Stanford University

C I S A C

Center for International Security and Cooperation

The Center for International Security and Cooperation, part of Stanford University's Institute for International Studies, is a multidisciplinary community dedicated to research and training in the field of international security. The Center brings together scholars, policymakers, scientists, area specialists, members of the business community, and other experts to examine a wide range of international security issues.

Center for International Security and Cooperation
Stanford University
Encina Hall
Stanford, California 94305-6165

(415) 723-9625

http://www.stanford.edu/group/CISAC/

High-Performance Computing, National Security Applications, and Export Control Policy at the Close of the 20th Century

Seymour E. Goodman
Peter Wolcott
Patrick T. Homer

May 1998

Seymour E. Goodman is professor of management information systems and policy at the University of Arizona. He studies the international dimensions of the information technologies and related public policy issues. He was a CISAC Science Fellow from 1994–1997 and is currently director of the Project on Information Technology and National Security at Stanford.

Peter Wolcott is assistant professor of information systems and quantitative analysis at the University of Nebraska at Omaha. His research interests relate to the international aspects of information technologies, particularly high-performance computing.

Patrick T. Homer is assistant professor of computer science at the University of Arizona Sierra Vista. In addition to HPC export controls, his research interests include system support for heterogeneity in scientific applications, distributed systems, and Internet applications for K–12 education.

This study was sponsored and funded by the Under Secretary of Commerce (BXA) and the Deputy Assistant Secretary of Defense for Counterproliferation Policy. William Foster, of the University of Arizona, provided administrative assistance. Megan Hendershott and Flavia Herrod provided editorial support. Diane Goodman provided administrative and editorial assistance. The opinions expressed here are those of the authors and do not necessarily represent positions of the Center, its supporters, the United States Government, or Stanford University.

ISBN 0-935371-50-8
Printed in the United States of America

Contents

Executive Summary vii
- Basic Premises vii
- Industry Trends Impacting Export Control viii
- Controllability of HPC Platforms and Performance ix
- National Security Applications xi
- Countries of National Security Concern xii
- Export Control at the Turn of the Century xii
- Outstanding Issues, Decisions, Concerns xiii
- Reference xiv

Chapter 1: Introduction 1
- The Basic Premises behind Export Control Thresholds 1
- Deriving a Control Threshold 2
- References 4

Chapter 2: Industry Trends Affecting Export Control 5
- Overview of Key Industry Trends 5
- Recent Trends in Processor Performance and Capability 6
- Future Microprocessor Developments and Challenges 9
- Significance of Microprocessor Developments for HPC 13
- Gluing the Pieces Together: Recent Trends in Systems Architecture and Capability 15
- Implications for the Export Control Regime 24
- References 25

Chapter 3: HPC Controllability and Export Control Thresholds 27
- Kinds of Controllability 27
- Scalability and the Export Control Regime 37
- Barriers to Scalability 40
- Establishing a Lower Bound 41
- Conclusions 43
- References 44

Chapter 4: Applications of National Security Interest 45
- Introduction 45
- Key application findings 47
- Computing Platforms and Scientific Applications 51
- Numerical Weather Forecasting 55
- Cryptographic Applications 69
- Signal and Image Processing 71
- Computational Fluid Dynamics 78
- Computational Chemistry and Materials 95
- Computational Structures and Mechanics 100
- Computational Electromagnetics and Acoustics 108
- Forces Modeling and Simulation 114
- Littoral Warfare 119
- Nuclear Applications 120
- Conclusion 126
- References 131

Chapter 5: Policy Issues, Options, and Implications 137
Policy Options for a Control Threshold 137
Export Control at the Turn of the Century 144
References 149

Appendix A: Applications of National Security Importance 150

Appendix B: People and Organizations 163

Appendix C: Glossary 168

Figures

Figure 1–1. Minimum, actual, and maximum computing power available for F–22 design 2
Figure 1–2. Establishing a range for a viable control threshold 3
Figure 2–1. Performance of microprocessors in volume production 6
Figure 2–2. Microprocessor feature sizes 7
Figure 2–3. High, low, and average transistor counts for RISC microprocessors, compared with Moore's Law 7
Figure 2–4. High, low, and average clock frequency for RISC microprocessors 8
Figure 2–5. Instructions issued per clock cycle 9
Figure 2–6. Industry projections of microprocessor performance 9
Figure 2–7. Percentage of Top500 systems based on commercial microprocessors/ proprietary processors 14
Figure 2–8. Price/performance of HPC systems 15
Figure 2–9. Memory hierarchy and interconnects 16
Figure 2–10. Interconnect latency and bandwidth 19
Figure 3–1. Sampling of platforms with low controllability (4Q 1997) 34
Figure 3–2. Sampling of platforms with moderate controllability (4Q 1997) 35
Figure 3–3. Configuration, attainable, and maximum performance of non-scalable systems 36
Figure 3–4. Configuration, attainable, and maximum performance for scalable systems 37
Figure 3–5. Minimum, maximum, and end-user attainable performance of selected HPC systems 38
Figure 3–6. Threshold of uncontrollable (attainable) performance 39
Figure 3–7. End-user attainable performance of categories of HPC systems 42
Figure 4–1. CTP of sample upper-bound applications 48
Figure 4–2. Conversion path for vector to parallel code 50
Figure 4–3. Grid examples 52
Figure 4–4. Weather applications 66
Figure 4–5. Data processing in sensor-based applications 72
Figure 4–6. Synthetic aperture radar 73
Figure 4–7. Origin2000 performance on RT_STAP benchmarks 77
Figure 4–8. CTP or current CFD applications 90
Figure 4–9. Radar reflection from various shapes 109
Figure 4–10. One-dimensional parallelizations 111
Figure 4–11. Communications between nodes and primary router 117
Figure 4–12. Communication patterns between clusters 117
Figure 4–13. Contribution of HPC to nuclear weapons development with and without test data 123
Figure 4–14. Application histogram 127
Figure 5–1. Distribution of applications examined in this study 143

Tables

Table ES–1.	Categories of computers based on commercial microprocessors (4Q 1997)	iii
Table 2–1.	Multimedia support in major microprocessor families	13
Table 2–2.	Memory latency	17
Table 2–3.	Current HPC systems and trends	23
Table 3–1.	HPC chassis	28
Table 3–2.	Categories of dependence on vendors	30
Table 3–3.	Impact of installed base on controllability (circa 4Q 1997)	30
Table 3–4.	Examples of platform controllability (4Q 1997)	32
Table 3–5.	Selected HPC systems from Tier 3 countries	33
Table 3–6.	Categories of computing platforms (4Q 1997)	41
Table 4–1.	DoD Computational Technology Areas (CTA)	46
Table 4–2.	Current and projected weather forecasting models used by FNMOC	58
Table 4–3.	Global ocean model projected deployment	63
Table 4–4.	Weather applications	68
Table 4–5.	Mission parameters for X-band SAR sensors	74
Table 4–6.	SIR-C/X-SAR processing times on the Intel Paragon	74
Table 4–7.	Signal processing systems sold by Mercury Computer Systems	75
Table 4–8.	Approximation stages in CFD solutions	79
Table 4–9.	Results of improvements to tiltrotor simulation	81
Table 4–10.	Results of porting two CFD applications from vector to parallel	82
Table 4–11.	Evolution of submarine CFD applications	84
Table 4–12.	Performance of parafoil CFD applications	88
Table 4–13.	CFD applications	91
Table 4–14.	Computational chemistry applications	99
Table 4–15.	Structural mechanics applications	107
Table 4–16.	Computational electromagnetic applications	112
Table 4–17.	Submarine design applications	113
Table 4–18.	Synthetic forces experiments	118
Table 4–19.	Computing requirements for first-principles simulations	124
Table 4–20.	Nuclear weapons related applications	125
Table 4–21.	Applications between 5000 and 10,000 Mtops	128
Table 4–22.	Applications between 10,000 and 15,000 Mtops	128
Table 4–23.	Applications between 15,000 and 20,000 Mtops	129
Table 5–1.	Categories of computing platforms (4Q 1997)	140
Table 5–2.	Trends in lower bound of controllability (Mtops)	140
Table 5–3.	Selected sample of national security applications and the rising lower bound of controllability	142

Executive Summary

Basic Premises

If high-performance computing (HPC) export control policy is to be effective, three basic premises must hold:

(1) There exist problems of great national security importance that require high-performance computing for their solution, and these problems cannot be solved, or can only be solved in severely degraded forms, without such computing assets.

(2) There exist countries of national security concern to the United States that have both the scientific and military wherewithal to pursue these or similar applications.

(3) There are features of high-performance computers that permit effective forms of control.

This study applies and extends the methodology established in *Building on the Basics* [1]. Its objective has been to study trends in HPC technologies and their application to problems of national security importance to answer two principal questions:

- Do the basic premises continue to be satisfied as the 20th century draws to a close?
- In what range of performance levels might an export-licensing threshold be set so that the basic premises are satisfied?

The study concludes that export controls on HPC hardware are still viable, although much weaker than in the past. In particular, while applications of national security interest abound, it is increasingly difficult to identify applications that strongly satisfy all three basic premises, i.e. are of extreme national security importance *and* would likely be effectively pursued by countries of national security concern *and* would be severely retarded without levels of computing performance that could be effectively controlled.

Industry Trends Impacting Export Control

HPC industry trends having the strongest impact on the export control regime include:

(1) Developments that increase the performance of HPC products within given market/price niches, and

(2) Developments that enhance scalability and, more generally, the ability to apply the computing power of multiple smaller systems to the solution of a single computational problem.

Some of the most significant developments are advances in microprocessors, interconnects, and system architectures.

Microprocessors

Microprocessor performance will continue to improve dramatically through the end of the century. In 1997, nearly all microprocessor developers had volume products above 500 Mtops. In 1998, the first microprocessors to exceed 1500 Mtops will be in volume production. By 1999, some microprocessors will exceed 2000 Mtops; in 2000, processors of nearly 7000 Mtops will reach volume production. Industry projections are that in 2001 microprocessors of 7–10,000 Mtops will ship. Improvements in performance will come from a combination of more functional units, multiple central processing units on a chip, on-chip graphics processors, and increased clock frequency. Industry feels that such improvements can be made without having to make significant technological breakthroughs this century.

Interconnects

In multiprocessor systems, actual performance is strongly influenced by the quality of the interconnect that moves data among processors and memory subsystems. Traditionally, interconnects could be grouped into two categories: proprietary high-performance interconnects used within individual vendor products, and industry standard interconnects such as local area networks. The two categories represent very different qualities, measured in bandwidth and latency. In recent years, a new class of interconnect has emerged represented by products from Myricom, Digital Equipment Corporation (DEC), Essential Communications, Dolphin Interconnect Solutions, Inc., and others. These "clustering interconnects" offer much higher bandwidth and lower latency than local area networks. While they can be used to integrate a number of individual systems into a single configuration that can perform useful work on many applications, they still have shortcomings compared to proprietary high-performance interconnects. These shortcomings may include lower bandwidth, higher latency, greater performance degradation in large configurations, or immature system software environments.

The implication for the export control regime is that commercially available clustering interconnects, while useful, should not be considered equal substitutes for the high-performance, proprietary interconnects used within most high-performance computing systems today.

System Architectures

The dominant trend in overall system architecture in recent years has been toward either a distributed shared memory system or a hierarchical modular system. Vendors today are pursuing one strategy or the other, or both simultaneously. In distributed shared memory systems, memory is physically distributed but logically shared. A consequence is that memory access time may not be uniform. In hierarchical modular systems, multiprocessor nodes have

memory that is logically and physically shared, while between nodes a distributed memory message passing paradigm is used. It is unlikely that dramatically different architectures will be used within the next three to five years. The implication for the export control regime is that the difference between large and small configuration systems, from the perspective of user applications and systems management, is decreasing. Users of small configurations will be better positioned to test and improve their applications without the need to use the larger systems. The larger systems will only be needed for those runs that require greater resources.

Controllability of HPC Platforms and Performance

Building on the Basics asserted that there were computational performance levels that could be attained so easily that control thresholds set below these levels would be ineffective. The principal factors influencing the so-called lower bound of controllability are:

(1) the performance of computing platforms that have qualities (size, price, numbers installed, vendor distribution channels, age, dependence on vendor support) that make them difficult to monitor;

(2) the scalability of platforms; and

(3) the performance of systems available from foreign sources not supporting U.S. export control policies.

Controllability of Platforms

The performance of systems tends to correlate with factors that influence controllability. More powerful systems tend to be sold in smaller numbers, be more expensive, have larger configurations, and require greater amounts of vendor support for installation and maintenance. Table ES–1 illustrates a few of the controllability factors, together with performance, of categories of computer systems based on commercial microprocessors available in 4Q 1997.

Type	*Units installed*	*Price*	*End-user attainable performance*
Multi-rack HPC systems	100s	$750K–10s of millions	20K+ Mtops
High-end rack servers	1000s	$85K–1 million	7K–20K Mtops
High-end deskside servers	1000s	$90–600K	7K–11K Mtops
Mid-range deskside servers	10,000s	$30–250K	800–4600 Mtops
UNIX/RISC workstations	100,000s	$10–25K	300–2000 Mtops
Windows NT/Intel servers	100,000s	$3–25K	200–800 Mtops
Laptops, uniprocessor PCs	10s of millions	$1–5K	200–350 Mtops

Table ES–1. Categories of computers based on commercial microprocessors (4Q 1997)

Controllability is a function not only of the nature of technologies and markets, but also of the resources brought to bear on enforcement. A key decision for the U.S. Government is which of these categories are considered controllable, and which are not.

Scalability

Scalability, the ability to increase the capability of a computer system incrementally by adding processors, memory, and input/output facilities, has become a principal design objective of most HPC systems of the 1990s. It has become increasingly possible to upgrade systems in the

field without a great deal of vendor support. Scalability is a problem for export control. When systems are extensively and easily scalable without vendor support, end-users may acquire, completely legitimately, multiple small configurations lying below the control threshold and then, on their own, recombine CPUs and memory to create a single configuration with a performance above the control threshold. For example, some rack-based systems sold in 1997–1998 may be sold in configurations of less than 1000 Mtops, and scaled by competent end-users to over 15,000 Mtops by adding CPU, memory, and I/O boards.

The government has two main options for coping with scalability. First, make licensing decisions based on the performance of a configuration about to be sold, and make government officials and vendors responsible for maintaining close and long-term oversight of end-user installations. This option, the current practice, requires a high level of post-sale vigilance. As the number of foreign installations grows and the ease with which systems can be scaled increases, the cost of enforcement rises and the certainty of success decreases.

A second option is to make licensing decisions based on the end-user attainable performance of a system when making licensing decisions, rather than the performance of a specific configuration. The *end-user attainable performance* is defined as the performance of the largest configuration of an individual, tightly coupled system an end-user could assemble without vendor support, using only the hardware and software provided with lesser configurations. A two-processor rack-based system would be treated differently from a two-processor deskside system, because the former can be upgraded relatively easily to a higher performance than the latter. Systems requiring extensive vendor support for upgrades, such as traditional supercomputers, have end-user attainable performance precisely equal to the configuration installed.

The second option is a more conservative approach to licensing in that it assumes a worst-case scenario, that end-users will increase the performance of a configuration they obtain to the extent they can. At the same time, however, it reduces the need to detect post-shipment upgrades, for upgrading would have been taken into account during the licensing process. Furthermore, this option provides policy makers with a more precise means to distinguish between systems with different controllability characteristics.

Foreign Sources of HPC Systems

The controllability of HPC systems today is not greatly influenced by HPC systems originating in foreign countries, with the notable exception of Japan. The international market is overwhelmingly dominated, at least in the mid- and high-end server categories, by U.S. companies and their international business partners. Indigenous systems from countries such as Russia, India, and China are not internationally competitive. Indigenous systems from these countries with performance above 2000 Mtops are available in only single-digit units or tens of units at best, rather than thousands or tens of thousands of units.

Establishing a Lower Bound of Controllability

In establishing a lower bound of controllability, we factored in a time lag needed for a given product's market to mature. In the case of rack-based systems, we conservatively estimated this time lag to be about two years. Smaller systems are sold more quickly and in higher volumes. We have used a one-year time lag for mid-range servers. Given these time lags, we estimate that the end-user attainable performance of rack-based systems whose markets have matured to lie between 15,000 and 30,000 Mtops in 2000, depending on the vendor. The end-user attainable performance for mid-range, deskside systems will reach approximately 6500 Mtops that same year. The latter figure will rise dramatically in late 2000 or 2001 as tens or hundreds of thousands of mid-range servers in the 15,000+ Mtops range are shipped. Configurations of four or eight CPUs, with each CPU measuring 4–7000 Mtops, will constitute the "sweet spot" of the mid-range market.

National Security Applications

A major finding of this study is that there is no lack of computationally demanding applications of national security interest, nor is there likely to be in the foreseeable future. The computational requirements of moving to larger problem sizes, finer resolutions, multidisciplinary problems, etc., create demands for compute cycles and memory that are, for all practical purposes, insatiable. The first basic premise is, and will continue to be, satisfied.

The major change in applications over the last several years has been the extent to which practitioners have used parallel computing platforms not only in research settings, but also in production environments. The combination of mature parallel hardware/software platforms from vendors, platform-independent application programming interfaces like the Message Passing Interface (MPI), and industry trends toward microprocessor-based systems have prompted practitioners to make the transition from parallel vector-pipelined platforms to massively parallel platforms for most high-end applications.

The methodology used for this report and its predecessor requires the establishment of an "upper bound" for the control threshold that lies at or above the lower bound, but below the performance requirements of key applications of national security concern, or clusters of national security applications. This study has cataloged in detail an extensive number of national security applications. The national security community must decide which of these have the greatest significance for the nation's security. Are there performance levels at which there is a relatively greater density of national security applications? There appear to be, although these tend to be found around the performance levels of "workhorse" computing systems widely used in the national security community. We have observed application clusters at:

- 4000–6000 Mtops. Key applications in this range include JAST aircraft design, non-acoustic anti-submarine warfare sensor development, and advanced synthetic aperture radar computation. The number of applications at this performance range is growing rapidly with the increase in the number of systems in this range.
- 8000–9000 Mtops. Applications here include bottom-contour modeling of shallow water in submarine design, some synthetic aperture radar applications, and algorithm development for shipboard infrared search and track.
- 10,000–12,000 Mtops. Applications here include global and regional weather forecasting, image processing, and moderate-sized particle dynamics problems.
- 15,500–17,500 Mtops. Applications here include modeling turbulence around aircraft under extreme flight conditions, and moderate-sized battlefield simulations.
- 20,000–22,000 Mtops. A sizable cluster of applications here includes advanced weather forecasting, impact of blasts on underground and surface structures, advanced aircraft design, and modeling of complete submarine hulls.

The selection of a control threshold is facilitated when there is a clear understanding of which applications are of greatest national security concern. While this study discusses a large number of applications of national security interest, it is not clear which of these are of extreme national security concern. The national security community must bear the responsibility of determining which are considered critical to U.S. interests.

Countries of National Security Concern

The second basic premise of export control policy states that there exist countries of national security concern with the scientific and military wherewithal to pursue computationally demanding applications of national security importance. Capability depends not only on having (1) the necessary hardware and systems software, but also (2) the application codes, valid test data, and correct input data, and (3) the expertise necessary to use the codes and interpret the results correctly. (2) and (3) are often more of a limiting factor than (1). There exist a few clear examples of foreign countries having the expertise necessary to pursue particular applications successfully. Nuclear weapons development and stockpile stewardship is one, although the computational performance necessary for weapons development, given the necessary test data, is at or below today's UNIX/RISC workstations. At the level of computing readily available today and the near future, additional computing power does not compensate for the lack of test data. Military-grade weather forecasting is another. A critical question, which we have been unable to pursue satisfactorily in this study, is which countries are able and likely to productively use HPC to pursue which applications. It does not appear that the U.S. Government is effectively gathering such intelligence in a systematic fashion.

Export Control at the Turn of the Century

This study has concluded that the export control regime can remain viable for the next several years and offers policy makers a number of alternatives for establishing thresholds and licensing practices that balance national security interests and the realities of HPC technologies and markets. Nevertheless, there are a number of trends that will make the regime less successful in achieving its objectives than has been the case in the past. In the future, the probability will increase that individual restricted end-use organizations will be able to successfully acquire or construct a computing system to satisfy a particular application need. A number of factors contribute to this "leakage."

First, if policy makers do decide that systems with installed bases in the thousands or tens of thousands are controllable, it is inevitable that individual units will find their way to restricted destinations.

Second, industry is working intensively toward the goal of seamless scalability, enhanced systems management, single-system image, and high efficiency across a broad range of performance levels. Systems with these qualities make it possible for users to develop and test software on small configurations yet run it on large configurations. An inability to gain access to a large configuration may limit a user's ability to solve certain kinds of problems, but will not usually inhibit their ability to develop the necessary software.

Third, clustered systems are improving in the quality of both their interconnects and supporting software. Foreign users are able to cluster desktop or deskside systems into configurations that perform useful work on some applications. Such systems are not the equivalent of vendor-supplied, fully integrated systems. However, because it is difficult to prevent the construction of clustered systems, the control regime will leak.

Finally, given the nature and volume of foreign sales of today's HPC market, where most U.S. vendors sell approximately 50 percent of their output abroad, monitoring and control of large numbers of systems is difficult. It is exacerbated by the apparent lack of an effective multilateral regime governing re-exports. It is, moreover, difficult to imagine effective re-export controls in today's political and commercial world that would prevent the acquisition of one or two machines by many of the end-users of greatest concern.

Nevertheless, even an imperfect export control regime offers a number of benefits to U.S. national security interests. First, licensing requirements at appropriate levels force vendors and government agencies to pay close attention to who the end-users are and what kinds of

applications they are pursuing. Second, the licensing process provides government with an opportunity to review and increase its knowledge about end-users brought to its attention. The government should improve its understanding of end-users of concern so that it can make better decisions regarding those end-users. Finally, while covert acquisition of high-performance computers is much easier today than in the past, users without legitimate access to vendor support are at a disadvantage, especially for operational or mission-critical applications.

Outstanding Issues, Decisions, Concerns

Periodic reviews. This study documents the state of HPC technologies and applications during 1997–early 1998, and makes some conservative predictions of trends in the next two to five years. The rapid pace of change in this industry continues unabated. The future viability of the export control policy will depend on its keeping abreast of change and adapting in an appropriate and timely manner. When based on accurate, timely data and an open analytic framework, policy revisions become much more sound, verifiable, and defensible. There is no substitute for periodic reviews and modification of the policy. While annual reviews may not be feasible given policy review cycles, the policy should be reviewed no less frequently than every two years.

Controllability of computing systems and the objectives of HPC export control. The U.S. Government must decide which categories of computing systems are controllable. The answer depends not only on technology and market features and the enforcement resources to be applied, but also on the ultimate goal of the policy. Is the principal objective (a) to prevent individual HPC systems from being acquired by a small and specific set of often well-funded foreign end-users of greatest national security concern, or (b) to prevent large numbers of systems from penetrating a nation's civilian and military communities? Since (a) is more difficult than (b) at a given control threshold, the determination of which systems are controllable will be influenced by the policy's ultimate objective.

Controlling scalability. Policy makers must decide how they will prevent end-users from scaling systems beyond the level authorized in export licenses. Current practice holds vendors and government officials responsible for monitoring system configurations after shipment, an increasingly difficult and uncertain endeavor. An alternative approach would make licensing decisions based on the end-user attainable performance of a system. While more conservative, it relieves much of the burden and uncertainty of post-shipment monitoring by controlling for scalability at the time an export license is granted rather than after the system has been shipped to an end-user.

Applications of national security importance. The current study has surveyed a substantial number of applications of national security importance to determine whether or not there are applications that can and should be protected using export controls of high-performance computing. While the study has enumerated a number of applications that *may* be protected, it has not answered the question of which applications are of greatest national security importance and *should* be protected. This question can only be answered by the national security community, and it is important that it be answered. If an application area lacks a constituency willing to defend it in the public arena, it is difficult to argue that it should be a factor in setting export control policy.

During the Cold War, when the world's superpowers were engaged in an extensive arms race and building competing spheres of influence, it was relatively easy to make the argument that certain applications relying on high-performance computing were critical to the nation's security. Because of changes in the geopolitical landscape, the nature of threats to U.S. national security, and the HPC technologies and markets, the argument appears to be more difficult to make today than in the past. We have found few voices in the applications community who feel that export control on controllable levels of HPC hardware is vital to the

protection of their application. Constituencies for the nuclear and cryptographic applications exist, although they are not unanimous in their support of the policy. An absence of constituencies in other application areas who strongly support HPC hardware export controls may reflect an erosion of the basic premises underlying the policy. If this is the case, it should be taken into account; where such constituencies exist, they should enter into the discussion.

Reference

[1] Goodman, S. E., P. Wolcott, and G. Burkhart, *Building on the Basics: An Examination of High-Performance Computing Export Control Policy in the 1990s*, Center for International Security and Arms Control, Stanford University, Stanford, CA, 1995.

Chapter 1: Introduction

This study is a successor to *Building on the Basics: High-Performance Computing Export Control in the 1990s* [1] (published in a slightly updated form in [2]). That study established a framework and methodology for deriving an export control threshold by taking into account both applications of national security concern and the technological and market characteristics of a rapidly changing high-performance computing (HPC) industry. One objective of the study was to establish a process for updating the policy that would be transparent, objective, defensible, and repeatable. The current study, undertaken two years after the first, applies the methodology and framework of the first study to determine (a) whether or not a control threshold exists that could be part of a viable export control policy, and (b) what the range of possible thresholds might be.

In addition to recommending that the process be repeated regularly, the earlier study recommended a much more comprehensive analysis of applications of national security concern than was possible in 1995. Consequently, this study provides greatly enhanced coverage of national security applications, their computational nature and requirements, and the manner in which such applications are pursued given the changes in the HPC industry.

This introduction provides a brief review of the framework developed in *Building on the Basics*. Chapter 2 analyzes key trends in the HPC industry from the perspective of those elements of significance to the export control regime. Chapter 3 provides an expanded analysis of the concept of the lower bound of controllability and establishes a set of options for s in establishing the lower bound of a range of viable control thresholds. Chapter 4 provides extensive coverage of a broad spectrum of national security applications to give s insight into the upper bound for a control threshold. Chapter 5 integrates chapters 3 and 4 into a concrete set of policy options and implications.

The Basic Premises behind Export Control Thresholds

The HPC export control policy has been successful in part because it has been based on three premises that were largely true for the duration of the Cold War:

(1) There are problems of great national security importance that require high-performance computing for their solution, and these problems cannot be solved, or can only be solved in severely degraded forms, without such computing assets.

(2) There are countries of national security concern that have both the scientific and military wherewithal to pursue these or similar applications.

(3) There are features of high-performance computers that permit effective forms of control.

If the first two premises do not hold, there is no justification for the policy. If the third premise does not hold, an effective export control policy cannot be implemented, regardless of its desirability.

While a strong case can be made that all three premises held during the Cold War, there have been significant changes that impact this policy. In particular, following the dissolution of the Soviet Union, threats to national security have become smaller, but more numerous; there have been dramatic advances in computing technologies; and the use of HPC within the U.S. national security community has expanded.

If the premises are still valid, it should be possible to derive a control threshold in a way that is explicit, justifiable, and repeatable. If the premises are not valid, then the analysis should clearly illustrate why no effective control policy based on the premises is possible.

Deriving a Control Threshold

The first premise postulates that there exist applications with high minimum computational requirements. In other words, if the minimum computational resources (especially, but not exclusively, performance) are not available, the application cannot be performed satisfactorily. To establish the performance requirements, we asked applications practitioners to identify the computer configuration that they would need to carry out the application. The composite theoretical performance (CTP) of such a configuration was used to quantify the Mtops used for this application.[1]

In some cases, the configuration used was more powerful than was necessary to do the application satisfactorily. Figure 1–1 shows the performance of the minimum acceptable configuration and the configuration actually used for the F-22 aircraft design application.

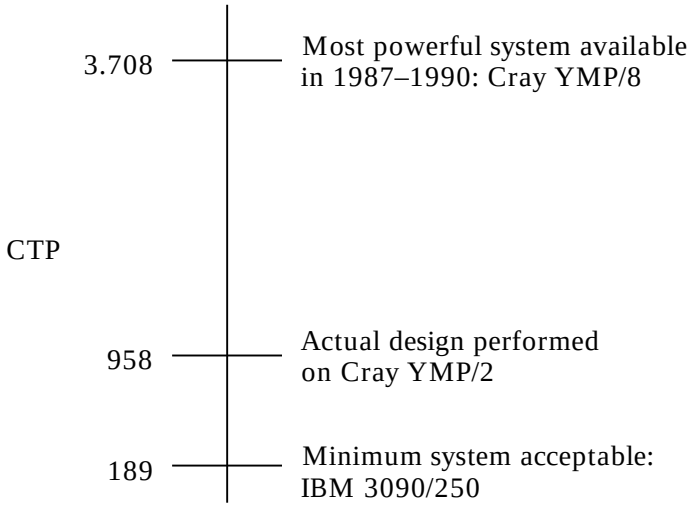

Figure 1–1. Minimum, actual, and maximum computing power available for F-22 design

The third premise requires that systems above the minimum performance level of a particular application have characteristics that permit their export and use to be closely monitored, controlled, and when necessary, denied. If there exist systems that cannot be controlled whose performance exceeds the minimum necessary to carry out the application, then the U.S.

[1] The composite theoretical performance is measured in millions of theoretical operations per second (Mtops). Mtops ratings consider both floating-point and non-floating-point operations, and account for variations in word length, numbers of processors, and whether the system is based on a shared memory or distributed memory paradigm.

Government will be unable to control that application solely by trying to deny the necessary computer power.

In order for the control regime to be justifiable, there must be some applications that satisfy both the first and third premises.

Over time, the computational performance of the most powerful uncontrollable system(s) rises. As it rises, it overtakes the minimum computing requirements of individual applications. If the minimum and actual performance levels for particular applications are plotted over time, the dynamic may be illustrated as shown in Figure 1–2. For illustration purposes, this figure uses only hypothetical data.

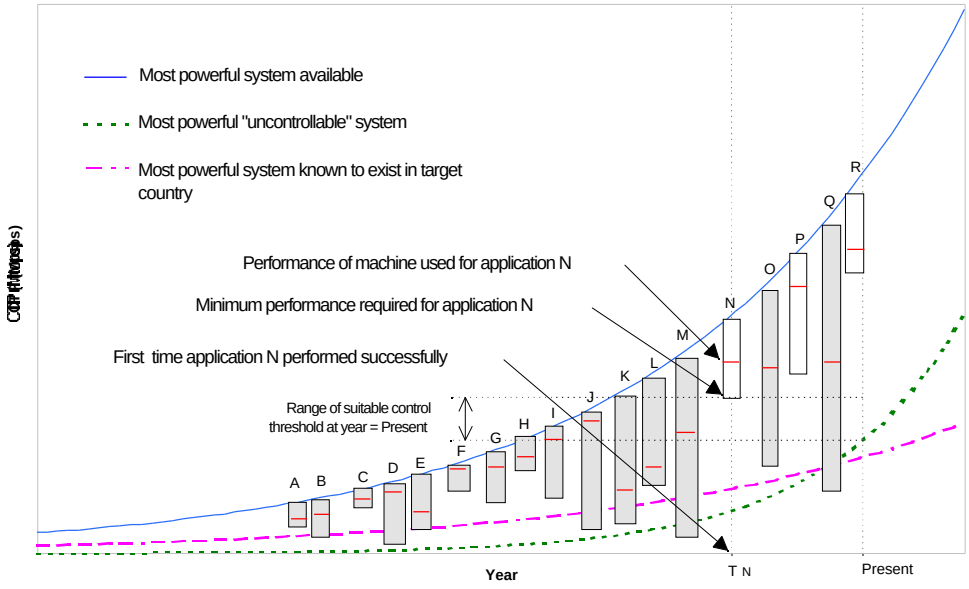

Figure 1–2. Establishing a range for a viable control threshold

Under current practice, a control threshold is established at a particular point in time and remains in effect until revised through policy changes. The set of viable thresholds—those that satisfy the three basic premises—must at the same time lie between two bounds: The "lower bound" is determined by the level of the most powerful uncontrollable systems. The "upper bound" is determined by those national security applications whose minimum performance requirements lie above the lower bound. In Figure 1–2, applications *N*, *P*, and *R* are those that can be protected by export control of HPC hardware.

The selection of a specific control threshold further takes into account the nature of the computer market for systems whose performance, measured in Mtops, lies within the range between the lower and upper bounds. Ideally, a control threshold would be established below a point where there were numerous or particularly significant applications of national security concern, but above the performance level of systems enjoying large markets.

The following chapters supply data for the model. Chapter 2 discusses industry trends that impact the export control policy. Chapter 3 discusses the determination of a lower bound. Chapter 4 discusses the computational requirements of a substantial number of applications of national security concern. Chapter 5 integrates the results of the previous chapters into a

discussion of policy options and implications.

References

[1] Goodman, S. E., P. Wolcott, and G. Burkhart, *Building on the Basics: An Examination of High-Performance Computing Export Control Policy in the 1990s,* Center for International Security and Arms Control, Stanford University, Stanford, CA, 1995.

[2] Goodman, S., P. Wolcott, and G. Burkhart, *Executive Briefing: An Examination of High-Performance Computing Export Control Policy in the 1990s,* IEEE Computer Society Press, Los Alamitos, 1996.

Chapter 1: Introduction

This study is a successor to *Building on the Basics: High-Performance Computing Export Control in the 1990s* [1] (published in a slightly updated form in [2]). That study established a framework and methodology for deriving an export control threshold by taking into account both applications of national security concern and the technological and market characteristics of a rapidly changing high-performance computing (HPC) industry. One objective of the study was to establish a process for updating the policy that would be transparent, objective, defensible, and repeatable. The current study, undertaken two years after the first, applies the methodology and framework of the first study to determine (a) whether or not a control threshold exists that could be part of a viable export control policy, and (b) what the range of possible thresholds might be.

In addition to recommending that the process be repeated regularly, the earlier study recommended a much more comprehensive analysis of applications of national security concern than was possible in 1995. Consequently, this study provides greatly enhanced coverage of national security applications, their computational nature and requirements, and the manner in which such applications are pursued given the changes in the HPC industry.

This introduction provides a brief review of the framework developed in *Building on the Basics*. Chapter 2 analyzes key trends in the HPC industry from the perspective of those elements of significance to the export control regime. Chapter 3 provides an expanded analysis of the concept of the lower bound of controllability and establishes a set of options for s in establishing the lower bound of a range of viable control thresholds. Chapter 4 provides extensive coverage of a broad spectrum of national security applications to give s insight into the upper bound for a control threshold. Chapter 5 integrates chapters 3 and 4 into a concrete set of policy options and implications.

The Basic Premises behind Export Control Thresholds

The HPC export control policy has been successful in part because it has been based on three premises that were largely true for the duration of the Cold War:

(1) There are problems of great national security importance that require high-performance computing for their solution, and these problems cannot be solved, or can only be solved in severely degraded forms, without such computing assets.

(2) There are countries of national security concern that have both the scientific and military wherewithal to pursue these or similar applications.

(3) There are features of high-performance computers that permit effective forms of control.

If the first two premises do not hold, there is no justification for the policy. If the third premise does not hold, an effective export control policy cannot be implemented, regardless of its desirability.

While a strong case can be made that all three premises held during the Cold War, there have been significant changes that impact this policy. In particular, following the dissolution of the Soviet Union, threats to national security have become smaller, but more numerous; there have been dramatic advances in computing technologies; and the use of HPC within the U.S. national security community has expanded.

If the premises are still valid, it should be possible to derive a control threshold in a way that is explicit, justifiable, and repeatable. If the premises are not valid, then the analysis should clearly illustrate why no effective control policy based on the premises is possible.

Deriving a Control Threshold

The first premise postulates that there exist applications with high minimum computational requirements. In other words, if the minimum computational resources (especially, but not exclusively, performance) are not available, the application cannot be performed satisfactorily. To establish the performance requirements, we asked applications practitioners to identify the computer configuration that they would need to carry out the application. The composite theoretical performance (CTP) of such a configuration was used to quantify the Mtops used for this application.[1]

In some cases, the configuration used was more powerful than was necessary to do the application satisfactorily. Figure 1–1 shows the performance of the minimum acceptable configuration and the configuration actually used for the F-22 aircraft design application.

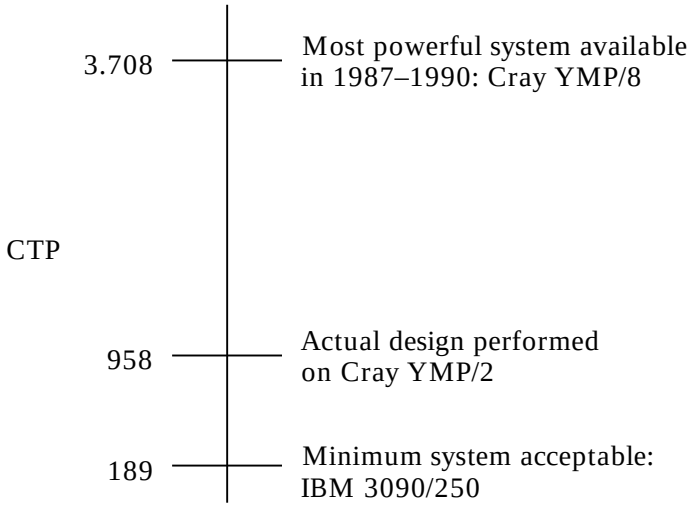

Figure 1–1. Minimum, actual, and maximum computing power available for F-22 design

The third premise requires that systems above the minimum performance level of a particular application have characteristics that permit their export and use to be closely monitored, controlled, and when necessary, denied. If there exist systems that cannot be controlled whose performance exceeds the minimum necessary to carry out the application, then the U.S.

[1] The composite theoretical performance is measured in millions of theoretical operations per second (Mtops). Mtops ratings consider both floating-point and non-floating-point operations, and account for variations in word length, numbers of processors, and whether the system is based on a shared memory or distributed memory paradigm.

Government will be unable to control that application solely by trying to deny the necessary computer power.

In order for the control regime to be justifiable, there must be some applications that satisfy both the first and third premises.

Over time, the computational performance of the most powerful uncontrollable system(s) rises. As it rises, it overtakes the minimum computing requirements of individual applications. If the minimum and actual performance levels for particular applications are plotted over time, the dynamic may be illustrated as shown in Figure 1–2. For illustration purposes, this figure uses only hypothetical data.

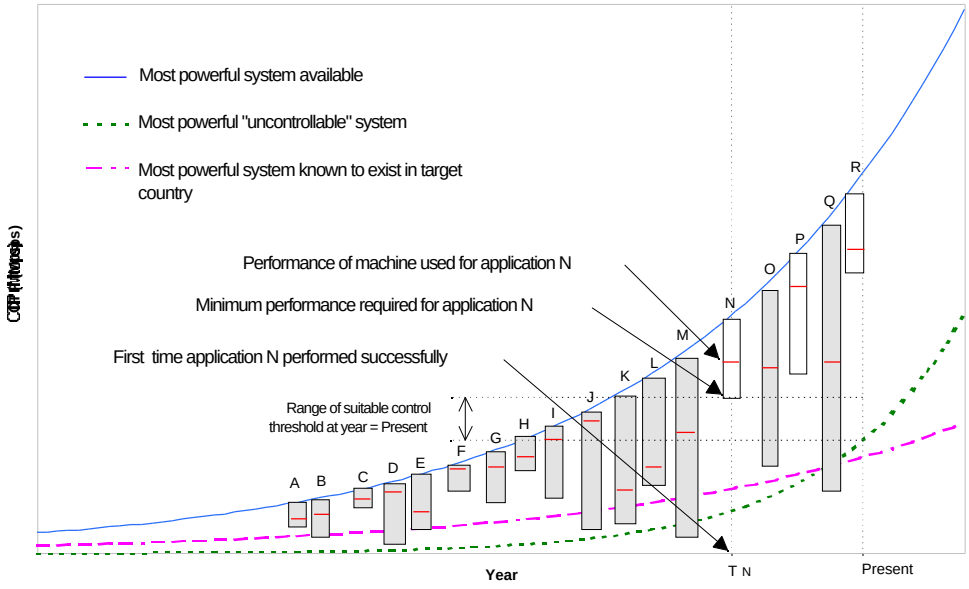

Figure 1–2. Establishing a range for a viable control threshold

Under current practice, a control threshold is established at a particular point in time and remains in effect until revised through policy changes. The set of viable thresholds—those that satisfy the three basic premises—must at the same time lie between two bounds: The "lower bound" is determined by the level of the most powerful uncontrollable systems. The "upper bound" is determined by those national security applications whose minimum performance requirements lie above the lower bound. In Figure 1–2, applications *N*, *P*, and *R* are those that can be protected by export control of HPC hardware.

The selection of a specific control threshold further takes into account the nature of the computer market for systems whose performance, measured in Mtops, lies within the range between the lower and upper bounds. Ideally, a control threshold would be established below a point where there were numerous or particularly significant applications of national security concern, but above the performance level of systems enjoying large markets.

The following chapters supply data for the model. Chapter 2 discusses industry trends that impact the export control policy. Chapter 3 discusses the determination of a lower bound. Chapter 4 discusses the computational requirements of a substantial number of applications of national security concern. Chapter 5 integrates the results of the previous chapters into a

discussion of policy options and implications.

References

[1] Goodman, S. E., P. Wolcott, and G. Burkhart, *Building on the Basics: An Examination of High-Performance Computing Export Control Policy in the 1990s,* Center for International Security and Arms Control, Stanford University, Stanford, CA, 1995.

[2] Goodman, S., P. Wolcott, and G. Burkhart, *Executive Briefing: An Examination of High-Performance Computing Export Control Policy in the 1990s,* IEEE Computer Society Press, Los Alamitos, 1996.

Chapter 2: Industry Trends Affecting Export Control

Overview of Key Industry Trends

What will be the impact of industry trends on the export control regime? To understand which factors are most significant, we must understand what kinds of changes have the greatest impact on the analysis framework used to determine viable control thresholds. The framework seeks to identify performance levels that are above uncontrollable performance levels and below the performance required for key applications of national security concern. Consequently, the greatest impact on the selection of a viable control threshold comes from factors that impact:

(1) the lower bound of controllability, and

(2) the application of high-performance computing technologies to problems of national security concern.

Of particular interest are trends that make it possible to pursue important applications using hardware and software platforms that are considered uncontrollable.

In this chapter, we examine the factors that most strongly impact these two aspects of the framework. Other factors will be covered in greater depth in subsequent chapters. The factors discussed here are:

(1) *Performance.* A primary factor driving the threshold of controllability upward over time is the improved performance of products occupying a given market/price niche. Since the composite theoretical performance is a processor-oriented metric, we focus our attention on developments in processors. However, realizable performance of systems is also a function of the speed of memory, interconnects, I/O, and systems software. We briefly touch on these factors as well.

(2) *Scalability and the ability to harness the computing power of multiple, smaller systems.* As we discuss at greater length in the next chapter, the ability to increase the performance of a system incrementally, or to combine the resources of multiple, independent systems into a computing cluster, has a significant impact on the performance that can be attained through straightforward combinations of uncontrollable platforms. We identify industry advances in interconnects and systems software that make it easier to combine systems or to extend the performance of existing systems.

Recent Trends in Processor Performance and Capability

There can be no doubt that advances in microprocessor technologies are the dominant forces shaping the export control regime today. This is true for at least three reasons. First, the performance of microprocessors is increasing rapidly. All other factors being equal, the threshold of controllability rides this curve, since all uncontrollable platforms today are based on microprocessor technology. Second, microprocessors and other technologies traditionally associated with workstation markets are today being used by all HPC vendors from the low-end desktop uniprocessors to the most powerful massively parallel systems. Third, microprocessors are the most straightforward and uncontroversial contributors to CTP.

Figure 2–1 illustrates the dramatic improvements in microprocessor performance since 1992. All of the data represent microprocessors in volume production; systems based on the microprocessors are generally available from vendors. The chart indicates that by 1997, nearly all of the major RISC/UNIX workstation vendors were incorporating processors above 500 Mtops into their products. The figures for 1999 are estimates, based on publicly available data. Since vendors often introduce microprocessors at clock speeds slightly different from what had been announced, actual figures may vary slightly. Intel processors in volume production will lie somewhere in the range indicated by the two Intel arrows.

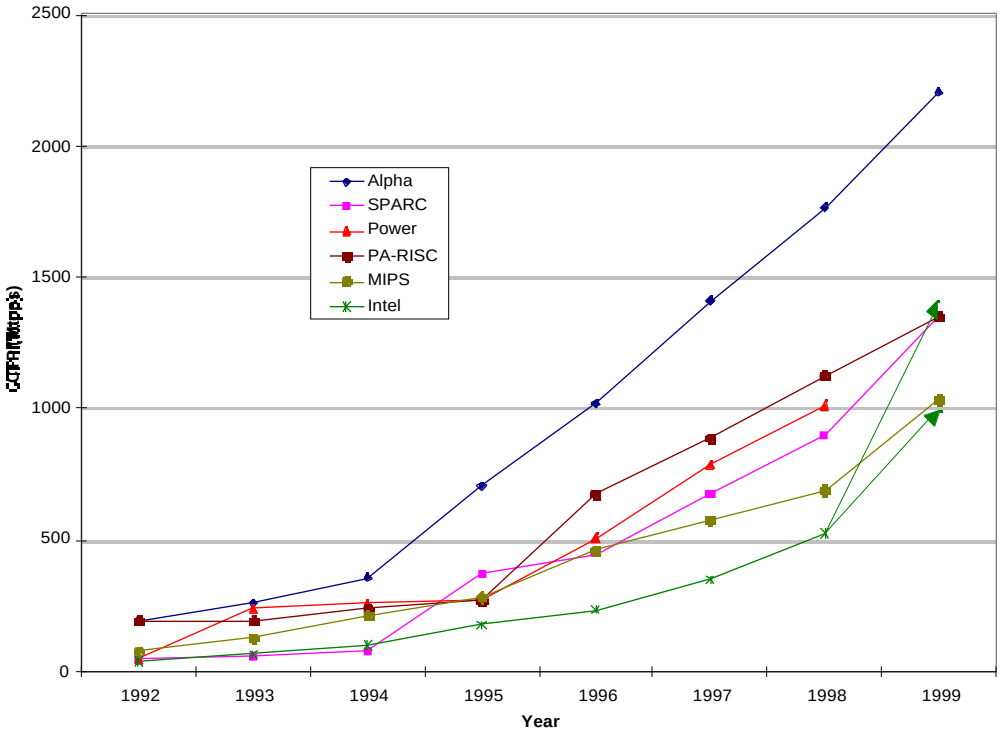

Figure 2–1. Performance of microprocessors in volume production

The improvements in microprocessor performance have been driven primarily by advances in process technologies and in microprocessor architectures. Advances in process technologies have led to smaller feature sizes and greater numbers of transistors per microprocessor. These developments have permitted more functional units, control logic, and cache to be placed in a given amount of space. Carefully designed architectures have permitted the clock rates to increase. Since the number (and type) of instructions issued and

the clock rate are two of the most significant parameters in the calculation of the CTP, these factors have been at the heart of the performance increase. Figure 2–2 and Figure 2–3 illustrate how feature sizes have shrunk and transistor counts have grown since 1992.

	1994	***1995***	***1996***	***1997***	***1998***	***1999***	***2000***	***2001***
Power	0.50	0.50	0.29	0.25	0.25	0.18	0.18	0.18
MIPS	0.50	0.50	0.35	0.35	0.25	0.18	0.18	0.18
Intel	0.60	0.35	0.35	0.35	0.25	0.18	0.18	
PA-RISC	0.80	0.55	0.50	0.50	0.25	0.25	0.18	0.18
Alpha	0.50	0.50	0.35	0.35	0.25	0.25	0.18	0.18
SPARC	0.70	0.50	0.42	0.25	0.18	0.18	0.13	0.13

Fabrication process generations:

0.50 micron
0.35 micron
0.25 micron
0.18 micron
0.13 micron

1994 ***1995*** ***1996*** ***1997*** ***1998*** ***1999*** ***2000*** 2001

Figure 2–2. Microprocessor feature sizes

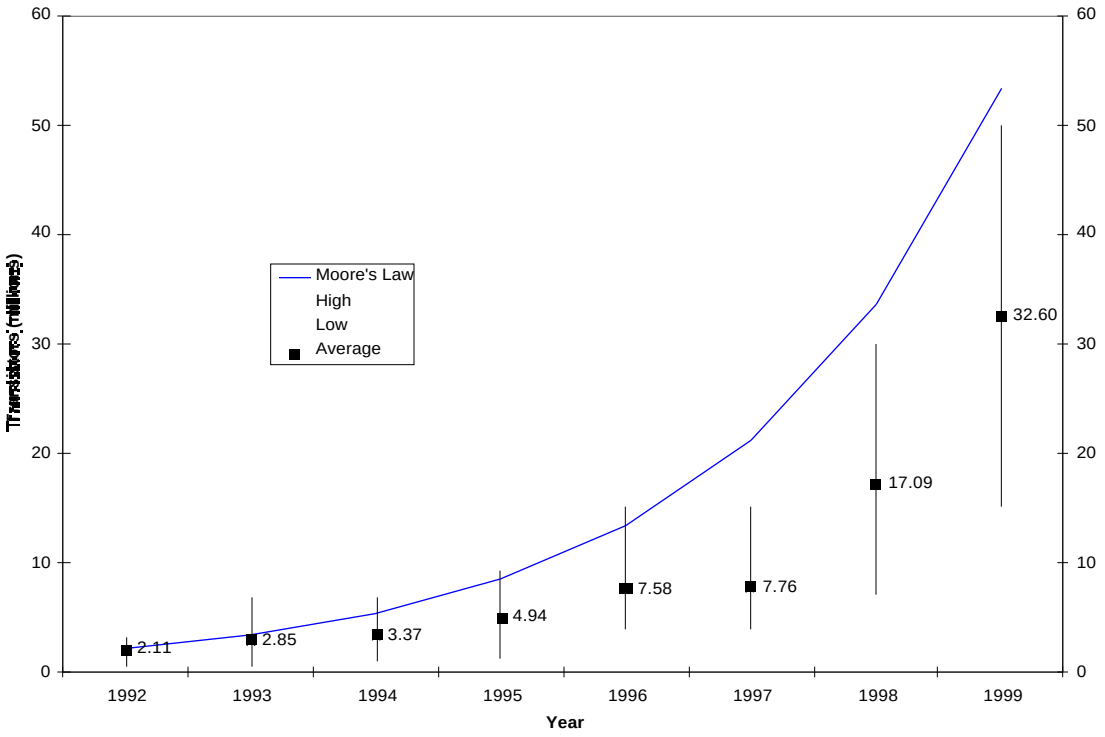

Figure 2–3. High, low, and average transistor counts for RISC microprocessors, compared with Moore's Law

Figure 2–4 and Figure 2–5 show how clock frequencies and instruction issue rates have increased during the same period. The slight drop in the average clock frequency during 1995 reflects the fact that MIPS Technologies introduced in that year the R8000 at 75 MHz, a

processor with more power but a slower clock than its predecessor, the R4400 (250 MHz). The slower clock was necessary because the R8000 processor was a dual-chip processor whose signals needed extra time to propagate from one chip to the other.

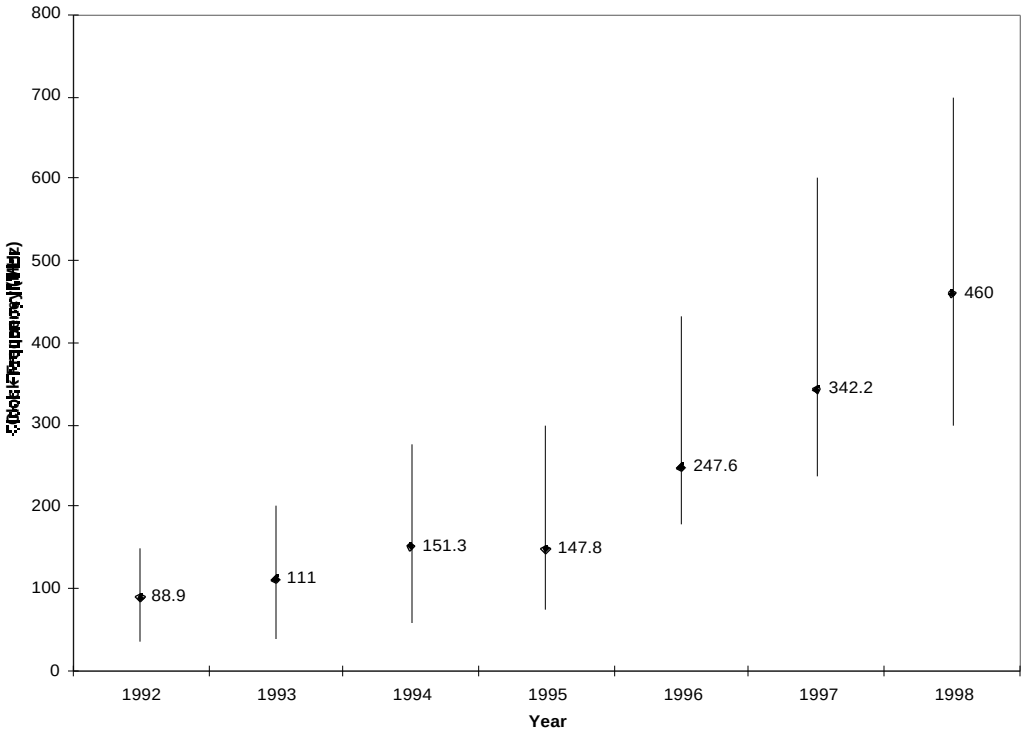

Figure 2–4. High, low, and average clock frequency for RISC microprocessors

Figure 2–5 illustrates that sustained issue rates of four instructions per clock cycle is the standard throughout most of the industry. The jump to higher instruction issue rates by most of the microprocessor vendors, combined with the increases in clock speed enabled by process improvements, accounts for the sharp increase in processor performance in 1994–1996 shown in Figure 2–1. The next sharp increase in issue rates is projected to come in 1999–2000.

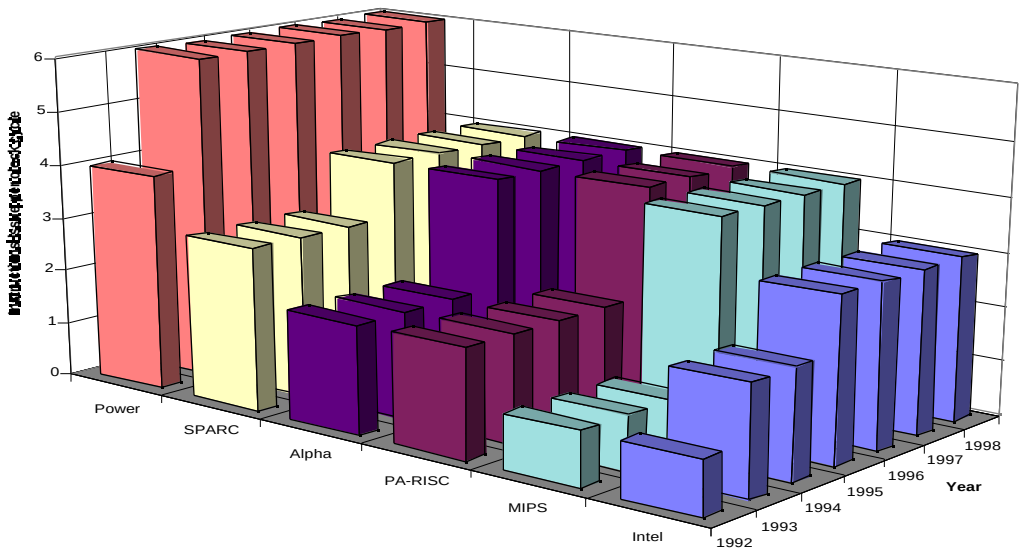

Figure 2–5. Instructions issued per clock cycle

Future Microprocessor Developments and Challenges

Over the next three–four years, microprocessor performance measured in CTP is likely to grow dramatically. Figure 2–6 shows some industry projections—representing no single vendor—for processor performance over the next several years.

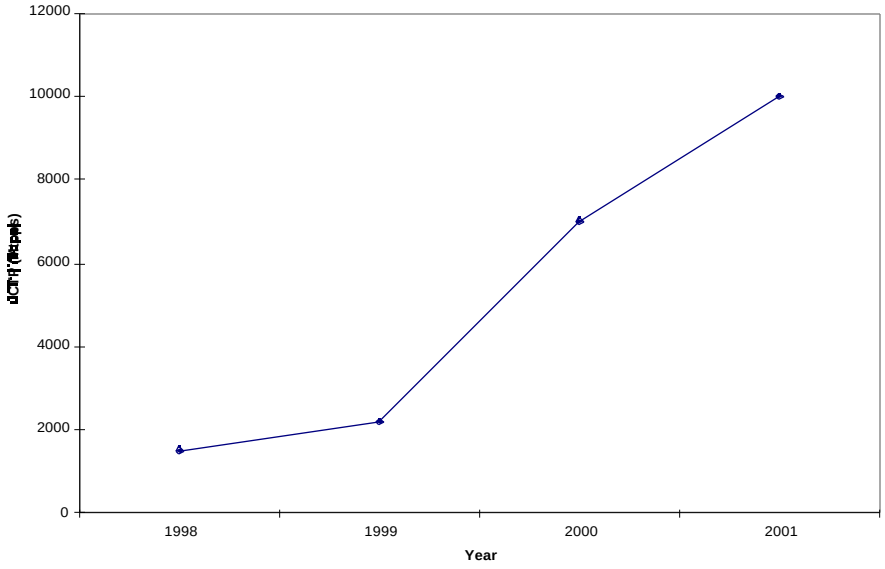

Figure 2–6. Industry projections of microprocessor performance

The growth in performance is fueled by a combination of factors, including

- advances in process technologies leading to more transistors and greater clock speed, and
- changes to the number and nature of functional units as vendors design chips to address the needs of their most significant markets.

Process Technologies

Moore's Law states that the number of transistors on a chip will double approximately every 18 months. The data in Figure 2–3 show that the average number of transistors used in processors has not increased this rapidly during the last few years. However, if industry predictions are correct, transistor counts will "catch up" in the near future, reaching on the order of 50 million transistors by 1999 [1]. The road maps for photolithography indicate that current trends will continue until at least 2003–2004. Vendors are now counting on feature sizes of 0.18 microns by the early part of the 21st century. Current technological approaches may be able to reduce feature size to one more generation, or 0.13 microns [2]. Beyond this point, feature size will be less than the wavelength of the ultraviolet light (UV) used in lithography today, and the industry will have to move to light with smaller wavelengths, such as X ray. It is still unclear how smooth the transition to post–UV lithography will be. Historically, the semiconductor industry has shown a remarkable ability to continue a breathtaking rate of technological advance over decades. Eventually the fundamental laws of physics will halt the trends toward smaller feature sizes, but it is not at all clear that this will be within the next ten years.

As feature size shrinks and gate delays decrease, reducing the cycle time has been a popular and relatively easy way of increasing performance. Figure 2–3 shows that between 1992 and 1997 the average clock frequency increased by nearly 400 percent, substantially more than the 160 percent improvement many microprocessor designers had predicted over these five years [3]. Unfortunately, as feature size drops below 0.25 microns and clock frequencies increase, signals are no longer able to propagate across the entire die. In addition, the signal delays within individual wires increase as the wires become thinner. In [2], Doug Matzke calculates that in a microprocessor with 0.1 micron feature size and a 1.2 GHz clock, only 16 percent of the die is reachable within a clock period. Industry is recognizing these limitations. The Semiconductor Industry Association (SIA) projects that in 2012, clock rates will be about 2.7 GHz, nearly an order of magnitude less than if the trends of the last five years were to continue for the next fifteen [4].

Microprocessor Architectures

If we suppose that trends in shrinking feature size continue, microprocessors manufactured using 0.18 micron technology are likely to have up to 50 million transistors in 1999–2000, and a hundred million or more in 2001–2004. The SIA has predicted that by 2007 the number of transistors will reach 350 million, and 800 million by 2010 [5]. A more recent SIA road map indicates that microprocessors will have 1.4 billion transistors by 2012, provided a number of fundamental fabrication problems are solved [4]. A crucial question for microprocessor architects is how to best use this abundance. For the export control regime, an important question is what impact the new architectures are likely to have on performance.

A number of leading researchers are considering how a billion transistors would be used [6–11]. Although there is considerable variety (and lack of consensus) in the specific techniques being taken to solve this problem, there are three principal strategies employed [8]:

- place a multiprocessor on a single chip;
- integrate not only the processor but also memory and parts of the I/O subsystem on a single chip;

- build a large uniprocessor employing instruction issue rates up to 16 or 32 instructions per cycle.

A single-chip multiprocessor is based on two main observations: (a) It is becoming increasingly difficult to extract parallelism from a single thread of execution. To do so requires a disproportionate amount of chip real estate devoted to instruction scheduling, branch prediction, and so forth. (b) The problems of wire delays on chips with features under 0.25 microns will force designers to create partitioned chips with highly localized functionality within each partition. The benefits of a single-chip multiprocessor are that it is inherently partitioned and is fundamentally designed to accommodate multiple threads of execution, greatly increasing the amount of parallelism that can be exploited. In a system proposed in [10], a single chip consists of eight CPUs which share 8 Mbytes of secondary cache. Each CPU can issue two instructions per cycle. If a single-chip multiprocessor with these characteristics were implemented with a 2 GHz clock, it would have approximately 16 times the performance (4x instruction issue and 4x clock) of a typical 1997 microprocessor. Depending on the year in which it is manufactured, the processor would represent a doubling of processor performance every two or three years.

A second approach is typified by the Berkeley V-IRAM project [9]. In this design, most of the on-chip memory is occupied not by static RAM (SRAM) caches whose contents are understood to be copies of instructions and data stored in main memory, but by dynamic RAM (DRAM) which is treated as main memory. In one proposed layout, DRAM would occupy 99 percent of the transistors (800 million), while the CPU and caches would occupy just 3 million and the vector unit would occupy 4 million. The vector unit would have two arithmetic units, each capable of producing eight results per clock. With the 1 GHz clock proposed by the developers, this processor would likely have a CTP approximately eight times today's microprocessors, or on the order of 6000 Mtops. An implementation would require 0.13 micron technology, which could be feasible in 2002–2004. The performance gain of such a processor would represent, on the average, a doubling of microprocessor performance every two years from 1998 through 2004.

The third approach employs a large uniprocessor using 24 to 48 highly optimized, pipelined functional units. The goal is to place on a single chip those elements that must be close together in order to minimize latency, and to minimize data traffic to off-chip locations. The functional units, instruction issue logic, and extensive cache memory are placed on-chip, while main memory, I/O, and other processors are located off-chip. A key challenge under such circumstances is enabling the processor to issue and usefully execute 16 to 32 instructions each clock cycle without overwhelming the off-chip memory bandwidth. In a design proposed by Patt et al. [7], the challenges are addressed by devoting nearly 90 percent of the transistors to various kinds of caches, with only about 10 percent to logic (6 percent for the execution core itself). By comparison, logic occupies nearly 40 percent of the transistors in the MIPS R10000 and just over 50 percent in the Alpha 21164. Assuming a 2 GHz clock in 2010, a large uniprocessor is likely to have 16 to 32 times the (theoretical) performance of the typical 700 Mtops processor of 1997, or 11K to 22K Mtops. This represents a doubling of performance every three years.

In each of the approaches discussed above, increasing the number of instructions issued and executed per clock period and shortening the clock period increases theoretical performance. The principal differences lie in areas that have no great impact on the CTP as currently formulated. While the CTP is a function of the number and types of computational elements, the clock period, and the word length, the approaches outlined above differ from each other mostly in the techniques used to keep the functional units occupied and to contain design complexity, programming complexity, or both. Instruction issue logic and caches play a critical role in delivering real performance, but do not contribute to the performance measured by the CTP.

Will the delivered performance of tomorrow's microprocessors grow faster, slower, or at the same rate as the CTP? The question cannot, of course, be settled until chips are manufactured and tested. Chip designers will certainly devote a disproportionate amount of chip real estate to memory, cache, and administrative functions. If designers are very successful in these efforts, better use may be made of each execution unit than has been the case in the past, and the deliverable performance will increase faster than the CTP. In other words, while the CTP may grow by a factor of 16 to 32, the observable performance on real applications may grow more quickly. It is not, however, a foregone conclusion that such efforts will be successful.

Chip designers face enormous challenges in keeping functional units occupied. The current state of the art has difficulty keeping four, let alone six, functional units busy. Microprocessors such as the DEC 21264, the MIPS R10000, and the PowerPC 604 typically achieve only about 0.5 to 1.5 sustained instructions per cycle (out of a potential of four per cycle) on real-world problems [12]. Part of the problem is that functional units spend most of their time waiting for instructions or data. The other part of the problem is that functional units today are pipelined, meaning that at any point in time there may be from five to twelve instructions at various stages of execution. The processor must fetch the next instruction before the current one has finished executing. In portions of code involving conditional statements (IF-THEN) the "next" instruction is dependent on the result of a current calculation. The processor must guess which instruction will be next; if it guesses wrong, the pipeline will stall until the correct instructions can be fetched. This penalty may be a wait of several cycles, dramatically reducing utilization. If the potential number of instruction issues per clock period increases to six or sixteen, the task of keeping all, or even a comparable fraction, of the functional units busy increases substantially.

At the same time, increasing clock frequencies makes the penalty for missing a needed instruction or data element greater. For example, if the CPU clock runs at 400 MHz, a 2.5 nanosecond (nsec) clock period, but main memory access time is 60 nsec, then the time a functional unit must wait for a data element is 24 CPU cycles. The cost in utilization of that functional unit is enormous. The difficulty lies in the fact that the gap between the speed of the CPU and the speed of main memory is widening, not shrinking or even holding constant. At the same time, the presence of growing numbers of functional units will require the volume of data flowing across chip pins to grow correspondingly. The challenges are so formidable that Richard Sites, a senior architect at Digital Equipment Corporation, has stated, "over the coming decade memory subsystem design will be the only important design issue for microprocessors" [3].

Impact of Emerging Applications

The cost of microprocessor fabrication is enormous, and increasing. In 1993, a 0.8 micron facility cost approximately $200–300 million to build and mature [13]. Today, new fabrication plants, which must be built for new generations of process technology, cost $1–2 billion each [14,15]. In five or ten years, the cost could rise to $4 billion [4]. This cost must be recovered in a very few years, before the next fabrication plant is built. The only way to recover these costs is to sell hundreds of thousands of high-priced processors or millions of low-priced ones. The only way to sell large volumes of microprocessors is to design them to serve the mass markets of desktop, embedded, or entertainment systems.

What applications will drive microprocessor design, and what are the implications of the new designs for CTP performance? There is broad agreement that multimedia and user interface workloads will consume a growing proportion of a microprocessor's processing power. Multimedia-intensive applications include animation, videoconferencing, visualization, image processing, realistic simulation, and so forth. Nearly all CPUs introduced in recent years contain features for improved multimedia performance. Table 2–1 shows measures taken by various vendors to support multimedia workloads.

Processor family	*Multimedia capability*
Intel	Multimedia Extension (MMX)
Alpha	Motion-video instructions (MVI)
MIPS	MIPS digital media extensions (MDMX)
SPARC	Visual Instruction Set (VIS)
PA-RISC	MAX2

Table 2–1. Multimedia support in major microprocessor families

Multimedia applications have a number of characteristics that distinguish them from traditional workloads. These include [16]:

- real-time processing
- heavy use of 8- or 16-bit values rather than 64-bit
- continuous, or streaming input data
- fine-grained parallelism as each element in an input data stream undergoes the same transformations
- coarse-grain parallelism among audio, video, and background tasks
- good spatial and temporal locality in instructions in the pieces of code that consume most of the cycles (tight loops)
- high memory bandwidth requirements
- high network bandwidth requirements

The approach taken in one fashion or another by most microprocessor vendors to date has been the addition of specialized multimedia instructions to the existing instruction set. A common technique (MIPS, Intel, SPARC) is to use existing floating-point registers to process not just one (64-bit) data element at a time, but multiple 8- or 16-bit data elements that have been packed into a 64-bit word. The techniques used on the Alpha and PA-RISC chips utilize integer registers for the same purpose. The extended instructions perform a variety of tasks, including packing and unpacking pixels into words, performing arithmetic and logic operations on pixels, and so forth [17–19].

So far, the impact of multimedia extensions on CTP has been minimal. The extended instruction sets have not fundamentally changed the computational elements in a chip; they have only changed the use to which these elements have been put. If chip designers decide to create graphics units in addition to the set of more traditional fixed- and floating-point units, the CTP could correspondingly increase. By applying the same instructions on multiple pixels or audio samples, these units can look a great deal like small vector units. The addition of such units could push single microprocessor CTP levels over 10,000 Mtops by 2002.

Significance of Microprocessor Developments for HPC

The significance of microprocessor developments for high-performance computing systems is very easily stated. By far the majority of high-performance computing systems today employ commercially available microprocessors as their compute engines. The percentage of the world's 500 most powerful computing installations employing commercially available microprocessors has grown from approximately 10 percent in 1993 to 75 percent in 1997. Figure 2–7 illustrates this trend clearly.

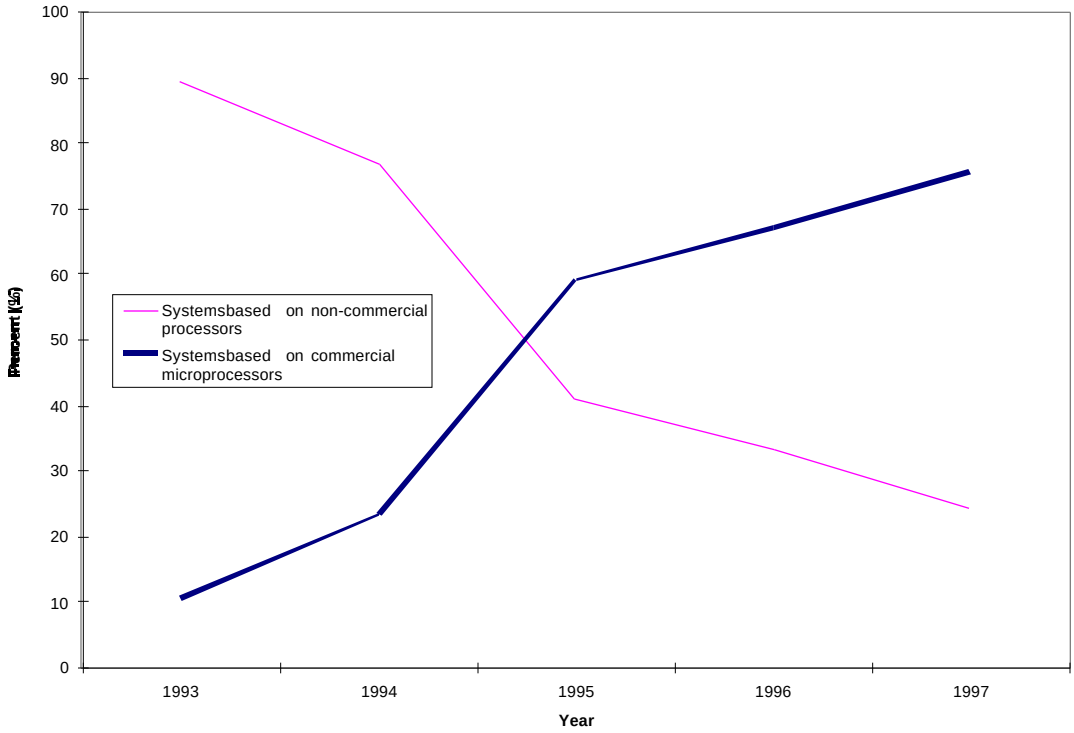

Figure 2–7. Percentage of Top500 systems based on commercial microprocessors/proprietary processors. Source: [Dongarra et al., Top500 Lists]

Most high-performance computing systems today are built using commercial microprocessor technologies largely for economic reasons. The price/performance of commercial microprocessor-based systems is compelling for large segments of the HPC market. For example, Figure 2–8 shows the price per Mtops of models introduced during 1997 by various vendors. The prices are based on advertised vendor list prices at the time of introduction. (The C90, introduced in 1991, is provided for additional comparison.) The Cray J90 does not use commercial microprocessors, but does use CMOS components manufactured using processes comparable to those used by the leading microprocessor vendors.

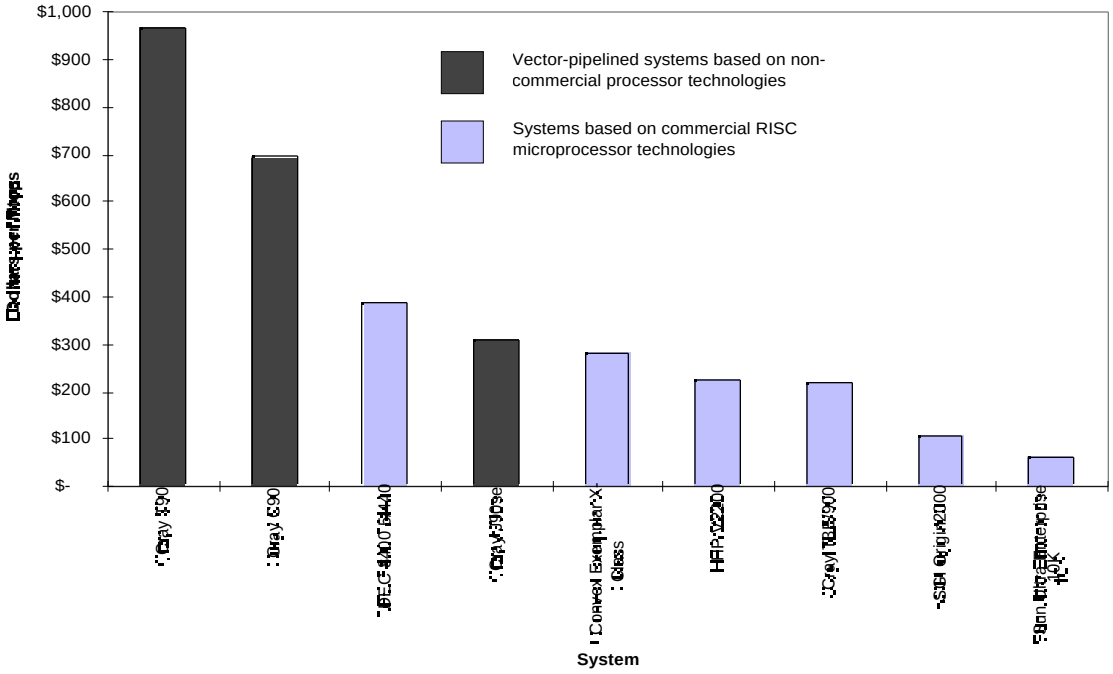

Figure 2–8. Price/performance of HPC systems

The figure illustrates the price/performance advantage that systems based on CMOS technologies in general, and commercial CMOS microprocessors in particular, enjoy over traditional vector-pipelined systems that do not employ these technologies.

Gluing the Pieces Together: Recent Trends in Systems Architecture and Capability

If systems from the low end to the high end of all HPC companies' products lines are being constructed from the same underlying microprocessor components, then computational performance as measured by the CTP will ride the rising tide of microprocessor performance. But what industry trends are shaping vendor and practitioner abilities to harness the power of aggregations of microprocessors and apply it in a concentrated fashion to real-world applications? The advances that provide the "glue" include:

(1) microprocessor design features to support multiprocessing

(2) interconnect and memory subsystem improvements

(3) systems software advances

Microprocessor Support of Multiprocessing

While providing greater uniprocessor performance has been a constant goal of microprocessor designers, the need to build into the microprocessor itself elements that will facilitate the creation of multiprocessor systems is of growing importance.

A leading example of this trend is Sun Microsystems' UltraSPARC-III, introduced in 1997 and scheduled for sampling in 1998 [20,21]. Designers decided to forgo maximizing performance in favor of features that enable the CPU to function more effectively in a multiprocessor environment. One measure being taken is to reduce a CPU's contention with other CPUs for shared resources. A dominant architecture in the mid-1990s for mid-range HPC systems has been the symmetrical multiprocessor (SMP) architecture in which all CPUs have shared and uniform access to all of main memory. Local, cache memory is considered just a copy of main memory. While providing a very convenient programming and operating model, SMP systems have often suffered performance degradation in large configurations due to contention among the CPUs for memory and memory access busses. In the UltraSPARC-III, each CPU has its own main memory, which limits CPU contention to times when one CPU must access the memory of another. The logic not only for cache access but also for main memory access is migrated onto the microprocessor itself. A second set of multiprocessor-oriented design features of the UltraSPARC-III tries to minimize the impact of delays in accessing off-chip memory. These features include extensive speculative execution and non-stalling execution pipelines.

Interconnect and Memory Subsystems

Supplying the CPUs with an adequate volume of data and instructions in a timely fashion is fundamental to system performance. In an ideal world, all memory would be accessible quickly, available in large volumes, and very cheap. The reality is that these three quantities cannot all be optimized simultaneously. The fastest memory, used in on-chip registers and caches, is available in small quantities at high cost. The cheapest memory, available as disk or tape storage, is available in large volumes, but is very slow. Second-level cache and main memory have intermediate qualities. Providing large memory spaces with high bandwidth and low latency is extremely expensive and one of the reasons traditional supercomputers have million-dollar-plus prices. Because of the price, performance, and volume characteristics of different kinds of memory, computing systems at all performance levels have adopted elaborate hierarchical memory structures that integrate different kinds of memories in varying volumes. Figure 2–9 illustrates the types of memory and the types of interconnect used to move data from one to the other.

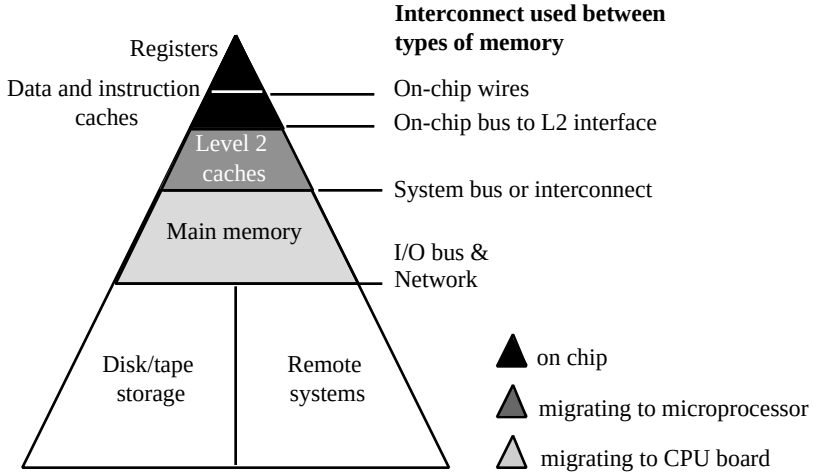

Figure 2–9. Memory hierarchy and interconnects

Since the execution units of microprocessors are designed to operate only on data located in the registers, managing memory involves a complex process of moving data elements and

instructions up and down this hierarchy so that the right data elements and instructions are in the registers at the right time. A system's ability to carry out this process effectively is vitally dependent on the memory management logic and the bandwidth and latency of the memory and interconnect subsystems.

Designing the memory system is one of the great challenges of system architecture today. As microprocessor clock frequencies and the number of instructions issued per cycle increase, the system becomes increasingly sensitive to failures of the memory system to deliver the right data element to the functional units at the right time. In the event of such a failure, the CPU may have to wait until the appropriate data element or instruction is fetched from wherever it might be in the memory hierarchy and placed in the registers. In the past, accessing data in main memory required a single clock cycle. This is no longer true. Table 2–2 shows the typical latency, measured in clock cycles, to fetch a data element or instruction from various locations in memory. Due to the penalty incurred by a memory access "miss," designers have made enormous efforts to migrate the data and instructions most likely to be needed in the near future closer to the CPU.

Type of memory	*Location*	*Access time (cycles)*
Registers	on-chip	1
Data and instruction cache	on-chip	~1-2
Level 2 cache	off-chip	~6-10
Local main memory	on-board	~60-100
Remote main memory	off-board	~100-200+
Secondary storage, networked systems	on-disk, on-network	~1000s-10000s+

Table 2–2. Memory latency

Under the current CTP formulation, memory subsystem characteristics affect Mtops performance in only two ways. First, in systems with logically shared main memory the contribution of multiple CPUs is weighted more heavily than in systems with logically distributed main memory. Second, CPUs linked together by traditional networking technologies do not contribute to the CTP of the system. Consequently, most of the developments in memory and interconnect systems are likely to have only minimal impact on the CTP, even though they may have substantial impact on real-world performance. From the perspective of the CTP, the most significant memory and interconnect advances are those that will enable larger numbers of processing elements to be combined effectively into single systems. Of principal interest are:

(1) advances in system busses that accommodate greater numbers of CPUs

(2) the trend away from shared system buses to switched interconnects

(3) the growing use of networking or clustering interconnects with performance substantially better than traditional networks

In 1995, Intel publicized widely that its forthcoming Pentium Pro (P6) processor was to be designed for parallel processing. One of the principal advances was the creation of a new 64-bit system bus optimized for high-bandwidth access to main memory by up to four CPUs [22]. This form of on-board multiprocessing has become quite widespread, enabling many original equipment manufacturers (OEM) such as Compaq and Dell to sell multiprocessors without having to invest heavily in multiprocessing technologies of their own.

Intel, however, was a latecomer to the world of symmetrical multiprocessing, especially multi-board multiprocessing. Since the late 1980s and early 1990s vendors such as Silicon

Graphics, Sun Microsystems, and Digital Equipment Corporation have been marketing products employing multiple processors accessing memory through a single, shared bus. To accommodate processor counts as high as 36 (in the case of Silicon Graphics Challenge servers), these vendors used busses that were hundreds of bits wide and had high bandwidth and modest latency.

Because the bus is a shared medium, however, the greater the number of processors, the less bandwidth is available, on average, for individual processors and the greater the likelihood of CPUs encountering contention for the system bus. For this reason, systems could not scale beyond a modest number of processors. In practice, systems with more than 8–12 processors suffered substantial performance degradation.

HPC systems designed to accommodate large numbers of processors were designed from the start to avoid the bottlenecks of shared interconnects. Cray's T3D and T3E, IBM's SP2, Intel's Paragon, and others employed switched or crossbar interconnects that would scale with the number of processors so that the interconnect bandwidth available per processor remained relatively constant across configuration sizes. These systems, however, employed distributed rather than shared memory models to avoid memory bottlenecks, even at the logical level.

To ease the bottlenecks of a shared bus and make the interconnects scalable to larger configurations, several vendors have begun, or will soon begin, implementing switched or crossbar interconnects even in shared-memory systems. Silicon Graphics, Inc., is particularly noteworthy in this respect. Until 1996, SGI's multiprocessors relied on a wide, low-latency bus shared by all processors. With the introduction of the Origin family, SGI abandoned this approach and began using a crossbar interconnect that can accommodate up to 64 processors (128 with a specialized "metarouter") and feed 780 Mbytes/sec (peak) to each pair of processors. This bandwidth, available to each pair of processors, exceeds the total bandwidth of the bus used in SGI's PowerChallenge and Challenge families.

Bandwidth and latency remain two of the principal qualities by which interconnects are evaluated. Figure 2–10 shows a number of different interconnects used today plotted on a log-log scale. The bandwidth is calculated on a per-processor basis. Latency reflects the number of clock cycles needed for a data element to be fetched or sent from memory across the interconnect. The figures shown here reflect hardware latencies. Number of clock cycles is a more useful parameter than elapsed time, because the processing penalty incurred is directly related to the number of cycles a processor must wait for a needed data element. Thus, a 10 microsecond wait for a 100 MHz processor is much less serious than for a 500 MHz processor.

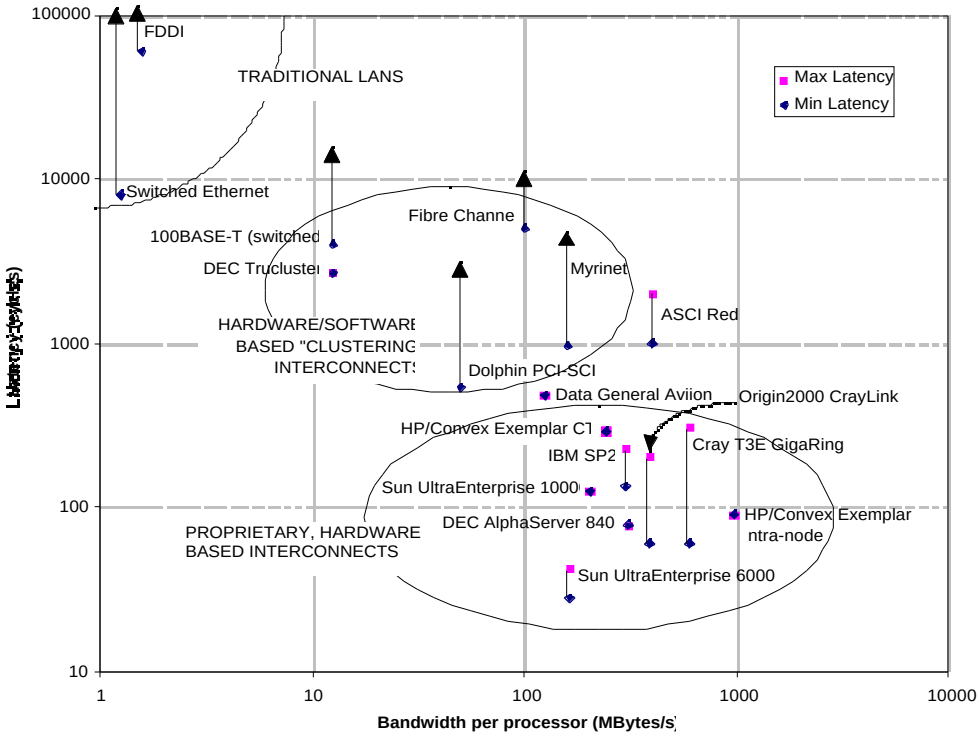

Figure 2–10. Interconnect latency and bandwidth

The chart shows that interconnects fall into three categories whose latency and bandwidth may differ from one to the next by an order of magnitude or more. The poorest performing interconnects are the traditional local area and wide area networks. Because they are designed to operate cheaply, span several hundred meters or even kilometers, and provide reliable data transport over unreliable media, these networks have extremely high protocol and operating system overhead. Data exchange may consume tens or hundreds of thousands of CPU cycles. Because the media are often shared, the per-processor bandwidth is very low.

At the other extreme are proprietary, in-box interconnects designed for the highest bandwidth and lowest latency feasible. Designed to function over very short distances in highly controlled environments, these interconnects exhibit very low overhead. Data exchange is managed entirely by the hardware. Designing and building such interconnects is very difficult, costly, and requires highly specialized expertise and engineering ability.

Between these two extremes is a growing class of modest latency, moderate-to-high bandwidth interconnects. These hold a great deal of promise for aggregating multiple conventional workstations or servers into unified “clusters” whose aggregate performance can be applied to computational problems with much more efficiency than networked systems of the past. This class of interconnects, which emerged only in the middle of the 1990s, offers an excellent means of creating systems with modest levels of parallelism at modest cost using off-the-shelf technologies. While they have substantially less overhead than networks, using the interconnects requires a combination of hardware and software mechanisms that so far have prevented their achieving latencies as low as the proprietary interconnects. In fact, the gap between the clustering interconnects and the proprietary interconnects is somewhat greater than what is shown here. The operating system and network protocols can increase the latencies by an order of magnitude or more. Furthermore, traditional networking and today’s clustering interconnects have much higher and less predictable variation in latency than do the

proprietary interconnects. The latter are usually designed to provide uniform, or at least predictable, latency with a well-known upper limit.

Two clustering interconnect technologies in particular are noteworthy: Myrinet from Myricom, Inc., and the Scalable Coherent Interconnect embodied in products from a number of vendors. Myrinet [23] is a Gigabit-per-second packet-communication technology that can be used either as a local area network (LAN) or as a "system area network" connecting components of a single configuration, often housed in the same rack. Components are connected to a switch by a single, non-shared link. What distinguishes the network is its low latency and high throughput, 160 Mbytes per port. In most networks, packets are sent across a link as a stream of bits. All bits that are part of a single packet are collected at an intermediate node before any bits are transmitted along to the next node. Myrinet employs "cut-through" routing in which the switch does not wait for an entire packet to arrive before transmitting the leading bits to their destination. The interconnect has round-trip latencies between approximately 10 and 200 microseconds, depending on the packet size and the control software used. At present, Myrinet switches are limited to eight ports. Switches may be linked to switches to expand the number of nodes, but the behavior of such combinations may not scale well for moderate or heavy loads, due in part to the blocking nature of the switch [24].

Another noteworthy development is the Scalable Coherent Interconnect (SCI), developed through a collaboration between individuals at National Semiconductor, Hewlett-Packard, Dolphin, MIPS Technologies, MicroUnity Systems, the University of Wisconsin, and others [25]. SCI is a 1992 ANSI/IEEE standard that describes an interconnect supporting a shared-memory model across memory that is physically distributed [25–27]. It acts like a processor/memory bus, but permits higher levels of parallelism than a shared bus, which can quickly become saturated. SCI is significant from both a networking perspective and a systems architecture perspective (described below). From a networking perspective, SCI enjoys low latencies (a few microseconds) because it keeps data as ordinary variables stored at ordinary memory addresses. In contrast, networking protocols spend a great deal of time moving data from memory locations to buffers which are passed from the I/O system to the operating system to the user and back again. Numerous HPC vendors, including HP/Convex, Cray, Sun Microsystems, Data General, and Sequent, have created, or commissioned, implementations for use in their own systems. Dolphin Interconnect Solutions is a leading vendor of SCI products.

Finally, it should be pointed out that the interconnect used in the SP2 is more closely related to the clustering interconnects than to the proprietary, hardware-oriented interconnects among which it is plotted in Figure 2–10. First, there is no hardware message-passing mechanism, so the hardware latency is unattainable. The minimum latency in practice is likely to be on the order of tens of microseconds [28]. Second, the Power microprocessors have much slower clocks than their competitors, so latency measured in clock cycles is lower in the SP2 than in other systems with faster clocks. At the same time, the SP high-speed switch is able to scale efficiently to much higher node counts than other clustering interconnects.

The classification of interconnects described above has a number of points of significance for the export control regime. First, the decision not to compute an aggregate CTP for systems joined by a local area network is probably justified. While these traditional clusters may be very useful as throughput engines in which different serial processes are farmed out to run on single workstations, the poor performance of the interconnect makes it enormously difficult, if not impossible, to apply them all to a single application. The performance gap between networks and proprietary interconnects is huge.

Second, there are significant differences between interconnects available today that impact real performance; however, these differences are not always reflected in the CTP metric. In other words, eight Alpha workstations on a LAN are not equivalent to eight workstations connected via Myrinet, which are not equivalent to an eight-processor AlphaServer 8400 or a

small configuration Cray T3E. Each system uses the same processor, but aggregations using poorer interconnects should not be viewed as equal substitutes for those using more sophisticated interconnects. Similarly, the presence of clusters in foreign countries should not, by itself, legitimize the export of systems with identical processors but more sophisticated interconnects.

In addition to differences in the performance of the interconnects, there are substantial differences in the software environments of clusters and integrated systems that make the former much more difficult to use in a production environment than the latter. What distinguishes, for example, the SP2 from a cluster of workstations? While the hardware is similar to that used in clusters, most of the value added by the SP lies in software. By far, the bulk of IBM's SP2 and RS/6000 SP development dollars have been spent on software. Creating a system that has the ability to concentrate the available hardware resources efficiently on a single problem across a broad spectrum of applications involves much more than connecting pieces of hardware. A great deal of specialized software is required to harness the CPU cycles efficiently, provide the necessary input/output and file management, make the system easy to monitor and manage, and ensure reliable operation over extended periods of time. In other words, creating a system with a great deal of capacity (total CPU cycles) is easy. Creating a system with high capability and good usability features is difficult, especially across a broad spectrum of applications.

While significant strides are being made toward developing the systems software necessary to effectively harness clusters, at present such efforts are, for the most part, still in the proof-of-concept stage. Where clusters are being used in a production capacity, the range of applications is limited. A recent NASA workshop addressed the opportunities and limitations of so-called Beowulf-class clusters. These are clusters using only mass-market commercial off-the-shelf components, freely available operating systems, and industry-standard software packages. The workshop determined that "Beowulf-class systems can deliver multi-Gflops performance at unprecedented price-performance but that software environments were not fully functional or robust, especially for larger 'dreadnought' scale systems [hundreds of processors]" [24].

Finally, the emerging "clustering interconnects" make parallel processing more accessible and affordable than ever before to practitioners throughout the world. Clusters of workstations using clustering interconnects have been used to run important applications characterized by high computation/communication ratios. They are likely to be a permanent feature on the international HPC landscape.

Trends in Multiprocessor Architectures

In the past, high-performance computing systems have been neatly divided into shared-memory multiprocessors with modest numbers of processors and distributed-memory systems with a potentially large numbers of processors. The former offered programmers a familiar programming model with a single global memory space and uniform access times to all main memory locations. The latter supported much greater numbers of processors, but at the cost of a distributed memory programming model that forced applications programmers to play close attention to, and manage, the physical location of data throughout the system.

Distributed Shared Memory Architectures

One of the major architectural shifts in recent years has been toward a distributed shared memory (DSM) architecture in which memory is physically distributed, but logically shared. Like traditional distributed-memory architectures, DSM architectures accommodate larger numbers of processors than traditional shared-memory architectures. Like traditional shared-memory architectures, however, the DSM architectures support the shared-memory programming model. Because of the physical distribution of memory, it is no longer possible

to guarantee uniform access times to all memory locations. Consequently, such systems are frequently called Non-Uniform Memory Access (NUMA) systems.

In 1991, Kendall Square Research became the first company to offer commercial systems incorporating distributed shared memory. Convex introduced a physically distributed but globally shared memory in its Scalable Parallel Processor (SPP)-1000 in 1994. More recent adopters of this architecture include Data General (NUMAliiNE systems, 1995), Silicon Graphics (Origin2000, 1996), and Sequent Computer Systems (NUMA-Q, 1996).

Underlying each of these systems is a means of moving data among physically distributed memory locations, while maintaining the illusion of a single, shared memory space to the processors. If multiple copies of data elements are to be stored, then those copies must be kept consistent with each other. "Cache coherent" is a term frequently associated with NUMA architectures that refers to this consistency.

Kendall Square pioneered such a technology with its ALLCACHE interconnect, and engineers at Dolphin Interconnect Solutions, Inc., acquired the intellectual property rights to ALLCACHE and further extended these and other ideas. In the years since its introduction, SCI has been implemented by a number of different vendors in a variety of capacities. Data General and Sequent employ SCI to build systems based on Intel microprocessors that scale to 32 processors. Cray Research based its GigaRing on SCI technologies, although the Cray T3E does not support a global shared memory paradigm. Dolphin Interconnect has developed clustering technologies for Sun Microsystems that employ SCI. The SGI Origin2000 is not based on SCI, but on a proprietary cache-coherent NUMA design.

In their current product lines, many HPC vendors have combined architectural features of the shared-memory SMP and distributed memory MPP architectures to create new systems based on physically distributed, logically shared memory systems employing crossbar or switched interconnects that scale well beyond the limits of traditional SMP platforms.

The strong emergence of DSM systems comes just at the time when practitioners using the most powerful systems have gained considerable experience using distributed memory programming models such as that supported by the Message Passing Interface (MPI). The role of MPI as a catalyst in facilitating the migration of codes to distributed memory systems is discussed in Chapter 4.

Hierarchical, Modular HPC

A second major trend in HPC architecture, often coupled with the trend toward DSM, is the grouping together of processors into multiprocessor modules with uniform access to locally shared memory, and combining these modules into larger systems which may employ a distributed- or NUMA- architecture between modules. Rather than build systems as a single tier of processors joined by a single interconnect, the hierarchical, modular design often employs one kind of interconnect to group together 4–16 processors into nodes, and another kind to aggregate the multiprocessor nodes.

The motivation for these kinds of architectures comes from two quarters. One approach is represented by HP/Convex's SPP and Exemplar systems and, more recently, IBM's RS/6000 SP. It is easier and cheaper to provide a given level of interconnect performance for a few processors than for many. At the same time, many applications have good data and instruction locality; that is, the data and instructions needed in the near future are likely to be located close to the instructions and data recently used. Designers can take advantage of these features to build systems in which a limited number of processors have fast and wide access to a portion of memory located nearby, and more limited access to memory further away.

HP/Convex's SPP and Exemplar systems use "hypernodes" of up to 16 processors which use a very high bandwidth and low latency crossbar interconnect to link these processors with shared memory. The Coherent Toroidal Interconnect (CTI) is based on SCI to connect the hypernodes with each other. With the introduction of its "high nodes" for the RS/6000 SP in 1996, IBM introduced a new architecture in which each node consisted not of a single

processor, but of a symmetric multiprocessor (SMP) system with multiple processors joined by a shared bus to shared memory.

The alternative path to hierarchical, modular architectures comes from vendors of traditional SMP systems who wish to aggregate them into larger configurations beyond the limits of a shared bus. Silicon Graphics' PowerChallenge Array epitomized this approach. The PCA consisted of multiple, shared-memory PowerChallenge SMPs connected by a switched interconnect into a distributed-memory cluster. Sun Microsystems is also offering clustering technologies to unite multiple SMP systems. In the lower-end markets, several companies are taking advantage of Intel's creation of so-called "Standard High Volume (SHV) servers," or "quads," which are single-board units consisting of four Pentium Pro processors linked by a 528 Mbytes/sec bus to shared main memory [22]. These four-processor units form basic building blocks which can be combined in a number of ways to create larger shared or distributed memory configurations. Sequent, Data General, and NCR are among the companies actively pursuing this route. Each of these companies views its proprietary interconnect as part of the value it adds to systems development.

Table 2–3 summarizes the developments of various HPC vendors with respect to these trends.

Vendor System	*Year*	*Distributed Shared Memory*	*Hierarchical Modular Systems*
IBM RS/6000 SP	Various	Not supported.	Nodes are SMP systems, currently up to 8-way, increasing to 32-way in the future. Nodes interact via a high-speed switch.
SGI PowerChallenge Array	1995	Not supported.	PowerChallenge XL SMPs clustered together via high-speed interconnect. Distributed memory outside of SMPs; shared memory within.
SGI Origin2000	1996	Employs Scalable Shared Memory Processing (S2MP). Shared memory up to 32 processors (mid-1997) increased to 64 and 128 during late 1997 and 1998.	Smallest module is dual-processor board. Hierarchical modular architecture much less pronounced than in PowerChallenge Array.
HP/Convex Exemplar	1996	Supported by Coherent Toroidal Interconnect (CTI).	Hypernodes consist of 16-way SMP architecture. Multiple hypernodes connected by CTI.
Sequent NUMA-Q	1996	Supports CC-NUMA architecture via IQ-Link, based on SCI.	Uses Intel SHV modules of four Pentium Pro processors linked via IQ-Link.
Data General AViiON 2000	1996	Uses the Synchronous Coherent Memory technology.	Uses Intel SHV modules.
Cray T3E	1996	Uses a distributed memory programming model, although global memory addressing is supported.	Not supported. Uses single-tier architecture. Future generation of T3E to merge into Origin line.
Sun UltraEnterprise	1996	Not currently supported, although future developments likely to incorporate some related ideas.	Clustering solutions join SMPs into a distributed memory, message-passing configuration.
DEC AlphaServer	various	Not currently supported. Galaxies Software Architecture likely to offer some global perspective on memory, although not necessarily a shared memory programming model.	Great emphasis on clustering SMP systems together using Trucluster or other technologies. Galaxies Software Architecture represents next stage of evolution of these technologies.

Table 2–3. Current HPC systems and trends

Major Shifts in HPC Architectures?

What kinds of major shifts in system architecture can we expect over the next three to five years? There is little indication that architectures will change dramatically in the next half-decade. Both SGI/Cray and IBM have introduced new architectures since 1996. For them, developments in the coming years will be largely incremental. SGI/Cray is faced with the great technical challenge of integrating the SGI and Cray Research product families and overcoming current financial difficulties. IBM will expand the size of its nodes, but is unlikely to introduce a dramatically new architecture soon. It will take some time for software developers to learn to take full advantage of the several different kinds of memory available in the recent RS/6000 SP systems.

The leading candidates for new architectures are Sun Microsystems and DEC, each of which has gone for many years without introducing a new architecture. Both companies have released few details about their future plans. That which has been released would seem to indicate plans to "do the same thing, only better" than their competition, rather than break new architectural ground.

Although the underlying architectures may not deviate dramatically from current ideas, we can expect significant improvements in the ease and efficiency with which the power of large numbers of processors can be harnessed. The number of processors that fit in a given space is likely to increase. The HP V2200 and Sun UltraEnterprise 4000 are indicative of a trend that is likely to be repeated throughout the industry. These two systems pack over a dozen processors into a deskside chassis. In general, however, a good deal remains to be done in developing systems software that truly scales with the number of processors, offers distributed computing services, provides the user with a simple interface to the system, and has good performance for large configurations.

Implications for the Export Control Regime

This chapter has presented a sampling of some of the most significant trends in the HPC industry from the perspective of the export control regime. From them, a number of conclusions can be drawn:

- The growth of microprocessor performance will continue into the early part of the next century at rates comparable to the recent past. In 1997, most RISC microprocessors passed the 500 Mtops mark. By 1999, most will exceed 1000 Mtops, and the first microprocessors above 2000 Mtops will be on the market. Industry projects that CTP levels will rise dramatically around the turn of the century, reaching approximately 10,000 Mtops for processors in volume production in 2001.
- Microprocessor performance 5 to 10 years into the future will rely more on growing numbers of functional units and volume of on-chip memory, and less on higher clock frequencies, whose rate of increase will taper off. While progress beyond this point will require solving some very difficult technological challenges, it is certainly premature to say that the industry will hit fundamental and insurmountable barriers.
- At a minimum, multiprocessor systems will ride the microprocessor performance curve.
- Recent advances in low-cost, commercially available interconnects have staked out a "middle ground" between traditional, proprietary high-speed interconnects and the local area networking technologies used in workstation clusters in the past. While the gap between these new clustering interconnects and higher-end proprietary interconnects has narrowed, it has not vanished.
- A key distinguishing factor between clusters and integrated systems provided by vendors is software. Creating a system with a great deal of capacity (total CPU cycles) is easy.

Creating a system with high capability and good usability features is difficult, requiring extensive expenditure of time, money, and expertise. Existing clustering efforts using mass-market commercial off-the-shelf technologies have demonstrated sustained performance of over one Gflops, but such systems still lack fully functional and robust software environments, particularly for configurations with large (>100) nodes.

- The existence of parallel systems employing clustering interconnects should not, for the present, legitimize the export of systems with comparable numbers of processors using much higher speed and lower latency interconnects combined with a stable and substantial suite of systems software.
- Although dramatic changes in architectures do not appear to be on the horizon, the cumulative effect of incremental changes in architecture, hardware, and software will be a substantial increase in the ability of users throughout the world to apply multiprocessor computing resources to applications of their choice. From a user's perspective, the distinction between the high- and low-end machines is shrinking. Emerging and commercially available interconnects make it much easier to assemble modest configuration systems that behave similarly to the highest-end machines, minus only the raw computing performance. The movement into the mainstream of NUMA or related technologies will enhance the ability of practitioners to harness the power of larger numbers of processors using techniques learned on smaller configurations.

References

[1] Gwennap, L., "PowerPC Team Outlines Future Plans," *Microprocessor Report,* Vol. 10, No. 11, Aug 26, 1996.

[2] Matzke, D., "Will Physical Scalability Sabotage Performance Gains?," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 37–39.

[3] "Architects Look to Processors of Future," *Microprocessor Report,* Vol. 10, No. 10, Aug 5, 1996.

[4] Takahashi, D., "Chip Firms Face Technological Hurdles That May Curb Growth, Report Suggests," *Wall Street Journal,* Dec 1, 1997, p. B8.

[5] The National Technology Road Map for Semiconductors, Semiconductor Industry Association, San Jose, CA, 1994.

[6] Burger, D. and J. R. Goodman, "Billion-Transistor Architectures," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 46–48.

[7] Patt, Y. N. et al., "One Billion Transistors, One Uniprocessor, One Chip," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 51–57.

[8] Smith, J. E. and S. Vajapeyam, "Trace Processors: Moving to Fourth-Generation Microarchitectures," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 68–74.

[9] Kozyrakis, C. E. et al., "Scalable Processors in the Billion-Transistor Era: IRAM," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 75–78.

[10] Hammond, L., B. A. Nayfeh, and K. Olukotun, "A Single-Chip Multiprocessor," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 79–85.

[11] Waingold, E. et al., "Baring It All to Software: Raw Machines," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 86–93.

[12] Lipasti, M. H. and J. P. Shen, "Superspeculative Microarchitecture for Beyond AD 2000," *IEEE Computer,* Vol. 30, No. 9, Sep, 1997, pp. 59–66.

[13]Gwennap, L., "Estimating IC Manufacturing Costs," *Microprocessor Report*, Vol. 7, No. 10, Aug 2, 1993.

[14]Smith, N. P., "Hits & Misses: Benchmarks Again, Fab Costs, and Jell-O," *HPCWire*, Nov 10, 1995 (Item 8060).

[15]Yu, A., "The Future of Microprocessors," *IEEE Micro*, Dec, 1996, pp. 46–53.

[16]Diefendorff, K. and P. K. Dubey, "How Multimedia Workloads Will Change Processor Design," *IEEE Computer*, Vol. 30, No. 9, Sep, 1997, pp. 43–45.

[17]Gwennap, L., "Multimedia Boom Affects CPU Design," *Microprocessor Report*, Vol. 10, No. 16, Dec 5, 1994.

[18]Gwennap, L., "UltraSPARC Adds Multimedia Instructions," *Microprocessor Report*, Vol. 8, No. 16, Dec 4, 1994.

[19]Gwennap, L., "Digital, MIPS Add Multimedia Extensions," *Microprocessor Report*, Vol. 10, No. 15, Nov 18, 1996.

[20]Wilson, R., "Sun's processor focuses on the problems of intense multiprocessing—UltraSPARC-III targets servers, LAN switches," *Electronics Engineering Times*, No. 974, Oct 6, 1997.

[21]Song, P., "UltraSparc-3 Aims at MP Servers," *Microprocessor Report*, Vol. 11, No. 14, Oct 27, 1997.

[22]Gwennap, L., "Intel's P6 Bus Designed for Multiprocessing," *Microprocessor Report*, Vol. 9, No. 7, May 30, 1995, pp. 1–6.

[23]Myrinet. A Brief, Technical Overview, Myricom, Inc., http://www.myri.com/myrinet/overview.html, July 7, 1997.

[24]Sterling, T. et al., Findings of the first NASA workshop on Beowulf-class clustered computing, Oct. 22–23, 1997, Pasadena, CA, Preliminary Report, Nov. 11, 1997, Jet Propulsion Laboratory.

[25]Beck, A., "Dave Gustavson Answers Questions about SCI: Part I," *HPCWire*, Oct 4, 1996 (Item 10249).

[26]Beck, A., "Dave Gustavson Answers Questions about SCI: Part II," *HPCWire*, Oct 11, 1996 (Item 10282).

[27]Beck, A., "Dave Gustavson Answers Questions about SCI: Part III," *HPCWire*, Oct 18, 1996 (Item 10316).

[28]IBM Corp., The RS/6000 SP High-Performance Communication Network, http://www.rs6000.ibm.com/resource/technology/spsw1/spswp1.book1.html.

Chapter 3: HPC Controllability and Export Control Thresholds

As discussed in *Building on the Basics* [1,2], there are levels below which a control threshold is not viable. At these levels, the policy seeks to control that which is uncontrollable and becomes ineffective. The 1995 study discussed in some depth the factors that influence controllability and made determinations regarding what performance levels were, and were not, controllable. In this chapter we refine the discussion and elaborate upon some of the ideas presented in the earlier work. We conclude with a discussion of a number of options for policy makers regarding the lower bound of the range within which viable control thresholds may exist.

Kinds of Controllability

During the 1990s, the definition of controllability has become considerably more complex than it was in previous decades. In particular, industry-wide trends toward parallel processing and systems scalability make it necessary to make a distinction between two related but not interchangeable concepts:

(1) controllability of platforms, and

(2) controllability of performance levels.

The first refers to the ability to prevent entities of national security concern from acquiring and using hardware/software platforms for unsanctioned purposes. The latter refers to the ability to prevent such entities from acquiring a particular level of computational capability by any means they can. In the past, the distinction between the two was minimal; today, it is critical. We explain each concept in more depth below.

Controllability of Platforms

The controllability of platforms refers to the extent to which the export and use of computing systems can be adequately monitored and regulated. When systems are highly controllable, the probability that diversion can be prevented is close to 100 percent. The cost of carrying out a successful diversion and maintaining the system may be so high that, given the low odds of success, an effort is not seriously undertaken. For systems that have low controllability, the premium in extra time and money expended for a diversion may be so small, and the barriers to diversion so low, that successful diversion is likely to occur with some regularity, regardless of what the export control policies are.

While there is no precise, widely agreed upon means by which controllability can be assessed and measured, there is agreement that platform controllability is a function of a number of factors. These include:

- size
- infrastructure requirements
- vendor dependence
- size of installed base
- age
- vendor distribution networks
- price

Size. Small units are more easily transported than large ones. Today's computers fall into a number of categories, shown in Table 3–1, based on the size and nature of the chassis.

Chassis	*Sample Systems*	*Commentary*
Proprietary	Most Cray and HP/Convex, CM-5	Many high-end systems have a unique and distinctive appearance, due to their highly customized chassis. The chassis is designed primarily to meet the technical requirements of the system, and to offer an attractive appearance. The physical size can range from rather large to rather small. The Cray C90 has a footprint of 80 ft^2 including the I/O subsystem, while a Cray-2 has a footprint of only 16 ft^2. To be sure, many proprietary systems require extensive cooling and I/O subsystems that occupy additional space.
Multiple-rack	IBM RS/6000 SP (SP2), Origin2000	The basic physical chassis is a rack. Vendor racks differ from one another and are certainly not interchangeable, but they are rectangular: 5–7 feet high, 2–4 feet wide, and 3–4 feet deep. Large configurations consist of multiple, interconnected racks. In the extreme, configurations can be huge. The ASCI Red (Intel) configuration at Sandia National Laboratories has a footprint of 1000 square feet.
Single-rack	PowerChallenge XL, AlphaServer 8400, Ultra Enterprise 6000, IBM R40, HP T-500, T-600	The physical chassis is a rack, as just described. The system may not scale beyond a single rack. If it does, the multi-rack configurations have substantially different programming paradigms or performance characteristics than the single-rack system. The Silicon Graphics PowerChallenge Array is a good example. The individual PowerChallenge racks are shared-memory systems, while the array offers a distributed-memory paradigm.
Deskside	Origin200 dual-tower, PowerChallenge L and Challenge, AlphaServer 4100 and 2100, IBM J40, Ultra Enterprise 3000, 4000.	The desk-side chassis rarely stands taller than three feet. Such systems are highly compact and portable. They typically offer multiprocessing capability, with maximum numbers of processors ranging from two to eight.
Desktop	Numerous uniprocessor workstations, some dual-processor workstations.	The desktop chassis is the most popular for today's single-user workstations and personal computers. The number of processors is typically one or two. "Tower" cabinets that may sit beside a desk for convenience fall into this category as well.

Table 3–1. HPC chassis

Infrastructure requirements. The principal distinction in infrastructure requirements is between systems that are air-cooled and those that require liquid cooling. Air-cooled units are easier to install and maintain. Liquid-cooled units, e.g. the Cray-2, can require extensive cooling systems whose size, power, and maintenance requirements exceed those of the computer itself.

A dominant trend in high-performance computer manufacturing has been the use of CMOS rather than Emitter-Coupled Logic (ECL) components. The latter has been used in the past because it offered higher speeds. However, it also required substantially more power per unit of volume and therefore required liquid-cooling systems. The economics of high-performance computing has forced all HPC vendors to offer CMOS systems and reduce the number of models employing the more costly ECL.

Vendor dependence. The more dependent a system is on the vendor for installation, maintenance, and operation, the more controllable the system is. Periodic visits to an installation provide a vendor with an opportunity to assess not only the location of a system, but also the nature and extent of its use. Although a vendor may not be able to assess with certainty all the uses to which a system is put—especially if the user is trying to conceal such information—the more frequent and involved the site visits, the more closely the vendor can monitor the system. Two extremes illustrate the range of "care and feeding" that systems require. On the one hand, large, customized configurations of pioneering systems such as those developed under the Accelerated Strategic Computing Initiative (ASCI) program require continuous, on-site presence by vendor representatives. Large-configuration but commercially available systems might involve vendor representatives intensively during system installation, acceptance testing, and upgrading. During routine operation, skilled employees of a systems integrator (e.g., Nichols Research, Northrop Grumman) monitor and operate the system. At the other extreme, personal computers sold through mail-order companies usually require no vendor involvement once the package leaves the warehouse. Users are responsible for, and usually capable of managing, the installation, upgrade, operation, and maintenance of such systems. Moreover, an extensive infrastructure of third-party companies providing support services exists throughout the world.

Between these two extremes are a number of degrees of customer dependence on vendors. These can be classified using the categories shown in Table 3–2, which are ranked from greatest dependence to least.

Vendor Dependence	*Sample Systems*	*Commentary*
Total	Cray C90 and T90, large configurations of massively parallel platforms such as the SP2. Possibly Cray T3D and T3E, depending on capabilities of user.	The customer is totally dependent on the vendor. The vendor, or a vendor representative, performs all installation, operation, and maintenance tasks on a continuous basis. Users have no direct interaction with the system. Jobs submitted for runs are loaded and executed by vendor representatives.
High	Cray J90, HP/Convex Exemplar, IBM's RS/6000 SP (formerly called the SP2), Cray Origin2000.	The vendor installs, maintains, and upgrades the system, but the customer may manage the system during routine operation and perform software upgrades.
Moderate	DEC's AlphaServer 8400, SGI Challenge XL, PowerChallenge XL, Sun's Ultra Enterprise, HP's T-series	The vendor typically installs the system and provides supporting services during unusual circumstances, such as system failure or extensive upgrades. The customer may perform routine maintenance and simple CPU and memory upgrades. Third-party companies may provide such services as well as the vendor.

Low	AlphaServer 2100, AlphaServer 4100, SGI Challenge L , PowerChallenge L	The vendor may provide advisory services as requested by the customer. Customers with in-house expertise can function well without such services, while customers with little expertise may rely on them more heavily. An infrastructure of large and small companies providing this expertise may exist.
None	Many single and dual-processor workstations and most personal computers and Intel-based servers fall in this category.	The vendor is not usually involved in installing, maintaining, or operating the system. The typical customer does not need such support. Third-party individuals or companies may provide any services necessary.

Table 3–2. Categories of dependence on vendors

Installed base. It is probably not possible to determine precisely at what point the number of installations becomes so large that the vendor loses the ability to track the location and use of each unit. Such a point is a function not only of the number of units installed, but also of the nature of the distribution and supporting services networks and the amount of resources the vendor is willing to bring to bear on the problem. However, it is helpful to classify a given model's installed base in one of several categories. Table 3–3 provides a possible classification of the installed base, and identifies some representative systems for each category, circa fourth quarter 1997.

Installed Base	*Commentary*
10s	Traditional supercomputers lie in this category. Vendors have a precise understanding of where individual systems are located.
100s	This category often includes high-end systems that have been shipping only a few months, but which may eventually enjoy a substantially larger installed base. This category also includes a number of systems pioneering new markets (e.g., commercial HPC) for their vendors. Vendors usually are able to monitor individual installations with a fair degree of confidence.
1000s	Most high-end servers fall in this category. It is a gray area. Monitoring of individual installations becomes difficult, but may be possible, depending on the nature of the vendor's sales and support infrastructure. When the installed base reaches the high 1000s, there is a high probability that there will be leaks in control mechanisms.
10,000s	Most mid-range, deskside servers fall in this category. Systems may be considered mass-market products. As a practical matter, it is all but impossible to keep close track of each unit.
100,000s	UNIX/RISC workstations and Windows NT servers fall in this category. Systems are increasingly like commodities.

Table 3–3. Impact of installed base on controllability (circa 4Q 1997)

Age. Systems based on microprocessor technologies have product cycles that track the microprocessor cycles. At present, vendors are introducing new generations of their microprocessors approximately every one to two years. Over the course of each generation, vendors increase the clock speed a number of times. Consequently, the majority of units of a particular model are sold within two years of the product's introduction. Systems like the Cray vector-pipelined systems have product cycles on the order of three to four years.

As systems grow older, they are frequently sold to brokers or refurbishers and resold, without the knowledge or involvement of the original vendor. Most users keep their systems

running productively for three to four years; it is difficult to amortize system costs in less time than this. However, a secondary market for rack-based systems is likely to start developing approximately two years following their first shipments. Desktop and deskside models may be found on secondary markets a year following introduction.

Vendor distribution networks. If systems are sold through a network of value-added resellers (VAR), distributors, OEMs, or other non-direct channels, no single individual or entity may have a complete understanding of the history of a particular system.

Price. Since markets (and size of the installed base) are often strongly correlated to price, systems with lower prices are likely to be less controllable than more expensive systems for at least three reasons. First, the lower the price of a target system, the greater the number of organizations at restricted destinations able to afford the system. The number of attempts to acquire a particular kind of system may increase. Second, vendors are willing to commit more resources toward controlling an expensive system than a cheap system. Profit margins are usually lower on the cheaper systems that occupy more price-competitive market niches. In addition, if a fixed fraction of the system price—say, 5 percent—were devoted to control measures, substantially more money would be available for this purpose from the sale of a $10 million system than from the sale of a $100,000 system. Third, customers prefer to have vendors install and upgrade expensive systems. The issue is risk management. The systems cost too much to risk an improperly conducted installation.

Based on these factors, systems can, at a given instant in time, be classified according to their controllability. Table 3–4 lists a sample of systems in each of four controllability categories. Costs have been rounded off. The installed base figures apply to most but not necessarily all models in a given category.

Controllability of Performance Levels

Controllability of platforms has little to do, inherently, with performance. If a particular system has a high price, a small installed base, high dependence on the vendor for operations and maintenance, etc., then the vendor is likely to be able to keep close track of that system, regardless of its performance. For example, the Cray YMP/1, as a platform, is still controllable, even though it has a performance of 500 Mtops, which is lower than the performance of many widely available uni- and dual-processor workstations. Efforts to prevent Cray YMP *platforms* from being diverted are likely to be successful, even though efforts today to prevent foreign acquisition of Cray YMP *equivalent performance* are unlikely to be successful.

What determines the controllability of a particular performance level? Principally,

- the performance of systems available from foreign sources not supporting U.S. export control policies,
- the performance of uncontrollable platforms, and
- the scalability of platforms.

Vendor	Model/ Processor	Size	Cooling	Year Delivered	Dependence on Vendor	Cost of Entry-level System (at Introduction)	Installed Base	Distribution Network
CONTROLLABILITY: VERY HIGH								
Cray	T90	non-standard chassis	water	1995 4Q	Total	$2.5 million		Direct
Cray	T3D	non-standard chassis	air	1993 3Q	Total/High	$1 million	typically 10s	Direct
Cray	T3E/600	non-standard chassis	water	1996	Total			Direct
CONTROLLABILITY: HIGH								
HP/ Convex	Exemplar X-Class	non-standard chassis	air	1997 1Q	High	$ 200,000		Direct
Cray	J916	non-standard chassis	air	1995 1Q	High	$ 200,000		Direct
Sun	UltraEnterprise 10000	rack	air	1996	High/ Moderate	$ 500,000	typically 100s	Direct/ VAR
SGI	Origin2000 R10000/180	rack	air	1996 4Q	High/ Moderate	$ 100,000		Direct/ Dealership/ VAR
IBM	SP2 Power2 High Node	multiple racks	air	1996 3Q	High	$ 150,000		Direct/ VAR
CONTROLLABILITY: MODERATE								
DEC	AlphaServer 8400 EV5	rack	air	1995 2Q	High/ Moderate	$ 200,000		Direct
SGI	Power Challenge L R10000/200	desk-side	air	1996 1Q	High/ Moderate	$ 100,000		Direct/ Dealership/ VAR
SGI	Origin2000 R10000/180	desk-side	air	1996 4Q	Low	$ 30,000	typically 1000s	Direct/ Dealership/ VAR
Sun	Ultra Enterprise 5000	rack	air	1996	High/ Moderate	$ 100,000		Direct/ Dealership/ VAR
IBM	SP2 Power2/77	multiple racks	air	1995	High	$ 100,000		Direct/ VAR
CONTROLLABILITY: LOW								
Sun	UltraEnterprise 3000	desk-side	air	1996	Moderate/ Low	$ 50,000		Direct/ Dealership/ VAR
DEC	AlphaServer 2100	short rack	air	1995 2Q	Low	$ 60,000	typically 10,000s	Direct/ Dealership/ VAR
DEC	AlphaServer 4100	desk-side	air	1996 2Q	Low	$ 60,000		Direct/ Dealership/ VAR
HP	HP 9000 4x0	desk-side	air	1995	Low	$ 50,000		Direct/ Dealership/ VAR

Table 3–4. Examples of platform controllability (4Q 1997)

Performance of foreign systems. Systems available from foreign sources not cooperating with U.S. export control policies may also be considered uncontrollable platforms. Foreign availability has traditionally been an important element of the export control debate. Separating out this category of uncontrollable platforms is analytically helpful, since it adds to the set of uncontrollable platforms systems that are uncontrollable not because of their technical or market qualities, but because of their point of origin. Today, the systems developed indigenously in countries outside the United States, Japan, and Western Europe are not commercially competitive with U.S. systems, nor do they have particularly high levels of performance. Table 3–5 shows some of the leading systems from Tier 3 countries.[1]

Model	*Country*	*Organization*	*Year*	*Description*	*Peak Mflops*	*Mtops (est.)*	*Units installed (est.)*
MVS-100	Russia	Kvant Scientific Research Institute	1995	32 i860s	2400	1,500	~20
MP-3	Russia	VNII Experimental Physics	1995	8 i860s	600	482	<20
MP-XX	Russia	VNII Experimental Physics	1997?	32 Pentium(Pro)	6400	5,917	pending?
Param-9000	India	Center for the Development of Advanced Computing	1995	64(?) SuperSparc II/75 (200 max)	4,800 (15,000 max)	3,685	<30
Param-10000	India	Center for the Development of Advanced Computing	1998	UltraSPARC			
Pace-2	India	Defense Research and Development Laboratories			2,000 (max)		<10
Pace+	India	Defense Research and Development Laboratories	1998?		30,000 (max)		
Dawning-1000	PRC	National Research Center for Intelligent Computing Systems	1995	32 i860	2400	1,500	
Galaxy-II	PRC	National Defense University of Science and Technology	1993	4 CPUs	400		<20
Galaxy-III	PRC	National Defense University of Science and Technology	1997	128 processor (max)	13000 (max)		<10

Table 3–5. Selected HPC systems from Tier 3 countries

[1] The 1996 revisions to export control policy classified countries into four categories, or tiers, with different control thresholds for each. Tier 1 includes NATO and NAFTA member states and other export control partners of the United States. Tier 2 includes most South and Latin American, African, Asian, and Eastern European countries. Tier 3 includes nuclear powers such as Russia, the People's Republic of China, and India; and most Middle Eastern countries, former Soviet republics, and other countries of nuclear proliferation concern. Tier 4 includes the so-called pariah states of Libya, North Korea, Iraq, etc.

The table illustrates that while a few countries do have established HPC programs, their indigenous systems are not in widespread use and, for the most part, have performances under 4000 Mtops, usually far below this level. It is likely that the Russians are able to construct a system employing 32 Pentium Pro/200 processors; it is not clear whether they have in fact done so. The country continues to have difficulty financing development projects.

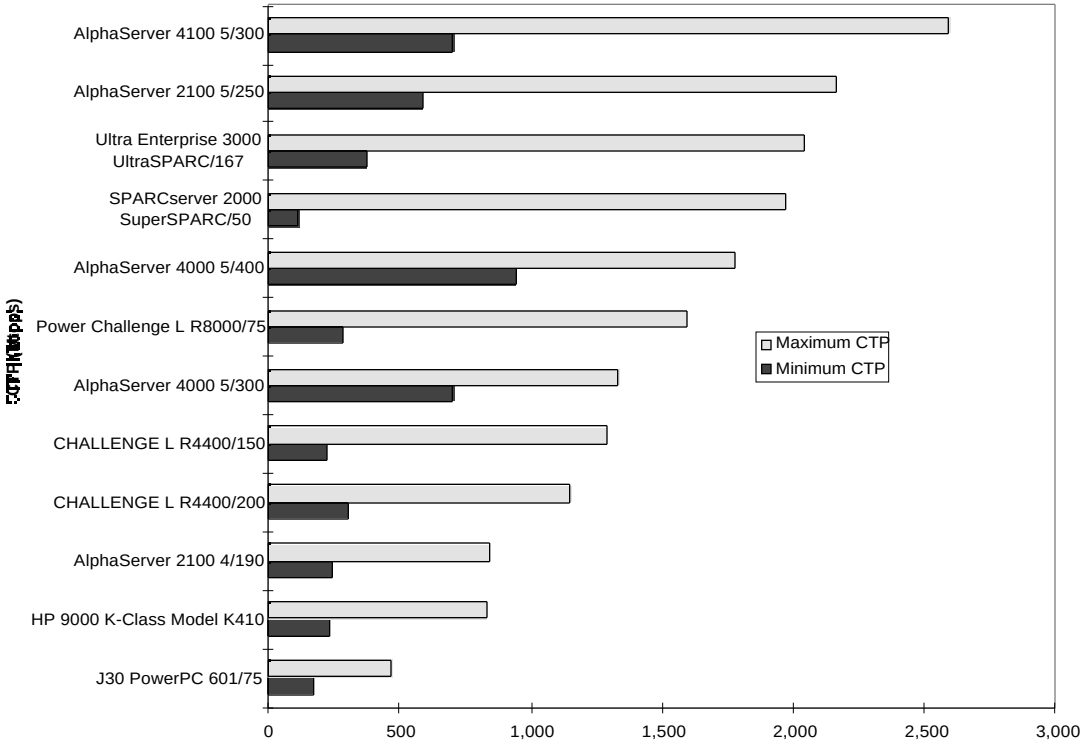

Figure 3–1. Sampling of platforms with low controllability (4Q 1997)

Performance of uncontrollable platforms. The performance level provided by the most powerful uncontrollable system is, clearly, uncontrollable. The fact that there may be controllable *platforms* with *lower* performance (e.g., the Cray YMP example) does not negate this reality. Figure 3–1 and Figure 3–2 show the performance of a number of systems that have low and moderate controllability, circa 4Q 1997.

A number of the systems in Figure 3–2, such as the AlphaServer 4100/466, the PowerChallenge L R10000/200, the IBM J50, and various HP K-Class desk-side servers, will slip into the Low Controllability category in 1998 on the strength of their installed base. These systems have CTP levels in the 4000–5000 Mtops range.

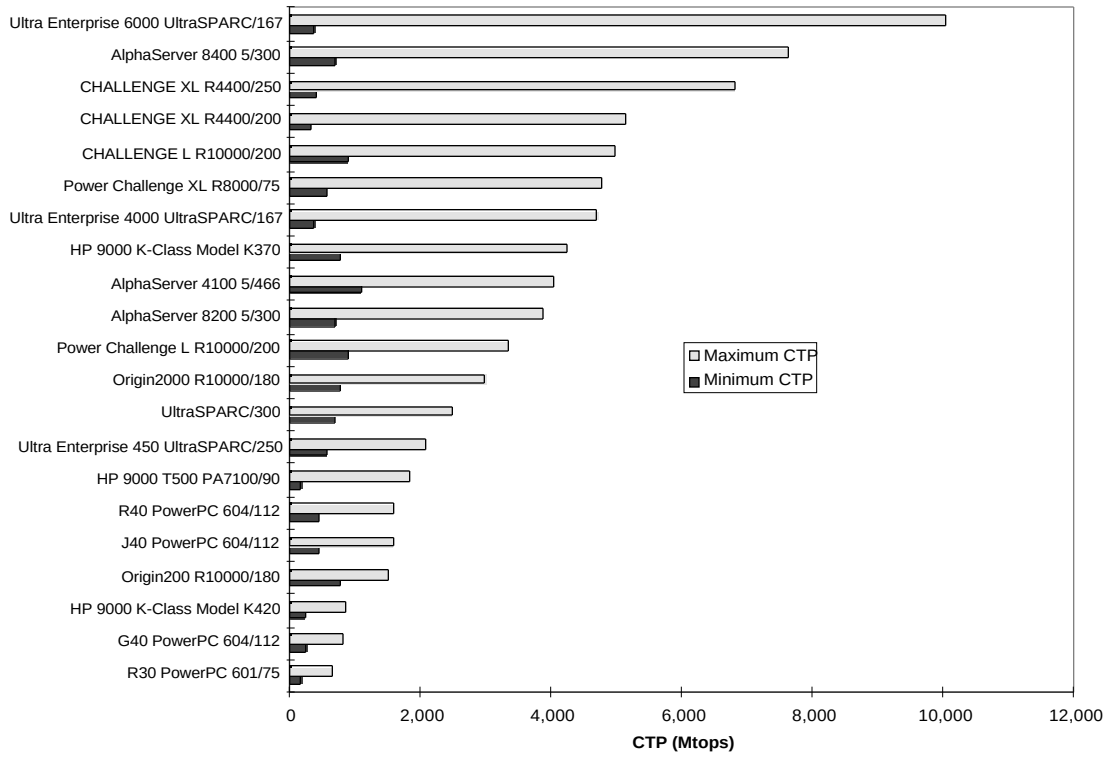

Figure 3–2. Sampling of platforms with moderate controllability (4Q 1997)

Scalability of Platforms. Scalability of platforms, the third determinant, makes it possible to increase the performance of a system incrementally by adding computational resources, such as CPUs, memory, I/O, and interconnect bandwidth, rather than by acquiring a replacement system. The controllability of performance levels is strongly affected when the range of scalability is large, and users without the assistance of vendor representatives can add computational resources in the field.

To see this, we must draw a distinction between several different kinds of performance:

- *Configuration performance.* The configuration performance is the CTP of a specific configuration to be shipped to a customer. This is the performance that is currently used to determine if a system requires an individually validated license. If an IVL is required, the configuration performance is included in the export license application.
- *Maximum performance.* The maximum performance is the CTP of the largest configuration of a system that could be constructed given the necessary resources, vendor support, and specialized technologies. All commercial systems today have a maximum number of processors, a limit established by design. For example, a Cray C90 has a maximum configuration of sixteen processors (21,125 Mtops). The SGI Origin2000 has a maximum configuration of 64 processors, with a maximum performance of 23,488 Mtops. (The Cray Origin2000 includes a specialized "metarouter" that enables scaling to 128 processors.) The DEC Alpha 4100 can be configured with up to 4 processors (4,033 Mtops).
- *End-user attainable performance.* The end-user attainable performance is the performance of the largest configuration of an individual, tightly coupled system an end-user could assemble without vendor support, using only the hardware and software provided with

lesser configurations. This definition does not apply to more loosely coupled agglomerations such as clusters. By "lesser configuration" we usually mean a configuration that can be exported under existing export control policies.

The scalability of a system has enormous significance for the relationship of these three quantities. Figure 3–3 illustrates their relationship when systems are *not* scalable. Suppose (a) represents two uniprocessor configurations of a hypothetical vector-pipelined system that the vendor sells in configurations between one and four processors. Each uniprocessor configuration falls below the control threshold, but dual- and four-way configurations lie above the threshold. Upgrading from one to two or four processors requires substantial vendor involvement and specialized technology, represented by the circular arrows in (c). When systems cannot be upgraded in the field, upgrading requires the complete replacement of one configuration by another. The diagram illustrates that if two uniprocessors are exported, the result is two uniprocessor systems. Without the specialized expertise and technology provided by the vendor, it is not possible to combine two uniprocessor systems into a dual-processor system. In this case, the attainable performance is equivalent to the configuration performance.

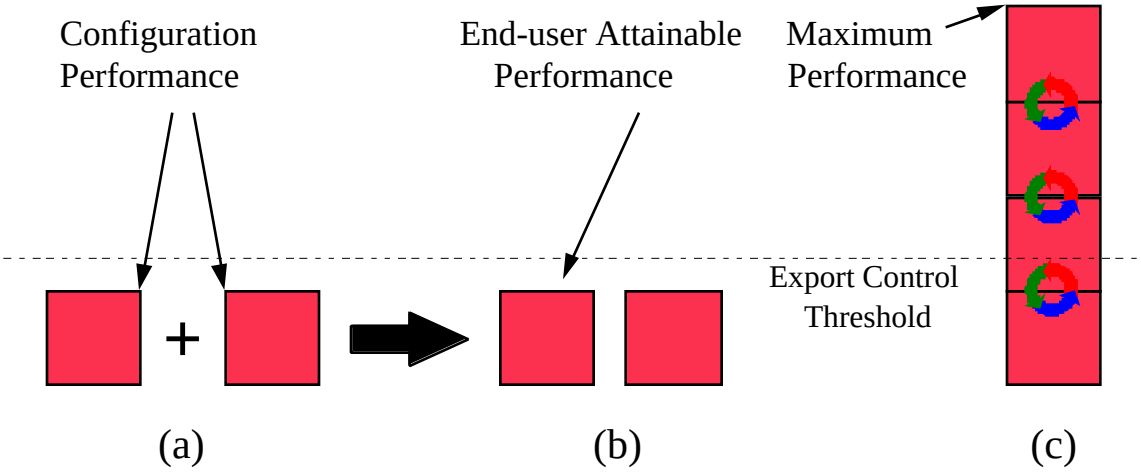

Figure 3–3. Configuration, attainable, and maximum performance of non-scalable systems

Figure 3–4 illustrates the three concepts when systems *are* easily scalable. Suppose there is a hypothetical rack-based parallel processing system where each rack can have up to 16 processors. Two racks can be integrated into a maximum configuration system (32 processors), but only by the vendor, who must install, say, a proprietary high-speed switch represented by the circular arrows shown in (c). Within a rack, CPU and I/O boards can be easily added or removed without specialized expertise or technology. Further suppose that an eight-processor configuration has a performance below the export control threshold, while a full rack's performance exceeds it.

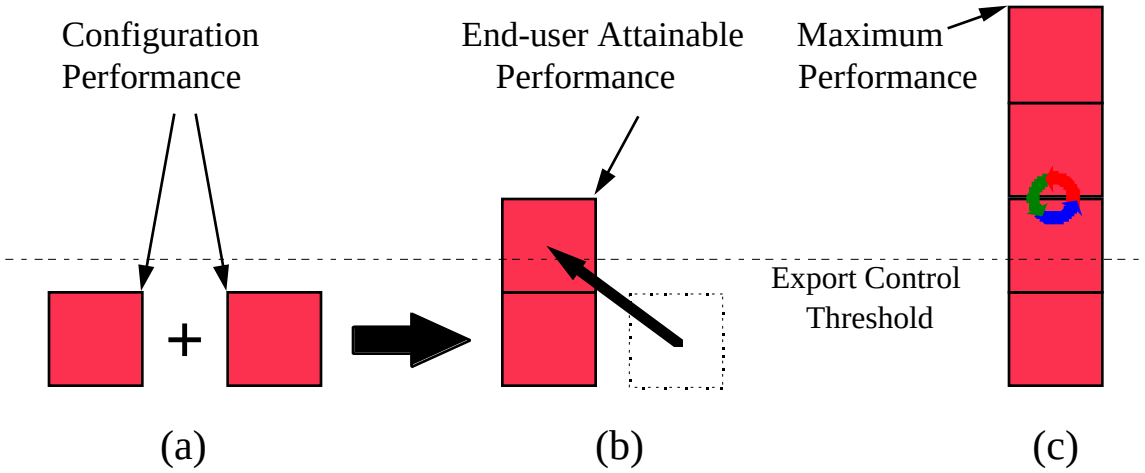

Figure 3–4. Configuration, attainable, and maximum performance for scalable systems

Under these conditions, the configuration performance and end-user attainable performance are quite different. Two half-full racks may be exported perfectly legitimately under general license (a). Once in place, the CPU, memory, and I/O boards can be moved from one rack to the other as shown in (b) without vendor knowledge or assistance. The resulting configuration would have the same computational capacity (aggregate operations per second and volume of memory) as the two originals, but significantly greater capability as measured in Mtops. Acquiring additional half-full racks would not help the user attain the maximum performance (c), however, since individual racks are not shipped with a high-speed switch.

Scalability and the Export Control Regime

The trend toward scalable architectures causes fundamental problems for the export regime as currently formulated. The regime relies on a single technological feature, configuration performance, to distinguish those systems that can provide foreign entities with a particular level of computational capability from those that cannot. In other words, the control regime tries to restrict the performance an end-user can attain by placing restrictions on configuration performance.

When systems are not scalable, the current regime works well. Configuration performance and end-user attainable performance are identical, and controlling the former results in effective controls over the latter. Under these circumstances, the challenge for policy makers is to identify which platforms are uncontrollable and set the control threshold at or above their performance.

When systems are readily scalable, the current regime becomes unstable at many control thresholds. Control thresholds set below the performance of small configuration scalable systems are not viable. The nature of most high-performance computing systems manufactured today is that their smallest configurations are one or two processor systems. This level of performance, ranging today from roughly 200 to 2000 Mtops, depending on the processor, is filled with uncontrollable platforms. According to Dataquest, a market research firm, nearly 200,000 UNIX-RISC systems were shipped during the first quarter of 1997 [3]. According to International Data Corp. (IDC), over 600,000 UNIX/RISC workstations were shipped in 1997 [4]. Each of these systems is above 200 Mtops, and most are likely to be above 500 Mtops. By the end of 1998, nearly all UNIX-RISC systems shipped will be above 500 Mtops, and dual- or quad-processor systems above 2000 Mtops will be commonplace.

Unfortunately, setting controls marginally higher than the performance of workstations does not prevent foreign entities from attaining significantly higher performance levels under the current control regime. Figure 3–5 illustrates the end-user attainable performance for several systems introduced within the last two years. Each of these systems has a minimum configuration at the one- or two-processor level that could be shipped under general license.

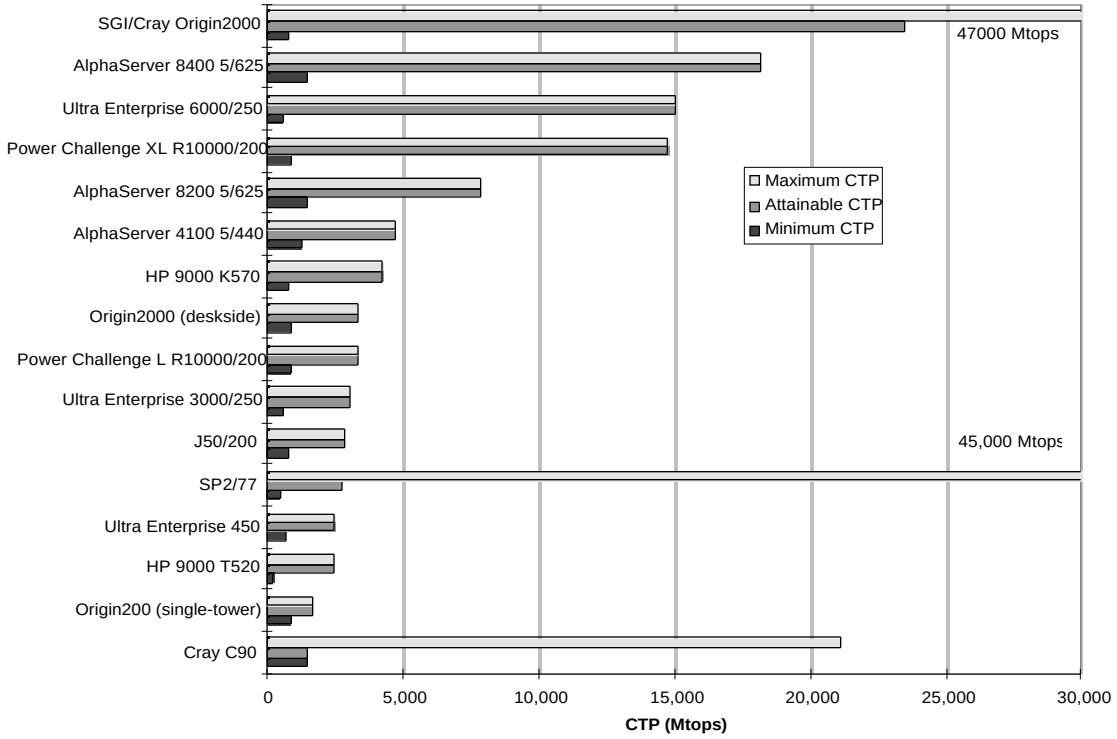

Figure 3–5. Minimum, maximum, and end-user attainable performance of selected HPC systems

When thresholds are set between the minimum and end-user attainable performance of systems, scalability must be controlled by ongoing and long-term monitoring of installations by vendors, U.S. Government officials, or both. This is an expensive and uncertain effort, especially if the number of installations is large.

Figure 3–6 shows how the level of end-user attainable or uncontrollable performance has risen during the 1990s. The graph incorporates a time lag to allow for markets to mature. A system introduced in 1996 is plotted in 1998. The systems reflected in the graph are mostly rack-based systems with very easy scalability whose minimum configurations lie beneath control thresholds. The rapid rise in performance has been fueled by a combination of increasing processor performance and, to a lesser extent, increasing numbers of processors in a system. The atmospheric performance of the SGI systems reflects the ability to link together multiple SGI racks into a single, integrated, shared memory system using no more than cables.

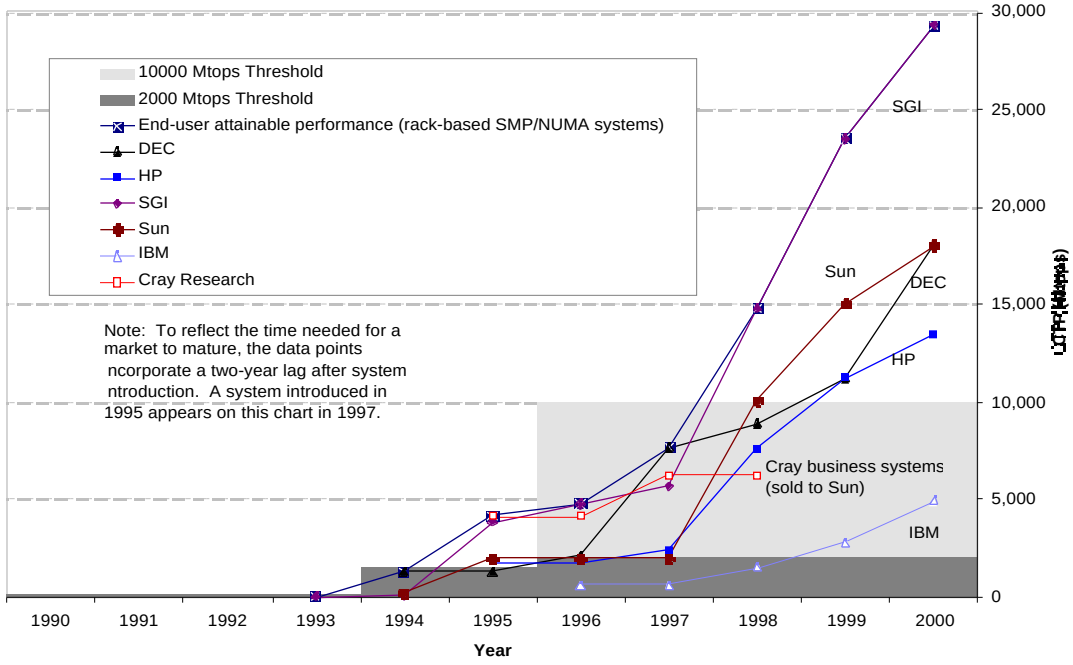

Figure 3–6. Threshold of uncontrollable (attainable) performance

A fundamental difficulty is that the present control regime uses the configuration performance as a surrogate for end-user attainable performance during licensing considerations. When systems are easily scalable, these two values can be quite different, and the control policy may fail to achieve its objective.

An alternative approach is to take into account the end-user attainable *performance of a system during licensing considerations rather than the configuration performance. Under such circumstances, export licensing decisions are made directly on the basis of what the policy seeks to control in the first place: the performance a foreign entity can attain by acquiring U.S. computer technology. No surrogate values are needed.*

The following example illustrates one of the principal differences between control regimes based on configuration and those based on end-user attainable performance. Suppose a vendor has two systems it seeks to export. Both systems use the same components, but one system can be scaled easily to 8 processors; the other, to 16. The two customers have requested four-processor versions of the two models. Further suppose that the control threshold is set at a performance level equivalent to an eight-processor configuration. Under the current regime, each configuration could be exported under general license because the configuration performance of each lies below the control threshold. However, one customer would be able to scale the system to a performance level well above the threshold without ever having had the export subject to review by licensing authorities.

Although they have the same configuration performance, the two systems would be treated differently under a regime based on end-user attainable performance. Under this regime, the system scalable to eight processors could be exported under general license, but the other system would require an individually validated license and be subject to review by licensing officials.

Barriers to Scalability

At the heart of the concept of end-user attainable performance is the extent to which end-users can scale smaller systems into larger ones. There are at least three factors that may prevent end-users from scaling systems and thereby limit the attainable performance:

(1) specialized expertise

(2) specialized software

(3) specialized hardware

Specialized expertise continues to be needed to upgrade some of the massively parallel systems, such as the Cray T3D and T3E and HP/Convex Exemplar machines, and to upgrade parallel vector-pipelined systems like the Cray C90, T90, or J90. Even the IBM SP2 requires rather substantial expertise when new racks of nodes are added. Specialized expertise is usually not required, although it may be customary, for SMP system upgrades.

Limitations in the operating system and interconnect have placed ceilings on scalability. The operating system provides an array of services to applications software including process management, I/O management, and file management. For a system to be fully scalable, these services, as well as the hardware, must be scalable. While considerable progress has been made in scalable operating systems, limitations continue to exist, especially for operating systems supporting shared-memory architectures. Although some systems support shared-memory architecture for up to 64 processors, such processor counts have not been typical in the industry. Systems based on Windows NT still do not scale well beyond 8 processors, although this number is increasing.

Specialized hardware is sometimes required when scaling beyond the smaller configurations. For example, up to four Origin2000 racks with a total of 64 processors can be joined together using just cables. Scaling past 64 processors, however, requires a metarouter, a rack-based unit with a sophisticated proprietary interconnect manufactured only by Cray. The number of systems requiring a metarouter is likely to be quite limited, probably not more than a few hundred.

For SMP-class systems, the most common barrier to scalability is the size of the chassis. Desktop, deskside, and rack-based models may all incorporate precisely the same component base. Indeed, most computer companies try to leverage the same or similar components throughout their product lines in order to save on development and production costs. However, a deskside system with room for four or six processors cannot be scaled to 16 or 32 processors.

Intentionally, we limit our definition of end-user attainable performance to integrated, tightly coupled systems provided by HPC vendors. Although it has been clearly demonstrated that competent users can cluster together multiple workstations using readily available interconnects and create systems with substantial computing power [5], determining and applying the end-user attainable performance in such contexts is problematic. First, it would be very difficult to determine the end-user attainable performance in an uncontroversial manner. The performance of clusters is highly variable, especially as they grow in size. What are the limits to scalability? When does performance begin to degrade unacceptably? These are open research questions today. Second, all computing systems sold today can be networked. The end-user attainable performance of all platforms would degenerate into an ill-defined large number that is of little practical use for licensing officials. The consequence of limiting the discussion of end-user attainable performance to individual, tightly coupled systems is that some end-users will, in fact, cluster together systems and perform useful work on them. Since clusters are only as controllable as their most controllable elements, many kinds of clusters will be beyond the reach of the export control policy. The policy is likely to "leak." This will be a fact of life in today's world. We discuss "leakage" of the policy in the final chapter.

Establishing a Lower Bound

Any of the factors mentioned above—specialized software, hardware, or expertise—may be taken into account when considering a system's end-user attainable performance. Assuming that the end-user attainable performance of systems can be determined, how should the lower bound of controllability be determined? As before, the lower bound of controllability should be the greater of: (a) the performance of systems widely available from countries not supporting U.S. export control policies, and (b) the end-user attainable performance of uncontrollable platforms.

Table 3–6 offers a classification of computing systems. The end-user attainable performance figures are those of systems introduced in 1997. Each category has different controllability features; controllability decreases as one moves down the table.

Type	*Units installed*	*Price*	*End-user attainable performance*
Multi-rack HPC systems	100s	$750K–10s of millions	20K+ Mtops
High-end rack servers	1000s	$85K–1 million	7K–20K Mtops
High-end deskside servers	1000s	$90–600K	7K–11K Mtops
Mid-range deskside servers	10,000s	$30–250K	800–4600 Mtops
UNIX/RISC workstations	100,000s	$10–25K	300–2000 Mtops
Windows NT/Intel servers	100,000s	$3–25K	200–800 Mtops
Laptops, uni-processor PCs	10s of millions	$1–5K	200–350 Mtops

Table 3–6. Categories of computing platforms (4Q 1997)

The classification presented here, when used in conjunction with the concept of end-user attainable performance, makes it possible to use a performance-based metric to distinguish more precisely than before systems that are more controllable from those that are less controllable. For example, a control threshold set at 4–5000 Mtops would permit the export of mid-range systems under general license while subjecting rack-based systems, even small configuration racks, to extra licensing considerations. A threshold set at 2500 Mtops would restrict most RISC-based desk-side systems, while permitting the export of desktop machines under general license.

The determination of which category of systems is, in fact, uncontrollable depends on the amount of resources industry and government are willing to bring to bear on monitoring and control measures. Suppose the lower bound of controllability were determined to lie with the deskside machines. Figure 3–7 shows the anticipated performance of such systems through the turn of the century. The mid-range systems have been plotted one year following their introduction since the installed base grows rapidly, but not instantly. Rack-based systems, defining the line of uncontrollable performance under the current control regime, have been plotted two years after introduction to account for the slower growth of their installed base. The systems contributing to the high-end deskside systems have been plotted either one or two years after introduction, depending on the rate at which their installed base grows.

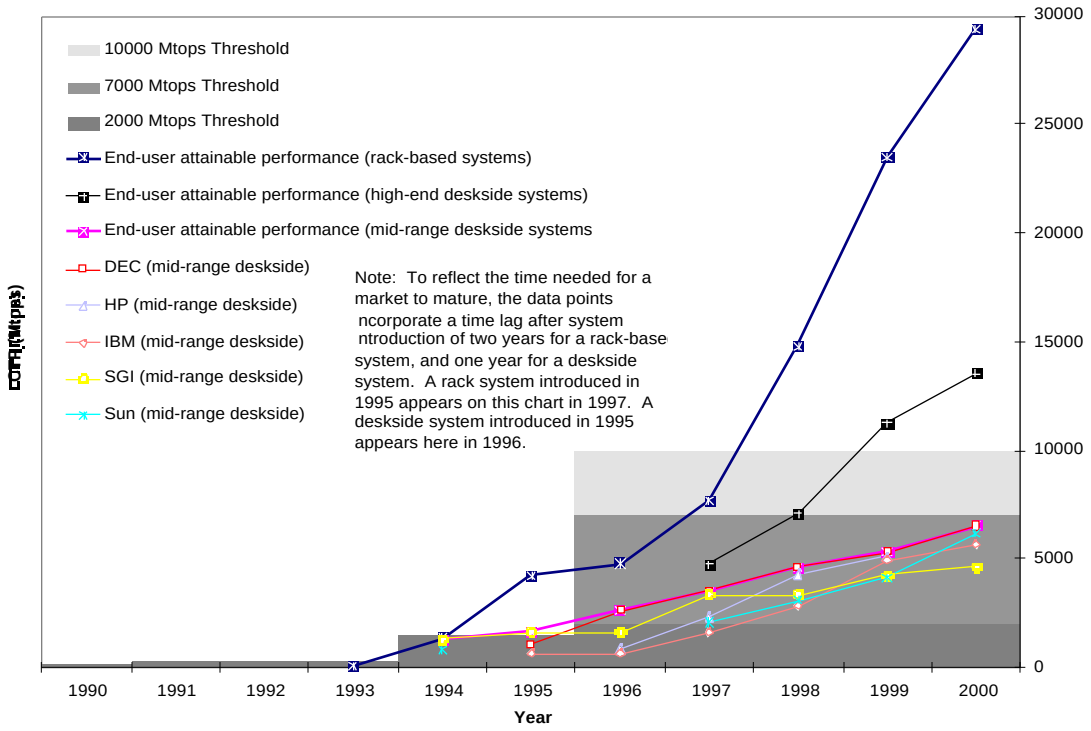

Figure 3–7. End-user attainable performance of categories of HPC systems

Projections of when new products will be introduced and at what performance are based on two assumptions: First, the maximum number of processors in a deskside box will remain constant. Second, the time lag between volume manufacture of a new microprocessor and its appearance in deskside systems will remain the same as at present. Under these assumptions, the end-user attainable performance is likely to approach 4300 Mtops in 1998, 5300 Mtops in 1999, and 6300 Mtops in 2000. We should note that this trend line is consistent with the administration's decision in 1995 to set a control threshold of 2000 Mtops for Tier 3 military end-users. This was approximately the performance of deskside mid-range systems in 1995/1996.

While this chart reflects approximately linear increases in mid-range system performance through the end of the century, there is a strong possibility that their performance will increase sharply in 2000–2001 (factoring in the one-year time lag). As new generations of processors, principally Intel's Merced, are incorporated into servers, mid-range systems with four or eight processors may have CTP values well above 15,000 Mtops. These systems will constitute the "sweet spot" of the mid-range market and have installed bases on the order of tens or even hundreds of thousands of units.

Unlike in past years, control thresholds that discriminate between categories of computer systems do not discriminate between HPC vendors. In today's HPC market there are no major HPC vendors whose sole business is the highest-end machines. Cray was acquired by Silicon Graphics, Inc.; and Convex by Hewlett-Packard. Recently, Compaq announced its acquisition of Digital Equipment Corporation. Each of the remaining U.S. HPC vendors now offers a broad product line "from desktops to teraflops" based on the same components. Each vendor has small boxes targeting the very large lower-end HPC markets. Moreover, Figure 3–7 indicates that the vendor offerings in these markets have quite comparable performances as measured by the CTP, so each is competitive in export markets.

Conclusions

In the preceding discussion we have tried to answer the question, "What is the lower bound of a viable control threshold?" A control threshold that seeks to control the performance available to an end-user by controlling the performance of shipped configurations, works well when systems are not easily upgradeable by the end-user. A dominant characteristic of most of today's high-performance computing systems is that they *are* easily scalable, up to a point. Preventing end-users from upgrading systems in the field requires a great deal of post-installation monitoring by vendors and U.S. Government officials. Even so, it is difficult to know with certainty the extent to which end-users are upgrading. As the number of installed systems grows, so does the difficulty.

An alternative control regime would call for export licensing decisions based not on the configuration performance of a system being shipped, but on the *end-user attainable* performance of that system. The attainable performance is defined as the performance of the largest individual, tightly coupled configuration an end-user can assemble without vendor support using only the hardware and software provided with lesser configurations. By "lesser configuration" we usually mean a configuration that can be exported under existing export control policies. If end-user attainable performance is used as a factor in licensing decisions, it is much less necessary to determine whether or not end-users are upgrading systems. This possibility will have been accounted for in the licensing process before the system is shipped.

- Under current policy mechanisms, the level of uncontrollable performance will rise to nearly 15,000 Mtops in 1998, over 23,000 Mtops in 1999, and approximately 29,000 Mtops in the year 2000.
- If a control mechanism based on end-user attainable performance is used, and deskside mid-range systems are considered uncontrollable, a lower bound of controllability would reach 4600 Mtops in 1998, 5300 Mtops in 1999, and 6500 Mtops in the year 2000. This number is likely to increase sharply in 2000–2001 as mid-range systems based on new microprocessors like Intel's Merced and IBM's G4+ enter the market. Configurations with as few as four CPUs, each with a CTP of 4–7000 Mtops, could have a performance of 15–25 Mtops. Systems with 4 to 8 processors, shipped in volumes of tens or hundreds of thousands of units, will constitute the "sweet spot" of the mid-range market.
- Policy makers will need to do the following:
  (a) Decide whether to base licensing decisions on configuration performance, or on end-user attainable performance.
  (b) Determine what systems are, in fact, uncontrollable, given the resources government and industry are willing to use to enforce the policy. The use of end-user attainable performance as the basis for licensing decisions supports distinctions between classes of machines that are not possible under the current policy.
  (c) Employ a two-year, one-year, or no-year lag in plotting system performance data on graphs such as those presented in this chapter. We have used a two-year lag for rack-based systems and a one-year lag for desk-side systems.
- Foreign indigenous HPC systems continue to have very small installed bases and modest performance relative to U.S. mid- and high-end HPC systems. They do not have a major impact on the determination of a lower bound of controllability.

References

[1] Goodman, S. E., P. Wolcott, and G. Burkhart, *Building on the Basics: An Examination of High-Performance Computing Export Control Policy in the 1990s*, Center for International Security and Arms Control, Stanford University, Stanford, CA, 1995.

[2] Goodman, S., P. Wolcott, and G. Burkhart, *Executive Briefing: An Examination of High-Performance Computing Export Control Policy in the 1990s*, IEEE Computer Society Press, Los Alamitos, 1996.

[3] "NT Has Cut Into UNIX Workstation Mkt Says Dataquest," *HPCWire*, Jun 20, 1997 (Item 11401).

[4] Gomes, L., "Microsoft, Intel to Launch Plan to Grab Bigger Piece of Market for Workstations," *Wall Street Journal*, Mar 6, 1998, p. B3.

[5] Sterling, T., et al., Findings of the first NASA workshop on Beowulf-class clustered computing, Oct. 22–23, 1997, Pasadena, CA, Preliminary Report, Nov. 11, 1997, Jet Propulsion Laboratory.

Chapter 4: Applications of National Security Interest

Introduction

Choosing an Upper Bound for a Control Threshold

The first basic premise of the HPC export control policy requires the existence of problems of great national security importance that require high-performance computing for their solution [1]. If such applications do not exist, it is difficult to justify HPC export controls. When they do exist, they can be used to set an upper bound on the performance level at which a viable control threshold can be established. Just as the lower bound of controllability can be used to determine a performance level below which a threshold is not viable, the performance requirements of applications of national security interest can be used to determine an upper bound above which a threshold should not be set. This chapter examines such applications and evaluates their need for high-performance computing to determine an upper bound.

Chapter 3 illustrated that the lower bound is in part a function of policy makers' determination of what levels of performance are controllable. Similarly, the upper bound is in part a function of what the national security community feels are applications "of great national security importance." There are two principal positions that can be taken.

The first position is that the set of applications of great national security importance is broad and deep, distributed across the entire spectrum of computing performance. Under this view, any level of computing performance can be useful to an adversary, and export controls are beneficial whenever they create an obstacle, however small, to the adversary's pursuit of national security applications. If export controls force an adversary to spend a few more days or months and a bit more money on a development effort, then an important national security objective has been achieved. This position leads to a "control what you can" approach to export controls. In this case, the upper bound collapses onto to the lower bound, forcing the threshold to track the lower bound.

The second position is that not all applications being pursued by the national security community are of great national security importance, nor can all applications be pursued effectively by all foreign entities of national security concern. The set of applications to be protected is more limited than in the first position, and allows for the possibility of an upper bound that lies above the lower bound, creating a range of performance levels at which a control threshold is viable. The set of applications may be limited in a number of ways. A distinction may be made between applications that are of great national security priority that yield distinct military or diplomatic advantage, and those that are merely of national security

interest. Performance levels at which there are particularly large numbers of applications may influence the choice of an upper bound. The set of applications that are to be protected may be chosen based on an analysis of applications that are critical to foreign entities' national security objectives. The upper bound may also take into account the willingness and ability of individual countries to pursue particular applications in ways that threaten U.S. national security interests, leading to different thresholds for different countries.

The objective of this chapter is not to decide among the options just described. Instead, it tries to provide policy makers with an understanding of the kinds of applications that may be pursued at various performance levels, and of the trends in U.S. practitioners' use of the technology that are likely to have the greatest impact on the export control policy in the future. In particular, the chapter does not try to establish which of the applications are of greatest national security importance and must be protected by HPC export controls. That is a decision that can only be made by the national security community. It is important that this decision be made. If an application area lacks a constituency willing to defend it in the public arena, it is difficult to argue that it should be a factor in setting export control policy.

Selection of Applications

The Department of Defense (DoD) has identified ten Computational Technology Areas (CTA), listed in Table 4–1. Through the High Performance Computing Modernization Program (HPCMP), DoD provides funding for the acquisition and operation of high-performance computing centers to pursue applications in these areas. Nuclear, cryptographic and intelligence are also areas with programs that fund applications that use high-performance computing.

CFD	Computational Fluid Dynamics	CEN	Computational Electronics and Nanoelectronics
CSM	Computational Structural Mechanics	CCM	Computational Chemistry and Materials Science
CWO	Climate/Weather/Ocean Modeling and Simulation	FMS	Forces Modeling and Simulation / C4I
EQM	Environmental Quality Modeling and Simulation	IMT	Integrated Modeling and Test Environments
CEA	Computational Electromagnetics and Acoustics	SIP	Signal/Image Processing

Table 4–1. DoD Computational Technology Areas (CTA)

The applications chosen for this study are based on those application areas that are currently being funded by one or more agencies of the U.S. national security community. We have tried to focus on applications that lie above the lower bounds discussed in Chapter 3. However, it is often quite useful to include applications that in the past were performed on supercomputers, even though those levels of performance may not be controllable today.

This chapter summarizes key findings for upper bound applications. After providing a general overview of computational techniques and their impact on the choice of computer hardware, the chapter discusses specific findings in various application areas. Areas that have an operational need for HPC are discussed first, followed by those areas that apply HPC to research and development. The computational requirements of the former are "hard" in the sense that computational results must be obtained within a specific time period; receiving the results late is of no practical value. Obtaining research and development results quickly provides a competitive edge in the race to new technologies; receiving them later means the new technologies come online slower and/or at greater cost. Thus, HPC export controls may

play a larger role in operational applications than for research and development applications. Finally, the chapter concludes with some options for policy makers to consider with regard to upper-bound applications.

Key application findings:

- Upper-bound applications of potential national security interest that require high levels of computing continue to exist and will exist for the foreseeable future.
- Computational science is playing an increasingly important role in research, development, and testing.
- Upper bound applications continue to demand higher performance platforms.
- Scientific applications in all computational areas are continuing to transition from parallel vector platforms (PVP) to massively parallel platforms (MPP).
- The Message Passing Interface (MPI), a library package that facilitates writing parallel applications, is supporting application development across a wide range of machines.
- For some communities, the development of parallel codes is beginning to "come naturally."
- Parallel platforms are being used for production work in some application areas, including nuclear and ocean modeling applications.
- Memory size and bandwidth remain bottlenecks in many application areas.

Upper–Bound Applications Exist

The first criterion for classification as an upper-bound application is national security interest, as explained above. The second criterion is the application's computing needs. Does it require HPC for its solution? In some cases, the answer is yes; the application is not solvable on lesser machines. In other cases, the application is solvable to at least some degree on lesser machines. The question then becomes the degree to which HPC provides an edge through allowing completion in a much shorter time frame or better understanding through multiple experiments in the same time frame.

This study examined applications in all the areas listed above. A large number of applications satisfy the computing criterion of upper-bound applications, and would seem to be of national security interest. Figure 4–1 shows a sampling of these. As detailed in later sections of this chapter, these applications continue to demand computing performance above the lower bound of controllability, and in most cases will continue to do so for the foreseeable future.

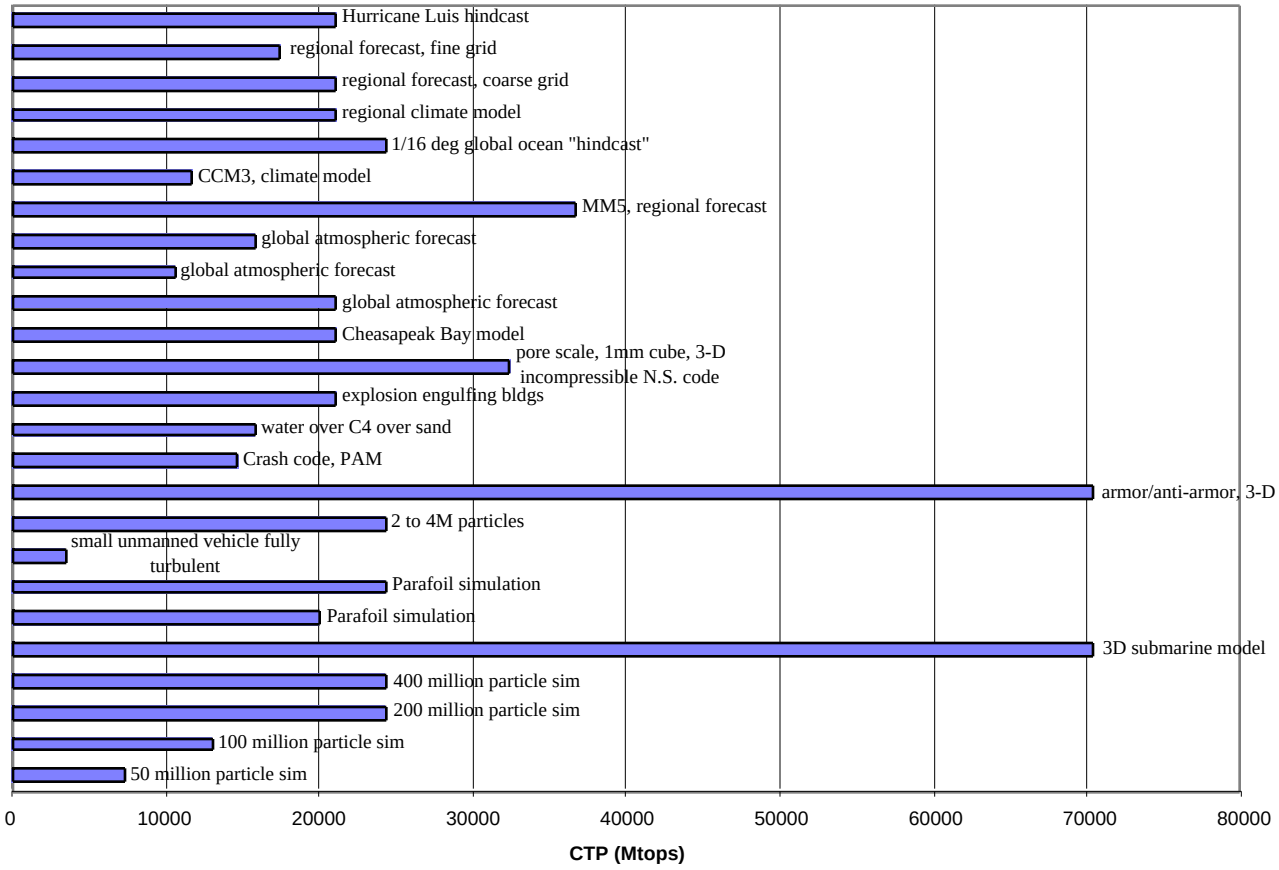

Figure 4–1. CTP of sample upper-bound applications

Role of Computational Science

Computational science uses simulation techniques to model physical systems. Simulation is playing an increasingly important role in research and development in almost every area of inquiry. Simulation:

- reduces dangers that exist in the development and testing of prototypes
- supports the investigation of phenomena that cannot be observed
- reduces costs by reducing the number of prototypes that need to be built and tested
- increases the effectiveness of an experimental program while reducing the number of actual experiments needed

In some areas, computer simulation is already tightly integrated into the development and testing cycle. In other areas, simulation techniques are still being developed. In all areas, simulation techniques will see continued improvement as better computer hardware becomes available. This will allow experiments that are not possible today due to constraints of memory and/or processors. Better hardware will also support parametric studies, that is, allow the researcher to run a suite of tests varying one or more of the parameters on each run.

In some areas, high-performance computing directly supports operational needs. Weather forecasting is a clear example of this. Improved hardware leads directly to more accurate global forecasts. Better global forecasts lead to more accurate regional forecasts. Both are directly dependent on the computing resources available.

Demand for Greater Resources

Three factors drive computational science to demand greater computing resources. First, increasing the accuracy, or resolution, of an application generally allows more detail to become apparent in the results. This detail serves as further validation of the simulation results and provides greater insight into the process being modeled. Second, increasing the scale of a simulation allows a larger problem to be solved, for example studying a full-size submarine complete with all control surfaces. Larger problems thus become a way to understand mechanisms that previously could only be studied through actual experiments, if at all. Third, using models with greater fidelity allows more accurate simulation of physical phenomena. Examples include using full physics rather than reduced physics in nuclear applications and solving the full Navier-Stokes equations in fluid dynamics problems.

All three—increased resolution, increased scale, and improved fidelity—offer the possibility of a quantum leap in understanding a problem domain. Examples include studies of the dynamics among atoms and molecules of an explosive material as the explosive shock wave proceeds, or eddies that form around mid-ocean currents. Ideally, a good simulation matches the results of actual observation and provides more insight into the problem than experiment alone.

Increased resolution and scale generally require a greater than linear growth in computational resources. Consider, for example, doubling the resolution of a three-dimensional simulation. This requires doubling the computation in each of the three dimensions, a factor of eight times the original. And this understates the usual situation, as doubling the resolution often requires increases in the number of timesteps over which the computation is run.

Transition from Parallel-Vector to Massively Parallel Platforms

Scientific applications in all computational areas are in transition from PVP to MPP platforms. Hardware developments and the demands—both price and performance—of the commercial marketplace are driving this transition. These trends have already been discussed in detail in Chapter 2. Computational scientists in some application areas made early moves to MPP platforms to gain more memory and/or processor resources than were available in the PVP platforms. However, all areas of computational science are now faced with the need to transition to MPP platforms.

The development of platform-independent message-passing systems, such as PVM [2] and p4 [3], supported the first wave of ports from traditional PVP machines to MPP machines. Platform independence assured the scientists their work would survive the vagaries of a marketplace that has seen the demise of a number of MPP manufacturers. The development of the Message Passing Interface [4] standard in 1994 and its wide support among computer manufacturers has accelerated the move to MPP platforms by scientists.

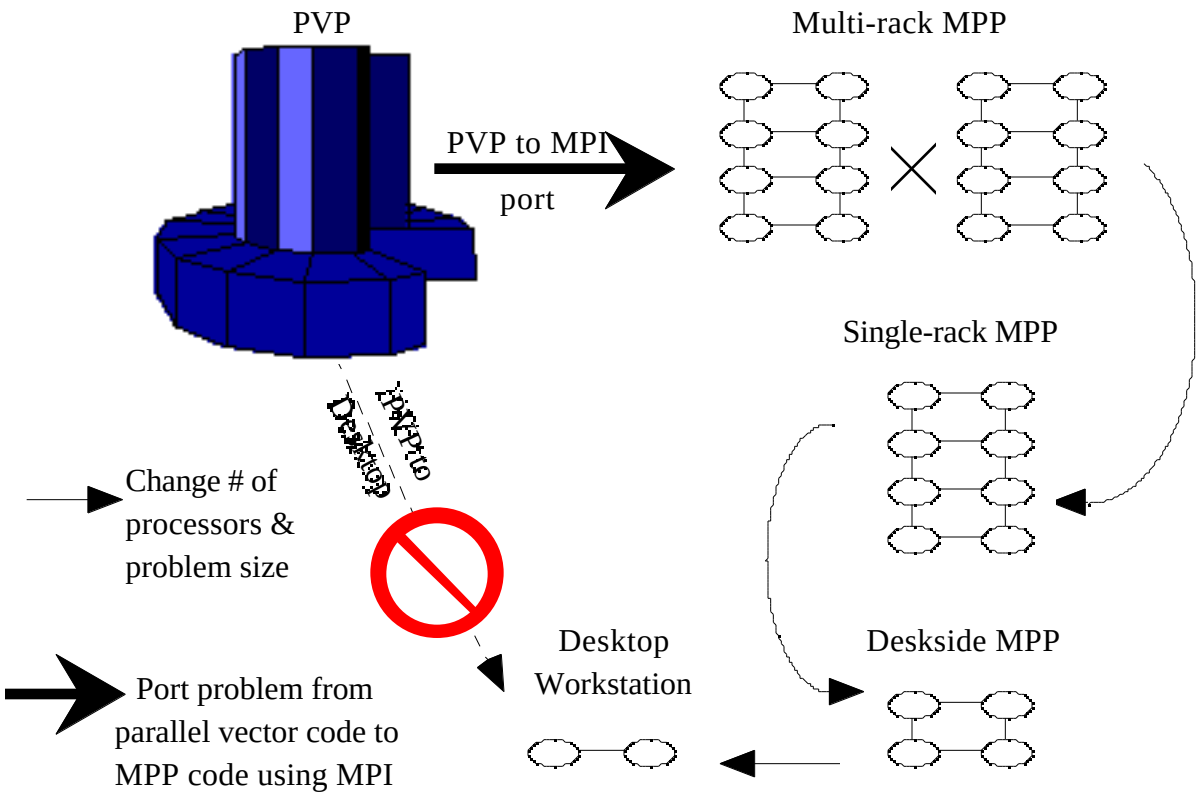

Figure 4–2. Conversion path for vector to parallel code

The move to MPI has resulted in applications that run effectively on a broad range of computing platforms and problem sizes, in contrast to PVP codes that rely heavily on vector operations for efficient execution. Figure 4–2 illustrates the transitions possible. From a code optimized for vector operations, it is not easy to obtain a code that runs well on a desktop workstation, or any strictly parallel machine. However, once the effort has been made to port the PVP code to the MPP platform using MPI, it is an easy transition to run the same code on smaller or larger MPP systems by simply specifying the number of processors to use. Since all the computer manufacturers provide support for MPI, the code is now portable across an even wider variety of machines.

The smaller systems generally cannot run the same problems as the larger systems due to memory and/or time constraints. However, being able to run the code on smaller systems is a great boon to development. Scientists are able to test their code on the smaller, more accessible, systems. This shortens the development cycle and makes it possible for researchers without direct access to the larger platforms to be involved in the development and verification of new algorithms.

Parallel Coding Starting to "Come Naturally"

Production codes are now entering their second generation on MPP platforms in some application areas. Examples include the nuclear, ocean modeling, and particle dynamics

fields. High memory requirements are a common characteristic of these applications. Starting roughly with the Thinking Machines CM2, MPP machines began offering greater memory sizes than were available on PVP machines. This offered the opportunity to run larger simulations if the researcher was willing to put in the effort to port the code. Now, these researchers are entering second—and third—generation revisions of their codes, and have production as well as research and development experience with MPP machines. Thus, there is a growing cadre of researchers for whom parallel codes have become the ordinary way of working.

Parallel Platforms Used for Production Work

Large MPP platforms are being used for production work as well as for research and development. In general, this use is in application areas that were among the earliest to make the move from the PVP platforms. Examples include nuclear simulations, particle simulations, cryptographic work, and ocean modeling.

Memory Size and Bandwidth Remain Bottlenecks

Many application areas are moving to MPP platforms primarily to take advantage of increased aggregate memory sizes. The memory to CPU bandwidth then becomes a problem on the commodity RISC-based processors used in MPP systems. Many problems become sensitive to the size of the cache and often require work to optimize cache performance.

It should be noted that increased memory sizes on today's standard workstations allow execution of applications that were on supercomputer-class systems a few years ago. These problems were often run as single-CPU jobs on the PVP machines, as memory size is the primary requirement. On workstations, they can require long execution times and large disk storage space. However, given the wait time for sufficient resources on high-end computers, it is often quicker to use a workstation.

Computing Platforms and Scientific Applications

This section provides a brief discussion of key factors that affect the performance of an application on a parallel computer. A great deal of detail has been omitted to make the discussion accessible. Details that affect specific application areas are covered later in this chapter.

Determinants of Performance

Most scientific applications make use of grids to subdivide the problem space. Grids can be unstructured or structured. Unstructured grids are typically triangular for 2-D problems and tetrahedral for 3-D problems. Finite element (FE) solution techniques are used with unstructured grids. Structured grids are "logically" Cartesian or rectangular in form. Finite difference (FD) or finite volume (FV) solution techniques are used with structured grids. Figure 4–3 gives some two dimensional examples of solution grids.

In parallel processing, the grids are divided in some fashion across two or more processes, a step known as "domain decomposition." Each process performs the computations on its portion of the grid. At various times during the computation, it is normally necessary for the processes to share data from their portion of the grid with the processes handling adjacent portions of the grid. When executing on a parallel machine that has multiple processors, each process is typically assigned to one processor. This spreads the computational work over as many processors as are available.

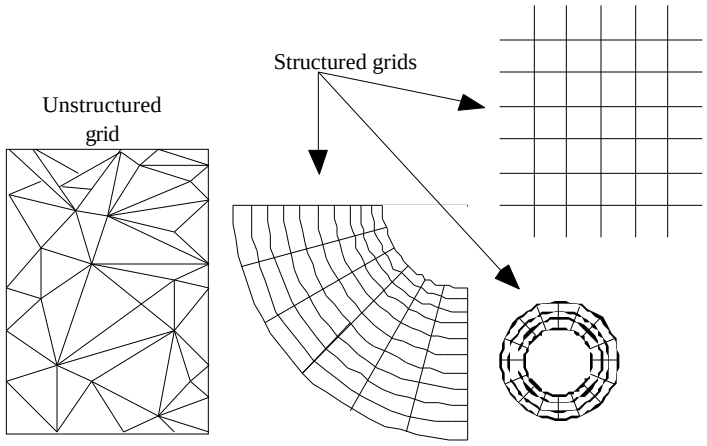

Figure 4–3. Grid examples

The total execution time (T) of a given processor is the sum of the times spent in each of three states:

$$T = T_{comp} + T_{comm} + T_{idle}$$

T_{comp} is a function of the number of processes executing on the processor. For each process, T_{comp} is further dependent on the grid size, the number of timesteps, the number and types of operations executed in each time step, and the time needed to execute an operation. More complex (and realistic) performance models take into account differences in the time needed to execute an operation depending on where data are located (e.g., main memory or secondary storage).

T_{comm} is the amount of time the processor spends on exchanging data with other processors. This is dependent on the platform and the architecture—shared memory vs. distributed memory.

T_{idle} is the time a processor spends waiting for data. Generally, this will be while waiting for another processor to complete a task and send the needed data, or merely waiting for requested data to arrive at the CPU.

If execution time is measured from the instant the first processor begins until the last processor ceases activity, then the total execution time for each processor will be the same, although the percentage of time spent in each state is likely to vary.

If only compute time directly advances the state of the computation and the amount of computation necessary for the execution of a particular model and problem size is relatively constant, then performance can best be improved by minimizing the time the processors spend communicating or being idle. The communication requirements strongly depend on the domain decomposition used—how a problem is divided across the multiple processes and how the processes are divided across the machine's available processors. Different decompositions of the same problem onto the same set of processors can have widely different communication patterns. One goal of problem decomposition is to maximize the amount of computation performed relative to the amount of communication.

Applications and the algorithms they use can differ considerably in how easy it is to find decompositions with good computation-to-communication ratios. So-called "embarrassingly parallel" applications are those that lend themselves to decompositions in which a great deal of computation can be performed with very little communication between processors. In such cases, the execution of one task depends very little, or not at all, on data values generated by

other tasks. Often, applications with this characteristic involve solving the same problem repeatedly, with different and independent data sets, or solving a number of different problems concurrently.

Shared-Memory Multiprocessor Systems

In shared-memory multiprocessors, all processors share a common memory. Systems in this category include symmetrical multiprocessors (SMP) such as the Silicon Graphics PowerChallenge, Sun Microsystems' Enterprise 10000, and Cray's traditional vector-pipelined multiprocessors.

In the abstract, shared-memory systems have no communication costs per se. Accessing the data values of neighboring grid points involves merely reading the appropriate memory locations. Most shared-memory systems built to date have been designed to provide uniform access time to main memory, regardless of memory address or which processor is making the request.

In practice, the picture is more complex. First, processors must coordinate their actions. A processor cannot access a memory location whose content is determined by another processor until this second processor has completed the calculation and written the new value. Such synchronization points are established by the application. The application must balance the workload of each processor so they arrive at the synchronization points together, or nearly so. Time spent waiting at a synchronization point is time spent not advancing the computation.

Second, as described in Chapter 2 (Interconnect and memory subsystems) today's machines include caches as part of the memory design. On shared-memory architectures, caches necessarily involve maintaining duplicate copies of data elements. Maintaining the consistency of the copies is a great system-design challenge. Fortunately, managing cache memory is not (explicitly) a concern of the programmer.

The existence of cache memory means that memory access is not uniform. Applications or algorithms with high degrees of data locality (a great deal of computation is done per data element and/or data elements are used exclusively by a single processor) make good use of cache memory. Applications which involve a great deal of sharing of data elements among processors or little computation per data element are not likely to use cache as effectively and consequently have lower overall performance.

Third, when the processors in a shared-memory multiprocessor access memory, they may interfere with each other's efforts. Memory contention can arise in two areas: contention for memory ports and contention for the communications medium linking shared memory with the processors. Memory may be designed to give simultaneous access to a fixed and limited number of requests. Usually, separate requests that involve separate banks of memory and separate memory ports can be serviced simultaneously. If, however, two requests contend for the same port simultaneously, one request must be delayed while the other is being serviced.

A shared communications medium between memory and processors can also be a bottleneck to system performance. Internal busses, such as those used in many SMP systems can become saturated when many processors are trying to move data to or from memory simultaneously. Under such circumstances, adding processors to a system does not increase aggregate performance. This is one of the main reasons why maximum configuration SMPs are rarely purchased and used effectively.

Some systems use switched interconnects rather than a shared medium between processors and memory. While more expensive, switched interconnects have the advantage that processors need not contend with each other for access. Contention for memory banks may still be a problem.

Distributed–Memory Multiprocessor Systems

Distributed memory machines can overcome the problems of scale that shared memory systems suffer. By placing memory with each processor, then providing a means for the processors to communicate over a communications network, very large processor counts and aggregate memory sizes can be achieved. In principle, there are no limits to the number of processors or the total memory, other than the cost of constructing the system. The SGI Origin2000, Cray T3E, IBM RS/6000 SP, and clusters of workstations are examples of distributed memory architectures.

For distributed memory machines, the communication time, T_{comm}, is the amount of time a process spends sending messages to other processes. For a given communication (sending or receiving a message), communication time breaks into two components: latency and transmission time. Latency—the time to establish a connection or perform a context switch—is independent of the message size. It is dependent on the distance between processors and the communication channel. The distinction between closely coupled and loosely coupled machines lies primarily in the latency costs, as explained in Chapter 2. Transmission time is a function of the communication bandwidth—the amount of data that can be sent each second—and the volume of data in a message.

For applications that are not embarrassingly parallel, the ratio of computation to communication can sometimes be improved by increasing the granularity of the problem, or the portion of the problem handled by a single processor. As an illustration, suppose that a problem involved a two-dimensional grid of data points. Updating the value at each point involves obtaining the values of each of the four neighboring points. If a different processor is used to update each grid point, four communications are required to obtain the neighbor's data values. If, however, each processor were responsible for updating 8 x 8 = 64 grid points, 64 computations would be needed (one for each grid point), but only 8 x 4 = 32 communications. No communication is involved in accessing data from a neighboring grid point when the data values are stored in the memory of the same processor. Communication occurs only for fetching the data values of grid points lying outside the perimeter of the 64-point grid, since these are located in the memory of other processors. In a two-dimensional grid, the amount of computation is proportional to the number of points in a sub-grid (the area), while the amount of communication is proportional only to the number of points on the perimeter (the surface). Further, the messages often can be batched along an edge. Following the same example, the one processor can request the eight values for one side of its grid in a single message, and can send its own eight values on the same edge in one message. The amount of data sent will be the same, requiring the same bandwidth; however, the latency will be reduced since the start-up cost needs to be paid for only one message, not eight.

While increasing the granularity for a fixed-sized problem improves the computation-to-communications ratio, it also reduces the amount of parallelism brought to bear. (The extreme case would be putting the entire problem on a single processor: excellent computation-to-communications ratio, but no parallelism.) For a fixed number of processors, however, the computation-to-communication ratio can be improved by increasing the size of the problem and, correspondingly, the number of grid points handled by a single processor. This assumes, however, that the processor's memory is large enough to hold the data for all the points.

Numerical Weather Forecasting

Introduction

Weather conditions have always played a vital role in a host of military and civilian activities. The ability to predict in advance the weather conditions in a particular area can provide a substantial advantage to those planning and executing military operations. For this reason, in spite of its obviously beneficial civilian applications, weather forecasting can be considered an application of national security significance. Weather forecasting imposes *operational* constraints, due to the need for timely delivery of forecasts to customers. This operational demand sets weather forecasting apart from most of the other application areas covered in this report.

Since the dawn of the computer age, the most advanced computing systems have been brought to bear on computationally demanding weather forecasting problems. Weather forecasters have been among the leading adopters of new supercomputing technology. While this tradition continues to the present, weather forecasters have been challenged by the broadening base of computing platforms that, in principle, can be used in weather forecasting. Besides the well-established vector-pipeline systems, practitioners have other options, including massively parallel and symmetrical multiprocessor systems.

Numerical weather forecasting was identified in *Building on the Basics* as a crucial upper-bound application. This section examines the computational nature of numerical weather forecasting in much greater detail than did the previous study, with the goal of understanding to what degree such applications can be usefully executed on a variety of system architectures, principally vector-pipeline and massively parallel processors. Ultimately, the goal is to understand the degree to which forecasting of a nature useful to military operations can be performed on the uncontrollable computer technology of today and tomorrow.

This section starts by examining the relationship between the computational nature of the application and the major high-performance computer architectures currently available. The next subsection includes a discussion of empirical results obtained from leading forecasting centers in the United States and Europe regarding the use of various platforms to run their atmospheric models. Then we discuss global and regional ocean forecasts and their similarities and differences from atmospheric models. Finally, we conclude with a brief discussion of future trends in numerical weather forecasting and computational platforms, and alternative sources of competitive advantage for U.S. practitioners.

The Computational Nature of Numerical Weather Forecasting

The state of the weather at any given point in time can be defined by describing the atmospheric conditions at each point in the atmosphere. These conditions can be characterized by several variables, including temperature, pressure, density, and velocity. Over time, these conditions change. How they change is represented mathematically by a set of partial differential and other equations that describe the fluid dynamic behavior of the atmosphere and other physical processes in effect. The fluid dynamics equations include:

- conservation of momentum
- conservation of water substance[1]
- conservation of mass

[1] The conservation of water substance equations govern the properties of water in various states: vapor, cloud, rain, hail, snow.

- conservation of energy
- state (pressure) equation

Equations for physical processes describe such phenomena as radiation, cloud formation, convection, and precipitation. Depending on the scale of the model, additional equations describing localized turbulence (e.g., severe storms) may be used.

Collectively, these equations constitute an atmospheric model. In principle, if the state of the weather at a given time is known, the state of the weather at a future time can be determined by performing a time integration of the atmospheric model based on these initial conditions.

The basic equations and the atmosphere itself are continuous. However, the atmospheric models approximate the actual conditions by using a collection of discrete grid points to represent the atmospheric conditions at a finite number of locations in the atmosphere. For each grid point, values are stored for each of the relevant variables (e.g., temperature and pressure).

Future weather conditions are determined by applying discrete versions of the model equations repeatedly to the grid points. Each application of the equations updates the variables for each grid point and advances the model one timestep into the future. If, for example, a single time step were 2 minutes, a 24-hour forecast would require 24 hours/2 minutes = 720 applications (iterations) of the model's equations to each grid point.[2]

The fluid dynamics of the atmosphere are typically modeled using either a finite difference or spectral transform method. In the finite difference methods, variables at a given grid point are updated each time step with new values computed by applying a function to the variables of neighboring grid points. Then the new values are exchanged with neighboring grid points.

The spectral transform methods are global spherical harmonic transforms and offer good resolution for a given amount of computational effort, particularly when a spherical geometry (like the earth) is involved.[3] The drawback, however, is that the amount of computation necessary in spectral methods grows more quickly than in finite difference methods as the resolution increases. Practitioners are divided on which method is better.

Numerical weather prediction models can also vary in the scale of the model, the length of the forecast, the physical phenomena to be modeled, and the degree of resolution desired. The models vary in the number of grid points used, the number of time-steps taken, and the equations employed. In the case of spectral methods, the number of waves considered in the harmonics will also vary.[4]

Current models used in military weather forecasting can be classified as global, mesoscale (regional),[5] theater, or battlespace (storm-scale). The domain for these models ranges from the entire planet (global) to a 1000 x 1000 km region (battlespace). For each of these models, weather can be forecast for a variety of time periods into the future. Typical forecasts today are from 12 hours to 10 days.

---

[2] The naval models described below use timesteps of 440s for global models, 120s for regional models, and smaller time steps for each under certain circumstances, such as when there are high atmospheric winds or when higher resolution is needed for localized conditions.

[3] The basic grid point mesh has a rectangular solid geometry. The spherical geometry of the earth means that if the same number of grid points represents each latitude, the geographic distance between these points is less as you approach the North or South Pole. Such considerations introduce complexities into the finite difference algorithms.

[4] The resolution of models using spectral techniques is usually indicated by the convention *Tn*, where *n* is the number of harmonics considered. A model designated T42 has a lower resolution than one designated T170. The NOGAPS (Global) model used by FNMOC is designated T159.

[5] Mesoscale models typically model a domain with a scale of a few thousand kilometers. Such models are particularly useful for forecasting severe weather events, simulating the regional transport of air pollutants, etc. Mesoscale models are regional models, but the latter term is also used to describe extracts from global models.

Models can also be classified as hydrostatic or non-hydrostatic. Hydrostatic models are designed for use on weather systems in which the horizontal length scale is much greater than the vertical length scale. They assume that accelerations in the vertical direction are small, and consequently are not suitable for grid spacing below 5–10 km. Non-hydrostatic models, on the other hand, do not make this assumption and are often employed in systems that model, for example, thunderstorms or other phenomena on this scale. Current military models that emphasize global and regional forecasts are hydrostatic, but work is being done to develop non-hydrostatic models as well.

The resolution of the model depends on both the physical distance represented by two adjacent grid points, and the length of the time step. A model in which grid points represent physical coordinates 20 km apart will generally give more precise results than one in which grid points offer a resolution of 80 km. Similarly, shorter time steps are likely to yield better results than longer ones. The greater the resolution, the greater the fidelity of the model. For example, certain kinds of turbulence may be modeled with grid points at 1 km, but not 80 km, resolution. At the same time, increasing the number of grid points or decreasing the time step increases the amount of storage and computation needed.

The choice of grid size, resolution, and timestep involves balancing the type of forecast desired against the computational resources available and the operational time constraints of the forecast. To provide useful predictions, a model must run significantly faster than real time (e.g., a 24-hour forecast that takes 24 hours to compute is of no use.) For this reason, there are clear limits to the size of the model that can be run on a given platform in a given time window. On the other hand, there are points beyond which greater resolution is not needed, where practitioners feel a particular model size is sufficient for a desired forecast. For example, the "ultimate" domain size necessary for storm-scale prediction is a 1024 x 1024 x 32 grid [5].

Military Numerical Weather Forecasting

Military Numerical Weather Forecasting Models

Table 4–2 shows some of the models used by the Fleet Numerical Meteorology and Oceanography Center (FNMOC) for military numerical weather prediction as of the end of 1995 and expected models in use by 2001 [6]. The global models are currently their principal products. The right-hand column indicates the elapsed time needed to run the model.

Model Type	*Definition*	*Grid Resolution: 1995*	*Wall Time: 1995*	*Grid Resolution: 2001 (expected)*
Global Analysis	MVOI[6]	Horizontal: 80 km Vertical: 15 levels	10 min/run	Horizontal: 40 km Vertical: (as required)
Global Forecast	NOGAPS[7] 3.4	Horizontal: 80 km Vertical: 18 levels	30 min/forecast day 10 day forecast	Horizontal: 50 km Vertical: 36 levels
Regional 9000 km x 9000 km	NOGAPS Extracts	Same as Global Forecast		Same as Global Forecast
Theater 7000 km x 7000 km	NORAPS[8] 6.1 relocatable areas	Horizontal: 45 km Vertical: 36 levels	35 min/area	Horizontal: 18 km Vertical: 50 levels
Mesoscale 3500 km x 3500 km	NORAPS 6.1 2 areas NORAPS 6.1 2 areas	Horizontal: 15 km Vertical: 36 levels Horizontal: 20 km Vertical: 36 levels	60 min/area 75 min/area	Horizontal: 6 km Vertical: 50 levels
Battlespace 1000km x 1000km		N/A		Horizontal: 2 km Vertical: 50 levels

Table 4–2. Current and projected weather forecasting models used by FNMOC

To understand the significance of the wall time figures, it is important to understand the real-time constraints of a center like FNMOC. Weather forecasting is not a one-time computation. FNMOC's many customers require a steady stream of forecasts of varying duration. Given a finite number of CPU hours per day, the CPU time used to generate one forecast may take time away from another. For example, if a global forecast were to take 45 rather than 30 minutes per forecast day, one forecast might start encroaching on the time allocated for the next forecast. Either the number of forecasts, or the duration of each, would have to be reduced. For example, when the mesoscale model was run to support NATO operations in Bosnia, improving resolution from 20 km to 15 km doubled the run time. Better resolution would be possible computationally but impossible operationally on the current hardware. Operational considerations place strong constraints on the types of models that can be run with the available hardware.

The Traditional Environment for Military Numerical Weather Forecasting

The current state of military numerical weather forecasting in the United States is a result of a number of current and historical circumstances that may or may not characterize other countries' weather forecasting efforts.

- *Global forecasting.* Traditionally, FNMOC customers have desired global weather forecasts and regional forecasts based on the global models.
- *Demanding operational environment*. On the basis of a modest number of different atmospheric and oceanographic models, FNMOC provides approximately 48,000 distinct products to customers throughout the armed forces. These customers require a steady flow of forecasts, day after day, which they use in an operational context. Consequently,

[6] Multi-variate Optimum Interpolation. Used to provide the initial conditions for both the global and regional models (NOGAPS and NORAPS), but at different resolutions.
[7] Navy Operational Global Atmospheric Prediction System
[8] Navy Operational Regional Atmospheric Prediction System

FNMOC is a conservative computing center, unwilling to rely on unproven hardware, software, algorithms, or models.

- *Diverse customer base.* The customers for the 48,000 products use a wide array of technologies, and FNMOC needs to be able to deliver its products regardless of customer technology. A great deal of effort is spent making sure that the products are of a format and size that can be delivered to a given customer's platform in time for the results to be useful.
- *Reliance on vector-pipelined processors.* FNMOC for years has relied on traditional vector-pipelined processors to execute its model: a Control Data Corporation Cyber 205 and, in recent years, a Cray Research C90. There is a close relationship between the nature of the hardware available and the types of models and codes developed. Because FNMOC has always had vector-pipelined machines and, until recently, has been able to anticipate acquiring top-of-the-line vector-pipelined machines in the future, the research and development programs were oriented toward such platforms. The feedback loop was strong. As models and codes were developed that run well on vector-pipelined systems, the incentive to acquire another vector-pipelined system was strong. Such systems provided the greatest operational continuity, and were the lower-risk, lower-effort (although not necessarily lower-cost) acquisition option. This has changed with the increasing dominance of MPP systems. FNMOC is currently in an acquisition phase for an MPP system and is porting and testing a number of operational codes to use in evaluating MPP choices.

In summary, FNMOC operates on very demanding problems in a very demanding environment and is, for good reasons, rather conservative in its use of high-performance computing. The Center is not likely to dramatically change the models, code, or hardware/software platforms it uses until proven alternatives have been developed. FNMOC practitioners make it clear that their current models and codes do not (yet) run well on distributed-memory or non-vector-pipelined systems.[9] This is only partly a function of the nature of numerical weather forecasting (discussed in the next section). It is as much, or more, a function of the development and operational environments at FNMOC and the Naval Research Labs that develop the supporting models. Since the current generation of software tools is inadequate for porting "dusty deck, legacy" Cray code to massively parallel platforms, the human effort of porting the models is comparable to developing new models from scratch. Furthermore, there is, as described below, little guarantee that the end result would be superior to current practice. The risk to FNMOC operations of moving to dramatically new models and platforms prematurely is considerable. Consequently, those interested in understanding the suitability of non-traditional hardware/software platforms for (military) numerical weather forecasting should consider as well the experience of practitioners at other meteorological centers, who might be less constrained by operational realities and more eager to pioneer innovative approaches.

Computing Platforms and Numerical Weather Forecasting

This section examines the computational characteristics of numerical weather forecasting, with an emphasis on the problems of moving from vector platforms to massively parallel platforms. The first half of the section examines the problem in general. The second half explores a number of specific weather codes and the platform(s) they execute on.

[9] The Cray vector-pipelined machines reportedly provide a sustained processing rate of 40 percent of the peak performance on FNMOC codes. For example, the 8-processor C90 has a peak performance of 7.6 Gflops and provides just over 3 Gflops of sustained performance.

Numerical Weather Forecasting and Parallel Processing

Like many models describing physical phenomena, atmospheric models are highly parallel; however, they are not embarrassingly so. When a model is run, most of the equations are applied in the same way to each of the grid points. However, computation done at one grid point is not, in fact, independent of that done at another. The dependencies between grid points (described below) may require communications between processors as they exchange grid point values. The cost in CPU cycles of carrying out this sharing depends a great deal on the architecture and implementation of the hardware and system software.

There are three kinds of data dependencies between grid points that complicate the parallel processing of weather forecasting models. First, at the end of each time step, each grid point must exchange values with neighboring grid points. In higher-order finite difference schemes, the data needed to update a single grid point's variables are drawn not just from the nearest neighbor grid points, but also from points that are two or more points away. The spectral methods, an alternative to the finite difference methods, often involve transforms (between physical, Fourier, and spectral space) that have non-local communications patterns.

Second, global operations, such as computing the total mass of the atmosphere, are executed periodically. Good parallel algorithms exist for such computations.

Third, there are physical processes (e.g., precipitation, radiation, or the creation of a cloud) that are simulated using numerical methods that draw on data values from a number of different grid points. The total physics component of an atmospheric model can involve two dozen or more transfers of data per grid point per time step. However, the grid points involved in such a simulation usually lie along the same vertical column. Assigning grid points such that all grid points in the same vertical column are assigned to one processor reduces communications costs.

In many models, a so-called semi-Lagrangian transport (SLT) method is used to update the moisture field at grid points. This method computes the moisture content at a grid point (the "arrival point") by tracking the trajectory of a particle arriving at this grid point back to its point of origin in the current time step ("departure point"). The moisture field of the departure point is used in updating the moisture field at the arrival point. This computation can involve significant communications overhead if different processors are handling the arrival and departure points.

Overall performance of a model is strongly affected by load-balancing issues related to the physics processes. For example, radiation computations are performed only for grid points that are in sunlight [7].

Performance of Computing Platforms: Empirical Results

Since the first weather forecasting models were run on the ENIAC, meteorologists have consistently been among the first practitioners to make effective use of new generations of supercomputers. The IBM 360/195, the Cray 1, the Cyber 205, the Cray YMP and Cray C90 have all been important machines for weather forecasting [8]. In 1967, weather codes were adapted for the SOLOMON II parallel computer, but the amount of effort devoted to adapting weather models and algorithms to parallel systems, both shared memory and distributed memory, has grown dramatically only in the 1990s, as parallel systems have entered the mainstream of high-performance computing.

Meteorologists at leading research institutes and centers throughout the world have done a great deal of work in adapting weather codes for parallel platforms and obtaining empirical results. These efforts include:

- The Community Climate Model (CCM2) and Mesoscale Model (MM5), National Center for Atmospheric Research (NCAR) [9].

- The Parallel Community Climate Model (PCCM2), Argonne National Laboratory & Oak Ridge National Laboratory [7]
- The Integrated Forecasting System (IFS), European Centre for Medium-range Weather Forecasts (ECMWF) [10]
- Global weather forecasting model, National Meteorological Center [11].
- Atmospheric general circulation model, University of California, Los Angeles (UCLA) [12].
- Advanced Regional Prediction System (ARPS), Center for the Analysis and Prediction of Storms, University of Oklahoma [5,13,14].

The weather and climate forecasting problem domain currently embraces a significant number of different models, algorithms, implementations, and hardware platforms that make it difficult to provide highly detailed performance comparisons. At the same time, the general conclusions are remarkably consistent:

(1) PVP systems currently offer much higher levels of sustained performance relative to peak performance than MPP systems. The experiences of civil weather and climate forecasting practitioners have been consistent with those of FNMOC on this point. The reasons for the differences in efficiency between the two architectures are not fully understood, but appear to be related to high memory bandwidth requirements and code structures that are better suited to vector-pipelined architecture than RISC microprocessors [8].

(2) Porting models designed for PVP platforms to MPP platforms is very difficult. The effort involved is often comparable to that of developing the original model.

(3) Weather and climate forecasting problems are amenable to solution on MPP systems and do scale acceptably well, even though sustained performance, as a fraction of peak, is low.

(4) Obtaining acceptable performance on MPP platforms is very difficult. Internationally recognized numerical weather forecasting practitioners at leading research centers have worked diligently for years to make the transition. They have had some success, but not the resounding success that might have been hoped for in the early years of commercial MPP systems. The current state-of-the-art knowledge in applying parallel hardware and software to these problems is generally available, and still maturing.

(5) There is a good deal of interest in the use of SMP systems for regional weather forecasting. NCAR's regional weather forecasting model MM5 has been ported to SMP systems from all leading mid-range systems vendors. In many cases, the performance on small-configuration (usually single-processor) systems is comparable or better than small-configuration Cray systems such as the YMP and J90 [15]. Parallel versions are under development.

Ocean Forecasts

Global ocean forecasts have two purposes. The first is to predict the ocean currents. An ocean current has eddies and temperature variations that occur within and around the current flow. The temperature differences across the fronts and eddies associated with strong currents are important in submarine and anti-submarine warfare. Ocean currents also impact ship routing decisions, both lowering fuel costs and shortening transit times. This latter application of ocean modeling has civilian and military uses.

The second purpose of an ocean forecast is to provide the sea surface temperature field used as one of the major inputs to the atmospheric forecasts. An ocean model that can provide

more accurate surface temperatures will enable an atmospheric forecast with improved accuracy. To date, however, attempts to directly couple the two models to provide this input have not been successful. It is believed this is primarily due to the ocean component using a higher resolution than the atmospheric component. Higher resolution is required for the ocean because the physical scales of motion are smaller.

Ocean models take as their input atmospheric fields: winds, heat fluxes, and net evaporation/precipitation. These are typically provided by an atmospheric forecast, or by climatology based on observations. In an ocean simulation, or a climate study, atmospheric fields and perhaps an ocean climatology are the only input. In an ocean nowcast or forecast, additional information is required about the actual ocean state. This is typically provided by satellite. Sea surface temperature and sea surface height are the most useful data sets. For example, a cross-section of the Gulf Stream will show variations in ocean height of up to a meter and in surface temperature of several degrees as you move across the current. Altimeter-equipped satellites provide the sea surface height, which is generally a clearer signal of the underlying currents than sea surface temperature. These data come from the GEOSAT series of satellites from the United States. Both Japan and France have similar satellites. Existing satellite altimeters can measure differences in height of a few centimeters [16]

Ocean models are complicated by the need to accurately predict behavior below the surface, yet there is very little data about the current state of the ocean below the surface. Most of the observation-based data is about the surface or near surface. Statistical inference is used to initially move from the surface to the deep ocean, but observations are so rare in deep water that the statistics must be based primarily on earlier results from the ocean model. This places a premium on the accuracy of the ocean model. The ocean model also "dynamically interpolates" the oceanic data to provide global synoptic fields (a *nowcast*) and can be run forward in time to produce a forecast. A typical satellite altimeter will cover the same spot of the ocean only once every 17 days, so the ocean model fills in the gaps.

Ocean models are characterized by the size of the cells used to horizontally divide the ocean. Thus, a 1/4 degree ocean model uses cells that are 1/4 degree longitude by 1/4 degree latitude on each side, yielding a resolution of about 25 km on a side. The first global ocean current forecast system has 1/4 degree resolution and is run routinely by the Naval Research Laboratory (NRL) detachment at Stennis Space Center. It will enter "op-check" as a candidate Navy operational product at FNMOC as soon as there is an operational source for the processed altimeter fields required as input to the model. Regional current models are used for finer granularity in areas of particular interest, such as smaller seas, gulfs, and coastal areas. These models run at various finer resolutions that depend on the nature of the region covered and its operational importance. The global model can provide input to the regional models, as has been demonstrated by a regional model of the California current system [17].

The 1/4 degree global ocean model is too coarse to provide useful information about strong western boundary currents, such as the Gulf Stream, but it does provide accurate nowcasts and forecasts of large scale events, such as El Niño[10]. The memory capacity and speed of FNMOC's CRAY C90 computer system limit the resolution of the ocean model. Time slots on an operational forecasting computer are a limited resource, and FNMOC can not allocate more than about one hour per day to ocean forecasting.

NRL is developing more accurate models. In part, these are awaiting procurement decisions at FNMOC. However, even these models will not be able to produce sufficient accuracy within the one-hour time constraint to optimally model the ocean currents. A 1/32 degree (3 km resolution) ocean model is the goal. This will provide good coverage and detail in the open ocean, and provide much better input to the regional and coastal models. This

[10] A coupled ocean/atmosphere model would be required to forecast the onset of El Niño, but a stand-alone ocean model can forecast the oceanic effects of El Niño once it has started.

goal is still some years out. The current roadmap for FNMOC with regard to ocean current models is shown in Table 4–3.

Date	*Cell Resolution*		*Coverage*
present	1/4 degree	25 km	global
2001	1/8 degree 1/16 degree 1/32 degree	12-13 km 7 km 3 km	global Pacific Atlantic
~2007	1/32 degree	3 km	global

Table 4–3. Global ocean model projected deployment [16]

The vertical resolution in these models is six vertical layers. The size of a vertical layer depends on the density of the water, not its depth. The depth of the same density will change by several hundreds of meters as the grid crosses a current system or eddy. Using fixed-height layers would increase the vertical requirement to 30 or more layers to gain the same results.

Researchers at NRL are working with Fortran and MPI. In what has become a standard technique across disciplines, the communication code is concentrated within a few routines. This makes it possible to optimize the code for a particular platform by easily substituting the calls for a proprietary communication library, such as Cray's SHMEM.

It takes five to six years to move a model from a research tool to a production tool. In addition to the time needed to develop the improved models, the model must be verified. The verification stage requires leading-edge computing and significant chunks of time. The model is verified using "hindcasts"—executions of the model using historic input data and comparing the output with known currents from that time frame. For example, NRL tested a 1/16 degree (7 km) global ocean model in May–June 1997 on a 256-node T3E-900 (91,000 Mtops). It required 132,000 node hours, about 590 wall clock hours.[11] Such tests not only establish the validity of the new model, they improve the previous model. By comparing the results of two models at different resolutions, researchers are often able to make improvements on the lower resolution model. Thus, the research not only works toward new production models for the future, but also improves today's production codes.

Ocean models made the transition to MPP platforms much earlier than the atmospheric models. "Oceanographic codes, though less likely to have operational time constraints, are notorious for requiring huge amounts of both memory and processor time" [18]. Starting with the Thinking Machines CM-5, ocean codes on MPP platforms were able to outperform the Cray PVP machines [16]. Ocean codes are also more intrinsically scalable than atmospheric codes. There is not as much need to communicate values during execution, and it is easier to overlap the communication and computation parts of the code. Sawdey et al. [18], for example, have developed a two-tier code, SC-MICOM, that uses concurrent threads and shared memory within an SMP box. When multiple SMP boxes are available, a thread on each box handles communication to the other SMP boxes. Each box computes a rectangular subset of the overall grid. The cells on the outside edges of the rectangular subset are computed first, then handed to the communication thread. While communication is in progress, the remaining cells inside the rectangular subset are computed. This code has been tested on an SGI PowerChallenge Array using four machines with four CPUs each (1686 Mtops each), and on a 32-node Origin2000 (11,768 Mtops) where each pair of processors was treated as an SMP.

The dominant machine in ocean modeling today is the Cray T3E. It is allowing a doubling of the resolution over earlier machines. Several systems are installed around the world with 200+ nodes for weather work, including a 696-node T3E-900 (228,476 Mtops) at the United

[11] 224-nodes were used as compute nodes. The remaining 32 nodes were used for I/O.

Kingdom Meteorological Office. These systems are allowing research (and some production work) on ocean models to advance to higher resolutions.

Modeling Coastal Waters and Littoral Seas

Modeling ocean currents and wave patterns along coasts and within seas and bays is an application area with both civilian and military interests. These include:

- Logistics Over The Shore (LOTS) operations in support of an amphibious operation or military operations near a coast
- Current and wave height predictions for Navy surface and subsurface operations in restricted waterways
- Wind and wave conditions for minesweeping operations
- Breakwater design and erosion models for harbors and beaches
- River flows into the ocean to model currents and the spread of contaminants
- Tide highs and lows

The military time constraints are hard. Regional forecasts are needed to support specific military operations. Civilian work does not typically have real-time constraints. The more common civilian model is a long-term, months to years, model. This would be used for beach erosion studies, harbor breakwater design, etc.

The Navy is not able to run coastal and littoral models for all areas of the world due to limits on computer resources. Currently, about six regional models are run each day at resolutions greater than the global ocean model. These are chosen with current operational needs and potential threats in mind. They include areas such as the Persian Gulf, the Gulf of Mexico, the Caribbean, and the South China Sea.

It is possible to execute a regional model on fairly short notice for a new area. For example, the Taiwan Strait regional model was brought up fairly quickly during the Chinese naval missile exercises. However, better regional models can be produced when the model is run frequently. This allows the researchers to make adjustments in the model based on actual observations over time. Thus, the Persian Gulf model that is executed now is much better than the model that was initially used during the Persian Gulf crisis [16].

Coastal wave and current models are dependent on a number of inputs:

- mesoscale regional atmospheric forecast—provides wind forcing data
- global current model—provides first approximation for local currents
- temperature data—surface temperature from satellites, sub-surface temperatures from probes dropped by ships or aircraft in the region
- detailed coastal and bottom geography
- river outflow into the region

The Spectral Wave Prediction System (SWAPS) uses nested components. The first component is the deep-water wave-action model. This provides the input to the shallow-water wave-action model, for water 20 m or less in depth. Finally, when needed, the harbor response model and/or the surf model are used to provide high resolution in these restricted areas. The wave action model is limited by the accuracy of the atmospheric model [19]. This layered model is typical of the approach used in the regional models. Forty-eight-hour forecasts are

the norm at FNMOC, although five-day forecasts are possible. The limitation is the compute time available.

While 3 km resolution is the goal for deep ocean models, this is only marginal for coastal work. A few years ago, it was thought that a 1 km resolution would be a reasonable target. Now, the goal is down to as low as 100 m. The accuracy desired is dependent on the nature of the water in the region. For example, a large freshwater river flowing into a basin will cause a freshwater plume. Both the flow of the fresh water and the differing density of fresh and salt water have to be handled correctly by the regional model [16].

One major difference in the deep and shallow ocean models is the method of dividing up the vertical layers in the model. The deep ocean model uses six layers whose size is determined by the density of the water, not its depth. Over a continental shelf and on into shallow waters, a fixed-depth set of layers is used. Thus, both the method of determining the layer depth and the number of layers is different in the two models. NRL plans to add a "hybrid-grid" capability to the global model, allowing it to use fixed depth layers over the shelf. However, this requires significantly more computer time and only makes sense when the global model "resolves" the shelf, e.g. at 3 km. Separate coastal models, with even higher resolution, will still be required, but coupling them to the global model will be greatly simplified.

The Navy is working on new ways of delivering regional forecasts to deployed forces. One approach is to continue to compute the regional forecasts at FNMOC or at the Naval Oceanographic Office (NAVOCEANO), and send the results to those ships that need it. The other alternative is to place computing resources onboard ship to allow them to compute the regional forecast. The onboard ship solution has two principal drawbacks. First, the ship will need input from the global and regional atmospheric forecasts and the global ocean forecast to run the regional model. Sending this data to the ship may well exceed the bandwidth needed to send the completed regional forecast. Second, there would be a compatibility problem with deployed computers that were of various technology generations. FNMOC is looking into options for allowing deployed forces greater flexibility in accessing the regional forecasts when they need them. Among the options are web-based tools for accessing the forecasts and databases onboard ships that contain the local geographic details in a format such that the regional model can be easily displayed on top of the geographic data [16].

Weather Performance Summary

Figure 4–4 shows the CTP requirements for weather applications studied in this report. Table 4–4 shows a summary of the performance characteristics of the weather applications described in this report sorted first by year and then by the CTP required. It should be noted that some applications appear more than once at different CTPs. This is particularly true of FNMOC's global atmospheric forecast. As FNMOC has upgraded its Cray C90 from 8 to 12 to (soon) 16 processors, it has been able to run a more accurate forecast within the same time constraints. Thus, the increase in CTP does not give a longer-range forecast, but provides a 10-day forecast with greater accuracy over the 10-day period.

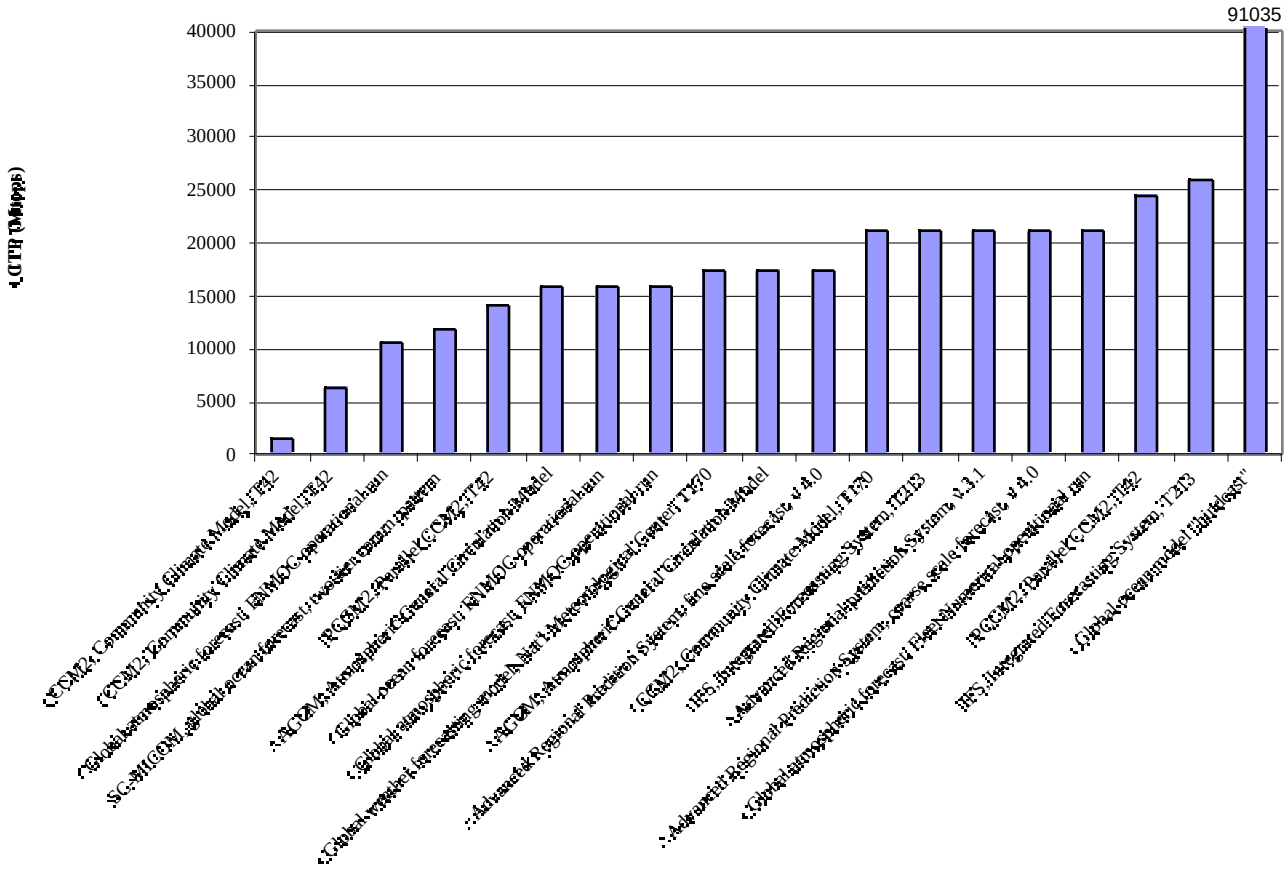

Figure 4–4. Weather applications

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
Cray C98	1994	10625	~5 hrs	Global atmospheric forecast, FNMOC operational run [20]	480 x 240 grid; 18 vertical layers
Cray C90/1	1995	1437		CCM2, Community Climate Model, T42 [9]	128 x 64 transform grid, 4.2 Gflops
Cray T3D/64	1995	6332		CCM2, Community Climate Model, T42 [9]	128 x 64 transform grid, 608 Mflops
IBM SP-2/128	1995	14200		PCCM2, Parallel CCM2, T42 [7]	128 x 64 transform grid, 2.2 Gflops
IBM SP-2/160	1995	15796		AGCM, Atmospheric General Circulation Model [12]	144 x 88 grid points, 9 vertical levels, 2.2 Gflops
Cray T3D/256	1995	17503		Global weather forecasting model, National Meteorological Center, T170 [11]	32 vertical levels, 190 x 380 grid points, 6.1 Gflops
Cray T3D/256	1995	17503		AGCM, Atmospheric General Circulation Model [12]	144 x 88 grid points, 9 vertical levels, 2.5 Gflops
Cray C916	1995	21125		CCM2, Community Climate Model, T170 [9]	512 x 256 transform grid, 2.4 Gbytes memory, 53. Gflops
Cray C916	1995	21125		IFS, Integrated Forecasting System, T213 [10]	640 grid points/latitude, 134,028 points/horizontal layer, 31 vertical layers
Cray C916	1995	21125		ARPS, Advanced Regional Prediction System, v 3.1 [5]	64 x 64 x 32, 6Gflops
Paragon 1024	1995	24520		PCCM2, Parallel CCM2, T42 [7]	128 x 64 transform grid, 2.2 Gflops
Cray T3D/400	1995	25881		IFS, Integrated Forecasting System, T213 [10]	640 grid points/latitude, 134,028 points/horizontal layer, 31 vertical layers
Cray T3D/256	1996	17503	105 min	ARPS, Advanced Regional Prediction System, v 4.0, fine scale forecast [14]	96 x 96 cells, 288 x 288 km, 7 hr forecast
Cray C916	1996	21125	45 min	ARPS, Advanced Regional Prediction System, v 4.0, coarse scale forecast [14]	96 x 96 cells, 864 x 864 km, 7 hr forecast
Origin2000/32	1997	11768		SC-MICOM, global ocean forecast, two-tier communication pattern [18]	
Cray C912	1997	15875	~ 5 hrs	Global atmospheric forecast, FNMOC operational run [20]	480 x 240 grid; 24 vertical layers

Cray C912	1997	15875	1 hr	Global ocean forecast, FNMOC operational run [16]	1/4 degree, 25 km resolution
Cray T3E-900/256	1997	91035	590 hrs	Global ocean model "hindcast" [16]	1/16 degree, 7 km resolution
Cray C916	1998	21125	~5 hrs	Global atmospheric forecast, FNMOC operational run [20]	480 x 240 grid; 30 vertical layers

Table 4–4. Weather applications

The Changing Environment for Numerical Weather Forecasting

Future Forecasting Requirements

In the next five years, there are likely to be significant changes in the forecasting models used and the types and frequency of the forecasts delivered. The following are some of the key anticipated changes:

Coupling of Oceanographic and Meteorological Models (COAMPS) and development of better littoral models. Currently, meteorological forecasts are made with minimal consideration of the changing conditions in the oceans. Oceans clearly have a strong impact on weather, but in the current operational environment the computing resources do not permit a close coupling. The COAMPS models will be particularly important for theater, mesoscale, and battlespace forecasting, which are likely to be focused on littoral areas in which the interplay of land, air, and ocean creates particularly complicated (and important) weather patterns. Current estimates are that such computations will require a Teraflops of sustained computational performance and 640 Gbytes of memory.

Shorter intervals between runs. Currently, the models used by FNMOC are being run between one and eight times per day. In the future, customers will want hourly updates on forecasts. This requirement places demands on much more than the computational engine. The amount of time needed to run the model is just one component of the time needed to get the forecast in the hands of those in the field who will make decisions based on the forecast. At present, it takes several times longer to get the initial conditions data from the field and disseminate the final forecast than it does to run the model itself.

Greater resolution in the grids. In the next three to four years, FNMOC practitioners anticipate that the resolution of the models used will improve by a factor of two or more. Table 4–2, page 58, shows these expected changes. Some of the current customers view resolutions of less than 1 km as a future goal. Weather forecasting with this resolution is necessary for forecasting highly local conditions, such as the difference in weather between neighboring mountain peaks or valleys. Such forecasting is closely related to the next requirement of the future:

Emphasis on forecasting weather that can be sensed. As battlespace forecasting becomes practical, the ability to predict weather phenomena that will impact the movement of troops and equipment, placement of weapons, and overall visibility will be important sources of military advantage. Forecasts will be designed to predict not only natural weather phenomena such as fog, but also smoke, biological dispersion, oil dispersion, etc. These forecasts have obvious civilian uses as well.

Ensemble Forecast System. An Ensemble Forecast System (EFS) using NOGAPS has been included in the operational run at FNMOC. The purpose of EFS is both to extend the useful range of numerical forecasts and to provide guidance regarding their reliability. The FNMOC ensemble currently consists of eight members, each a ten-day forecast. A simple average of

the eight ensemble members extends the skill of the medium-range forecast by 12 to 24 hours. Of greater potential benefit to field forecasters are products that help delineate forecast uncertainty. Another alternate method is to make simple average ensemble forecasts (SAEF) of several Weather Centers with different analyses and models. Both of these systems attempt to evaluate the effect of chaos theory on weather forecasting by using many different events to determine the probability distributions of the weather in the future.

Future Computing Platforms

FNMOC is currently in a procurement cycle. Their future computing platforms are likely to include MPP systems. The migration from the current PVP machines to the MPP platforms is likely to require a huge effort by the NRL and FNMOC practitioners to convert not only their codes but also their solutions and basic approaches to the problem. Gregory Pfister has the following comment about the difficulty of converting from a shared-memory environment to a distributed-memory one:

> This is not like the bad old days, when everybody wrote programs in assembler language and was locked into a particular manufacturer's instruction set and operating system. *It's very much worse.*
>
> It's not a matter of translating a program, algorithms intact, from one language or system to another; instead, the entire approach to the problem must often be rethought from the requirements on up, including new algorithms. Traditional high-level languages and Open Systems standards do not help [21].

A great deal of effort continues to be made by researchers at the NRL and elsewhere to adapt current codes to parallel platforms and develop new models less grounded in the world of PVP architectures. There is little doubt that parallel platforms will play increasingly central roles in numerical weather forecasting.

Currently, work is in progress on porting the NOGAPS global forecast model at NRL for use as a benchmark in the procurement process. The code is being tested on several platforms including the T3E, SP-2, and Origin2000, and is to be ready by spring 1998 for use in procurement decisions. There are problems with load balancing in the parallel version. For example, latitude ring interleaving assigns a mix of sunshine and dark latitude bands to each processor. The interleaving spreads the computational load more evenly, but complicates the communication patterns among the processors. The right mix to use is a subject of ongoing research.

Microprocessor hardware is likely to continue its breathtaking rate of advance through the end of the century. The prevalence of systems based on commercial microprocessors is likely to result in steady improvements in the compilers and software development tools for these platforms. These developments are likely to increase the sustained performance as a percentage of peak performance of the massively parallel systems.

Cryptographic Applications

Cryptographic applications are those involved with the design and implementation of security codes, and the attempt to "break" codes to be able to read the encoded message. The historical experiences from World War II of the Ultra project in England to decode the German ciphers, and the similar project in the United States on the Japanese codes, gave the Allies significant military advantages.

The advent of parallel systems, be they low-end clusters or high-end massively parallel systems, has made it possible to break selective codes. The problem space is essentially divided across the available platforms, and each machine works on its portion, looking for a

plain text result. Thus, it is possible using uncontrollable technology to acquire an ability to decipher messages.

However, being able to decipher a single message, or determine one key, does not necessarily imply an ability to read all the messages of one's adversary. There are two problems that must be solved if one needs a more complete ability to read messages encoded using a wide variety of keys. The first is a real-time issue: decoding the message weeks after an event may be of no use. The second is a volume issue: a single message may not be useful without the context of other messages on the topic. Thus, there is a quantitative difference between the ability to determine a single key and being able to read in real time a usable portion of an adversary's message traffic.

We have had extensive discussions with the intelligence community about cryptographic applications. While we cannot discuss directly the details of those conversations, we can make the following conclusions:

- The cryptanalysis community is aware of the growth in performance of uncontrollable systems and its impact on the ability of adversaries both to decipher codes and to create and use codes that are more difficult to break.
- The transition from the vector-pipeline processor to massively parallel systems is in progress.
- The applications developers are not happy about the transition to the MPP systems due to specific limitations of those systems.

Problems with PVP to MPP Transition [22]

The cryptanalysis community has had the luxury of significant input into the design of the PVP processor, in particular those manufactured by Cray Research. The advent of MPP platforms and their dependence on general-purpose processors has greatly reduced this influence. The market for the MPP platforms is much larger than the scientific community alone, and is driven by commercial needs.

For cryptographic applications, the vector turns out not to be the primary advantage of the PVP systems. Instead, it is the high performance of the complete system that is important. The attention paid to the performance of each of the subsystems on the Cray PVP machines does not (as yet at least) have a corollary in the MPP architectures.

First among the concerns is memory, both in size and in bandwidth. MPP platforms offer larger memories than current PVP systems. The bandwidth, however, is not adequate. This is evident in other applications described in this paper, and cryptographic applications are no exception. Worse, cryptographic techniques have a requirement for random access to memory. The cache performance of microprocessors is predicated on the assumption of locality of access. The cost to load a whole cache line when only one value is needed leads to poor performance. Cache coherence protocols can actually make a series of random remote accesses worse. Finally, microprocessors generally provide no way to bypass the cache.

Second, cryptographic applications are primarily integer, not floating-point, codes. Less than 5 percent of operations in typical applications are floating-point codes. PVP machines exhibit excellent performance on either integer or floating-point operations. Microprocessors, on the other hand, give much better performance on floating-point than on integer operations. The microprocessor integer unit is not distinct from the control logic, and the system optimizes for the control logic. Additionally, the split between floating-point and integer operations affects the number of registers available to each. For applications that are nearly exclusively integer, this means that half the register set is unavailable.

Some special operations are available in PVP systems as a result of lobbying by the cryptanalysis community. They would very much like to see some or all of these in microprocessors. These include:

- population count—counts the number of ones or zeros in a bitstream. First appeared in the Cray 1.
- bit matrix multiply—64 vector registers with 64 bits each can be viewed as a 64 x 64 bit array. Allows matrix multiply using two such registers. First appeared in the Cray YMP. The C-90 and T-90 hardware versions are now 100 times faster than a software implementation.

The hardware vendors are aware of the desire for similar operations in today's commodity processors. However, very few are even considering adding the instructions. If these or other operations were added, it is necessary that they be accessible from high-level languages. The compiler does not have to be able to recognize when to use the instruction, but compiler directives or direct assembly commands should be available to the programmer. This is a productivity requirement given the need to develop all software in-house rather than take advantage of third-party software.

The last point is a combined system issue: ease of use. The MPP market has been very volatile, with new systems appearing that have at best limited connection to previous systems, and system software that is not retrofitted to earlier systems. This places a tremendous burden on the application developers, both in the learning curve and in the actual hours committed to developing and maintaining code. This is not a problem unique to cryptographic applications. What makes it harder for this community, however, is the complete lack of third-party solutions. They cannot rely on a set of companies to provide updated software to match new computer systems.

Signal and Image Processing

Introduction

Signal and image processing involves coupling computing and sensing technologies such as radar or sonar. The result is a system that creates images of remote objects and processes the resulting images for consumption by humans or input into other automated systems (e.g., guidance or decision support). The applications can be computationally intense, due to the need to cope with large volumes of sensor data and perform any necessary filtering, enhancement, or interpretation functions, often in real time. In addition, because many of the platforms are to be installed in airborne, space, or sea-faring vehicles, these systems are likely to have strong size, weight, and power constraints.

Two categories of signal and image processing applications that are particularly computationally demanding are synthetic aperture radar (SAR) and space-time adaptive processing (STAP). SAR combines signal processing with a mobile radar system (airborne or space-based) that uses a small antenna aperture to produce radar images comparable to what could be delivered by a radar system with a prohibitively large antenna aperture. STAP refers to algorithms designed to extract a desired radar signal by filtering out the effects of clutter and interference under circumstances in which the radar platform and clutter sources are moving with respect to each other.

Signal and image processing makes use of high-performance computing to support operational requirements.

The Computational Nature of Signal and Image Processing

Signal and Image Processing Overview

Most signal and image processing applications involve the processing of data collected by a sensor. The processing capability involved can be broken into two general categories, front-

end processing and back-end processing, as shown in Figure 4–5. In front-end processing, the computational activities tend to be rather static, i.e. the same manipulation of the data is performed over and over again without great concern for the meaning or significance of the objects contained in the image. Front-end processing systems usually have two dominant concerns:

- Keeping up with the flow of data from the sensors.
- Providing the necessary computational performance in a package that meets the size, weight, and power constraints of the host platform (e.g., aircraft).

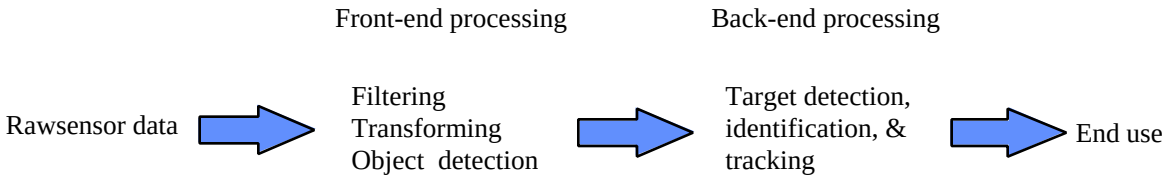

Figure 4–5. Data processing in sensor-based applications

Keeping up with the flow of data from the sensors has two crucial components: throughput and latency. A sensor typically generates data at a rate that is a function of the number of bits used to represent each image or data sample and the rate at which samples are being taken. A basic requirement for a computing system is that it be able to process data at least as fast as it is generated. In addition, there are latency requirements for the maximum time a system is allowed to take to process a single image. The throughput can often be increased by, for example, increasing the number of processors at work; doubling the number of processors to process twice as many images in a given unit of time. Increasing throughput may not, however, improve the latency. Doubling the number of processors does nothing to increase the speed at which a single processor can finish processing a single image. In general, latency requirements present a more challenging problem to systems designers than throughput requirements. Trends in front-end processing systems are discussed later in this section.

Back-end processing focuses on understanding the content of the image to identify and track targets. While such systems may be deployed on the same platform as the sensor, many processing systems today relay the output of the front-end processing to a system closer to the end-user. Consequently, systems for back-end processing may not operate under the same size, weight, and power constraints as for front-end processing.

Synthetic Aperture Radar (SAR)

SAR is a remote sensing technique used primarily to obtain high-resolution ground images from sensors flown on an aircraft, satellite, or other space vehicle. Unlike passive sensors which detect the reflected energy of sunlight or the thermal or microwave radiation emitted by the earth, SAR is an active sensor technique that generates pulses of electromagnetic waves toward the ground and processes the reflected signals. Consequently, SAR can be used day or night and under any weather conditions for a variety of civilian and military purposes in areas such as geology, agriculture, glaciology, military intelligence [23,24].

An ongoing objective of SAR systems is to improve the system's resolution, defined as how well the system is able to distinguish between two closely spaced objects. In conventional radar systems, the resolution in the cross-track direction (perpendicular to the line of flight) is determined by the duration of a pulse. Shorter pulses (at the cost of greater energy requirements) give greater resolution. Resolution in the direction of flight is limited by the size of the physical antenna and the operating frequency of the radar. Antenna theory dictates that

an antenna several kilometers in length would be needed to resolve ground objects 25 meters in diameter [25]. Building an antenna of this size is not feasible. Synthetic aperture radar is a technique to improve the resolution of a radar system without increasing the size of the physical antenna by taking advantage of the motion of an air- or space-borne sensor. Figure 4–6 shows a standard SAR geometry. A small antenna travels in the direction of the line of flight and emits short pulses of microwave energy across an area of earth, delineated by the dark lines, at a particular repetition rate. Each pulse will cause an echo pattern that will be captured as a matrix of data points.

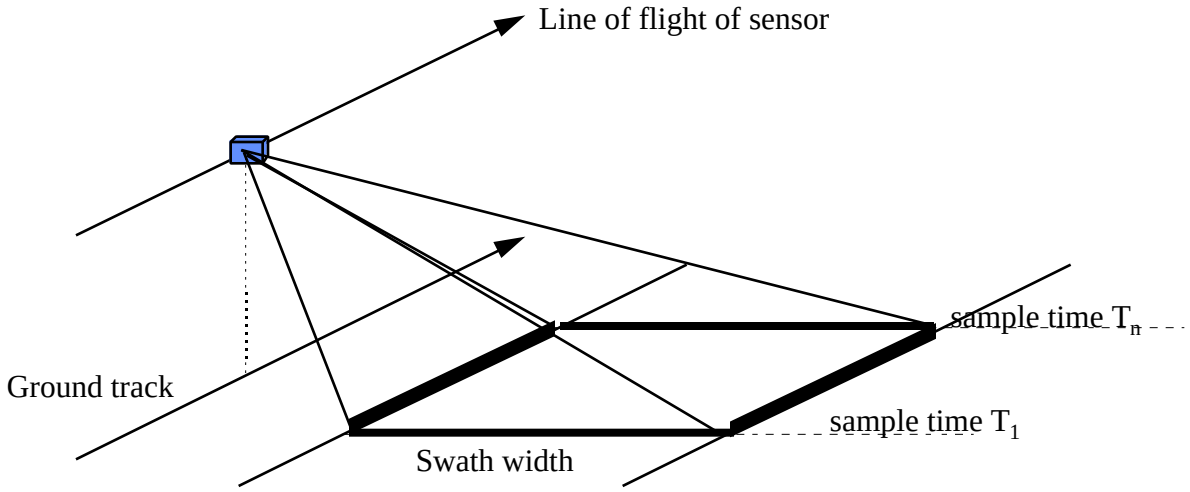

Figure 4–6. Synthetic aperture radar

The sensor transmits pulses repeatedly as it passes by an area. Because of the sensor's motion, a given point target lies at different distances from the sensor for different pulses. Echoes at different distances have varying degrees of "Döppler shift,"[12] which can be measured. Integrating the individual pulse echoes over time creates an image.

Each energy pulse can result in a large number of data points. A typical image may consist of hundreds of thousands of samples, with each sample requiring 5–8 bytes of data [26]. Performing the necessary signal processing using discrete Fourier transforms (a widespread signal processing function) and matrix transformations may require several billion computational operations per second [25].

The reflections of these pulses are recorded and integrated to achieve an effective antenna width equal to the distance traveled during the time period, and a maximum resolution in the direction of the line of flight equal to half the size of the physical airborne antenna. The resolution in the direction of the swath width (perpendicular to the line of flight) is proportional to the length of each microwave pulse such that shorter pulses should lead to better resolution. Since there are physical limits to how short the pulses can be and still generate useful echoes, a variety of complex techniques have been developed that employ pulses of varying frequencies to obtain high resolution [25].

[12] The Döppler effect is the change in frequency of a signal that results when the distance between signal source and receiver are changing. The classic example is the drop in pitch (frequency) of a train whistle as the train passes a stationary observer.

	X-SAR sensor	*RASSP SAR Benchmark*
Date	1993	1995
Physical antenna size	12.1 x 0.33 meters	
Nominal altitude	250 km	7.26 km
Frequency Band	9.5 – 19 MHz	Ka-Band
Platform	Intel Paragon (various configurations)	Intel Paragon (12 compute nodes)
Pulse repetition frequency	1302 - 1860 pulses per second	556 pulses per second
Sampling frequency	11.25 – 11.5 million samples per second	1.125 million samples per second

Table 4–5. Mission parameters for X-band SAR sensors. Sources: [26-28]

Table 4–5 illustrates some of the parameters for the X-band SAR sensor deployed by NASA on the space shuttle. It also illustrates parameters for the Rapid Prototyping of Application Specific Signal Processors (RASSP) SAR benchmark developed by MIT Lincoln Labs and implemented on an Intel Paragon [28,29]. The RASSP SAR benchmark is an image formation problem that has moderate computational and throughput requirements that are consistent with the requirements for some onboard real-time processing on unmanned aerial vehicles (UAV). The Spaceborne Imaging Radar-C/X-Band Synthetic Aperture Radar (SIR-C/X-SAR) was flown in two space shuttle missions in 1994 during which 4 Terabytes of data were collected and processed in subsequent years by various teams of researchers. Table 4–6 shows the time taken on an Intel Paragon to perform the correlation processing of data collected by the space shuttle's systems over a 20-second period. These results represent the first use of a commercial MPP system to process high-volume SAR data in near real time [30].

		Paragon XP/S 35+	
No. Channels of imaging data	*No. Nodes*	*CTP*	*Time (sec)*
1	64	2,621 Mtops	36.0
4	256	7,315 Mtops	49.0
8	512	13,235 Mtops	55.0

Table 4–6. SIR-C/X-SAR processing times on the Intel Paragon. Source: [30]

Research done at the Jet Propulsion Laboratory demonstrated high sustained performance on Grand Challenge SAR problems. In November 1997, investigators achieved over 10 Gflops sustained performance on a 512 processor Cray T3D (32,000 Mtops). In 1998, investigators achieved 50 Gflops sustained performance on a Cray T3E with 512 nodes (115,000 Mtops). Thus, only recently has commercial HPC technology been able to process SAR data at a rate that matches the data production of radar devices such as those flown on the space shuttle [31].

SAR computations have been performed by special-purpose, often customized, digital signal processing (DSP) hardware. Because of the size, weight, and power constraints, the systems have employed exotic technologies such as stackable memory units that permit a given memory to be packaged in a smaller than normal space.

There are two dominant trends in SAR hardware development. First, there is a great deal of pressure to use systems based on commercially available technologies. Second, the growing

computational demands of SAR have fueled efforts to develop sizeable parallel processing platforms consisting of tens or hundreds of processors.

In the efforts to move away from customized hardware solutions, two general approaches have been pursued. One approach has been to use general-purpose parallel processing systems, like the Intel Paragon, for front-end data processing. There have been a number of difficulties with this approach. While the systems have adequate computational capability, they tend to use processors that consume too much power[13] or employ packaging that does not make the best use of the space available. The two factors are often related. One of the advantages of physically separating components is that heat is more easily dissipated. Also, the architecture of general-purpose systems in many cases is poorly suited for signal processing applications. One example is the memory system architecture. To build general-purpose systems, system architects have taken great pains to develop sophisticated cache systems. Unrelenting real-time data streams continually present processors with new data, making the overhead required to maintain the cache systems a liability [32]. As a result, general-purpose systems used for signal processing applications are normally in non-embedded settings, such as those used to process the shuttle's SIR-C/X-SAR data.

The alternative approach has been to utilize commercially available processors that have the best performance/power characteristics, and architect systems around them that are oriented towards the specific demands of signal processing applications. Besides having good size/power/weight parameters, advanced systems must have high-throughput/low-latency interconnects, and, preferably, the flexibility necessary to incorporate a mix of heterogeneous processors that will perform the computational tasks for which each is best suited. For example, Mercury Computing Systems, a market leader, has an architecture that permits low-power general-purpose processors such as the i860 and the PowerPC 603e to be integrated with digital signal processors such as the SHARC [33].

The companies that have emerged as leaders in the embedded systems market—Mercury Computing Systems, Sky Computers, CSPI, and Alacron—are not general-purpose systems vendors. Instead, they began as providers of single-board accelerators employing one or more CPUs such as the i860 or specialized DSP processors and later expanded into multi-board systems with scalable interconnects. Because some of the fundamental algorithms of signal processing like the fast Fourier transforms require a good deal of inter-node communication, these interconnects must have high throughput and low latency characteristics; conventional networking interconnects are not suitable. The resulting machines are called DSP multi-computers. Most of the DSP multi-computers are not particularly powerful in comparison with today's general-purpose systems. Table 4–7 shows two systems that have been deployed by Mercury within the last two years.

Year	*System*	*Usage*	*CTP*
1997	Mercury Race (20 i860 processors, ruggedized) [34]	SAR system aboard P3-C Orion maritime patrol aircraft	866
1996	Mercury Race (52 i860 processors) [35]	Sonar system for Los Angeles class submarine	1,773

Table 4–7. Signal processing systems sold by Mercury Computer Systems

In spite of their modest performance, the systems are quite controllable if one considers other qualities. First, the systems are rarely found as stand-alone systems; relocating these embedded systems would significantly hamper the function of their host system (e.g.,

[13] For example, Mercury Computer Systems, Inc., a leading vendor of embedded computing systems, used the i860 (40 MHz), but did not use the i860 (50 MHz). The latter provided 25 percent better performance, but required three times as much power [119].

submarine) and could hardly go unnoticed. Second, such systems have not been sold in great numbers outside military customers, although Mercury is cultivating real-time embedded commercial markets such as video servers and hospital imaging systems. Even so, the configurations of the latter are significantly smaller than those listed above. Third, the systems deployed in military applications frequently have special qualities such as ruggedization not needed for stationary commercial systems.

Although many of the systems deployed in recent years have modest performance relative to commercial general-purpose systems, the performance has improved, and is poised to increase dramatically. First, the number of processors in a configuration has increased significantly. Until recently, systems employing more than a handful of processors were extremely rare. DSP multi-computers with 20 or more processors are now relatively mainstream systems. Second, as the embedded systems industry moves from the modest i860s (CTP 66.7 Mtops) to much more powerful PowerPC 603e (450 Mtops at 200 MHz), the performance of configurations will increase sharply. For example, in 1997, Mercury Computer Systems shipped a 140-processor system with a CTP of over 27,000 Mtops [33].

Space-Time Adaptive Processing (STAP) Benchmark

STAP is an advanced signal processing technique designed to improve the functional performance of airborne radar systems with little or no modification to the basic radar design. The technique, employed after some standard pre-processing is performed on a signal, suppresses the sidelobe clutter. STAP adaptively combines samples from multiple signal channels and pulses to nullify clutter and interference. The algorithm takes the output of pre-processing and calculates and assigns different weights to the results before recombining them into a final output. "Adaptive" refers to the manner in which the algorithm adjusts the weightings in response to changing conditions in the signal environment.

STAP algorithms are computationally intensive and beyond the capabilities of most processors used in current airborne platforms [36]. However, they are likely to be important in the future as platforms are upgraded. Benchmarks run on available technologies provide some indication of the performance needed. MITRE Corporation and Rome Laboratories developed RT_STAP, a benchmark for evaluating the application of scalable high-performance computers to real-time implementation of STAP techniques [36]. The choice of system will necessarily be a function not only of performance but also of the size/weight/power/cost factors described above.

RT_STAP is based on the Multi-Channel Airborne Radar Measurements (MCARM) data collection system developed by Westinghouse for Rome Laboratory in the early 1990s. The system uses a 32-element antenna, transmitters, and receivers capable of generating L-band (1.3 GHz) radar measurements. The 32 antenna outputs are combined to produce 22 channels. Generating signals with a pulse repetition rate of 250–2000 Khz and a pulse width of 50 to 100 microseconds, the system then records data at a sampling rate of 5 million samples per second [36].

The RT_STAP benchmark suite provides three tests of varying difficulty. The easiest benchmark, requiring a sustained throughput of 0.6 Gflops, reflects the computational demands of technology in current radar systems that perform a function similar to STAP. The medium and hard benchmarks, requiring sustained processing rates of 6.46 and 39.81 Gflops, respectively, are more fully adaptive implementations of STAP and generally beyond the capabilities of today's widely deployed systems.

The computational requirements are determined not only by the complexity of the algorithm used, but also by the overriding constraint that the system's throughput and latency allow processing of the data at least as fast as the sensor can generate it. In the case of RT_STAP the system must process input data volumes of 245.8 thousand, 1.97 million, and 2.7 million samples with a maximum latency of 161.25 milliseconds for the easy, medium, and hard cases, respectively.

Figure 4–7 shows the performance of various configurations of the SGI Origin2000 on the RT_STAP benchmark. The dark solid line indicates the threshold at which a configuration is able to process the necessary data within the 161.25 millisecond latency limit.

		Time (msec)		
# Processors	CTP (Mtops)	***Easy***	***Medium***	***Hard***
1	448	135	1285	5650
2	781	70	690	2890
4	1,518	45	335	1475
8	2,990	30	170	750
16	5,908	25	90	390
32	11,768	20	60	205

Figure 4–7. Origin2000 performance on RT_STAP benchmarks. Source: [37]

The benchmarks illustrate a number of points. First, a single R10000 processor can perform the kinds of clutter cancellation functions used in current radar systems (easy case). Second, the requirements of the medium benchmark exceed the performance of a deskside Origin2000 (8-processor maximum). Third, the hard or fully adaptive STAP algorithm exceeds the capability of a 32 processor (two-rack) Origin2000. Of course, the space/weight/power constraints of actual airborne platforms are likely to preclude the Origin2000. Nevertheless, the benchmarks give an indication of the minimum performance requirements necessary for STAP algorithms of varying complexity.

Summary and Conclusions

- Signal and image processing is a candidate application area of national security interest. It requires controllable high-performance computing technology to satisfy both operational and research needs.
- Signal and image processing applications such as SAR and STAP are computationally demanding, and require platforms subject to stringent size/weight/power restrictions. Moreover, algorithms such as fast Fourier transform (FFT) that are at the heart of many signal and image processing applications have inter-node communications requirements that demand high-bandwidth, low-latency interconnects.
- The performance of most signal and image processing platforms deployed in air-, space-, or underwater vehicles today is, in most cases, substantially below that of widely available general-purpose systems.
- The packaging requirements of embedded systems, and the nature of the current customer base, make export control of embedded systems feasible, provided the controls are based on factors other than performance as measured by CTP.
- The performance of embedded systems has increased dramatically within recent years as multiprocessor systems with an order of magnitude more than processors in the past are deployed as mainstream configurations.
- The performance of embedded systems will continue to jump sharply in the near future with the transition to processors like the PowerPC 603e, which have CTP ratings nearly eight times greater than the previous mainstay, the i860.

Computational Fluid Dynamics

Computational Fluid Dynamics (CFD) applications span a wide range of air and liquid fluid flow problems. Application areas studied for this report include aerodynamics for aircraft and parachutes, and submarine movement through the water. The basic CFD equations are the same for all these applications. The differences lie in specifics of the computation and include such factors as boundary conditions, fluid compressibility, and the stiffness or flexibility of solid components (wings, parafoils, engines, etc.) within the fluid. The resolution used in an application is dependent on the degree of turbulence present in the fluid flow, size of the object(s) within the fluid, time scale of the simulation, and the computational resources available.

CFD represents the first of a number of application areas studied for this report that do have primarily research requirements for high-performance computing rather than operational requirements. We begin with a description of CFD in aerodynamics, specifically airplane design. Much of the general description of CFD applies to all the application areas; thus, subsequent sections describe only differences from the aerodynamic practice.

CFD in Aerodynamics

The Navier-Stokes (N-S) equations are a set of partial differential equations that govern aerodynamic flow fields. Closed-form solutions of these equations have only been obtained for a few simple cases due to their complex form.

Numerical techniques promise solutions to the N-S equations; however, the computer resources necessary still exceed those available. Current computing systems do not provide sufficient power to resolve the wide range of length scales active in high-Reynolds-number turbulent flows.[14] Thus, all simulations involve solution of approximations to the N-S equations [38]. The ability to resolve both small and large turbulence in the fluid flow separates the simple solutions from the complex solutions.

Solution Techniques

Ballhaus identifies four stages of approximation in order of evolution and complexity leading to a full Navier-Stokes solution [38]:

[14] The Reynolds number provides a quantitative measure of the degree of turbulence in the model. The higher the Reynolds number, the more accurately turbulence can be represented by the model.

Stage	*Approximation*	*Capability*	*Grid*	*Compute factor*
I	linearized inviscid	subsonic/supersonic pressure loads vortex drag	3×10^3	1
II	nonlinear inviscid	Above plus: transonic pressure loads wave drag	10^5	10
III	Reynolds averaged Navier-Stokes	Above plus: separation/reattachment stall/buffet/flutter total drag	10^7	300
IV	large eddy simulation	Above plus: turbulence structure aerodynamic noise	10^9	30,000
	full Navier-Stokes	Above plus: laminar/turbulent transition turbulence dissipation	10^{12} to 10^{15}	3×10^7 to 3×10^{10}

Table 4–8. Approximation stages in CFD solutions

There are 60 partial-derivative terms in the full Navier-Stokes equations when expressed in three dimensions. Stage I uses only three terms, all linear. This allows computations across complex geometries to be treated fairly simply. Subsequent stages use progressively more of the N-S terms. Finite-difference approximations are used and require that the entire flow field be divided into a large number of small grid cells. In general, the number of grid cells increases with the size of the objects in the flow field, and with the detail needed to simulate such features as turbulence.

In aircraft design, the Stage I approach can supply data about an aircraft in steady flight at either subsonic or supersonic speeds. Stage II allows study of an aircraft changing speeds, particularly crossing the sound barrier. It also supports studies of maneuvering aircraft, provided the maneuvers are not too extreme. Thus, an aircraft making a normal turn, climb, and/or descent can be modeled at this stage. A jet fighter making extreme maneuvers, however, requires the addition of the viscous terms of the N-S equations. Stage III can provide this level. It uses all the N-S terms, but in a time-averaged approach. The time-averaged turbulent transport requires a suitable model for turbulence, and no one model is appropriate for all applications. Thus, this is an area of continuing research.

Simulating viscous flow complete with turbulence requires another large step up the computational ladder. Hatay et al. [39] report the development of a production CFD code that does direct numerical simulation (DNS) that includes turbulent flow. DNS is used to develop and evaluate turbulence models used with lower stage computations. The code is fully explicit; each cell requires information only from neighboring cells, reducing the communication costs. Hatay et al. give results for both NEC SX-4 and Cray C90 parallel machines. The code has also been ported to the T3D/E and SP-2 MPP platforms. These platforms offer greater memory capacity, necessary to solve usefully large problems. The code illustrates some of the difficulties with moving CFD applications to parallel platforms. The first task was to run the code on a single processor, but make use of all the memory available on a shared-memory machine. Then, considerable effort is required to get the communication/computation ratio low enough to achieve useful speedup over more than a few processors. Finally, the code was ported to the distributed memory platforms.

Aircraft Design

Building on the Basics established that aircraft design, even for military aircraft, generally falls at or below the threshold of uncontrollability. This has not changed since the 1995 study. Two things have occurred in the past two years, both continuations of trends that stretch back to the early 1990s. First, aircraft design is moving to what can be termed "whole plane" simulations. This refers to the three-dimensional simulation of an entire aircraft, including all control surfaces, engine inlets/outlets, and even embedded simulations of the engines. Within engine development, this same principle is evident in moves to develop high-fidelity simulations of each engine component and combine them into a single run. While such whole plane and whole engine simulations are still in development, significant progress has been made.

Second, aircraft codes are moving to parallel platforms from vector platforms. This transition has lagged behind what has been done in other application areas. This is largely due to the poor scalability of the applications used in aerodynamics, and to the memory bandwidth and memory size requirements of the applications. A secondary consideration is the need to validate a code before industry is willing to use it.

Whole Plane and Whole Engine Designs

Historically, aircraft designers have used numerical simulation in combination with wind tunnel experiments. The simulations support more efficient use of expensive wind tunnel time. In turn, this combination has enabled designers to reduce the number of prototypes that need to be built and tested when designing new airframes and engines.

The main difference today from a few years ago lies in the scale of the numerical simulations. Before, designers would study in detail individual components of an aircraft or an engine. Today, the goal is to model the entire aircraft, the entire engine, and simulations that have combined the aircraft and its engines, by combining the individual component simulations. Such models will not replace wind-tunnel experiments, nor the construction and testing of prototypes. Instead, they will again increase the effectiveness of each wind-tunnel test and further reduce the number of tests required. The end result will be a prototype with a greater chance of success delivered in less time. To date, whole-plane simulations typically take advantage of symmetry to model only one-half or one-quarter of the aircraft. Where full 3-D simulations are run, they do not take into account the full Navier-Stokes equations. In particular, the turbulence fields are not simulated. Thus, whole aircraft simulations are at Stage II or at best a simplified version of Stage III.

Whole-plane design needs to model the interactions between the aircraft structure and the fluid flow. Efforts are under way to solve the integration problems that arise when combining CFD and structural mechanics. Grid generation needs to be improved since both disciplines use slightly different grids. Additionally, differences in time scales between structural response and the fluid flows must be addressed.

NASA and aeronautics and engine manufacturers are collaborating on research projects toward this goal. One example from the aeronautics side is a recently completed 3-year cooperative research agreement (CRA) that included IBM, aeronautics firms, universities and two NASA research centers [40]. Researchers at Boeing worked on tiltrotor design using the NASA Metacenter [41]: two IBM SP-2 installations, 160-nodes (16,706 Mtops) at Ames and 48-nodes (6500 Mtops) at Langley. The heart of the rotor program is Boeing's comprehensive rotor analysis code that couples aerodynamics and mechanical dynamics of a rotor. The code incorporates a number of models of different effects associated with helicopter rotor blades and jet engine fan blades. Several of these models were ported to the SP-2 during the study. Table 4–9 summarizes the improvements. The improvement from 80 hours to 2 to 4 hours in run time allows designers to run parametric studies that change only one or two variables on each run. Such studies provide a much more comprehensive understanding of the problem.

Code	*Performance Result*
wake calculation, including rotor performance, vibration, and noise	previous time: 80 hours current time: 2 to 4 hours
coupled codes: rotor motion and deflections rotor CFD analysis	linear speedup on up to 32 processors

Table 4–9. Results of improvements to tiltrotor simulation

In another example, the Computing and Interdisciplinary System Office at NASA Lewis Research Center is leading a project that includes jet engine companies to develop a numerical wind tunnel for air breathing propulsion systems.[15] The goals are to improve product performance and reduce product development time and cost through improved engine simulations. The work is an outgrowth and extension of the Numerical Propulsion System Simulation project [42]. Sub-goals include improved simulations for the individual engine components and interdisciplinary studies combining CFD with structures. A major thrust throughout the project has been the development of parallel codes for a variety of platforms, either by porting vector codes or through development of new codes. Other software projects include work on mesh generation codes to support coupling CFD codes with structures codes, the development of an engine geometry database, and the use of high-speed satellite and terrestrial links for remote operation of engine simulations.

One specific result of this NASA effort is the ENG10 project [43]. ENG10 is an axi-symmetric, complete engine CFD code that models the airflow within and around the jet engine using a Runge-Kutta solution technique. It has been ported from a single-processor version optimized for a vector platform to a parallel version that has been ported to the IBM RS/6000 SP, the Cray T3D, and a cluster of workstations. Considerable effort has been put into optimizing the code through improvements in the data layout and communication patterns. The result is a code that shows a speedup of 6.2 on eight processors in a cluster using an ATM network. Further work with this code is focusing on incorporating results from other engine component codes to improve the accuracy of the simulation. A database developed by General Electric Aircraft Engines is being used to provide input from a variety of engine components to ENG10 [44].

Move from Vector to Parallel Systems

Memory bandwidth continues to be the primary deterrent to porting CFD codes to parallel platforms. This has also been the deterrent in the past to running usable codes on smaller platforms in general. The workstations of a few years ago had neither the raw memory capacity nor adequate memory-CPU bandwidth. Memory capacity has improved in recent years with the decline in the memory cost, although bandwidth problems remain. Most CFD codes on PVP platforms run in single-processor mode, since memory performance is the main requirement. It was not generally worth modifying the code for parallel operation.

For the commodity-based processors used on current MPP platforms and workstations, the bandwidth problem is a function of the cache performance and main memory performance. CFD codes have a high ratio of data fetches to floating point operations that can easily stress the cache- and memory-management hardware.

Even before considering issues of parallelism, a CFD code must be revised to handle the differences in memory performance between the PVP and the MPP models. The layout of data for a vector processor is based on appropriate memory strides. The data layout for cache-based systems requires spatial and temporal locality of data. This forces the use of zero- or

[15] http://www.lerc.nasa.gov/WWW/CISO/

unit-stride memory arrangements, so a cache line brought from memory will contain not only the currently needed value, but value(s) for the next iteration as well [45]. Once the data issues are resolved, the code can be extended to take advantage of additional processors.

In addition to the memory-related difficulties, larger CFD problems, when ported to parallel platforms, have the following characteristics:

- unstructured grids
- multi-block grids
- inter-species equations
- combustion equations for engine codes

Two examples to illustrate the difficulty in porting CFD codes to parallel platforms come from work sponsored by NASA Ames. The first is a port of ARC3D [46]. The port involved a version of ARC3D that is a simple 3-D transient Eüler variant on a rectilinear grid. This code has been very inefficient in the past on cache-based systems. The initial port from the C90 to the Origin2000 was easily completed involving only the proper setting of compiler flags to meet the requirement for 64-bit precision. However, optimizing the code then took two months and involved rewriting five major routines, about 1000 lines of code. By comparison, the second example of a port of CFL3D was done without special optimization [47]. CFL3D uses a finite volume approach to solve time-dependent thin-layer Navier-Stokes equations in three dimensions. Table 4–10 shows the results of these two ports. The interesting comparison is the percent of peak achieved in the two cases. Such percentages are typical of codes ported from vector to parallel platforms. Fifty percent of peak is possible for well-vectorized code on a Cray vector system. The results here of 10 percent to 25 percent are more typical of MPP platforms.

Code	*Processors*	*CTP*	*Speedup (% of linear)*	*Mflops (% of peak)*
ARC3D	64	23,488	48.93 (76%)	98.6 (25%)
CFL3D	12	4,835	12.3 (103%)[16]	39.7 (10%)

Table 4–10. Results of porting two CFD applications from vector to parallel

Military vs. Civilian Design

Military and civilian design differ in computational requirements primarily in the military's need to run simulations that push the envelope of the aircraft's performance. This requires more parametric studies. Components of military aircraft and engines have to perform in a greater range of flight attitudes, altitudes, and speeds than do civilian aircraft. Such demands can be met by providing more computers rather than necessarily more powerful computers. Military designs will also push the solution of Navier-Stokes equations at the transonic boundary. Civilian supersonic airliners will cross this boundary relatively rarely. Military jets, on the other hand, may cross the boundary a significant number of times during a single flight. Again, this imposes a greater requirement for parametric studies. The extreme maneuvers possible in combat need Navier-Stokes solutions at high Reynolds numbers. This allows inclusion of the turbulence effects of the maneuvers.

[16] The slightly super-linear speedup for 12 processors is believed to be an anomaly associated with the memory layout on the Origin2000. The speedup for fewer processors was near- or sub-linear.

Trends for the future:

- Legacy aeronautic CFD codes will continue to use vector platforms.
- New codes are being developed for parallel platforms and some key legacy codes are being ported.

  The net result of these two points has been and will be a decline in the use of vector platforms. NASA, for example, has not replaced Cray YMP platforms at its research centers, with the exception of NASA Ames. This has been done in part to reduce operating costs by concentrating the systems at one location, but also recognizes both the initial expense in acquiring vector machines and the increasing move to parallel platforms.
- Larger 2-D and simple 3-D solutions being performed on workstation class machines.
- Clusters of workstations are becoming the platform of choice for the future.

  Workstations have seen dramatic increases in memory size and improved CPU-memory bandwidth. Many problems that were run on the vector machines due to memory requirements now fit comfortably on workstation platforms. Clusters of workstations are offering an increasingly attractive platform for CFD work, due to the improved workstation performance and the relatively low cost of the clustered environments. Experiments with clustered workstations for CFD applications have been in progress for several years. NASA Lewis, for example, has used a cluster of workstations for development of engine codes since 1992. Lewis is currently sponsoring a project headed by Pratt & Whitney to develop engine applications on clustered computing systems[17]. NASA Ames is experimenting with a 40-node cluster and has plans to expand the cluster to 500-nodes by fall 1998 [48]. Finally, NASA sponsored a recent workshop on the use of clustered computing [49].

Submarine Design

CFD techniques are used in submarine design for several purposes, including:

- reducing the noise of the submarine's passage
- improving maneuverability in shallow water
- effect of shock-wave impact on the submarine's hull
- safe and efficient torpedo launch

Comparison with Standard CFD Techniques

CFD techniques when employed under water are complicated by the incompressibility of the medium. That is, air molecules will compress as a result of pressure changes due to the passage of an aircraft, while water molecules will not compress as a submarine passes. Thus, the flow has more restrictions. Turbulence and viscous flows become important features, and are computationally expensive to model. For a submarine at a shallow-depth, an additional complication is handling the boundary at the surface of the water, in particular reflections from this surface. Finally, shallow-water operations are further complicated by the non-linear maneuvering characteristics of the submarine and by interactions with the ocean bottom.

Evolution

Table 4–11 shows the evolution of CFD work with submarine design and representative platforms used at each stage. As can be seen, the large grid sizes require large main memories

[17] http://www.lerc.nasa.gov/WWW/CISO/projects/CAN/

and disk space. The disk space requirement is used for storing checkpoint files that are viewed off-line with interactive graphics tools to monitor execution during the run and to analyze results.

Date	*Problem*	*Platform/Performance*
1989	Fully appended submarine Steady-state model ~250,000 grid elements Eüler flow (not Navier-Stokes) Non-viscous flow	Cray YMP-8 3708 Mtops
1993	Smooth ellipsoid submarine: 2.5 million grid elements Turbulent flow Fixed angle of attack Unsteady flow	C90, 4 processors, shared memory code, fully auto-tasked 5375 Mtops One week turnaround
1994	1/2 Submarine hull, without propulsion unit 1–2 million grid elements Steady flow Fixed angle of attack	C90, 1 processor 1437 Mtops Overnight turnaround
1997	Small, unmanned vehicle: 2.5 million grid elements Fully turbulent model w/ Reynolds numbers	128-node iPSC 860 3485 Mtops 120 hours wall clock
Future 1998	Full 3-D submarine: complex repulsors and appendages moving body through the water graphics to monitor during the run to identify possible errors; done off-line checkpoint files	Origin2000, 192-nodes estimated several months turn-around time 2 Gword memory 10 Terabytes disk storage 70,368 Mtops

Table 4–11. Evolution of submarine CFD applications

Interdisciplinary Problems

CFD work is being combined with other disciplines in submarine work. Structural mechanics are used to determine vibration effects on the submarine and its appendages from the flow of water. Flow solvers are being combined with adaptive re-meshing techniques to solve transient problems that occur with moving grids. The solver is also integrated with rigid body motion to simulate fully coupled fluid-rigid body interaction in three dimensions [50]. This last work has also led to a torpedo launch tube simulation that is now being integrated into a larger submarine design code to support safer and more efficient torpedo launch systems.

Another example is the impact of an underwater explosion on a submarine [51]. This work is intended to enhance the survivability of submarines. An underwater explosion generates an expanding wave of bubbles. The impact of these bubbles against the hull is greater than that of the blast alone. Tightly coupled, self-consistent models, both in structural mechanics and CFD, are required for this simulation. Complications arise from:

- different time scales
  - microseconds for blast and initial bubble formation
  - seconds for bubbles to form and expand to contact hull
- bubbles collapse non-linearly
- mapping the surface of the bubble as it expands and contracts, in particular its contact points with the hull

The current product of this work is a vector code that runs on the Cray C90 and T90. It is a three-dimensional model with the following characteristics:

- full CFD model with
  - simple explosive model
  - bubble in 3-D
  - non-linear nature of the bubbles
- body in fluid modeled with structural mechanics
  - simple geometries for the body: sphere or cylinder
  - structural response of the body
- simple explosives: a state equation

In the near term, the plan is to decouple the CFD and the structure codes. The CFD needs a fast parallel machine, but the structural code does not. Separating the two will support a better utilization of computing resources.

Separating the codes is also a first step toward the plan to use several machines to provide a more complete model. Over the next three to five years, both the explosion and the structural response will be generated from first principles, and fully appended submarines will replace the simple cylinder. It is anticipated that several machines will be needed to run such a model, including large parallel machines for the CFD and explosion codes, smaller systems for the structural response code, and graphics systems to monitor and analyze the results. In terms of current machines, the researchers are planning to couple a Cray T90, IBM RS/6000 SP, SGI Origin2000, and a graphics workstation.

Trends for the future:

Submarine design will continue to focus on several issues:

- shallow water studies—an operational demand and necessary to safely operate at high speeds in littoral seas
- coupling smaller component simulations into larger codes, as in the torpedo launch tube simulation
- coupling of structural mechanics with CFD
- coupling of explosive effects, computed from first principles
- continued need for massively parallel platforms, rather than clusters of workstations, to keep turnaround times useful.

  Underwater CFD is memory driven, as is the case with CFD in air. However, the computation requirements are larger due to water's incompressibility. When combined with the size of grids needed for the simulation of a complete submarine, the continued use of a tightly integrated platform is essential.

Surface-Ship Hull Design

CFD plays an important role in the development of surface-ship hulls and propulsors that have low resistance, high efficiency, good maneuverability, and reduced acoustic signatures. Traditionally, these objectives have been pursued through extensive testing of reduced-scale physical models and subsequent scaling of these results to full-scale ship dimensions. The goal was to predict as accurately as possible the performance of a proposed ship's design. Such

approaches have been undertaken for over 100 years [52]. Until the early 1980s, empirical methods dominated efforts to design new hulls. Over the last decade and a half, computational methods have come to play a dominant role during the design phase. Testing remains critical to development efforts, but is most important during the validation phase rather than the design phase. For example, the design of the USS *Spruance* (DD-963) during the 1970s used ten physical models. In contrast, the design of the USS *Arleigh Burke* (DDG-51) during the late 1980s employed only four physical models [53].

Surface-ship design is challenging in part because of the number of different physical phenomena at play. Unlike aircraft, ships are displacement bodies held in place by forces of buoyancy. Their movement is strongly affected by the drag forces, whose impact can be analyzed using viscous flow analysis. While aircraft dynamics are dominated by lift forces, ship performance is often dominated by viscous forces. Viscous forces are not the only forces at work, however. At greater speeds or in rough waters, wave resistance forces play a growing and sometimes dominant role. In contrast to viscous forces, wave resistance can be suitably modeled as a lift force and is therefore amenable to kinds of potential flow analysis. These calculations have been done adequately since the early 1970s on machines no more powerful than today's personal computers. Other aspects of surface-ship design reflect a mixing of lift, drag, and other forces. For example, propellers are subject to lifting forces, and the effects of pitch and blade width, diameter, and rotation speed for given thrust loading and ship speed on performance can be determined without sophisticated computing. However, additional factors complicate prediction. Fluid flow into the propeller is difficult to predict. Separation of fluid flows about the tips of the propeller increases computational demands substantially. In addition, calculations designed to minimize the acoustic noise of propellers are computationally very demanding. Rudders also are subject to lifting forces, but the fact that they are usually buried in the hull boundary layer and operate at high angles of attack in the propeller's wake make potential flow analysis suspect [52].

The most computationally demanding problems faced by ship hull and propulsion designers are those in which the viscous, rather than lifting, forces dominate. Currently, codes to solve such problems employ Reynolds averaged Navier-Stokes solutions. The full Navier-Stokes equations have been solved using direct numerical solution only for special cases at Reynolds numbers too low to be of practical interest. Computers that can do direct numerical solution at the necessary Reynolds numbers (10^9) are unavailable today. While the next stage of code evolution would be to implement Large Eddy Simulation (LES) solutions, progress in this area has not been as rapid as had been anticipated. These codes are not ready for production use.

In the United States, the principal computing platform for surface-ship hull and propulsor design has been the Cray C90. The codes typically run on a single processor. A problem of 1–2 million grid points can run to completion overnight. While ship hull design has traditionally been performed on single-processor vector pipelined machines, the migration to parallel codes is well under way. Parallel Reynolds-averaged Navier-Stokes codes will be in production use in the near future. The principal benefit of MPP platforms in the near term will be the ability to do increased numbers of parametric studies. In the longer term, a main objective is to reduce turnaround time rather than increase the size of the problem undertaken. If turnaround can be reduced to an hour or less, close to real-time coupling of empirical tests and simulation can be achieved.

Parachute/Parafoil Simulations

The Army High Performance Computing Research Center (AHPCRC) is sponsoring several parachute projects. These include:

- cargo drops from high altitude with a robot-controlled parafoil

- paratrooper drops from next-generation military transport planes
- paratrooper drops from multiple aircraft in formation

Comparison with Standard CFD Techniques

These computations differ in two significant ways from standard CFD used in aircraft design. First, the parachute or parafoil is not a rigid object. Wings and other airframe parts on aircraft can be treated as rigid objects to a first approximation, or as slowly flexing objects. A parachute, however, is a membrane that flexes frequently and often during flight. This flexing strongly affects the airflow past the membrane and is affected by the airflow. The membrane is essentially a zero thickness material. This becomes a coupled problem between the fluid flow and the membrane and requires a different set of solvers.

Second, in the case of paratroopers or cargo leaving the aircraft, two objects are moving away from each other. Each object, the aircraft and the paratrooper, has a grid. The paratrooper grid moves across the grid around the aircraft. This requires different solution techniques, specifically the use of finite element methods. In addition, it requires an ability to re-partition the mesh among the processors.

Navier-Stokes equations are solved in each grid. Finite element methods are employed, rather than finite difference or finite volume. The finite element approach has better support for irregular grids. Each grid point contains a shape function and stores geometric location information since the points move relative to each other.

The Newton-Rhaphson solution method is used. The computational burden of this approach is dependent on four factors:

- number of time steps—determined by the length of the simulation. Typically 500 to 1000.
- number of non-linear iterations to converge to a solution; a requirement of the Newton-Rhaphson approach. Typically 3 to 5.
- number of linear iterations: required to solve each non-linear iteration. Typically, 20 to 100.
- number of grid points

The time of a solution is proportional to the product of these four factors.

Only the last item, the number of grid points, is effectively under the control of the researcher; the other three are required. To date, problem sizes have been on the order of a million grid points. A parafoil solution, for example, requires about 2.3 million points, rising to 3.6 million points if the foil is flared (i.e., one or both back corners pulled down to steer or slow the foil) [54].

Current Performance

Table 4–12 shows the results AHPCRC has obtained on a one million equation problem. The higher percent of peak on the C90 does not help. These problems are driven by memory requirements. Processor performance is a secondary, though important, consideration. The low percent of peak is mitigated by the scalability of the code, allowing higher sustained performance given enough processors. AHPCRC is acquiring a 256-processor T3E-1200 to replace the CM-5 [55].

Sustained	*Peak*	*% of Peak*	*Machine*	*CTP*
3 Gflops	3.6 Gflops	~83%	C90, 12 nodes	15,875
12 Gflops	65 Gflops	~25%	CM-5, 512 nodes	20,057
33 Gflops	230 Gflops	~14%	T3E-900, 256 nodes	55,000

Table 4–12. Performance of parafoil CFD applications

Problems such as a paratrooper leaving an aircraft require a re-partition scheme for distributing grid points among the available processors. A fixed allocation of grid to processors will leave those processors near the paratrooper with more work to do than those further away. As the paratrooper moves across the aircraft's grid, the workload shifts across processors. To provide better balance, the workload needs to be redistributed by pausing the computation and reassigning grid points to processors. For the paratrooper problem, this must be done every 20 to 30 steps, or 30 to 50 times during the simulation.

Future Trends

Parachute requirements include having multiple simultaneous jumpers in the simulation. Paratrooper drops are driven by two factors: the need to keep the paratroopers close to each other to avoid spreading the troops too widely by the time they reach the ground, and the need to avoid interference between troopers. Current simulations have only one paratrooper leaving the aircraft and falling away. This is at the limit of memory and processor power of current machines. Researchers estimate it will be about five years before machines with sufficient resources will be available to solve this problem.

Parafoils that are steerable by automatic or robotic means are of interest to the Army for cargo drop operations and to NASA for use in emergency crew return vehicles for the space station. Current simulations can only model the parafoil in steady flight or in very simple maneuvers. Factors such as differing air pressures with altitude, winds, and moisture content (clouds, rain, etc.) need to be added to the simulations.

Improved re-partitioning packages are needed that can automate the process of determining when to re-partition the grid. The current technique is to select an interval based on the researcher's understanding of the problem. A better approach would be to develop metrics that support monitoring the computation and re-partitioning only when needed.

Improved graphical tools are needed to view the results of the simulations in three areas. First, the current techniques are not well suited to the unstructured grids used. Second, many data sets are too large to view as a whole. Last, viewing large data sets imposes I/O bottlenecks.

Chemical and Biological Warfare Modeling

CFD techniques are used to study the propagation of gas clouds. The gas cloud might be a chemical or biological agent. If used by the attacker, the goal is to determine the correct location, time, weather, etc., to gain the maximum advantage from the agent. For the defender, the goal is to determine the best means to localize the agent and to determine paths (escape or avoidance) around the contaminated agent [56].

A defensive application is useful to a military unit trying to determine the best path to use to avoid areas covered by a chemical or biological agent. The requirements are for tools to detect the presence of the agent, and for applications that can determine from such data the likely spread of the agent. For civil defense, particularly against terrorist attacks, the requirements are similar. For anti-terrorist use, an additional requirement is to model the spread of a gaseous agent within structures, for example large buildings and subway systems. Military use is more concerned with open-air use.

The standard technique has been to use "Gaussian puffs." These are small clouds that are Gaussian in all three dimensions. The spread of the agent is then modeled by using many of these small clouds. This technique can handle a constant turbulence field. It works well in open areas: fields, meadows, etc. A model based on this approach is in use as part of the chemical defense simulator developed at the Edgewood Arsenal (see description in Forces Modeling and Simulation, page 114). This Gaussian approach does not work well in dense areas, such as within a building or tunnel, in a city, or in a forest.

A better approach for dense areas uses a zonal model. Upper and lower zones are modeled in each room, section, etc. Gas tends to collect in the upper zone of the room, hence the need for the two levels. This is a quasi-analytical model; that is, it does not fully solve the equations, but uses models for different situations. Current systems based on this zonal approach are able to accurately predict the spread of an agent within a large building faster than real time. This raises the possibility of a defensive set of tools based on a database of buildings, subway tunnel networks, etc. In the event of an attack, the zonal model would be combined with data about the structure under attack to predict the spread of the agent and to evaluate various countermeasures, evacuation routes, etc. Had such a tool and database been readily available at the time of the nerve gas attack on the Tokyo subway, the civil response to the emergency would have been more effective.

CFD Performance Points

The figure below plots the CTP of machines used to run CFD applications examined for this study. The plot shows only those applications that have appeared since 1995 and require at least 2000 Mtops. The plateaus that appear in the plot occur at the CTP of machines in common use for CFD applications. For example, the plateau at 20,000 Mtops represents applications that run on the 16-processor Cray C90 and the 512-node CM-5. The relative lack of applications above 30,000 Mtops is due to the lack of machines currently available at that level of performance. It is expected that additional applications, or enhancements of current applications, will appear at levels above 30,000 Mtops as machines at those levels become more readily available.

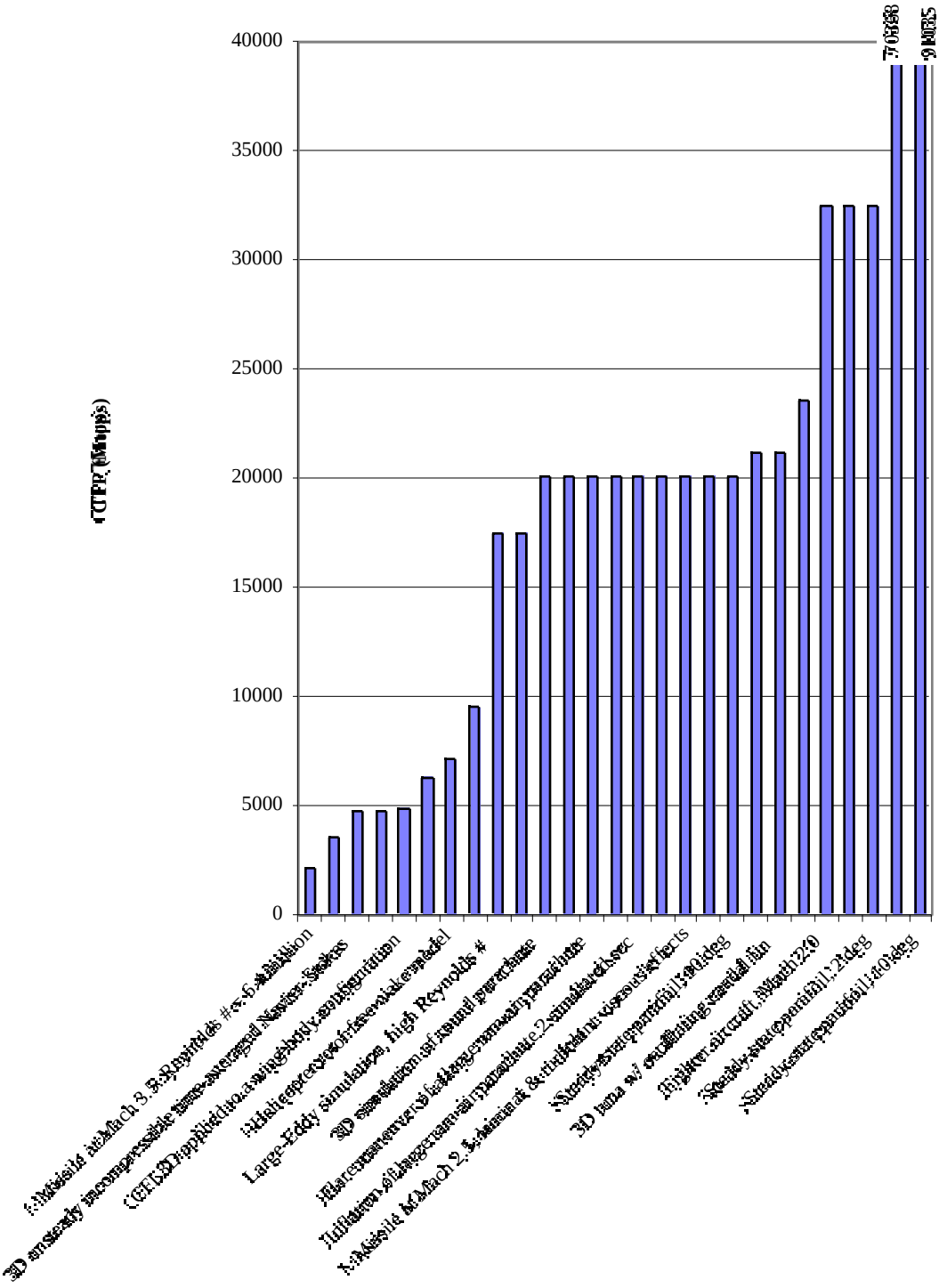

Figure 4–8. CTP for current CFD applications

Table 4–13 provides a summary of specific CFD applications examined for this study:

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
IBM 360/67	1972		18 CPU hr	First 3-D nonlinear transonic computation (STAGE II), over aircraft wing [57]	100,000 grid points
Cyber 203	1982		1-2 hours	SCRAMJET model [58]	30x109 nodes
Cray-1S	1984	195	20 CPU hr	Simulation of after-body drag for a fuselage with propulsive jet. Reynolds averaged Navier-Stokes [57]	
Cray-2	1984	1300	15 CPU m	Simulation of 2-D viscous flow field about an airfoil [57]	
Cray-2	1980s	1300	100 CPU h	Simulation of external flow about an aircraft at cruise. Steady flow. Steady Navier-Stokes simulation [59]	1.0 million grid points.
Cray XMP/48 (1proc)	1988	353	20-50 hr	Flow simulation around complete F-16A aircraft (wings, fuselage, inlet, vertical and horizontal tails, nozzle) at 6 deg angle of attack, Mach 0.9. Reynolds Averaged Navier-Stokes. Reynolds number = 4.5 million [60]	1 million grid points, 8 Mwords (one 2 Mword zone in memory at a time), 2000–5000 iterations
Cray-2	1988	1300	20 hr	Simulation of flow about the space shuttle (Orbiter, External Tank, Solid Rocket Boosters), Mach 1.05, Reynolds Averaged Navier-Stokes, Reynolds number = 4 million (3% model) [60]	750,000 grid points, 6 Mwords.
Cray YMP/8	1989	3708		Model of flow around a fully appended submarine. Steady state, non-viscous flow model [56]	250,000 grid points
Cray YMP/1	1991	500	40 CPU h	Simulation of viscous flow about the Harrier Jet (operating in-ground effect modeled) [59]	2.8 million points, 20 Mwords memory.
Cray C90/4	1993	5375	1 week	Modeling of flow around a smooth ellipsoid submarine. Turbulent flow, fixed angle of attack [56]	2.5 million grid points
CM-5/512	1995	20057	500 timesteps	Parafoil with flaps flow simulation [61]	2,363,887 equations solved at each timestep

Table 4–13. CFD applications

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
CM-5/512	1995	20057	15,000 CPU sec, 30 CPU sec per each of 500 timesteps.	Flare maneuver of a large ram-air parachute. Reynolds number = 10 million. Algebraic turbulence model [61,62]	469,493 nodes, 455,520 hexahedral elements. 3,666,432 equations solved per timestep.
CM-5/512	1995	20057		Fighter aircraft (F-18?) at Mach 2.0 [61]	3D mesh of 367,867 nodes, 2,143,160 tetrahedral elements, and 1.7 million coupled nonlinear equations solved per timestep.
Cray T3D/512	1995	32398		Modeling paratroopers dropping from aircraft. Moving grid. Cargo aircraft travelling at 130 Knots. High Reynolds number. Smagorinsky turbulence model [61]	880,000 tetrahedral elements for half of the domain.
Cray T3D/512	1995	32398	500 timesteps	Steady-state parafoil simulation. 2 deg angle of attack. Reynolds number = 10 million [61]	38 million coupled nonlinear equations at every pseudo-timestep.
Cray T3D/512	1995	32398		Fighter aircraft (F-18?) at Mach 2.0 [61]	3D mesh of 185,483 nodes and 1,071,580 tetrahedral elements
Cray J916/1	1996	450	300 CPU hr	Modeling of transonic flow around AS28G wing/body/pylon/nacelle configuration. 3-D Reynolds Averaged full Navier-Stokes solution [63]	3.5 million nodes, 195 Mwords memory
Cray YMP	1996	500	8 CPU hr, 3000 timesteps	Large-Eddy simulation at high Reynolds number [64]	2.1 million grid points, 44 Mwords
Cray YMP/1	1996	500	6 CPU hr	Modeling of transonic flow around F5 wing (Aerospatiale). 3-D Reynolds Averaged full Navier-Stokes solution [63]	442368 cells (192x48x48).
Cray C90/1	1996	1437	200 CPU hr	Simulation of unsteady flow about an F-18 High Alpha Research Vehicle at 30, 45, 60 deg angle of attack [65]	2.5 million grid points for half-body modeling. 40 Mwords memory
Cray C90/1	1996	1437	19 CPU hr	Simulation of turbulent flow around the F/A-18 aircraft at 60 degree angle of attack. Mach 0.3. Reynolds number = 8.88 million [66]	1.25 million grid points, 100 Mwords of memory.

Table 4–13. CFD applications (cont.)

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
Cray C90/1	1996	1437	3 CPU hr	Modeling of flow over a blended wing/body aircraft at cruise [67]	45 Mwords of memory
Cray T3D/16	1996	2142	20,000 CPU sec. 50 CPU sec x 400 steps	Aerodynamics of missile at Mach 3.5. Reynolds number = 6.4 million [68]	500x150 (75,000) elements in mesh. 381,600 equations solved every timestep.
SGI PowerChallenge (R8000/150)/16	1996	9510	3.6 hr, 3000 timesteps	Large-Eddy simulation at high Reynolds number [64]	2.1 million grid points, 44 Mwords
Cray T3D/256	1996	17503	52,000 CPU sec.130 CPU sec x 400 timesteps	Aerodynamics of missile at Mach 2.5, 14-degrees angle of attack for laminar and turbulent viscous effects [68]	944,366 nodes and 918,000 elements. 4,610,378 coupled nonlinear equations solved every timestep.
Cray T3D/256	1996	17503	2 hr, 3000 timesteps	Large-Eddy simulation at high Reynolds number [64]	2.1 million grid points, 44 Mwords
CM-5/512	1996	20057	500 timesteps	Steady-state parafoil simulation, 10 deg angle of attack, Reynolds number = 10 million [54]	2.3 million equations every timestep
CM-5/512	1996	20057	500 timesteps	Inflation simulation of large ram-air parachute, Box initially at 10 deg angle of attack and velocity at 112 ft/sec, 2 simulated seconds [69]	1,304,606 coupled nonlinear equations solved every timestep.
CM-5/512	1996	20057		Flare simulation of large ram-air parachute [69]	3,666,432 coupled nonlinear equations solved every timestep.
CM-5/512	1996	20057	7500 CPU sec = 50 CPU sec x 150 timesteps	Aerodynamics of missile at Mach 2.5, 14 deg angle of attack for laminar and turbulent viscous effects [68]	763,323 nodes and 729,600 elements. 3,610,964 coupled nonlinear equations solved in each of 150 pseudo-time steps.
Cray C916	1996	21125		3-D simulation of flow past a tuna w/ oscillating caudal fin. Adaptive remeshing. Integrated with rigid body motion [50]	

Table 4–13. CFD applications (cont.)

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
CM-5/512	1996	20057		3-D simulation of round parachute, Reynolds number = 1 million [70]	
IBM SP-2/32	1996	4745	80 minutes	3-D unsteady incompressible time-averaged Navier-Stokes. Multiblock transformed coordinates [71]	3.3 million points
CM-5/512	1996	20057		3-D study of missile aerodynamics, Mach 3.5, 14 deg angle of attack, Reynolds number = 14.8 million [72]	340,000 element mesh, nonlinear system of 1,750,000+ equations solved every timestep.
iPSC860/128	1997	3485	5 days	Small unmanned undersea vehicle. Fully turbulent model with Reynolds numbers [56]	2.5 million grid points
CM-5/512	1997	20057	30 hours	Parafoil simulation [54]	1 million equations solved 500 times per run
Cray C90/16	1997	21125	9 months	3-D simulation of submarine with unsteady separating flow, fixed angle of attack, fixed geometry [56]	
Origin2000/64	1997	23488		ARC3D: simple 3-D transient Eüler variant on a rectilinear grid [46]	
Cray T3E-900/256	1997	91035	500 timesteps	steady-state parafoil simulation, 10 deg angle of attack, Reynolds number = 10 million [54]	2.3 million equations every timestep
Origin2000/12	1997	4835	4329 sec	CFL3D applied to a wing-body configuration: time-dependent thin-layer Navier-Stokes equation in 3-D, finite-volume, 3 multigrid levels [47]	3.5 million points
IBM SP-2/64	1997	7100	2 to 4 hours	Helicopter rotor free-wake model, high-order vortex element and wake relaxation [40]	
IBM SP-2/32	1997	4745		Helicopter rotor motion coupled with rotor CFD, predict 3-D tip-relief flow effect, parallel approximate factorization method [40]	
IBM SP-2/45	1997	6300	2 to 4 hours	Helicopter blade structural optimization code, gradient-based optimization technique to measure performance changes as each design variable is varied [40]	up to 90 design variables, one run per variable
Origin2000/192	1998	70368	months	3-D simulation of submarine with unsteady flow, fixed angle of attack, and moving body appendages and complex repulsors [56]	

Table 4–13. CFD applications (cont.)

CFD Summary and Conclusions

The following conclusions can be drawn about CFD based on the preceding discussion:

- CFD is a candidate application area of national security interest, especially in the submarine and military aircraft design areas. CFD represents primarily a research use of high-performance computing. A rapid-response defense against gas attacks, if created, would be an operational use.
- Grid sizes demand large memory size—the dominant reason for porting codes from PVP to MPP platforms.
- CPU-memory bandwidth bottleneck is the primary deterrent to achieving good CFD performance on MPP platforms. This is exacerbated by the need to optimize for cache size on the RISC-based processors.
- Need to couple CFD with structures—both membrane and rigid-body. This coupling is currently a driving requirement in aeronautics and parachute work.
- 2-D and simple 3-D calculations now possible on workstation-class platforms, if sufficient memory is available.
- Larger 3-D calculations minimally possible on some mid-range (2000 to 5000 Mtops) if turnaround time requirement sufficiently long and sufficient memory is available.
- Clusters of workstations becoming a viable tool in aircraft and engine design. This is driven by the economics of PVP and MPP platforms vs. the workstation clusters.

Computational Chemistry and Materials

Computational chemistry, also known as molecular or particle dynamics, involves the study of the interactions of atoms and molecules. The interactions can span a wide range of forces. The particular field of study determines the forces studied in a computational run.

Particle Dynamics

Particle dynamic simulations mimic the motion of atoms. Atoms may be bonded to other atoms to create molecules of varying size and complexity. The initial conditions for the simulation include the spatial coordinates of every atom, the bonds between the atoms making up the molecules, and the initial energies of the atoms (i.e., the atoms may already be in motion at the start of the simulation). The simulation iteratively computes the total potential energy, forces, and coordinates for each atom in the system. These computations are repeated at each time step. The potential energy involves both bonded and non-bonded terms. Bonded energy terms derive from the energy contained within chemical bonds and the motion of the atoms around the bond. Thus, the bonded energy terms consist of bond energy, angle energy, and dihedral-angle energy. The non-bonded energy terms derive from the forces that act between all atoms, whether bound together or not. These include van der Waals potentials (the attractive force between atoms or molecules of all substances) and the potentials that result from the electrical attraction and repulsion due to unbalanced electric charges.
Once the potential energy is computed for each atom, the forces acting on the atom are used to update its position and velocity. The new location and potential energy for each atom are stored and the process is then repeated for the next iteration.

The non-bonded energy terms account for most of the calculation effort. The non-bonded energy of one atom is dependent on all the atoms in the simulation, both those nearby and those far away. This results in an *n*-squared problem where *n* is the number of particles in the simulation. By contrast, the bonded energy terms are relatively easy to compute, since each

atom is bound to a small number of nearby atoms. Large organic molecules complicate this picture somewhat. For example, the molecules used in protein-folding simulations have at least 10,000 atoms. Since the protein can fold in a number of ways, it becomes non-trivial to determine which atoms are in "nearby" folds.

The algorithmic efforts in particle simulations have primarily been directed at the non-bonded energy terms. As the number of particles in the simulation grows, the effort to compute the non-bonded energy at each step potentially grows as n^2. Simulations involving hundreds of millions of atoms are not uncommon, yet such numbers are small relative to Avogadro's number.[18] The usual technique employed to reduce the computation time is to ignore the contribution of atoms far away. Atoms beyond a certain distance are either not included in the computation or are included only as an aggregate value representing all distant atoms in a given direction.

The number of particles and the computational requirements combine to make parallel machines the primary choice for particle simulations. The straightforward approach is to divide the simulation space evenly across the processors, with each processor performing the calculations for particles that are within its space. For simple simulations, this usually suffices. Each processor will communicate with other processors on each time step to get updated information about the other atoms. For simulations that ignore the effects of distant atoms, each processor need only communicate with those processors nearest in the simulation space. These communications have to convey the locations of all the atoms for each processor, since each of these atoms needs to be considered in computing the updated energies for the atoms the processor is responsible for. When effects of distant particles are included, the communication increases. However, it does so gradually. Only aggregate position and energy values need to be communicated by each processor to the distant processors. In some simulations, the processors are grouped into regions and only the data for the region is sent to the distant processors.

This simple view is inadequate when the particles in a simulation may be subject to widely varying densities across the simulation space, for example a simulation where particles are attracted to electrodes. A simulation that starts with an even distribution of particles across the region, and a balanced workload for the processors, will gradually find more particles being handled by the processors that are near the electrodes, while other processors away from the electrodes are left idle. Such load imbalances can be corrected by varying the region for which each processor is responsible. Changing the region size and redistributing the particles takes time. Thus, the frequency of the load balancing has to be matched in some way with the motion of the particles. When the load balancing is not done often enough, processors are idle too much of the time. When the load balancing is done too often, the load balancing itself begins to use too much computation time.

Explosions

In addition to considering the classical dynamics interactions of particles colliding (as described above), explosions must consider the chemical changes in the molecules themselves. These changes release energy into the system, resulting in a reaction wave that travels through the explosive material at hypersonic speeds, ranging from 1–10 km/sec. Molecular simulations of the shock wave propagation have been done since the 1960s. Studies of shock-initiated systems have been done since the 1970s. However, these were very simple models, and not able to do justice to the complex chemical processes present when using modern explosives.

DoD is a major source of research funds for explosive simulations. The Army, for example, is studying solid chemical propellants used to fire projectiles. The objective is to

[18] Avogadro's number is the number of particles (6.023×10^{23}) present in an amount of a compound that is equal to the atomic weight. For example, 12 grams of carbon (just under one-half ounce) would have 6.023×10^{23} carbon atoms.

propel the shell from the gun with minimal damage to the barrel. Some work is being done on non-solids, for example the use of liquid propellants that are inert until mixed in the barrel. The Federal Aviation Administration provides some support in the area of fuel-air explosive mixtures for the purpose of designing better aircraft fuel systems.

An ability to simulate explosions at the molecular and atomic level can provide the following benefits:

- safer experimentation through use of simulation to reduce the number of actual tests
- improved safety handling for explosive materials
- understanding explosions at a level not observable in live tests
- tailoring explosives for specific purposes

An explosive simulation solves two basic sets of equations. The first set is the classical equations of motion for the *N* atoms present in the system. These equations provide the kinetic energy for the system. The second set is the potential energy within the molecule in the form of the inter-nuclear distances and bond angles. The potential energy is the most computationally intensive part of the simulation.

A good explosive simulation must first be able to model the complex organic molecules that make up today's chemical explosives. Second, imperfections in the crystalline structure of the explosive are believed to contribute significantly to the explosion. Hot spots that result from shear stress and defects in the crystal are thought to play an important role in detonation. Third, differences in compression of the crystal and initiation of the shock wave need to be represented. Realistic models need to simulate molecules of 20 to 30 atoms each and have at least 30 to 40 thousand such molecules in a three-dimensional model [73]. Current models still fall well short of this goal.

Within the past five years, two-dimensional models have been developed. These models use diatomic molecules, and have up to 10,000 molecules in a simulation. The model will initiate a shock wave at one end of the two-dimensional grid of molecules, then follow the shock wave through the material. One of the difficulties is having a long enough crystal in the direction of the shock wave. The crystal will undergo several distinct phases as the shock wave approaches and passes each section. First, the molecules are disturbed from their initial positions. Then they undergo disassociation. Simulations have suggested that the molecules form intermediate polyatomic products that then decompose, releasing heat. Finally, the end products of the reaction are formed as the tail of the shock wave passes. Today's simulations, while small, are already showing unexpected results in the vicinity of the shock wave. The polyatomic intermediate products, for example, have not been observed in actual live explosions.

Current simulations have well-developed classical mechanics effects. Additional work is underway in the area of quantum mechanics. An area of intense research is how to fit the quantum mechanics into the simulations. In an explosion, there are too many variables to be able to solve the equations adequately. Quantum mechanical reactions in solids have not been done; researchers are currently working from first principles. Researchers at the Army Research Laboratory, Aberdeen, are using Gaussian94 [74]. It is designed to handle quantum mechanical effects—molecular structures, vibrational frequencies, etc.—for molecules in the gas phase and in solution. ARL is adopting the code for use with solid explosives.

Current simulations are running on platforms that generally are below the current export thresholds. For example, an ARL two-dimensional code that includes only classical dynamics [75]:

1992 IBM RS6000 model 550, single processor, one month run (less than 500 Mtops)

1997 SGI PowerChallenge, 4 processor, overnight run (1686 Mtops)

The PowerChallenge code is moderately scalable to 4 processors. It continues to see shorter run times to 16 processors; however, the growth curve is decidedly sub-linear. Above 16 processors, the code exhibits negative growth. In part, this is a load balancing issue, since a large amount of work is located within the shock wave and the shock wave is moving rapidly through the crystal. It is difficult to concentrate sufficient processor power on the area of the shock wave without encountering excessive communication costs.

A primary computational consideration for these simulations is storage space, both in main memory and secondary storage. The space requirement scales with the number of atoms for the classical dynamics. For the quantum mechanics, the space requirement scales as the number of electrons to the fifth power (n^5). The quantum mechanics integrals are used repeatedly during a simulation. The common technique is to compute them once, then store the solutions for use during the simulation. These temporary files can occupy 30 to 40 Gigabytes of disk space on workstation implementations. Historically, the disk space requirement has driven the use of vector machines for these simulations. Now, desktop and small parallel systems can be used if equipped with sufficient disk space.

A secondary problem with scaling to larger problems is a visualization problem. There are no codes suitable for viewing a three-dimensional explosion. Most visualization tools today are oriented toward volumetric data rather than particle data. The primary technique for viewing large numbers of particles is to reduce the data set by eliminating some fraction of the particles, or combining them into representative particles. With an explosion, this is not sufficient, since the researcher is particularly interested in what is happening in the area of the highest concentration and greatest volatility; i.e., the shock wave.

Performance Measurements

Table 4–14 shows the performance of a variety of computational chemistry problems over time. The length of a run is generally dependent on two factors: the time the researcher has available on the platform and his patience. Disk storage space is a third, less significant factor. The more disk space available, the more often the simulation can save the current state for later visualization.

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
Cray XMP/1	1990	353	1000 hours	Molecular dynamics of bead-spring model of a polymer chain [76]	Chain length=400
Cray YMP	1991	500	1000 hours	Modeling of thermodynamics of polymer mixtures	lattice size - 112^3 chain size = 256
Cray YMP	1991	500	1000 hours	Molecular dynamics modeling of hydrodynamic interactions in "semi-dilute" and concentrated polymer solutions	single chain 60 monomers
Cray C90/2	1993	2750	12269 sec	Determination of structure of E. coli trp repressor molecular system [77]	1504 atoms 6014 constraints
CM-5/512		20057	8106 sec		
Cray C90/2		2750	117 sec		530 atoms 1689 distance & 87 dihedral constraints
CM-5/256		10457	492 sec		
Cray C90/1	1993	1437	.592 sec/ timestep	MD simulation using short-range forces model applied to 3-D configuration of liquid near solid-state point [78]	100,000 atoms
Intel iPSC860 64 nodes		2096	3.68 sec/ timestep		1 million atoms
Cray T3D/256		17503	1.86 sec/ timestep		5 million atoms
Intel Paragon/1024		24520	.914 sec/ timestep		5 million atoms
Intel Paragon/1024		24520	.961 sec/ timestep		10 million atoms
Intel Paragon/1024		24520	8.54 sec/ timestep		50 million atoms
Cray T3D/512		32398	.994 sec/ timestep		5 million atoms
Cray T3D/512		32398	1.85 sec/ timestep		10 million atoms
IBM SP1/64	1993	4074	1.11 sec timestep	Molecular dynamics of SiO_2 system [79]	0.53 million atoms
Intel Touchstone Delta/512		13236	4.84 sec/ timestep		4.2 million atoms
Paragon 256	1995	7315	82 sec/ timestep	Particle simulation interacting through the standard pair-wise 6-12 Lennard-Jones potential [80]	50 million particles

Table 4–14. Computational chemistry applications

Paragon 512	1995	13004	82 sec/ timestep		100 million particles
Paragon 1024		24520	82 sec/ timestep		200 million particles
Paragon 1024		24520	82 sec/ timestep		400 million particles
SGI PowerChallenge 4 nodes	1997	1686	overnight	Explosion simulation [73]	30 thousand diatomic molecules

Table 4–14. Computational chemistry applications (cont.)

Conclusions

The discussion of computational chemistry leads to the following conclusions:

- There are applications of national security interest in computational chemistry; in particular, the work with explosives.
- Computational chemistry requires very high levels of computing power. This is largely a memory-driven area, due to the number of particles desired and the n^5 requirement of quantum mechanics.
- Larger problem sizes are needed. The particle count in current simulations is still miniscule compared to Avogadro's number. Modeling realistically useful explosive devices, for example, requires orders-of-magnitude larger computing systems.
- Small systems can substitute for large computing systems on mid-size problems, but only if the researcher is willing to wait long enough (i.e., months) for the results, and if there is sufficient memory and secondary storage (at least tens of Gigabytes) available on the system.
- Considerations other than raw performance, such as visualization tools and development of appropriate algorithms to handle quantum mechanics, are also on the critical path to better chemistry simulations.

Computational Structures and Mechanics

Since the military use of gunpowder started in Western Europe, there has been a continuing race between the development of structures that are able to withstand high-velocity projectiles and improved projectiles designed to penetrate such structures. The advent of armored ships during the last century and the development of armored vehicles in this century have extended this race to include structures on the move. As a result, the military has long been keenly interested in the study of structural mechanics. In the past, this study has been carried out through extensive, costly, and dangerous field experiments. These experiments are still necessary today; however, the growing power of high-performance computers has given rise to the use of Computational Structural Mechanics (CSM) to both supplement and extend the results of field experiments.

Two specific application areas that arise from this work were studied for this project: the response of earth and structures to the effects of explosive loading and the impact of projectiles in earth, structures, and armor (also known as armor/anti-armor). In both, a judicious combination of experiment and simulation is used to characterize the dominant physical phenomena. In addition, simulation provides insight into transformations occurring

during the experiments that cannot be observed due to the extreme conditions. Finally, simulation helps direct subsequent field experiments.

Survivability of Structures

Structure survivability involves research in four areas:

- weapons effects,
- structural dynamics,
- material mechanisms, and
- structural design.

This work investigates the offensive use of weapons to penetrate, damage, and destroy buildings. It also looks at the defensive features of buildings that make them more resistant to the effects of various types of weapons.

During the Cold War, much of the effort in this field was devoted to studying the effects of nuclear blasts and how to design structures to withstand such explosions. Since the end of the Cold War, and in particular since the Gulf War, the work has focused more on the effects of conventional munitions and the ability of structures to withstand such explosions.

Two weapons effects are studied. The first is the effect of conventional explosions on various types of structures. The second is the ability of various types of projectiles to penetrate through layers of natural and man-made materials.

Conventional Explosions

Conventional explosions are described in the previous section, beginning on page 96. To understand their effects on structures, a finite volume technique is used with a structured grid instead of modeling individual molecules. The technique follows the strong shock wave interference as it moves outward from the source of the explosion through the grid of cells. The pressure can be determined from the shock wave front at any point as the wave moves outward. Where the shock wave encounters the ground or a structure, the pressure is used to determine how the ground or building will respond to the blast.

To accurately calculate the pressure, a very fine mesh is needed in the vicinity of the shock wave. However, it is not feasible to use such a fine mesh throughout the problem space due to both the computational and memory requirements that would be imposed. Therefore, as the shock wave moves outward, the grid needs to be re-meshed to keep a sufficiently fine mesh near the shock front. The current state-of-the-art is to re-mesh after a predetermined number of timesteps. Adaptive meshing techniques that automatically readjust the mesh when needed are a current area of study.

The time integration step for studying explosions or high-strain-rate material behavior is very small. Typically, numerical stability criteria limit time steps to the sub-microsecond region. The total time simulated is based on the duration of the explosion or impact event. Many conventional air-blast simulations are run for tens of milliseconds, while high-velocity projectile impacts are on the order of several hundred microseconds. There is a direct relationship between the time scale and the resolution (i.e., the size of the cells used in the finite volume technique). Since the computation is following the shock wave through the cell, a time step is needed such that the wave does not travel through more than one cell. Thus, doubling the resolution in each of three dimensions and halving the time step raises the computational workload by a factor of 16.

CTH is one code used for modeling the expansion of the shock wave outward from an explosive device. CTH is an Eülerian hydrodynamic code, uses a structured mesh, and computes shear stresses. It provides the pressure due to the shock wave in any of the cells

during the simulation. This pressure can then be used to determine the effect of the blast on the ground and building(s) in the vicinity of the explosion. CTH is used in nuclear and armor/anti-armor applications (see below for both) as well as structural problems.

Originally developed for use on PVP processors, CTH took advantage of microtasking to utilize the available processors. It has been ported to a variety of systems from a single-CPU PC to the largest MPP machines (i.e., SGI Origin2000, Cray T3E, and ASCI Red). It uses a strictly data parallel programming model. The main difficulty in the parallel port was determining the domain decomposition strategy. Each processor is assigned a sub-domain (50 x 50 x 50 is typical). A set of ghost cells surrounds the sub-domain. The processor executes within its sub-domain as though it were the only processor. Message passing is used to exchange data from the ghost cells between adjacent sub-domains [81].

Structural Response

Modeling the effect of an explosion on a building imposes two fundamental problems. First, the explosive shock wave requires microsecond time scales and the building response uses a seconds-to-minutes time scale. It may take 10 to 100 microseconds for the shock wave to reach and surround a building, several seconds for the structure of the building to begin to respond to the shock wave, and up to a minute for the building to collapse. The one exception to the time scale difference is the shock wave's effect on glass, which shatters on a microsecond time scale.

Second, different techniques are used to model structural response and shock wave propagation. Structural response uses a non-linear Lagrangian code operating on a finite element (FE) mesh. DYNA3D is a code used extensively in a variety of structural applications. For use in survivability calculations, the code has incorporated numerical models, which better reflect the use of brittle materials such as concrete and mortar that are often present in military and other hardened structures.

Ideally, the conventional explosion and the structure simulations should be coupled. This would allow a more accurate and realistic determination of the effect of a blast on a structure. Work is proceeding on coupling the CTH and DYNA3D codes. An approach being used is to develop a combined solver, the Arbitrary Lagrangian Eüler (ALE) formulation currently under development by DOE as part of the ASCI project. Estimates are that it will take four to five years to get the coupling accomplished. This time frame is due to the difficulty imposed by the different solution techniques and to the realization that no current machine is capable of performing the calculation. It is expected that advances in computing technology will provide such a machine within five years.

As a representative example of this area, a simulation of an explosion engulfing a set of buildings requires [82]:

- 12 milliseconds simulated time
- Cray C916, used 14 processors
- all the memory, 825 Mwords
- 3-day continuous run, ~325 CPU hours
- DYNA3D analysis to study effects on window glass and doors was done offline after the blast simulation was completed

No structural mechanics were factored into this simulation. A much longer simulated time would be required to handle structural response, since the buildings respond much slower than the explosion. Glass, however, is different, as it blows out almost instantaneously. The glass effects are computed off-line after the simulation. Thus, in this experiment, the

explosion simply moves up to and begins to surround the buildings. Additional examples are given in Table 4–15, page 108.

Impact of Projectiles on Armor

Armor/anti-armor simulations study the effectiveness of armor protection for vehicles and buildings and the lethality of projectiles designed to penetrate such armor. Traditionally, this work has been based on actual field experiments. Over the past two decades, computer simulations have become increasingly useful in these studies. Initially, the simulations attempted to replicate the results of field experiments. These early simulations were two-dimensional and could model only simple impacts, such as a projectile hitting the armor at a right angle. As better computers have become available, simulations have been expanded into three dimensions and are now able to model impacts at a variety of angles. The simulations also model more realistic armor, including multi-layer armor, and simulate targets that reflect the curvature and slope of actual armor. Today, the simulations are helping to shape the field experiments that are conducted.

Penetration Mechanics

The impact of a high-velocity projectile on an armor target generates a hypersonic shock wave that moves outward from the point of impact through the material of the armor. The shock wave will deform both the armor and the projectile. Approximate analytical methods are available for studying this problem, but suffer from necessary simplifying assumptions. If a computational approach is to yield a satisfactory and complete description of the impact and its effects, a numerical solution is essential. The numerical approach uses finite difference and finite element methods in finding solutions to the partial differential equations. If a complete description is desired, the solution must account for the geometry of the projectile and target, the materials of which each is composed, and the propagation of the hypersonic shock wave that moves outward from the point of impact. This hypersonic shock wave is very similar to the shock wave in air that was explained above in the description of the structure survivability work. The shock wave causes dramatic changes in the material comprising the armor, including changes of state that require the inclusion of hydrodynamic forces. Finally, the projectile and target will deform under the impact, and the initiation and propagation of this deformation must be included in the numerical approach [83].

Computational Constraints

Solutions to such penetration mechanics problems are both computation and memory intensive. A fine scale mesh is needed to accurately represent the hypersonic shock wave moving outward through the armor target and the projectile. Increasing the resolution of this mesh improves the accuracy of the simulation. However, the same problem is encountered here as in the explosive study described above. To double the resolution requires a 16-fold increase in computational cycles, and an 8-fold increase in the number of cells in the mesh.

A variety of effects need to be included in the computation:

- hydrodynamic forces
- propagation of the shock wave
- deformation of the projectile and target
- materials comprising the armor and projectile

All of these contribute to the time needed to perform the calculations on each timestep.

The early (1970s) simulations in this field modeled two-dimensional impacts with the projectile impacting a flat target at 90 degrees, not a realistic combat situation. Armor faces

are both curved and sloped in an effort to deflect the projectile and diminish its effectiveness. To be realistic, a simulation must take into account the actual shapes used for armor and the likelihood that the shell will impact at an angle other than 90 degrees.

The transition from 2-D to 3-D required large increases in the memory and time. 2-D simulations require a few tens of thousands of cells. The early 3-D simulations started at half a million cells. Today, simulations using tens of millions of cells are common, with demonstration projects using up to one billion cells.

Parametric Studies

Much of the real benefit in running these simulations comes from parametric studies. For example, a series of runs might vary the angle of impact, the velocity of the projectile, and/or the composition of the target or projectile. It is the need for parametric studies that results in the gap between demonstration projects that are run once as a proof-of-concept, and the size of the more common projects where multiple runs can be made.

Performance Points and History

In the 1970s, the Cyber machines supported 2-D simulations involving tens of thousands of cells. These simulations had simple projectile and target geometries and included some elastic constituative models. The limiting factor in the transition to 3-D simulations was insufficient memory.

The 256 Mword memory of the Cray-2 provided sufficient memory to begin doing 3-D simulations involving 0.5 to 1.5 million cells. At first, the simulations were run using only one processor, with the other three idle to allow access to the entire memory of the machine. Later, the code (CTH) was modified to take advantage of Cray's microtasking to allow all four processors to work on the simulation.

Today, a transition is in progress to MPP platforms. The CTH code described above is the primary code for this work. Workstation versions of the code are sufficient for 2-D computations—an example of an application that used to require the highest level of computing, but now runs on much smaller platforms. 3-D simulations are being run on MPP platforms with turn around times of 100 to 400 hours.

Future Work

- Researchers typically limit computational runs to between 100 and 400 hours. This is a reflection of the computer resources currently available and a "threshold of pain" issue on how long to wait for results from a single run.
- As additional computing resources come on line, researchers will use them in two ways. One is to increase the resolution used in the simulation. The second is to simulate more complex (i.e., realistic) targets and projectiles.
- More parametric studies are needed. Currently, researchers are able to do only three to five runs for a parametric study. They need to be able to do tens to hundreds of runs.

Computational Geomechanics

Computational geomechanics simulates soil-structure interactions. Example simulations include vehicle-ground interaction, pile driving, and mass movements by avalanche. In the context of national security issues, an example application is the "Grizzly," a special-purpose M-1 tank chassis that will serve as a breaching tool to clear paths through mine fields, wire, rubble, and other obstacles. A plow attached to the Grizzly is being designed to clear a path by lifting anti-personnel mines and pushing them to either side. Computer simulation potentially allows the testing of the plow in a variety of soil and terrain conditions, and against a variety of mine types. Benefits of the simulations include improvements in the design of the

plow and testing of the plow in various conditions without the need and cost to transport the prototype tank to different locations.

Soil-structure interactions can result in the soil behaving as though in different "phases" that mirror the solid-liquid-gas phases that matter typically occupies. For the "solid" phase, the soil behaves as a large solid, or a small number of large solids. For the "liquid" phase, such as occurs in an avalanche or a mudslide, the soil behaves very much like a liquid flowing in a current. For the "gas" phase, the soil behaves as individual particles moving independently. No one model is currently able to handle these different phases [84].

The mine-plow application models the soil as independent particles that interact with each other when they come into contact; a technique known as a Discrete Element Model (DEM). The technique has similarities with that described earlier for molecular dynamics. It is based on the application of Newton's laws of motion to the individual particles. Unlike molecular dynamics, only those particles in actual contact with the particle of interest can affect its motion. The particles used in these simulations are much larger than the atoms and molecules used in molecular dynamics. Particles here can be as small as individual grains of sand or other soil particles and range upwards in size to represent rocks or mines.

The acceleration of each particle for a specific time step is found by first computing the total force on the particle. Once the force is known, Newton's laws of motion are integrated across the time step to determine the current velocity of the particle. The total force acting on a particle is the combination of two types of forces. The first type is in the direction of the particle's motion and due to physical contact with other particles. This force is treated very much like a spring, either compressing as the two particles move closer together or releasing as the particles separate. The second type of force occurs in directions perpendicular to the motion of the particle. This force is also modeled as a spring, but with the addition of a frictional slider.

For the plow simulation, the overall computation uses three steps:

(1) calculating the force applied by the plow blade and boundary conditions,

(2) calculating the interaction forces between particles, and

(3) integrating to obtain the revised motion of the particles.

One of the largest computational burdens is imposed by the need to determine which particles are in contact with each other. The simplistic approach of looking at all possible particle-particle pairs leads to an untenable *n*-squared problem. A better, linear solution uses a cell method of dividing the particles in a three-dimensional grid. Particles that touch a particular particle can then be located only in the particle's own cell, or in cells that immediately adjoin the particle's cell. A neighbor's list is updated for each particle, and the cell size is set large enough to handle the largest particle present in the simulation.

Domain decomposition on parallel platforms is done in the two horizontal directions, creating a set of columns, each of which is assigned to a processor. Care has to be taken when assigning particles to cells as the simulation progresses to avoid having a small numerical error result in the particle seeming to be in two cells at once. Having the owning cell determine ownership for the new position of the particle solves this.

Current runs are examining tens of thousands of particles, typically 40,000 to 100,000. The need is for runs that can handle several million particles and to perform a parametric study. The work over the next year will examine different soil conditions, mine types, and other obstacles within and on top of the soil, and the effect of varying terrain.

In addition to the computational resources, researchers are looking for solutions to the analysis of the simulations. Visualization techniques do not adequately support particle counts in the million-plus range. Reduction algorithms are needed to lower the number of particles actually rendered. These reduction algorithms can be time consuming in their own right.

Attempts to do near real-time simulations with small numbers of particles have required almost as much computing power for the reduction algorithms as for the particle dynamics. In off-line analysis, smaller computers can be used for the data reduction, but will add considerably to the total time required to produce the visualization.

Performance Points

Table 4–15 shows performance points for a number of structural mechanics problems. Historical data is included to show the progression of problem sizes and computational requirements over time.

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
Cyber	1980			2-D modeling of simple projectile striking simple target [85]	10,000s of grid points
Cray-2/1	1987	1300	400 hours	3-D modeling of projectile striking target. Hundreds of microsecond time scales [85]	.5–1.5 million grid points 256 Mword memory
Cray YMP/1	1990	500	39 CPU sec	Static analysis of aerodynamic loading on solid rocket booster [86]	10,453 elements, 9206 nodes, 54,870 DOF[19] 256 Mword memory
Cray YMP/2		958	19.79 CPU sec		
Cray YMP/4		1875	10 CPU sec		
Cray-2	1992	1300	5 CPU hours	Modeling aeroelastic response of a detailed wing-body configuration using a potential flow theory [86]	
Cray-2	1992	1300	6 CPU days	Establish transonic flutter boundary for a given set of aeroelastic parameters [86]	
Cray-2	1992	1300	600 CPU days	Full Navier-Stokes equations [86]	
C916	1995	21125	900 CPU hours	Hardened structure with internal explosion, portion of the overall structure and surrounding soil. DYNA3D, nonlinear, explicit, FE code. Nonlinear constitutive models to simulate concrete & steel [87]	144,257 solid & 168,438 trust elements for concrete & steel bars, 17,858 loaded surfaces, 500,000 DOF, 60 msec simulated time
Cray C912	1996	15875	435 CPU hours	Water over C4 explosive in container above wet sand, building at a distance [88]	13 million cells, 12.5 msec simulated time
Cray C912	1996	15875	23 CPU hours	Water over C4 explosive in container above wet sand, alternate scenario: container next to building, finite element [88]	38,000 elements, 230 msec simulated time, 16 Mwords memory
Cray C916	1996	21125	325 CPU hours 3 day continuous run	Explosion engulfing a set of buildings, DYNA3D analysis to study effects on window glass & doors done off-line after the blast simulation completed [82]	825 Mwords memory

Table 4–15. Structural mechanics applications

[19] DOF = degrees of freedom

workstation	1997	500		2-D modeling of simple projectile striking simple target [85]	10,000s of grid points
Cray T3E-900 256 nodes	1998	91035	450,000 node hours	Grizzly breaching vehicle plow simulation, parametric studies, different soil conditions & blade speeds [89]	2 to 4 million particles (soil, rock, mines, obstacles, etc.)

Table 4–15. Structural mechanics applications (cont.)

Conclusions

The preceding discussion leads to the following conclusions:

- High-performance computing in combination with field experiments shortens the research and development time in CSM applications. As such, it provides a competitive advantage that makes these applications candidate applications of national security interest.
- Importance of parametric studies cannot be overemphasized, particularly in armor/anti-armor work. An order of magnitude increase is needed in the number of parametric runs.
- Coupled applications are not yet feasible in explosive studies. Differences in time scales between the millisecond scale of the explosion and the second scale of the structural response to the explosion are the principal problem to overcome. The coupled application also requires more memory and computer time than current systems provide. An order of magnitude increase in computer time would be necessary with current systems, or an order of magnitude improvement in computer performance.
- Greater memory sizes lead to greater resolution in simulations. Doubling the resolution for a three-dimensional problem requires an eight-fold increase in memory size.

Computational Electromagnetics and Acoustics

Introduction

Computational electromagnetics and acoustics is the study of the electromagnetic (light and radar) and acoustic (sound) properties of physical objects by finding multidimensional solutions to the governing electromagnetic or acoustic equations. Such studies find applications in the design of tactical vehicles and antennas, the development of laser systems, and the detection of buried munitions such as mines. Much of the attention in this area is focused on determining how "visible" an object such as an aircraft or a submarine is to various forms of electromagnetic or acoustic sensing devices. Some studies focus on the nature of electromagnetic or acoustic energy generated from an outside source and *reflected* by the object in question. Other studies focus on the nature of the *emitted* energy of an object, for example the way the vibration caused by a submarine's mechanical parts interacts with its construction and the surrounding water to produce acoustic waves that can be picked up by remote sensors. This section focuses on two applications in particular, radar cross section and acoustic signature.

Radar Cross Section

Overview of Radar Cross Section

The radar cross section is the measure of a target's ability to reflect radar signals in the direction of the radar receiver where they can be sensed. The magnitude of a target's radar cross section, and hence its visibility to the radar system, is a function of the target's projected profile, its reflectivity, and its directivity. Intuitively, the projected profile is the size of the shadow of the target when illuminated by the radar signal. The reflectivity is the percentage of the radar's power reflected back in the direction of the radar receiver. The directivity is the ratio of the power reflected back toward the radar receiver to that which would have been reflected back by an object (like a sphere) that reflects uniformly in all directions. Figure 4–9 shows the reflection from various shapes.

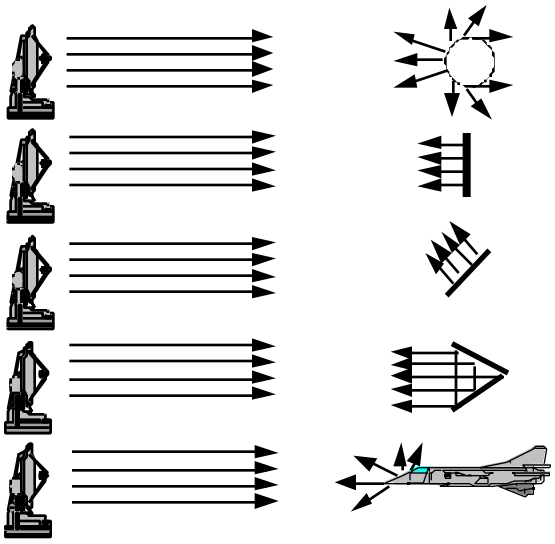

Figure 4–9. Radar reflection from various shapes

Reducing a target's radar cross section has a direct impact on its survivability. A reduced radar cross section not only makes the target more difficult to detect, it reduces the amount of power needed to generate jamming signals to counter the radar. Efforts to reduce a target's radar cross section focus on altering the external shape, utilizing special absorbent materials, or taking measures to alter the nature of the reflected signal. External design changes seek to minimize the projection profile, especially in the directions from which radar signals are most likely to come (e.g., from the front or from below), and to reduce the directivity by using tilted or rounded surfaces that reflect less energy back to the radar source. Using materials or colors that better absorb the electromagnetic waves at the frequencies likely to be employed by radar can reduce reflectivity. Generating jamming signals whose electromagnetic characteristics nullify those of the reflected signal effectively reduces radar reflectivity.

If the radar signal wavelength is much smaller than the radius, a perfect sphere's radar cross section is independent of the wavelength of the radar signal, and is computed as the area of the cross section of the sphere (πr^2). Other shapes may have different radar cross sections depending on the wavelength of the radar signal. For example, a flat plate has a radar cross section equal to $4\pi a^2/\lambda^2$, where a is the area of the plate and λ is the wavelength. Since

wavelength is inversely proportional to frequency, a given plate will have a radar cross section that is one hundred times larger under a 10 GHz radar signal than under a 1 GHz signal.

The reflective nature of simple geometric shapes is well understood, and the corresponding radar cross sections can be solved using analytic means. Targets such as ships and aircraft are geometrically very complex, however. They have a great many reflecting surfaces and corners of varying shapes. Not only are their shapes too complex to solve analytically, but only within the last year has it become possible to compute the radar cross section for an entire aircraft for any frequency approaching that used in operational systems [90]. In practice, the radar cross section is measured after a scale model of the vehicle has been built. Unfortunately, modifications to the target's shape at this relatively late point in the design process are costly to implement. The ability to compute radar cross sections for realistic shapes is very desirable in large part because it would make it possible to evaluate accurately the results of a candidate design early in the design process, when changes are inexpensive and easy to make.

Computational Nature of Radar Cross Sections

Calculating the radar cross section computationally involves solving numerically from first principles the partial differential or integral Maxwell equations. These equations describe the properties over time of changing electric and magnetic fields (light, radio, radar) through media with various refractive and permeability properties.

One leading approach to solving these equations is called a characteristics-based method. As in many computations from the field of computational fluid dynamics, characteristic-based electromagnetics computations define a system with a set of initial values and compute the state of the system over space and time, one time-step at a time. The evolution of the system can be thought of as tracking the movement of information representing changing electric and magnetic fields in time and space. These trajectories are called characteristics [91]. To set up a characteristics-based solution method, the three-dimensional domain being modeled is represented by a set of grid points (for a finite difference method) or cells (for a finite volume method). Each grid point/cell consists of the variables dictated by the governing equations and, depending on the algorithm, additional memory locations for storing the flux, or rate of change of the fields. There are six dependent variables and three spatial independent variables with nine directional variables. After adding in the flux values, nearly 50 memory locations per grid point/cell are required.

The computational demands of such problems can be enormous. To obtain the necessary numerical resolution, cells must be no larger than one-tenth the wavelength of the radar. This requires at least ten elementary finite volume cells (or ten nodes in a finite difference approach) for each wavelength. The greater the frequency of the radar signal, the smaller the wavelength, and the finer the mesh must be. In high-frequency radar beyond the X Band (approximately 10 GHz), the wavelength is reduced to a centimeter or less. Modeling an aircraft with a wingspan of 10 meters requires 10 meters x 100 centimeters/meter x 10 cells/centimeter = 10,000 cells in each dimension, or $10{,}000^3$ = 1 trillion cells. Since each cell must store approximately 50 variables, approximately 50 trillion variables are manipulated. In each time-step, hundreds of trillions of operations are needed to advance the computation. If multiple viewing angles or multiple frequencies are to be computed, the computational requirements are astronomical [92].

A "holy grail" of computational electromagnetics has been the computation of the radar cross section of an entire modern fighter aircraft for a fixed incident angle and a 1 GHz radar signal. While this frequency is less than the 3–18 GHz of normal radar, it is nevertheless an important milestone. Only recently has a problem of this size come within the reach of available computing platforms. At 10 cells points per wavelength, 50 million are needed for a modeling space of 15 meters x 10 meters x 8 meters. On a single-processor Cray Y-MP, advancing the computation a single time-step would require nearly four hours of compute

time. Completing a single simulation requires at least 500 timesteps per run and multiple viewing angles, resulting in months of compute time [93].

Since the mid-1990s, availability of scalable multiprocessors with high-bandwidth, low-latency interconnects has opened up avenues of substantial progress in the field [91]. While the large number of available CPU cycles is certainly useful on MPP systems, the large memory space is the chief advantage [94].

A central task in running the application on a parallel system is determining how to partition the domain to balance the load across processors and minimize inter-node communication. The best way to do this is a matter of ongoing research. One simple, but not necessarily optimal, approach that has been used productively is to partition the three-dimensional domain into a set of planes as shown in Figure 4–10 [92,93]. Each plane is assigned to a processor. In each timestep, grid points/cells exchange values with their neighboring grid points/cells. The amount of data exchanged between nodes is therefore directly proportional to the number of grid points/cells in each node. In addition, each computing node must issue a minimum of two global synchronization operations. In short, although the problem partitions nicely, a considerable amount of communication does take place, and some studies have indicated that the capability of the interconnect is the factor most limiting scalability. Up to the point where the interconnect becomes saturated this kind of problem exhibits over 90 percent scaling, if the problem size is increased as the number of processors grows. The point at which saturation occurs varies from model to model, but is in the range of 128–256 nodes for the Cray T3D, Intel Paragon, and IBM RS/6000 SP [92,95,96].

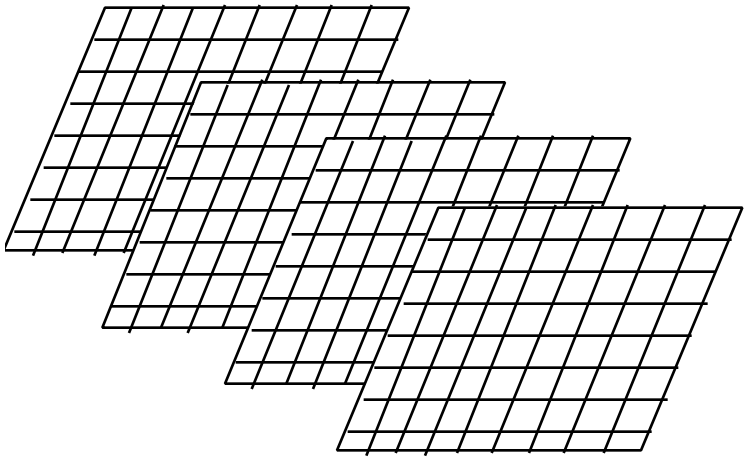

Figure 4–10. One-dimensional parallelization

Are tightly coupled multiprocessors absolutely necessary for solving computational electromagnetics problems? Strictly speaking, the answer is no, for many applications. However, the better the interconnect, the better the obtainable performance. Loosely coupled systems can solve problems. Timely solutions require homogeneous, tightly coupled systems [94]. Moreover, military and civilian computational electromagnetics differ not so much in the applications or numerical methods employed, but in the extreme nature of military usage. While designers of civilian aircraft may be interested in the radar cross section for a specific radar frequency employed by flight controllers, designers of military aircraft are interested in the radar cross section across a range of possible radar frequencies. The additional demands of military applications create a need for the more tightly coupled high performance systems.

The field of computational electromagnetics continues to evolve rapidly. New numerical methods, domain partitions, and computational platforms have made previously unsolvable

problems accessible. For example, researchers at the University of Illinois at Urbana/ Champaign have used an SGI Origin2000 to compute the radar cross section of the VFY218 aircraft under a radar wave of 2 GHz, becoming the first to do so at this frequency [97]. The equations solved contained 1.9 million unknowns. The breakthrough was the development of very efficient algorithms for calculating the scattering of radar waves off large objects.

Machine	*Year*	*CTP*	*Time*	*Problem*	*Problem size*
Cray YMP/1		500	5 CPU hr per timestep	Signature of modern fighter at fixed incident angle at 1 GHz [93]	50 million grid points @ 18 words/grid point
Cray C916	1994	1437		Radar cross section on perfectly conducting sphere [92]	48x48x96 ((221 thousand)
Cray C90/1	1995	1437	12,475 sec	Radar cross section of perfectly conducting sphere, wave number 20 [96]	97x96x192 (1.7 million grid points), 16.1 points per wavelength
Cray C90/1	1994	1437	161 CPU hour	Compute magnitude of scattered wave-pattern on X24C re-entry aerospace vehicle [95]	181x59x162 grid (1.7 million)
Cray C90/1	1995	1437	1 hour	Submarine acoustic signature for single frequency [98]	
Cray C90/1	1995	1437	200 hour	Submarine acoustic signature for full spectrum of frequencies [98]	
Cray T3D/128	1995	10056	12,475 sec	Radar cross section of perfectly conducting sphere, wave number 20 [96]	97x96x192 (1.7 million grid points), 16.1 points per wavelength
Origin2000/128	1997	46,928		Radar cross section of VFY218 aircraft under 2 GHz radar wave	1.9 million unknowns

Table 4–16. Computational electromagnetic applications

Table 4–16 summarizes some of the computational results obtained in this field. It illustrates a number of points. First, PVP systems do not have the memory or compute cycles to solve realistic problems in a time frame acceptable to practitioners. Second, when computing platforms with more processors and larger memory spaces become available, more realistic problems are attempted rather than solving less realistic problems more quickly. Consequently, the availability of MPP systems during the 1990s has resulted in these applications moving from machines of less than 2000 Mtops to machines above 10,000 Mtops.

Submarine Acoustic Signature Analysis

Overview of Acoustic Signature Analysis

The most important quality of submarines is the ability to avoid detection. Consequently, a key submarine design element is the evaluation of acoustic signals that emanate from or are reflected by the submarine. The evaluation of acoustic signatures is an ongoing requirement; as sensors become more sophisticated, submarine design must advance as well in order to maintain the ability to be undetected at a given distance. Acoustic signature analysis involves evaluating the radiated noise from the submarine's internal mechanical parts, as well as how the submarine reflects the acoustic signals of remote sonar systems.

Computational Nature of Signature Analysis

Computational acoustics uses finite element methods. In contrast to electromagnetics problems, acoustic signature problems must take into account substantially more variables of a structural nature such as the pressure inherent in the acoustic signal and the stiffness of the submarine materials. The finite element method, which is well suited to describing complex shapes and structures, is used to model not only the shape of the submarine, but also its internal workings, its materials, and the behavior of the acoustic signal throughout the submarine and the surrounding water. This is a coupled problem and difficult to solve. A design must be evaluated across a broad spectrum of frequencies. As with computational electromagnetics, as the signal frequency increases, the problem size must increase.

Frequency response analysis involves analyzing the reflected signals across a range of frequencies in order to discover those frequencies at which resonance occurs. The most demanding problems solved today in a production environment involve on the order of 100,000 grid points and over 600,000 equations. For a problem of this size, depending on the kinds of materials involved, three to seven hours are needed on a single Cray C90 processor to analyze a single frequency [98,99]. A complete analysis, involving a sweep across two hundred different frequencies, requires up to 1400 hours on the same machine [98]. Each frequency can be analyzed independently of the others. In this respect, the overall analysis is an embarrassingly parallel problem that can be carried out on multiple machines nearly as easily as on a single one. However, the analysis of a single frequency is difficult to parallelize. Current U.S. practice uses a commercial finite element package called NASTRAN. While a great deal of effort has been made to parallelize this package, the parallel versions have not yet been used in production environments. A transition to parallel codes will probably occur within the next few years [99,100]. This application area continues to rely on powerful, tightly integrated platforms with large memory bandwidth and I/O capabilities. Table 4–17 shows some instances of acoustic signature applications and other applications related to submarine design.

Machine	*Year*	*CTP*	*Time*	*Problem*
Cray C90/1	1995	1437	hours	Submarine acoustic signature for single frequency
Cray C90/1	1995	1437	weeks	Submarine acoustic signature for full spectrum of frequencies
Intel Paragon		4600	overnight	Non-acoustic anti-submarine warfare sensor development
		8000		Bottom contour modeling of shallow water in submarine design

Table 4–17. Submarine design applications

Future Developments

Acoustic signature analysis will continue to be able to take advantage of increases in computing power. First, because computational requirements increase as the signal frequency increases, there are limits to the frequencies that can be analyzed using today's PVP systems. Increases in computing power and memory would enable practitioners to increase the range of frequencies they analyze, or to use finer frequency sweeps, or to solve problems with larger grids. Second, greater fidelity is needed when modeling the interaction between the internals of the submarine, the submarine's material composition, and the surrounding fluid to be able to analyze tertiary interaction effects. When parallel codes are introduced into the production environment, acoustic signature analysis will be able to take advantage of the large memory sizes and increased CPU cycles in today's MPP systems.

Summary and Conclusions

(1) Computational electromagnetics is a candidate application area of national security interest. It is still primarily a research tool, with uses in the design of military aircraft.

(2) Computational electromagnetics applications such as the computation of radar cross sections are computationally demanding. Only within the last year or two have computational platforms and algorithms developed to the point where realistic problems become possible, e.g., the 1 GHz radar cross section of a fighter aircraft. The CTP levels of the computers solving such problems are in the tens of thousands of Mtops range.

(3) The current state of the art does not permit the computation of the radar cross section of aircraft under commonly used (3-18 GHz) radar frequencies.

(4) Computational electromagnetics problems scale efficiently on MPP platforms if the problem size increases as additional CPUs are added.

(5) While problems can be solved on loosely coupled systems, the presence of a high-bandwidth, low-latency interconnect gives a substantial time advantage, making parametric studies feasible.

(6) Military applications have more stringent computational requirements and require tightly coupled, homogeneous computing platforms for their solution.

(7) The field of computational electromagnetics is evolving rapidly; breakthroughs in computational methods are reducing the amount of computing power needed to solve a given problem.

(8) Operational acoustic signature analysis for submarine design remains wedded to the PVP systems running the NASTRAN finite element package. While a complete analysis across a range of signal frequencies can be carried out on multiple machines, a single frequency analysis continues to require substantial computing power.

(9) The capabilities of systems currently used for acoustic signature analysis are relatively modest in CTP terms. However, as parallel, production versions of the analysis codes being used become available, there is likely to be a sharp increase in the performance of platforms used and a dramatic increase in the complexity of problems being solved. This transition is likely to take place within the next few years.

Forces Modeling and Simulation

DoD is placing great emphasis on virtual environments for training in and analysis of combat operations. There are three types of simulations in use: live, virtual, and constructive. Live simulations replace sensor information with computer data feeds. The simulated sensor readings are correlated with physical cues to match as closely as possible what the warrior would actually experience. The classic example of live simulation is a flight simulator. Virtual

simulation puts the simulator into a scenario with other entities; i.e., a tank crew will interact with other tanks in a realistic scenario. Some of the other tanks will have live crews; others will be computer-generated entities. Both live and virtual simulations are characterized by the participation of the man-in-the-loop; i.e., the trainee operating his vehicle or aircraft.

The third category, constructive simulation, has no man-in-the-loop feature. Instead, the battles are fought with no or minimal human intervention. Computer-generated forces (CGF) are used to populate the battlefield. Commanders are able to control units, or the computer can control the units. CGFs were initially invented to allow tank crews to have an opponent. The fourteen crews that make up a tank company would each be in their own simulator. CGFs would represent the enemy tanks in the scenario. This removed the need to construct and man simulators to represent enemy forces [101].

History

Simulator Network (Simnet) was the first system to support all three types of simulation. It was developed in the mid-1980s by DARPA. The information needed for the simulation is sent over local or wide-area networks, allowing widely dispersed simulators to participate. Simnet, however, suffered from several limitations:

(1) battles took place only in clear weather at noon,

(2) commands between various command levels were not handled well, and

(3) simulation time always proceeded in continuous time—that is, simulations could not "skip ahead."

The work to extend Simnet has led to the Distributed Interactive System (DIS) protocol, and associated IEEE standard (1278) which has specifications for more vehicle and weapon types and non-visual effects [101]. The DIS standard, however, is still limited in that it does not allow all simulations to interact. As a result, a new distributed architecture, High-Level Architecture (HLA), was adopted in October 1996. HLA is designed to support heterogeneous hardware and software, multiple independently designed sub-architectures, simulations distributed over local- and wide-area networks, and 3-D synthetic terrain environments and databases.

More critically, HLA addresses interoperability among the different levels of command that may be represented in a simulation. These levels can range from the individual weapon system through platoon, company, battalions, etc. HLA treats these as aggregates. For example, a company is represented as three platoon entities and one commander entity [102].

In 1994, the first large, real-time simulated exercise was held: Synthetic Theater of War - Europe (STOW-E). The exercise used Simnet and DIS protocols, CGFs, and man-in-the-loop simulators from all the services. Wide-area networks and a transatlantic link connected 14 U.S. and European sites. Over two thousand entities, most computer-generated, were involved [101]. The 1995 Engineering Demonstration (ED-1A) exercise increased the number of entities to 5371 vehicles hosted at 12 sites in the continental United States [103].

Current State

Typical training installations will have a set of simulators for the type of unit(s) training there. For example, an Army installation might have a collection of M-1 or Bradley simulators. The Marine Corps is developing simulators for individual soldiers to train as small teams. These simulators can be used singly, with all other entities portrayed by CGFs. Or, the simulators might be combined for small unit training with only the "enemy" represented by CGFs.

CGF entities have four main components: action, perception, fatigue (physical fatigue for a soldier, or fuel and engine state of a vehicle), and damage. A single simulator, such as a tank, typically uses a PC or workstation-class machine. By comparison, a single workstation is

capable of generating several dozen CGFs or more for a simulation. The more detailed and "realistic" the CGFs, the fewer that can be hosted on a single machine due to the computational requirements.

As an example, the Chemical Biological Defense Command (CBDCOM) has developed a set of simulators for the Fox CBD vehicle and its associated equipment. The simulators include the Fox vehicle itself, the MM1 detector, the M21 standoff detector, and the LSCAD, a lightweight version of the M21. CBDCOM also designed threat scenarios, such as a missile attack that releases a gas cloud along the front of an advancing armored formation.

These simulation tools have been used to evaluate improvements to existing equipment and newly designed equipment. Designers now have the luxury of trying a number of ideas without the expense of building multiple prototypes. The simulations also provide the opportunity to develop tactics for the use of new equipment while it is still in development. Thus, when the equipment becomes available in quantity, the tactics for its use are already in place.

The computing equipment used at CBDCOM to participate in a simulated exercise includes PCs (under 500 Mtops) for the MM1 and M21 detectors. Three SGI Onyxes are employed for:

(1) the Fox vehicle simulator

(2) the chemical, biological, radiological simulator

(3) the after-action report (an analysis tool)

The largest Onyx is a four-processor system (625 Mtops). The Onyxes are used primarily for their interactive graphics capability, not number crunching [104].

The mix of machines at CBDCOM is typical of most installations. The system is expressly designed to be a cost-effective system requiring the use of commodity, off-the-shelf computing equipment.

Large-Scale Forces Modeling

Modular Semi-Automated Forces (ModSAF) projects support the simulation of large-scale military exercises. Unlike simulations described above, these do not normally involve man-in-the-loop simulators. Instead, the purpose is to allow commanders to examine large-scale tactics and to analyze the results of various tactical and even strategic decisions.

Nominally, it is possible to add entities to the standard DIS simulation by adding more workstations, each modeling some number of CGFs. However, the DIS standard is not sufficiently scalable. First, broadcast messages can overwhelm the networks as the number of entities increases. Second, the number of incoming messages can exceed the processing capabilities of the individual workstations. Much work in the ModSAF community is focused on solving these problems.

The Synthetic Forces Express (SF Express) project is exploring the utility of using parallel processors, both large MPP and smaller clustered systems, in conjunction with intelligent data management as a solution to the communications bottlenecks. The project is a collaborative effort among Caltech's Center for Advanced Computing Research, the Jet Propulsion Laboratory, and the Space and Naval Warfare Systems Center San Diego.

The general approach is to use "interest management" [105]. Essentially, entities in the simulation state the events they are interested in. Messages go only to those entities interested in that message. This allows, for example, a vehicle to be interested in other vehicles within a certain range, or that are within some specific physical area, such as a canyon, town, etc. Once the interests have been stated, the vehicle is no longer informed of vehicles/entities that do not match those interests. A vehicle's region of interest is dynamic and changes as that vehicle moves on the terrain database during the simulation.

The communications architecture uses a primary router node to handle communications for a cluster of standard simulation nodes, as shown in Figure 4–11. The simulation nodes are typically a subset of the processing nodes available on an MPP, or nodes in a PC cluster. Each simulation node talks to the router, rather than broadcasting its information, as is the case in the standard DIS simulation. A simulation node will have one to several dozen CGFs on it. Each CGF will declare its interests at the start of and during the simulation. The node will forward the union of the interests of its CGFs to the primary router. The primary router will collect all messages from its cluster of simulation nodes and forward to the simulation nodes only those messages that match each node's interests. Experiment has shown that one primary router can handle the communications and interest declarations for a cluster of simulation nodes that models a total of one to two thousand vehicles.

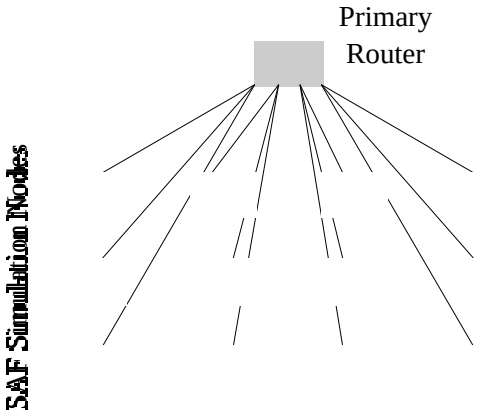

Figure 4–11. Communications between nodes and primary router

When the simulation scales beyond the capabilities of a single primary router node, additional nodes can be allocated to create new clusters, each with its own primary router node. At this point, two additional communication nodes are needed for each cluster as shown in Figure 4–12. The pop-up router (labeled U) handles the outgoing messages from the primary router. The pull-down router (labeled D) collects the interest declarations from the primary router and uses these to determine which messages in the pull-down layer are to be delivered to the primary router. This communications architecture is scalable and handles a variety of machines. Simulation nodes and routers can be a single processor workstation or a node in an MPP.

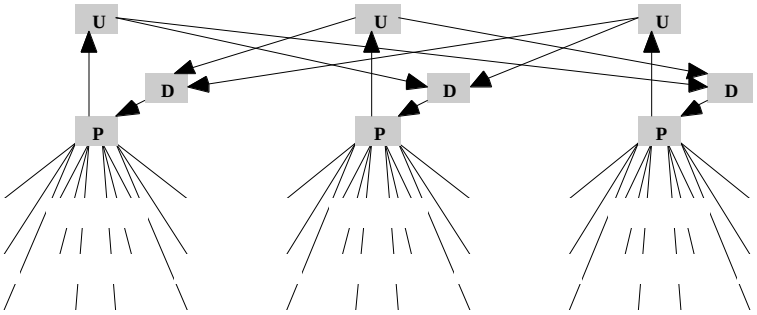

Figure 4–12. Communication patterns between clusters

A variety of tests have been conducted. They include tests on a single MPP host, on a cluster of workstations, and on multiple, geographically distributed MPPs. These tests are summarized in Table 4–18 [103]. A 50,000-vehicle simulation was also demonstrated at SC97 in November 1997. A 100,000-vehicle simulation was completed on March 16, 1998, involving thirteen heterogeneous MPPs at nine different sites [106]. Work is now being directed toward enriching the simulation environment. The current terrain database used for the simulations is a desert terrain with relatively few features. In addition to using more feature-rich terrain databases, work is proceeding on incorporating improved weather effects and such weather-related effects as gas clouds, smoke, etc. It is estimated that a simulation node that can support *x* vehicles in the current desert terrain will be able to support one-half *x* in a more complex terrain [107].

Test	*machine*	*nodes*	*CTP*	*vehicles*	*entities*
single MPP	ORNL Paragon	360	9626	5,713	6,697
		680	16737	10,913	13,222
		1024	24520	16,606	20,290
6 MPPs, June 97, 2 hr simulation	Caltech HP Exemplar	256	60345	13,905	
	ORNL Paragon	1024	24520	16,995	
	NASA Ames SP-2	139	14057	5,464	
	CEWES SP-2	229	21506	9,739	
	Maui SP-2	128	13147	5,086	
	Convex HP Exemplar	128	34425	5,348	
Cluster	two Beowulf systems	80 2 x 40		3,000	

Table 4–18. Synthetic forces experiments

Conclusions

The preceding discussion leads to the following conclusions:

- The simulated training environments do not use controllable computing technology. These systems are built from PCs and small workstations.
- Forces modeling, when applied to large-scale forces, does use controllable HPC technology, primarily to achieve high realism coupled with real-time operation. Moderate-size (one or two divisions) simulations require MPP systems at about 10,000 Mtops. Larger, theater simulations require networks of MPPs.
- Large, theater simulations are still in the research and development stage. Within the next one to two years, these simulations will become available for training battlefield commanders and for use in devising and evaluating realistic operational plans.
- Individual, networked workstations are not yet usable for modeling large-scale forces. Clusters of workstations have been used in some large-scale simulations. Clusters will become more viable as work on providing a scalable communications layer continues to reduce the volume of communications.

Littoral Warfare

Littoral warfare refers to military operations in coastal areas. This type of warfare has a particular impact on naval operations, and imposes some difficulties on air operations as well. Particularly with the end of the Cold War, the U.S. Navy has become more concerned with operations in littoral seas.

Operations in littoral areas impose requirements that demand high-performance computing. Some of these applications have already been covered in this report. They include submarine maneuvers in shallow waters (see Submarine Design, page 83), and regional weather forecasts (see Modeling Coastal Waters and Littoral Seas, page 64). Two additional areas are examined briefly in this section: electronic warfare and surveillance.

Electronic Warfare [108]

Traditional electronic warfare (EW) for the Navy has been in the open ocean, an environment that is relatively easy to manage due to less interference with propagation than over land. In the current world situation, however, the Navy needs to operate closer to land, and has taken on the task of providing EW support for inland forces.

The electronic database of the past has been one that contains an estimated order of battle for the opposing forces. The database was relatively static and based on archival information. The estimates of radar and EW effectiveness were limited to simplistic coverage zones as a function of altitude and range. Tactical maps were produced from this data by simply penciling in the location of an emitter and its coverage.

Simulations that involved, for example, tracking a missile in its flight over the ocean were relatively easy, since the environment was sparse. Tracking that same missile over land is considerably harder. HPC has been particularly useful during the research phase of such projects. The assumption is not that a ship will have cutting-edge computing. Instead, the researchers use HPC to develop the algorithms and determine which environmental factors are significant and which are not. By the time the code is developed and the environmental factors have been reduced to only those that are significant, the code is expected to perform well on the computing platforms that will then be available in the fleet.

Parallel versions of the current codes have been developed, primarily for shared memory parallel systems such as the SGI PowerChallenge and Onyx systems. These are coarse grain parallel codes, deriving the parallelism over the target's bearing or aspect angle. A vendor's parallel tool will often not be suitable, as it attempts to parallelize the inner as well as the outer loops. This forces the programmer to "fix" what the auto-parallelizer has done.

The Navy is working on a new, integrated EW system. It will use current intelligence estimates of the opposing force's electronics capabilities and integrate terrain information and weather into estimates of electronic coverage. EW assessments can be displayed on screens that show locations of friendly forces and geographic information. Bringing this data together is primarily a network and communication bandwidth problem, not a computing problem. The integration of the data into a useful tool for the on-scene commander, however, is a computing problem. The Naval Research Laboratory is developing the tools needed to integrate the various EW information streams into an integrated product. They are developing adaptive techniques that will select from a range of fidelities. The choices will be based on the information available—intelligence estimates, environmental data (weather, sea state, etc.), terrain data, force dispositions, and the time available to compute a solution.

Surveillance [109]

The Advanced Virtual Intelligence Surveillance Reconnaissance (ADVISR) program is developing an integrated system of underwater, surface, airborne, and space-borne surveillance sensors. The project is using HPC in three ways. First, simulations model the performance of the sensors. The current program, for example, is designing underwater

sensors that will rest on the continental shelf off a coastline, with potentially hundreds of sensors involved in an actual deployment. The sensors will communicate with a master sensor that can then be queried by friendly forces for information. During development, an individual sensor on a workbench is wired into a simulation. The simulation provides all the inputs that the sensor would "hear" if it were in the water off a targeted coast. The sensor then performs its analysis functions. When it needs to communicate with the master sensor or other sensors in the array, it does so; the simulation handles the other side of such conversations. This allows the researchers to test the sensor in the lab without having to actually deploy the sensor in the ocean each time. Further, simulation can model a large number of the sensors. This level of simulation requires that each sensor be represented in the simulation. A few of the sensors would be present on the workbench. The remaining sensors, and there can be up to several hundred, are simulated. Currently, this level of simulation is being performed on the 336-node Intel Paragon (9093 Mtops) located at the Space and Naval Warfare Systems Command (SSC) in San Diego.

The second use of HPC in this project occurs in the visualization area. Researchers are developing a virtual world that will allow tactical commanders to view their environment. The virtual world will incorporate both land and underwater topography, locations of friendly, hostile, and neutral forces on land and sea, locations of deployed sensors, and various environmental factors including weather patterns. The virtual world is being used to test the sensors themselves. Objects such as submarines, schools of fish, etc., can be placed in the virtual world. The simulation determines how well the sensors can detect the objects. The virtual world also supplies outputs that allow experimentation with techniques for displaying this information to tactical commanders. The virtual world currently requires two multi-processor SGI Onyx systems.

The third use of HPC in this project integrates the results of all the sensors to produce the inputs to the virtual world. The goal is to reduce the volume of information coming from sensors to a usable level that commanders can assimilate without leaving out critical information. This requires not only the collection of the information, but a decision-making process that prioritizes the information before sending it to the display system. This integration of information is being developed and tested on the Intel Paragon at SSC. Thus, it currently requires a level of computing beyond that normally found in fleet flagships. It is anticipated that the level of computing in the fleet will improve with developments in the computer industry before the system is ready for deployment. The final level of computing needed will likely be less due to (1) algorithmic improvements in the software and (2) decisions as to which information streams ultimately will prove useful and which can be left out of the final product.

Summary

Littoral warfare research and development currently has high computational demands. While these demands will be somewhat less by the time the systems are deployed, an implicit assumption is that better shipboard and airborne computing platforms will be available. HPC is allowing the research and simulation of the future systems to take place today, so that the software will be in place when the new hardware arrives.

Nuclear Applications

Introduction

The development of nuclear weapons involves the study of physical phenomena in the extreme. The phenomena are highly non-linear, often in non-equilibrium, and frequently on scales of time and space that are very difficult to measure and interpret empirically.

Furthermore, the development process is multidisciplinary, involving studies of kinetics, fluid dynamics, structures, complex chemical reactions, radiation transfer, neutron transfer, etc. [110]. Since the mid-1960s, computer programs have been developed to try to model these complex processes. Nuclear weapon development and maintenance uses dozens of different codes that represent various pieces of this complex phenomenon.

Evolution

The earliest nuclear weapons were developed using hand-operated calculators. After the dawn of the computer age, however, the national laboratories were often the first users of new systems from companies such as Control Data Corporation, Cray Research, and Thinking Machines. As early adopters they were willing to accommodate a good deal of pain in dealing with immature hardware and software platforms for the sake of obtaining the highest levels of computing power available. The development of nuclear weapons and advances in supercomputers were closely linked throughout the 1970s and 1980s.

The evolution of the computations and the computers to support them fall, very roughly, into three phases, corresponding to one-dimensional, two-dimensional, and three-dimensional computation.

The early weapons employed single, simple primaries (detonation devices) whose behavior could be satisfactorily modeled using one-dimensional computations. Such computations required on the order of only a single megabyte of main memory and were performed on machines such as the CDC 7600 and Cray-1S (195 Mtops). During the 1970s, these machines were used to carry out two-dimensional computations, albeit at under-resolved resolutions and less than full physics.

During the 1980s, with the introduction of the Cray XMP and, later, the Cray YMP, two-dimensional modeling with well resolved, full physics became common. Two-dimensional computations were done on single processors of the Cray XMP (approximately 350 Mtops) and Cray YMP (500 Mtops). Production runs on more than one processor were uncommon, because multiprocessor jobs used CPU cycles less efficiently than did multiple jobs running on individual processors. Over time, two-dimensional codes were improved. Resolutions were increased, approximations needed to simplify code and reduce computational requirements were improved, and codes were enhanced to include a more complete set of physical and chemical models. Many production codes continue to be 2–D codes. The 1980s also saw the first three-dimensional, under resolved, reduced physics computations.

During the mid-1990s, the objective of computational research shifted from weapons development to stockpile maintenance. The question asked was no longer, "How can we build a new device with a higher yield that will fit into a smaller container and survive more hostile environmental conditions?" Rather, research was oriented toward understanding, for example, the affects of aging on the ability of an existing device to function as designed. Many of these non-normal phenomena are, unlike detonation phenomena, inherently three-dimensional. They include [111,112]:

(1) the effects of microscopic fractures;

(2) chemical changes such as polymer breakdown, metal corrosion, and debonding;

(3) physical processes related to fissure analysis and stability;

(4) the operation of physical devices with parts moving in three dimensions.

The shift from 2-D modeling to 3-D modeling requires an enormous jump in computational capability. Not only does a third dimension compound the number of grid points, it also requires more variables per grid point and, as the resolution of grids increases, smaller time-steps. While the 3-D under resolved, reduced physics modeling codes can execute on a Cray

YMP, this system does not have the memory necessary to execute a model that is sufficiently precise and accurate to be useful. In the United States, the Intel Paragon was the first machine with a large enough memory (32 Gbytes) to run credible 3-D resolution reduced physics simulations. This system, which represents a hundred-fold increase in CTP value over the single processor Cray YMP, is considered a minimum for credible 3D simulation. Systems with ten times more power (1000x over a Cray YMP, or 500,000 Mtops) are considered suitable for a broad range of 3D simulations in this application domain. Full physics 3D modeling requires one or two orders of magnitude more computing power.

Gaining Confidence

In each of these cases, the principal objective was not only to build a device, but also to have a certain assurance, or confidence, that the device would function as intended. Unlike mechanical devices like airplanes in which a single unit can be tested, modified, and certified before being used by a customer, nuclear devices either work or they do not. There is no opportunity for in-the-field modification to correct an error. Gaining confidence in a device can be done in two ways. The traditional way is to build a device and test (explode) it; if the test is successful, then build several more devices just like it and assume that these copies will behave as the original. As computational methods and capabilities have advanced, simulations of physical processes are increasingly able to model the behavior of a particular device under particular circumstances.

The confidence one can place in the results of the simulation is directly related to (a) the extent to which a code has been validated, and (b) the extent to which the phenomenon being simulated is similar to the phenomena against which the code was validated. Validation consists of running a simulation of a test that has been carried out and verifying that the computed results are acceptably close to the empirical results. If a validated code is used to model small variations on a proven design (e.g., replacing one small component with another one substantially similar to it), then practitioners have a great deal of confidence in the computed results. The more the simulated situation differs from established practice, the more computing power is needed for the simulation, and the less confidence one has in the computed results. In general, one will have little confidence in computed results alone for a simulation of a new weapons design concept, or a phenomenon that has not occurred during actual testing. A researcher with a high-performance computing system but inadequate test data has been compared with a drunk driving a car: the more the drunk has to drink the better he thinks he drives. Comparably, with only a high-performance computing system a researcher may obtain impressive results, but have no way of knowing whether those results match reality [113].

Greater amounts of simulation with higher spatial resolution and physics fidelity may not give full confidence, but they may help developers determine which tests will be most beneficial in the development process. The net result may be fewer (but not zero) tests, at lower costs, over a shorter time frame.

Confidence in computational results is gained in a number of ways as testing and experience grow [111]:

- The underlying physics models. Today, there is great confidence that these are correct in many areas of nuclear investigation. These models accurately reflect the nature of the world and are quite stable. Moreover, they are available to and understood by scientists throughout the world. Needless to say, there are many fields (e.g., three-dimensional compressible flow) in which the understanding and ability to model physical phenomena precisely are lacking.
- The numerical methods. The nature of the problem dictates the resolution, which dictates the computational requirements. If the machine being used is not able to offer the

necessary memory and CPU cycles, then one will have little confidence in the results, even if the numerical methods and algorithms are sound.

- The fidelity of the computation. Does the code actually model the real world? Code must be validated against empirical tests. Many of the phenomena modeled in weapons development or maintenance are not explicitly nuclear. To validate these parts of the code, non-nuclear test data can be used. In many (but not all) cases, such data is available internationally.
- Finally, there are parts of the code that model the highly non-linear effects of the nuclear explosion itself. These codes can be validated only with nuclear test data.

Of these four elements, only the numerical methods are directly affected by export control of HPC hardware. To have full confidence, however, all four elements must be available. The implication is that advanced hardware *by itself* contributes little to the development or maintenance of nuclear devices; in conjunction with nuclear and non-nuclear test data, however, computing may be quite beneficial.

When test data are available, incremental increases in computing power provide incremental benefit. Practitioners are able to run bigger problems, improve the resolution of codes, refine approximations, use improved physics models, reduce the turnaround time, or do a combination of all of these [81,111,114]. This relationship is shown in Figure 4–13. The figure is not drawn to scale.

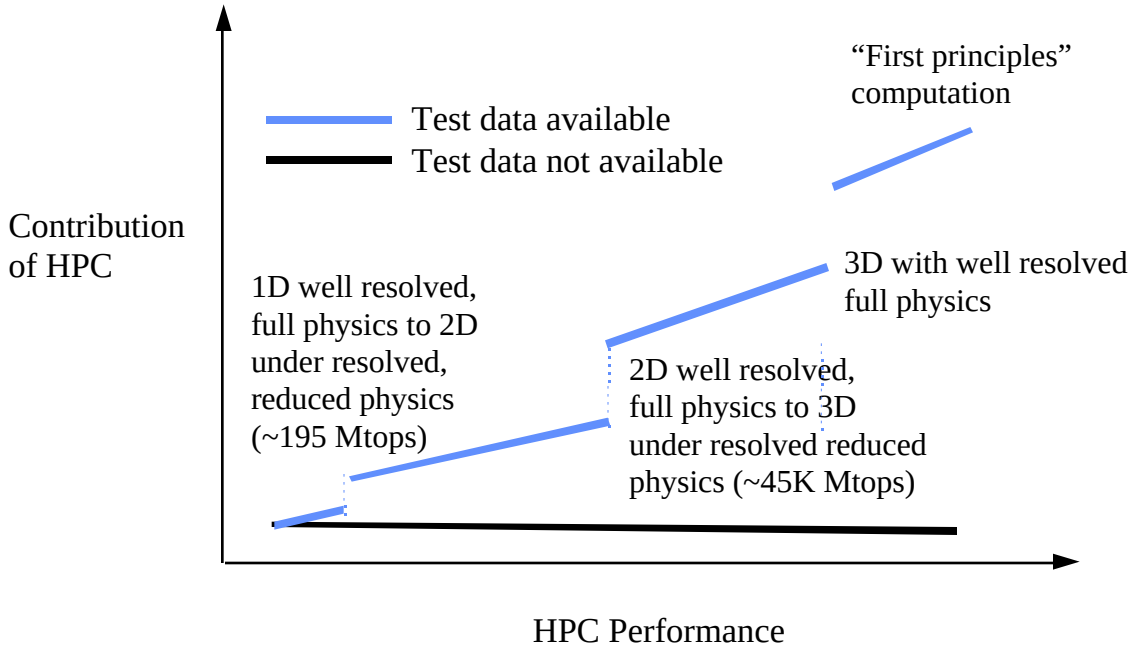

Figure 4–13. Contribution of HPC to nuclear weapons development with and without test data

Computing Platforms and Nuclear Applications

Parallel computation is a maturing discipline within the nuclear weapons community. During the late 1980s and early 1990s, the Department of Energy national laboratories decided to step off the upgrade path offered by Cray Research's family of PVP machines. Since the early years of this decade, the labs have invested heavily in understanding and developing parallel

code. Today, many codes are in their second or third generation on parallel platforms. Today, MPP platforms are the principal production systems; few PVP systems remain in use at the national labs.

Porting codes from PVP to MPP platforms is a non-trivial exercise. Sandia reports that porting the main production codes required one to two person-years of effort. This compares with fifteen to twenty person-years to develop the original PVP code. In addition, a good deal of additional work typically needs to be done to gain optimal performance on a particular platform.

Most of the production codes run and scale well on parallel platforms. Many codes, Monte Carlo calculations in particular, employ coarse-grain parallelism. These codes perform a large amount of computation relative to inter-node communication and are therefore relatively latency tolerant. Other codes, or portions of codes, are not as latency tolerant. These include several important physics packages and perhaps the deterministic particle transport codes, such as that represented by the SWEEP3D benchmark [115,116]. Wasserman et al. have compared the performance of two SGI systems employing the same microprocessor: the PowerChallenge and the Origin2000 [115]. They show that the improved latency-hiding memory systems of the Origin2000 have a substantial impact on the performance of benchmarks characterizing Los Alamos National Laboratory's nuclear applications workload. Detailed analysis of which codes are likely to benefit most from high-performance interconnects is a subject of ongoing research.

Trends for the Future

There is no shortage of exceedingly demanding applications within the nuclear weapons community. Although it has been online for only a year, the ASCI Red system at Sandia National Laboratory has already been run at capacity by one application: CTH. Researchers have outlined a dozen applications, ranging from the modeling of multi-burst ground shock (effects against deeply buried structures) to the simulation of neutron generator standoff, that require a Teraflops or more of sustained performance, and 100 Gbytes to multiple Terabytes of main memory. These requirements significantly exceed the capabilities of the 250,000 Mtops configuration available to a single application today.[20]

The major shift anticipated by the ASCI program is use of 3-D simulations based on first-principles physics modeling [112]. Such simulations do not rely on empirical data to establish the initial and boundary conditions and macro-level approximations during a computational run. Instead, they model the activity of nuclear particles directly, subject to the fundamental laws of physics. Table 4–19 shows ASCI projections of the computing requirements for 3-D simulations based on first principles.

Simulation	*Teraflops required*	*Terabytes required*
Radiation transport	123 Tflops	40 Tbytes
shock waves propagating through explosive components	150 Tflops	75 Tbytes
chemical changes (polymer breakdown, metal corrosion, debonding) for highly localized phenomena	30 Tflops	5.5 Tbytes
Detailed engineering simulations of microscopic degradation	100 Tflops	100 Tbytes

Table 4–19. Computing requirements for first-principles simulations. Source: [112]

[20] The full configuration ASCI Red has over 4500 compute nodes (CTP approx. 300,000 Mtops). Only about 3500 are available for a single application, however. The machine has an unclassified portion and a classified portion. A given application will use one or the other, but never both at the same time.

Summary and Conclusions

Table 4–20 illustrates the computational power used in the United States to run various nuclear weapons development and maintenance applications.

Machine	*Year*	*CTP*	*Time*	*Problem*
Cray-1S	early 1980s	195	117.2 CPU sec	LANL Hydrodynamics code 1
Cray-1S	early 1980s	195	4547.1 CPU sec	LANL Hydrodynamics code 2
Cray-1S	early 1980s	195	1127.5 CPU sec	LANL Hydrodynamics code 3
Cray XMP/1	mid 1980s	316	seconds	2-D, reduced physics simulation
Cray YMP/1	late 1980s	500	1000s of hours	1-D, full physics simulation
Cray YMP/1	late 1980s	500	overnight	2-D, almost full physics (e.g., Monte Carlo neutron transport)
Cray YMP/1		500	overnight	2-D, reduced physics simulations
	1995	1400		Credible 1-D and 2-D simulations
Intel Paragon	1994	46,000	overnight	3-D reduced physics simulation of transient dynamics of nuclear weapon
ASCI ++	??	50,000,000+	days	First principles 3-D modeling

Table 4–20. Nuclear weapons related applications. Sources: [117,118]

Table 4–20 and the preceding discussion lead to the following principal conclusions:

- The availability of test data remains critical to the development of modern nuclear weapons. Without test data, even very large and powerful computers will not be useful.
- When test data are available, any increase in computing capability provides increased benefit to the application. There are few break points at which an increase in computing power provides a disproportionate benefit.
- The nuclear weapons in the current U.S. stockpile were developed using computers of 500 Mtops (CTP of a single-processor Cray YMP) or fewer, albeit in an environment with a rich collection of test data.
- Workstations with modest numbers of processors (500–1400 Mtops) can execute fairly robust 1-D and 2-D calculations.
- The transition from two-dimensional to three-dimensional modeling is one point at which improvements in computing provide disproportionate benefit. The critical computational resource needed for this application is memory (approximately 30 Gbytes of RAM). Machines that have run 3-D reduced physics simulations at satisfactory resolution and precision *in an acceptable time frame* have CTP levels of 40–50,000 Mtops.
- No computers currently in operation are powerful enough to model all the key nuclear processes from first principles. The implication is that test data continues to be a critical element in nuclear weapon development and stockpile management. Without it, one cannot validate codes.

- Some production codes have been ported to MPP platforms and run as parallel codes. Other production codes run as serial codes on single processors of multiprocessor machines. PVP systems are being phased out. Codes run on a broad spectrum of architectures.

Conclusion

The choice of a viable control threshold is bounded from below by a determination of which performance levels are not controllable, and from above by a determination of which applications with performance requirements above this lower bound should be protected. As discussed in this chapter's introduction, there are two principal positions regarding national security applications that can be taken, each with its own implications for the choice of an upper bound:

(1) The set of applications of great national security importance is broad and deep, distributed across the entire spectrum of computing performance. All applications of any national security interest should be protected to the extent possible. Under this view, as long as there is any national security application above the lower bound of controllability, the export control policy can be justified.

(2) The export control policy should try to protect a subset of the applications of national security interest. It may be limited to those of *critical* national security importance, or to those that can be effectively carried out by foreign entities in ways that threaten U.S. national security interests. The choice of upper bound may also try to protect performance levels with the greatest density of applications.

This chapter provides numerous examples of applications of national security interest being pursued at performance levels above the threshold of controllability. If policy makers hold the first position, this chapter provides abundant support for a continuation of the policy. If policy makers hold the second position, this chapter provides a superset of applications that may be used to set a viable threshold's upper bound. In the sections below, we describe why national security applications with advanced computational requirements will continue to exist for the foreseeable future, how the applications in this study are distributed across performance levels, and which applications may be found within various performance bands. What remains is for policy makers to determine which applications are of such compelling national security interest that they should determine the upper bound for the control threshold. We also briefly discuss foreign entities' pursuit of national security applications and the use of mirror imaging as an analysis technique.

Application Characteristics

There is a wide range of applications of potential national security interest requiring high-levels of computing performance. The insatiable demand for greater resolution, increased scale, and improved fidelity drives computational requirements upward. Furthermore, the applications can be characterized by the computing requirements that make them doable on high-performance systems, but not doable in a usable form on lesser computing platforms:

- memory requirements—both large memory capacity and high memory-CPU bandwidth
- large computational requirements relative to time constraints—either to meet operational deadlines or to reduce research and development time

- coupled problems—e.g., combining CFD and structures for airframes, combining explosions and building structural response
- space/weight/time/power constraints—e.g., air- and space-borne signal processing
- deforming grid problems—e.g., parachutist falling through wake of aircraft, armor/anti-armor
- large matrix solutions needed—e.g., CFD problems, particularly those solving turbulence problems

Application Bands

Figure 4–14 shows a histogram of the specific applications studied for this report. Among applications that require high computing levels, the bulk of them fall in the 15,000 to 25,000 Mtops range, primarily because this represents the performance of systems that have been available to researchers over the past few years. As newer systems with increased performance come online over the next one to two years, there will be an increasing number of applications that will appear in the 30,000 and up ranges. These will fall into two categories:

- applications similar to current applications but with greater resolution and/or larger problem sizes, and
- applications not possible on current systems.

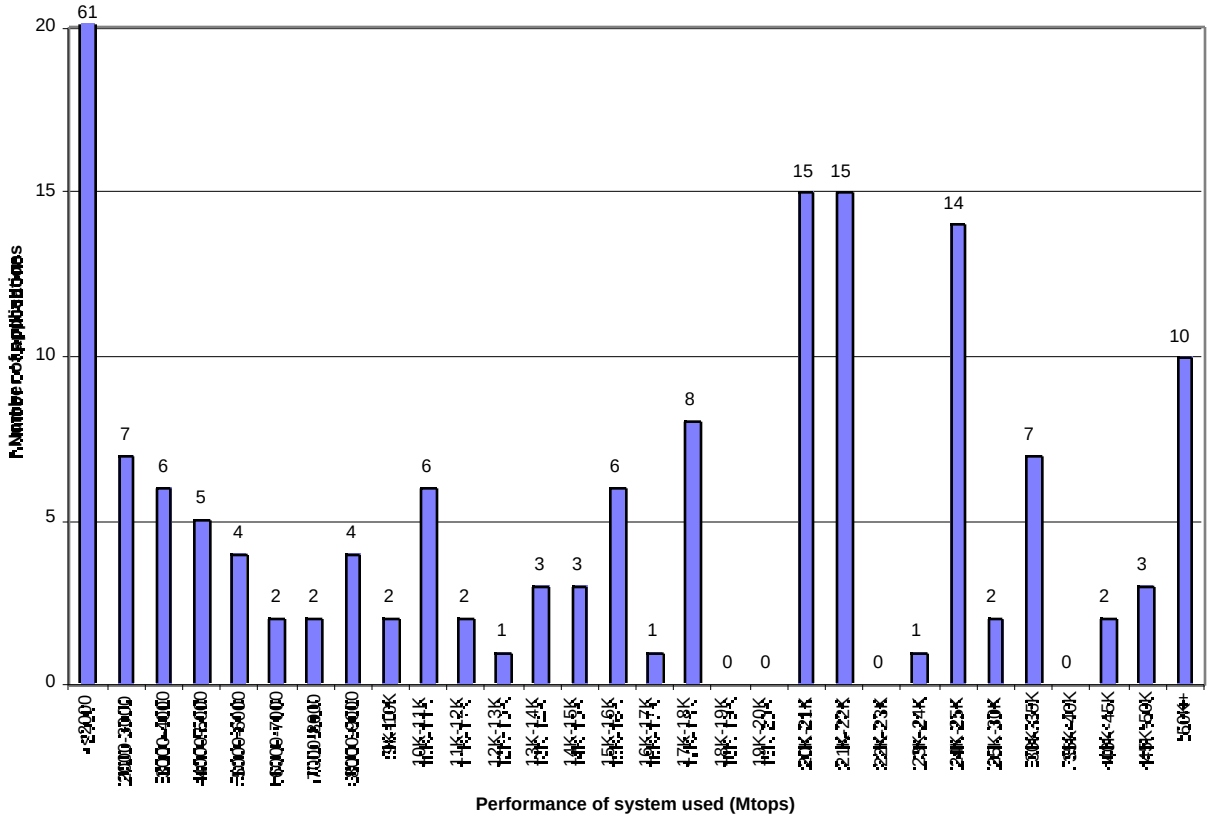

Figure 4–14. Application histogram

The next three tables take a closer look at performance levels of probable interest to policy makers. Table 4–21 shows applications in the 5000 to 10,000 Mtops range. These represent CFD applications that examine specific parts of an aircraft or simple geometric shapes, moderate-sized computational chemistry problems, brigade-level battle simulations, and coarse-grain weather forecast models. The weather forecast models at this level are not accurate very far into the future. A significant implication is that such forecasts would not be suitable inputs to regional forecasts over an area of concern to military planners.

CTP	*CTA*	*Problem*
5375	CFD	Modeling of flow around a smooth ellipsoid submarine. Turbulent flow, fixed angle of attack.
6300	CFD	Helicopter blade structural optimization code, gradient-based optimization technique to measure performance changes as each design variable is varied.
6332	CWO	CCM2, Community Climate Model, T42.
7100	CFD	Helicopter rotor free-wake model, high-order vortex element and wake relaxation.
7315	CCM	Particle simulation interacting through the standard pair-wise 6-12 Lennard-Jones potential, 50M.
8000	CEA	Bottom-contour modeling for submarine design.
9510	CFD	Large-Eddy simulation at high Reynolds number.
9626	FMS	5.7K vehicle battle simulation.

Table 4–21. Applications between 5000 and 10,000 Mtops

The next set of applications is shown in Table 4–22. These include simple radar cross section problems, computational chemistry problems of increasing size, and operationally useful weather forecast models. The weather forecast models are clustered just above 10,000 Mtops. These represent operational forecast models used by FNMOC (and others) starting about 1995. While FNMOC has since upgraded to more capable computers, this band represents a region of usable computing performance for weather forecasts. Global forecasts at this level of accuracy begin to be useful as inputs to regional forecasts. Computing power beyond this level allows global forecasts that are accurate further into the future.

CTP	*CTA*	*Problem*
10056	CEA	Radar cross section of perfectly conducting sphere, wave number 20
10457	CCM	Determination of structure of Eglin-C molecular system
10625	CWO	Global atmospheric forecast, FNMOC operational run (circa 1995)
11768	CWO	SC-MICOM, global ocean forecast, two-tier communication pattern
13004	CCM	Particle simulation interacting through the standard pair-wise 6-12 Lennard-Jones potential, 100M
13236	CCM	Molecular dynamics of SiO_2 system, 4.2M
14200	CWO	PCCM2, Parallel CCM2, T42

Table 4–22. Applications between 10,000 and 15,000 Mtops

Applications in the next range fall into two bands, as shown in Table 4–23. The first band is near 15,000 Mtops and the second is around 17,000 Mtops. Starting at this level, available computers do not uniformly fill the CTP range; thus, applications begin to "clump" around the CTP of specific machines. This range includes more accurate weather forecast models, simple explosion-building simulations, divisional-size battlefield simulations, and CFD problems that begin to model complete airframes of small objects such as missiles with some turbulent effects.

CTP	*CTA*	*Problem*
15796	CWO	AGCM, Atmospheric General Circulation Model
15875	CSM	Water over C4 explosive in container above wet sand, building at a distance
15875	CSM	Water over C4 explosive in container above wet sand, alternate scenario - container next to building, finite element
15875	CWO	Global atmospheric forecast, FNMOC operational run (circa 1997)
16737	FMS	11K vehicle battle simulation
17503	CCM	MD simulation using short-range forces model applied to 3D configuration of liquid near solid state point, 100K & 1M & 5M particles
17503	CFD	Aerodynamics of missile at Mach 2.5, 14-degrees angle of attack for laminar and turbulent viscous effects
17503	CFD	Large-Eddy simulation at high Reynolds number
17503	CWO	ARPS, Advanced Regional Prediction System, v 4.0, fine scale forecast
17503	CWO	Global weather forecasting model, National Meteorological Center, T170
17503	CWO	AGCM, Atmospheric General Circulation Model

Table 4–23. Applications between 15,000 and 20,000 Mtops

In general, applications offer the greatest strategic advantage when they open avenues of discovery or operation that are not possible or are severely retarded in the absence of the necessary computing power. The following features characterize the national security nature of the applications described in this report:

- modeling of events on time/space scales that cannot be observed, e.g., explosions, molecular modeling
- time savings that produce strategic advantage, e.g., weather forecasting, chemical and biologic warfare modeling
- enabling operational actions not otherwise possible, e.g., movement of submarines in shallow waters
- lending operational advantage, e.g., regional weather forecasting, cryptography, quieter submarines

Foreign Entities' Use of HPC to Pursue Applications of National Security Interest

This study catalogs an extensive set of applications of national security interest pursued by U.S. practitioners. An implicit assumption is that practitioners in other countries would need comparable computational power to carry out these or similar applications. An argument might be made that foreign entities of national security concern are not likely to mirror exactly

U.S. efforts in these application domains. More specifically, a foreign entity might pursue asymmetric applications in one of three ways:

(1) pursue applications not pursued by U.S. practitioners;

(2) pursue the same applications, but using different approaches that have different (i.e., lower) computing requirements;

(3) selectively pursue some applications and not others for lack of the scientific and technical wherewithal necessary to pursue a broad range of applications, or because such applications are not part of the country's national security objectives.

If either of the first two points are true, or become true at some future point, and the computing levels needed by such foreign practitioners fall below the lower bound of controllability, then the applications involved cannot be used to justify an HPC export control policy. The remaining applications, however, would still be usable.

In those cases where the third point is or becomes true, the HPC export control policy would still be effective as long as the selected applications are those that require levels of computing above the lower bound of controllability. A country that is able to acquire one or a few HPC systems might use them to pursue a small set of applications to the detriment of U.S. interests. Only the applications that the country is pursuing can be used to set an upper bound on HPC exports to that country.

It is difficult to acquire good information on the use of HPC for national-security-related applications by countries of national security concern. This is true whether one assumes foreign practice is the same as U.S. practice, or foreign practice involves different or more clever ways that might not have the same computing requirements. During our study, we approached the intelligence community on this issue, but were unable to locate information about foreign HPC uses. It is not clear whether the information is classified or is not known. It seems to be an area where little research is being done. Also, it is an area where it is difficult to determine how problems are being addressed. Unlike problems of counting ICBMs, for example, it is much harder to look inside foreign research institutions and determine what techniques they are using to solve problems, and the extent to which HPC is or is not essential. Mirror-imaging analysis has been used throughout much of the history of HPC export controls for precisely this reason.

Whatever the state of HPC use within a foreign country of concern to U.S. national security, two assertions are likely to be true. First, given the broad base of national security applications used in the United States, it is probable that any set of HPC applications pursued by a foreign country would be subset of the U.S. applications. Second, if a country lacks an established program in an application domain, the acquisition of a high-performance computer will not be sufficient to jump-start such a program. Pursuing nearly all the applications discussed in this study requires the presence of a number of enabling factors. Many applications, such as aircraft and submarine design, require highly sophisticated manufacturing technologies. But even if we limit our focus to the computational aspects of the applications, success rests on a three-legged stool consisting of: (1) hardware and systems software, (2) application codes, valid test data, and correct input data, and (3) the expertise necessary to use the codes and interpret results. If any of these is lacking, items (2) and (3) in particular, the application cannot be pursued successfully. Although much of the theoretical basis for the applications discussed in this chapter is available in the open literature, building and using large applications codes is a very difficult exercise. A promising way to develop true capability in one of the application domains is to start with small problems using small codes running on individual workstations and gradually migrate to larger problems, codes, and machines. This is a long-term process. If a country has not established some track record of pursuing a particular application over a multi-year period and given evidence of an ability to

develop and effectively use the associated codes and data, then a high-performance computing system by itself will be of little or no value and little threat to U.S. national security interests.

A final point must be made. Our methodology can accommodate data on foreign use of HPC, as it becomes available. It would constitute additional applications data points that would be factored into the determination of an upper bound for the control threshold. If the foreign data are classified, a classified annex to this report could be provided.

References

[1] Goodman, S. E., P. Wolcott, and G. Burkhart, *Building on the Basics: An Examination of High-Performance Computing Export Control Policy in the 1990s*, Center for International Security and Arms Control, Stanford University, Stanford, CA, 1995.

[2] Sunderam, V. S., "PVM: A framework for parallel distributed computing," *Concurrency-Practice and Experience*, Vol. 2, No. 4, Dec, 1990, pp. 315–339.

[3] Butler, R. and E. Lusk, *User's Guide to the p4 Parallel Programming System*, Technical Report ANL–92/17. Mathematics and Computer Science Division, Argonne National Laboratory, October, 1992.

[4] Document for a Standard Message-Passing Interface, Message Passing Interface Forum. March 22, 1994.

[5] Droegemeier, K. K. et al., "Weather Prediction: A Scalable Storm-Scale Model," in *High performance computing: problem solving with parallel and vector architectures*, Sabot, G. W., ed. Reading, MA: Addison-Wesley Publishing Company, 1995, pp. 45–92.

[6] Literature, Fleet Numerical and Oceanography Center, 1996.

[7] Drake, J., "Design and performance of a scalable parallel community climate model," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1571–1591.

[8] Drake, J. and I. Foster, "Introduction to the special issue on parallel computing and weather modeling," *Parallel Computing*, Vol. 21, No. 10, 1995, p. 1541.

[9] Hack, J. J., "Computational design of the NCAR community climate model," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1545–1569.

[10] Barros, S. R. M., "The IFS model: A parallel production weather code," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1621–1638.

[11] Sela, J. G., "Weather forecasting on parallel architectures," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1639–1654.

[12] Wehner, M. F., "Performance of a distributed memory finite difference atmospheric general circulation model," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1655–1675.

[13] Johnson, K. W., "Distributed Processing of a Regional Prediction Model," *Monthly Weather Review*, Vol. 122, 1994, pp. 2558–2572.

[14] Jahn, D. E. and K. K. Droegemeier, "Proof of Concept: Operational testing of storm-scale numerical weather prediction," *CAPS New Funnel*, Fall, 1996, pp. 1–3 (The University of Oklahoma).

[15] MM5 Version 2 Timing Results, National Center for Atmospheric Research, Feb. 3, 1998 (http://www.mmm.ucar.edu/mm5/mm5v2-timing.htm).

[16] Wallcraft, A., Interview, NRL, Stennis Space Center, Aug. 20, 1997.

[17] Wallcraft, A., Private Communication, Mar. 17, 1998.

[18] Sawdey, A. C., M. T. O'Keefe, and W. B. Jones, "A general programming model for developing scalable ocean circulation applications," in *ECMWF Workshop on the Use of Parallel Processors in Meteorology, Jan. 6, 1997*.

[19] Jensen, B., Interview, CEWES, Vicksburg, Aug. 22, 1997.

[20] Rosmond, T., Interview, Naval Research Laboratory, Monterey, July 28, 1997.

[21] Pfister, G. F., *In Search of Clusters. The Coming Battle in Lowly Parallel Computing*, Prentice-Hall, Inc., Upper Saddle River, NJ, 1995.

[22] Future Supercomputer Architecture Issues, National Security Agency, Feb. 11, 1997.

[23] Harger, R. O., *Synthetic Aperture Radar Systems: Theory and Design*, Academic Press, New York, 1970.

[24] Simoni, D. A. et al., "Synthetic aperture radar processing using the hypercube concurrent architecture," in *The Proceedings of the Fourth Conference on Hypercubes, Concurrent Computers, and Applications*, vol. II. :, 1989, pp. 1023–1030.

[25] Aloisio, G. and M. A. Bochicchio, "The Use of PVM with Workstation Clusters for Distributed SAR Data Processing," in *High Performance Computing and Networking, Milan, May 3-5, 1995. Proceedings*, Hertzberger, B. and G. Serazzi, eds. : Springer, 1995, pp. 570–581.

[26] Albrizio, R. et al., "Parallel/Pipeline Multiprocessor Architectcures for SAR Data Processing," *European Trans. on Telecommunication and Related Technologies*, Vol. 2, No. 6, Nov.-Dec., 1991, pp. 635–642.

[27] Brown, C. P., R. A. Games, and J. J. Vaccaro, "Implementation of the RASSP SAR Benchmark on the Intel Paragon," in *Proceedings of the Second Annual RASSP Conference, 24-27 July, 1995.* : ARPA, 1995, pp. 191–195.

[28] Brown, C. P., R. A. Games, and J. J. Vaccaro, *Real-Time Parallel Software Design Case Study: Implementation of the RASSP SAR Benchmark on the Intel Paragon*, MTR 95B95, The MITRE Corporation, Bedford, MA, 1995.

[29] Zuerndoerfer, B. and G. A. Shaw, "SAR Processing for RASSP Application," in *Proceedings of the 1st Annual RASSP Conference*. ARPA, 1994, pp. 253–268.

[30] Phung, T. et al., Parallel Processing of Spaceborne Imaging Radar Data, Technical Report CACR-108, California Institute of Technology, August, 1995.

[31] ESS Project FY97 Annual Report, NASA (http://sdcd.gsfc.nasa.gov/ESS/annual.reports/ess97/).

[32] Beck, A., "Mercury's B. Isenstein, G. Olin Discuss Architecture, Apps," *HPCWire*, Mar 28, 1997 (Item 10956).

[33] Mercury Delivers 38 GFLOPS With 200-MHz PowerPC Processors, *HPCWire*, Jan 31, 1997 (Item 10694).

[34] U.S. Navy To Use Mercury RACE Systems for Advanced Radar, *HPCWire*, Jun 27, 1997 (Item 11471).

[35] Mercury RACE Selected for Navy Sonar System, *HPCWire*, Jul 26, 1996 (Item 8916).

[36] Cain, K. C., J. A. Torres, and R. T. Williams, *RT*STAP: Real-Time Space-Time Adaptive Processing Benchmark, *MITRE Technical Report MTR 96B0000021*, The MITRE Corporation, Bedford, MA, 1997.

[37] Games, R. A., Benchmarking for Real-Time Embedded Scalable High Performance Computing, DARPA/Rome Program Review, May 6, 1997.

[38] Ballhaus Jr., W. F., "Supercomputing in Aerodynamics," in *Frontiers of* Supercomputing. Berkeley: University of California Press, 1986.

[39] Hatay, F. F. and S. Biringen, "Performance of a computational fluid dynamics code on NEC and Cray supercomputers: Beyond 10 Gflops," in *Proceedings of SC96, Nov. 1996, Pittsburgh, PA*. Los Alamitos: IEEE Computer Society, 1996,

[40] Conlon, J., "Propelling power of prediction: Boeing/NASA CRA leverages the IBM SP2 in

rotor analysis," *Insights*, No. 3, Sep, 1997, pp. 2–9.

[41] Jones, J. P. and M. Hultquist, "Langley-Ames SP2 Metacenter Enjoys Success, Plans for New Challenges," *NAS News*, Vol. 2, No. 25, July–August, 1997 (http://www-sci.nas.nasa.gov/Pubs/NASnews/97/07/article04.html. See also http://parallel.nas.nasa.gov/Parallel/Metacenter/).

[42] Claus, R. W. et al., "Numerical Propulsion System Simulation," *Computing Systems in Engineering*, Vol. 2, No. 4, 1991, pp. 357–364 (http://www.lerc.nasa.gov/WWW/CISO/projects/NPSS/).

[43] Stewart, M., ENG10 project web page, http://www.lerc.nasa.gov/WWW/ACCL/ENG10/.

[44] Stewart, M., Axisymmetric Jet Engine Analysis with Component Coupling, http://www.lerc.nasa.gov/WWW/ACCL/ENG10/PAPERS/components.ps.

[45] Van der Wijngaart, R. F., Efficacy of Code Optimizations on Cache-based Processors, NASA TechReport NAS-97-012, April 1997 (http://science.nas.nasa.gov/Pubs/TechReports/NASreports/NAS–97–012/).

[46] Taft, J., "Initial SGI Origin2000 Tests Show Promise for CFD Codes," NASNews, Vol. 2, No. 25, July-August, 1997 (http://www-sci.nas.nasa.gov/Pubs/NASnews/97/07/article01.html).

[47] Faulkner, T., "Origin2000 update: Studies show CFL3D can obtain 'reasonable' performance," *NAS News*, Vol. 2, No. 17, November–December, 1997 (http://science.nas.nasa.gov/Pubs/NASnews/97/11/).

[48] Fineberg, S. A. and J. C. Becker, "'Whitney' team weighs cluster network options," *NAS News*, Vol. 2, No. 27, November–December, 1997, p. 7.

[49] Sterling, T. et al., Findings of the first NASA workshop on Beowulf-class clustered computing, Oct. 22-23, 1997, Pasadena, CA, Preliminary Report, Nov. 11, 1997, Jet Propulsion Laboratory.

[50] Ramamurti, R. and W. C. Sandberg, "Simulation of flow past a swimming tuna," in *Contributions to DoD Mission Success from High Performance Computing 1996*. 1996, p. 52.

[51] Emery, M. H., J. P. Boris, and A. M. Landsberg, "Numerical Simulation of Underwater Explosions," in *NRL's DoD HPCMP Annual Reports (FY96)*.:,1996, .(http://amp.nrl.navy.mil/hpc/annual-reports/fy96/emery258.html)

[52] Morgan, W. B. and W.-C. Lin, Ship Performance Prediction, Krylov Centenary, Oct. 8–12, 1994, St. Petersburg, Russia.

[53] Morgan, W. B. and J. Gorski, Interview, Naval Surface Warfare Center, Jan. 5, 1998.

[54] Muzio, P. and T. E. Tezduyar, Interview, Army HPC Research Center, Aug. 4, 1997.

[55] U.S. Army HPC Research Center orders first Cray T3E-1200 Supercomputer, Cray Research Press Release, Nov. 18, 1997.

[56] Boris, J. et al., Interview, Naval Research Lab, Washington, DC, June 9, 1997.

[57] Ballbaus Jr., W. F., "Supercomputing in Aerodynamics," in *Frontiers of Supercomputing*, Metropolis, N. et al., eds. Berkeley, CA: University of California Press, 1986, pp. 195–216.

[58] Liu, S. K., *Numerical Simulation of Hypersonic Aerodynamics* and *the Computational Needs for the Design of an Aerospace Plane*, RAND Corporation, Santa Monica, CA, 1992.

[59] Simon, H. D., W. R. Van Dalsem, and L. Dagum, "Parallel CFD: Current Status and Future Requirements," in *Parallel Computational Fluid Dynamics: Implementation and Results*, Simon, H. D., ed. Cambridge: Massachusetts Institute of Technology, 1992, pp. 1–29.

[60] Holst, T. L., "Supercomputer Applications in Computational Fluid Dynamics," in *Supercomputing 88, Volume II: Science and Applications*, Martin, J. L. and S. F. Lundstrom, eds. Los Alamitos: IEEE Computer Society Press, 1988, pp. 51–60.

[61] Tezduyar, T. et al., "Flow simulation and high performance computing," *Computational Mechanics*, Vol. 18, 1996, pp. 397–412.

[62] Tezduyar, T., V. Kalro, and W. Garrard, *Parallel Computational Methods for 3D Simulation of a Parafoil with Prescribed Shape Changes*, Preprint 96-082, Army HPC Research Center, Minneapolis, 1996.

[63] Chaput, E., C. Gacherieu, and L. Tourrette, "Experience with Parallel Computing for the Design of Transport Aircrafts at Aerospatiale," in *High-Performance Computing and Networking. International Conference and Exhibition HPCN Europe 1996. Proceedings*, Liddel, H. et al., eds. Berlin: Springer-Verlag, 1996, pp. 128–135.

[64] Strietzel, M., "Parallel Turbulence Simulation Based on MPI," in *High-Performance Computing and Networking. International Conference and Exhibition HPCN Europe 1996. Proceedings*, Liddel, H. et al., eds. Berlin: Springer-Verlag, 1996, pp. 283–289.

[65] Chaderjian, N. M. and S. M. Murman, Unsteady Flow About the F-18 Aircraft, http://wk122.nas.nasa.gov/NAS/TechSums/9596/Show?7031.

[66] Ghaffari, F. and J. M. Luckring, Applied Computational Fluid Dynamics, http://wk122.nas.nasa.gov/NAS/TechSums/9596/Show?7126.

[67] Potsdam, M. A., Blended Wing/Body Aircraft, http://wk122.nas.nasa.gov/NAS/TechSums/9596/Show?7650.

[68] Sturek, W. et al., *Parallel Finite Element Computation of Missile Flow Fields*, Preprint 96-012, University of Minnesota Army HPC Research Center, Minneapolis, MN, 1996.

[69] Stein, K. et al., *Parallel Finite Element Computations on the Behavior of Large Ram-Air Parachutes*, Preprint 96-014, University of Minnesota Army HPC Research Center, Minneapolis, MN, 1996.

[70] Stein, K., A. A. Johnson, and T. Tezduyar, "Three-dimensional simulation of round parachutes," in *Contributions to DoD Mission Success from High Performance* Computing *1996*. :, 1996, p. 41.

[71] Pankajakshan, R. and W. R. Briley, "Efficient parallel flow solver," in *Contributions to DoD Mission Success from High Performance Computing*, 1996, p. 73.

[72] Army High Performance Computing Research Center, Army HPC Research Center, Minneapolis, MN, 1996.

[73] Rice, B. M., "Molecular Simulation of Detonation," in *Modern Methods for Multidimensional Dynamics Computations in Chemistry*, Thompson, D. L., ed. World Scientific Publishing Co., 1998 (To appear).

[74] Frisch, M. J. et al., *Gaussian 94*, Gaussian, Inc., Pittsburgh, PA, 1995.

[75] Rice, B. M., Interview, Army Research Lab, June 11, 1997.

[76] Binder, K., "Large-Scale Simulations in Condensed Matter Physics - The Need for a Teraflop Computer," *International Journal of Modern Physics C*, Vol. 3, No. 3, 1992, pp. 565–581.

[77] Pachter, R. et al., "The Design of Biomolecules Using Neural Networks and the Double Iterated Kalman Filter," in *Toward Teraflop Computing and New Grand Challenges. Proceedings of the Mardi Gras '94 Conference, Feb. 10-12, 1994*, Kalia, R. K. and P. Vashishta, eds. Commack, NY: Nova Science Publishers, Inc., 1995, pp. 123–128.

[78] Plimpton, S., "Fast Parallel Algorithms for Short-Range Molecular Dynamics," *Journal of Computational Physics*, Vol. 117, 1995, pp. 1–19.

[79] Nakano, A., R. K. Kalia, and P. Vashishta, "Million-Particle Simulations of Fracture in Silica Glass: Multiresolution Molecular Dynamics Approach on Parallel Architectures," in *Toward Teraflop Computing and New Grand Challenges. Proceedings of the Mardi Gras '94 Conference, Feb. 10–12, 1994*, Kalia, R. K. and P. Vashishta, eds. Commack, NY: Nova Science Publishers, Inc., 1995, pp. 111–122.

[80] Deng, Y. et al., "Molecular Dynamics for 400 Million Particles with Short-Range Interactions," in *High Performance Computing Symposium 1995 "Grand Challenges in Computer Simulation" Proceedings of the 1995 Simulation Multiconference, Apr. 9–13, 1995, Phoenix AZ*, Tentner, A., ed. : The Society for Computer Simulation, 1995, pp. 95–100.

[81] Camp, W. J. et al., Interviews, Sandia National Laboratories, Dec. 11, 1997.

[82] King, P. et al., "Airblast Effects on Concrete Buildings," in *Highlights in Computational Science and Engineering*. Vicksburg: U.S. Army Engineer Waterways Experiment Station, 1996, pp. 34–35.

[83] Kimsey, K. D. and M. A. Olson, "Parallel computation of impact dynamics," *Computational Methods in Applied Mechanical Engineering*, Vol. 119, 1994, pp. 113–121.

[84] Carrillo, A. R. et al., "Design of a large scale discrete element soil model for high performance computing systems," in *SC96 Proceedings, Pittsburg, PA, Nov. 1996*. Los Alamitos, CA: IEEE Computer Society, 1996, .

[85] Kimsey, K. and S. Schraml, Interview, Army Research Lab, Aberdeen Proving Ground, June 12, 1997.

[86] Farhat, C., "Finite Element Analysis on Concurrent Machines," in Parallel Processing in Computational Mechanics, Adeli, H., ed. New York: Marcel Dekker, Inc., 1992, ch. 7, pp. 183–217.

[87] Papados, P. P. and J. T. Baylot, "Structural Analysis of Hardened Structures," in *Highlights in Computational Science and Engineering*. Vicksburg: U.S. Army Engineer Waterways Experiment Station, 1996, pp. 64–65.

[88] Balsara, J. and R. Namburu, Interview, CEWES, Aug. 21, 1997.

[89] Horner, D. A., Interview, CEWES, Aug. 22, 1997.

[90] CCE Sets Record for Radar Profile Simulation, *HPCWire*, Jul 18, 1997 (Item 11576).

[91] Shang, J. S., "Computational Electromagnetics," *ACM Computing Surveys*, Vol. 28, No. 1, March, 1996, pp. 97–99.

[92] Shang, J. S., D. A. Calahan, and B. Vikstrom, "Performance of a Finite Volume CEM Code on Multicomputers," *Computing Systems in Engineering*, Vol. 6, No. 3, 1995, pp. 241–250.

[93] Shang, J. S. and K. C. Hill, "Performance of a Characteristic-Based, 3-D, Time-Domain Maxwell Equations Solver on the Intel Touchstone Delta," *Applied Computational Electromagnetics Society Journal*, Vol. 10, No. 1, May, 1995, pp. 52–62.

[94] Shang, J. S., Interview, Wright-Patterson AFB, June 18, 1997.

[95] Shang, J. S., "Characteristic-Based Algorithms for Solving the Maxwell Equations in the Time Domain," *IEEE Antennas and Propagation Magazine*, Vol. 37, No. 3, June, 1995, pp. 15–25.

[96] Shang, J. S., "Time-Domain Electromagnetic Scattering Simulations on Multicomputers," *Journal of Computational Physics*, Vol. 128, 1996, pp. 381–390.

[97] NCSA's Orign2000 Solves Large Aircraft Scatterer Problems, *HPCWire*, Jun 13, 1997 (Item 11365).

[98] Hurwitz, M., Interview, Naval Surface Warfare Center, Aug. 6, 1997.

[99] Hurwitz, M., E-mail communication, Dec. 2, 1997.

[100] Everstine, G., Interview, Naval Surface Warfare Center, Aug. 6, 1997.

[101] Karr, C. R., D. Reece, and R. Franceschini, "Synthetic soldiers," *IEEE Spectrum*, March, 1997.

[102] Garrett, R., "Creating a high level simulation architecture," in *Proceedings of the High Performance Computing Symposium: Grand Challenges in Computer Simulation, Apr. 9-13, 1995, Phoenix, AZ.* :, 1995, pp. 351–355.

[103] Brunett, S. and T. Gottschalk, Large-scale metacomputing gamework for ModSAF, A real-time entity simulation, Center for Advanced Computing Research, California Institute of Technology (Draft copy distributed at SC97, November, 1997).

[104] Miller, M. C., Interview, U.S. Army Chemical and Biological Defense Command (CBDCOM), Edgewood Arsenal, MD, June 12, 1997.

[105] Brunett, S. and T. Gottschalk, An architecture for large ModSAF simulations using scalable parallel processors, Center for Advanced Computing Research, Caltech. Draft paper received during interview at SPAWAR, San Diego, Jan. 13, 1998.

[106] Fusco, D., Private communication, March 27, 1998.

[107] Boner, K., D. Davis, and D. Fusco, Interview, Space and Naval Warfare Systems Center, San Diego, Jan. 13, 1998.

[108] Schuette, L., Interview, Naval Research Laboratory, Washington, DC, Jan. 9, 1998.

[109] Pritchard, B., Interview, Naval Warfare Systems Center, Jan. 13, 1998.

[110] Henderson, D. B., "Computation: The Nexus of Nuclear Weapons Development," in *Frontiers of Supercomputing*, Metropolis, N. et al., eds. Berkeley: University of California Press, 1986, ch. 12, pp. 141–151.

[111] Adams, T. et al., Interview, Los Alamos National Laboratory, Dec. 12, 1997.

[112] Accelerated Strategic Computing Initiative: PathForward Project Description, December 27, 1996, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, Sandia National Laboratory, 1996.

[113] Staffin, R., Interview, Department of Energy, Jan. 8, 1998.

[114] McMillan, C. F., P. S. Brown, and G. L. Goudreau, Interview, Lawrence Livermore National Laboratory, July 31, 1997.

[115] Wasserman, H. J. et al., "Performance Evaluation of the SGI Origin2000: A Memory-Centric Characterization of LANL ASCI Applications," in *SC97: High Performance Networking and Computing Conference Proceedings, San Jose, November 15-21, 1997.* Los Alamitos, CA: ACM Sigarch and IEEE Computer Society, 1997, .

[116] Wasserman, H., E-mail communication, Dec. 23, 1997.

[117] Ewald, R. H., "An Overview of Computing at Los Alamos," in *Supercomputers in Theoretical and Experimental Science*, Devreese, J. T. and P. van Camp, eds. New York and London: Plenum Press, 1985, pp. 199–216.

[118] The Role of Supercomputers in Modern Nuclear Weapons Design, Department of Energy (Code NN-43), Apr. 27, 1995.

[119] Games, R. A., Interview at Mitre Corp, June 23, 1997.

[120] Shang, J. S. and R. M. Fithen, "A Comparative Study of Characteristic-Based Algorithms for the Maxwell Equations," *Journal of Computational Physics*, Vol. 125, 1996, pp. 378–394.

Chapter 5: Policy Issues, Options, and Implications

Policy Options for a Control Threshold

As described in Chapter 1, a control threshold is viable only when it satisfies the three basic premises of the export control policy. These are:

(1) There exist applications of national security importance whose performance requirements lie above the control threshold;

(2) There exist countries of national security concern with the scientific and military wherewithal and interest necessary to pursue these applications should the necessary computational resources become available; and

(3) There are features of high-performance computing systems that make it possible to limit access by entities of national security concern to computational performance levels at or above the control threshold, given the resources the government and industry are willing to bring to bear on enforcement.

In this chapter we integrate the results of previous chapters to establish the range within which a control threshold may be viable and offer a number of policy options.

Lower-Bound Trends

The central question with respect to the lower bound of a viable range for a control threshold is what level of computational performance, across a broad spectrum of applications and workloads, is available to foreign entities of national security concern at various points in time. A control threshold set below this level is not likely to be effective. As was discussed in Chapter 3, the answer to the question is a function of several factors: the controllability of computing platforms, the availability of computing platforms from foreign sources beyond the control of the U.S. Government and its export control partners, and the scalability of platforms. Since controllability is a function not only of technological and market factors but also of enforcement resources and commitment, the determination of what is "controllable" is in part a decision that policy makers will have to make. In addition, policy makers will have to make a decision about how scalability is to be handled by the export-licensing process. If scalability is not taken into account explicitly, the computational performance attainable by end-users is likely to be considerably higher than if it is. Following a brief examination of past forecasts of controllability, this section presents a number of policy options and implications

with respect to the lower bound.

A Retrospective

The lower-bound projections made in 1995 were based on the assumption that the existing export-licensing process would remain fundamentally unchanged. That is, systems would be evaluated on the basis of the CTP of the precise configuration being exported, rather than on other factors such as the extent to which the system could be easily scaled to a larger configuration. The lower bound charted the performance level that could be reached by end-users by acquiring—legitimately or not—small configuration symmetrical multiprocessor systems and recombining them or adding boards to create larger configuration SMPs. The projections were made more conservative by incorporating a time lag such that systems did not factor into the projections until two years after they were introduced on the market. Given this regime, did the projections made in 1995 (see Figure 6 in [1] or Figure 3.3 in [2]) come to pass? For the most part, they did. The projections for 1996 and 1997 were based on systems introduced in 1994 and 1995. Developments after 1995 have not change those data points. Did the 1995 study accurately forecast the performance level end-users could attain in 1998? This level would have been based on systems introduced in 1996. Comparing Figure 3.3 of the earlier study with Figure 3–27 in Chapter 3, we can observe a few differences for 1998. The projected 16,000 Mtops performance of a Cray business systems division (the former CS 6400) did not come to pass. That division was sold to Sun Microsystems in 1996, and much of the CS 6400 technology was incorporated into the Ultra Enterprise 10000. The Ultra Enterprise 10000 was not introduced until 1997. Furthermore, we have chosen not to include it in the lower-bound curve, because, unlike other rack-based systems, it is not sold in small configurations and can therefore be "caught" by control thresholds set at the lower bound.

The SGI 1998 prediction (nearly 15,000 Mtops) appears to have been accurate. However, a qualification needs to be made. The system that defined that performance level was a 36-processor PowerChallenge (not PowerChallenge Array), introduced in 1996, based on the R10000/200 microprocessor. First, while this system was introduced in 1996, there were problems with the manufacturing of the R10000. Actual processors ran not at 200 MHz, but at 180 or, later, 195 MHz. Second, examination of the Top500 supercomputer list from November 1997 shows that the largest configuration sold was a 32-processor version (to the Japanese). There are numerous 24-processor installations, but no 36-processor installations. A conclusion that can be drawn about this (and, indeed, about all bus-based SMP platforms) is that applications have had difficulty in using the last few processors of a configuration effectively. Customers often do not feel that the additional cost of the full configuration is justified. While the nearly 15,000 Mtops level was theoretically attainable, it is unlikely that many users would actually try to install and use a full configuration.

Is the two-year time lag justified? We believe so. The rationale for that time lag was that two years would be needed for a secondary market to develop for the systems, usually rack-based, that defined that curve. The current study plots deskside systems using a one-year lag. A review of vendors of refurbished systems in early 1998 supports the prediction that by the end of 1997 or early 1998 systems with a full-configuration performance above 7000 Mtops would be available on the secondary markets. For example, refurbished DEC AlphaServer 8400 5/350 (CTP 7,639 Mtops, introduced in late 1996) and Sun Ultra Enterprise 4000/250, a high-end deskside system (CTP 7,062 Mtops, introduced in early 1997), are available on the secondary market. Refurbished systems vendors will configure and test the system before shipping, but usually perform no onsite services, except on special request and at additional cost. Year-old deskside systems are widely available on the secondary market. For example, AlphaServer 4100 5/466 (CTP 4033, introduced in early 1997) is widely available from brokers not only in the United States, but also in the Netherlands, Austria, and other countries. Likewise, individual boards are widely available from such sources. Systems introduced less than one year ago are largely absent from the secondary market. While the

scalability of systems has made it possible for end-users to reach a performance level of 7000 Mtops by legitimately acquiring configurations of lower performance, the presence on the secondary market of systems scalable to this level means that this performance can be reached even more easily without the knowledge of the U.S. Government, the HPC manufacturers, or affiliated organizations.

The Lower Bound Toward the Close of the Century

The course of the lower bound through the end of the century will depend on the decisions policy makers make regarding:

(1) Whether to continue to use configuration performance or end-user attainable performance as the basis for licensing decisions. If the latter,

(2) What constitute controllable categories of computing platforms for the amount of resources government and industry are willing and able to bring to bear on monitoring and enforcement.

If the export-licensing process continues to use configuration performance as the basis for a licensing decision, then the lower bound of controllable performance will rise very quickly. Foreign entities will be able to purchase small configurations legitimately, and, in the field, add additional CPUs, memory, and I/O capability on their own to reach much higher levels of performance than they originally acquired. The lower bound of controllability is defined by the extent to which users by themselves can acquire and scale systems with small minimum configurations, such as rack or, in some cases, multi-rack systems. Using the two-year lag described earlier, the lower bound is likely to increase to over 23,000 Mtops in 1999 and close to 30,000 Mtops in 2000, given a continuation of current licensing practices.

If licensing practices are changed to consider end-user attainable performance rather than configuration performance in licensing decisions, additional options become available to policy makers. Rather than treating all configurations at a given CTP level as equivalent, an approach based on end-user attainable performance differentiates systems that can be easily scaled to higher performance levels from those that cannot. For example, four processors in a chassis that can accommodate eight processors might be treated differently from four processors in a chassis that can accommodate sixteen processors. Control measures are based directly on that which concerns policy makers: the performance an end-user can attain. This concept is discussed at length in Chapter 3. A regime based on end-user attainable performance provides policy makers with a broader range of policy options because of two characteristics of the technologies and the markets: (a) systems with higher end-user attainable performance can be distinguished and treated differently from systems with lower end-user attainable performance, and (b) systems with higher end-user attainable performance are more controllable than those with lower end-user attainable performance. End-user attainable performance correlates with a number of factors that affect the controllability of computing platforms, including the installed base, price, size, and dependence on vendor support. More powerful systems tend to be sold in smaller numbers, be more expensive, have larger configurations, and require greater amounts of vendor support for installation and maintenance. Table 5–1, reproduced from Chapter 3, illustrates some of these qualities for seven categories of computing systems available in the fourth quarter of 1997. Systems based on non-commodity processors, such as vector-pipelined processors, have end-user attainable performance equivalent to the configuration performance, since they are not upgradeable by end-users without vendor support.

Type	*Units installed*	*Price*	*End-user attainable performance*
Multi-rack HPC systems	100s	$750K–10s of millions	20K+ Mtops
High-end rack servers	1000s	$85K–1 million	7K–20K Mtops
High-end deskside servers	1000s	$90–600K	7K–11K Mtops
Mid-range deskside servers	10,000s	$30–250K	800–4600 Mtops
UNIX/RISC workstations	100,000s	$10–25K	300–2000 Mtops
Windows NT/Intel servers	100,000s	$3–25K	200–800 Mtops
Laptops, uni-processor PCs	10s of millions	$1–5K	200–350 Mtops

Table 5–1. Categories of computing platforms (4Q 1997)

Under a system based on configuration performance, a dual-processor workstation, a dual-processor mid-range deskside system, and a dual-processor high-end rack server are indistinguishable, even though these platforms have rather different controllability factors. The use of end-user attainable performance helps to distinguish between them and, more generally, distinguish platforms that are controllable from those that are not. Which of these categories are controllable and which are not? Policy makers must answer this question since it is in large part a function of the resources government and industry are willing to bring to bear on enforcement.

Table 5–2 shows four possible decision options and the corresponding lower-bound performance levels (projected). As described earlier, the figures incorporate lag times of two years when rack-based systems and some high-end deskside systems are considered, and lag times of one year for smaller systems. Lower-bound figures for 2001 and possibly the end of 2000 are likely to increase sharply as markets mature for servers based on processors like Intel's Merced.

Decision	Lower bound performance in year		
	1998	1999	2000
Option 1. Continue to use configuration performance as basis for licensing decisions	14,800	23,500	29,400
Option 2. Use end-user attainable performance, and consider high-end rack-based systems controllable but high-end deskside systems uncontrollable	7100	11,300	13,500
Option 3. Use end-user attainable performance, and consider high-end deskside systems controllable, but mid-range deskside systems uncontrollable	4600	5300	6500
Option 4. Use end-user attainable performance, and consider deskside midrange systems controllable, but workstations, PC servers, and PCs uncontrollable.	2500	3600	4300

Table 5–2. Trends in lower bound of controllability (Mtops)

Option 3 requires a great deal more export licensing and enforcement effort than option 2, which requires a great deal more effort than option 1. In general, the more conservative the choice for the lower bound of controllability, the greater the resources and effort needed to support that decision.

How do these lower-bound figures compare with the control thresholds established by the administration in 1995? End-user attainable performance did not play a role then in export-licensing considerations. However, the numbers chosen in 1995 are not inconsistent with those shown above. The 7000 Mtops threshold corresponds to the performance of rack-based systems introduced in 1995. The 2000 Mtops threshold is slightly below the performance of mid-range deskside systems introduced in the same year. In short, these numbers lay along the same trend lines shown for later years under option 1 and option 2 in Table 5–2.

Upper-Bound Trends

Just as the lower bound of controllability can be used to determine a performance level below which a threshold is not viable, the performance requirements of applications of national security interest can be used to determine an upper bound above which a threshold should not be set. The upper bound is in part a function of what policy makers feel constitutes the set of applications "of great national security importance." There are two principal positions that can be taken, each with its own implications for the choice of an upper bound:

(1) The set of applications of great national security importance is broad and deep, distributed across the entire spectrum of computing performance. All applications should be protected to the extent possible.

    Under this view, the upper bound collapses on top of the lower bound, forcing control thresholds to track the lower bound closely.

(2) The export control policy should try to protect a subset of the applications of national security interest. It may be limited to those of critical national security importance, or to those that can be effectively carried out by foreign entities in ways that threaten U.S. national security interests. The choice of upper bound may also try to protect performance levels with high density of applications.

    Under this view, the upper bound may be above the lower bound, creating a range of performance levels within which a threshold may be established.

Chapter 4 establishes that there exist applications of national security importance requiring computational performance above any lower-bound figure shown in Table 5–2. The number of such applications is likely to grow dramatically in the coming years as increasing numbers of practitioners gain access to very powerful systems. There is no shortage of computationally demanding problems that will consume virtually any number of CPU cycles that become available. If policy makers hold the first position described above, this study provides abundant support for a continuation of the export control policy.

If policy makers hold the second position, the answer to the question, "What is the upper bound of a range of viable control thresholds?" is a function of both the kinds and quantities of applications with requirements at various performance levels above the lower bound of controllability. The applications examined in this study are listed in Appendix A, sorted by the CTP of the configuration used. Table 5–3 shows the rising trend lines for the years 1998–2000 of the lower bound of controllability under the four options presented in Table 5–2. The tables show a sampling of applications that may fall below the lower bound of controllability in each of the three years under each policy option.

Table 5–3 shows only a subset of the applications listed in Appendix A—those that in the authors' estimation are among the more significant. It is, however, the responsibility of the national security community to decide which applications are, in fact, the most important to protect. The identification of individual, particularly significant applications can be one factor helping to determine an upper bound for a control threshold.

Application	CTP	Configuration performance used for licensing decisions	End-user attainable performance used for licensing decision and... High end, rack-based systems considered controllable	High end deskside systems considered controllable	Midrange deskside systems considered controllable
6 million cell 3D shock physics simulation	44,000				
Steady-state simulation of parachute airfoil	32,398				
Large Mercury on-board radar system shipped in 1997	27,113	2000			
MD simulation of 400 million particles	24,520				
Synthetic forces experiments 20,290 entities	24,520				
Hardened structure with internal explosion.	21,125	1999			
Global atmospheric forecast, Fleet Numerical 30 vertical layers; ;	21,125				
Explosion engulfing a set of buildings	21,125				
Modeling of turbulent flow about a submarine	21,125				
Simulation of fighter aircraft at Mach 2.0.	20,057				
Flare maneuver of a large ram-air parachute.	20,057				
Aerodynamics of missile at Mach 2.5, 14-degrees angle of attack.	17,503				
10913 vehicle synthetic forces experiments	16,737				
Global atmospheric forecast, Fleet Numerical 24 verticle layers;	15,875				
Water over C4 over sand	15,875				
Crash code, PAM	14,698	1998			
5086 vehicle synthetic forces experiments	13,147		2000		
MD simulation of 102.4 million particles	12,680				
SC-MICOM, global ocean forecast	11,768				
Global atmospheric forecast, Fleet Numerical operational run	10,625		1999		
Long-range unmanned aerial vehicles (UAV) on-board data procesing	10,000				
Large-Eddy simulation at high Reynolds number	9,510				
Development of algorithms for Shipboard Infrared search & tracking (SIRST)	8,980				
2D FFT. 200 one-Mpixel images/s	8,263				
Bottom contour modeling of shallow water in submarine design.	8,000				
Topological Synthetic Aperture Radar data processing .	8,000				
50 million particle molecular dynamics simulation	7,315				
Helicopter rotor free-wake model, high-order vortex element and wake relaxation.	7,100			2000	
Modeling of turbulent flow around a smooth ellipsoid submarine.	5,375		1998		
JAST aircraft design	4,864			1999	
Helicopter rotor motion coupled with rotor CFD	4,745				
Non-acoustic anti-submarine warfare sensor development	4,600				
Model of steady state flow around a fully appended submarine .	3,708			1998	2000
3D shock physics simulation	3,708				
Fully turbulent flow around small unmanned vehicle.	3,485				1999
Full ship flow model.	3,485				
Aerodynamics of missile at Mach 3.5	2,142				1998

Table 5–3. Selected sample of national security applications and the rising lower bound of controllability

Another factor that may be taken into account in determining an upper bound is the relative density of applications of national security importance at various performance levels. The basic reality is that applications can be found across the entire spectrum of CTP levels. Any level of computing power can be useful if a practitioner has the appropriate knowledge, codes, and input and verification (test) data. Nevertheless, are there performance levels at which one finds relatively larger numbers of applications than at others? Not surprisingly, these tend to be found around the performance levels of "workhorse" computing systems. Figure 5–1 shows the distribution of the CTP of configurations used for applications examined in this study and listed in Appendix A.

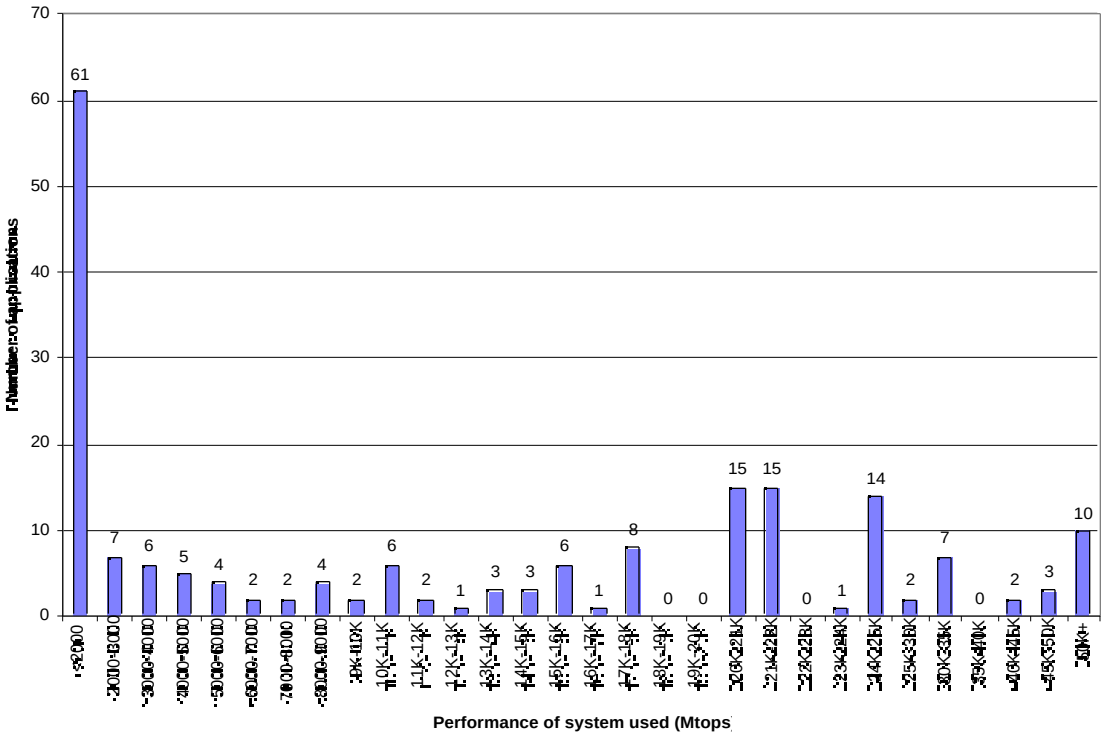

Figure 5–1. Distribution of applications examined in this study

Those of particular interest lying above some of the lower bounds discussed above are:

- 4000–6000 Mtops. Some key applications in this range are JAST aircraft design, non-acoustic anti-submarine warfare sensor development, and advanced synthetic aperture radar computation. The number of applications at this performance range is likely to grow rapidly since the number of systems in this range is growing quickly.
- 8000–9000 Mtops. Applications here include bottom-contour modeling of shallow water in submarine design, some synthetic aperture radar applications, and algorithm development for shipboard infrared search and track.
- 10,000–12,000 Mtops. Here we find a number of important weather-related applications.
- 15,500–17,500 Mtops. In this range, we find substantial fluid dynamics applications able to model the turbulence around aircraft under extreme conditions.
- 20,000–22,000 Mtops. This large cluster includes many applications run on full-configuration Cray C90s, including weather forecasting, impact of (nuclear) blasts on underground structures, and advanced aircraft design.

While these clusterings may be residual effects of the kinds of computing platforms employed by U.S. practitioners, they should be kept in mind by policy makers as they decide where to set control thresholds for Tier 2 and Tier 3 countries.

Countries of National Security Concern

The discussion above has established that it is possible today to set a control threshold such that the first and third basic premises of the export control policy are satisfied. Does the second basic premise hold as well? Are there countries of national security concern with the scientific and military wherewithal to purse the applications discussed in this study?

Pursuing nearly all the applications discussed in this study requires the presence of a number of enabling factors, including application codes, valid test and input data, and the expertise necessary to use the codes and interpret results. Acquisition of HPC hardware and systems software does not directly provide these other factors. If a country has not established some track record of pursuing a particular application over a multi-year period and given evidence of an ability to develop and effectively use the associated codes and data, then a high-performance computing system by itself will be of little or no value and little threat to U.S. national security interests.

Which countries have demonstrated the expertise necessary to use high-performance computers, and for which applications? One unquestionable example is nuclear weapons development and stockpile stewardship. Russian scientists and, to a lesser extent, the Chinese, have demonstrated that they have the ability to use high-performance computing resources effectively to pursue these applications. However, the level of computational capability needed to develop the equivalent of today's U.S. stockpile, when nuclear test data is available, is at or below the performance of many workstations. At the level of computing readily available today and the near future, additional computing power does not compensate for the lack of test data. However, the U.S. Government has made it policy that stockpile stewardship, requiring much higher levels of capability, is an application of concern for U.S. national security interests.

Weather forecasting is another application in which a variety of countries have demonstrated long-term expertise. Furthermore, the need for high levels of computing power is even more critical here than in nuclear weapons development, in large part because of the large volumes of data and strict timing limitations needed.

In contrast, many Tier 2 and Tier 3 countries have little or no experience in a host of national security applications. For example, there is little danger of most of these countries usefully employing HPC to develop submarines, advanced aircraft, composite materials, or a variety of other devices.

A critical question, which we have been unable to pursue satisfactorily in this study, is which countries are able and likely to use HPC to pursue which applications. We have requested such information from the U.S. national security community, but have received few answers. It does not appear that the U.S. Government is effectively gathering such intelligence in a systematic fashion. More specifically, the U.S. Government does not appear to have as good an understanding of individual end-use organizations of concern as is needed by the export control regime.

In contrast to nearly every other country in the world, the United States has a rich and deep base of experience, code, and data—not to mention hardware and systems software—that makes it uniquely positioned to pursue a very broad spectrum of computationally demanding applications. The U.S. community has benefited more than any other nation's from the dramatic growth of HPC capabilities and markets. The breadth and depth of U.S. capability in HPC applications is a national strategic advantage that should continue to be cultivated.

Export Control at the Turn of the Century

This study has concluded that the export control regime can remain viable through the end of the century and offers policy makers a number of alternatives for establishing thresholds and licensing practices that balance national security interests and the realities of HPC technologies

and markets. Some of these options offer policy makers a broader range of viable control thresholds than in the past. At the same time, there are several important trends in applications, technologies, and markets that will affect not only the range within which thresholds may be viable, but also how successful the policy is in achieving its objectives. These trends are those that affect the lower bound and the application of HPC technologies, particularly those of a less controllable nature, to problems of national security importance.

Trends Affecting the Lower Bound of Controllability

Trends impacting the lower bound of controllability were discussed at length in Chapters 2 and 3. Those having the strongest impact on future policies include:

- Rapidly rising microprocessor performance. If industry claims are true, CTP levels for microprocessors in full volume production will be at nearly 10,000 Mtops in 2001 or 2002. Conventional systems with modest numbers (4–8) of such processors and annual sales in the tens or hundreds of thousands of units will eclipse the performance of all but a fraction of today's highest-end supercomputers. Like today's Intel-based servers, these systems will be available through a myriad of outlets throughout the world.
- The expansion of commercial markets for high-end multiprocessors. To a large extent, parallel processing has become commonplace in commercial markets. Data warehousing, data mining, financial analysis simulation, and web server applications have become critical to national and multinational corporations. The result, already observed, is a dramatic expansion of the installed base of multiprocessor systems.
- The improvement of interconnects and software needed to create large configurations out of widely available hardware and software components. One of the important developments during the mid-1990s has been the emergence of so-called clustering interconnects offering substantial performance improvements over traditional local area networking technologies. Clusters based on these technologies are increasingly useful and available, although at present they lack the performance and manageability of large proprietary configurations across broad categories of applications. However, progress is being made rapidly; the gap in performance and manageability between more proprietary systems and those based on commercially available, off-the-shelf components should be tracked on an ongoing basis.
- Improved hardware and software scalability of individual vendor products. The existence of a lower bound of performance controllability is based on the premise that there exist discontinuities in scalability. That is, at certain points scaling to larger systems requires special hardware, software, or vendor expertise; results in disproportionate degradation of performance; or both. At present such discontinuities are often found as one scales from within a box to outside a box, from small multi-rack to large multi-rack configurations, or reaches the limits of the number of processors an operating system can accommodate, etc. If scalability across a large range of processors becomes not only seamless, but possible by an end-user without vendor involvement or specialized technologies, then it becomes very difficult to establish a defensible lower bound of performance controllability.

Trends in the Highest-End Systems

Improvements in microprocessors will drive the performance of uncontrollable platforms sharply upward. Because the highest-end systems will be built using these same microprocessors, however, they will ride the same performance curves. While today several million dollars must be spent to acquire a 50,000 Mtops computer, in a few years the same amount of money may acquire a system ten to twenty times more powerful. There will

continue to be low-end and high-end configurations, and a sizable performance gap between the two.

Will the lower bound of performance controllability ever rise to engulf all systems available to practitioners? If the number of systems whose performance is above the lower bound of performance controllability is zero, then no viable control threshold can be drawn without violating one of the basic premises of the export control policy. For the foreseeable future, the answer in the strictest sense will be "no," as long as entities such as the U.S. Government or commercial organizations are willing to fund, or participate in, leading-edge systems development. To say that the ASCI Red system at Sandia is "just a collection of PCs" is incorrect. A great deal of specialized expertise, software, and, to a lesser degree, hardware is involved in the development of this kind of system. Given the highly dynamic, competitive, and innovative environment of the HPC industry, it is difficult to imagine that a well-funded group of vendor and practitioner experts would not be able to create a system that offered some level of capability not available to a practitioner in another country.

Computing systems with performance of one, ten, or one hundred Tflops, such as those created under the ASCI program, will continue to be available in very small numbers and at very high cost. The level of performance they provide will continue to be controllable. But what about the performance offered by systems installed in hundreds of units? Will practitioners throughout the world be able to acquire or construct systems with a performance comparable to all but a few dozen of the world's most powerful systems? The answer is very dependent on the progress vendors make in the area of scalability and manageability.

Trends in Applications of National Security Importance

From the perspective of the export control regime, the most significant observation about HPC applications is that there will almost certainly always be applications of national security importance that will demand more computational resources than are available. This fact has been true for decades, and there is no reason to expect any change.

The most significant change since the early 1990s has been the extent to which parallel platforms and applications have entered the mainstream of scientific and commercial, research and production activities. This shift has been fueled by a number of contributing factors:

- the emergence of a vendor-independent application-programming interface, the Message Passing Interface (MPI), that offers adequate performance and insulation from changes in hardware and systems software;
- the maturing of vendors' hardware and software environments, making the systems reliable enough for practitioners other than the early adopters;
- a growing body of expertise regarding the development of parallel applications that reduces the learning curve for individual practitioners;
- the memory available on distributed-memory multiprocessors is substantially greater than that of traditional vector-pipelined processors, making MPPs the only practical platform for truly large applications;
- the acquisition of Cray Research, Inc., by Silicon Graphics, Inc., which has contributed to doubts about the future of parallel vector-pipelined systems.

While some applications remain wedded to the parallel vector-pipelined systems, these are becoming the exception rather than the rule. The significance for the export control regime is that no longer are high-end systems required for developing applications that can run on high-end systems. In past decades, foreign practitioners were unable to gain the expertise necessary

to use a Cray vector-pipelined system because they lacked access to this hardware. Developing applications to run on a Cray would be a pointless exercise. Today, however, MPI runs on all parallel platforms, from clusters of workstations to the largest integrated MPP. Developing applications to run on workstation clusters provides a great deal of experience and code that can be ported to larger platforms as they become available. The lack of hardware may at present be a barrier to achieving levels of performance, but it is no longer a great barrier to acquiring a good deal of the expertise needed to use large systems.

Future Effectiveness of the Export Control Policy

While an export control policy may remain viable at present, in the future "leakage" of the policy is likely to become much more common than it is today. The probability will increase that individual restricted end-use organizations will be successful in acquiring or constructing a computing system to satisfy a particular application need. This leakage will come about as a result of many of the trends discussed above.

First, if policy makers do decide that systems with installed bases in the thousands are controllable, it is almost inevitable that individual units will find their way to restricted destinations. Because of the nature of today's technologies and international patterns of trade, the cost in money and delay of covert acquisition of powerful computers is less today than even six to ten years ago.

Second, industry is working intensively toward the goal of seamless scalability, enhanced systems management, single-system image, and high efficiency across a broad range of performance levels. Systems with these qualities make it possible for users to develop and test software on small configurations yet run it on large configurations. An inability to gain access to a large configuration may limit users' ability to solve certain kinds of problems, but will not usually inhibit their ability to develop the necessary software or expertise.

Third, although clustered systems are likely to remain less capable than the integrated vendor-provided systems over the next few years, they are improving both in the quality of the interconnects and the supporting software. Foreign users are able to cluster desktop or deskside systems into configurations that perform useful work on some, perhaps many, applications. The control regime will not be able to prevent the construction of such systems. However, until clustered systems become comparable to more tightly integrated, vendor supplied and supported systems in every sense—systems management, I/O capability, software environment, and breadth of applications, as well as raw or even sustained performance—some advantage may be gained by restricting the export of the latter. How much benefit is a question that should be the subject of ongoing discussion.

Finally, given the nature and volume of foreign sales of today's HPC market, where most U.S. vendors sell approximately 50 percent of their output abroad, monitoring and control of large numbers of systems is difficult. It is exacerbated by the apparent lack of an effective multilateral regime governing re-exports. It is, moreover, difficult to imagine effective re-export controls in today's political and commercial world that would prevent the acquisition of one or two machines by many of the end-users of greatest concern.

Nevertheless, even an imperfect export control regime may offer a number of benefits to U.S. national security interests.

First, licensing requirements should force vendors and government to pay closer attention to who the end-users are and what kinds of applications they are pursuing. Vigilance is increased, and questionable end-users may be more reluctant to acquire systems through legitimate channels.

Second, the licensing process provides the government with an opportunity to review end-users brought to its attention. It forces the government to increase its knowledge of end-users throughout the world. The government should improve its understanding of end-users of concern so that it can make better decisions regarding those end-users.

Finally, while covert acquisition of high-performance computers is much easier today than in the past, users without legitimate access to vendor support are at a disadvantage. The more powerful the system, the greater the dependence on vendor support. Moreover, operational or mission-critical systems are much more dependent on fast, reliable support. End-users of concern pursuing these applications may be at a disadvantage even if the hardware itself is readily available and scalable.

Outstanding Issues and Concerns

Periodic Reviews. This study documents the state of HPC technologies and applications during 1997–early 1998, and makes some conservative predictions of trends in the next two to five years. The pace of change in this industry continues unabated. The future viability of the export control policy will depend on its keeping abreast of change and adapting in an appropriate and timely manner. When based on accurate, timely data and an open analytic framework, policy revisions become much sounder, verifiable, and defensible. There is no substitute for periodic reviews and modification of the policy. While annual reviews may not be feasible given policy review cycles, the policy should be reviewed every two years at the most.

The use of end-user attainable performance in licensing. The use of end-user attainable performance in licensing decisions is a departure from past practice. It is a more conservative approach to licensing in that it assumes a worst-case scenario, that end-users will increase the performance of a configuration they obtain to the extent they can. By the same token, however, it reduces or eliminates a problematic element of export control enforcement: ensuring that end-users do not increase their configurations beyond the level for which the license had been granted. If U.S. policy makers do not adopt the use of end-user attainable performance, then the burden of ensuring post-shipment compliance will remain on the shoulders of HPC vendors and U.S. Government enforcement bodies. If they do, then post-shipment upgrades without the knowledge of U.S. vendors or the U.S. Government should not be a concern, having been taken into account when the license was granted.

Applications of national security importance. The current study has surveyed a substantial number of applications of national security importance to determine whether or not there are applications that can and should be protected using export controls of high-performance computing. While the study has enumerated a number of applications that *may* be protected, it has not answered the question of which applications are of greatest national security importance and *should* be protected. This question can only be answered by the national security community, and it is important that it be answered. If an application area lacks a constituency willing to defend it in the public arena, it is difficult to argue that it should be a factor in setting export control policy.

During the Cold War, when the world's superpowers were engaged in an extensive arms race and building competing spheres of influence, it was relatively easy to make the argument that certain applications relying on high-performance computing were critical to the nation's security. Because of changes in the geopolitical landscape, the nature of threats to U.S. national security, and the HPC technologies and markets, the argument appears to be more difficult to make today than in the past. We have found few voices in the applications community who feel that export control on HPC hardware is vital to the protection of their application. Constituencies for the nuclear and cryptographic applications exist, although they are far from unanimous in their support of the policy. An absence of constituencies in other application areas who strongly support HPC hardware export controls may reflect an erosion of the basic premises underlying the policy. If this is the case, it should be taken into account; where such constituencies exist, they should enter into the discussion.

References

[1] Goodman, S. E., P. Wolcott, and G. Burkhart, *Building on the Basics: An Examination of High-Performance Computing Export Control Policy in the 1990s*, Center for International Security and Arms Control, Stanford University, Stanford, CA, 1995.

[2] Goodman, S., P. Wolcott, and G. Burkhart, *Executive Briefing: An Examination of High-Performance Computing Export Control Policy in the 1990s*, IEEE Computer Society Press, Los Alamitos, 1996.

Appendix A: Applications of National Security Importance

The following table provides a summary of national security applications reviewed in this study. It indicates the kind of problem solved, the HPC configuration on which it was solved, and the time required for the solution.

This selection, compiled through a combination of direct communications with practitioners and a review of published literature, is not an exhaustive listing. However, it does include many of the more important national security applications, and gives policy makers a rough idea of the kinds of applications being solved at various performance levels.

Two points in particular should be kept in mind when reading the table. First, the applications shown here constitute data points that often, in practice, lie along a continuum. The specific size of the application is often a function of the computational resources available in a given configuration and the "threshold of patience" of the practitioner. If the configuration available were slightly larger, or smaller, the practitioners in most cases would solve the same kinds of problem, but perhaps with a different grid size, or time-step, etc. In short, the absence of a particular type of application at some performance level should not be interpreted as a statement that no version of that application can be solved at that performance level.

Second, the CTP value shown is the composite theoretical performance of the configuration used to solve the problem. It is well known that any metric, including the CTP, does not perfectly predict the performance of all systems on all applications. Consequently, the CTP measures given here should be used only as rough indicators of the performance level required for a particular kind of application. The fact that a given problem was run on a machine with a CTP of *n* Mtops does not mean that all systems with CTP > *n* Mtops can solve the problem, or that all systems with CTP < *n* Mtops can not. The CTP simply does not have this kind of precision.

The following acronyms are used for applications categories:

CCM	Computational Chemistry and Materials Science	CWO	Climate/Weather/Ocean Modeling and Simulation
CEA	Computational Electromagnetics and Acoustics	FMS	Forces Modeling and Simulation/C4I
CFD	Computational Fluid Dynamics	Nuclear	Nuclear weapons development and stockpile maintenance
CSM	Computational Structural Mechanics	SIP	Signal/Image Processing

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
VAX 6210		1	SIP	35 min	Focus an image [1]	5040 x 1260 samples (18km x 8km)
Cray-1	1984	195	CFD	1.4 CPU hours	SCRAMJET wing-fuselage aerodynamic interaction simulation [2]	56,730 grid points, Mach 6
Cray-1S	1984	195	CFD	20 CPU hr	Simulation of after-body drag for a fuselage with propulsive jet. Reynolds averaged Navier-Stokes [3]	
Cray-1S	early 1980s	195	nuclear	1127.5 CPU sec	LANL Hydrodynamics code 3 [4]	
Cray-1S	early 1980s	195	nuclear	117.2 CPU sec	LANL Hydrodynamics code 1 [4]	
Cray-1S	early 1980s	195	nuclear	4547.1 CPU sec	LANL Hydrodynamics code 2 [4]	
Cray-1		195		24 hours	Crash simulation [5]	5,500 elements
Cray-1S		195	CFD		Nonlinear Inviscid (STAGE II): Above, plus transonic pressure loads; wave drag [3,6]	100,000 grid points
Cosmic Cube (6)	1991	293	SIP	1.81 millisec	Discrete Fourier transform algorithm [7]	5040 complex data sample
Cray X-MP/1	mid 1980s	316	nuclear		Two-dimensional, reduced physics simulation	
Cray XMP/48 (1proc)	1988	353	CFD	20-50 hr	Flow simulation around complete F-16A aircraft (wings, fuselage, inlet, vertical and horizontal tails, nozzle) at 6 deg angle of attack, Mach 0.9. Reynolds Averaged Navier-Stokes. Reynolds number = 4.5 million. [8]	1 million grid points, 8 Mwords (one 2 Mword zone in memory at a time), 2000-5000 iterations
Cray XMP/1	1990	353	CCM	1000 CPU hours	Molecular dynamics of bead-spring model of a polymer chain [9]	Chain length=400
Cray J916/1	1996	450	CFD	300 CPU hr	Modeling of transonic flow around AS28G wing/body/pylon/nacelle configuration. 3–D Reynolds averaged full Navier-Stokes solution. [10]	3.5 million nodes, 195 Mwords memory
Origin2000/1	1997	459	SIP	5.650 s	RT_STAP benchmark (hard) [11]	2.7 million samples per .161 sec
Cray Y-MP/1	1987	500	CSM	200 hours	3–D shock physics simulation [12]	200,000 cells
Cray YMP/1	1990	500	CSM	39 CPU sec	Static analysis of aerodynamic loading on solid rocket booster [13]	10,453 elements,9206 nodes,54,870 DOF 256 Mword memory
Cray YMP	1991	500	CCM	1000 CPU hours	Molecular dynamics modeling of hydrodynamic interactions in “semi-dilute” and concentrated polymer solutions [9]	single chain, 60 monomers
Cray YMP	1991	500	CCM	1000 CPU hours	Modeling of thermodynamics of polymer mixtures [9]	lattice size - 112EXP(3); chain size = 256
Cray YMP/1	1991	500	CFD	40 CPU h	Simulation of viscous flow about the Harrier Jet (operating in-ground effect modeled) [14]	2.8 million points, 20 Mwords memory.
Cray YMP/1	1993	500	CCM	1.47 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid-state point [15]	100,000 atoms
Cray YMP/1	1996	500	CFD	170 CPU hr (est)	Modeling of transonic flow around AS28G wing/body/pylon/nacelle configuration. 3D Reynolds Averaged full Navier-Stokes solution [10]	3.5 million nodes, 195 Mwords memory
Cray YMP/1	1996	500	CFD	6 CPU hr	Modeling of transonic flow around F5 wing (Aerospatiale). 3D Reynolds Averaged full Navier-Stokes solution [10]	442368 cells (192x48x48).
Cray YMP	1996	500	CFD	8 CPU hr, 3000 timesteps	Large-Eddy simulation at high Reynolds number [16]	2.1 million grid points, 44 Mwords

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
workstation	1997	500	CSM		2–D modeling of simple projectile striking simple target. [17]	10,000s of grid points
Cray Y-MP/1	late 1980s	500	nuclear	1000s hours	two-dimensional, almost full physics (e.g. Monte Carlo neutron transport)	
Cray Y-MP/1	late 1980s	500	nuclear	seconds	one-dimensional, full physics simulation	
Cray YMP/1		500	CEA	5 CPU hr per timestep	Signature of modern fighter at fixed incident angle at 1 GHz [18]	50 million grid points @ 18 words/grid point
Cray Y-MP/1		500	nuclear		two-dimensional, reduced physics simulations	100-500 Mbytes
Mercury Race (5 x 4 i860 processors, Ruggedized)	1997	866	SIP		SAR system aboard P3-C Orion maritime patrol aircraft [19]	
Cray YMP/2 256 MW	1990	958	CSM	19.79 CPU s	Static analysis of aerodynamic loading on solid rocket booster [13]	10,453 elements, 9206 nodes, 54,870 DOF
Cray Y-MP/2	late 1980s	958	CFD		Design of F-22 fighter [20]	
CM-5/32	1993	970	CCM	449 CPU sec	Determination of structure of Eglin-C molecular system [21]	530 atoms, with 1689 distance and 87 dihedral constraints
Cray-2/1	1987	1,098	CSM	400 hours	3D modeling of projectile striking target. Hundreds of microsecond time scales. [17]	.5-1.5 million grid points256 Mword memory
Cray-2	1992	1,098	CSM	5 CPU hours	Modeling aeroelastic response of a detailed wing-body configuration using a potential flow theory [13]	
Cray-2	1992	1,098	CSM	6 CPU days	Establish transonic flutter boundary for a given set of aeroelastic parameters [13]	
Cray-2	1992	1,098	CSM	600 CPU days	Full Navier-Stokes equations [13]	
Cray-2		1,098	CSM	2 hours	3–D modeling of symmetric, transonic, low angle of attack impact of warhead and defensive structure [20]	
Cray-2		1,098	CSM	200 hours	Penetration model against advanced armor [20]	
Cray-2		1,098	CSM	2000 hours	Modeling full kinetic kill effects against hybrid armors [20]	
Cray-2		1,098	CSM	40 hours	3–D modeling of asymmetric, transonic, low angle of attack impact of warhead and defensive structure [20]	
Cray-2	1984	1,098	CFD	15 CPU m	Simulation of 2–D viscous flow field about an airfoil [3]	
Cray-2	1988	1,098	CFD	20 hr	Simulation of flow about the space shuttle (Orbiter, External Tank, Solid Rocket Boosters), Mach 1.05, Reynolds Averaged Navier-Stokes, Reynolds number = 4 million (3% model) [8]	750,000 grid points, 6 Mwords.
Cray-2	1980s	1,098	CFD	100 CPU h	Simulation of external flow about an aircraft at cruise. Steady flow. Steady Navier-Stokes simulation. [14]	1.0 million grid points.
	1995	1,400	nuclear		Credible one- and two-dimensional simulations [22]	
Cray C90/1	1993	1,437	CCM	.592 sec/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	100,000 atoms
Cray C90/1	1994	1,437	CEA	161 CPU hour	Compute magnitude of scattered wave-pattern on X24C re-entry aerospace vehicle [23]	181x59x162 grid (1.7 million)
Cray C90/1	1994	1,437	CFD	overnight	Modeling of flow over a submarine hull with no propulsion unit included [24]	1-2 million grid points.

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
Cray C916	1994	1,437	CEA		Radar cross section on perfectly conducting sphere [25]	48x48x96 ((221 thousand)
Cray C90/1	1995	1,437	CEA	1 hour	Submarine acoustic signature for single frequency	
Cray C90/1	1995	1,437	CEA	1 hour	Submarine acoustic signature for single frequency [26]	
Cray C90/1	1995	1,437	CEA	12,475 sec	Radar cross section of perfectly conducting sphere, wave number 20 [27]	97x96x192 (1.7 million grid points), 16.1 points per wavelength
Cray C90/1	1995	1,437	CEA	200 hour	Submarine acoustic signature for full spectrum of frequencies [26]	
Cray C90/1	1995	1,437	CWO		CCM2, Community Climate Model, T42 [28]	128 x 64 transform grid, 4.2 Gflops
Cray C90/1	1996	1,437	CFD	19 CPU hr	Simulation of turbulent flow around the F/A-18 aircraft at 60 degree angle of attack. Mach 0.3. Reynolds number = 8.88 million [29]	1.25 million grid points, 100 Mwords of memory.
Cray C90/1	1996	1,437	CFD	200 CPU hr	Simulation of unsteady flow about an F-18 High Alpha Research Vehicle at 30, 45, 60 deg angle of attack [30]	2.5 million grid points for half-body modeling. 40 Mwords memory
Cray C90/1	1996	1,437	CFD	3 CPU hr	Modeling of flow over a blended wing/body aircraft at cruise. [31]	45 Mwords of memory
Origin2000/4	1997	1,517	SIP	1.475 s	RT_STAP benchmark (hard) [11]	2.7 million samples per .161 sec
SGI PowerChallenge4 nodes	1997	1,686	CCM	overnight	Explosion simulation [32]	30 thousand diatomic molecules
SGI Onyx	1990	1,700	SIP		Attack and Launch Early Reporting to Theater (ALERT) [20].	
Mercury Race (52 processor, i860)	1996	1,773	SIP		Sonar system for Los Angeles Class submarines [33]	
Cray YMP/4 256 MW	1990	1,875	CSM	10 CPU s	Static analysis of aerodynamic loading on solid rocket booster [13]	10,453 elements, 9206 nodes, 54,870 DOF
Intel iPSC/860/64	1993	2,097	CCM	.418 CPUs/time step	MD simulation using short-range forces model applied to 3D configuration of liquid near solid state point [15]	100,000 atoms
Intel iPSC/860/64	1993	2,097	CCM	3.68 CPUs/ timestep	MD simulation using short-range forces model applied to 3D configuration of liquid near solid state point [15]	1 million atoms
Cray 2, 4 proc	1990	2,100	CSM	400 hours	armor/anti-armor, 3-D	
Cray T3D/16	1996	2,142	CFD	20,000 CPU sec. 50 CPU sec x 400 steps	Aerodynamics of missile at Mach 3.5. Reynolds number = 6.4 million [34]	500x150 (75,000) elements in mesh. 381,600 equations solved every timestep.
CM-2		2,471	SIP	10 minutes	Creation of synthetic aperture radar image [20]	
Cray C90/2	1993	2,750	CCM	117 CPU sec	Determination of structure of Eglin-C molecular system [21]	530 atoms, with 1689 distance and 87 dihedral constraints
Cray C90/2	1993	2,750	CCM	12269 sec	Determination of structure of E. coli trp repressor molecular system [21]	1504 atoms6014 constraints
iPSC860/128	1997	3,485	CFD	120 hr	small unmanned vehicle, fully turbulent	
iPSC/860 128 nodes	1997	3,485	CFD	5 days	Full ship flow model [35]	
iPSC860/128	1997	3,485	CFD	5 days	Small unmanned undersea vehicle. Fully turbulent model with Reynolds numbers. [35]	2.5 million grid points
Cray YMP/8	1989	3,708	CFD		Model of flow around a fully appended submarine. Steady state, non-viscous flow model. [35]	250,000 grid points
Cray Y-MP/8	early 1990s	3,708	CSM	200 hours	3D shock physics simulation [12]	6 million cells

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
Cray Y-MP/8	mid 1990s	3,708	CSM	10-40 hrs	3–D shock physics simulation [12]	100K-1 million cells
IBM SP1/64	1993	4,074	CCM	1.11 sec timestep	Molecular dynamics of SiO2 system [36]	0.53 million atoms
Intel Paragon		4,600	CEA		Non-acoustic anti-submarine warfare sensor development [20]	
IBM SP-2/32	1996	4,745	CFD	80 minutes	3–D unsteady incompressible time-averaged Navier-Stokes. Multiblock transformed coordinates [37]	3.3 million points
IBM SP-2/32	1997	4,745	CFD		Helicopter rotor motion coupled with rotor CFD, predict 3–D tip-relief flow effect, parallel approximate factorization method [38]	
Origin2000/12	1997	4,835	CFD	4329 sec	CFL3D applied to a wing-body configuration: time-dependent thin-layer Navier-Stokes equation in 3–D, finite-volume, 3 multigrid levels [39]	3.5 million points
Intel Paragon/150	early 1990s	4,864	CFD		JAST aircraft design [20]	
Cray C90/4	1993	5,375	CFD	1 week	Modeling of flow around a smooth ellipsoid submarine. Turbulent flow, fixed angle of attack, [35]	2.5 million grid points
CM-5/128	1993	5,657	CCM	436 CPU sec	Determination of structure of Eglin-C molecular system [21]	530 atoms, with 1689 distance and 87 dihedral constraints
CM-5/128	1993	5,657	CCM	6799 CPU sec	Determination of structure of E. coli trp repressor molecular system [21]	1504 atoms, with 6014 constraints
Origin2000/16	1997	5,908	SIP	.39 s	RT_STAP benchmark (hard) [11]	2.7 million samples per .161 sec
IBM SP-2/45	1997	6,300	CFD	2 to 4 hours	Helicopter blade structural optimization code, gradient-based optimization technique to measure performance changes as each design variable is varied [38]	up to 90 design variables, one run per variable
Cray T3D/64	1995	6,332	CWO		CCM2, Community Climate Model, T42 [28]	128 x 64 transform grid, 608 Mflops
IBM SP-2/64	1997	7,100	CFD	2 to 4 hours	Helicopter rotor free-wake model, high-order vortex element and wake relaxation [38]	
Paragon 256	1995	7,315	CCM	82 sec/ timestep	Particle simulation interacting through the standard pair-wise 6-12 Lennard-Jones potential [40]	50 million particles
		8,000	CEA		Bottom contour modeling of shallow water in submarine design [20]	
		8,000	SIP		Topological Synthetic Aperture Radar data processing [20]	
Paragon /321	1995	8,263	SIP		2–D FFT [41].	200 x 1024 x 1024 (200 Mpixels) images/sec
Intel Paragon/321		8,980	SIP		Development of algorithms for Shipboard Infrared search & tracking (SIRST) [20]	
SGI PowerChallenge (R8000/150)/16	1996	9,510	CFD	3.6 hr, 3000 timesteps	Large-Eddy simulation at high Reynolds number [16]	2.1 million grid points, 44 Mwords
ORNL Paragon/360		9,626	FMS		Synthetic forces experiments [42]	5713 vehicles, 6,697 entities
		10,000	SIP		Long-range unmanned aerial vehicles (UAV) on-board data processing [20]	
Cray T3D/128	1995	10,056	CEA	12,475 sec	Radar cross section of perfectly conducting sphere, wave number 20 [27]	97x96x192 (1.7 million grid points), 16.1 points per wavelength
Cray T3D/128	1995	10,056	CEA	2,874 s	Radar cross section of perfectly conducting sphere [27]	128x96x92 cells (2.4 million cells), 600 time steps

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
CM-5/256	1993	10,457	CCM	492 CPU sec	Determination of structure of Eglin-C molecular system [21]	530 atoms, with 1689 distance and 87 dihedral constraints
CM-5/256	1993	10,457	CCM	7098 CPU sec	Determination of structure of E. coli trp repressor molecular system [21	1504 atoms, with 6014 constraints
Cray C98	1994	10,625	CWO	~5 hrs	Global atmospheric forecast, Fleet Numerical operational run [43]	480 x 240 grid; 18 vertical layers
Origin2000/32	1997	11,768	SIP	.205 s	RT_STAP benchmark (hard) [11]	2.7 million samples per .161 sec
Origin2000/32	1997	11,768	CWO		SC-MICOM, global ocean forecast, two-tier communication pattern [44]	
Intel Paragon/512	1995	12,680	CCM	84 CPU s/timestep	MD simulation of 102.4 million particles using pair-wise 6-12 Lennard Jones potential [40]	102.4 million atoms
IBM SP2/128	1995	13,147	CEA	3,304.2 s	Radar cross section of perfectly conducting sphere [27]	128x96x92 cells (2.4 million cells), 600 time steps
Maui SP-2/128		13,147	FMS	2 hours	Synthetic forces experiments [42	5086 vehicles
Intel Touchstone Delta/512	1993	13,236	CCM	4.84 sec/time step	Molecular dynamics of SiO2 system [36]	4.2 million atoms
Intel Paragon	1995	13,236	SIP	55 sec	Correlation processing of 20 seconds worth of SIR-C/S-SAR data from Space Shuttle [45]	
NASA Ames SP-2/139		14,057	FMS	2 hours	Synthetic forces experiments [42]	5464 vehicles
IBM SP-2/128	1995	14,200	CWO		PCCM2, Parallel CCM2, T42 [46]	128 x 64 transform grid, 2.2 Gflops
Origin2000/40	1997	14,698	CSM		Crash code, PAM	
IBM SP-2/160	1995	15,796	CWO		AGCM, Atmospheric General Circulation Model [47]	144 x 88 grid points, 9 vertical levels, 2.2 Gflops
Cray C912	1996	15,875	CSM	23 CPU hours	Water over C4 explosive in container above wet sand, alternate scenario: container next to building, finite element [48]	38,000 elements, 230 msec simulated time,16 Mwords memory
Cray C912	1996	15,875	CSM	36 hours	water over C4 over sand	
Cray C912	1996	15,875	CSM	435 CPU hours	Water over C4 explosive in container above wet sand, building at a distance [48]	13 million cells,12.5 msec simulated time
Cray C912	1997	15,875	CWO	~ 5 hrs	Global atmospheric forecast, Fleet Numerical operational run [43]	480 x 240 grid; 24 vertical layers
Cray C912	1997	15,875	CWO	1 hr	Global ocean forecast, Fleet Numerical operational run [49]	1/4 degree, 25 km resolution
ORNL Paragon/680		16,737	FMS		Synthetic forces experiments [42]	10913 vehicles, 13,222 entities
Cray T3D/256	1993	17,503	CCM	.0509 CPUs/time step	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	100,000 atoms
Cray T3D/256	1993	17,503	CCM	.405 CPUs/time step	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	1 million atoms
Cray T3D/256	1993	17,503	CCM	1.86 CPUs/time step	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	5 million atoms
Cray T3D/256	1995	17,503	CWO		Global weather forecasting model, National Meteorological Center, T170 [50]	32 vertical levels, 190 x 380 grid points, 6.1 Gflops
Cray T3D/256	1995	17,503	CWO		AGCM, Atmospheric General Circulation Model [47]	144 x 88 grid points, 9 vertical levels, 2.5 Gflops

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
Cray T3D/256	1996	17,503	CWO	105 min	ARPS, Advanced Regional Prediction System, v 4.0, fine scale forecast [51]	96 x 96 cells, 288 x 288 km, 7 hr forecast
Cray T3D/256	1996	17,503	CFD	2 hr, 3000 timesteps	Large-Eddy simulation at high Reynolds number [16]	2.1 million grid points, 44 Mwords
Cray T3D/256	1996	17,503	CFD	52,000 CPU sec.130 CPU sec x 400 timesteps	Aerodynamics of missile at Mach 2.5, 14-degrees angle of attack for laminar and turbulent viscous effects [34]	944,366 nodes and 918,000 elements. 4,610,378 coupled nonlinear equations solved every timestep.
CM-5/512	1993	20,057	CCM	8106 CPU sec	Determination of structure of E. coli trp repressor molecular system [21]	1504 atoms, with 6014 constraints
CM-5/512	1995	20,057	CFD	15,000 CPU sec, 30 CPU sec per each of 500 timesteps.	Flare maneuver of a large ram-air parachute. Reynolds number = 10 million. Algebraic turbulence model [52,53]	469,493 nodes, 455,520 hexahedral elements. 3,666,432 equations solved per timestep.
CM-5/512	1995	20,057	CFD	500 timesteps	Parafoil with flaps flow simulation. [52]	2,363,887 equations solved at each timestep
CM-5/512	1995	20,057	CFD		Parafoil with flaps flow simulation [52].	2,363,887 equations solved at each of 500 timesteps
CM-5/512	1995	20,057	CFD		Fighter aircraft at Mach 2.0 [52]	3–D mesh of 367,867 nodes, 2,143,160 tetrahedral elements, and 1.7 million coupled nonlinear equations solved per timestep.
CM-5/512	1996	20,057	CFD	500 timesteps	Steady-state parafoil simulation, 10 deg angle of attack, Reynolds number = 10 million [54]	2.3 million equations every timestep
CM-5/512	1996	20,057	CFD	500 timesteps	Inflation simulation of large ram-air parachute, Box initially at 10 deg angle of attack and velocity at 112 ft/sec, 2 simulated seconds [55]	1,304,606 coupled nonlinear equations solved every timestep.
CM-5/512	1996	20,057	CFD	7500 CPU sec = 50 CPU sec x 150 timesteps	Aerodynamics of missile at Mach 2.5, 14 deg angle of attack for laminar and turbulent viscous effects [34]	763,323 nodes and 729,600 elements. 3,610,964 coupled nonlinear equations solved in each of 150 pseudo-time steps.
CM-5/512	1996	20,057	CFD		steady-state parafoil simulation. 10 degree angle of attack [54]	2.3 million equations x 500 timesteps; Reynolds number 10 million
CM-5/512	1996	20,057	CFD		Inflation simulation of large ram-air parachute. Box initially at 10 degree angle of attack and velocity of 112 ft/sec. 2 seconds simulated [55]	1,304,606 coupled, nonlinear equations solved per each of 500 timesteps.
CM-5/512	1996	20,057	CFD		Flare simulation of large ram-air parachute [55]	3,666,432 coupled nonlinear equations solved every timestep.
CM-5/512	1996	20,057	CFD		3–D simulation of round parachute, Reynolds number = 1 million [56]	
CM-5/512	1996	20,057	CFD		3–D study of missile aerodynamics, Mach 3.5, 14 deg angle of attack, Reynolds number = 14.8 million [57]	340,000 element mesh, nonlinear system of 1,750,000+ equations solved every timestep.
CM-5/512	1997	20,057	CFD	30 hours	Parafoil simulation [54]	1 million equations solved 500 times per run
CM-5/512		20,057	CCM	8106 sec		

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
C916	1995	21,125	CSM	900 CPU hours	Hardened structure with internal explosion, portion of the overall structure and surrounding soil. DYNA3D, nonlinear, explicit, FE code. Nonlinear constitutive models to simulate concrete & steel [58]	144,257 solid & 168,438 trust elements for concrete & steel bars,17,858 loaded surfaces,500,000 DOF,60 msec simulated time
Cray C916	1995	21,125	CWO		CCM2, Community Climate Model, T170 [28]	512 x 256 transform grid, 2.4 Gbytes memory, 53. Gflops
Cray C916	1995	21,125	CWO		IFS, Integrated Forecasting System, T213 [59]	640 grid points/latitude, 134,028 points/horizontal layer, 31 vertical layers
Cray C916	1995	21,125	CWO		ARPS, Advanced Regional prediction System, v 3.1 [60]	64 x 64 x 32, 6Gflops
Cray C916	1996	21,125	CSM	325 CPU hours3 day continuous run	Explosion engulfing a set of buildings, DYNA3D analysis to study effects on window glass & doors done off-line after the blast simulation completed [61]	825 Mwords memory
Cray C916	1996	21,125	CWO	45 min	ARPS, Advanced Regional Prediction System, v 4.0, coarse scale forecast [51]	96 x 96 cells, 864 x 864 km, 7 hr forecast
Cray C916	1996	21,125	CSM	72 hours	explosion engulfing bldgs	
Cray C916	1996	21,125	CFD		3–D simulation of flow past a tuna w/ oscillating caudal fin. Adaptive remeshing. Integrated with rigid body motio [62]	
Cray C90/16	1997	21,125	CFD	9 months	3–D simulation of submarine with unsteady separating flow, fixed angle of attack, fixed geometry [35]	
Cray C916	1998	21,125	CWO	~5 hrs	Global atmospheric forecast, Fleet Numerical operational run [43]	480 x 240 grid; 30 vertical layers
Cray C916		21,125	CSM	200 hours	2–D model of effects of nuclear blast on structure [20]	
Cray C916		21,125	CSM	600 hours	3–D model of effects of nuclear blast on structure [20]	
Cray C916		21,125	CSM	several hundred hrs	Modeling effects of complex defensive structure [20]	
Cray C916		21,125	CFD		Modeling of turbulent flow about a submarine [20]	
CEWES SP-2/229		21,506	FMS	2 hours	Synthetic forces experiments [42]	9739 vehicles
Origin2000/64	1997	23,488	CFD		ARC3D: simple 3–D transient Euler variant on a rectilinear grid [63]	
Paragon 1024	1995	24,520	CWO		PCCM2, Parallel CCM2, T42 [46]	128 x 64 transform grid, 2.2 Gflops
Intel Paragon/1024		24,520	CCM	.914 sec/ timestep		5 million atoms
Intel Paragon/1024		24,520	CCM	.961 sec/ timestep		10 million atoms
ORNL Paragon/1024		24,520	FMS	2 hours	Synthetic forces experiments [42]	16995 vehicles
Intel Paragon/1024		24,520	CCM	8.54 sec/ timestep		50 million atoms
Paragon 1024		24,520	CCM	82 sec/ timestep		200 million particles
Paragon 1024		24,520	CCM	82 sec/ timestep		400 million particles
ORNL Paragon/1024		24,520	FMS		Synthetic forces experiments [42]	16606 vehicles, 20,290 entities

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
Intel Paragon/1024	1993	24,520	CCM	.0282 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	100,000 atoms
Intel Paragon/1024	1993	24,520	CCM	.199 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	1 million atoms
Intel Paragon/1024	1993	24,520	CCM	.914 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	5 million atoms
Intel Paragon/1024	1993	24,520	CCM	.961 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	10 million atoms
Intel Paragon/1024	1993	24,520	CCM	8.54 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	50 million atoms
Intel Paragon/1024	1995	24,520	CCM	160 CPUs/ timestep	MD simulation of 400 million particles using pair-wise 6-12 Lennard Jones potential [40]	400 million atoms,
Cray T3D/400	1995	25,881	CWO		IFS, Integrated Forecasting System, T213 [59]	640 grid points/latitude, 134,028 points/horizontal layer, 31 vertical layers
Mercury Race (140 PowerPC 603e processors)	1997	27,113	SIP		Large Mercury System shipped in 1997 [64]	
Cray T3D/512	1993	32,398	CCM	.0293 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	100,000 atoms
Cray T3D/512	1993	32,398	CCM	.205 CPUs/t timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15	1 million atoms
Cray T3D/512	1993	32,398	CCM	.994 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	5 million atoms
Cray T3D/512	1993	32,398	CCM	1.85 CPUs/ timestep	MD simulation using short-range forces model applied to 3–D configuration of liquid near solid state point [15]	10 million atoms
Cray T3D/512	1995	32,398	CFD	500 timesteps	Steady-state parafoil simulation. 2 deg angle of attack. Reynolds number = 10 million [52]	38 million coupled nonlinear equations at every pseudo-timestep.
Cray T3D/512	1995	32,398	CFD		Modeling paratroopers dropping from aircraft. Moving grid. Cargo aircraft traveling at 130 Knots. High Reynolds number. Smagorinsky turbulence model [52]	880,000 tetrahedral elements for half of the domain.
Cray T3D/512	1995	32,398	CFD		Fighter aircraft at Mach 2.0. [52]	3–D mesh of 185,483 nodes and 1,071,580 tetrahedral elements
Intel Paragon	mid 1990s	44,000	CSM	<24 hours	3–D shock physics simulation [12]	6 million cells
Intel Paragon	mid 1990s	44,000	CSM	several restarts	3–D shock physics simulation [12]	20 million cells
Intel Paragon	1994	46,000	nuclear		3–D reduced physics simulation of transient dynamics of nuclear weapon [12,65]	
ASCI Red	1997	46,000	CSM	few hundred hours	3–D modeling of explosive material impact on copper plate [12]	1.29 billion cells
Origin2000 /128	1997	46,928	CEA		Radar cross section of VFY218 aircraft under 2 GHz radar wave [66]	1.9 million unknowns
Origin2000 /192	1998	70,368	CSM	400 hours	armor/anti-armor, 3–D	

Machine	*Year*	*CTP*	*Category*	*Time*	*Problem*	*Problem size*
Origin2000 /192	1998	70,368	CFD	months	3–D simulation of submarine with unsteady flow, fixed angle of attack, and moving body appendages and complex repulsors [35]	
ASCI Red/1024	1997	76,000	CSM	<25 hours	3–D shock physics simulation [12]	100 million cells
ASCI Red/1024	1997	76,000	CSM	<50 hours	3–D shock physics simulation [12]	250 million cells
ASCI Red/1024	1997	76,000	CSM	few hours	3–D shock physics simulation [12]	2-4 million cells
		80,000	SIP		Tier 2 UAV on-board data processing [20]	
Cray T3E-900/256	1997	91,035	CFD	500 timesteps	Steady-state parafoil simulation, 10 deg angle of attack, Reynolds number = 10 million [54]	2.3 million equations every timestep
Cray T3E-900/256	1997	91,035	CWO	590 hrs	Global ocean model "hindcast" [49]	1/16 degree, 7 km resolution
Cray T3E-900256 nodes	1998	91,035	CSM	450,000 node hours	Grizzly breaching vehicle plow simulation, parametric studies, different soil conditions & blade speeds [67]	2 to 4 million particles (soil, rock, mines, obstacles, etc.)
ASCI ++	??	50,000,000+	nuclear		First principles 3–D modeling [68]	

References

[1] Aloisio, G. and M. A. Bochicchio, "The Use of PVM with Workstation Clusters for Distributed SAR Data Processing," in *High Performance Computing and Networking, Milan, May 3–5, 1995. Proceedings*, Hertzberger, B. and G. Serazzi, Eds. Springer, 1995, pp. 570–581.

[2] Liu, S. K., *Numerical Simulation of Hypersonic Aerodynamics and the Computational Needs for the Design of an Aerospace Plane*, RAND Corporation, Santa Monica, CA, 1992.

[3] Ballbaus Jr., W. F., "Supercomputing in Aerodynamics," in *Frontiers of Supercomputing*, Metropolis, N. et al., eds. Berkeley, CA: University of California Press, 1986, pp. 195–216.

[4] Ewald, R. H., "An Overview of Computing at Los Alamos," in *Supercomputers in Theoretical and Experimental Science*, Devreese, J. T. and P. van Camp, eds. New York and London: Plenum Press, 1985, pp. 199–216.

[5] Smith, N. P., "PAM Crash: HPC's First Killer App Adapts to New Needs," *HPCWire*, Mar 28, 1997 (Item 10955).

[6] Graves, R., "Supercomputers: A Policy Opportunity," in *Supercomputers: a key to U.S. scientific, technological, and industrial preeminence*, Kirkland, J. R. and J. H. Poore, eds. New York: Praeger Publishers, 1987, pp. 129–140.

[7] Albrizio, R. et al., "Parallel/Pipeline Multiprocessor Architectures for SAR Data Processing," *European Trans. on Telecommunication and Related Technologies*, Vol. 2, No. 6, Nov.–Dec., 1991, pp. 635–642.

[8] Holst, T. L., "Supercomputer Applications in Computational Fluid Dynamics," in *Supercomputing 88, Volume II: Science and Applications*, Martin, J. L. and S. F. Lundstrom, eds. Los Alamitos: IEEE Computer Society Press, 1988, pp. 51–60.

[9] Binder, K., "Large-Scale Simulations in Condensed Matter Physics – The Need for a Teraflops Computer," *International Journal of Modern Physics C*, Vol. 3, No. 3, 1992, pp. 565–581.

[10] Chaput, E., C. Gacherieu, and L. Tourrette, "Experience with Parallel Computing for the Design of Transport Aircrafts at Aerospatiale," in *High-Performance Computing and Networking. International Conference and Exhibition HPCN Europe 1996. Proceedings*, Liddel, H. et al., eds. Berlin: Springer-Verlag, 1996, pp. 128–135.

[11]Games, R. A., Benchmarking for Real-Time Embedded Scalable High Performance Computing, DARPA/Rome Program Review, May 6, 1997.

[12]Camp, W. J. et al., Interviews, Sandia National Laboratories, Dec. 11, 1997.

[13]Farhat, C., "Finite Element Analysis on Concurrent Machines," in *Parallel Processing in Computational Mechanics*, Adeli, H., ed. New York: Marcel Dekker, Inc., 1992, ch. 7, pp. 183–217.

[14]Simon, H. D., W. R. Van Dalsem, and L. Dagum, "Parallel CFD: Current Status and Future Requirements," in *Parallel Computational Fluid Dynamics: Implementation and Results*, Simon, H. D., ed. Cambridge: Massachusetts Institute of Technology, 1992, pp. 1–29.

[15]Plimpton, S., "Fast Parallel Algorithms for Short-Range Molecular Dynamics," *Journal of Computational Physics*, Vol. 117, 1995, pp. 1–19.

[16]Strietzel, M., "Parallel Turbulence Simulation Based on MPI," in *High-Performance Computing and Networking. International Conference and Exhibition HPCN Europe 1996. Proceedings*, Liddel, H. et al., eds. Berlin: Springer-Verlag, 1996, pp. 283–289.

[17]Kimsey, K. and S. Schraml, Interview, Army Research Lab, Aberdeen Proving Ground, June 12, 1997.

[18]Shang, J. S. and K. C. Hill, "Performance of a Characteristic-Based, 3-D, Time-Domain Maxwell Equations Solver on the Intel Touchstone Delta," *Applied Computational Electromagnetics Society Journal*, Vol. 10, No. 1, May, 1995, pp. 52–62.

[19]U.S. Navy To Use Mercury RACE Systems for Advanced Radar, *HPCWire*, Jun 27, 1997 (Item 11471).

[20]Goodman, S., P. Wolcott, and G. Burkhart, *Executive Briefing: An Examination of High-Performance Computing Export Control Policy in the 1990s*, IEEE Computer Society Press, Los Alamitos, 1996.

[21]Pachter, R. et al., "The Design of Biomolecules Using Neural Networks and the Double Iterated Kalman Filter," in *Toward Teraflops Computing and New Grand Challenges. Proceedings of the Mardi Gras '94 Conference, Feb. 10–12, 1994*, Kalia, R. K. and P. Vashishta, eds. Commack, NY: Nova Science Publishers, Inc., 1995, pp. 123–128.

[22]The Role of Supercomputers in Modern Nuclear Weapons Design, Department of Energy (Code NN-43), Apr. 27, 1995.

[23]Shang, J. S., "Characteristic-Based Algorithms for Solving the Maxwell Equations in the Time Domain," *IEEE Antennas and Propagation Magazine*, Vol. 37, No. 3, June, 1995, pp. 15–25.

[24]Morgan, W. B. and J. Gorski, Interview, Naval Surface Warfare Center, Jan. 5, 1998.

[25]Shang, J. S., D. A. Calahan, and B. Vikstrom, "Performance of a Finite Volume CEM Code on Multicomputers," *Computing Systems in Engineering*, Vol. 6, No. 3, 1995, pp. 241–250.

[26]Hurwitz, M., Interview, Naval Surface Warfare Center, Aug. 6, 1997.

[27]Shang, J. S., "Time-Domain Electromagnetic Scattering Simulations on Multicomputers," *Journal of Computational Physics*, Vol. 128, 1996, pp. 381–390.

[28]Hack, J. J., "Computational design of the NCAR community climate model," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1545–1569.

[29]Ghaffari, F. and J. M. Luckring, Applied Computational Fluid Dynamics, http://wk122.nas.nasa.gov/NAS/TechSums/9596/Show?7126.

[30]Chaderjian, N. M. and S. M. Murman, Unsteady Flow About the F-18 Aircraft, http://wk122.nas.nasa.gov/NAS/TechSums/9596/Show?7031.

[31]Potsdam, M. A., Blended Wing/Body Aircraft, http://wk122.nas.nasa.gov/NAS/TechSums/9596/Show?7650.

[32]Rice, B. M., "Molecular Simulation of Detonation," in *Modern Methods for Multidimensional Dynamics Computations in Chemistry*, Thompson, D. L., ed.: World Scientific Publishing Co., 1998. (To appear)

[33]Mercury RACE Selected for Navy Sonar System, *HPCWire*, Jul 26, 1996 (Item 8916).

[34]Sturek, W. et al., *Parallel* Finite *Element Computation of Missile Flow Fields*, Preprint 96-012, University of Minnesota Army HPC Research Center, Minneapolis, MN, 1996.

[35]Boris, J. et al., Inverview, Naval Research Lab, Washington, DC, June 9, 1997.

[36]Nakano, A., R. K. Kalia, and P. Vashishta, "Million-Particle Simulations of Fracture in Silica Glass: Multiresolution Molecular Dynamics Approach on Parallel Architectures," in *Toward Teraflop Computing and New Grand Challenges. Proceedings of the Mardi Gras '94 Conference, Feb. 10-12, 1994*, Kalia, R. K. and P. Vashishta, eds. Commack, NY: Nova Science Publishers, Inc., 1995, pp. 111–122.

[37]Pankajakshan, R. and W. R. Briley, "Efficient parallel flow solver," in *Contributions to DoD Mission Success from High Performance Computing 1996*, 1996, p. 73.

[38]Conlon, J., "Propelling power of prediction: Boeing/NASA CRA leverages the IBM SP2 in rotor analysis," *Insights*, No. 3, Sep, 1997, pp. 2–9.

[39]Faulkner, T., "Origin2000 update: Studies show CFL3D can obtain 'reasonable' performance," *NASNews*, Vol. 2, No. 17, November-December, 1997 (http://science.nas.nasa.gov/Pubs/NASnews/97/11/).

[40]Deng, Y. et al., "Molecular Dynamics for 400 Million Particles with Short-Range Interactions," in *High Performance Computing Symposium 1995 "Grand Challenges in Computer Simulation" Proceedings of the 1995 Simulation Multiconference, Apr. 9–13, 1995, Phoenix AZ*, Tentner, A., ed.: The Society for Computer Simulation, 1995, pp. 95–100.

[41]Rowell, J., "Rome Labs Deploys Paragon, 32 HIPPIs for COTS Radar Processing," *HPCWire*, Dec 4, 1995 (Item 7556).

[42]Brunett, S. and T. Gottschalk, Large-scale metacomputing gamework for ModSAF, A real-time entity simulation, Center for Advanced Computing Research, California Institute of Technology (Draft copy distributed at SC97, November, 1997).

[43]Rosmond, T., Interview, Naval Research Laboratory, Monterey, July 28, 1997.

[44]Sawdey, A. C., M. T. O'Keefe, and W. B. Jones, "A general programming model for developing scalable ocean circulation applications," in *ECMWF Workshop on the Use of Parallel Processors in Meteorology, Jan. 6, 1997*.

[45]Phung, T. et al., Parallel Processing of Spaceborne Imaging Radar Data, Technical Report CACR-108, California Institute of Technology, August, 1995.

[46]Drake, J., "Design and performance of a scalable parallel community climate model," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1571–1591.

[47]Wehner, M. F., "Performance of a distributed memory finite difference atmospheric general circulation model," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1655–1675.

[48]Balsara, J. and R. Namburu, Interview, CEWES, Aug. 21, 1997.

[49]Wallcraft, A., Interview, NRL, Stennis Space Center, Aug. 20, 1997.

[50]Sela, J. G., "Weather forecasting on parallel architectures," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1639–1654.

[51]Jahn, D. E. and K. K. Droegemeier, "Proof of Concept: Operational testing of storm-scale numerical weather prediction," *CAPS New Funnel*, Fall, 1996, pp. 1-3 (The University of Oklahoma).

[52]Tezduyar, T. et al., "Flow simulation and high performance computing," *Computational Mechanics*, Vol. 18, 1996, pp. 397–412.

[53]Tezduyar, T., V. Kalro, and W. Garrard, *Parallel Computational Methods for 3D Simulation of a Parafoil with Prescribed Shape Changes*, Preprint 96-082, Army HPC Research Center, Minneapolis, 1996.

[54]Muzio, P. and T. E. Tezduyar, Interview, Army HPC Research Center, Aug. 4, 1997.

[55]Stein, K. et al., *Parallel Finitel Element Computations on the Behavior of Large Ram-Air Parachutes*, Preprint 96-014, University of Minnesota Army HPC Research Center, Minneapolis, MN, 1996.

[56]Stein, K., A. A. Johnson, and T. Tezduyar, "Three-dimensional simulation of round parachutes," in *Contributions to DoD Mission Success from High Performance Computing 1996*, 1996, p. 41.

[57]Army High Performance Computing Research Center, Army HPC Research Center, Minneapolis, MN, 1996.

[58]Papados, P. P. and J. T. Baylot, "Structural Analysis of Hardened Structures," in *Highlights in Computational Science and Engineering*. Vicksburg: U.S. Army Engineer Waterways Experiment Station, 1996, pp. 64–65.

[59]Barros, S. R. M., "The IFS model: A parallel production weather code," *Parallel Computing*, Vol. 21, No. 10, 1995, pp. 1621–1638.

[60]Droegemeier, K. K. et al., "Weather Prediction: A Scalable Storm-Scale Model," in *High performance computing: problem solving with parallel and vector architectures*, Sabot, G. W., ed. Reading, MA: Addison-Wesley Publishing Company, 1995, pp. 45–92.

[61]King, P. et al., "Airblast Effects on Concrete Buildings," in *Highlights in Computational Science and Engineering*. Vicksburg: U.S. Army Engineer Waterways Experiment Station, 1996, pp. 34–35.

[62]Ramamurti, R. and W. C. Sandberg, "Simulation of flow past a swimming tuna," in *Contributions to DoD Mission Success from High Performance Computing 1996*, 1996, p. 52.

[63]Taft, J., "Initial SGI Origin2000 Tests Show Promise for CFD Codes," *NASNews*, Vol. 2, No. 25, July-August, 1997 (http://www-sci.nas.nasa.gov/Pubs/NASnews/97/07/article01.html).

[64]Mercury Delivers 38 GFLOPS With 200-MHz PowerPC Processors, *HPCWire*, Jan 31, 1997 (Item 10694).

[65]Nielsen, D., Private Communication, Jan. 15, 1998.

[66]NCSA's Orign2000 Solves Large Aircraft Scatterer Problems, *HPCWire*, Jun 13, 1997 (Item 11365).

[67]Horner, D. A., Interview, CEWES, Aug. 22, 1997.

[68]Accelerated Strategic Computing Initiative: PathForward Project Description, December 27, 1996, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, Sandia National Laboratory, 1996.

Appendix B: People and Organizations

The authors are grateful to the following people for providing information and commentary and/or reviewing draft versions of the report. Their participation does not necessarily imply endorsement of this study or its findings.

Industry:	
CenterPoint Ventures	Steven J. Wallach
Digital Equipment Corporation	Donald R. Ames, Michael Dennis, Keith Melchers, Robert Rarog
Hewlett-Packard Corp.	Constantine Anifantis, Rajiv Gupta
Hewlett-Packard/Convex Division	Lana Yeager
IBM	Henry R. Brandt, Timothy M. DiVincenzo, Ray DuPont, Jim Jardine, Mary J. Toups
Intel Corporation	Roger A. Golliver, David W. Rose
Motorola	Gregory B. Smith
NCR Corporation	Gerald Matthews
Nichols Research	Fletcher Kurtz, James A. Newhouse
SGI/Cray Research Division	William N. Bartolone, Alan Benfell, David D. Blaskovich, Vito Bonglorno, Jr., Jeff Brooks, Paul Ciernia, Lynne Mullane, Kathy Nottingham, Carol Woronow

Silicon Graphics, Inc.	Kenneth P. Jacobsen, John R. Mashey, Ed Reidenbach, David W. Rolston, Ann Scott, Timothy J. Sherbak, Ross A. Towle
Sun Microsystems, Inc.	Hans Luemers, Sharon Maule, Lisa Quock, William C. van Loo, Patrick Wang
Academia:	
Arctic Region Supercomputing Center	Frank Williams
California Institute of Technology	Dan Davis
Center for Analysis and Prediction of Storms and School of Meteorology, University of Oklahoma	Kelvin K. Droegemeier
Mississippi State University	Joe F. Thompson, J. Don Trolter
Ohio Supercomputer Center	Dana E. Hoffman
University of Colorado at Boulder	Bernard Udis
Department of Commerce:	
Bureau of Export Administration	Ian Baird, Amanda DeBusk, Sue E. Eckert, Tanya Hodge Mottley, James A. Lewis, Roger Majak, John E. McPhee, William Reinsch
Department of Defense:	
DoD High Performance Computing Modernization Office	F. Brett Berlin, Larry Davis, Roger S. Foster, Jeff Highland, Kay Howell
Defense Advanced Research Projects Agency	Howard Frank
Defense Technology Security Administration	Steve Boyce, Paul Koenig, Oksana Nesterczuk, Patrick T. Smith, David Tarbell
The MITRE Corp.	Richard A. Games
National Security Agency	George R. Cotter, Joe Davis, Tom Glenon, Norman S. Glick, A. Ray Miller, Erica Science

Office of the Secretary of Defense	Mitchel B. Wallerstein
Department of the Army:	
Army HPC Research Center	Paul C. Muzio, Tayfun E. Tezduyar
Army Research Laboratory–Aberdeen Proving Ground	John L. Cole, Carol Ellis, Chuck Kennedy, Kent D. Kimsey, Dan Pressel, Tony Pressley, Bob Reschly, Betsy Rice, Stephen Schraml
Edgewood Research, Development and Engineering Center	Miles C. Miller
U.S. Army Corps of Engineers Waterways Experiment Station	Jimmy Balsara, Jeff Holland, David Horner, James Houston, Robert Jensen, Louis H. Turcotte
Department of the Navy:	
Fleet Numerical METOC Center	Leo C. Clarke, William (Kim) Curry, Jeanne Frew, Randy Nottenkamper, Robert J. Plante
Naval Oceanographic Office	Alan Wallcraft
Naval Oceanographic Office DOD MSRC	Terry Blanchard
Naval Postgraduate School	Russell L. Elsberry, Art Schoenstadt, Albert J. Semtner, Jr.
Naval Research Laboratory	Jay P. Boris, Mark Emery, Ravi Ramamurti, William C. Sandberg
Naval Research Laboratory–Monterey	Tom Rosemond
Naval Research Laboratory–SSC	Margo Frommeyer
Naval Sea Systems Command	Patrick Fell
Naval Surface Warfare Center Carderock Division	Gordon Everstine, Fred Fisch, Joseph Gorski, Myles M. Hurwitz, William B. Morgan

Naval Surface Warfare Center Dahlgren Division	Robert Bevan, Wayne Chepren, John R Cogar, Bruce Copeland, Patrick Fell, Robert D. Harrison, Jr., Frank Maillie, David Millner, Ted Shelkey, Joe Veno, Charles H. Washington
Naval Undersea Warfare Center	Stephen A. Schneller
Space and Naval Warfare Systems Center	Kevin Boner, Robert Dukelow, David Fusco, Ron Hidinger, Steven Murray, Lynn Parnell, Robert Pritchard, George Seymour
Department of the Air Force: Air Force Institute of Technology	Gary B. Lamont
Wright Patterson AFB	John Blair, Jeff E. Graham, Ruth Pachter, Joseph Shang
Department of Energy:	Trisha Dedik, Normann H. Kreisman, Robin Staffin, Gil Weigand, Anatoli Welihozkiy
Lawrence Berkeley National Laboratory	William Kramer, Horst Simon
Lawrence Livermore National Laboratory	Paul S. Brown, Gerald L. Goudreau, Carol G. Hoover, Charles F. McMillan, Dale Nielsen, Jr., David Nowak, Stan Trost
Los Alamos National Laboratory	Thomas Adams, Adolfy Hoisie, Richard A. Krajcik, Olaf M. Lubeck, Arvid S. Lundy, Alex L. Marusak, William H. Reid, John Vandenneboom, Harvey Wasserman
Sandia National Laboratories	William J. Camp, Eugene Hertel, James S. Peery, Paul Yarrington
Department of State:	Robert Garel, Oliver John
Arms Control and Disarmament Agency	Mark Goodman, Joe Smaldone, Bernard Udis, Christian Westermann

Other U.S. Government:	
Central Intelligence Agency	Luisa M. Colón, Thomas Hall, Robert Sorensen
NASA Ames Research Center	Dennis Jesperson, Nateri Madaran, Subhash Saini
National Center for Environmental Protection	Misha Rancic
National Coordination Office for Computing, Information and Communications	John C. Toole
National HPCC Council	John Miguel
National Security Council	Gary Samore, Maureen Tucker

Appendix C: Glossary

ADVISR	Advanced Virtual Intelligence Surveillance Reconnaissance
AGCM	Atmospheric General Circulation Model
AHPCRC	Army High Performance Computing Research Center
ALE	Arbitrary Lagrangian Euler
ARPS	Advanced Regional Prediction System
ASC	Aeronautical Systems Center
ASCI	Accelerated Strategic Computing Initiative
CBDCOM	Chemical Biological Defense COMmand
CCM	Computational Chemistry and Materials Science
CCM2	Community Climate Model
CEA	Computational Electromagnetics and Acoustics
CEN	Computational Electronics and Nanoelectronics
CFD	Computational Fluid Dynamics
CGF	Computer Generated Force
CMOS	Complementary Metal Oxide Semiconductor
CRA	Cooperative Research Agreement
CSM	Computational Structural Mechanics
CTA	Computational Technology Area
CTI	Coherent Toroidal Interconnect
CTP	Composite Theoretical Performance
CWO	Climate/Weather/Ocean Modeling and Simulation
DARPA	Defense Advanced Research Projects Agency
DEC	Digital Equipment Corporation
DEM	Discrete Element Model
DIS	Distributed Interactive System

DNS	Direct Numerical Simulation
DoD	Department of Defense
DOE	Department of Energy
DRAM	Dynamic Random Access Memory
DSM	Distributed Shared Memory
DSP	Digital Signal Processing
ECL	Emitter-Coupled Logic
ECMWF	European Centre for Medium-range Weather Forecasts
EFS	Ensemble Forecast System
EQM	Environmental Quality Modeling and Simulating
EW	Electronic Warfare
FD	Finite Difference
FE	Finite Element
FFT	Fast Fourier Transform
FMS	Forces Modeling and Simulation/C4I
FNMOC	Fleet Numerical Meteorology and Oceanography Center
FV	Finite Volume
HLA	High Level Architecture
HPC	High-Performance Computing
HPCMP	High Performance Computing Modernization Program
IFS	Integrated Forecasting System
IMT	Integrated Modeling and Test Environments
IVL	Individually Validated License
JAST	Joint Advanced Strike Technology
LES	Large Eddy Simulation
LOTS	Logistics Over The Shore
MCARM	Multi-Channel Airborne Radar Measurement
MM5	Mesoscale Model
ModSAF	Modular Semi-Automated Forces
MPI	Message Passing Interface
MPP	Massively Parallel Platform
Mtops	Millions of Theoretical Operations Per Second
MVOI	Multi-Variate Optimum Interpolation
NAVO	Naval Oceanographic Office
NCAR	National Center for Atmospheric Research
NOGAPS	Navy Operational Global Atmospheric Prediction System
NORAPS	Navy Operational Regional Atmospheric Prediction System
NRL	Naval Research Laboratory
N-S	Navier-Stokes

NUMA	Non-Uniform Memory Access
OEM	Original Equipment Manufacturer
PCA	PowerChallenge Array
PCCM2	Parallel Community Climate Model
PVM	Parallel Virtual Machines
PVP	Pipelined Vector Processor
RASSP	Rapid Prototyping of Application Specific Signal Processor
RISC	Reduced Instruction Set Computing
SAEF	Simple Average Ensemble Forecast
SAR	Synthetic Aperture Radar
SCI	Scalable Coherent Interconnect
SF Express	Synthetic Forces Express
SHV	Standard High Volume
SGI	Silicon Graphics, Inc.
SIA	Semiconductor Industry Association
Simnet	Simulator Network
SIP	Signal/Image Processing
SLT	Semi-Lagrangian Transport
SMP	Symmetrical MultiProcessor
SPP	Scalable Parallel Processor
SRAM	Static Random Access Memory
SSC	Space and Naval Warfare Systems Command
STAP	Space-Time Adaptive Processing
STOW-E	Synthetic Theater of War – Europe
SWAPS	Spectral Wave Prediction System
UAV	Unmanned Aerial Vehicle
UV	UltraViolet
V-IRAM	Vector-Intelligent Random Access Memory