CODING

- Software-only protection with highest levels of encryption and security technology
- End-user license portability
- LIcensing for executables and components including Java, DLL, and Visual Basic scripts
- Automated license delivery with transparent end-user installation
- System database and license management for installed base
- Scalable and upgradable

Your software is new. Why settle for 15-year-old security / licensing techniques? Check out the leading edge solutions from Vigilant today!

VIGILANT
SYSTEMS, INC.

800-747-1419 or 480-281-6201
info@vigilant-systems.com

www.vigilant-systems.com

aware of both their needs and the computational world around them, and when this computational world—known as the grid—is endowed with the ability to support such dynamic applications, we will have reached an entirely new stage in computing. Once scientists and engineers begin to consider that the grid is their computer, rather than simply a disparate collection of unconnected machines, they will begin to rethink their algorithms and application paradigms to take advantage of the advanced capabilities it provides. They will simulate processes with a completely new level of realism, and integrate experimental and theoretical work, in real time, in ways never before possible. In this article, we'll describe the emerging grid world, concentrating on the type of dynamic applications that are now being developed to take advantage of it.

A Brave New World

The advocates of grid computing promise a world where large shared scientific research instruments, experimental data, numerical simulations, analysis tools, research and development, as well as people, are closely coordinated and integrated in "virtual organizations." This integration will be accomplished through web-based portals, woven together into modular wide-area distributed applications. One hypothetical scenario in astrophysics illustrates this. Gravitational wave detectors will rely on results from large-scale simulations for understanding and interpreting the enormous amounts of experimental data they collect. The grid infrastructure is used both to share these expensive and centralized resources among many scientists, as well as to integrate experimental data feeds with the simulation codes necessary to analyze them. For instance, the GEO600 detector in Hanover detects an event characteristic of a black hole or neutron star collision, supernova explosion, or some other cosmic catastrophe. Astronomers around the world are alerted and stand by ready to turn their telescopes to view the event before it fades, but the location of the event in the sky must first be found. This requires a time-critical analysis of a number of templates created from full-scale simulations.

In a research institute in Berlin, an astrophysicist accesses the GEO600 portal and, using the performance tool, finds the resources required for cross-correlating the raw data with the available templates. The brokering tool finds the fastest affordable machines around the world. Merely clicking to accept the portal's choice initiates a complex process by which executables and data files are automatically moved to these machines by the scheduling and data management tools. Then the analysis starts.

Twenty minutes later, on his way home, the astrophysicist's mobile phone receives an SMS message from the portal's notification unit, informing him that more templates are required and must be generated by a full-scale numerical simulation. He immediately contacts an international collaboration of colleagues who are experts in such simulations. Using a code composition tool in their simulation portal, his colleagues assemble a simulation code with appropriate physics modules suggested by present analysis. The portal's performance prediction tool indicates that the required simulation cannot be run on any single machine to which they have access. The brokering tool recommends that the simulation be run across two machines, one in the U.S. and the other in Germany, that are connected to form a large enough virtual supercomputer (see the "Distributed Computing" box) to accomplish the job within the required time limit. The simulation begins, and after querying a grid information server (GIS), decides by itself to spawn off a number of time-critical template generating routines, and to run asynchronously on various other machines around the world. An hour later, the network between the two machines degrades and the simulation again queries the GIS, this time deciding to migrate to a new machine in Japan while still maintaining connections to the various template generators at other sites (see the "Cactus Worm" box). All the while, the international team of collaborators monitor the simulation's progress from their workstations or wireless devices from an airport (where several team members happen to be), visualizing the physics results as they are computed. The template data are assembled and to sent the GEO600 experimenter in Germany for analysis. The entire process, which could not be performed on any single machine or at any supercomputing site available today, takes only a few hours.

A Grid Application Environment

The scenario illustrated in Figure 1 heralds a future that supports the seamless connection of people, resources, simulations, and experimental devices. Such an environment depends first upon the construction of a robust infrastructure of fundamental services for grid computing. This is a challenging, but exciting problem as this infrastructure must somehow smooth out inhomogeneities between different machines, different security policies, different operating systems, different filesystems, and the like. To be acceptable for real use, the infrastructure must enforce trusted security and be fault tolerant to an extreme; that is, able to recover gracefully from the failure of any single component. The underlying support infrastructure for the grid is being built incrementally from low-level services that are standardized at the protocol level (see the Global Grid Forum, http://www.gridforum.org/; the Applications Research Group, http://www.zib.de/ggf/apps/; and Globus, http://www.globus.org/). This ensures platform, vendor, and interorganizational neutrality for fundamental services such as security, information services, remote job execution, and I/O-enabling key grid features like single sign-on security that works across the boundaries of multiple cooperating organizations. This also includes low-level tools for moving files between machines or data storage archives, for executing commands on remote machines, and for obtaining information about resources such as processor speeds, network bandwidths, allocations, and scheduling possibilities.

As this low-level infrastructure is being deployed, the grid community faces a difficult task in bridging the gap between such low-level services and the high-level needs of the virtual organizations and the scientific community at large. For application users and developers, the grid environment must go much further, providing fundamental and easily incorporated building blocks for adding grid capabilities to code and tools. Finding appropriate means to control applications that have components that are distributed worldwide and package them so as to hide their complexity is a difficult and challenging problem.

Portals are currently a very popular method for deploying these grid applications. Portals are an application delivery mechanism that packages the myriad distributed components of a grid application under a single centrally accessible application server. The centralization is necessary to mediate scheduling and control of the distributed resources that comprise a virtual organization. Web-based portals have become the most popular implementation paradigm because of its pervasiveness, platform neutrality, and because the Web so closely matches the topology required for distributed resource management. These web-based portals have already been deployed successfully for a number of applications such as the NPACI HotPage grid computing portal, http://hotpage.npaci.edu/; see also the "The Astrophysics Simulation Collaboratory," http://www.ascportal.org/. A number of frameworks and components for Portal Development have also cropped up including the CoG Toolkits (http://www-fp.globus.org/cog/); GPDK (http://www-itg.lbl.gov/Grid/projects/GPDK/); NPACI GridPort (http://gridport.npaci.edu/); and even commercial vendors like Sun (http://www.sun.com/gridware/).

But the grid is not just about software and computers. The virtual organizations around which we build grids are composed of both machines and people who must share information in order to function as a single body. Therefore, grid application servers, like portals,

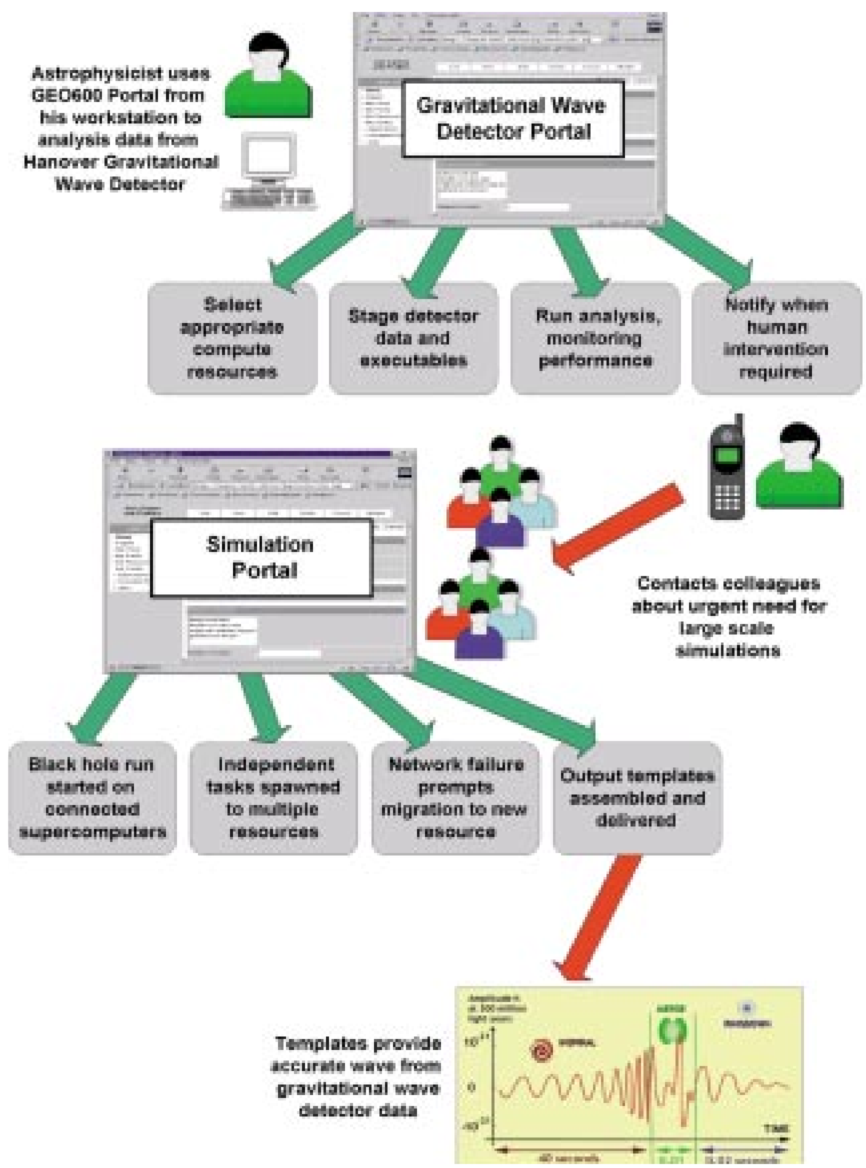

Figure 1: *Scenario depicting the seamless connection of people.*

must embed mechanisms that support both synchronous and asynchronous collaboration. The Access Grid (http://www.accessgrid.org/) is a grid application that is focused entirely on synchronous (online) collaboration through audio/video conferencing on the Multicast Backbone (http://www.lbl.gov/Web/MBONE.html). The ASC portal allows users to collaboratively assemble codes, start-stop simulations, and manipulate and view the data they create, and provide a repository of simulation components that can be shared with all colleagues of their virtual organization. Even the Cactus Code embeds a web server in every running simulation code. After providing the appropriate authentication, this web interface enables collaborating teams of researchers to write messages to each other about the running code, steer code parameters like the frequency at which the code writes data to disk, and even start and stop the code if it has gone awry. We have only begun to explore the collaborative interface possibilities enabled by grid infrastructure.

These collaborative interfaces will likely be a huge area of growth in the development of the grid. For users running applications in such a dynamic environment, new levels of pervasive information delivery must be incorporated in order to keep pace. For example, devices ranging from workstations to mobile laptops to PDAs, mobile phones with SMS messaging, and even Linux-based wristwatches will be part of a messaging fabric that should make it possible to check on the status of your virtual organization made up of myriad colleagues, resources, software, and simulations (see "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," by I. Foster, C. Kesselman, and S. Tuecke, *International Journal of Supercomputer Applications,* 2001, http://www.globus.org/research/papers/anatomy.pdf). As these devices will also have rather significant computing power, they may even be useful in powering the simulations themselves! Such new capabilities will require not only an advanced grid

Colliding Black Holes

Black holes, whose behavior is described by Einstein's General Theory of Relativity, are not just the stuff of science-fiction stories. Astrophysicists predict that black holes, formed from the cataclysmic collapse of stars many times more massive than our own sun, will actually be able to be observed and studied from Earth. To do this, however, we will require giant laser interferometric devices that can detect the gravitational waves black holes emit, in much the same way that telescopes already observe optical, X-ray, and gamma-ray emissions from black holes.

Gravitational waves, predicted by Einstein nearly a century ago, are believed to travel through the universe at the speed of light, barely interacting with the matter through which they pass. They are extremely hard to detect, so difficult that they have never been seen directly. Even with the current generation of interferometers, some of which are over four kilometers long and offer our first real hope for observing waves, their detection and interpretation will rely on accurate results from numerical simulations of the astrophysical processes that generate them. Such simulations are being carried out by an international group of scientists at the Max-Planck-Institute for Gravitational Physics in Germany, using supercomputers located in both the U.S. and Germany. Here, scientists have developed ground-breaking techniques for performing simulations of the inspiraling collision of two black holes. The scientists are aided in their physics by Cactus (http://www.cactuscode.org/), a sophisticated framework for parallel computations, which was specifically designed to enable such large-scale, computationally intensive, collaborative simulations. Figure 2 shows results from one such simulation, requiring over 100-GB of memory on a 256 processor SGI Origin 2000 supercomputer at NCSA. A large black hole, near the center of the figure, has just been formed from two smaller holes, just visible inside. The burst of gravitational waves is shown emanating from the collision. Although this is one of the largest and most sophisticated simulations performed to date, to simulate the entire inspiral and merger with the desired resolution would require significantly more computing capacity than is available on any existing supercomputer.

—G.A., E.S., and J.S.

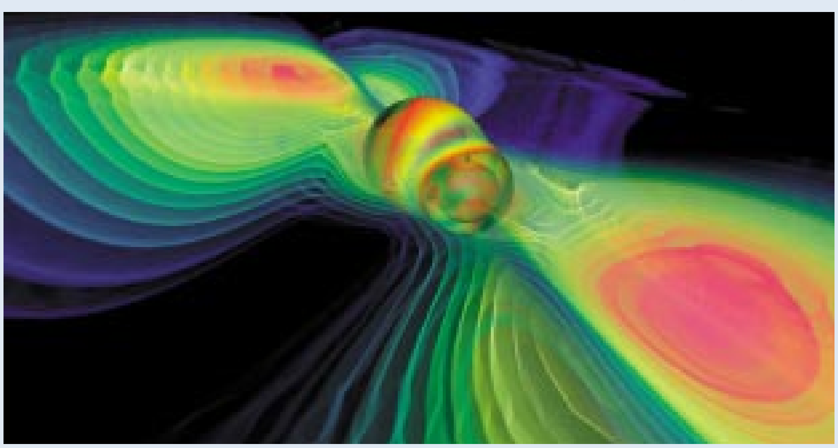

Figure 2: *Black hole simulation.*

infrastructure to harness the collective power of many devices, they will enable new paradigms for computing in science and engineering.

New Application Paradigms to Enable Science on the Grid

Higher level tools will open up new paradigms for applications to exploit all manner of devices to perform computations of scale and throughput never before possible. A common misconception regarding supercomputing on the grid is that it will replace current tightly coupled supercomputer designs with loosely coupled clusters of widely distributed machines. Actually, the grid expands supercomputing capabilities to problems that are traditionally underserved by existing supercomputing architectures. Rather than cannibalizing the market for large-scale machines, it actually expands their use to problems that benefit from integrating across multiple sites and offering novel HPC capabilities. This requires study of how the grid can enhance existing supercomputing paradigms, investigation of nontraditional application paradigms and development of APIs that enable scientists and engineers to easily express these problems in a grid-enabled environment. We'll discuss developing new grid computing paradigms later.

Enhanced Efficiency for HPC Centers: The largest HPC sites are increasingly faced with the issue of balancing the needs of capacity computing (e.g., huge memory) with capability computing (e.g. high throughput). The cost-per-megaflop of these massive centralized resources is much greater than that of smaller facilities, but they often find themselves clogged with small-scale jobs that require high throughput and little capacity. The capacity requirements of these tiny jobs inhibit their ability to run computations that fully exploit the capability of the resource as a single massive machine. Therefore, HPC Centers look to the grid as a means of creating a Virtual Machine Room that integrates the environments of many smaller partner sites. Simulation codes are scheduled on computers that are appropriately sized for their resource requests, thereby satisfying the capacity demands of users and opening up the leading edge HPC sites to deliver on the capability requirements of the largest applications.

Enabling Bigger Science: At the opposite end of the spectrum are a small number of numerical scientists who have resource requirements that exceed the capacity of any one supercomputing site. However, if they could simply combine together the capabilities of multiple supercomputers into a single metasupercomputer, they could bring their science to a new level.

Migration: The Cactus Worm

In the same way that nomadic tribes move their entire belongings to follow the best opportunities for hunting and farming, we believe that grid-aware applications should seek out and exploit the most appropriate resources to be found, moving on as current resources are depleted, or when new possibilities present themselves. Such intelligent migratory technology combines many basic grid building blocks, including information gathering, resource discovery, checkpointing, authentication, and file transfer. The implementation of this capability puts abstract concepts of artificial life and autonomous agents to work for the scientist's toolbox.

One example implementation of migration technology has already been prototyped using the Cactus Code (see http://www.cactuscode.org/Papers/CCGrid_2001.pdf.gz). Here, the addition of a single module, called the "Cactus Worm," endows any Cactus application with the ability to migrate between a set of resources; see Figure 3. The Cactus Worm was demonstrated running on the resources of the EGrid testbed at SC2000, and has now been enhanced to include better fault tolerance, robustness, and decision making.

—G.A., E.S., and J.S.

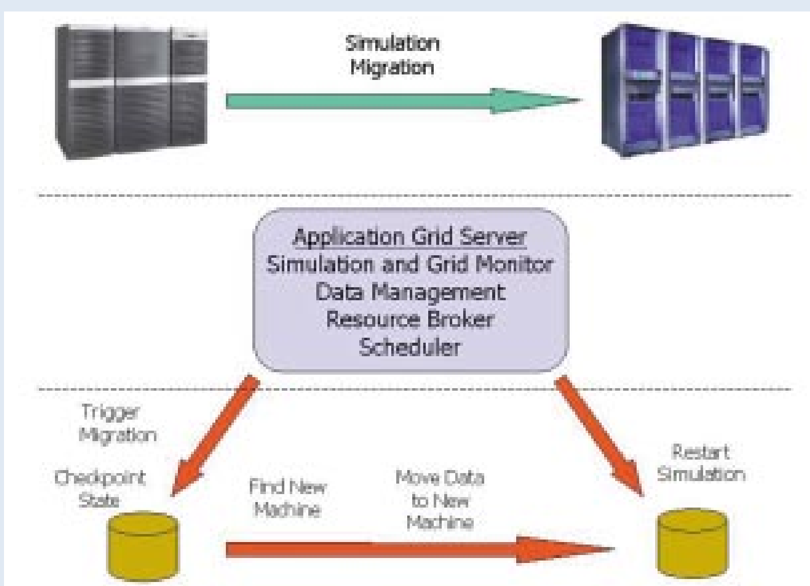

Figure 3: *Worm scenario.*

Distributed Computing

Distributed simulations harnessing the resources of multiple supercomputers have been performed in experimental environments for several years (see "Distributed Simulations with Cactus," http://www.cactuscode.org/Projects/Distributed/Distributed.html). Running these simulations required a large amount of human investment, simultaneously reserving machines and networks and setting up the executables, parameter files, and initial data correctly in different locations. Early on, such experiments showed that such distributed simulations were possible, but the computational overhead of the wide-area network (WAN) typically lead to large performance degradations during simulation.

Advances, both in infrastructure and simulation codes, are leading to dramatic improvements in both coordinating simulations and in the performance they can obtain. A recent set of experiments (see "Supporting Efficient Execution in Heterogeneous Distributed Computing Environments with Cactus and Globus," by G. Allen, T. Dramlitsch, I. Foster, N. Karonis, M. Ripeanu, E. Seidel, and B. Toonen, *Proceedings of Supercomputing 2001*, 2001) demonstrated the use of new, dynamically adaptive techniques that allow even tightly coupled simulations to run efficiently across remote machines using only standard networks. Combining an IBM SP2 machine at the San Diego Supercomputing Center with an array of Origin 2000 machines over 1500 miles away at the National Center for Supercomputing, a group of scientists from the Albert Einstein Institute in Germany and the University of Chicago in the U.S. ran the largest numerical relativity simulation to date across 1500 processors. The new techniques that enabled scaling of up to 88 percent for this tightly coupled simulation included compressing data before it was sent across the WAN to maximize bandwidth usage and performing additional calculations to reduce the frequency of communications and offset latency effects. With such techniques applied adaptively to the changing environment, simulation performance can be dramatically increased in extremely dynamic grid environments with little or no user intervention.

—G.A., E.S., and J.S.

Alternatively, if a single machine has the theoretical capacity but is unavailable in whole, they might be able to harness parts of multiple machines to carry out a single-capacity simulation now, rather than waiting weeks until a single resource can be completely dedicated to their calculation. The efficiency of such jobs will likely never approach that of a single tightly coupled supercomputer, but new adaptive computational techniques have improved performance immensely over the past few years (see the text box entitled "Distributed Computing"). The benefits of enabling scientific breakthroughs many years in advance far outweigh arguments about the reduced efficiency of such computations.

Managing Dynamic Numerics: The typical batch systems of supercomputers are designed to support problems that stay the same size throughout their run. However, new classes of simulation problems require a method to seek out new computational resources on a regular basis as the job is running (see the text box entitled "Migration: The Cactus Worm"). For instance, a simulation may need to spawn subtasks to analyze the data it has produced or to focus on particular discrete events such as formation of the apparent horizon of a black hole (see sidebar on black holes). Another case is a simulation technique called "Adaptive Mesh Refinement," where the simulation automatically increases the resolution of a computational mesh in areas that require it as the simulation is running. The simulation designer could not possibly know *a priori* when or how much increased resolution is going to be applied by this automated refinement; something that could easily double the resource requirements of the simulation in a matter of minutes. Grid-enabled simulation codes, such as the Cactus Worm, can automatically seek out additional resources as they are running in order to avoid resource starvation or to guarantee completion within a specified time period.

Data-Intensive Computing and Integration of Observational and Simulation Data: Examples of such computational problems are data processing from large sensor arrays. SETI@Home offers a glimpse into the sort of new computational paradigms necessary to support this sort of research (not the traditional domain of centralized HPC resources). The grid will offer a common substrate for the application needs of large scientific exploration instruments such as electron microscopes, radio telescopes, and weather satellites. The grid not only provides a means to tap into distributed computational capacity to process the data feeds from these resources, there is considerable development in grid interfaces that enable researchers to manage and mine the many petabytes of data, access these expensive instruments from remote sites, and better collaborate with other researchers across the world that comprise their Virtual Organization.

At the lowest level, the grid permits uniform remote access to files by homogenizing the file access and retrieval mechanisms, thanks to uniform security. However, that does little to directly address the enormous data management issues faced by scientists. Consider data archives that are many petabytes in size and grow by terabytes a day. How does one manage, search, enforce access-control policies, or generally make sense of this quantity of data? Several Data grid efforts (for instance, http://www.eu-datagrid.org/, http://www.ppdg.net/, http://www.griphyn.org/) are actively developing the infrastructure necessary to manage the massive datasets produced by detectors in particle accelerators, genomics, star surveys, electron microscopes, and gravitational wave detectors. Data grids manage data storage that spans multiple sites and is potentially fed by independent sets of instruments that cross international boundaries. Data grids are developing software that provides intelligent data migration across these sites and rapid/efficient data searching and subset retrieval on petabyte data stores. Furthermore, one needs to integrate the information from these observational instruments with the simulation codes that are used to analyze them.

Parameter Searches, NP-Complete, and NP-Hard Problems: There are many problems that require calculations that can only be solved by directly analyzing every possible discrete possibility (NP-Complete) or even all possibilities in a continuous parameter space (NP-Hard). Such problems are of critical importance to drug design and the understanding of the map of the human genome. For example, a protein folding prediction is where one must search an infinite (NP-Hard) search space using a tree-like search pattern where each leaf of the tree requires a resource-demanding simulation. Typical strategies to solving this problem involve reducing the parameter space using tree-search/tree-pruning algorithms, simulated annealing methods, genetic algorithms, and even interactively guided search spaces. It is simply not cost-effective to use tightly coupled supercomputing systems to attack these kinds of problems. Grid computing offers a much more appropriate computational infrastructure for solving such a high-throughput, loosely coupled computational problem.

The aforementioned application scenarios are only those that are currently benefiting from the nascent grid infrastructure. There are already active development efforts by our research group (as described in the sidebars), commercial entities, and academic and government institutions worldwide that are building production implementations of these scenarios. Currently, we are limited both by our imagination and limited application-oriented programming interfaces for grids. As the software infrastructure improves, application developers will be able to embed calls to grid-enabled routines directly in the applications, which will trivially enable new grid computing operations. Just as the emergence of the Web served as an engine for creative and novel uses for network infrastructure, the emergence of mature grid infrastructure and accessible grid programming interfaces will likely open up new categories of scientific applications that we haven't even thought of in the bounds of this

Cactus Code

The Cactus Code (http://www.cactuscode.org/) embodies a new paradigm for development and reuse of numerical software in a collaborative, portable environment, applicable to all areas of computational science and engineering. A freely available, open-source toolkit, Cactus lets one extend traditional single-processor code developed on laptop computers (using common languages like C, C++, Fortran 77, and Fortran 90) into full-blown parallel applications that can run on virtually any supercomputer. Cactus also provides access to myriad computational tools, such as advanced numerical techniques, adaptive mesh refinement, parallel I/O, live and remote visualization, and remote steering. The various computational layers and functionalities, as well as different scientific modules, can be selected at run time via parameter choices.

The name "Cactus" comes from the initial design principle of a module set (the thorns) that can be plugged easily into a basic kernel (the flesh). The flesh has evolved into a metacode with its own language to enable dynamic assembly of many different application codes from a set of thorns. The flesh controls how the thorns work together, coordinating dataflow and deciding when any routine should be executed. The modular approach enables scientists to design, use, and add thorns that work together through the flesh API.

In emerging research areas, it is not always clear what techniques will be best suited to solving a given scientific problem. Further, both the software and hardware used to run advanced scientific simulations change rapidly. These factors, combined with the complexity of the work undertaken, result in huge simulations that require intensive and often widely distributed collaboration. To support such work requires an environment like the one Cactus provides: a flexible, extensible, and portable computational infrastructure into which different components from different communities can be plugged. For example, while a physicist may develop a new formulation of Einstein's equations, a numerical analyst may have a more efficient algorithm for solving a required elliptic equation; a computer scientist may develop a more efficient parallel I/O layer; a software team may develop a new paradigm for parallel programming; a hardware vendor may develop a more efficient parallel-computer architecture; and so on.

—G.A., E.S., and J.S.

The Globus Project

The Globus Project (http://www.globus.org/) is developing the fundamental technologies needed for building a functioning grid and supporting grid applications. Many different Globus components, including resource and data-management mechanisms, security services, and information services are already deployed in different large-scale testbeds around the world, including the NASA Information Power Grid, DOE Science Grid, and the European DataGrid. Globus also provides a communication, security, and job-scheduling infrastructure necessary to support massive simulations that run across multiple supercomputers (metacomputing).

A key component of the Globus Toolkit, the Grid Security Infrastructure (GSI), greatly simplifies the use of multiple machines. Many computational scientists have accounts on more than one resource, and with each resource comes the possibility of differences in login name, password, security policies, and even incompatible authentication mechanisms (ssh or Kerberos5). For many users who hold accounts at multiple organizations, simply accessing a machine becomes a logistical nightmare. The Globus GSI provides secure, single sign-on, run-anywhere authentication that works across organizational boundaries. The GSI is already being used to enhance existing command-line tools like FTP and remote-shell interfaces. The GSI APIs are also enabling nontraditional applications like grid portals and advanced simulation frameworks such as the Cactus Worm to take advantage of this interorganizational security infrastructure to access resources that span the grid.

—G.A., E.S., and J.S.

article. This is only the beginning of the journey.

Building the Future

The basic capabilities we have described open the door for a wide range of new application scenarios. A user could connect into a colleague's running simulation through a portal. Then, at the press of a virtual button, he could clone a copy running on a different machine with a different configuration. He could also downsize the cloned simulation, investigating and analyzing only the features he is interested in. When he runs a simulation, it will automatically be staged on the most appropriate machine, taking into account the resources required, the timeframe required for a solution, and the desired cost. As the simulation runs, independent tasks will be continuously farmed to different machines, the results being gathered together in a single location for the user at completion.

These scenarios and technologies may not be so far from reality. Groundbreaking research from computational science and application groups over the last decade have developed not only a maturing infrastructure for grid computing, but have demonstrated its feasibility and possibilities. Progress has been so good that numerous major projects have recently been funded, in both Europe and the U.S., targeted at developing and using grids in different application communities. For instance, see Grid Adaptive Development Software (GrADS) at http://www.isi.edu/grads/; the GridLab Project, http://www.gridlab.org/; the GriPhyN Project, http://www.griphyn.org/; the European DataGrid Project, http://eu-datagrid.org/; the European Euro-Grid Project, http://www.eurogrid.org/; and the DFN Gigabit Project, http://www.zib.de/Visual/projects/TIKSL/.

One such project, which has just obtained funding from the EU Commission, seeks to enable exactly the scenarios described in this article, for general application programs and developers with little or no experience in grid programming. The GridLab Project (http://www.gridlab.org/) will build a Grid Application Toolkit to provide developers with these very building blocks. As suggested by the project's name, it aims to provide a grid environment or laboratory for users to both develop and test applications.

In addition, the Global Grid Forum (GGF; http://www.gridforum.org/) is the hub for the world-wide grid community of computational scientists, industrial partners, physicists, and engineers. The GGF seeks to ratify community standards for grid software and services, developing vendor- and architecture-independent protocols and interfaces. The GGF fosters and coordinates a growing number of working groups and research groups, each focused around a different aspect of grid computing. One of these, the Applications Research Group (http://www.zib.de/ggf/apps/) provides a link between the grid-aware application community and the developers of infrastructure and services, ensuring that new technologies are constructed in line with the needs and directions of the end users in this brave new world of computing.

Acknowledgments

We are indebted to many of our colleagues for helping to develop this vision of grid computing, especially Thomas Dramlitsch, Tom Goodale, Gerd Lanfermann, Thomas Radke, Andre Merzky, Werner Benger, Ian Foster, Carl Kesselman, Dave Angulo, Michael Russell, Greg Daues, Jason Novotny, and Jarek Nabrzyski. The development of Cactus has been enabled by support from many different sources, including the NSF, NCSA, the Max-Planck-Gessellschaft, the DFN, and Microsoft. ■

Return to Table of Contents
Next Article