CTWatch QUARTERLY

http://www.ctwatch.org/quarterly/ Volume 1 Number 1 February 2005

TRENDS IN HIGH PERFORMANCE COMPUTING

Cyberinfrastructure Technology Watch
http://www.ctwatch.org/

Opening Message

Welcome to the inaugural issue of the *Cyberinfrastructure Technology Watch Quarterly*. We hope that this publication provides a window to the future of the hardware, software and human resources required to build a useful, usable and enabling cyberinfrastructure for science and engineering. Our goal is to provide a venue for describing emerging cyberinfrastructure technologies and discussing current trends and future opportunities, critical information relevant to the building and using of cyberinfrastructure, and a resource for the entire community.

In 2003, the NSF Blue Ribbon Panel on Cyberinfrastructure provided a compelling vision of the future:

> *"... a new age has dawned in scientific and engineering research, pushed by continuing progress in computing, information, and communication technology, and pulled by the expanding complexity, scope, and scale of today's challenges. The capacity of this technology has crossed thresholds that now make possible a comprehensive "cyberinfrastructure" on which to build new types of scientific and engineering knowledge environments and organizations and to pursue research in new ways and with increased efficacy."*

The Blue Ribbon Panel's vision of cyberinfrastructure involves the coordination and integration of software, hardware and human resources to enable today's and future science and engineering applications. Coordinating compute, data, visualization, networking, field instruments, and other technologies presents enormous challenges to cyberinfrastructure builders and developers and pushes many of the component technologies to the limits.

Pioneers in the building and using of cyberinfrastructure have included a collection of advanced multi-disciplinary application projects including NSF's Network for Earthquake Engineering Systems (NEES) project (focusing on the development of enabling infrastructure for critical earthquake engineering experiments) and NIH's Biomedical Informatics Research Network (BIRN) project (focusing on distributed collaborations in brain imaging, human neurological disorders, and associated problems in animals); the development of community databases and data collections (the National Virtual Observatory data collection provides a comprehensive window of the heavens while the Protein Data Bank provides a global resource for protein information), and visionary technology-oriented projects such as OptIPuter (which is experimenting with a new generation of super optical networks). Perhaps the most visible project has been the Extensible Terascale Facility (ETF or TeraGrid), which involves a broad spectrum of national partners in the largest-scale, coordinated project to build and operate a production grid to date.

All of these projects demonstrate that the vision of a national cyberinfrastructure articulated by the Blue Ribbon Panel is complex and compelling, with both unprecedented opportunities and unprecedented challenges. Building a useful, usable and enabling cyberinfrastructure environment requires careful design and coordinated development, deployment and support of a robust set of integrated cyberinfrastructure technologies. Strategic investments and commitments will be required to achieve the vision laid out by the Blue Ribbon Panel.

Our goal is for the *Cyberinfrastructure Technology Watch (CTWatch) Quarterly* to provide a strategic resource for community efforts building the emerging cyberinfrastructure. We hope that you will find this inaugural issue and subsequent issues thought-provoking, illuminating and entertaining reading, and we hope that you will contribute to the community discussion on these critical topics. We look forward to your input.

CTWatch Quarterly Publishers

Fran Berman
Director, San Diego Supercomputer Center

Thom Dunning
Director, National Center for Supercomputing Applications

The NRC Report on the Future of Supercomputing

Susan L Graham
UNIVERSITY OF CALIFORNIA AT BERKELEY

Marc Snir
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Background

A variety of events led to a reevaluation of the United States supercomputing programs by several studies in 2003 and 2004. The events include the emergence of the Japanese Earth Simulator in early 2002 as the leading supercomputing platform; the near disappearance of Cray, the last remaining U.S. manufacturer of custom supercomputers; some criticism of the acquisition budgets of the Department of Energy's (DOE) Advanced Simulation and Computing (ASC) program; and some doubts about the level and direction of supercomputing R&D in the U.S. We report here on a study that was conducted by a committee convened by the Computer Science and Telecommunications Board (CSTB) of the National Research Council (NRC). It was chaired by Susan L. Graham and Marc Snir; it had sixteen additional members with diverse backgrounds: William J. Dally, James W. Demmel, Jack J. Dongarra, Kenneth S. Flamm, Mary Jane Irwin, Charles Koelbel, Butler W. Lampson, Robert F. Lucas, Paul C. Messina, Jeffrey M. Perloff, William H. Press, Albert J. Semtner, Scott Stern, Shankar Subramaniam, Lawrence C. Tarbell, Jr. and Steve J. Wallach. The CSTB study director was Cynthia A. Patterson, assisted by Phil Hilliard, Margaret Marsh Huynh and Herbert S. Lin. The study was sponsored by the DOE's Office of Science and the DOE's Advanced Simulation and Computing program.

The study commenced in March 2003. Information was gathered from briefings during 5 committee meetings; an application workshop in which more than 20 computational scientists participated; site visits to DOE labs and NSA; a town hall meeting at the 2003 Supercomputing Conference; and a visit to Japan that included a supercomputing forum held in Tokyo. An interim report was issued in July 2003 and the final report was issued in November 2004. The report was extensively reviewed by seventeen external reviewers in a blind peer-review process as well as by NRC staff. The prepublication version of the report (at over 200 pages), entitled "Getting up to Speed: The Future of Supercomputing," is available from the National Academies Press[1] and also from DOE[2]. The final published version of the report is due in early 2005.

[1] http://books.nap.edu/catalog/11148.html
[2] http://www.sc.doe.gov/ascr/workshopreportspage.htm

The study focuses on supercomputing, narrowly defined as the development and use of the fastest and most powerful computing systems—i.e., *capability computing*. It covers technological, political and economic aspects of the supercomputing enterprise. We summarize in the following sections the main findings and recommendations of this study.

Supercomputing Matters

The study documents past contributions of supercomputing to national defense and to scientific discovery, together with evidence of its increasing importance in the future. Numerical simulation and digital data analysis have become essential to research in most disciplines, and many disciplines have insatiable needs for more performance. In areas such as climate modeling or plasma physics, there is a broad consensus that up to seven orders of magnitude of performance improvements will be needed to achieve well-defined computational goals; and there is a clear understanding of the likely advances that will accrue from the use of better performing supercomputing platforms. Supercomputers are essential to the missions of government agencies in areas such as intelligence or stockpile stewardship; they are an essential tool to the solution of important societal problems. Finally, technologies developed on supercomputers broadly contribute to our economy. Examples include application codes (such as NASTRAN) that were initially developed in national labs and run on supercomputers and then disseminated to broad industrial bases; as well as core IT technologies (such as multithreading or vector processing) that were pioneered on supercomputers and migrated to broadly used IT platforms. For reasons explained later, we expect this "trickle-down" process to continue, and perhaps intensify, in coming years. Although it is hard to quantify in a precise manner the benefits of supercomputing, the committee believes that the returns on increased investments in supercomputing will greatly exceed the cost of these investments.

Supercomputing is the Business of Government

The public sector is the leading user and purchaser of supercomputers: According to International Data Corporation (IDC), more than 50 percent of high-performance computer (HPC) purchases and more than 80 percent of capability system purchases in 2003 were made by the public sector. The reason for this is that supercomputers are used mostly to produce "public goods" and are essential for many government missions. They are used to support government funded basic and applied research; and they are used to support DoD or DOE missions, and the missions of intelligence agencies. Supercomputing technologies have often migrated to mainstream computing, but on a time table that is longer than the horizon of commercial computer companies.

This state of affair implies that the government plays a crucial role in the supercomputing industry, since its acquisitions have a major impact on the health, indeed, the existence of such an industry. Historically, the government has played an active role in ensuring that supercomputers are available to fulfill its needs by funding supercomputing R&D and by forging long-term relationships with key providers. While active government intervention has risks, it is necessary in areas where the private market is nonexistent or too small to ensure a steady flow of products and technologies that satisfy government needs. Thus, one clearly needs an active government policy to ensure a steady supply of military submarines or aircrafts; whereas no active government involvement is needed to ensure a steady supply of PCs. Are supercomputers more like military submarines or like PCs? To answer this question we need first to look at the current state of supercomputing in the US.

Supercomputing Thrives -- Supercomputing Falters

The Success of the Killer Micros

There are strong signs that supercomputing is a healthy business overall, and a healthy business in the US. Supercomputers at academic centers and government laboratories are used to do important research; supercomputers are used effectively in support of essential security missions; good progress is being made on stockpile stewardship, using supercomputing simulations. The large majority of supercomputers are US made: according to IDC, US vendors had 98% market share in capability systems in 2003; 91% of the TOP500 systems, as of June 2004, were US made.

On the other hand, companies that primarily develop supercomputing technologies, such as Cray, have a hard time staying in business. Supercomputers are a diminishing fraction of the total computer market, with a total value of less than $1 billion a year. It is an unstable market, with variations of more than 20 percent in sales from year to year. It is a market that is almost entirely dependent on government acquisitions.

The current state of supercomputing is largely a consequence of the success of commodity-based supercomputing. Most of the systems on the TOP500 list are now *clusters*, i.e., systems assembled from commercial, off-the-shelf (COTS) processors and switches; more than 95 percent of the systems use commodity microprocessor nodes. On the other hand, on the first TOP500 list of June 1993 only a quarter of the systems used commodity scalar microprocessors and none used COTS switches.

Cluster supercomputers have ridden on the coattails of Moore's Law, benefiting from the huge investments in commodity processors and the fast increase in processor performance. Indeed, the top performing commodity-based system on the June 1994 TOP500 list had 3,689 nodes; in June 2004 it had 4,096 nodes. While the number of nodes increased only by 11 percent, the system performance, as measured by the Linpack benchmark, improved by a factor of 139 in ten years! Cluster technology offers, for many applications, supercomputing performance at a cost/performance of a PC: as a result, high-performance computing can be afforded by an increasing number of users. Indeed, the verdict of the market is that clusters offer better value for money in many sectors where custom vector systems were previously used.

Victory is not Complete

Yet clusters cannot satisfy all supercomputing needs. For some problems, acceptable time to solution can be achieved only by scaling to a very large number of commodity nodes. Communication overheads become a bottleneck. A *hybrid* supercomputer, where commodity processors are connected via a custom network interface (connecting to the memory bus, rather than to an I/O bus) can support higher per-node bandwidth with lower overheads, thus enabling efficient use of a larger number of nodes. (The Cray XT3 and the SGI Altix are examples of such systems). A *custom* supercomputer, built of custom processors, can provide higher per-node performance and thus reduce the need to scale to a large number of nodes, at the expense of using more intra-node parallelism. (The Cray X1 and the NEC SX6 are the two current examples of such systems). Custom processors are especially important for codes that exhibit low locality and, thus, do not take advantage of caches. In such a case, it is important that the intra-node parallelism of the processor support a large number of concurrent memory accesses, as vector or heavily multithreaded processors do.

The success of clusters has reduced the market for hybrid and custom supercomputers to the point where the viability of these systems are heavily dependent on government support. Government investment in the development and acquisition of such platforms has shrunk. Computer suppliers are reluctant to invest in custom supercomputing due to the small size of the market, the uncertainty of the financial returns, and the opportunity cost of not applying skilled personnel to products designed for the broader IT market. Furthermore, academic research on the design of supercomputers has diminished. From the mid-nineties to the early 2000's, the number of published papers on supercomputing or high-performance computing has shrunk by half; the number of National Science Foundation (NSF) grants on parallel architecture design has shrunk by half; and large projects that build prototype systems have disappeared. The reduced research investment is worrisome, as it will be harder to benefit from advances due to Moore's law in the future. Some of the main obstacles are summarized next.

Problems in the Offing

Memory latency continues to increase, relative to processor speed: An extrapolation of current trends would lead to the conclusion that by 2020 a processor will execute about 800 loads and 90,000 floating point operations while waiting for one memory access to complete—an untenable differential. While the problem affects all processors, it affects scientific computing and high-performance computing earlier, as commercial codes can usually take better advantage of caches.

Global communication latency continues to increase and global bandwidth continues to decrease, relative to node speed. Again, an extrapolation of current trends would lead by 2020 to a global bandwidth of about 0.001 word per flop and a global latency equivalent to almost 1 Mflops. The problem affects tightly coupled HPC applications much more than loosely coupled commercial workloads.

Improvement in single processor performance is slowing down: It is hard to further increase pipelining depth or instruction-level parallelism, so that increasing chip gate counts do not contribute much to single processor performance. To stay on Moore's curve of microprocessor performance, vendors need to use increasing levels of on-chip multiprocessing. This is not a major problem for many commercial applications that can cope with modest levels of parallelism, but will be a problem for high-end supercomputers that will need to cope with hundreds of thousands of concurrent threads.

As circuit size shrinks and the number of circuits in a large supercomputer grows, mean-time-to-failure decreases. The largest computer systems are more affected by this problem than modest size computers.

It's the Software, Stupid

Although clusters have reduced the hardware cost of supercomputing, they have increased the programming effort needed to implement large parallel codes. Scientific codes and the platforms these codes run on have become more complex while the programming environments used to develop these codes have seen little progress. As a result, software productivity is low. Programming is done using message-passing libraries that are low-level and contribute large communication overheads. No higher-level programming notation that adequately captures parallelism and locality, the two main algorithmic concerns of parallel programming, has emerged. The application development environments and tools used to program complex parallel scientific codes are generally less advanced and less robust than those used for general commercial computing. Hybrid or custom systems could support more efficient parallel programming models, e.g., models that use global memory. But this potential is largely unrealized, because of the very low investments in supercomputing software such as compilers, the desire to maintain compatibility with the prevalent cluster architecture, and the fear of investing in software that runs only on architectures that may disappear in a few years. The software problem will worsen as higher levels of parallelism are required and as global communication becomes relatively slower.

A Fragile Ecosystem

The problems listed above indicate a clear need for change. We need new architectures to cope with the breakdown in current designs due to the diverging rate of improvement of various components (e.g., processor speed vs. memory speed vs. switch speed). We need new languages, new tools, and new operating systems to cope with the increased levels of parallelism, and the low software productivity. We need continued improvements in algorithms to handle larger problems, new models (to improve performance or accuracy), and to exploit changing supercomputer hardware characteristics.

But it takes time to realize the benefits of research. It took more than a decade from the first vector product until vector programming was well supported by algorithms, languages and compilers; it took more than a decade from the first massively parallel processor (MPP) products to well-supported standard message-passing programming environments. As the research pipeline has emptied, we are in a weak position to cope with the obstacles that are likely to limit supercomputing progress in the next decade.

Change is inhibited by the large investments in application software. While new hardware is purchased every three to five years, large software packages are maintained and used over decades. Changes in architectures and programming models may require expensive recoding, a nearly impossible task for poorly maintained, large "dusty deck" codes. Ecosystems are created through the mutually reinforcing effect of hardware and software that supports well a certain programming model, application software designed for such a programming model, and people that are familiar with the programming model and its environment. Even though the ecosystem may be caught in a "local minimum" and better productivity could be achieved with other architectures and programming models, change requires coordination in all aspects of technology (hardware and software), and very large investments in code rewriting and people retraining to overcome the potential barrier.

Progress also will be hampered by the small size and fragility of the supercomputing ecosystem. The community of researchers that develop new supercomputing hardware and software and applications is small. For example, according to the Taulbee surveys of the last few years, out of more than 800 CS PhDs that graduate each year in the U.S., only 36 specialize in computational sciences (and only 3 are hired by national laboratories). Since supercomputing is a very small fraction of the total IT industry, and since large system skills are needed in many other areas (e.g., Google), people can easily move to new jobs. There is little flow of personnel among the various groups in industry working on supercomputing and little institutional memory: the same problems are solved again and again. The loss of a few tens of people with essential skills can critically hamper a company or a lab. Instability of long-term funding and uncertainty in policies compound this problem.

Recommendations

Our report concludes that the U.S. government has unique supercomputing needs that will not be satisfied without government involvement. In this sense, producing custom supercomputers and supercomputing unique technologies is like producing cutting-edge weapon systems. However, there are essential differences: not only are custom supercomputers essential to our security, they can also accelerate many other research and engineering endeavors. Furthermore, custom supercomputers are much more closely related to commercially available products, such as clusters, then, say, military aircraft are to civilian aircraft. There is a significant reuse of commercial technologies in custom supercomputers and a continuous flow of invention from custom supercomputers to commodity systems. Finally, the development cycles are much shorter and the development costs are much lower.

This leads to the following overall recommendation:

Overall Recommendation: To meet the current and future needs of the United States, the government agencies that depend on supercomputing, together with the U.S. Congress, need to take primary responsibility for accelerating advances in supercomputing and ensuring that there are multiple strong domestic suppliers of both hardware and software.

To facilitate the government's assumption of that responsibility, the committee makes eight recommendations.

Recommendation 1. To get the maximum leverage from the national effort, the government agencies that are the major users of supercomputing should be jointly responsible for the strength and continued evolution of the supercomputing infrastructure in the United States, from basic research to suppliers and deployed platforms. The Congress should provide adequate and sustained funding.

A small number of government agencies are the primary users of supercomputing. These agencies are also the major funders of supercomputing research. At present, those agencies include the Department of Energy (DOE), including its National Nuclear Security Administration and its Office of Science; the Department of Defense (DoD), including its National Security Agency (NSA); the National Aeronautics and Space Administration (NASA); the National Oceanic and Atmospheric Administration (NOAA); and the National Science Foundation (NSF). The increasing use of supercomputing in biomedical applications suggests that NIH should be added to the list. There is a significant overlap among the supercomputing needs of these agencies.

The model we envisage is not a loose coordination, where each agency informs the others of its plans, but an integrated effort based on a joint long term plan. This 5-10 year High End Computing (HEC) plan would be based on both the roadmap that is the subject of Recommendation 5 and the needs of the participating agencies. Included in the plan would be a clear delineation of the responsibilities of various agencies. Joint planning and coordination of acquisitions will reduce procurement overhead and provide more stability to vendors. Agencies that support research will coordinate their efforts to ensure adequate funding of research addressing major roadblocks, as described in Recommendation 6. A more integrated effort by a few agencies may fund industrial development. House and Senate appropriation committees would ensure that budgets passed into law are consistent with the HEC plan.

Recommendation 2. The government agencies that are the primary users of supercomputing should ensure domestic leadership in those technologies that are essential to meet national needs.

Since the broad market on its own will not satisfy some of the supercomputing needs, the government should ensure the continued availability of needed unique technologies. The U.S. government may want to restrict the export of some technologies, and thus may want these technologies to be produced in the U.S. More importantly, no other country is certain to produce these technologies. The United States needs to invest in supercomputing not in order to be ahead of other countries, but in order to have the tools needed to support critical agency missions in areas such as signals intelligence and weapon stewardship. These investments will also broadly benefit scientific research and the U.S. economy.

Recommendations 3 through 8 outline some of the actions that need to be taken by these agencies to maintain leadership.

Recommendation 3. To satisfy its need for unique supercomputing technologies such as high-bandwidth systems, the government needs to ensure the viability of multiple domestic suppliers.

The viability of vendors of unique supercomputing technologies depends on stable, long-term government investments at adequate levels: both the absolute investment level and its predictability matter, because of the lack of alternative support. Such stable support can be provided either via government funding of R&D expenses or via steady procurements (or both). The model proposed by the British UKHEV initiative, whereby government solicits and funds proposals for the procurement of three successive generations of a supercomputer family over four to six years is a good example of a model that reduces instability.

The most important unique supercomputing technology identified in this report is custom supercomputing systems. The committee estimated the R&D cost for such a product to be about $70 million per year. This includes both the hardware platform and the software stack. The cost would be lower for a vendor that does not do both.

There also are many supercomputing unique technologies in the software area, leading to the following recommendation:

Recommendation 4. The creation and long-term maintenance of the software that is key to supercomputing requires the support of those agencies that are responsible for supercomputing R&D. That software includes operating systems, libraries, compilers, software development and data analysis tools, application codes, and databases.

The committee believes that higher and more coordinated investments could significantly improve the productivity of supercomputing platforms. The models for software support are likely to be varied—vertically integrated vendors that produce both hardware and software, horizontal vendors that produce software for many different hardware platforms, not-for-profit organizations, software developed in the open source model, etc. However, no matter which model is used, stability and continuity are essential. Software has to be maintained and evolved over decades; this requires a stable cadre of software developers with intimate knowledge of the software. Independent software vendors (ISVs) can play an important role in developing and maintaining software products; the government can help by ensuring that software is developed in national labs only when it can not be bought.

Recommendation 5. The government agencies responsible for supercomputing should underwrite a community effort to develop and maintain a roadmap that identifies key obstacles and synergies in all of supercomputing.

A roadmap is necessary to ensure that investments in supercomputing R&D are prioritized appropriately. It should be developed with wide participation from researchers, developers of both commodity and custom technologies and users; it should be driven both top-down from application needs and bottom-up from technology barriers; it should be, as much as possible, quantitative in measuring needs and capabilities; finally, it should not ignore the strong interdependencies of technologies.

The roadmap should be used by agencies and by Congress to guide their long-term research and development investments. It is important also to invest in some high-risk, high-return research ideas that are not indicated by the roadmap, to avoid being blindsided.

Recommendation 6. Government agencies responsible for supercomputing should increase their levels of stable, robust, sustained multiagency investment in basic research. More research is needed in all the key technologies required for the design and use of supercomputers (architecture, software, algorithms, and applications).

The decreased research investments at a time in which roadblocks are accumulating puts the supercomputing enterprise at risk. A major correction is needed. The committee estimated the investment needed to support research in core supercomputing technologies at $140 million per year. (This estimate does not include application development or platform acquisition.) It is important that this investment focus on universities, both because of the importance of a free flow of information at an early stage, and because of the role of universities in educating the future cadre of supercomputing practitioners. Finally, research should include a mix of small, medium and large projects, including demonstration systems where technologies are integrated. Such systems are important to study the interplay of technologies and validate them in a realistic environment; they should not be confused with product prototypes and should not be expected to support users.

Recommendation 7. Supercomputing research is an international activity; barriers to international collaboration should be minimized.

Research has always benefited from the open exchange of ideas. In light of the relatively small community of supercomputing researchers, international collaborations are particularly beneficial. The fast development cycles, the fast technology evolution, and the frequent flow of ideas and technologies between supercomputing and the broader IT industry require close interaction between the supercomputing industry and the broader IT industry, and between supercomputing research and the broader IT research. The strategic advantage to the U.S. from supercomputing is not due to a single product, but from a broad capability to acquire and exploit effectively systems that can best reduce the time to solution of important computational problems. Looser export restrictions would not erode this advantage but would benefit U.S. vendors; in particular, restrictions that affect commodity clusters that can be assembled from widely available components lack any rationale.

Barriers also reduce the benefit of supercomputing to science. Science is a collaborative international endeavor, and many of the best U.S. graduate students are foreigners. A restriction on supercomputer access by foreign nationals means that supercomputers are less available to support science in the U.S.

Recommendation 8. The U. S. government should ensure that researchers with the most demanding computational requirements have access to the most powerful supercomputing systems.

Supercomputing is important for the advancement of science. NSF supercomputing centers, as well as DOE science centers, have done an important service in providing supercomputing support to scientists. However, these centers have seen a broadening of their mission with constant budgets, and have been under pressure to support an increasing number of users. It is important that sufficient stable funding be provided to support an adequate science supercomputing infrastructure. Capability systems should be available to scientists with the most demanding problems and should be used only for jobs that need this capability; supercomputing resources should be available to educate the next generation and to develop the needed software infrastructure. Finally, it is important that the science communities that use supercomputers have a strong say and a shared responsibility for the provision of adequate supercomputing infrastructure, with budgets for the acquisition and maintenance of such infrastructure being clearly separated from the budgets for IT research.

What Next

The final report was presented at the 2004 Supercomputing Conference and at briefings attended by DOE staff, congressional staff, staff from the Office of Science and Technology Policy and staff from the Office of Management and Budget. Its content was covered by the general press and by trade publications. The report was generally well received, with words of caution on the difficulty in allocating more money to supercomputing in a world of retrenching research budgets. People in the audience at Supercomputing rightly remarked that similar recommendations appeared in previous reports yet were not acted upon. While it would be better if these recommendations had been acted upon, it is good that various reports push similar recommendations: In order to effect change, the community has to speak in one voice. The recent High-End Computing Revitalization Act is a step in the right direction, but much more is needed. The agencies and the scientists that need supercomputers have to act together and push not only for an adequate supercomputing infrastructure now, but for adequate plans and investments that will ensure they have the tools they need in five or ten years.

Acknowledgments

The authors wish to thank Cynthia A. Patterson for her careful editing of this text.

Recent Trends in the Marketplace of High Performance Computing

Erich Strohmaier
LAWRENCE BERKELEY NATIONAL LABORATORY

Jack J. Dongarra
UNIVERSITY OF TENNESSEE/
OAK RIDGE NATIONAL LABORATORY

Hans W. Meuer
UNIVERSITY OF MANNHEIM

Horst D. Simon
LAWRENCE BERKELEY NATIONAL LABORATORY

Introduction

"The Only Thing Constant Is Change" — Looking back on the last four decades this seems certainly to be true for the market of High-Performance Computing systems (HPC). This market was always characterized by a rapid change of vendors, architectures, technologies and the usage of systems[1]. Despite all these changes the evolution of performance on a large scale however seems to be a very steady and continuous process. Moore's Law is often cited in this context. If we plot the peak performance of various computers of the last six decades in Fig. 1, which could have been called the 'supercomputers' of their time[2,3], we indeed see how well this law holds for almost the complete lifespan of modern computing. On average we see an increase in performance of two magnitudes of order every decade.

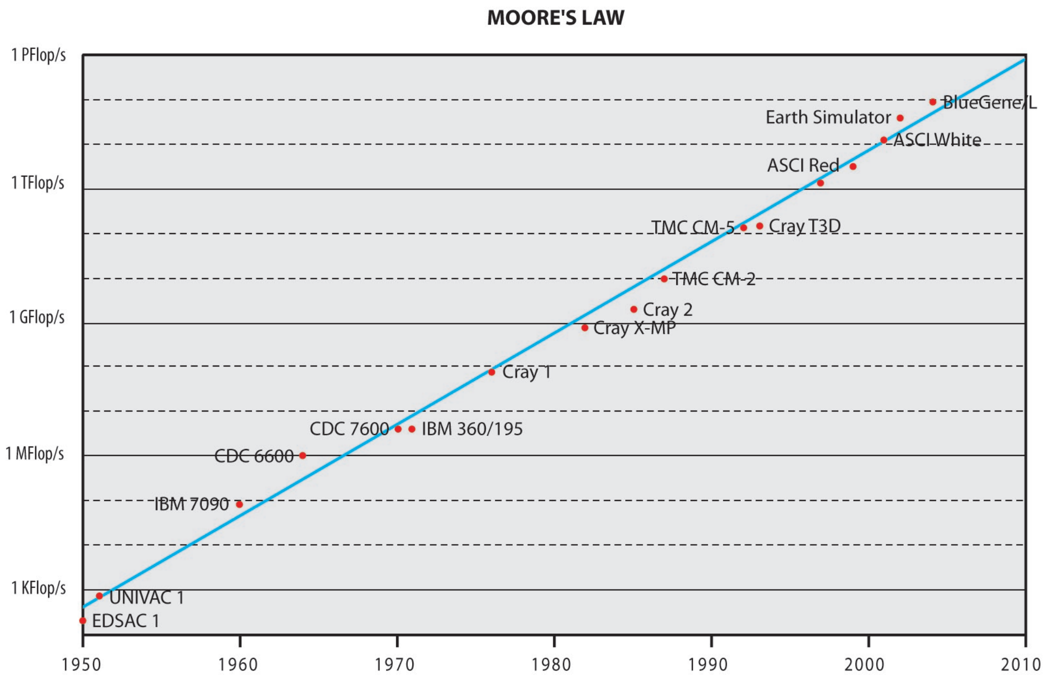

Fig. 1. Performance of the fastest computer systems for the last six decades compared to Moore's Law.

Explosion of Cluster Based Systems

At the end of the 1990s, clusters were common in academia but mostly as research objects and not primarily as general purpose computing platforms for applications. Most of these clusters were of comparable small scale and, as a result, the November 1999 edition of the TOP500 listed only seven cluster systems. This changed dramatically as industrial and commercial customers started deploying clusters as soon as applications with less stringent communication requirements permitted them to take advantage of the better price/performance ratio -roughly an order of magnitude- of commodity based clusters. At the same time, all major vendors in the HPC market started selling this type of cluster to their customer

[1] E. Strohmaier, J.J. Dongarra, H.W. Meuer, and H.D. Simon, *The Marketplace of High-Performance Computing*, Parallel Computing 25 (1999) 1517.

[2] R. W. Hockney, C. Jesshope, Parallel Computers II: Architecture, *Programming and Algorithms*, Adam Hilger, Ltd., Bristol, United Kingdom, 1988.

[3] H. W. Meuer, E. Strohmaier, J. J. Dongarra, and Horst D. Simon, TOP500, www.top500.org.

base. In November 2004 clusters were the dominant architectures in the TOP500 with 294 systems at all levels of performance (see Fig 2). Companies such as IBM and Hewlett-Packard sell the majority of these clusters and a large number of them are installed at commercial and industrial customers.

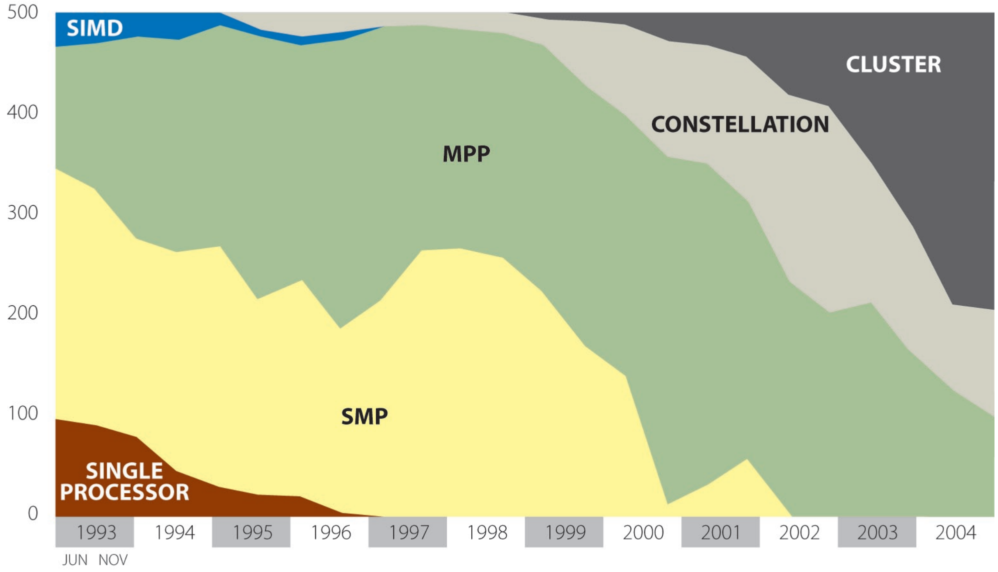

Fig. 2. Main Architectural Categories seen in the TOP500.

In addition, there still is generally a large difference in the usage of clusters and their more integrated counterparts; clusters are mostly used for capacity computing while the integrated machines primarily are used for capability computing. The largest supercomputers are used for capability or turnaround computing where the maximum processing power is applied to a single problem. The goal is to solve a larger problem or to solve a single problem in a shorter period of time. Capability computing enables the solution of problems that cannot otherwise be solved in a reasonable period of time (e.g., by moving from a 2D to a 3D simulation, using finer grids, or using more realistic models). Capability computing also enables the solution of problems with real-time constraints (e.g., predicting weather). The main figure of merit is time to solution. Smaller or cheaper systems are used for capacity computing, where smaller problems are solved. Capacity computing can be used to enable parametric studies or to explore design alternatives; it is often needed to prepare for more expensive runs on capability systems. Capacity systems will often run several jobs simultaneously. The main figure of merit is sustained performance per unit cost. Traditionally, vendors of large supercomputer systems have learned to provide for this first mode of operation as the precious resources of their systems were required to be used as effectively as possible. By contrast, Beowulf clusters are mostly operated through the Linux operating system (a small minority using Microsoft Windows) where these operating systems either lack the tools or these tools are relatively immature to use a cluster effectively for capability computing. However, as clusters become on average both larger and more stable, there is a trend to use them also as computational capability servers.

There are a number of choices of communication networks available in clusters. Of course 100 Mb/s Ethernet or Gigabit Ethernet is always possible, which is attractive for economic reasons, but has the drawback of a high latency (~ 100 μs). Alternatively, there are, for instance, networks that operate from user space, like Myrinet, Infiniband, and SCI. The network speeds as shown by these networks are more or less on par with some integrated parallel systems. So, possibly apart from the speed of the processors and of the software that is provided by the vendors of traditional integrated supercomputers, the distinction between clusters and this class of machines becomes rather small and will, without a doubt, decrease further in the coming years.

Intel-ization of the Processor Landscape

The HPC community had started to use commodity parts in large numbers in the 1990s already. MPPs and Constellations (Cluster of SMP) typically used standard workstation microprocessors even though they still might have used custom interconnect systems. There was however one big exception, virtually nobody used Intel microprocessors. Lack of performance and the limitations of a 32-bit processor design were the main reasons for this. This changed with the introduction of the Pentium III and especially in 2001 with the Pentium 4, which featured greatly improved memory performance due to its redesigned front-side bus and full 64-bit floating point support. The number of system in the TOP500 with Intel processors exploded from only 6 in November 2000 to 318 in November 2004 (Fig. 3).

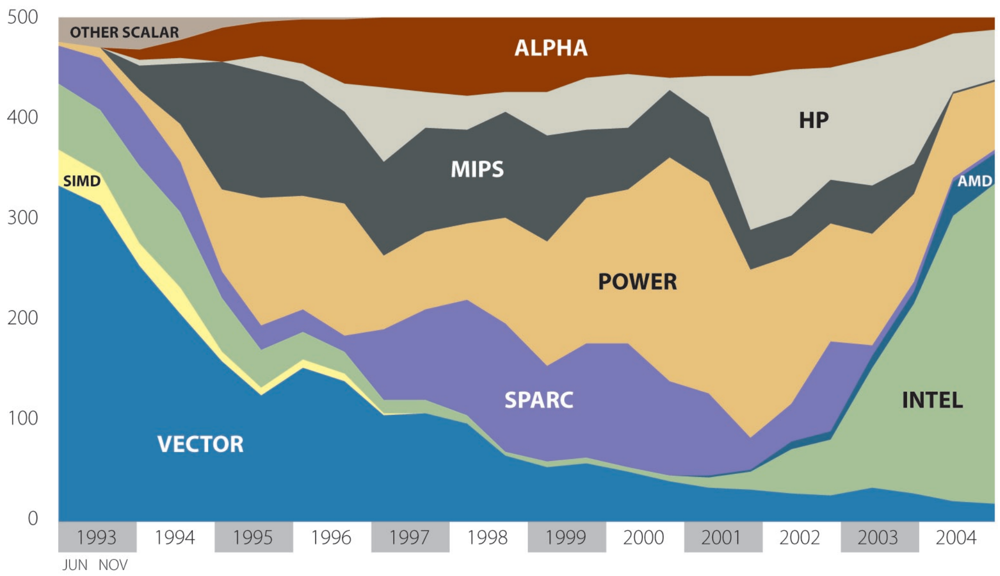

Fig. 3. Main Processor Families seen in the TOP500.

New Architectures on the Horizon

Interest in novel computer architectures has always been great in the HPC community, which comes as little surprise as this field was born, and continues to thrive, on technological innovations. Some of the concerns of recent years were the ever increasing space and power requirements of modern commodity based supercomputers. In the BlueGene/L development, IBM addressed these issues by designing a very power and space efficient system. BlueGene/L uses not the latest commodity processors available but computationally less powerful and much more power efficient processor versions developed mainly not for the PC and workstation market but for embedded applications. Together with a drastic reduction of the available main memory this leads to a very dense system. To achieve the targeted extreme performance level an unprecedented number of these processors (up to 128,000) are combined using several specialized interconnects. There was and is considerable doubt whether such a system would be able to deliver the promised performance and would be usable as a general purpose system. First results of the current beta-System are very encouraging and the one-quarter size beta-System of the future LLNL system was able to claim the number one spot on the November 2004 TOP500 list.

Contrary to the progress in hardware development, there has been little progress, and perhaps regress, in making scalable systems easy to program. Software directions that were started in the early 1990's (such as CM-Fortran and High-Performance Fortran) were largely abandoned. The payoff to finding better ways to program such systems and thus expand the domains in which these systems can be applied would appear to be large.

The move to distributed memory has forced changes in the programming paradigm of supercomputing. The high cost of processor-to-processor synchronization and communication requires new algorithms that minimize those operations. The structuring of an application for vectorization is seldom the best structure for parallelization on these systems. Moreover, despite some research successes in this area, without some guidance from the programmer, compilers are generally able neither to detect enough of the necessary parallelism nor to reduce sufficiently the inter-processor overheads. The use of distributed memory systems has led to the introduction of new programming models, particularly the message passing paradigm, as realized in MPI, and the use of parallel loops in shared memory subsystems, as supported by OpenMP. It also has forced significant reprogramming of libraries and applications to port onto the new architectures. Debuggers and performance tools for scalable systems have developed slowly, however, and even today most users consider the programming tools on parallel supercomputers to be inadequate.

All these issues prompted DARPA to start a program for High Productivity Computing Systems (HPCS) with the declared goal to develop a new computer architecture by the end of the decade with high performance and productivity. The performance goal is to install a system by 2009, which can sustain Petaflop/s performance level on real applications. This should be achieved by the combination of a new architecture designed to be easily programmable and combined with a complete new software infrastructure to make user productivity as high as possible.

Projections

Based on the current TOP500 data, which cover the last twelve years and the assumption that the current performance development continues for some time to come, we can now extrapolate the observed performance and compare these values with the goals of the mentioned government programs. In Fig. 4, we extrapolate the observed performance values using linear regression on the logarithmic scale. This means that we fit exponential growth to all levels of performance in the TOP500. This simple fitting of the data shows surprisingly consistent results. In 1999, based on a similar extrapolation[1], we expected to have the first 100 TFlop/s system by 2005. We also predicted that by 2005 no system smaller then 1 TFlop/s should be able to make the TOP500 any more. Both of these predictions are basically certain to be fulfilled next year. Looking out another five years to 2010 we expect to see the first PetaFlops system at about 2009[1] and our current extrapolation is still the same. This coincides with the declared goal of the DARPA HPCS program.

[1] E. Strohmaier, J.J. Dongarra, H.W. Meuer, and H.D. Simon, *The Marketplace of High-Performance Computing*, Parallel Computing 25 (1999) 1517

Recent Trends in the Marketplace of High Performance Computing

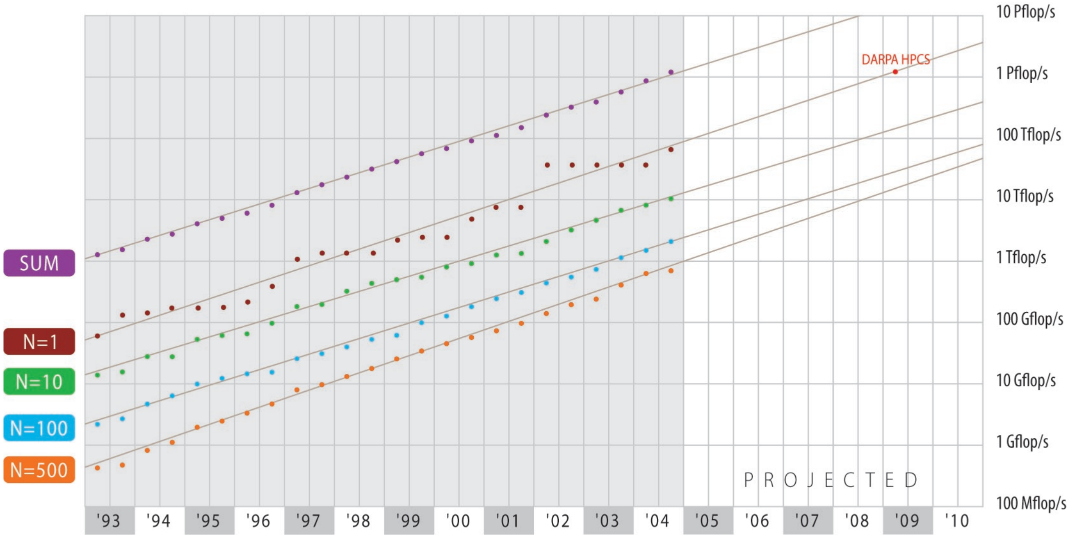

Fig. 4. Extrapolation of recent growth rates of performance seen in the TOP500.

Looking even further into the future we could speculate that, based on the current doubling of performance every year, the first system exceeding 100 Petaflop/s should be available around or shortly after 2015. Due to the rapid changes in the technologies used in HPC systems there is however again no reasonable projection possible for the architecture of such a system in 10 years. The end of Moore's Law as we know it has often been predicted and one day it will come. New technologies, such as quantum computing that would allow us to further extend our computing capabilities are well beyond the capabilities of our simple performance projections. However, even as the HPC market has changed its face several times quite substantially since the introduction of the Cray 1 four decades ago, there is no end in sight for these rapid cycles of re-definition. And we still can say that in the High-Performance Computing market "The Only Thing Constant Is Change".

Scientific Data Management in the Coming Decade

Jim Gray
MICROSOFT

David T. Liu
BERKELEY

Maria Nieto-Santisteban
JOHNS HOPKINS UNIVERSITY

Alex Szalay
JOHNS HOPKINS UNIVERSITY

David DeWitt
WISCONSIN

Gerd Heber
CORNELL

Data-intensive science – a new paradigm

Scientific instruments and computer simulations are creating vast data stores that require new scientific methods to analyze and organize the data. Data volumes are approximately doubling each year. Since these new instruments have extraordinary precision, the data quality is also rapidly improving. Analyzing this data to find the subtle effects missed by previous studies requires algorithms that can simultaneously deal with huge datasets and that can find very subtle effects – finding both needles in the haystack and finding very small haystacks that were undetected in previous measurements.

The raw instrument and simulation data is processed by pipelines that produce standard data products. In the NASA terminology[1], the raw *Level 0* data is calibrated and rectified to *Level 1* datasets that are combined with other data to make derived *Level 2* datasets. Most analysis happens on these *Level 2* datasets with drill down to *Level 1* data when anomalies are investigated.

We believe that most new science happens when the data is examined in new ways. So our focus here is on data exploration, interactive data analysis, and integration of *Level 2* datasets.

Data analysis tools have not kept pace with our ability to capture and store data. Many scientists envy the pen-and-paper days when all their data used to fit in a notebook and analysis was done with a slide-rule. Things were simpler then; one could focus on the science rather than needing to be an information technology professional with expertise in arcane computer data analysis tools.

The largest data analysis gap is in this man-machine interface. How can we put the scientist back in control of his data? How can we build analysis tools that are intuitive and that augment the scientist's intellect rather than adding to the intellectual burden with a forest of arcane user tools? The real challenge is building this *smart notebook* that unlocks the data and makes it easy to capture, organize, analyze, visualize, and publish.

This article is about the data and data analysis layer within such a smart notebook. We argue that the *smart notebook* will access data presented by science centers that will provide the community with analysis tools and computational resources to explore huge data archives.

New data-analysis methods

The demand for tools and computational resources to perform scientific data-analysis is rising even faster than data volumes. This is a consequence of three phenomena: (1) More sophisticated algorithms consume more instructions to analyze each byte, (2) Many analysis algorithms are super-linear, often needing N^2 or N^3 time to process N data points, and (3) IO bandwidth has not kept pace with storage capacity. In the last decade, while capacity has grown more than 100-fold, storage bandwidth has improved only about 10-fold.

[1] Committee on Data Management, Archiving, and Computing (CODMAC) Data Level Definitions, http://science.hq.nasa.gov/research/earth_science_formats.html

These three trends: algorithmic intensity, nonlinearity, and bandwidth-limits mean that the analysis is taking longer and longer. To ameliorate these problems, scientists will need better analysis algorithms that can handle extremely large datasets with approximate algorithms (ones with near-linear execution time), and they will need parallel algorithms that can apply many processors and many disks to the problem to meet cpu-density and bandwidth-density demands.

Science centers

These peta-scale datasets required a new work style. Today the typical scientist copies files to a local server and operates on the datasets using his own resources. Increasingly, the datasets are so large, and the application programs are so complex, that it is much more economical to move the end-user's programs to the data and only communicate questions and answers rather than moving the source data and its applications to the user's local system.

Science data centers that provide access to both the data and the applications that analyze the data are emerging as service stations for one or another scientific domain. Each of these science centers curates one or more massive datasets, curates the applications that provide access to that dataset, and supports a staff that understands the data and indeed is constantly adding to and improving the dataset. One can see this with the SDSS at Fermilab, BaBar at SLAC, BIRN at SDSC, with Entrez-PubMed-GenBank at NCBI, and with many other datasets across other disciplines. These centers federate with others. For example BaBar has about 25 peer sites and CERN LHC expects to have many Tier1 peer sites. NCBI has several peers, and SDSS is part of the International Virtual Observatory.

The new work style in these scientific domains is to send questions to applications running at a data center and get back answers, rather than to bulk-copy raw data from the archive to your local server for further analysis. Indeed, there is an emerging trend to store a *personal workspace* (a *MyDB*) at the data center and deposit answers there. This minimizes data movement and allows collaboration among a group of scientists doing joint analysis. These personal workspaces are also a vehicle for data analysis groups to collaborate. Longer term, personal workspaces at the data center could become a vehicle for data publication, posting both the scientific results of an experiment or investigation along with the programs used to generate them in public read-only databases.

Many scientists will prefer doing much of their analysis at data centers because it will save them having to manage local data and computer farms. Some scientists may bring the small data extracts "home" for local processing, analysis and visualization, but it will be possible to do all the analysis at the data center using the personal workspace.

When a scientist wants to correlate data from two different data centers, then there is no option but to move part of the data from one place to another. If this is common, the two data centers will likely federate with one another to provide mutual data backup since the data traffic will justify making the copy.

Peta-scale data sets will require 1,000-10,000 disks and thousands of compute nodes. At any one time some of the disks and some of the nodes will be broken. Such systems require a mechanism in place to protect against data loss, and provide availability even with a less than full configuration — a self-healing system is required. Replicating the data in science centers at different geographic locations is implied in the discussion above. Geographic replication provides both data availability and protects against data loss. Within a data center one can combine redundancy with a clever partitioning strategy to protect against failure at the disk controller or server level. While storing the data twice for redundancy, one can use different organizations (e.g. partition by space in one, and by time in the other) to optimize system performance. Failures should be automatically recovered from the redundant copies with no interruption to database access, much as RAID5 disk arrays do today.

All these scenarios postulate easy data access, interchange and integration. Data must be self-describing in order to allow this. This self-description, or metadata, is central to all these scenarios; it enables generic tools to understand the data, and it enables people to understand the data.

Metadata enables data access

Metadata is the descriptive information about data that explains the measured attributes, their names, units, precision, accuracy, data layout and ideally a great deal more. Most importantly, metadata includes the data lineage that describes how the data was measured, acquired or computed.

If the data is to be analyzed by generic tools, the tools need to "understand" the data. You cannot just present a bundle-of-bytes to a tool and expect the tool to intuit where the data values are and what they mean. The tool will want to know the metadata.

To take a simple example, given a file, you cannot say much about it – it could be anything. If I tell you it is a JPEG, you know it is a bitmap in http://www.jpeg.org/ format. JPEG files start with a header that describes the file layout, and often tells the camera, timestamp, and program that generated the picture. Many programs know how to read JPEG files and also produce new JPEG files that include metadata describing how the new image was produced. MP3 music files and PDF document files have similar roles. Each is in a standard format, each carries some metadata, and each has an application suite to process and generate that file class.

If scientists are to read data collected by others, then the data must be carefully documented and must be published in forms that allow easy access and automated manipulation. In an ideal world there would be powerful tools that make it easy to capture, organize, analyze, visualize, and publish data. The tools would do data mining and machine learning on the data, and would make it easy to script workflows that analyze the data. Good metadata for the inputs is essential to make these tools automatic. Preserving and augmenting this metadata as part of the processing (data lineage) will be a key benefit of the next-generation tools.

All the derived data that the scientist produces must also be carefully documented and published in forms that allow easy access. Ideally much of this metadata would be automatically generated and managed as part of the workflow, reducing the scientist's intellectual burden.

Semantic convergence: numbers to objects

Much science data is in the form of numeric arrays generated by instruments and simulations. Simple and convenient data models have evolved to represent arrays and relationships among them. These data models can also represent data lineage and other metadata by including narrative text, data definitions, and data tables within the file. HDF[2], NetCDF[3] and FITS[4] are good examples of such standards. They each include a library that encapsulates the

[2] http://hdf.ncsa.uiuc.edu/HDF5/

[3] http://my.unidata.ucar.edu/content/software/netcdf/

[4] http://fits.gsfc.nasa.gov/

files and provides a platform-independent way to read sub-arrays and to create or update files. Each standard allows easy data interchange among scientists. Generic tools that analyze and visualize these higher-level file formats are built atop each of these standards.

While the commercial world has standardized on the relational data model and SQL, no single standard or tool has critical mass in the scientific community. There are many parallel and competing efforts to build these tool suites – at least one per discipline. Data interchange outside each group is problematic. In the next decade, as data interchange among scientific disciplines becomes increasingly important, a common HDF-like format and package for all the sciences will likely emerge.

Definitions of common terminology (units and measurements) are emerging within each discipline. We are most familiar with the Universal Content Descriptors (UCD[5]) of the Astronomy community that define about a thousand core astrophysics units, measurements, and concepts. Almost every discipline has an analogous ontology (a.k.a., *controlled vocabulary*) effort. These efforts will likely start to converge over the next decade, probably as part of the converged format standard. This will greatly facilitate tool-building and tools since an agreement on these concepts can help guide analysis tool designs.

[5] http://vizier.u-strasbg.fr/doc/UCD.htx

In addition to standardization, computer-usable ontologies will help build the Semantic Web: applications will be semantically compatible beyond the mere syntactic compatibility that current-generation of Web services offer with type matching interfaces. However, it will take some time before high-performance, general-purpose *ontology engines* will be available and integrated with data analysis tools.

Database users, on the other hand, are well positioned to prototype such applications: a database schema, though not a complete ontology in itself, can be a rich ontology extract. SQL can be used to implement a rudimentary *semantic algebra*. The XML integration in modern Database Management Systems (DBMS) opens the door for existing standards like RDF and OWL.

Visualization, or better *visual exploration,* is a prime example of an application where success is determined by the ability to map a question formulated in the conceptual framework of the domain ontology onto the querying capabilities of a (meta-) data analysis backend. For the time being, a hybrid of SQL and XQuery is the only language suitable to serve as the target assembly language in this translation process.

Metadata enables data independence

The separation of data and programs is artificial – one cannot see the data without using a program and most programs are data driven. So, it is paradoxical that the data management community has worked for 40 years to achieve something called *data independence*, a clear separation of programs from data. Database systems provide two forms of data independence termed *physical data independence and logical data independence*.

Physical data independence comes in many different forms. However, in all cases the goal is to be able to change the underlying physical data organization without breaking any application programs that depend on the old data format. One example of physical data independence is the ability of a database system to partition the rows of a table across multiple disks and/or multiple nodes of a cluster without requiring that any application programs be modified. The mapping of the fields of each row of a relational table to different disks is another important example of physical data independence. While a database system might choose to map each row to a contiguous storage container (e.g. a record) on a single disk page, it might also choose to store large, possibly infrequently referenced attributes of a table corresponding to large text objects, JPEG images, or multidimensional arrays in separate storage containers on different disk pages and/or different storage volumes in order to maximize the overall performance of the system. Again, such physical storage optimizations are implemented to be completely transparent to application programs except, perhaps, for a change in their performance. In the scientific domain, the analogy would be that you could take a working application program that uses a C struct to describe its data records on disk and change the physical layout of the records without having to rewrite or even recompile the application program (or any of the other application programs that access the same data). By allowing such techniques, physical data independence allows performance improvements by reorganizing data for parallelism, at little or no extra effort on the part of scientists.

Modern database systems also provide *logical data independence* that insulates programs from changes to the logical database design, allowing designers to add or delete relationships and to add information to the database. While physical data independence is used to hide changes in the physical data organizations, logical data independence hides changes in the logical organization of the data. Logical data independence is typically supported using *views*. A view defines a virtual table that is specified using a SQL query over one or more base tables and/or other views. Views serve many purposes including increased security (by hiding attributes from applications and/or users without a legitimate need for access) and enhanced performance (by materializing views defined by complex SQL queries over very large input tables). But views are primarily used to allow old programs to operate correctly even as the underlying database is reorganized and redesigned. For example, consider a program whose correct operation depends on some table T that a database administrator wants to reorganize by dividing vertically into two pieces stored in tables T' and T". To preserve applications that depend on T, the database administrator can then define a view over T' and T" corresponding to the original definition of table T, allowing old programs to continue to operate correctly.

In addition, data evolves. Systems evolve from EBCDIC to ASCII to Unicode, from proprietary-float to IEEE-float, from marks to euros, and from 8-character ASCII names to 1,000 character Unicode names. It is important to be able to make these changes without breaking the millions of lines of existing programs that want to see the data in the old way. Views are used to solve these problems by dynamically translating data to the appropriate formats (converting among character and number representations, converting among 6-digit and 9-digit postal codes, converting between long-and-short names, and hiding new information from old programs.) The pain of the Y2K (converting from 2-character to 4-character years) taught most organizations the importance of data independence.

Database systems use a *schema* to implement both logical and physical data independence. The schema for a database holds all metadata including table and view definitions as well as information on what indices exist and how tables are mapped to storage volumes (and nodes in a parallel database environment). Separating the data and the metadata from the programs that manipulate the data is crucial to data independence. Otherwise, it is essentially impossible for other programs to find the metadata which, in turn, makes it essentially impossible for multiple programs to share a common database. Object-oriented programming concepts have refined the separation of programs and data. Data classes encapsulated with methods provide data independence and make it much easier to evolve the data without perturbing programs. So, these ideas are still evolving. But the key point of this section is that an explicit and standard data access layer with precise metadata and explicit data access is essential for data independence.

Set-oriented data access gives parallelism

As mentioned earlier, scientists often start with numeric data arrays from their instruments or simulations. Often, these arrays are accompanied by tabular data describing the experimental setup, simulation parameters, or environmental conditions. The data are also accompanied by documents that explain the data.

Many operations take these arrays and produce new arrays, but eventually, the arrays undergo *feature extraction* to produce *objects* that are the basis for further analysis. For example, raw astronomy data is converted to object catalogs of stars and galaxies. Stream-gauge measurements are converted to stream-flow and water-quality time-series data, serum-mass-spectrograms are converted to records describing peptide and protein concentrations, and raw high-energy physics data are converted to events.

Most scientific studies involve exploring and data mining these object-oriented tabular datasets. The scientific file-formats of HDF, NetCDF, and FITS can represent tabular data but they provide minimal tools for searching and analyzing tabular data. Their main focus is getting the tables and sub-arrays into your Fortran/C/Java/Python address space where you can manipulate the data using the programming language.

This Fortran/C/Java/Python file-at-a-time procedural data analysis is nearing the breaking point. The data avalanche is creating billions of files and trillions of events. The file-oriented approach postulates that files are organized into directories. The directories relate all data from some instrument or some month or some region or some laboratory. As things evolve, the directories become hierarchical. In this model, data analysis proceeds by searching all the relevant files – opening each file, extracting the relevant data and then moving onto the next file. When all the relevant data has been gathered in memory (or in intermediate files) the program can begin its analysis. Performing this *filter-then-analyze*, data analysis on large datasets with conventional procedural tools runs slower and slower as data volumes increase. Usually, they use only one-cpu-at-a-time; one-disk-at-a-time and they do a brute-force search of the data. Scientists need a way (1) to use intelligent indices and data organizations to subset the search, (2) to use parallel processing and data access to search huge datasets within seconds, and (3) to have powerful analysis tools that they can apply to the subset of data being analyzed.

One approach to this is to use the MPI (Message Passing Interface) parallel programming environment to write procedural programs that stream files across a processor array, each node of the array exploring one part of the hierarchy. This is adequate for highly-regular array processing tasks, but it seems too daunting for ad-hoc analysis of tabular data. MPI and the various array file formats lack indexing methods other than partitioned sequential scan. MPI itself lacks any notion of metadata beyond file names.

As file systems grow to petabyte-scale archives with billions of files, the science community must create a synthesis of database systems and file systems. At a minimum, the file hierarchy will be replaced with a database that catalogs the attributes and lineage of each file. Set-oriented file processing will make file names increasingly irrelevant, analysis will be applied to “all data with these attributes” rather than working on a list of file/directory names or name patterns. Indeed, the files themselves may become irrelevant (they are just containers for data). One can see a harbinger of this idea in the Map-Reduce approach pioneered by Google[6]. From our perspective, the key aspect of Google Map-Reduce is that it applies thousands of processors and disks to explore large datasets in parallel. That system has a very simple data model appropriate for the Google processing, but we imagine it could evolve over the next decade to be quite general.

[6] "MapReduce: Simplified Data Processing on Large Clusters," J. Dean, S. Ghemawat, ACM OSDI, Dec. 2004.

The database community has provided automatic query processing along with CPU and IO parallelism for over two decades. Indeed, this automatic parallelism allows large corporations to mine 100-Terabyte datasets today using 1,000 processor clusters. We believe that many of those techniques apply to scientific datasets[7].

[7] "Parallel Database Systems: the Future of High Performance Database Systems", D. DeWitt, J. Gray, CACM, Vol. 35, No. 6, June 1992.

Other useful database features

Database systems are also approaching the peta-scale data management problem driven largely by the need to manage huge information stores for the commercial and governmental sectors. They hide the file concept and deal with data collections. They can federate many different sources letting the program view them all as a single data collection. They also let the program pivot on any data attributes.

Database systems provide very powerful data definition tools to specify the abstract data formats and also specify how the data is organized. They routinely allow the data to be replicated so that it can be organized in several ways (e.g., by time, by space, by other attributes). These techniques have evolved from mere indices to materialized views that can combine data from many sources.

Database systems provide powerful associative search (search by value rather than by location) and provide automatic parallel access and execution essential to peta-scale data analysis. They provide non-procedural and parallel data search to quickly find data subsets, as well as many tools to automate data design and management.

In addition, data analysis using data cubes has made huge advances, and now efforts are focused on integrating machine learning algorithms that infer trends, do data clustering, and detect anomalies. All these tools are aimed at making it easy to analyze commercial data, but they are equally applicable to scientific data analysis.

Ending the impedance mismatch

Conventional tabular database systems are adequate for analyzing objects (e.g., galaxies, spectra, proteins, events, etc.). But even there, the support for time-sequence, spatial, text and other data types is often awkward. Database systems have not traditionally supported science's core data type: the N-dimensional array. Arrays have had to masquerade as blobs (binary large objects) in most systems. This collection of problems is generally called the *impedance mismatch*, meaning the mismatch between the programming model and the database capabilities. The impedance mismatch has made it difficult to map many science applications into conventional tabular database systems.

But, database systems are changing. They are being integrated with programming languages so that they can support object-oriented databases. This new generation of object relational database systems treats any data type (be it a native float, an array, a string, or a compound object like an XML or HTML document) as an encapsulated type that can be stored as a value in a field of a record. Actually, these systems allow the values to be either stored directly in the record (embedded) or to be pointed to by the record (linked). This linking-embedding object model nicely accommodates the integration of database systems and file systems – files are treated as linked-objects. Queries can read and write these extended types using the same techniques they use on native types. Indeed we expect HDF and other file formats to be added as types to most database systems.

Once you can put your types and your programs inside the database you get the parallelism, non-procedural query, and data independence advantages of traditional database systems. We believe this database, file system, and programming language integration will be the key to managing and accessing peta-scale data management systems in the future.

What's wrong with files?

Everything builds from files as a base. HDF uses files. Database systems use files. But, file systems have no metadata beyond a hierarchical directory structure and file names. They encourage a do-it-yourself data model that will not benefit from the growing suite of data analysis tools. They encourage do-it-yourself access methods that will not do parallel, associative, temporal, or spatial search. They also lack a high-level query language. Lastly, most file systems can manage millions of files, but by the time a file system can deal with billions of files, it has become a database system.

As you can see, we take an ecumenical view of what a database is. We see NetCDF, HDF, FITS, and Google Map-Reduce as nascent database systems (others might think of them as file systems). They have a schema language (metadata) to define the metadata. They have a few indexing strategies and a simple data manipulation language. They have the start of non-procedural and parallel programming. And, they have a collection of tools to create, access, search, and visualize the data. So, in our view they are simple database systems.

Why scientists don't use databases today

Traditional database systems have lagged in supporting core scientific data types but they have a few things scientists desperately need for their data analysis; non-procedural query analysis, automatic parallelism, and sophisticated tools for associative, temporal, and spatial search.

If one takes the controversial view that HDF, NetCDF, FITS, and Root are nascent database systems that provide metadata and portability but lack non-procedural query analysis, automatic parallelism, and sophisticated indexing, then one can see a fairly clear path that integrates these communities.

Some scientists use databases for some of their work, but as a general rule, most scientists do not. Why? Why are tabular databases so successful in commercial applications and such a flop in most scientific applications? Scientific colleagues give one or more of the following answers when asked why they do not use databases to manage their data:

- We don't see any benefit in them. The cost of learning the tools (data definition and data loading, and query) doesn't seem worth it.
- They do not offer good visualization/plotting tools.
- I can handle my data volumes with my programming language.

- They do not support our data types (arrays, spatial, text, etc.).
- They do not support our access patterns (spatial, temporal, etc.).
- We tried them but they were too slow.
- We tried them but once we loaded our data we could no longer manipulate the data using our standard application programs.
- They require an expensive guru (database administrator) to use.

All these answers are based on experience and considerable investment. Often the experience was with older systems (a 1990 vintage database system) or with a young system (an early object-oriented database or an early version of Postgres or MySQL.) Nonetheless, there is considerable evidence that databases have to improve a lot before they are worth a second look.

Why things are different now

The thing that forces a second look now is that the file-ftp *modus operandi* just will not work for peta-scale datasets. Some new way of managing and accessing information is needed. We argued that metadata is the key to this and that a non-procedural data manipulation language combined with data indexing is essential to being able to search and analyze the data.

There is a convergence of file systems, database systems, and programming languages. Extensible database systems use object-oriented techniques from programming languages to allow you to define complex objects as native database types. Files (or extended files like HDF) then become part of the database and benefit from the parallel search and metadata management. It seems very likely that these nascent database systems will be integrated with the main-line database systems in the next decade or that some new species of metadata driven analysis and workflow system will supplant both traditional databases and the science-specific file formats and their tool suites.

Some hints of success

There are early signs that this is a good approach. One of us has shown that the doing analysis atop a database system is vastly simpler and runs much faster than the corresponding file-oriented approach[8]. The speedup is due to better indexing and parallelism.

We have also had considerable success in adding user defined functions and stored procedures to astronomy databases. The MyDB and CasJobs work for the Sloan Digital Sky Survey give a good example of moving-programs-to-the-database[9].

The BaBar experiments at SLAC manage a petabyte store of event data. The system uses a combination of Oracle to manage some of the file archive and also a physics-specific data analysis system called Root for data analysis[10].

Adaptive Finite Element simulations spend considerable time and programming effort on input, output, and checkpointing. We (Heber) use a database to represent large Finite Element models. The initial model is represented in the database and each checkpoint and analysis step is written to the database. Using a database allows queries to define more sophisticated mesh partitions and allows concurrent indexed access to the simulation data for visualization and computational steering. Commercial Finite Element packages each use a proprietary form of a "database". They are, however, limited in scope, functionality, and scalability, and are

[8] "When Database Systems Meet the Grid," M. Nieto Santisteban et. al., CIDR, 2005, http://www-db.cs.wisc.edu/cidr/papers/P13.pdf

[9] "Batch is back: CasJobs serving multi-TB data on the Web," W. O'Mullane, et. al, in preparation.

[10] "Lessons Learned from Managing a Petabyte," J. Becla and D. L. Wang, CIDR, 2005, http://www-db.cs.wisc.edu/cidr/papers/P06.pdf

typically buried inside the particular application stack. Each worker in the MPI job gets its partition from the database (as a query) and dumps its progress to the database. These dumps are two to four orders of magnitude larger than the input mesh and represent a performance challenge in both traditional and database environments. The database approach has the added benefit that visualization tools can watch and steer the computation by reading and writing the database. Finally, while we have focused on the ability of databases to simplify and speedup the production of raw simulation data, we cannot understate its core competency: providing declarative data analysis interfaces. It is with these tools that scientists spend most of their time. We hope to apply similar concepts to some turbulence studies being done at Johns Hopkins.

Summary

Science centers that curate and serve science data are emerging around next-generation science instruments. The world-wide telescope, GenBank, and the BaBar collaborations are prototypes of this trend. One group of scientists is collecting the data and managing these archives. A larger group of scientists are exploring these archives the way previous generations explored their private data. Often the results of the analysis are fed back to the archive to add to the corpus.

Because data collection is now separated from data analysis, extensive metadata describing the data in standard terms is needed so people and programs can understand the data. Good metadata becomes central for data sharing among different disciplines and for data analysis and visualization tools.

There is a convergence of the nascent-databases (HDF, NetCDF, FITS,..) which focus primarily on the metadata issues and data interchange, and the traditional data management systems (SQL and others) that have focused on managing and analyzing very large datasets. The traditional systems have the virtues of automatic parallelism, indexing, and non-procedural access, but they need to embrace the data types of the science community and need to co-exist with data in file systems. We believe the emphasis on extending database systems by unifying databases with programming languages, so that one can either embed or link new object types into the data management system, will enable this synthesis.

Three technical advances will be crucial to scientific analysis: (1) extensive metadata and metadata standards that will make it easy to discover what data exits, make it easy for people and programs to understand the data, and make it easy to track data lineage; (2) great analysis tools that allow scientists to easily ask questions, and to easily understand and visualize the answers; and (3) set-oriented data parallelism access supported by new indexing schemes and new algorithms that allow us to interactively explore peta-scale datasets.

The goal is a *smart notebook* that empowers scientists to explore the world's data. Science data centers with computational resources to explore huge data archives will be central to enabling such notebooks. Because data is so large, and IO bandwidth is not keeping pace, moving code to data will be essential to performance. Consequently, science centers will remain the core vehicle and federations will likely be secondary. Science centers will provide both the archives and the institutional infrastructure to develop these peta-scale archives and the algorithms and tools to analyze them.

PITAC's Look at Computational Science

Dan Reed
UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL

In June 2004, the President's Information Technology Advisory Committee (PITAC) was charged by John Marburger, the President's Science Adviser, to respond to seven questions regarding the state of computational science:

1. How well is the Federal Government targeting the right research areas to support and enhance the value of computational science? Are agencies' current priorities appropriate?

2. How well is current Federal funding for computational science appropriately balanced between short term, low risk research and longer term, higher risk research? Within these research arenas, which areas have the greatest promise of contributing to breakthroughs in scientific research and inquiry?

3. How well is current Federal funding balanced between fundamental advances in the underlying techniques of computational science versus the application of computational science to scientific and engineering domains? Which areas have the greatest promise of contributing to breakthroughs in scientific research and inquiry?

4. How well are computational science training and research integrated with the scientific disciplines that are heavily dependent upon them to enhance scientific discovery? How should the integration of research and training among computer science, mathematical science, and the biological and physical sciences best be achieved to ensure the effective use of computational science methods and tools?

5. How effectively do Federal agencies coordinate their support for computational science and its applications in order to maintain a balanced and comprehensive research and training portfolio?

6. How well have Federal investments in computational science kept up with changes in the underlying computing environments and the ways in which research is conducted? Examples of these changes might include changes in computer architecture, the advent of distributed computing, the linking of data with simulation, and remote access to experimental facilities.

7. What barriers hinder realizing the highest potential of computational science and how might these be eliminated or mitigated?

Since that time, I have chaired a PITAC subcommittee composed of Ruzena Bajcsy (UC-Berkeley), Manuel Fernandez (SI Ventures), José-Marie Griffiths (UNC-CH) and Randall Mott (Dell) to prepare a response to these questions. The subcommittee has also been assisted by two consultants, Chris Johnson (Utah) and Jack Dongarra (Tennessee). The subcommittee has solicited input at public meetings and held a Birds-of-a-Feather (BoF) Town Hall meeting at SC04 in November 2004.

Based on this input and extended discussions, the subcommittee has developed a working definition of computational science, which it is using to prepare a draft report. This definition, which is still in flux, attempts to recognize the interplay among algorithms and software, computer and information science and infrastructure:

> *Computational science is a rapidly growing multidisciplinary field that uses advanced computing capabilities to understand and solve complex problems. Computational science fuses three distinct elements: (a) algorithms(numerical and non-numerical) and modeling and simulation software developed to solve science (e.g., biological physical, and social), engineering and humanities problems; (b) computer and information science that develops and optimizes the advanced system hardware, software, networking, and data management components needed to solve computationally demanding problems; and (c) the computing infrastructure that supports both the science and engineering problem solving and the developmental computer and information science.*
>
> *Computational science has several advantages over experimentation and theory. First, it often enables solution of problems more efficiently, more rapidly and less expensively. Second, it can solve problems computationally that otherwise could not be solved safely. Finally, it can solve problems whose solution is otherwise impossible (e.g., due to the inability to recreate experimental conditions).*

The subcommittee has issued two interim working summaries, which are available on the web site of the National Coordination Office (NCO)[1]. These summaries contain draft findings and recommendations, which are still evolving. Preliminary findings, reported at the November PITAC meeting, include the following:

- Computing has become the third component of scientific discovery, complementing theory and experiment.
- The explosive growth in the resolution of sensors and scientific instruments has led to unprecedented volumes of experimental data. Computational science now broadly includes modeling, simulation and scenario assessment using sensor data from diverse sources.
- Complex multidisciplinary problems, from public policy through national security to scientific discovery and economic competitiveness, have emerged as new drivers of computational science, complementing the historical focus on single disciplines.
- Developing leading edge computational science applications is a complex process involving teams of people that must be sustained for a decade or more to yield the full fruits of investment.
- Short-term investment and limited strategic planning have led to excessive focus on incremental research rather than on the long-term research with lasting impact that can solve critical problems.
- Interdisciplinary education in computational science and computing technologies is inadequate, reflecting the traditional disciplinary boundaries in higher education. Only systemic change to university organizational structures will yield the needed outcomes.
- Computational science would benefit from a roadmap outlining decadal priorities for investment, with a clear assessment of those priorities derived from a survey of the problems and challenges. Agencies could then respond to these with a strategic plan in recognition of those priorities and funding requirements.

The subcommittee invites comments on responses to the charge, its preliminary findings and draft recommendations. Comments can be sent to "pitac-comments@nitrd.gov".

[1] http://www.nitrd.gov/pitac/meetings/

CTWatch QUARTERLY

February 2005

TRENDS IN HIGH PERFORMANCE COMPUTING

GUEST EDITOR

JACK DONGARRA, UTK/ORNL

http://www.ctwatch.org/

http://icl.cs.utk.edu/ http://www.ncsa.uiuc.edu/ http://www.sdsc.edu/

CTWatch Quarterly is a publication of the CyberInfrastruture Partnership (CIP).

http://www.ctwatch.org/quarterly/ quarterly@ctwatch.org