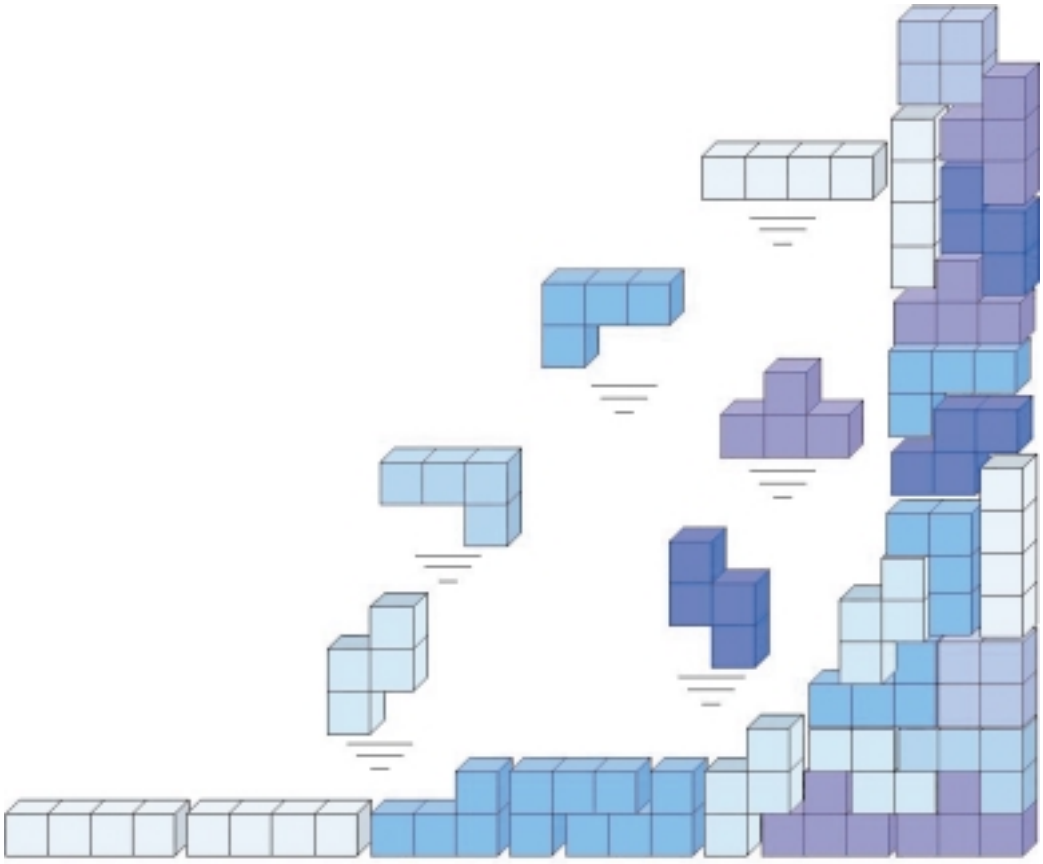
VALUATION OF ULTRA-SCALE COMPUTING SYSTEMS:
A White Paper



by
Valuation Project Team

December 22, 1999

TABLE OF CONTENTS

ULTRA-SCALE COMPUTING VALUATION PROJECT TEAM

David Bailey, Lawrence Berkeley National Laboratory/NERSC
Ian Foster, Argonne National Laboratory
Thuc Hoang, U.S. Department of Energy, Defense Programs
Morris A. Jette, Jr., Lawrence Livermore National Laboratory
Thomas Klingner, Los Alamos National Laboratory
William T.C. Kramer, Lawrence Berkeley National Laboratory/NERSC
Tina Macaluso, Science Applications International Corporation
Paul Messina, U.S. Department of Energy, Defense Programs
Dale Nielsen, Jr., Lawrence Livermore National Laboratory
Dan Reed, University of Illinois, Champaign-Urbana
Larry Rudolph, Massachusetts Institute of Technology/Co-Chair
Paul H. Smith, U.S. Department of Energy, Defense Programs/Co-Chair
James L. Tomkins, Sandia National Laboratories
John Towns, National Center for Supercomputing Applications
Mike Vildibill, San Diego Supercomputer Center

Valuation Project Team: This team served to develop and guide this effort to better define the metrics with which a clear case can be made for the need and value of ultra-scale computing systems in science and engineering.

EXECUTIVE SUMMARY

The Ultra-Scale Computing Valuation Project was undertaken to gain insight on utilization issues for both users and managers of the largest scientific computing systems and to begin developing appropriate metrics and models for such systems. The objective of this Project was to define a consensus-based approach within the community for assessing the value of ultra-scale computing systems. The ultimate goal is acceptance of a proposed approach and of the resulting recommendations by the appropriate stakeholders. Ultra-scale computers are general-purpose computers in actual use, whose computing power (the combination of aggregate processor speed, memory size, and I/O speeds) is within a factor of ten of the highest performance machine available. These ultra-scale computers are normally outside the area of focus of the commercial market forces. Today's ultra-scale machines incorporate thousands of powerful processors, are considered accelerated development machines, and are purchased to enable much larger and more complex computations than can be performed presently on more conventionally available platforms.

Participants in this effort included experts from universities, the federal government, national laboratories, and the computing industry. Several meetings were held, operational data were analyzed, and many discussions took place to arrive at the conclusions and recommendations outlined in this report.

An initial brainstorming session characterizing approaches to assessing the value of advanced computer systems kicked off the Project. This early session revealed a range of current practices among participants and resulted in a decision to collect and analyze job trace data from several supercomputing facilities. Some participants defined utilization as the fraction of node hours used out of the total time the advanced computing platform is available for use, while others defined utilization as the fraction of time the platform is in use regardless of its availability. Participants agreed that the distinction between these two definitions is relevant mostly for new machines where the machine is unavailable for significant periods of time and, as a machine matures, there is less down time and the two definitions converge. According to either definition, a computer would be considered fully utilized if adding more jobs to the queue of jobs awaiting execution serves only to increase the average delay for jobs.

Participants noted that neither of the above definitions refers to the utilization rate for any of a computer's sub-systems such as memory, disks, or processors. Rather, it assumes that when a job is assigned to a particular node (set of tightly coupled processors, memory, disks, etc.) within a parallel computer, all of those resources are unavailable for use by other jobs and therefore are considered utilized. Theoretically, peak utilization would be achieved when jobs were assigned to all nodes all of the time.

As the Project progressed, it became increasing evident that utilization is not a sufficient measure of ultra-scale computer valuation. The first problem is that different organizations using different platforms use different approaches to address and measure utilization. Some progress has already been made on this front and more is expected as the recommendations of this report are implemented. A second, more

serious problem with utilization as a metric is that driving utilization to too high a level almost always results in an overall slowdown in system performance. When the slowdown is significant, the effect of achieving very high utilization is a counter-productive decrease in the ability of the system to support the applications for which its acquisition was justified. A third and more subtle weakness with utilization is that it does not measure the capability quality of the machine. In fact, the replacement of many capacity jobs by any capability job requiring the same total amount of resource can only decrease the utilization. Utilization as a measure penalizes exactly those capability jobs that are the driving rationale for the creation of large, integrated, ultra-scale machines.

The bottom line for the participants is a consensus that the value provided by an ultra-scale machine can only truly be measured — in the long run — by the scientific output produced by using it. One must ask, "Is the system doing what it was designed and funded to do?" Ultimately this is not measured by node-hours used, but by capabilities converted into discoveries or other valuable scientific or technical outcomes.

In the end, the facilities that operate ultra-scale computing systems should be judged in the same way other national facilities, such as accelerators, are judged. Typically, periodic peer review is used to assess whether the missions and goals of the facilities are being met. Such peer reviews have worked very well to ensure the effectiveness and efficiency of facilities that serve the targeted scientific community. The value of ultra-scale computing facilities and the scientific output of the systems should be evaluated in a similar manner.

As a result of this Project, participants were able to identify operational similarities at the sites they represented while recognizing that there are few general practices for measuring use and assessing value that will hold across all sites. Rather, what is needed is a sufficiently flexible and graded approach that can be used by each site to measure the contributions of advanced computing systems to scientific thinking and meeting programmatic objectives. This approach must recognize that the first-of-their-kind status of ultra-scale platforms directly impacts initial utilization. Other factors that affect system performance and its overall value, such as allocation policies, utilization tradeoffs, and the absence of sufficient tools for measuring performance, were identified.

Acceptable ways to evaluate "ultra-scale" computing systems are being defined and a degree of consensus on these approaches is emerging within the ultra-scale computing community. Analysis of trace data provided by the Project Team revealed desired operation ranges of response time and throughput (number of jobs) for a given workload. It was important to consider different classes of jobs and differing workloads in the analysis. It was also learned that attempting to obtain greater throughput than that obtained by running the machine within the Desired Operation Range (DOR) of each system results in a rapid deterioration of system response time. Because of these considerations, the large-scale computing research and applications programs in government and academia agree that developing understandable and defensible measures for assessing value and utilization of these platforms is essential. The

community must make every effort to measure how effectively our national computing resources are being used, so that continued improvements can be made.

The ultimate impact of this effort on individual sites and the agencies that manage them is tied to a willingness to define metrics for arriving at a desired operation range — on a system by system basis — and subsequent agreement to modify practices and policies to move toward the optimum. Further research is needed in key areas such as designing more efficient scheduling algorithms. Ramifications of this Project for the high-end computing industry include probable changes in future procurement arrangements and recognition that new tools are required by managers of ultra-scale computing platforms to address utilization considerations.

INTRODUCTION

High-end computing efforts are driving the creation of new classes of applications and design of advanced computing platforms and system components that are far more sophisticated than those of the past. Improvements in hardware and software present challenges for use because each evolutionary stage is first-of-its-kind and requires a new learning curve for those who will use and manage it effectively. But the benefits to be gained from running complex simulations cannot afford to wait until the systems are mature; new application codes are being developed in tandem with the new hardware design and development. Ultimately, the advances in learning what these new platforms make available to the entire scientific community determine the overall value of ultra-scale computing systems.

> "The most constant difficulty in contriving the engine has arisen from the desire to reduce the time in which the calculations were executed to the shortest which is possible."
>
> Charles Babbage, circa 1847

ULTRA-SCALE COMPUTING SYSTEMS: A Requirement for Cutting-Edge Science and Engineering

Across the Federal government, agencies are working to solve science and engineering problems of unprecedented complexity. The Department of Energy's (DOE) Office of Defense Programs is charged with ensuring the safety, security, and reliability of the Nation's nuclear weapons stockpile, but must do so without further testing, as a policy of the United States government. This restriction requires the use of computer simulations of a scale and complexity never before attempted. At the same time, National Aeronautics and Space Administration (NASA) is managing enormous collections of earth observation and planetary data while the Department of Defense (DoD) incorporates computer simulations into all aspects of peace keeping and warfare. Concurrently, the National Science Foundation (NSF) and the DOE Office of Science are working to resolve the mysteries of long-term global climate change and other large-scale system simulations.

While these mission needs and science drivers push the Federal mandate for advanced computing technologies into new frontiers, the actual market for the largest computing systems has always been relatively small. Forming working partnerships between the Federal research programs that need the advanced systems and the companies that will produce them is the only reasonable way to create the environment needed for these complex simulations. The term “Ultra-scale computing systems” is used to refer to all those general-purpose computers in actual use, whose computing power (the combination of aggregate processor speed, memory size, and I/O speeds) is within a factor of ten of the highest performance machine available.

> "Ultra-scale computers are all those general-purpose computers in actual use, whose computing power (the combination of aggregate processor speed, memory size and I/O speeds) is within a factor of ten of the highest performance machine available."
>
> Paul H. Smith, DOE

National Security Example

The challenge faced by the DOE Accelerated Strategic Computing Initiative (ASCI) is particularly great. Since President Clinton's 1995 announcement that the U.S. would pursue a "zero yield" Comprehensive Test Ban Treaty, the Department of Energy has taken a new approach to its responsibility to ensure confidence in the safety, performance, and reliability of the national nuclear stockpile. Without nuclear testing as the final arbiter of scientific judgement, weapons scientists must rely more heavily on computers to simulate the aging process and its impact on weapons systems.

The DOE Stockpile Stewardship Program was established to develop new means of assessing the performance of nuclear weapons systems, predicting safety and reliability, and certifying functionality. ASCI was created as the focus of DOE's simulation and modeling efforts aimed at providing high-fidelity computer simulations of weapon systems that will enable scientists to make the necessary judgments to maintain the credibility of the nuclear deterrent. The 2004-2010 timeframe is the key target for having available operational ASCI computing systems of sufficient power and codes of adequate fidelity so a smooth transition from "test-based" certification and assessment can be made. Experimental data from above ground test facilities and archival data from fifty years of nuclear tests must be integrated with improved scientific understanding to provide high-confidence predictive simulation capabilities for supporting decisions about the enduring stockpile.

Today's ASCI applications require computing capability several orders of magnitude greater than what existed in 1995. The programmatic strategy is to build future high-speed computing systems by scaling commercially viable building blocks, both hardware and software. Such an aggregation of commodity building blocks into 10- to 100-teraOPS (trillion operations per second) systems requires significant development of integration and scaling technologies that are not currently being driven by commercial market forces.

Climate Modeling Example

The United States must have unprecedented acceleration and extension in state-of-the-art climate modeling to reduce today's uncertainties about long-term climate change. This is needed by early next decade to support national and international energy and environmental policies. To achieve substantial advances, considerable progress must be made in climate simulation model development while significantly reducing

uncertainties in model-based projections of climate change. At the same time, there is a need to increase both the availability and usability of climate change projections to the broader climate change research and assessment communities.

Progress depends on advances in scientific knowledge which depend, in part, on results generated by models and on the physical ability of computers, databases, networks, and associated computational infrastructure to process large amounts of data in short periods of time. Current estimates are that improved climate models require 40-teraflops machines to run atmospheric and ocean models. They require data storage facilities for holding petabytes of raw and processed climate data, and they require networking infrastructure that will allow information to be pushed across the network at high speeds in tandem with white board interaction, visualization, and animation. Finally, there must be workstations, PCs, and other machine interfaces to give users access to the benefits of this new research environment.

GOVERNMENT AND INDUSTRY WORKING TOGETHER

While Moore's Law advancements drive the rapid evolution of processor performance, computer companies experience few commercial incentives to speed up their development efforts beyond this rate. Because evolution at the speed of Moore's Law does not meet the ASCI timeframe, DOE created the ASCI PathForward program to speed near-term development of essential technologies that will be required to meet ASCI time and scale requirements (see ***Figure 1***). The goal of the strategy, which focuses on integration and scaling technologies rather than on the actual commodity building blocks, is to cultivate a sustained commercial effort so that unique platforms need not be built continuously. Additionally, this strategy will provide development platforms that can serve as interim stages to reach the ASCI milestones.

The PathForward program consists of multiple partnerships with computer companies to develop technologies that are expected to either not be in the current business plans of computer manufacturers or not be available in the timeframe or the scale required by ASCI. The technologies currently being developed through PathForward are in three critical areas: interconnect, storage, and software. Through PathForward, ASCI will stimulate the U.S. computing industry to develop high-performance computers with speeds and memory capacities hundreds of times greater than currently available models and ten to several hundred times greater than the largest computers likely to result from current development trends. Successful deployment of the PathForward strategy results in continuing partnerships between DOE, the Defense Programs' National Laboratories, other government agencies, and various U.S. computer manufacturers to accelerate the development of larger and faster computer platforms and software.

HEC industry needs targeted investment to meet Stockpile Stewardship requirements

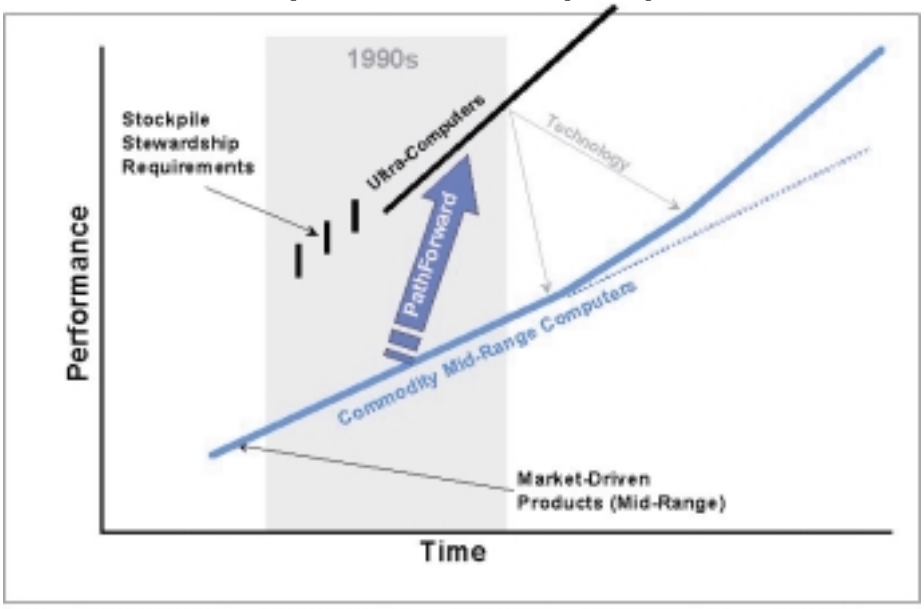


Figure 1: Stockpile Stewardship Requirements

CONSEQUENCES OF PUSHING THE ENVELOPE

Most components of the ultra-scale computing platform have changed substantially within the past few years. Allocation methodologies have evolved along with system architectures. System managers, operators, and users have all had to learn how to program differently to use the new massively parallel capability. The resulting paradigm shift to parallel architectures has been followed by substantial increases in the number of highly parallel jobs, which in turn have caused changes in allocation and user behavior.

Generally speaking, the overall goal for an ultra-scale computing platform is to capitalize on the new capability afforded by the new machine at the earliest time. This means that managers strive toward achieving the desired turnaround for a given class of work a large fraction of the time. In the case of ASCI, for example, application results cannot afford to wait until the systems are fully mature. Interim milestones must be achieved. Managers of ultra-scale computing platforms have to deal with issues of capacity versus capability as discussed in the following section. There is no sharp distinction between these two for most ultra-scale computing facilities.

Capacity versus Capability Computing

There are many application code types and usage patterns that need to be satisfied within the community of users of ASCI systems. There are several dimensions along which these might be classified, but the one of greatest use in the management of

resources is the determination of whether an application needs 'capacity' resources or 'capability' resources.

The need for capacity resources is generally considered to be one of needing access to a large number of compute cycles over some period of time. Furthermore, these cycles need only to be delivered in relatively small quanta. Typically, this means that the application can be decomposed into a large number of runs, each requiring a modest number of processors. Capacity computing needs can be met by a variety of means (for example, tailored job queues that use a fraction of an ultra-scale system, several smaller systems, or workstation clusters) and are easy to schedule. However, in the aggregate, they can consume a large enough fraction of resources to require careful scheduling.

The need for capability resources is generally considered to be the need for access to a significant collection of resources in a coordinated fashion within a finite window of time. This category can be further broken down into a number of types. The National Computational Science Alliance (referred to as the "Alliance," throughout this document) defines, in ***Table 1***, four types of capability computing calculations. Capability computing is meant to be the utilization of over half of the system's CPU, memory, disk, and/or I/O resources to support individual applications.

Table 1. Alliance Job Size Categories

One: Hero Calculations	***Three: Rapid Turnaround Supercomputing***
These are simulations that are so memory and/or CPU-intensive the dedication of a large resource for days to weeks is required. • ***Metric:*** Routinely Execute a Problem 100x Bigger or Longer Running Than Possible on a Modern Desktop in Same Turnaround Time	Simulations that are too large for desktop systems, though not at the scale of "Hero" or high-throughput simulations. This is the Largest Category of Users and Projects. • ***Metric:*** Execute a Problem 10x Larger Than Possible on a Modern Desktop in 1/10th the Turnaround Time
Two: High Throughput Calculations	***Four: "Big Data" Applications***
Typically a large number of uncoupled simulations surveying a parameter space, perhaps as a part of an optimization calculation, where rapid turnaround is required. • ***Metric:*** Execute 100 Desktop-Sized Problems With the Same Turnaround Time As One Problem on the User's Desktop	This involves interactive analysis and visualization of data that is too large to be moved to or displayed on the users' desktops. • ***Metric:*** Ability to Visually Interact With Data That Resides on Systems 100x Larger Than Possible on a Modern Desktop With the Same Level of Interactivity

Ultra-scale computing platform managers in the ASCI program define three categories of job problem sizes (hero, medium, and small) that are comparable to the Alliance categories (see ***Figure 2***).

Figure 2: ASCI Application Problem Size

Generally, for every long production run there may be 5-10 development runs that need to take place on the ultra-scale platform to take advantage of the capabilities of a resource. From a scientific perspective, switching from small machines to ultra-scale machines midway through the research process presents problems. Large simulations require a new capability environment that can handle a workload of mixed job sizes, including (1) development jobs requiring various numbers of nodes, (2) using the entire machine for some jobs, including algorithm scaling studies, and (3) successfully running some number of large (defined at 25% or more of the entire platform) and long (greater than four hour) jobs. For example, at one laboratory a large ASCI job requires 240 wall clock hours and several thousand processors. Typically, a job this size will be broken into segments that will each require a few hours to run. The ensemble of segments that represents the whole job might take two weeks to run to completion.

THE IMPORTANCE OF METRICS

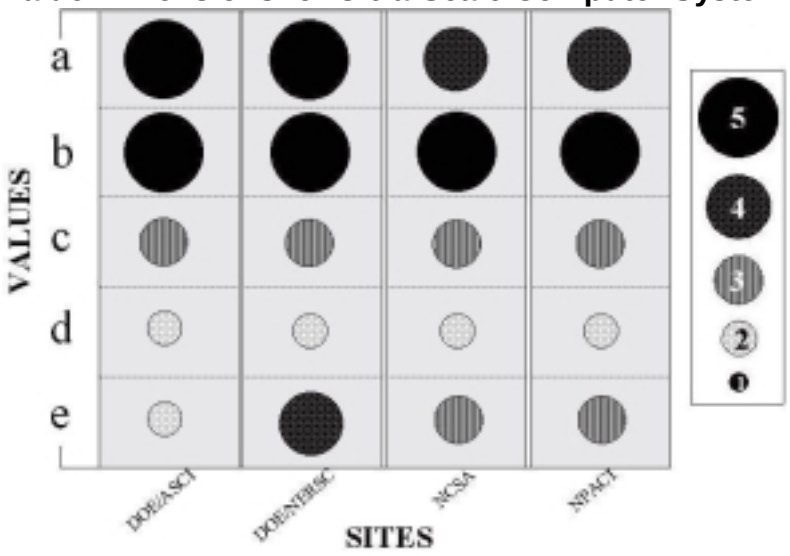
Systematic, logical thinking essential to scientific computing inherently acknowledges that what cannot be measured, cannot be managed. If it cannot be managed, it cannot be improved. But, as in many fields, assessing the overall value of a highly sophisticated resource dedicated to pushing the forefront of knowledge requires complicated analysis and calculation. Ultimately, the value of the ultra-scale computing platform must be defined and measured in terms of usefulness to the user and the return on investment (ROI) provided to the stakeholder. The original needs of the program must be assessed. Why was the platform purchased in the first place? Are the original objectives being met? Are the numbers of projects and users served meeting expectations? To determine ROI, the value of returns must first be assessed and

understood, and then it must be assigned an overall, aggregate, weighted value. ROI is not measured in dollars alone, but in value to the users and the stakeholders. This means the dimensions of value, such as the following, must initially be acknowledged, then measured:

a. *Availability of the system:* Are enough system resources available to support mission critical applications in some acceptable manner? Are users able to achieve objectives on priority jobs?
b. *Capability machines for mission objectives:* Was enabling capability jobs a reason the platform was purchased in the first place? Can an important task be performed overnight? Can the resources required to accomplish the necessary simulations be reasonably acquired?
c. *Response time:* Do the simulations enabled by the resources exhibit a demonstrable decrease in turnaround time, as expected by users? For specific classes of applications or users, is the response time appropriate?
d. *Throughput:* Is the throughput (number of jobs) meeting expectations?
e. *Providing needed allocation:* Is the system sufficiently agile to meet diverse user needs? Are the numbers of projects and users served meeting expectations?

To assess the dimensions of "value," focus should be placed on these five areas. Each site performed a self-assessment on the dimension of "value," using a scale of 1-5 with 5 being the highest value (see ***Table 2*** below). As expected, there were variations in the assessment of the value of the selected parameters. Benchmarks for chosen metrics will be a function of the particular system and the value attached to it for that system.

Table 2. Value Dimensions for Ultra-Scale Computer Systems by Site



Converging beyond anecdotal evidence when describing the value of a scientific system is difficult. Analyzing trace data and turn-around times to assess utilization metrics may not sufficiently measure the long-term productivity of scientific research. The bottom line for managers of ultra-scale computing platforms is answering questions such as

those posed above. Only when the metrics used reflect the complex answers to these questions will the true value of the platforms be fully understood. The ultimate success of the metrics and the effort to apply them depends on the extent to which consensus-based approaches to defining these difficult concepts are adopted by high performance computing organizations.

SELECTING AN APPROPRIATE USAGE MODEL FOR MEASURING UTILIZATION

Recent attention has focused on the utilization of the ultra-scale platforms under management by the DOE and other government programs. Understanding efficient utilization for all computer systems requires an appropriate usage model. This can best be demonstrated with three examples:

- Desktop PCs and single-user workstations are purchased with the objective of increasing the productivity of individuals performing a wide spectrum of in-office technical tasks. Guaranteed availability on demand is the dominant requirement. Informal probing has shown average utilization in the range of 1% to 2%.
- Shared departmental and enterprise production computers are purchased to provide computer cycles at the lowest possible cost per cycle consistent with programmatic turn-around time requirements. Low cost per compute cycle is achieved at the expense of predictability for task completion times and responsiveness to individual users.
- Accelerated development computers are purchased to (1) shorten the time to develop critical new software, (2) create new capabilities, (3) perform key new calculations, (4) increase the productivity of key scientists and engineers, and (5) decrease time to market. For simulations performed on highly parallel platforms, this means advancing parallel simulation software, which in turn requires experimentation for a full range of problem sizes up to and including use of the largest system sizes available.

Many ultra-scale computing platforms, notably those required by ASCI, fall under the category of accelerated development computers. A very large, accelerated development computer would be considered fully utilized if adding more work to the queue of jobs awaiting execution serves only to increase the average delay for jobs in the waiting queue without increasing the throughput. Note that this definition makes no reference to the utilization rate for any of a computer's many sub-system components such as memory, disks, or processors. It does assume that when a job is assigned to a particular node (set of tightly coupled processors, memory, disks, etc.) within a parallel computer, all of those resources are unavailable for use by other jobs, and therefore are considered utilized. The peak theoretical utilization would then be achieved when jobs were assigned to all nodes all of the time.

"The computer would be considered fully utilized if adding more work to the queue of jobs awaiting execution serves only to increase the average delay for jobs in the waiting queue without increasing the throughput."

Clarification

There are some very important clarifications required before this definition for computer utilization can be accepted. In particular, it matters greatly why there is a queue of waiting work. Consider the following examples. First, if the system is often unavailable due to hardware or system software problems, a long queue will develop if the submission rate remains constant and this will lead to a low utilization with long wait times. This is not a utilization problem, however, because the utilization rate is merely a symptom, and not the underlying problem. This situation should obviously be categorized as a reliability or availability issue. Remediation lies with the system vendor/integrator and is unrelated to the question of whether or not there exists a sufficient workload to utilize the system effectively. Second, if a long queue exists due to an inability of the workload scheduler to place more than a certain number of jobs on the system at once even when the queue of waiting work contains jobs which would fit on unused nodes, this clearly *is* a system utilization issue. Remediation requires the development of a more capable scheduling mechanism. Finally, it is not an issue if users queue many jobs and a large number of required simulations, creating long queue wait times, provided the users understand that greater demand has been placed on the system than it can handle in a short period of time.

Utilization as a Metric

Historically, managers of advanced computing platforms have used a variety of approaches to assess system utilization. The NASA Numerical Aerospace Simulation (NAS) Facility, for example, has operated parallel supercomputers for the past 11 years, including the Cray C-90, Intel iPSC/860, Intel Paragon, Thinking Machines CM-5, IBM SP-2, and SGI Origin 2000. The variability of the *Available Node Utilization* of some of those machines is shown in ***Figure 3***. Recognizing the range of machine architectures, a time span of more than six years, large numbers of different users, and thousands of minor configurations and policy changes, the utilization of these machines (see Jones & Nitzberg report listed in Appendix) shows three general trends:

- scheduling using a naïve first-in, first-out, first-fit policy results in 40-60% utilization
- switching to the more sophisticated dynamic backfilling scheduling algorithm improves utilization by about 15 percentage points (yielding up to about 70% utilization)
- reducing the maximum allowable job size further increases utilization

This last policy however defeats one of the purposes for buying ultra-scale machines, namely to gain new capability. Most surprising is the consistency of these trends across different platforms and user communities.

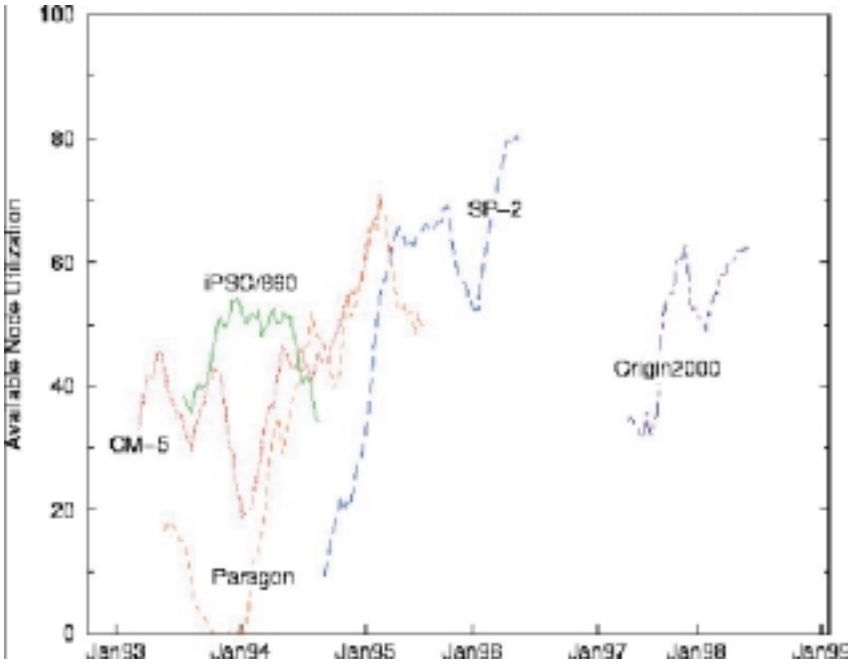


Figure 3: NASA Overall Comparison

During this Project, participants spent a great deal of time describing what is meant by utilization of ultra-scale platforms. Based on the various definitions of utilization, it was determined that a computer would be considered fully utilized if adding more jobs to the queue of jobs awaiting execution serves only to increase the average delay for jobs. Typically, to measure utilization, modest length periods are defined and set aside as being inappropriate for measurement.

Experience with a New System

One example of the process a major facility must go through in placing a new ultra-scale capability into service is demonstrated in ***Figure 4***. In 1997, NERSC at Lawrence Berkeley National Laboratory (LBL) transitioned its primary production computing capability to a massively parallel ultra-scale computer by placing into service a large, early delivery Cray T3E. At the time of introduction, this system was the largest unclassified supercomputer in the US and represented a 20-fold increase in raw computing power to the 2,500 scientists who use NERSC. NERSC, working closely with Cray Research, was able to improve utilization through the gradual introduction and exploitation of major system software functionality such as job migration and system initiated checkpoint/restart. During the first 18 months in service, the T3E utilization increased from approximately 55% to over 90% (***Figure 4***) while still focusing most of the system resources on large jobs. This represents almost a factor of two in price performance increase for the system or the equivalent (in 1999 costs) of $10.25M. At the same time the system was improving, T3E users were making improvements in applications to better utilize the system and improve its scientific output.

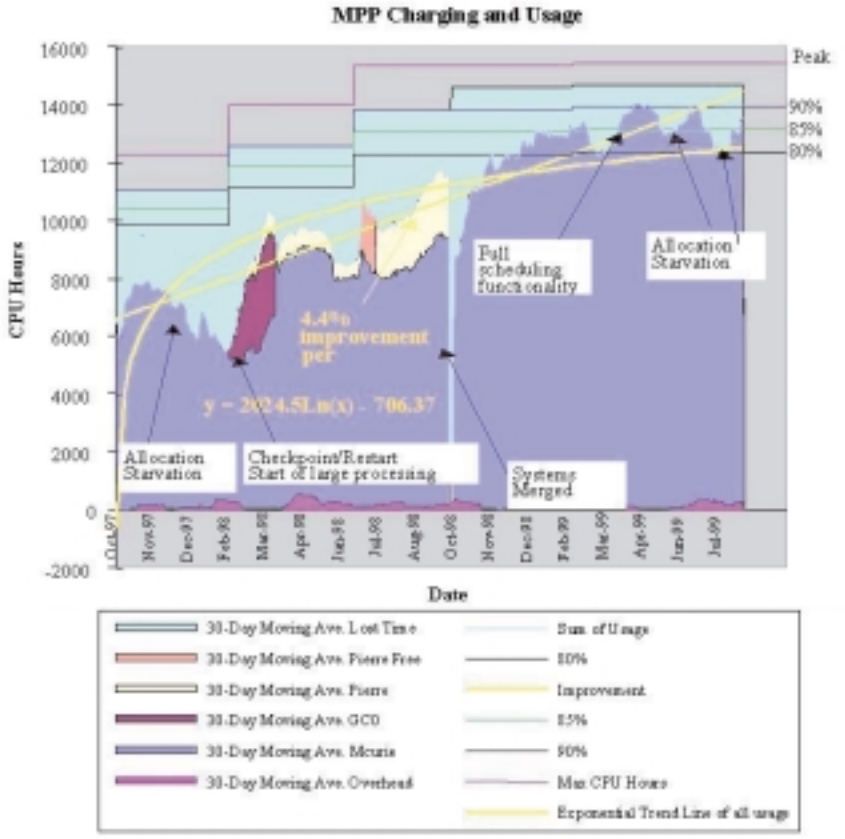


Figure 4: NERSC Evolution of T3-E Utilization

It is unfortunate that this scenario must be repeated for every new architecture delivered. But rapid changes in the high performance computing industry make it virtually inevitable that each new type system will experience the same learning curve. This is partly due to the facility's and its users' needs to learn how best to implement, tune, and run the new system and applications. But it is also due to the fact that few of the basic system software capabilities are transportable to the general system or can be shared among vendors. High performance computing vendors have little incentive to invest in and maintain advanced system software capabilities since, until recently, there has not been a reward structure for creating a system that is more effective rather than yielding faster performance.

Generally, ultra-scale computing platform managers assign preference to large jobs to ensure there are sufficient resources to run and there is a trade-off between quick turnaround for development jobs and maximum efficiency for production jobs. This trade-off translates into decisions that must be made between (1) asynchronous user behavior and interactivity and (2) using a batch queue system to provide sustained loading over longer time versus peak loading to maximize interactivity over shorter time periods.

Scheduling and Utilization for the ASCI Systems

What criteria are used to schedule runs on the ultra-scale platforms? The DOE labs responsible for managing ASCI report using criteria based on these priorities: (1) allocate resources to the most important work, (2) maximize responsiveness during working hours for program development, including debugging large scale applications, and (3) maximize resource utilization.

Hundreds of thousands of components and services are tightly interconnected and failure of even one node can affect a large fraction of the machine. (The impact of component failures varies widely depending on what was running on that part of the machine.) New state-of-the-art machines often use new technologies and incur unforeseen reliability problems initially. Ultra-scale machines often require incremental installation of equipment and on-site hardware upgrading as characteristics are discovered and engineering changes are incorporated. A similar (or in some case, worse) situation exists in the system software available on such machines. System software "crashes" occur and often require extensive periods of analysis before the causes can be identified and addressed. Often new system software features required for large applications such as parallel programming models, advanced languages, message passing, shared memory, communications protocols, etc. are immature on new architectures or require extensive modification for large scale operation. System hardware and system software may also lack functionality taken for granted in more mature computers, such as time-sharing or effective memory management.

> "From the time vector processors were introduced, it took over 10 years for vectorizing compilers to fully mature."
>
> Dan Reed, U. of Illinois

The existence of a long queue on a reliable and efficiently scheduled system may not indicate full utilization. Once a scheduler has determined to start a job, it is the responsibility of "system software" to actually load the problem into the memory of the allocated compute nodes and initialize communication between the nodes. Simple implementations of this loading and initialization process (which initializes nodes sequentially, for example, *not* in parallel) were common among early parallel computer manufacturers and adequate for machines with small numbers of processors. For these ultra-scale systems with several thousand processors, such implementations are completely inadequate because with such a serial bottleneck those system functions take far too long to be practical.

Application codes that inefficiently use a parallel computer's computation and communication resources during a simulation directly reduce the overall throughput rate for a system. The expectation for an application code with good parallel scaling efficiency is that using twice as many nodes for a given problem should reduce the execution time by "almost" half. Any job that doubles its node count while achieving significantly less than such a 50% reduction in execution "wastes" the additional

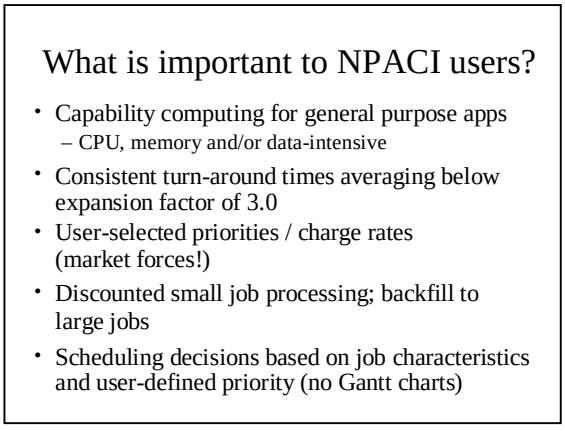
resources because it blocks other jobs from using the same resources, with little or no benefit. It is important to note that utilization measures of individual system components (such as communication, memory, and CPUs) may or may not indicate low utilization; in fact, node utilization as a measure by itself can actually mask the presence of this serious problem. A thorough scaling study on each of the codes scheduled to consume significant amounts of time on the system must be performed to determine application efficiency.

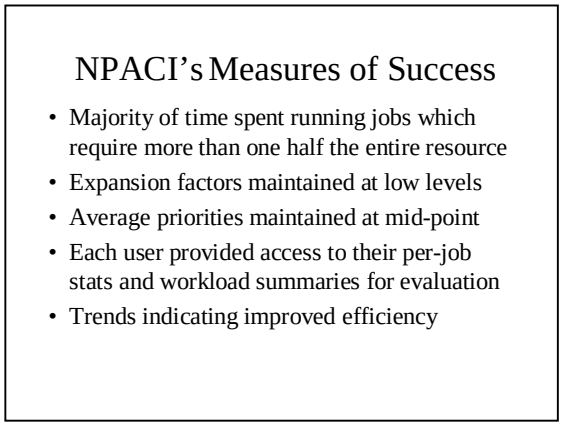
Scheduling and Utilization—Other Examples

It is important to note the strategies employed by various centers and those responsible for resource management of significant computational resources. In the following, a few examples are provided.

Within the National Science Foundation Partnership for Advanced Computational Infrastructure (NSF PACI) Program, there is a strong parallel at the Leading Edge Site (LES) of both the National Partnership for Advanced Computational Infrastructure (NPACI) and the Alliance with respect to the issues being addressed for the management of ultra-scale resources. Though the scale is not as great, these centers manage the largest resources available to the NSF computational science and engineering research communities.

While the practices of each LES within PACI differ to some degree, the general strategies are quite similar. The Alliance large-scale simulation goal is to satisfy the needs of capability computing as defined earlier in this paper for members of the Alliance. The intent is to do this using means that are fair to all those entitled to access to these resources while making the most effective and efficient use of the resources possible. Sometimes, these considerations conflict.





The questions of access and amount of resource to which users are entitled are handled by a peer review process. It is only important to this discussion to know that such a process exists. Peer review is also the process used to allocate resources at other sites including NERSC and NASA.

Considerable resources are used in application development and debugging and in the preliminary simulations necessary to determine the correct parameters for the largest scale simulations. For these jobs, the systems are continually monitored and measured in a variety of ways. Weekly meetings are held to discuss the systems performance and feedback received from the user community either anecdotally or through direct reports of problems. The environment is constantly tweaked as these issues change and as the demands of the users evolve. The process is complicated by sporadic system failures that must be dealt with in a manner that has minimal impact on the users' experience.

Principal investigators from the projects that receive the largest time allocations are invited to participate in these meetings on a regular basis to express their concerns. During the meeting, regular requests by users for scheduling of resources to conduct capability computing runs are also addressed. The Alliance has decided to set aside 256 processors in the form of a single system image Origin2000 to enable these jobs. This machine represents approximately 17% of the Origin resources at the LES of the Alliance. Resources for which such use will be assigned priority are expected to grow to 50% of the total Origin2000 resource over the next six months.

The managers of advanced platforms are charged with maintaining the delicate balance between user needs, most effective use of resources, and provision of a productive user environment.

Impacts of Allocation of Resource Decisions

The ways that access to an ultra-scale system is authorized greatly influences utilization. If system resources are either under or over allocated, usage and client satisfaction can be greatly degraded. Consider the typical impact the following allocation methods can have on utilization:

1. Exact division of resources (typically CPU time and storage) and strict enforcement of limits subdivide the resources of the system on some priority basis. By exactly cutting up the total resources and limiting users to their share, a system can easily become underutilized. There is a "feast or famine" mentality that causes users to hoard time early in the allocation period. If enough users hoard time, then utilization is low early in the period because there are not enough jobs to keep the system busy all the time. Later, when everyone tries to use their time, they may not be able to use it completely. The results are underutilized resources AND degradation of quality of service.
2. Some sites mitigate the previous impact by periodically taking time away from users who have not used it and redistributing it to users who are short on time. Since time lost due to idle processors cannot be recreated, this typically results in a scarcity of work early in the period and a severe backlog late in the period. It also is often cumbersome to implement and manage. If the recapture of time is done too frequently, it results in several mini feasts and famines throughout the allocation period.
3. Another way to address the concern of underutilization is to oversubscribe the system by allocating more resources than it can possibly deliver. This is done in one of two ways. If there is too much oversubscription, users expect to be able to do

more work than the system can perform. If enough users attempt this, the system will be clogged with work and users. More resources will be needed to manage the overload and the system loses efficiency as the quality of service degrades even though it may appear to be using all the CPU time. The second way to overallocate system resources is to institute a priority system, so that a user can submit work at a lower priority, risking expanded run times, while being charged less. Alternatively, a user can ask for higher priority and "pay" more. While this typically yields higher utilization, it also extends the slowdown factors of the system sometimes to the extent that it takes so long to run a job the scientists do not even bother to try.

Most sites use one or more of these methods, often in combination, to try to balance keeping the system utilized on the one hand but still responsive on the other hand. No method is entirely successful, so it takes dedicated system managers to be constantly monitoring and tuning the methods.

UTILIZATION TRADE-OFFS

Managing utilization of the ultra-scale computing platforms requires systems operators to decide among a large number of complex trade-offs. Factors to be considered include:

- *Job mix* - This includes the categories and size of job described above (for example, large production runs requiring thousands of processors versus smaller development runs to test and validate code). Job mix requires adequate management of memory, internal network bandwidth, and file system concurrently; the risk is that a large capability job may be starved if any single resource is not managed well. This is complicated by the fact that different types of jobs will require memory and CPU resources in differing proportions (for example, large jobs in chemistry versus computational fluid dynamics).
- *People priorities* - Some users and/or projects might be considered "more equal" than others because they are completing higher priority work. This means that resources must be available to meet the high priority needs — sometimes to the exclusion of other users and other jobs — forcing managers to provide guaranteed access to fewer, key users at the possible cost of lower utilization.
- *Learning curves* - The optimal target for a platform is usually running on ¼ to ½ of the entire machine. Use of such large fractions of systems, particularly early in existence, is not likely as system operating software is still being developed and applications programmers are still becoming familiar with the scheduling processes and operational algorithms required to make effective use of the ultra-scale platform. One conceivable alternative is to run only small jobs. However, in practice, the only way to ensure that a machine is ready is to subject it to real jobs and real workloads. Therefore, when users are kept off the machine with the goal of fixing all the problems, the net result is serious delays in the development and scaling of applications to make use of the capabilities and features of the system. This in turn leads to further utilization problems.
- *Absence of Tools* - Because the ultra-scale platforms are first-of-their-kind, tools for measuring efficiency, accounting for use, and for tuning system parameters for higher levels of efficiency are not yet in place. There is an imbalance between the

size and diversity of the software needed and the size of the new systems. Initially, accusations of low utilization are often met with anecdotal evidence and little systematic data; time and sponsored efforts are needed to evolve better tools for these platforms.

All of these factors, and the trade-offs that must be made among them, have to be balanced when managing ultra-scale computing platforms. Managers must respond to a highly complex problem with a large number of degrees of freedom. Scheduling efficient use of all of the resources is like a "Tetris" problem (shown on the cover); the right job at the right time is needed to consume whatever resources are available. If there is conflict or overlap, utilization efficiency may decrease.

Utilization Should Not Be the Sole Metric

As these arguments are meant to demonstrate, utilization is not a universally defined term and different organizations use different approaches to define it. **The Project participants believe strongly that the true measure of the value of ultra-scale computing systems in the long run should be the scientific output of these systems**. Are the systems doing what they were designed and funded to do? How is this measured? The answer is that the overall value of the ultra-scale platform must be assessed to those that have purchased it and taken advantage of its capabilities. This is very effectively achieved by periodic peer review of the facility, as is done with national facilities. In the end, the facilities that operate ultra-scale computing systems should be judged in the same way other national facilities such as accelerators are judged. Typically, periodic peer review is used to assess whether they are meeting their missions and goals. Assessments evaluate and provide guidance in the areas such as:

- Does the facility operate well? Are the systems run well, are they reliable, is the facility meeting user expectations, etc.?
- Is the facility doing the appropriate research and development necessary to keep it at the forefront of its discipline?
- Is the facility doing what it can to ensure, in the aggregate, that the best science is being produced from its resources?

Such peer reviews have worked very well to ensure the effectiveness and efficiency of facilities that serve the targeted scientific community. The value of ultra-scale computing facilities and the scientific output of the systems should be evaluated in a similar manner.

There is no single metric for utilization because every platform manager, program, and complex problem to be solved is working towards specific (and somewhat different) objectives. The managers of the programs and the platforms must first define the overall value of the new tool in meeting objectives and then assess how successfully those objectives are being met with respect to the use of this sophisticated resource.

A NEW CONCEPTUAL APPROACH

General Hypothesis:
The consequence of maximizing node utilization is a slowdown of job completion time to a point that users are no longer productive.

Although some consideration of utilization is appropriate, a slowdown effect in the system can result when utilization is driven too hard (***Figure 5***). If the slowdown is significant, the effect of focusing on utilization can be counterproductive on overall system performance and on the ability of the system to be used for the type of applications for which its acquisition was justified. The Project team examined this general hypothesis for ultra-scale computing systems and the curve shown in ***Figure 5***. It was found that a "smoothly" running system (for example, ultra-scale computer systems) will find optimum utilization at the "knee" of the curve. One would want to increase utilization from small values until the slowdown becomes too large. Acceptable slowdown values may be different for different operations. Slowdown impacts user behavior which, in turn, affects the amount of load on the system (reduced utilization) and, more importantly, ultimately affects what the user is able to accomplish.

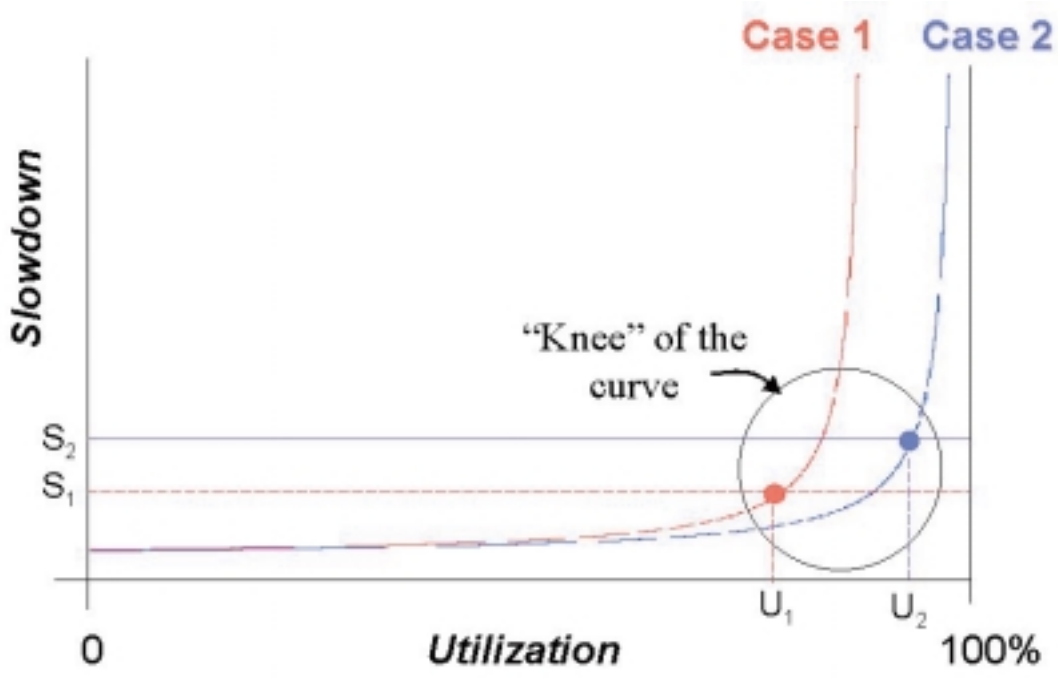


Figure 5: Slowdown vs. Utilization

Preliminary examination of the sample data showed that this normally expected "slowdown-utilization" curve does occur. The implication of this hypothesis is that systems operating at the "knee" are operating at the best range for those systems, that is, at the ideal point.

Trace Data Analysis

As part of the Project, trace data from several sites (see ***Table 3***) were collected and examined. It should be noted that the data did not cover the same time period or even the same length of time at each site. Furthermore, different machines collected the data, used different schedulers, and had different workloads. Because some of the trace data contained partially complete records, some information was lost as it was converted to a standard format. Despite all these differences across the data sets, a standard pattern was detected. Although acceptable slowdown associated with utilization was found to be near 60%, the data clearly show that there is no absolute acceptable utilization number. For the purpose of this section, instantaneous utilization is the percentage of total time that is used by running jobs – not percentage of available time or fraction allocated to jobs.

Before describing the analysis, it is important to highlight a standard queuing theory expectation. It is a well-established fact for service systems that the average response time increases as the "offered load" increases. The response time is flat until the load crosses a threshold. Then, the response time increases exponentially.

Table 3. The data analysis was based on trace logs from these sites.

Org	Machine	Max CPUs*	Period	#Jobs	#Queues
LANL	SGI Origin	2048	7/24/99 – 8/31/99	30,000	18
LLNL	SP-2	1344	Months	20,000	3
NASA	SGI Origin	256	Months	32,000	2
NPACI	SP-2	128	1/1/99 – 9/27/99	22,000	4
NPACI	T3E	272	5/1/99 – 9/27/99	5,000	40
NPACI	T90	14	1/1/99 – 9/27/99	25,000	45
NCSA	SGI Origin	512	6/30/99 – 7/30/99	10,000	36
NERSC	T3E	644	1/1/99 – 7/1/99	90,000	12

*These are the largest number of CPUs for which trace data were available, not the size of machine.

Since a high performance computer is an example of a service system, such a pattern should occur. In many systems, it is possible to submit jobs to "closed queues" that may not be "opened" for quite some time, for example, the weekend queue. For this and other reasons, the offered load was not used. Instead, the average system utilization during the lifetime of each job was measured. Utilization was taken to be the fraction of the total available CPU hours, during the lifetime of a job, that were being used to execute jobs.

Instead of response time, the related measure of *slowdown* was computed. Slowdown is defined as the elapsed job time (from submission to completion) divided by the run time. For example, a slowdown of two indicates that a job spent as much time waiting to run as it did actually being run. Some sites have job queues that are active only during certain time periods, such as late night and weekends. A job submitted on Monday, for a weekend queue, would incur at least a five-day waiting time. In this analysis, the submit time was changed to be just before the queue open time. Two other modifications were made to the data, (1) jobs with run times of less than one

minute were excluded, and (2) jobs with very high slowdown values (due either to queues that were turned on/off or due to an inability to determine exactly when a queue becomes active) were excluded. Both these job classes obscured the results. Finally, the average instantaneous utilization (considering all the included jobs) is noted on the plots below.

The plots that follow reveal that indeed, at higher utilization levels, the slowdown (for example, response time) does increase. It appears that the facility managers do try to keep the response time reasonable. There were two types of anomalous situations found. The first happens when the response time decreases at higher utilization levels. The other occurs when response time increases at lower utilization levels. Further investigation revealed that one must first separate the jobs into different classes because some systems have batch queues for large jobs, others for interactive daytime jobs, and even queues for very long, highly parallel jobs. The slowdown versus utilization curves all fit the same pattern but each has a different Desired Operation Range (DOR). When the analysis is focused on the important queues, most of the jobs are found to reside in the DOR.

Major conclusions to be drawn from the analysis of trace data are as follows:

- High-end and ultra-scale computer workloads exhibit a pattern of acceptable response time up to a certain instantaneous utilization level, which one refers to as the DOR. When instantaneous utilization is pushed higher than that level, average response time increases precipitously and to levels that negatively impact human productivity.
- For many of the systems studied and for the job classes that matter most, the DOR occurs around 60% instantaneous utilization.
- The location of the DOR can change through improvements in system software (for example, gang scheduling) and scheduler queues that are particularly well matched to the workload characteristics. Thus, more mature systems with more capable system software and a well-characterized workload can achieve desired operation ranges at higher instantaneous utilization levels in the later stages of the system life cycle.

The figures that follow show average slowdown as a function of system instantaneous utilization for individual sites. This requires some explanation. For each job, the average system instantaneous utilization was computed for the lifetime of that job, and the job was assigned to one of ten utilization buckets (from 10 to 100%). In addition, the slowdown for that job was calculated as the ratio of job lifetime divided by job runtime. Finally, the weight for each instantaneous utilization bucket was computed, expressed as a fraction of the whole weight, and displayed as the size of the bubble. Bubbles with high slowdown values indicate poor system response time. Bubbles with low utilization levels indicate poor system usage. Ideal performance has large bubbles.

A vertical line was drawn indicating the percentage of total node hours for all trace jobs divided by the total number of node hours in the time period. This line is not the average instantaneous utilization of the jobs in the curve, since there may be periods when the system was unusable.

The Site A plot reveals the characteristic rise with most of the big bubbles at the DOR of the curve. At eighty percent, many jobs are seen to suffer from a large slowdown value.

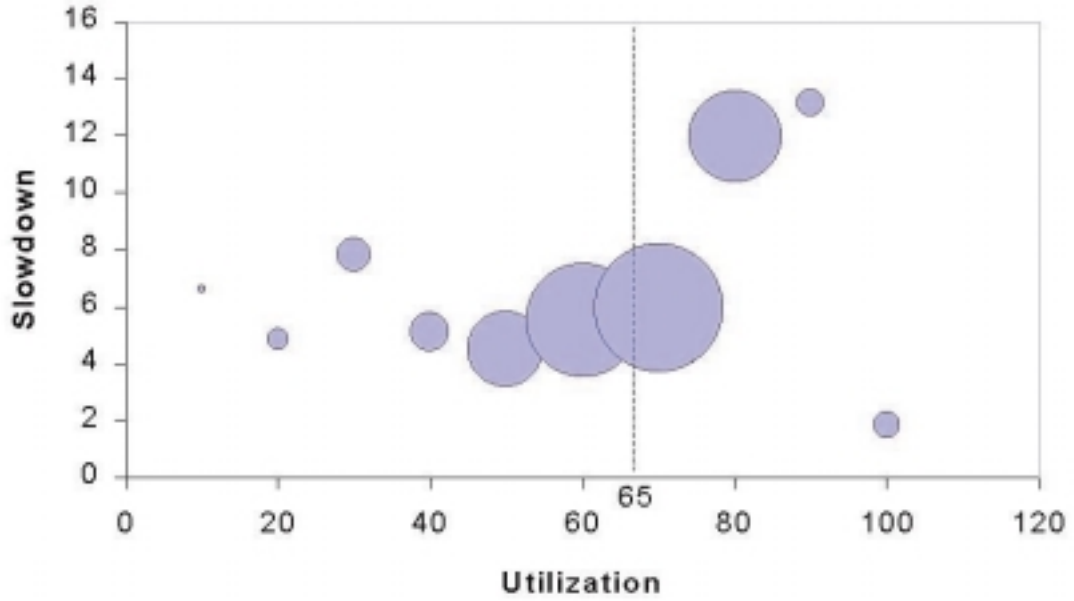


Figure 6: Site A - Slowdown vs. Utilization

The Site B curve looks very similar to the one before it, except that everything happens at a lower utilization level. At 60% utilization, the response time rises, so the DOR occurs at a lower utilization level.

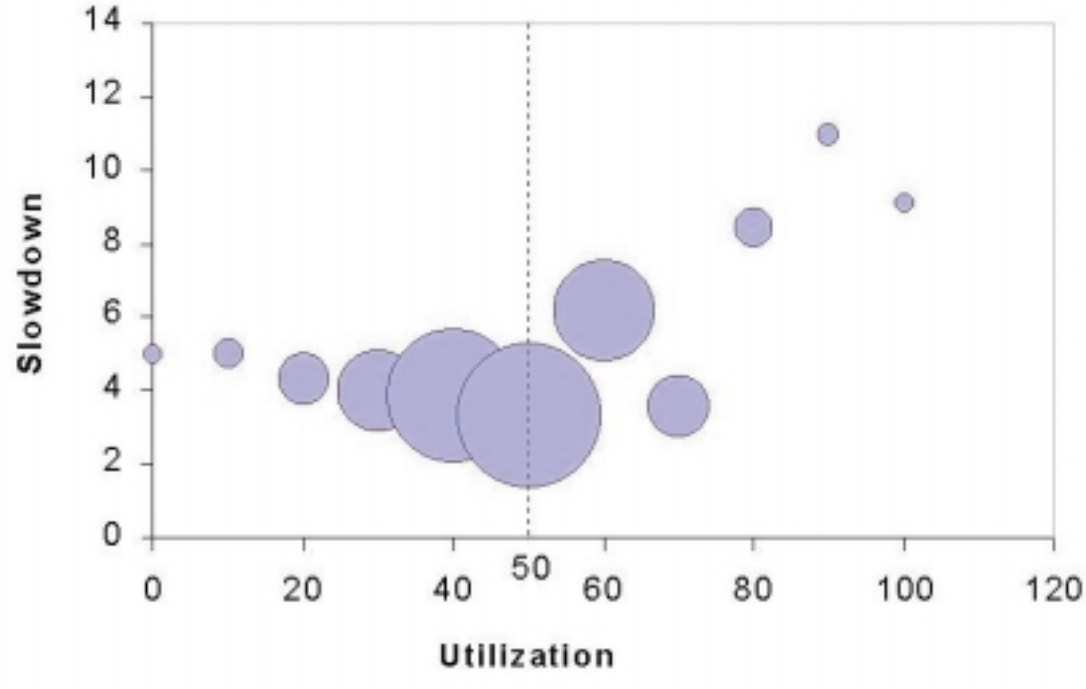


Figure 7: Site B - Slowdown vs. Utilization

The Site C curves below show a slightly different pattern. Slowdown values are very low and it is easy to see the increase at higher levels. As one can see, the vertical line appears more to the left than would be evident from the distribution of the bubbles. This is because either a long downtime or a short trace period exists. Site C1 has smaller jobs than Site C2; thus the desired operation ranges are in different places, although both seem to manage their systems very well.

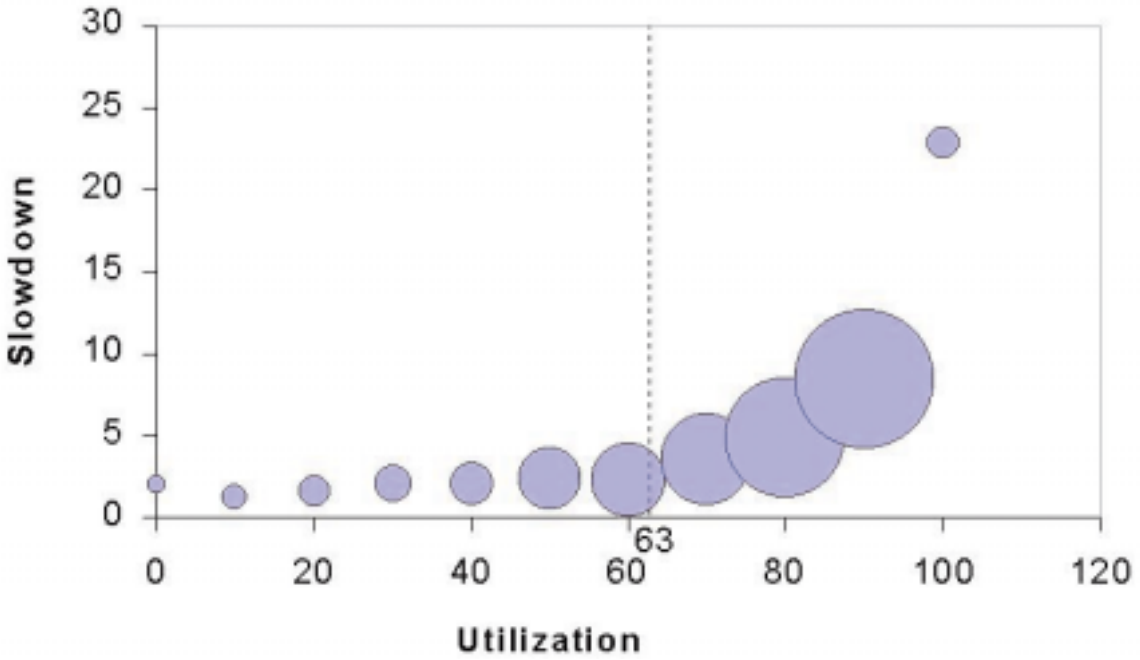


Figure 8: Site C1 - Slowdown vs. Utilization

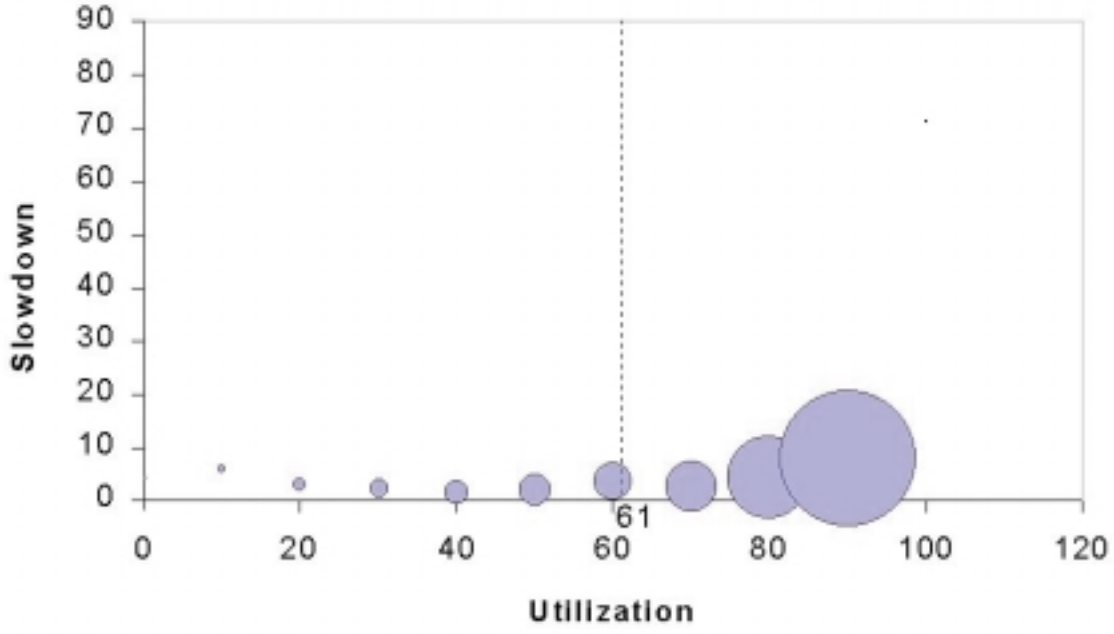


Figure 9: Site C2 - Slowdown vs. Utilization

The plot for Site D does not show the typical pattern. Most of the jobs have a low response time (not just most of the jobs, but most of the job weight). But there are high slowdown values at low instantaneous utilization values. The reason for this counterintuitive pattern is that there are a number of job classes that are overlaid in this data.

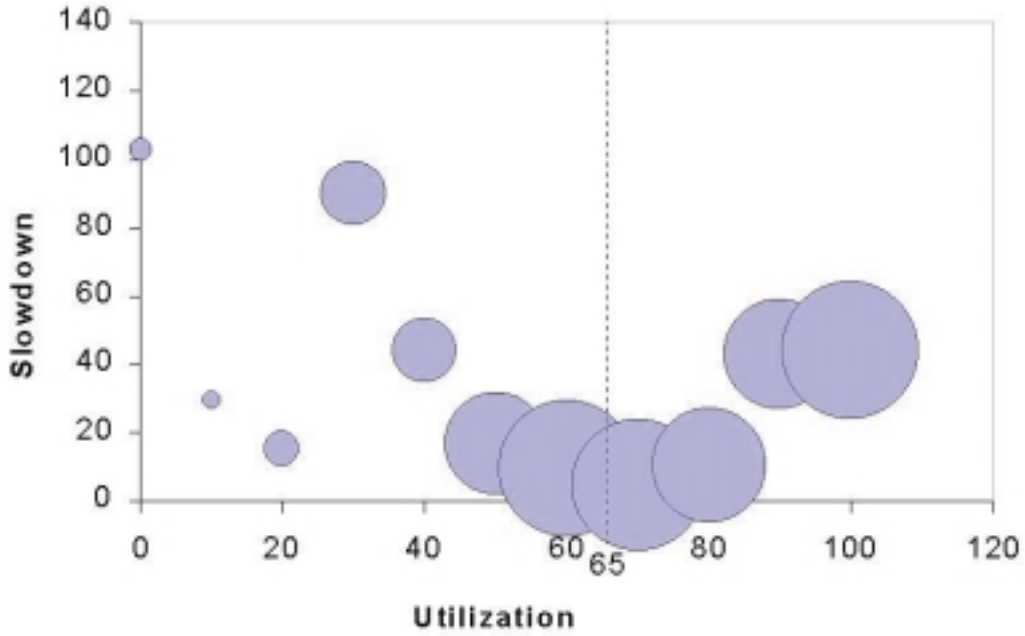


Figure 10: Site D - Slowdown vs. Utilization

Finally, a yet different phenomenon is observed for Site E. The bubbles appear to have no real pattern. When the plots for the individual job classes are examined (Figures 11a and 11b), however, it is evident that in each case the plots follow the usual pattern. The instantaneous utilization appears a bit on the high side for the response time.

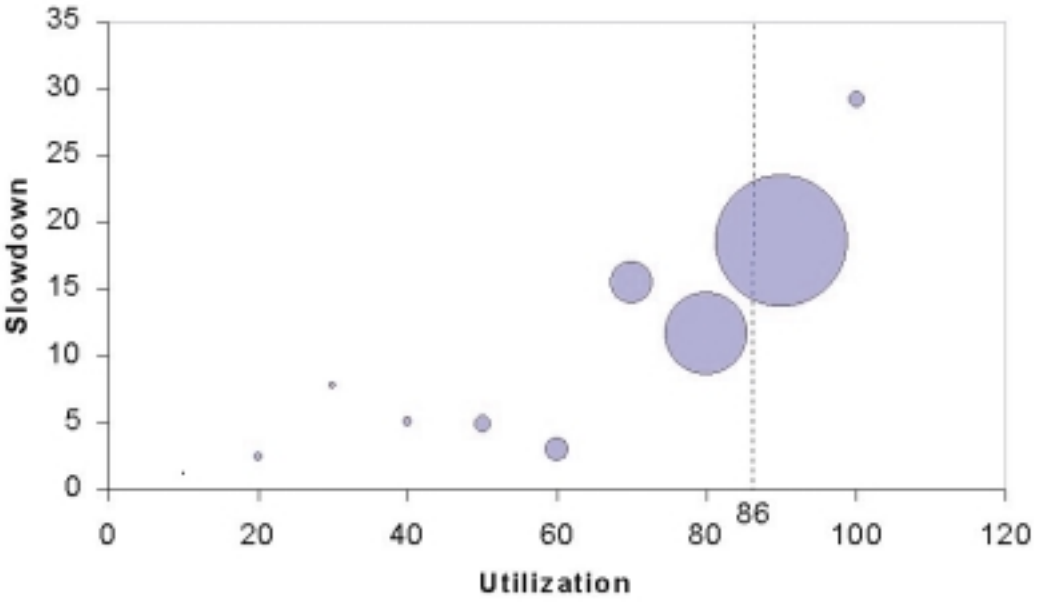


Figure 11: Site E - Slowdown vs. Utilization

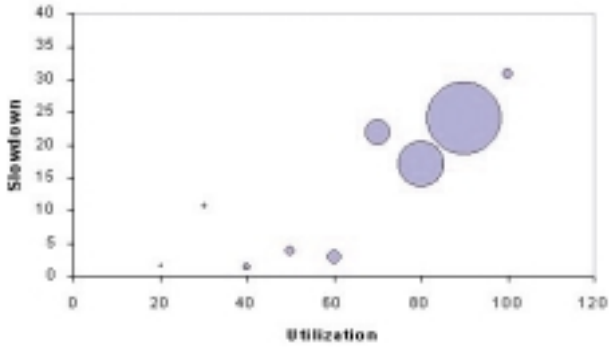


Figure 11a: Site E1 – Slowdown vs. Utilization

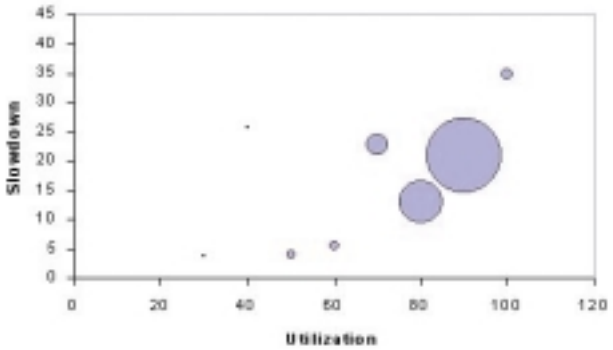


Figure 11b: Site E2 – Slowdown vs. Utilization

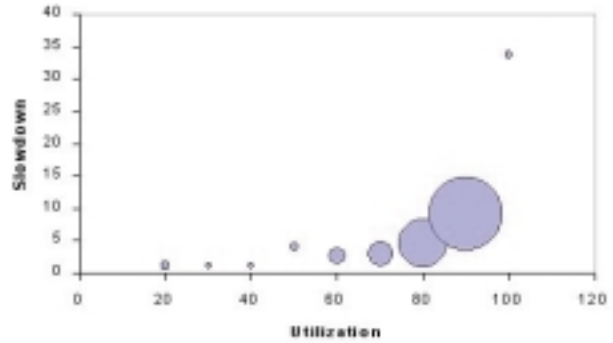


Figure 11c: Site E3 – Slowdown vs. Utilization

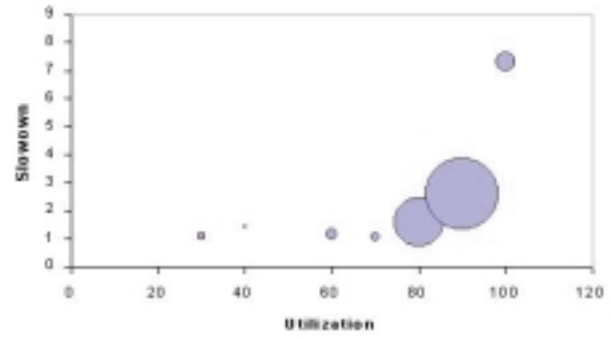


Figure 11d: Site E4 – Slowdown vs. Utilization

Recommended Approach

After discussing the different ways that Project participants measure the utilization and assess the overall value of advanced computing platforms, participants agreed that the ultra-scale computing community could and should come to an agreement on a set of metrics and respective definitions. Project participants recommend that advanced computing platform managers test the metrics for a limited period of time to ensure validity. Once the metrics are determined to be valid, the recommendation is that the high performance computing community adopt and use these metrics to assess system usefulness. This means actually making the system measurements needed for the metrics. It is proposed that facilities report the result, perhaps annually at gatherings such as the annual Supercomputing meeting, and/or post the results on a common web site maintained for this purpose. The results of these measurements should be used in future decision-making.

*Recommended Approach***:**

- Agree on a set of defined metrics
- Test the metrics for a limited period of time before accepting them
- Adopt the defined metrics, report against these metrics, and use in future decision-making
- Consider actions that can be taken to enhance the value to the customer:
  - Develop and use better schedulers
  - Put small jobs on more appropriate systems
  - Monitor the metrics
  - Assign mission requirements top priority
  - Survey the users
  - Assess system balance

In addition, suggestions are offered for changes that can be made to ultra-scale computing system characteristics to enhance the value of the system to the customer. Changes include designing better job schedulers, running systems in a way that increase utilization without adversely impacting the primary mission of the facility, and being willing to collect and monitor job trace and other data needed for the utilization metrics. Project participants further encourage managers of advanced computing systems to continue to assign top priority to mission requirements and to survey users on a regular basis, modifying operational practices as needed to meet key users' needs.

NERSC's Effective System Performance (ESP) Test

One activity currently under way to provide more meaningful measures of system-level performance is the ESP Test. This test is presently under development by a group of researchers in the NERSC of Lawrence Berkeley Laboratory. By analogy to “theoretical peak” performance versus “sustained performance on an individual benchmark," the ESP test attempts to distinguish the “theoretical maximum” system capability from “measured” system capability. In particular, the ESP measures the efficiency of the job scheduling system and the ease with which the system can handle large configuration jobs and typical system management tasks.

The ESP test, as it is currently formulated, employs a set of 82 individual jobs, consisting of multiple copies and different sized versions of nine different application programs. Two of these jobs are "full configuration" (FC) jobs, namely calculations that use the entire system. The remaining jobs are submitted to the system's job scheduling software in an order given by a particular pseudo-random number generator, at certain staggered times. The first of the two FC jobs is submitted at a certain predetermined time in the test. This job is to be run immediately upon submission. Immediately after the FC job is completed, the system is entirely shut down and then rebooted. After the system has been rebooted, the suite shall be restarted, and then at a certain time, the second FC job is submitted. The objective of the test is to minimize the total elapsed wall clock time until all jobs are completed. The ESP test is illustrated in ***Figure 12***. More information on the ESP test can be found at http://www.nersc.gov/~dhbailey.

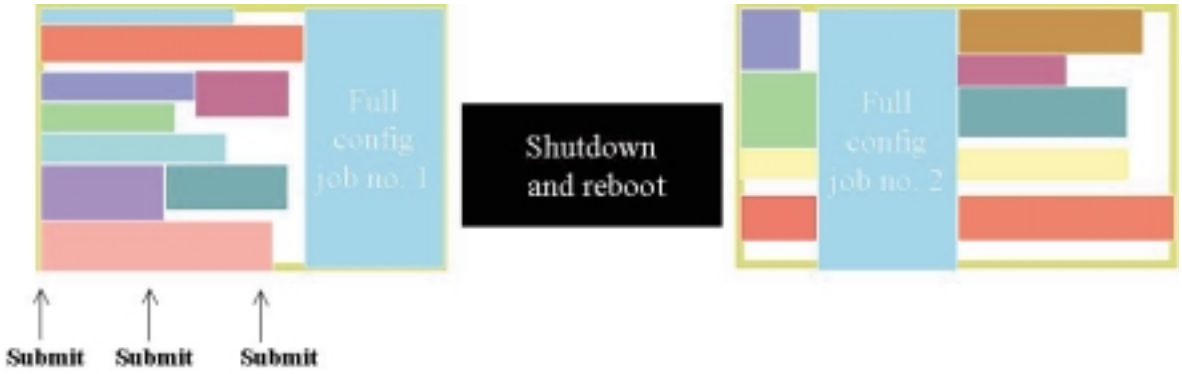


Figure 12: ESP Test Timeline

Once the test run is complete, a system effectiveness ratio E is computed as

$$E = \frac{p_1 t_1 + p_2 t_2 + \ldots + p_n t_n}{P\ (S + T)}$$

where

p_i = Number of processors utilized by job i
t_i = Wall clock run time in seconds required by job i on a dedicated system
n = Number of individual jobs run during the test
P = Total number of processors in the system
S = Time required for shutdown and reboot
T = Averaged wall clock run time (not counting shutdown and reboot)

Note that if the job scheduling system is perfect, if shutdown and reboot times are zero, and if there is no system contention among concurrently running jobs, E would be unity. On real-world systems, E will be less than unity. When multiplied by a sustained performance figure obtained by using a standard multiprocessor benchmark, for example, one acquires an "effective system performance" statistic, which is intended to characterize real-world system-level performance.

This effort is an example of a specific approach that can be taken to measure system effectiveness.

RECOMMENDATIONS AND CONCLUSIONS

Project participants recommend that changes such as the following be considered by the ultra-scale computing community to better measure system usefulness and assess the value of advanced platforms:

- Change the day-to-day operations of ultra-scale platforms so that the use of this sophisticated resource lies close to the left side of the "knee" of the "slowdown vs. utilization" curve ***(Figure 5)*** shown earlier in this paper. Exact changes to be made, both near-term changes in practice and longer-term modifications to policy, should be dictated by the mission requirements of each computing platform and the particular needs of its key users.
- Undertake concerted efforts to better characterize the workloads for each platform. Managers must thoroughly understand the sizes and types of jobs that are to run on their ultra-scale platforms in order to move utilization towards the optimal point on the curve for each site.
- Design benchmark suites and performance models to predict the effectiveness of the systems instead of solely measuring utilization after the fact. It is recommended that the results of the ESP effort mentioned in the previous section, for example, be brought to the attention of advanced platform managers in government, industry, and at educational institutions.
- Adjust the configuration of advanced computing platforms. This may require a re-balancing among processor, memory, and interconnect bandwidth capabilities to better address the specific job mix of the particular computing system. It is recognized that additional resources may be required.
- Recognize the importance of undertaking needed research in key areas such as:
  - Designing more efficient scheduling algorithms. Determine how to incorporate other job parameters (memory usage, interruptible jobs, and dynamic node size flexibility) into the scheduling software.
  - Techniques for increasing the efficiency of applications by employing moldable and malleable jobs. What percentage of user jobs could be configured, with reasonable effort, as interruptible, as adjustable CPUs at job launch time, or as adjustable CPUs during execution? What improvement in system efficiency could be achieved by means of such changes?
  - Analyzing performance statistics across a wider variety of system resources.
- Establish performance requirements for collecting trace and other data as part of future procurements and encourage the inclusion of better schedulers and other tools in the system.
- Require system vendors to provide access to more system statistics as a performance specification.

- Develop procurement specifications for future purchases of advanced platforms so that the importance of valuation metrics and related system software is evident to the vendors.
- Ensure open and ongoing dialogues with platform users and make changes needed to increase the ability of the users to benefit from new capabilities afforded by massively parallel systems.
- Assess the potential impact of various system enhancements, such as faster job launch, checkpoint-restart, gang scheduling, more sophisticated job scheduling, and backfill schemes. Which is likely the most cost-effective? Encourage software developments in these areas. Establish ways to share the testing and validation needed across sites to develop full production tools.
- Change scientific applications codes to take advantage of new ultra-scale computing resource capabilities.
- Establish a balanced investment strategy for obtaining and managing ultra-scale computing platforms that seeks to both improve scheduling and increase the efficiency of individual jobs.

Recommendations

- Change the day-to-day operations to operate at the knee of the curve
- Characterize the "workloads" for each platform
- Design benchmark suites and performance models to predict system effectiveness
- Adjust system configuration (processor, memory and BW) to address the specific job mix
- Undertake needed research:
  - Efficient scheduling algorithms
  - Techniques for moldable and malleable jobs
  - Analyze performance statistics across a wider variety of system resources
- Change procurement specifications:
  - Emphasize the importance of valuation metrics
  - Include development of better schedulers and tools
  - Access to more system statistics for performance assessments
- Ensure dialogue with "Users"
- Change scientific applications codes to take advantage of new resource capabilities
- Encourage software developments in tools such as schedulers and workload generators
- Establish a balanced investment strategy for obtaining and managing ultra-scale computing systems

The following research questions were identified during the Project:

- Which system software enhancements and operating strategies have the greatest potential for improving system effectiveness?
- How do the job size/run time distributions vary between various HPC centers? How do these distributions vary between different disciplines?
- What percentage of jobs has power-of-two CPU sizes?
- What impact does job size distribution have on system-level efficiency?

The purpose of the Ultra-Scale Computing Valuation Project has been achieved; acceptable ways to evaluate "ultra-scale" computing systems are being defined and a consensus on these approaches is emerging within the ultra-scale computing community. Reaching agreement on understandable and defensible measures is important to the large-scale computing research and applications programs in government and academia, such as the ASCI program at DOE and the PACI program at NSF, as well as others. Presently, generally accepted metrics do not exist for this evaluation. It is evident, however, that the answer is not found by merely assessing node utilization.

The Project Co-Chair's contention given all the system limitations and constraints is that "things are in good shape as far as the running of the advanced computing platforms is concerned," is based on sound peer review of the approaches currently used to manage utilization of advanced computing platforms. Participants agree that a balance of research, development, and implementation considerations is necessary, but argue that the success of high-end computing efforts aimed at enabling new classes of applications should be measured primarily by whether the use does, in fact, result in new knowledge. If so, then the advanced computing tools used were worth the investment.

> "After looking at real data, I must say that given what is possible now, things are in good shape as far as the running of the advanced computing platforms is concerned."
>
> Larry Rudolph, MIT

Participants agree that the ultra-scale computing community should focus on creating the right-size tools for every scientific and programmatic mission. There is recognition of the responsibility of computing systems managers and the overseeing agencies to determine how best to measure the overall value of each system to its users. In addition, ways must be defined to make needed measurements and compare against recognized benchmarks and to establish operational practices that are optimal for each site and the scientific goals that site is designed to achieve.

APPENDIX

A.1. Project meeting dates, places, and agendas

ULTRA-SCALE COMPUTING VALUATION "BRAINSTORMING SESSION"
PARK CITY, UTAH
JUNE 24-25, 1999

Thursday, June 24, 1999

8:30am	Goals for the Ultra-Scale Computing Valuation, Paul H. Smith, DOE
9:00	ASCI Interests and Involvement, Paul Messina, DOE
9:30	Setting the Stage on Technical Issues, Larry Rudolph, MIT
10:30	How Deep and How Wide Should We Go? (Brainstorming on Scope of Effort)
11:30	What is Realistic? (Brainstorming on Metrics)
Noon	Working Lunch
1:30	Where Do We Go From Here? Discuss/Identify Next Event(s)/Step(s)
2:30	Identify Individuals to be Engaged in Next Step(s)/Event(s)
3:45	What Would be the Outcome of Our Suggestions? What are the Potential Impacts?
4:30	Where are We So Far? What Else Needs to be Considered?
5:15	Adjourn
7:00	Working Dinner

Friday, June 25, 1999

8:30am	Flesh Out First Day's Actions and Outcomes — Any Second Thoughts? Do We Still Think It is Good?
11:00	Summarize Results
11:30	What is Next? Who is Responsible? When is it Due?
Noon	Adjourn

TRACE-DATA ANALYSIS WORKING SESSION
DENVER, CO
AUGUST 20, 1999

Friday, August 20, 1999

8:30am	Ultra-Scale Computing Valuation & Activities to Date, Paul Smith
8:50	Proposed Metrics, John Towns
9:05	Basis for Trace Data Analyses, Larry Rudolph
9:25	SDSC NPACI System Workload, Mike Vildibill
9:45	LLNL Configuration and Workload Characteristics, Moe Jette
10:05	Workload Information for DOE ASCI Red, Jim Tomkins
11:00	LANL System Workload Characteristics, Tom Klingner
11:30	LBL NERSC System and Workload Characteristics, Bill Kramer
Noon	Working Lunch
12:30	NASA/ARC System Workload, Chris Kuszmaul
12:50	NCSA Data Analysis Results, John Towns
1:10	Comparisons & Conclusion of Sites' Data Analyses, Larry Rudolph
2:00	Proposed Draft Product Write-up, Paul Smith
3:00	Discussion on Draft Product Write-up, Paul Smith
3:30	Writing Assignments for Sites' Sections Write-up, Larry Rudolph
4:30	Action Items – Things to Do Next, Paul Smith

ULTRA-SCALE COMPUTING VALUATION SUMMIT
PROVIDENCE, RI
OCTOBER 5-6, 1999

Tuesday, October 5

9:00am Opening Remarks and Introductory Comments, Paul H. Smith and Larry Rudolph

9:30 Values for Ultra-scale Computing Systems**,** Paul Messina, lead

PANEL: Addressing the Importance of Valuation Metrics to the Ultra-Scale Computing Community
Dan Hitchcock, John Towns, Mike Vildibill

11:30 Historical Measurements, Bill Nitzberg

Noon Working Lunch

1:00 Metrics and Measurements, John Towns, lead

PANEL: Addressing operations at LANL, LLNL, SNL, NERSC, NCSA and SDSC
Mike Vildibill, Jim Tomkins, Bill Kramer, Moe Jette, Tom Klingner, Larry Rudolph

4:00 Effective System Performance, Bill Kramer
Workload Characterization: The Need and the Challenge, Dror Feitelson

5:30 Adjourn

Wednesday, October 6

9:00am Recommendations, Larry Rudolph, lead

PANEL: Addressing What Technically Should be Done to Improve Efficiency

10:30 What Changes Could Be Made? David Bailey, lead

PANEL: Views on potential actions to be taken by Federal Programs
Dan Hitchcock, Thuc Hoang, Larry Davis, David Greenberg, Bill Nitzberg

11:30 Conclusions, Larry Rudolph

11:50 Closing Comments, Paul H. Smith

Noon Adjourn

BIRDS-OF-A-FEATHER SESSION
PORTLAND, OR
NOVEMBER 17, 1999

Wednesday, November 17

5:30pm Opening Remarks and Introductory Comments, Paul H. Smith
Importance of Metrics to Ultra-Scale Computing Community, Paul Messina
Metrics (the good and the bad), John Towns
Trace Data and the Results, Larry Rudolph
Potential Actions, David Bailey
Group Discussion and Suggestions, Ian Foster, all
Closing Comments, Paul H. Smith

A.2. Trace data collection and recommended standard format

- The Standard Workload Format
  ==========
  The standard workload format abides by the following principles:
  The files are portable and easy to parse:
  - Each workload is stored in a single ASCII file.
  - A single line in the file represents each job (or roll).
  - Lines contain a predefined number of fields, which are mostly integers, separated by white space. Fields that are irrelevant for a specific log or model appear with a value of -1.
  - Comments are allowed, as identified by lines that start with a ';'. In particular, files are expected to start with a set of header comments that define the environment or model.
  - The current version, described here, is version 2.1

- The log file consists of a header section and a data section

Header Comments (each line begins with a ;)
==========
The first lines of the log should be of the comments with the format ";Label: Value". These are special header comments with a fixed format, used to define global aspects of the workload. Predefined labels are:

1. Version: Version number of the standard format that the file uses. The format described here is version 2.
2. Computer: Brand and model of computer
3. Installation: Location of installation and machine name
4. Acknowledge: Name of person(s) to acknowledge for creating/collecting the workload.
5. Information: Web site or email that contain more information about the workload or installation.
6. Conversion: Name and email of whoever converted the log to the standard format.
7. UnixStartTime: When the log starts, in Unix time (seconds since the epoch)
8. TimeZone: a value to add to times given seconds since the epoch. The sum can then be fed into gmtime (Greenwich time function) to get the correct date and hour of the day. The default is 0, and then gmtime can be used directly. Note: do not use localtime, as then the results will depend on the difference between your time zone and the installation time zone.
9. StartTime: When the log starts, in human readable form, in this standard format: Tuesday, 1 Dec 1998, 22:00:00
10. EndTime: In the same format as StartTime.
11. MaxNodes: Integer, number of nodes in the computer (describe the sizes of partitions in parentheses if applicable).
12. MaxRuntime: Integer, in seconds. This is the maximum that the system allowed, and may be larger than any specific job's runtime in the workload.
13. MaxMemory: Integer, in kilobytes. Again, this is the maximum the system allowed.
14. Allow Overuse: Boolean. 'Yes' if a job may use more than it requested for any resource, 'No' if it can't.
15. Queues: A verbal description of the system's queues. Should explain the queue number field (if it has known values). As a minimum it should be explained how to tell between a batch and interactive job.
16. Partitions: A verbal description of the system's partitions, to explain the partition number field. For example, partitions can be distinct parallel machines in a cluster, or sets of nodes with different attributes (memory configuration, number of CPUs, special attached devices), especially if the scheduler knows this.
17. Note: There may be several notes, describing special features of the log. For example, ``The runtime is until the last node was freed; jobs may have freed some of their nodes earlier''.

The Data Fields

==========

(Use a -1 to indicate a blank or missing data for a field; minimum of 18 fields, max of 20 or more if explained in header)

1. Job Number — a counter field, preferably from 1.
2. Submit Time — in seconds. The earliest time the log refers to is zero, and is the submittal time of the first job. The lines in the log are sorted by ascending submittal times.
3. Wait Time — in seconds. The difference between the job's submit time and the time at which it actually began to run. For jobs submitted to queues that are closed, this time should be from the time the queue is opened.
4. Run Time — in seconds. The wall clock time the job was running (end time minus start time). Note that when values are rounded to an integral number of seconds (as often happens in logs) a run time of 0 is possible and means the job ran for less than 0.5 seconds. On the other hand, it is permissible to use floating point values for time fields.
5. Number of Allocated Processors — an integer. In most cases this is also the number of processors the job uses; if the job does not use all of them, we typically don't know about it.

* 6. Average CPU Time Used — both user and system, in seconds. This is the average over all processors of the CPU time used, and may therefore be smaller than the wall clock runtime. If a log contains the total CPU time used by all the processors, it is divided by the number of allocated processors to derive the average.

* 7. Used Memory — in kilobytes. This is again the average per processor.

8. Requested Number of Processors.
9. Requested Time. This can be either runtime (measured in wallclock seconds), or average CPU time per processor (also in seconds) — the exact meaning is determined by a header comment. If a log contains a request for total CPU time, it is divided by the number of requested processors.

*10. Requested Memory (again kilobytes per processor).

11. Completed? 1 if the job was completed, 0 if it was killed. If information about checkpointing or swapping is included, other values are also possible (see below).
12. User ID — a natural number, between one and the number of different users.

*13. Group ID — a natural number, between one and the number of different groups. Some systems control resource usage or prio by groups rather than by individual users.

*14. Executable (Application) Number — a natural number, between one and the number of different applications appearing in the workload. In some logs, this might represent a script file used to run jobs rather than the executable directly; this should be noted in a header comment.

15. Queue Number — a natural number, between one and the number of different queues in the system. The nature of the system's queues should be explained in a header comment. This field is where batch and interactive jobs should be differentiated: we suggest the convention of denoting interactive jobs by 0.
16. Partition Number — a natural number, between one and the number of different partitions in the systems. The nature of the system's partitions should be explained in a header comment. For example, it is possible to use partition numbers to identify which machine in a cluster was used.

*17. Preceding Job Number — this is the number of a previous job in the workload, such that the current job can only start after the termination of this preceding job. Together with the next field, this allows the workload to include feedback as described below.

18. Think Time from Preceding Job — this is the number of seconds that should elapse between the termination of the preceding job and the submittal of this one.
19. Job Weight — or job priority. 1 is lowest priority.
20. Number of Processors actually used — typically, the number of requested processors is equal to the number of allocated processors, which is equal to the number of processors used. Some accounting systems have all three numbers available.

If a log contains information about checkpoints and swapping out of jobs, a job can have multiple lines in the log. In fact, we propose that the job information appear twice. First, there will be one line that summarizes the whole job: its submit time is the submit time of the job, its runtime is the sum of all partial runtimes, and its code is 0 or 1 according to the completion status of the whole job. In addition, there will be separate lines for each instance of partial execution between being swapped out. All these lines have the same job ID and appear consecutively in the log. Only the first has a submit time; the rest only have a wait time since the previous burst. The completed code for all these lines is 2, meaning ``to be continued''; the completion code for the last such line is 3 or 4, corresponding to completion or being killed. It should be noted that such details are only useful for studying the behavior of the logged system, and are not a feature of the workload. Such studies should ignore lines with completion codes of 0 and 1, and only use lines with 2, 3, and 4. For workload studies, only the single-line summary of the job should be used, as identified by a code of 0 or 1.

Sample log file
==========

```
; Version: 2
; Computer: IBM SP2
; Installation: LLNL ASCI Blue
; Acknowledge: Moe Jette
; Information: http://www.llnl.gov/
; Conversion: Dror Feitelson (feit@cs.huji.ac.il) Aug 1 1999
; UnixStartTime: 926348742
; TimeZone: -28800
; StartTime: Monday, 10 May 99, 07:05:42
; EndTime: Monday, 7 Jun 99, 07:04:50
; MaxNodes: 336 (23 admin/I/O, 8 debug, 305 batch; 4 processors/node)
; Partitions: 1=batch, 2=debug
; Information: 332 MHz PowerPC 604e processors
; Information: 1536 MB memory per node
; Information: 8AM - 5PM: jobs limited to 2hr, 128 nodes
;              5PM - 8AM: jobs limited to 8hr, 256 nodes
;              5PM fri - 8AM mon: jobs limited to 12hr, 256 nodes
```

A.3. Supporting Documents and Reading Materials

Various materials, reports and documents, including this report can be found at this web address:

www.dp.doe.gov/valuation

We call your attention to the following, additional relevant papers:

- *Evaluating System Effectiveness in High Performance Computing Systems*, Adrian T. Wong, Leonid Oliker, William T. C. Kramer, Teresa L. Kaltz and David H. Bailey (1999)
- *HPC Modernization Metrics Discussion Paper,* DOD (1999)
- *Job Scheduling Strategies for Parallel Processing*, Edited by Dror Feitelson and Larry Rudolph, Lecture Notes in Computer Science, Volume 1659, Springer Publishers (1999)
- *LLNL Configuration and Workload Characteristics*, Moe Jette, LLNL (1999)
- *Tri-Lab Concept Paper on Supercomputer Utilization,* written by experts from LANL, LLNL, SNL (1999)
- *Scheduling for Parallel Computing: A Historical Perspective of Achievable Utilization*, James Patton Jones and Bill Nitzberg, NASA Ames Research Center (1998)
- *The Future of Scientific Computing for Grand Challenge Problems*, Andrew Arvind and John Marshall, Massachusetts Institute of Technology (1996)
- See also: Parallel Workloads Archive at http://www.cs.huji.ac.il/labs/parallel/workload

ACRONYM GLOSSARY

Alliance.....National Computational Science Alliance
ARC .....Ames Research Center
ASCI.....Accelerated Strategic Computing Initiative
ASCII.....American Standard Code for Information Interchange

CCS .....Center for Computational Sciences
CPU ..... Central Processing Unit

DoD.....Department of Defense
DOE .....Department of Energy
DOR ..... Desired Operation Range

ESP.....Effective System Performance

FC .....Full Configuration

GSFC ..... Goddard Space Flight Center

HEC .....High End Computing
HPCMO.....High Performance Computing Modernization Office

LANL ..... Los Alamos National Laboratory
LBL.....Lawrence Berkeley National Laboratory
LES ..... Leading Edge Site
LLNL .....Lawrence Livermore National Laboratory

MIT..... Massachusetts Institute of Technology
MPP ..... Massively Parallel Processors

NAS..... Numerical Aerospace Simulation Facility
NASA .....National Aeronautics and Space Administration
NCSA ..... National Center for Supercomputing Applications
NERSC.....National Energy Research Scientific Computing Center
NPACI ..... National Partnership for Advanced Computational Infrastructure
NSF..... National Science Foundation

PACI..... Partnership for Advanced Computational Infrastructure

ROI..... Return on Investment

SDSC ..... San Diego Supercomputer Center
SNL .....Sandia National Laboratories

VMR ..... Virtual Machine Room

ACKNOWLEDGMENTS

A debt of gratitude is owed to the members of the "Valuation Project Team" for the significant amount of quality time and energy given in support of this effort. The editors also extend thanks to those individuals who participated in one or more of the meetings, or otherwise made contributions to the discussions on this topic. We especially thank Tina Macaluso for her Herculean effort in documenting the discussions in the various meetings, assistance in generating the early drafts, and in editing the Report. We also thank Mary Goroff and Michael Mishoe for all their help with the many and varied logistics for this Project. In addition, we thank Debra Rubin-Bice for her assistance in preparing the various versions of the document and updating the web site for this body of work.

Paul H. Smith

Larry Rudolph

ULTRA-SCALE COMPUTING VALUATION PROJECT PARTICIPANTS

David Bailey ..... Lawrence Berkeley National Laboratory/NERSC
Bill Blake ..... Compaq Computer Corporation
Larry Davis ..... Department of Defense/HPCMO
Dror Feitelson ..... Hebrew University
Tom Formhals ..... SUN Microsystems, Inc.
Ian Foster ..... Argonne National Laboratory
Mary Goroff ..... California Institute of Technology
David Greenberg ..... Institute for Defense Analysis/CCS
Dan Hitchcock ..... U.S. Department of Energy, Office of Science
Thuc Hoang ..... U.S. Department of Energy, Defense Programs
Morris A. Jette, Jr. ..... Lawrence Livermore National Laboratory
Thomas Klingner ..... Los Alamos National Laboratory
William T.C. Kramer ..... Lawrence Berkeley National Laboratory/NERSC
Chris Kuszmaul ..... National Aeronautics and Space Administration/ARC
Tina Macaluso ..... Science Applications International Corporation
Phil Merkey ..... National Aeronautics and Space Administration/GSFC
Paul Messina ..... U.S. Department of Energy, Defense Programs
Michael Mishoe ..... U.S. Department of Energy, Defense Programs
Dale Nielsen, Jr. ..... Lawrence Livermore National Laboratory
Bill Nitzberg ..... National Aeronautics and Space Administration/ARC
Stephen Perrenod ..... SUN Microsystems, Inc.
Dan Reed ..... University of Illinois, Champaign-Urbana
Ralph Roskies ..... Pittsburgh Supercomputing Center
Larry Rudolph ..... Massachusetts Institute of Technology
Steve Scott ..... Silicon Graphics Computer Systems
Paul H. Smith ..... U.S. Department of Energy, Defense Programs
Carol Sutcliffe ..... International Business Machines Corporation
James L. Tomkins ..... Sandia National Laboratories
John Towns ..... National Center for Supercomputing Applications
Mike Vildibill ..... San Diego Supercomputer Center
Manuel Virgil ..... Los Alamos National Laboratory
Gil Weigand ..... U.S. Department of Energy, Defense Programs