2004 VGrADS Annual Report

I. Project Activities

The "Computational Grid," as described in *The Grid: Blueprint for a New Computing Infrastructure* and demonstrated by many proof-of-concept applications, promises to connect computers, databases, and people in a network to solve problems in scientific research and other diverse fields. However, the complexity, unreliability, and overhead of low-level operations in today's systems obscure the Grid's potential. The five-year Virtual Grid Application Development Software (VGrADS) project is attacking a fundamental part of this problem how to more effectively program these highly complex and dynamic systems. It is developing software tools that simplify and accelerate the development of Grid applications and services, while delivering high levels of performance and resource efficiency. This improved usability will greatly expand the community of Grid users and developers. In the process, VGrADS will contribute to both the theory and practice of distributed computation.

To address these aims, VGrADS is exploring, defining, and implementing a hierarchy of virtual resources and a set of programming models for Grid computing. It is conducting research in three key areas:

1. Virtual Grid (vgrid) architectures, enabling a separation of concerns between high-level services and the Grid's inherent complexity.
2. Programming models, compilers, component libraries, and tools supporting creation of Grid applications.
3. Core software technologies, including performance-efficient scheduling, fault tolerance, and economic models for resource management, allowing scalable Grid computations.

VGrADS is pursuing this agenda by collaborating with leading scientific applications to elicit key challenges, validate results, and disseminate technology. It will distribute software that it creates in open-source form for the research community. It will also build on its PIs' past successes in human resource development by using existing programs to attract and retain women and minorities in computational science.

During the current reporting period (10/1/03 5/31/04), VGrADS research focused on the three inter-institutional efforts described in the following sections: *Applications, VGrADS Programming Tools,* and *VGrADS Execution System*. Project publications and additional information can be found at http://www.hipersoft.rice.edu/vgrads. The project Web site includes a participants-only area where VGrADS researchers exchange drafts of project documents and other materials.

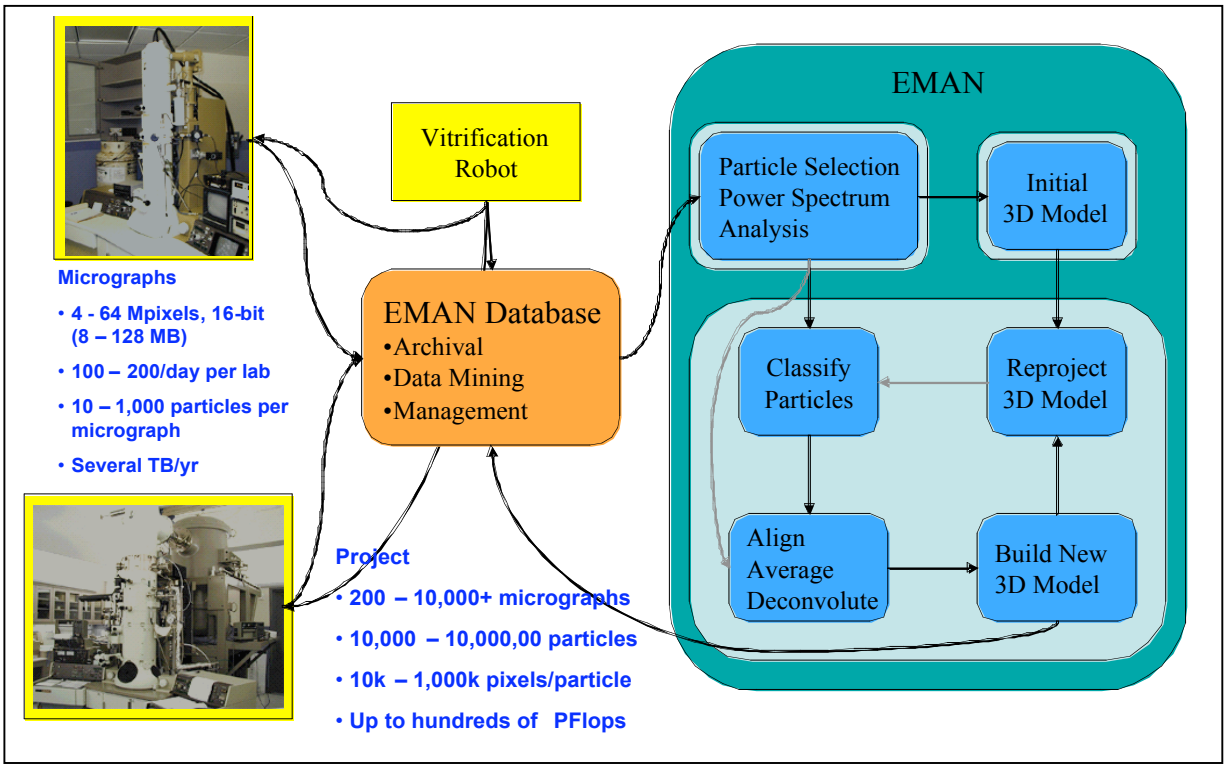
Project design and coordination during the current reporting period were enabled through weekly technical teleconferences involving researchers from each of the VGrADS sites, three PI teleconferences, a kickoff meeting at Rice for PIs and senior researchers (2/10 11/04), and communication via VGrADS mailing lists. In addition, subproject participants met on a regular basis to exchange ideas and develop research plans.

1 Applications (Rice, UCSD, UCSB, UH, UNC)

We have begun work on four Grid applications under VGrADS. In most cases, these applications were started under the GrADS project, but further development is driven by the needs and goals of the VGrADS project. In particular, we are deriving requirements for virtual grid (vgrid) functionality from the applications' needs. We summarize recent work on each application in the following four subsections.

1.1 EMAN

The Rice and UH VGrADS research efforts have focused on the molecular structure determination package EMAN, which we take as a model for workflow-style applications. EMAN is a package for Electron Micrograph Analysis developed within the National Center for Macromolecular Imaging at Baylor College of Medicine by Steve Ludtke, a senior researcher in Dr Wah Chiu's group. The package processes thousands to possibly tens of thousands of micrographs from electron microscopes iteratively in the determination of a macromolecular structure. Most of the computations are of a throughput nature consisting of fitting individual micrographs to a hypothesized 3-D structure. An improved hypothesized structure is then generated from the fits made, a computation that requires consolidation of all fits in a "tightly coupled" computation. The application and the iterative nature of the processing are illustrated below.



One aspect of making applications Grid aware is the creation of performance models for its components in order to make proper decisions for resource selection and scheduling of the components. The construction of performance models has been the focus of our efforts. One approach we have pursued (mostly at UH) makes use of a set of equations, the structure of which is derived from knowledge of the component's computational requirements and the dependence upon input variables such as micrograph sizes, number of micrographs, and variables controlling the processing of the micrographs. Model parameters are then determined from execution traces. We have also pursued (mostly at Rice) a more black-box approach based on instrumentation of object code, followed by benchmarking runs. The benchmark results are then used to create a polynomial model without explicit prior knowledge of the application structure. In both approaches, the objective is to separate application characteristics invariant with respect to architectures from those that are dependent. For instance, the number of floating point operations should be independent of the platform used for the processing, but the number of instructions will depend upon both that platform and the compiler used, as well as compiler options used. Similarly, memory access patterns should have a strong correlation to the application, while the number of cycles required will depend significantly on the memory system.

We have derived a model for the main EMAN components and made a first mostly manual validation of the model. We are now in the process of automating the model fitting and investigating the sensitivity of the results to compilers being used as well as the impact of compiler options being used. The main platforms targeted at the moment are Opteron and Itanium2 based clusters.

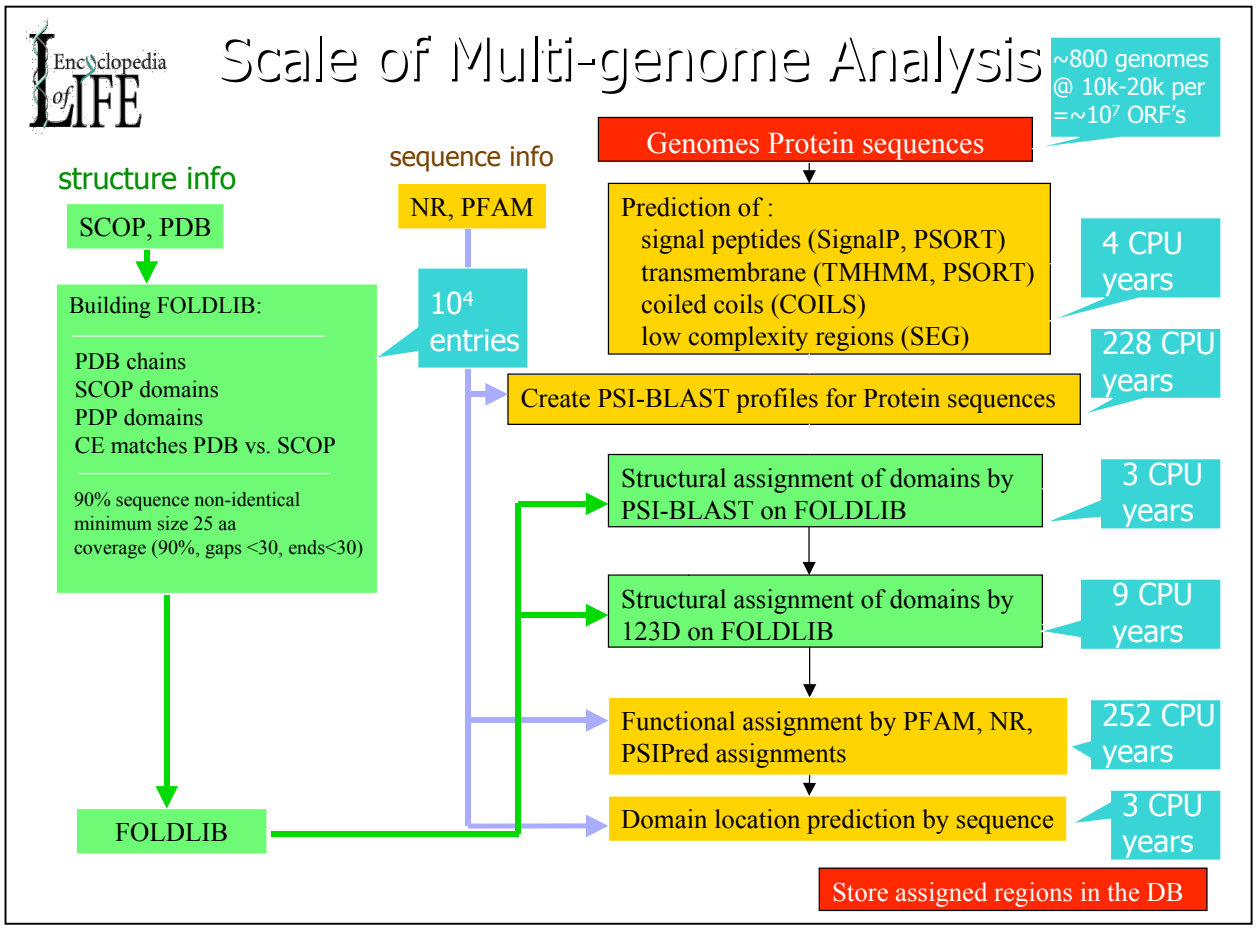
In addition, EMAN has served as the first test case for work on the new VGrADS binder, which is described in the Programming Tools section.

1.2 EOL

UCSD VGrADS work has focused on the Encyclopedia of Life (EOL), a collaborative global project designed to catalog the complete genome of every living species in a flexible reference system. It is an open collaboration led by the San Diego Supercomputer Center, and currently has three major development areas: (i) Creating protein sequence annotations using the integrated genome annotation pipeline (iGAP); (ii) Storage of these annotations in a data warehouse where they are integrated with other data sources; and (iii) A toolkit area that presents the data to users in the presence of useful annotation and visualization tools. In VGrADS we will focus on (i) because it is the major computational element. The key goal of (i) is to discover relationships across genomes, and thus involves extensive computation and access to databases that contain data derived from the iGAP processing of multiple genomes (dozens initially and ultimately hundreds). However, this coupling across genomes is achieved exclusively thru accesses and updates to these shared databases. In the future, (ii) may be of interest as well.

EOL is not a single code, but rather a script (iGAP) that glues together a number of well-known community software packages. These packages operate on input files ("sequence files") as well as on community databases (actually, flat ASCII files containing biological

sequences). Like EMAN, the script is essentially a workflow application performing a pipeline of operations on each input genome. The analysis we are concerned with is shown in the figure below, in somewhat idealized form.



The overall model for the EOL computation is then as follows: a set of independent jobs (one job per genome), where each job consists of independent sub-jobs (one sub-job per sequence), where each sub-job consists of a "chain" of the above 4 steps.

At the moment EOL is deployed on a Grid that aggregates AIX and Linux clusters in several institutions (see http://eol.sdsc.edu:8080/eol/resources.jsp). The databases are fully replicated and installed on each cluster (either in each local disk, or over a GPFS, maybe even NFS). The computation is controlled by APST (http://grail.sdsc.edu/projects/apst), which handles all logistics of application deployment (interaction with Globus, SGE, PBS, etc.), and the Biology Workflow Management System (BWMS), which was developed specifically for EOL. Essentially, EOL researchers submit a full genome for computation to APST, and APST schedules and runs all involved sequences through the iGAP steps. The major current limitation of this arrangement – one that we hope vgrids will help solve – is that the scheduler does not take account of data movement costs.

EOL was demonstrated in its current deployment at SC'03. A paper reporting on this deployment was accepted for publication and will be available soon. Status of ongoing computations can be seen on-line at http://eol.sdsc.edu:8080/eol/genomestatus1.jsp.

1.3 GridSAT

The UCSB VGrADS application work has focused on GridSAT, a parallel and complete satisfiability solver used to solve non-trivial SAT problems in a grid environment. The application uses a parallel solver algorithm based on Chaff to (attempt to) solve SAT problems of the form 'given a large, non-trivial Boolean expression, is there a variable configuration (and what are the variable values) which results in the expression evaluating to TRUE?' The system stands apart from other SAT solvers in the sense that it was designed explicitly to run in grid environments, and has built in intelligent parallelism scheduling mechanisms. As a result of this design, the system has been used to successfully and quickly solve several previously unknown problems by utilizing vast amounts of computational resources.

The fundamental algorithm of GridSAT is a parallelization of the Chaff SAT solver. Chaff is a highly optimized complete sequential solver based on the Davis-Putnam-Logeman-Loveland (DPLL) algorithm with many heuristic optimizations. GridSAT modifies Chaff to make it possible to divide the search space between many clients. In particular, both the sequential and parallel algorithms use a "decision" stack to keep track of progress. The initial decision stack contains only variable assignments that are definite. The algorithm heuristically selects a variable and a Boolean value to assign to it. This is called a decision and adds a new level to the decision stack. The first variable in this level will be the decision variable. Additional variable assignments are added to the current level as new values are deduced from previous decisions. The process continues until the algorithm runs into a conflict (i.e. a variable is deduced to be both true and false), at which point the algorithm backtracks by popping decisions from the stack. It terminates when either a solution is found (a satisfiable problem) or it deduces that no such solution exists (an unsatisfiable problem).

A major improvement to the DPLL algorithm is "learning", which enables the algorithm to deduce new clauses and store them in a local database. New clauses are generated after a conflict and are used in backtracking. The database of new clauses can grow indefinitely, but can be controlled by heuristically deleting some new clauses. (New clauses carry only redundant information that can be used to speed optimization. A key to GridSAT's success is integrating these learning heuristics with runtime information about the status of the Grid (or vgrid). In particular, it attempts to split the program onto additional resources (i.e. assigning sub-problems to other Grid nodes) only when needed since the problem is tightly coupled. It splits the problem if memory is exhausted on a node, choosing the best available resources and tailoring the split itself to make the best use of those resources. A relatively unique, application-driven scheduling mechanism does this using resource information gleaned from the execution system, particularly the MDS and NWS components. The splitting mechanism also allows nodes to exchange the learned clauses.

Satisfiability problems can be arbitrarily hard and may take long periods of time even using large resource pools. The duration on some of the problems we investigated ranges from seconds to weeks. During this time, planned and un-planned resource and network failures are frequent. GridSAT clients take checkpoints to recover from such failures. GridSAT clients are automatically restarted when a resource recovers and are assigned a sub-problem from the available list of checkpoints. GridSAT clients can be configured to take two types of checkpoints: Light or Heavy. Light checkpoints only involve saving the first level of the decision stack. This is on the order of the number of variables, which is small and may reach 10s of kilobytes. This kind of checkpoint is managed by the master so as not to be overwhelming. Heavy checkpoints, however, are much larger and are in the order of 100s of megabytes. Such files cannot be managed by a single host and are best stored in a distributed storage environment such as IBP. This solution also avoids having a communication bottleneck, which would exist if the checkpoints were stored at some central location.

GridSAT executions could take days or weeks, depending on the problem search space, resource pool, and resource attributes. By using dynamic internal scheduling techniques, individual client checkpoints, and ability to utilize new resources during the system's lifetime, the GridSAT application accomplishes the goal of keeping volatile grid resources busy until the application completes.

1.4 LEAD

In response to the need for a national cyberinfrastructure in mesoscale meteorology, the UNC VGrADS research team (in collaboration with researchers at UIUC and elsewhere) is creating an integrated, scalable framework -- known as Linked Environments for Atmospheric Discovery (LEAD) -- for identifying, accessing, preparing, assimilating, predicting, managing, analyzing, mining, and visualizing a broad array of meteorological data and model output, independent of format and physical location. The meteorology of LEAD is being developed under a separate ITR award (NSF #0315594); the VGrADS project collaborates with the project in the area of enabling Grid computations for this application.

A transforming element of LEAD is dynamic workflow orchestration and data management, which allows the use of analysis tools, forecast models, and data repositories not in fixed configurations or as static recipients of data, as is now the case, but rather as dynamically adaptive, on-demand, Grid-enabled systems that can (a) change configuration rapidly and automatically in response to weather; (b) continually be steered by new data; (c) respond to decision-driven inputs from users; (d) initiate other processes automatically; and (e) steer remote observing technologies to optimize data collection for the problem at hand. Toward these goals, LEAD research is focused on creating a series of interconnected, heterogeneous virtual IT "Grid environments" that are linked at several levels to enable data transport, service chaining, interoperability, and distributed computation.

VGrADS has only begun discussions with the LEAD principals on how to best collaborate. Some initial areas where we see possible joint work include

- Workflow Orchestration – the construction and scheduling of data sources including dynamic (real-time) sensor streams, simulation outputs and data mining. Orchestration

must include these constraints and use load monitoring to make decisions.

- Interaction and Control of Dynamically Adaptive Sensors – the protocols, command interfaces, and linkages with meteorological models and other tools needed for two-way adaptivity.

- Data Streaming – to support robust, high bandwidth transmission of multi-sensor data. LEAD's dynamic and real-time functionality will require more flexible content-based subscription approaches to data stream management and control, methods for asynchronous pull-based retrieval, and batching methods for relaying large volumes of archival data.

- Distributed Monitoring and Performance Estimation – to enable soft real-time performance guarantees by estimating resource behavior. To identify resource classes capable of satisfying performance expectations, and to verify that the resources behave as expected during ensemble execution, LEAD requires an extended set of Grid monitoring and performance estimation tools, as well as Grid services and dynamic orchestrations based on the Illinois Autopilot Grid toolkit and other previous GrADS monitoring work.

- Data Management – in support of the storage and cataloging of observational data, model output and results from data mining. LEAD requires grid support for the three cornerstones of management: data dissemination, over the network data storage (including cataloging, archival of processes, metadata, and ephemeral data), and user view maintenance (including the MyLEAD personalization of LEAD's computing and data-intensive research environment).

- Data Mining – to provide tools that enable users to glean insights from data and model output. The LEAD data mining capabilities will be implemented as sets of federated mining components capable of operating on multiple platforms as independent grid services, dynamically linked for analysis tasks.

- Semantic and Data Interchange Technologies – to enable use of heterogeneous data by diverse tools and applications. To allow orchestration of workflows involving both heterogeneous information from different data sources, and chains of disparate services, in a seamless, dynamic, automated fashion, LEAD will incorporate ontologies, interchange technologies, and other concepts being explored in the Semantic Web and Semantic Grid. Coupling this with grid computing technologies like VGrADS is a heretofore unexplored idea.

2 VGrADS Programming Tools (Rice, UH)

Work on tools has primarily consisted of two general thrusts. First, we have been refining the infrastructure that we developed under the previous GrADS project. Second, we have been interpreting the results of our experience for the purpose of helping to establish the design for both the virtual grid layer, reported on elsewhere in this document, and the new class of tools proposed to run on top of them.

With regard to the first of these thrusts, we have continued our work on the new binder mechanism that permits the execution of applications on heterogeneous resources. In a

nutshell, this new binding and execution strategy consists of preparing a script that will run on each of the resources on which an application is to be executed. Then the relevant script is executed on each resource, which results in compilation of the components and linking against preinstalled component libraries.

This last capability is a step toward the component machine that was described on the proposal. Components that are to be available at different Grid sites are advertised via the GrADS Information Service (GIS), which has been extended to include information about which component libraries are installed at each location in the GrADS testbed. This permits component installations to be taken into account in the scheduling process.

We have also continued work on workflow scheduling and performance model construction, which are described in Section 2.1. Much of this work has focused on the EMAN bio-imaging application, which is described in Section 1.1.

The tools project has also collaborated with the virtual grid group to define useful virtual grid abstractions that will permit tools to focus on simplified models of Grid resources without sacrificing performance. Preliminary work on this collaboration is described in Section 3.1.

2.1 Scheduling Workflow Applications

We have developed and implemented new strategies for scheduling and executing Workflow applications on Grid resources using the GradSoft infrastructure (see http://hipersoft.cs.rice.edu/grads/publications/GrADSIPDPS02.pdf). Workflow scheduling is based on heuristic scheduling strategies that use combined computational and memory hierarchy application component performance models. The Workflow is executed using a novel strategy to bind and launch the application onto heterogeneous resources. We have applied these strategies in the context of launching EMAN (see http://ncmi.bcm.tmc.edu/homes/stevel/EMAN/doc), a bio-imaging workflow application, onto the Grid. The workflow scheduling techniques were also applied in the context of scheduling "Montage" (see http://montage.ipac.caltech.edu/) workflows as a part of the Pegasus planner (see http://pegasus.isi.edu/). The simulation results for workflow completion times for different "Montage" workflows show that, by using the heuristic workflow scheduling, workflow completion times improve by an order of magnitude (more than 20 times) over the random scheduling strategy used by Pegasus for heterogeneous platforms. There is an improvement of more than 20% for homogeneous platforms. The workflow completion times produced by this approach are within 10% of that using a very expensive AI scheduler that does not scale to 2047 jobs.

We are currently working on launching the latest version of EMAN on the Grid with a large input data size [4 GB] to further validate the workflow scheduling techniques. We are also investigating the issues of better and more comprehensive Workflow-DAG scheduling for heterogeneous Grid systems, which minimizes data movements over high latency interconnects and handles large distributed databases, multiple DAGs and DAGs having nodes representing tightly coupled parallel computations. We are developing these strategies

in the context of scheduling three of the VGrADS applications – EMAN, EOL and LEAD. In future, we intend to investigate the issues of scheduling onto virtual grid abstractions.

2.2 Automatic Construction of Performance Models

One aspect of making applications Grid aware, in the sense of our current infrastructure, is the construction of performance models for the application's components in order to make proper decisions for resource selection and scheduling. The construction of performance models for the EMAN application has been a major focus of research efforts at both Rice and UH.

The approach we have pursued makes use of a set of equations, the structure of which is derived from knowledge of the component's computational requirements and the dependence upon input variables such as micrograph sizes, number of micrographs, and variables controlling the processing of the micrographs. Model parameters are then determined from execution traces. The objective is to attempt to separate application characteristics invariant with respect to architectures from those that are dependent. For instance, the number of floating point operations should be independent of the platform used for the processing, but the number of instructions will depend upon both that platform and the compiler used, as well as compiler options used. Similarly, memory access patterns should have a strong correlation to the application, while the number of cycles required will depend significantly on the memory system.

To better understand the impact of the memory system on performance, we have been pursuing a novel approach that uses application binary analysis to instrument preliminary runs of the application in order to determine the distance, in memory accesses, between accesses to the same cache block. Using the input from several training runs, we can fit curves to the data to determine the effect of increasing problem size on this distance, which we call *reuse distance*. This provides us with a reasonably accurate performance model that is able to predict which memory access will result in cache misses.

By integrating the components described above, we have derived a model for the main EMAN components and made a first mostly manual validation of the model. We are now in the process of automating the model fitting and investigating the sensitivity of the results to compilers being used as well as the impact of compiler options being used. The main platforms targeted at the moment are Opteron and Itanium2 based clusters.

3 VGrADS Execution System (UCSD, UCSB, USC, UTK, UNC)

3.1 Development of Virtual Grid Abstractions

The core activity amongst all of the execution system teams, led by the UC San Diego team, is to develop and refine the notion of virtual grids. The core elements include the virtual grid abstractions, implementation technologies, and underlying statistical resource classification techniques. To effectively support applications while achieving efficient resource selection and scheduling, virtual grids must enable applications to express simply their desired resource abstraction and preferences. To develop these abstractions, we have pursued a systematic

study of several leading grid application efforts – Encyclopedia of Life (SDSC/UCSD), EMAN (UHouston/Rice), LEAD (Illinois/UNC), and SATisfiability (UCSB) – developing for each of them a simple virtual grid description that expresses desired resource abstraction. The UCSD group is spearheading a large team effort to evaluate these abstractions for appropriateness for application expression and optimization.

We expect that an initial design of the abstractions interface will be completed at the end of the first year of the project (October 2004). An important goal is to characterize the performance benefits and potential losses due to use of a simple performance abstraction. An important subsequent step is to include a coordinated evaluation of the underlying implementation issues such as rapid resource selection, scheduling, rescheduling, and fault resilience. In the following sections, we summarize the following activities which directly support the development and evaluation of virtual grids: 1) grid resource configuration generation, 2) automatic resource characterization, 3) application studies, 4) grid modeling, simulation, and benchmarking, and 5) techniques for fault-tolerance.

3.2 Realistic Grid Resource Configuration Generation

Study of dynamic, adaptive grid applications and middleware depends on the ability to experiment with a wide range of realistic resource environments and to do detailed, performance and behavior-accurate simulations. The latter need is ably met by the MicroGrid system. However, to date little work has addressed the problem of realistic grid resource generation. Leveraging the observation that the majority of high capability grid resources are Linux clusters, the UCSD team is building a general-purpose grid resource configuration generator. Using several databases of Linux cluster configurations to populate a statistical model of grid resources and an array of statistical tools, we have designed and validated statistically a grid resource generator one for which the output has been shown to be statistically the same as other sample data sets (which are as large as 10,000 processors!). Further, by analyzing historical sample data and technology trends, we have extended our model to generate representative resource structures for grids of the future. We will use this grid resource generator to study the efficacy of our virtual grid abstractions and their implementation techniques, and make it generally available to the research community to enable better-grounded study of software and application behavior in grid resource environments.

3.3 Automatic Resource Characterization

To develop the automatic resource characterization capabilities necessary to build virtual grids, the UC Santa Barbara team is developing models and prediction techniques for resource availability. We have developed resource availability sensors for Linux/Unix workstations and Condor-controlled resources that use the NWS to record resource status. Availability sensing is a critical component of the virtual grids and obtaining accurate measurements suitable for automatic characterization proved more difficult than first anticipated. Indeed, we found that previous sensing data such as that taken from the NCSA administrative logs, while providing useful high-level insight, did not contain the accuracy required to form models and make accurate resource predictions, particularly with quantifiable confidence bounds.

To digest this data, we have developed an automatic modeling technique that uses Maximum Likelihood Estimation (MLE) and Expectation Maximization (EM) to derive availability models from measurement data. The system uses two goodness-of-fit tests (Kolmogorov-Smirnoff and Anderson-Darling) to evaluate competitive models to identify the most accurate fit. While EM-fit hyperexponential models of 6 or more parameters often fit the observation data best, the additional accuracy over a two-parameter MLE-determined Weibull is small. This result is particularly surprising since the hyperexponential uses more parameters to describe the data. Weibull confidence bounds and invertability are both tractable and computationally cheap to compute. Given only a small loss of fit accuracy, we found Weibull models to be a previously uninvestigated excellent choice. Further, for the purposes of making future predictions, we found non-parametric techniques based on resampling and Binomial modeling to be far superior. The Binomial method, in particular, can make future availability predictions (with provable confidence bounds) with as few as 20 measurements.

In summary, we have developed availability sensing technology more accurate than previous systems, automatic modeling technology that produces accurate and useful parametric statistical models, identified the most useful – the Weibull family of models, and automatic availability prediction capability that uses non-parametric techniques to make future predictions, complete with verifiable confidence bounds.

3.4 EOL Application Modeling

As part of the evaluation and refinement of our ideas for virtual grids, the UC San Diego team has developed a simple resource abstraction for the Encyclopedia of Life application. Because we have a close partnership with this application team, we have quickly settled on a high level abstract model for the virtual grid abstraction – and several refinements appropriate as the EOL implementation is enhanced for greater capability and efficiency. In addition to the macro-scale modeling, we have also engaged in detailed application behavioral modeling for subsets of the computation – PSI-BLAST and 123D – which correspond to the major compute-intensive parts of the backend annotation pipeline. These models provide reasonable estimates for the computational work in each subtask and are being used to evaluate the value of precise dynamic resource information in efficient execution of these phases of EOL.

3.5 Grid Modeling, Simulation, and Benchmarking

As an ongoing activity, which began with the GrADS project and will continue to support research in VGrADS, the UC San Diego team has completed several new releases of the MicroGrid system (April 2004). These new MicroGrid releases include automatic synthesis of internet-like topologies using well-established tools such as BRITE, and they enable large-scale simulation of large-scale networks (20,000 routers) by efficiently exploiting parallel resources such as the TeraGrid. Recent advances include the generation of realistic multi-AS networks, enabling realistic modeling of Internet routing structures. In addition, we have recently developed new load balancing techniques that improve our previous profile-based techniques with a new hierarchical approach to improve parallel simulation efficiency dramatically. Parallel efficiencies of 40% on 128 nodes enable experiments with networks

comprising large fractions of the Internet (for example a 20,000 router network, similar in scale to AT&T's network). These capabilities enable the study of large-scale grid systems and applications.

Several grid benchmarking efforts exist, for example the Grid Assessment Probes (GRASP) benchmark from UCSD, the NAS Grid Benchmarks (NGB) from NASA, and GridBench at the University of Cyprus. The UTK team is investigating these benchmarks for the VGrADS project, to see how they can be applied to the concept of a virtual grid. We are also examining the output from these benchmarks, to see if they generate useful and appropriate information for the purposes of grid applications. We are also measuring the overheads associated with some grid infrastructures (e.g., NetSolve, Globus) for operations such as file transfer and resource querying.

3.6 Fault Tolerance and Resilience

FT-MPI is a fault tolerant implementation of the MPI 1.2 specification being developed at UTK. The approach to adding fault tolerance is to enhance the error modes of MPI to allow for damaged communicators to be repaired. The UTK team is exploring the use of FT-MPI as a mechanism for adding fault tolerance to MPI applications that run under the VGrADS framework. At UNC/Illinois, we are also exploring the behavior of MPI applications when subjected to faults, as a basis for assessing the resilience of these applications to system failures; this is joint work with the Los Alamos Computer Science Institute (LACSI). Quantitative data from this assessment, together with measurements of system failure modes, will be used to develop predictive performability (i.e., integrated performance and reliability) models for application Grid scheduling.

II. Findings

During the reporting period (10/1/03–5/31/04), VGrADS research focused on three inter-institutional efforts: *Applications, VGrADS Programming Tools,* and *VGrADS Execution System*. The following sections summarize the findings of each subproject.

1 Applications (Rice, UCSD, UCSB, UH, UNC)

Work on applications during this reporting period focused on determining requirements for virtual grids (vgrids) to support common programming practice.

EMAN showed that support for "clusters" (collections of relatively homogeneous processors, connected by a relatively homogeneous connectivity, with a common file system) was required for applications with tightly coupled components. Higher-level "bags" (collections of heterogeneous resources) are also appropriate for systems built from such components, such as parameter sweeps. Finally, it appears that multi-level scheduling schemes, like our current distinction between workflow and MPI scheduling, are needed to separate concerns in real programs.

EOL led to much the same findings as EMAN - the need for homogeneous clusters and heterogeneous bags of resources. It also demonstrated the need for explicit handling of data movement in the virtual grid. This handling includes both performing the movement itself and estimating its cost, which is needed for effective scheduling.

GridSAT has very different requirements from the other applications. Here, the set of tasks is extremely dynamic, thus pointing out the requirement for run-time expansion (and contraction) of the virtual grid. It also relies more heavily than the other applications on topology information (in particular, identifying resources with good connectivity). Finally, fault tolerance is useful in GridSAT both to handle unreliable resources and to effect rescheduling when new resources become available.

LEAD has requirements much like EMAN and EOL for running an ensemble of simulations - the need for tightly-connected clusters and loosely-coordinated larger collections. It adds the need for streaming data, as opposed to the fixed data collections used in the other two applications. LEAD also allows interesting trade-offs between fidelity of results and fault tolerance, which require more investigation at both the application and system level before they can be refined into vgrid requirements.

2 VGrADS Programming Tools (Rice, UH)

The Binder in the VGrADS infrastructure is responsible for making the final modifications to the application before it is launched on the grid. An important goal of the Binder was to design a mechanism to cope with heterogeneous architectures present on the VGrADS testbed. The Binder achieved this by maintaining a high-level representation of the application until after grid scheduling decisions were made and only then compiling on the target machines.

We have demonstrated that tools that automatically construct performance models can be developed and that the constructed models are accurate enough to be used in scheduling applications on the Grid.

Through our application studies, we have demonstrated that the VGrADS strategy of using performance models for scheduling, particularly DAG scheduling, is extremely effective on heterogeneous computational resources, improving performance over current performance-blind strategies by up to factors of 20 in some cases.

3 VGrADS Execution System (UCSD, UCSB, USC, UTK, UNC)

Based on early evaluations, the core ideas of virtual grid abstractions are promising – both simplicity and sufficient specificity for applications appears achievable. Based on application case studies, the simple virtual grid descriptions are rich enough to capture the critical distributed architectural features of applications.

Automatic modeling technologies that produce accurate and useful parametric statistical models can be built using the Weibull family of models. These models can be combined with automatic availability prediction capability that uses non-parametric techniques to make future predictions, complete with verifiable confidence bounds.

Scalable online simulation of large grid applications, software, and grids consisting of 10,000 compute resources and 20,000 routers is feasible and can be achieved with reasonable performance using TeraGrid or other cluster resources, and is supported in the MicroGrid software.

A critical need for systematic, scientific evaluation of a wide range of dynamic middleware, network service, and applications, realistic generation of grid resources can be achieved with good statistical accuracy and reasonable computational effort and with moderate input data (10,000 processors). Generation is based on models that are calibrated by library input data, enabling automated recalibration as well as projection to future grid resource environments.

III. VGrADS Education, Outreach, and Training Activities

1. Training and Development Activities

The VGrADS project has provided opportunities for graduate students to become involved in an exciting and important research project. Through participation in VGrADS project meetings, email, and phone conversations, students have been able to interact with, learn from, and contribute toward the research of off-site VGrADS participants. The multi-site nature of this project has given students first-hand exposure to a wider range of research approaches and specialty areas than would typically be possible. These students bring their insights back to other students in their research groups who are not exposed to as many 'outside' collaborators, enriching the experience for other graduate students as well.

The VGrADS project has been a vehicle for collaboration among faculty, students, and staff. Additionally, the VGrADS project has provided students with a chance to build a very large-scale system in which all of the components must work together efficiently. The students have learned goal-setting and management techniques for distributed teams, and have learned how to use a variety of group communication techniques to make distributed teams effective. Since research groups are developing components of the system at various VGrADS sites, the project has also provided an opportunity for participants to collaborate closely with researchers with different expertise.

VGrADS Education, Outreach and Training efforts will begin active work with underrepresented students during the summer of 2004, by including two VGrADS students as part of an expansion of Rice University's existing summer program, Alliances for Graduate Education and the Professoriate, which brings undergraduate students to Rice to work directly with scientists who are doing active research.

2. Outreach Activities

Two meeting were held during the spring of 2004 to begin the planning process for the summer outreach activities (2-11-2004 and 4-6-2004). The VGrADS Education Outreach and Training Committee (EOTC), chaired by Richard Tapia, will continue to review and plan VGrADS EOT programs, including efforts to encourage minorities and women to pursue careers in computer science.

- Future planning sessions will occur via teleconferences and e-mail, and will occur regularly, beginning during the summer of 2004.
- The committee will coordinate EOT efforts among VGrADS sites.
- The EOTC will include a representative from each of the VGrADS sites.

We will be aggressively supporting two meetings devoted to increasing diversity in computer and computational science --

Grace Hopper Celebration of Women in Computing
Richard Tapia Celebration of Diversity in Computing

- Rice's Center for Excellence and Equity in Education (CEEE) will be presenting a panel at the Grace Hopper Celebration of Women in Computing, in October 2004, utilizing Rice students, CEEE employees, and CEEE workshop participants. The presentation will show that a variety of methodologies are required for success in the process of encouraging diversity. The same or a similar panel will be presented at Tapia 2005.
- VGrADS will provide travel support to these conferences for student participants and attendees.

VGrADS will provide content and support to the NSF-funded Computer Science Computing and Mentoring Partnership (CS-CAMP), during June 2004.

- Make presentations and provide curriculum at CS-CAMP sessions (Tapia and Cooper)
- Participate in planning and development of future CS-CAMP sessions, in preparation for scaling the program.
- Disseminate materials nationally, through NSF supported workshops, including the National Computational Science Institute's (NCSI) summer workshops, and the Supercomputing Conference.

In addition, VGrADS researchers have presented VGrADS research results and ideas in a variety of forums, including technical computer science presentations, presentations to applications groups, and presentations to students.