Chapter 4

Applications Working Group

Chair:	
Geoffrey C. Fox	Syracuse University
	gcf@npac.syr.edu
Associate Chair:	
Rick Stevens	Argonne National Laboratory
	stevens@mcs.anl.gov

Working Group:	
Tony Chan	University of California at Los Angeles
Dwight Duston	BMDO
Walter Ermler	U. S. Department of Energy
Jim Fischer	NASA Goddard
Bruce Fryxell	NASA Goddard
Ed Giorgio	U. S. Department of Defense
Jim Glimm	SUNY, Stony Brook
Jacob Maizel	National Institutes of Health
Rob Schrieber	NASA RIACS
Paul Stolorz	Jet Propulsion Laboratory
Francis Sullivan	Supercomputing Research Center
Richard Zippel	Cornell University

4.1 Introduction

The Applications Working Group played two major roles in the workshop. First, the needs of important applications are the motivation for designing and building Petaflops machines. This is discussed in Section 4.2 on general terms. Second, the characteristics of potential Petaflops scale applications can be used to guide the other three workshop activities; devices, architectures, and software for Petaflops computers. Our general findings in this area can be found in Section 4.3.

In Section 4.4, we approach the issues of Sections 4.2 and 4.3 from the point of view of particular applications. Section 4.5 describes algorithmic issues.

4.2 Applications Motivation of a Petaflops Computer Program

We show in Table 4.1 a wide set of applications, which are potential uses of Petaflops machines. We divide these into three major areas:

1. Large Scale Simulations (grand challenges) extrapolated from Teraflops machines. Two sub-classes can be separated.
   - Problem size naturally increases (an example is turbulence where more grid points are needed to increase spatial resolution),
   - Problem size is unchanged but there is a need to increase simulated time (an example is protein dynamics with 10,000 atoms and one needs to achieve millisecond simulated time with $\sim 10^{-14}$ second basic time step).
2. Data Intensive Applications that rely on Petabyte $\rightarrow$ exabyte of primary and secondary storage.
3. Novel Applications.

There is no doubt that these can be used to build a strong case for Petaflops machines. As we discuss in the examples of Section 4.4, many applications require Petaflops, or in some cases higher, performance for realistic results. The need for this performance level follows directly from the physical structure of the problem in some cases, and from the size of the base dataset in others. The following observations qualify and expand these remarks.

1. Our working group did not have the broad expertise to establish the Petaflops motivation in full detail.

2. We can give examples, as described in Section 4.4. However, we recommend that our work be refined by appropriate "domain expert groups."

3. We can note generally that computation is and will be increasingly important in economy, society, education, academic, and U.S. needs to be in the lead in the continuing future—just as it is now with HPCC.

4. One cannot predict the critical applications 10–20 years from now. New national problems will arise, and surely HPCC will be critical in many of them. Most of our exemplars will be important, if not the most important Petaflops scale problems.

5. Many Petaflops scale applications will involve integration of disparate activities and will require changes in current modus operandi. For instance, the NII (and applications such as interactive television) will impact society in a nontrivial way. Agile manufacturing requires database (CAD) simulation, design, analysis, manufacturing, and marketing to be integrated. Petaflops computing enables this, but the multidisciplinary character has implications for hardware, software, and hardest of all, the structure of manufacturing companies.

6. We recommend a program to investigate new algorithms needed by Petaflops scale applications and the special architectural features of Petaflops machines. This is expanded in Section 4.5.

7. Historically new algorithms, new difficulties, and indeed new applications have been identified as one increases power of computer—even by a "mere" factor of 10. This is likely to occur in "all" application areas. Today's typical achieved maximum performance is a Gigaflops. Thus, we expect a set of revolutions or "just" minirevolutions as we extrapolate a factor of 10^6 to Petaflops performance.

Table 4.1: Petaflops Application Areas

A. Biology, Biochemistry, and Biomedicine

- A1 Design better drugs
- A2 Understand the structure of biological molecules (protein folding)
- A3 Genome informatics and phylogeny
- A4 Process data from medical instruments
- A5 Simulate functions of human body
  - Blood flow through heart
- A6 Neural networks in cortex
- A7 Real time three-dimensional biosensor data fusion (the virtual human)
- A8 Analysis of integrated medical database to improve quality and cost of health care

B. Chemistry, Chemical Engineering

- B1 Design and understand nature of catalysts and enzymes
- B2 Simulate new chemical plants and distribution (pipeline) systems

C. Physics

- C1 Understand the nature of new materials
- C2 Simulate semiconductors used in chips
- C3 Design fusion energy system (Numerical Tokamak)
- C4 Simulate nuclear explosions
- C5 Matter transporter (three-dimensional fax and edit)
- C6 Understand properties of fundamental particles (QCD)

D. Space Science and Astronomy

- D1 Evolve the structure of the early universe into the epoch of the current observable world (cosmology)
- D2 Understand how galaxies are formed
- D3 Understand large scale astrophysical systems (stars, gas clouds, globular clusters)
- D4 Understand dynamics of Sun
- D5 Understand collision of black holes and emission of gravitational waves
- D6 Analyze new optical and radio astronomy data to combine data from many telescopes and minimize impact of Earth's atmosphere

E. Artificial Intelligence

- E1 New neural network learning and optimization algorithms
- E2 High-level searches of full text databases
- E3 Decipher new military coding methods (cryptography)
- E4 Deep search of game trees for social models and games such as computer chess

F. Study of Climate and Weather

- F1 Forecast weather and predict global climate
- F2 Forecast severe storms (tornadoes, hurricanes)
- F3 Study coupling of atmosphere, ocean, Earth use with economic and political decisions
- F4 Integrate models and weather data for optimal interpolation (data assimilation)

G. Environmental Studies

- G1 Model flow of pollutants and ground water in the Earth (flow in porous media)
- G2 Model air and water quality and relation to policy
- G3 Model ecological systems
- G4 Analyze data from planet Earth to understand nature and use of land (Earth Observing System)

H. Geophysics and Petroleum Engineering

- H1 Analyze three-dimensional seismic data to obtain better well placement
- H2 Model oil reservoir to optimize effectiveness of secondary and tertiary oil extractions
- H3 Analyze models of and data from earthquakes to improve predictions of how and when quakes will occur

I. Aerospace, Mechanical, and Manufacturing Engineering

- I1 Build more energy efficient cars, airplanes, and other complex artifacts using computational fluid dynamics, structural analysis, and multidisciplinary optimization for a graph of expected performance and memory needs in the aircraft industry (see Figure 4.1)
- I2 Design new propulsion systems
- I3 Simulate new combustion materials
- I4 Simulate radar signature of new vehicles (stealth aircraft)
- I5 Simulate chips used in new computers
- I6 Simulate electromagnetic properties of high-frequency circuits
- I7 Simulate manufacturing processes
- I8 Optimal scheduling of manufacturing systems

J. Military Applications

- J1 Simulate new military sensor and communication systems
- J2 Control military operations with data fusion and spatial reasoning
- J3 Integrate human, online military systems, and computer simulations in exercises (SIMNET)

K. Business Operations

- K1 Simulations and complex database analysis for advanced decision support in business and politics
- K2 Support integrated agile manufacturing system
- K3 Dynamic scheduling of air and surface traffic when disrupted by weather and crises
- K4 Control and image analysis for advanced robots
- K5 Economic modeling on Wall Street
- K6 Graphics for digital movies
- K7 Linkage analysis to find correlated records in a database indicating anomalies and fraud in health care, securities, credit card operations, and similar areas
- K8 Analysis of customer data to optimize marketing (market segmentation)

L. Society

- L1 Large-scale simulations and database searches for education
- L2 Electronic shopping and other interactive television
- L3 Support world-wide digital library and information systems (text, images, video)
- L4 Integrate and update intelligent agents on the NII (Knowbot garage)
- L5 Image analysis for large databases to find just the right picture (missing persons, cover of magazine)
- L6 Support of up to a million simultaneous players in large virtual environment linked to advanced home video games

4.3 Issues/Characteristics for Architecture (Hardware) and Software of Petaflops Machines

There were particularly fruitful interactions between architecture and application groups. We can make the following general remarks on characteristic features of Petaflops scale applications.

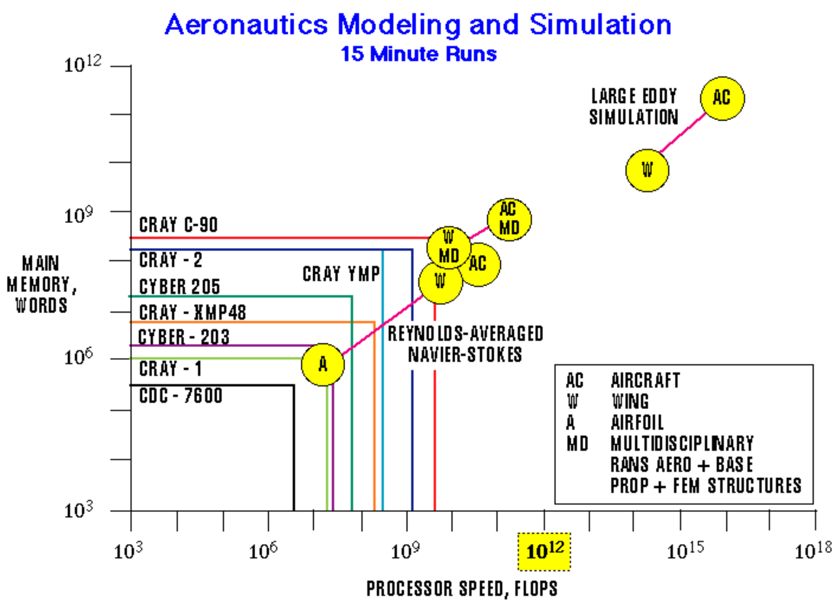

Figure 4.1: Aeronautics Modeling and Simulation

1. We need to establish a common "language" (set of terms) to discuss memory hierarchy/parallelism and communication in a hardware/software /algorithmic implementation neutral fashion. We recommend a near-term activity to refine initial steps begun here to define applications for architecture and software communities. This implementation neutral description of applications and architectures should also help discussions between software and architecture communities. The need for this agreed terminology was highlighted by our discussions with the architecture group where latter noted that application scientists described the computational structure of their problem inappropriately—using, for instance, the language of MIMD distributed memory machines when issues were more general and reflected memory hierarchy. As the target Petaflops machine could have a mix of these architectures, a distributed set of hierarchical memory nodes, translating application scientist specifications into Petaflops designs led to vigorous confused debate.

2. Our discussions with architecture group isolated several classes of machines—three based on memory hierarchy processor trade-offs and others based on memory size and I/O requirements.

3. We made a list of architectural features, which are shown in Tables 4.2, 4.3, and 4.4. These were used as a guide in preparing exemplar application discussions in Section 4.4.

4. We recommend that once a better framework is agreed (see item 1.), that "domain experts" be asked to refine our study (as begun in Section 4.4) in a broader range of potential Petaflops scale applications. Potential application domains are already given in Table 4.1.

5. We identified a general rule for time stepped algorithms (Section 4.4.1).

   $$\text{Memory} = \left[\frac{\#\ \text{flops}}{1\ \text{Gigaflops}}\right]^n\ \text{Gigabyte}^*$$

   *Put here memory needed at a Gigaflops performance.

   $n = 3/4$ for fixed total simulation time, but $n < 3/4$ if needed (as often one does) to increase total simulated time. This rule predicts a memory size of 30 Terabytes is appropriate for a Petaflops machine if 1 Gigabyte is appropriate for a Gigaflops machine. This estimate is consistent with NASA's aerospace predictions in Figure 4.1.

Table 4.2: Some General Application Characteristics

1.	Is Petaflops performance needed by this application, and if so, why?
2.	Are new algorithms needed?
3.	What are size characteristics of problem? How does size scale as we increase performance of computer?
4.	Does this problem have special precision needs?
5.	What is nature of computation? Does it involve flops or some other sort of OPS?
6.	What are I/O requirements of problem in terms of bandwidth and (secondary) storage size?

6. Current "rule" memory (bytes) = performance (flops) is modified because up to "now," solving problems has been constrained by machine size and so one scales problem "blindly." On the other hand, the Petaflops machines will solve real problems constrained by the "physics" of the situation.

7. Some interesting characteristics of a Petaflops machine are

   - Petaflops machines will do in five minutes what it takes Gigaflops machines 10 years to do
   - 10^{13} bytes = 8 (bytes/word) $\times$ 10 components/grid points $\times\, 5000^3$ (grid size)
   - 10^{15} bytes = 2300 years video
$= 10^9$ books
$= 3 \times 10^8$ Megapixel images
$= 3 \times 10^{10}$ compressed images

8. Some Petaflops applications are somewhat less demanding on hardware characteristics than today's problems (e.g., they are applications with a lot of compute needs and this implies lower internode communication bandwidth to node compute power ratio).

Table 4.3: Some Architectural Characteristics of Applications

1.	What is memory required for a Petaflops performance?
2.	What are secondary and tertiary storage needs?
3.	Can this application use a metacomputer (networked computers)?
4.	Can this application use unconventional architectures (e.g., neural networks, content addressable memory, associative processor)?
5.	What is the expected realized versus peak performance for this application?
6.	Can this application use a SIMD architecture or a MIMD collection of SIMD "nodes"?
7.	Is this application sensitive to latency?
8.	What degree of local parallelism is present in this application? This is in addition to "overall" data parallelism and can be exploited in shared memory multiprocessor nodes.
9.	How many nodes does "overall" data parallelism support? Note: Burton Smith points out that product P of performance (Petaflops) and best possible memory access time (nanosecond) is lower bound to overall concurrency needed in application. P is at least $10^6 (= 10^{15} \times 10^{-9})$. Characteristics 8 and 9 break up this minimal P for a given problem into amount that can be supported by coarse-grain data parallelism (characteristic 8) and that (characteristic 9), which can be exploited in a finer grain (e.g., shared memory) architecture.
10.	Discussions with architecture group developed three strawman architectures. • Architecture I: Around 200 Teraflops nodes—shared memory architecture • Architecture II: Around 10,000 0.1 Teraflops nodes—switch or similar general interconnect • Architecture III: Around 10^6 1 Gigaflops nodes—mesh interconnect architecture Can applications use these architectures? The answer to this question is related to those of previous characteristics.

Table 4.4: Software Technology Issues for Each Application

1.	What operating system support does the application need?
2.	What compiler and tool support does the application need?
3.	Does the application naturally fit particular programming paradigms?
4.	Are there special user interface issues?

9. It is interesting to consider real-time applications so that Petaflops performance is required to keep up with the machine and people in loop (e.g., defense simulation and control).

10. Many real-world simulations need Petaflops because problems have multiple length scales.

11. All members of the applications working group felt that Petaflops central supercomputers should be accompanied by the natural scaling Teraflops level workstations distributed among the users.

4.4 Exemplar Applications

4.4.1 Porous Media

From today's machines to the Petaflops computer, there is a factor of 10^4 in speed. How will this produce value in problems of major importance to society?

Most important problems are already solved at some level, but most solutions are insufficient and need improvement in various respects:

- under resolution of solution details, averaging of local variations and under representation of physical details
- rapid solutions to allow efficient exploration of system parameters

- robust and automated solution, to allow integration of results in high-level decision, design, and control functions
- inverse problems (history match) to reconstruct missing data require multiple solutions of the direct problem.

For PDE-based problems, the computational effort scales inversely as grid size to the fourth power, h^4, and often, especially for implicit problems, higher powers, such as h^7 can occur.

For field scale oil reservoir simulation, grids on the order of 100 meter spacing might be common. Geological variation occurs on all length scales, down to the pore size of the rock, about a micron. Not all of this variation needs to be simulated, fortunately. The interwell separation is perhaps 400 meters, and flow between wells is the important variable to be predicted. Variation on the range of 10 to 20 meters is not well represented by averaging methods, and is better computed, so that there is a utility in refining grids by a factor of 5 to 10.

On the basis of these considerations, we propose the following simulation, for which the Petaflops machine would be necessary:

1. $10^3 \times 10^3 \times 10^2 = 10^8$ grid elements
2. 30 species
3. 10^4 time steps
4. 3×10^9 words of memory per case
5. 300 cases considered (geostatistical parameters; economic or operating parameters; history matching iterative solutions of the direct problem)
6. 10^{12} words of memory (all cases considered in parallel)
7. 3×10^{14} grid $\times$ time cells total
8. 10^{19} flops
9. $10^4\,\mathrm{sec} = 3$ hrs. Petaflops computational time

At these length scales, geological data is not known, except in a statistical sense, and so statistical ensemble averages will provide average performance as well as a measure of variability associated with these averages and the possibility or probability of outlier solutions, such as early breakthroughs.

Similar issues apply to ground water remediation sites. Here the sites and well spacings are typically smaller, but the same scaling of grid to well spacing arguments apply. Commonly narrow conduction bands, or isolated time events, such as runoff during storms dominate total migration of contaminants so that accurate resolution in space and time is needed for reliable predictive capability.

Complicated chemistry, included binding of contaminants to absorption sites, or the trapping of contaminant bearing water in semi-isolated micropores gives rise to the disturbing phenomena of sites which appear to be remediated by a pump and treat method, only to have the contaminant reemerge when treatment is terminated. For this reason, physical processes, and system variables often need an increased accuracy of description, as well as finer grid resolution.

What are the architectural issues which result from this problem?

For memory, we see that memory size is determined almost entirely by the application, and is nearly independent of system architecture. The Petaflops machine is mainly justified to solve large problems, rather than to solve problems of a fixed size more rapidly. For the easiest problems, we have

$$\text{memory} \sim \text{speed}^{3/4}$$

but for many cases, and especially the more computationally difficult ones, the exponent will be smaller, because:

- some of the extra computational power will be devoted to solution for longer total time, or exploration of more parameter values
- for implicit problems, or nonlocal force laws, the computational work grows more rapidly than h^4.

These scaling laws should be developed with known proportionality coefficients coming from today's machines, which appear to be well balanced for a broad mix of problems.

There is a similar scaling law for communication latencies in memory hierarchies. Communication of n bytes takes $an + b$ units of time, where a and b are measured dimensionlessly in units of floating point operations. Here b is latency and a is bandwidth.

The number of floating point operations which can be usefully performed between communication steps is proportional to the local memory size. Consider a two-level hierarchy, with M bytes stored at m locations. After

$O(mM)$ floating point operations, there will be a need to communicate $O(m)$ domain decomposition boundary information messages of size $O(M^{2/3})$ in the most favorable case, and $O(m^2)$ messages of size $O(M)$ in the worst case. For the more common favorable case, the communication cost is

$$m(aM^{2/3} + b)$$

and the computational cost is

$$mM$$

so we need

$$mb + maM^{2/3} << mM$$

or

$$b << M$$

and

$$a << M^{1/3}$$

Constants in these relations can be determined from current balanced machines. The same arguments can be extended to multilevel memory in a hierarchy.

In the unfavorable cases, the problem has not been parallelized successfully at a conceptual level, and the machine will fail for these problems in the absence of further algorithmic work. For example, dense matrices may achieve a more nearly sparse character if a better basis is used, such as a multipole or wavelet basis. Such algorithmic methods may move a problem from an unfavorable to a favorable case.

For porous media problems, the pressure is solved implicitly. This problem has a bad condition number as the mesh is refined. Thus, there is a need for preconditioners (approximate solution methods) of a hierarchical nature, to put these problems in the favorable scaling cases. Clearly research issues of a numerical analysis nature will arise.

4.4.2 Computational Astrophysics

Need for a Petaflops Computer

There is a wide variety of interesting and challenging problems to be solved in this field which require a correspondingly wide variety of computational techniques and machine capabilities. A partial list of possible applications includes multidimensional stellar evolution—from star formation to the death of the star in a supernova explosion, cosmological simulations, galaxy formation, accretion disk structure and evolution, and the formation of supersonic jets. There are several reasons why these problems will require (at least) a Petaflops computer in order to obtain reliable solutions. One is the need to simulate enormous ranges in length and time scales, requiring very large computational grids (possibly using adaptive mesh techniques) and huge numbers of time steps. Very large grids also are required to reduce the numerical dissipation in order to represent accurately the extremely low viscosity of most astrophysical gases. This is particularly important in regimes where turbulence and convective motions are important. Most problems will require fully three-dimensional simulations. Calculations containing 10^9 grid points or more will have to be done routinely. Many problems will require complex physics calculations at each grid point, such as nuclear reaction networks involving perhaps hundreds of species and thousands of reactions, detailed equation of state calculations, complex radiative transfer, and solution of elliptic equations to calculate, for example, self gravity. Calculations of galaxy formation will require coupling of a gas dynamics code to an N-body code to follow the motions of individual stars. Finally, it will be necessary to run each simulation many times to fully understand the physics of the object(s) being studied. These additional simulations will involve parameter studies, modifying the initial conditions, using different input physics, and so on.

Architectural Issues

The applications in this field can be divided into two major classes, those that require global communications and those that need only local communications. Applications requiring only local communications involve explicit gas dynamics on regular grids with only zero-dimensional microphysics, such as reaction networks and equation of state calculations at each grid point. Applications which use global communication include implicit gas dynamics methods, adaptive grids, radiative transfer, self-gravity calculations, and N-body tree codes. These two set of applications have somewhat different machine requirements, which will be addressed separately below.

For applications which use only local communications, virtually any number of processors would be used. Certainly 10^6 processors would present no significant difficulty and perhaps even 10^9 processors could be used effectively on some applications. Only nearest neighbor communications would be involved and a sufficiently large number of operations can be done at each grid point between communications steps that issues of latency and communication bandwidth become less critical. Applications of this type should run efficiently on any balanced architecture and obtain a significant fraction of peak speed.

For applications requiring global communications, computers with a smaller number of faster processors will be easier to use. These applications also will require faster communications with lower latency to run efficiently. It seems unlikely that these applications will achieve a large fraction of peak speed using conventional hardware and software approaches. There is a high probability that applications in this class will become increasingly important in the future as algorithms for multidimensional implicit gas dynamics, multidimensional radiative transfer, and adaptive grids become more popular.

I/O and Memory Requirements

Consider a reasonable problem size using a 1000^3 grid. Assuming a few hundred variables per grid point, the calculation would require 1–10 Teraflops of memory. Typically, one would like to store the values of each variable at least 1,000 times during the calculation, which would require approximately 10 Petabytes of data storage. If this calculation could be completed in 10 hours of cpu time, the required I/O speed would be one Petabyte per hour. This should be considered as only a representative example, and is certainly not an extreme case. Many simulations will require significantly more capabilities than this, particularly in the areas of storage and I/O speed.

4.4.3 Lattice QCD

Lattice QCD is a perfect problem for simple parallel computer architectures. High efficiency is very easy to reach. The Petaflops threshold will allow a dramatic change in the scope of numerical simulations of lattice QCD, which will become a really effective phenomenological tool and support to experiments. Weak interaction physics will be understood in a seriously quantitative way, and it will be possible to compute scattering amplitudes with high precision. Experiments like the one that are planned in this period (beauty and phi factories) will be able to exploit such a powerful help (quantitative predictions from the microscopic theory, without approximations).

As we already said, numerical simulations of lattice QCD can have very high efficiency even on very simple architectures. The problem is computationally intensive, since one always operates on complex 3×3 matrices: a low cpu memory bandwidth is acceptable. Since one is simulating a virtual world, and only needs to write on disk a few average numbers (apart from backups and check points), a powerful I/O channel is not needed. The problem is local and homogeneous, and the mapping to processor architecture straightforward. The cost-effective mesh architecture of Class III in Table 4.3 seems satisfactory. Further, a reasonable 200^4 lattice requires around 10 Terabytes of memory to match a Petaflops performance. Larger problems than this would require major new algorithms such as the multiscale renormalization group.

4.4.4 Computational Quantum Chemistry—HIV Protease Structure

Petaflops will have many applications studying the electronic structure of macromolecules, clusters, surfaces, and solids. Here, we discuss an example: The structure and properties of HIV protease by means of *ab initio* quantum chemical methods.

The detailed electronic structure of the HIV protease molecule can be elucidated using methods based on Hartree-Fock theory and its extensions (post-Hartree-Fock theory) such as configuration interaction and many-body perturbation theory. Using present day computers, application of these theories to such biological molecules is prohibitively expensive at the lower levels of approximation and become intractable as higher levels are used. These and related methods of *ab initio* quantum chemistry utilize the complete nonrelativistic Hamiltonian operator for a system comprised of N electrons and M nuclei and require the calculation of integrals involving kinetic energy, nuclear attraction and nuclear repulsion operators [Mulliken:81a].

The Hartree-Fock-Roothaan (HFR) procedure is used in most present day *ab initio* applications to polyatomic systems. It, or components thereof, also can be used as a point of departure for post-Hartree-Fock methods. HFR calculations use basis sets to define one-electron orbitals in the construction of molecular orbitals (MOs) as linear combinations of atomic orbitals (LCAOs). The basis functions are usually Gaussian-type functions (GTFs) centered at the atomic nuclei for a polyatomic system. For an m basis function representation of the N electrons in the system, the HFR procedure results in a total of $(m^2 + m)/2$ kinetic energy, $(m^2 + m)/2$ nuclear attraction, and $(m^4 + 2m^3 + 3m^2 + 2m)/8$ electron repulsion integrals. Although the $O(m^2)$ one-electron integrals is manageable with current computing technologies, the $O(m^4)$ scale-up of the two-electron integrals soon exhausts the capabilities of even the most advanced massively parallel processing supercomputers—in spite of the fact that the computation of such integrals (matrix elements) is "embarrassingly parallel" [Harrison:93a].

At the Hartree-Fock level, the $O(m^4)$ integrals are assembled into a Fock matrix of order n (where n equals the number of contracted GTFs; refer to example below) and its eigensolutions are extracted. This process is repeated over and over until solutions are invariant to within a pre-chosen threshold (i.e., attain self-consistency). Typically, this can require hundreds of iterations. A benchmark is a HFR calculation on a cluster of 135 beryllium atoms represented by a $(3s2p)/[2s1p]$ basis set of contracted GTFs [Ross:94a]. (The larger basis of 3 s-type and 2 p-type "primitive" GTFs is collected, using fixed coefficients derived from free-atom calculations, into the smaller "contracted" set of 2 s-type and 1 p-type GTFs for computational efficiency [Mulliken:81a].) The Be_{135} basis set contains $m = 1215$ primitive and $n = 675$ contracted GTFs. The $O(m^4)$ integrals step ($\sim 10^{12}$), which was reduced significantly using molecular point group symmetry, consumed 24 CRAY X-MP cpu hours.

Biological molecules such as HIV protease contain approximately $O(1,500)$ nuclei and $O(5,000)$ electrons. A modest, but not unreasonable, basis set could contain 10,000 basis functions. Simple scaling of the Be_{135} result indicates that it would require $\sim$ 30 cpu years on the approximately 1 Gigaflops CRAY X-MP to calculate the required 10^{16} two-electron integrals. This calculation reduces to a very tractable 15 minutes on a Petaflops computer.

To determine the equilibrium structure of HIV protease using Hartree-Fock theory would require repeating this 15-minute calculation for different molecular geometries until the total energy reaches a minimum. Since there are nearly 4,500 vibrational degrees of freedom, an unrestricted geometry search scaling roughly as the number of nuclei squared becomes prohibitive even on a Petaflops computer. Fortunately, it is appropriate to focus on the active site of the molecule and restrict the atomic motions to the few in its vicinity. The problem of geometry variation then can be addressed in hours to days.

Reactions and interactions at the active site are of particular interest in HIV protease. Because chemical bonds are formed and cleaved, it may be necessary to use higher levels of approximation than Hartree-Fock theory to calculate accurate results. Configuration interaction and many-body perturbation theory approaches can be used in such cases [Mulliken:81a]. A requirement for these post-HF methods is the so-called four-index transformation of the $O(m^4)$ basis function (contracted GTF) integrals to the MO basis—an $O(m^5)$ process [Covick:90a]. This step is followed in the configuration interaction (CI) procedure, for example, by construction of the Hamiltonian matrix comprised of elements connecting excitations or configurations involving so-called virtual MOs. In a full-CI procedure, the order L of the Hamiltonian matrix grows roughly as m^N for a configuration space of m MOs containing N electrons. In the case of HIV protease, this is $L = 10,000^{5000}$ configurations, a calculation that would take an inconceivable length of time. A more feasible procedure would be to treat at the CI level only those electrons in the vicinity of the active site. In this instance, the CI procedure becomes tractable and the calculation may be limited by the $O(m^5)$ transformation step. Furthermore, the excitation levels could be constrained to replacements of electrons from occupied to virtual MOs in order of $i = 1, 2, 3, \ldots$ for single, double, triple, etc., excitations. The CI calculation then scales roughly as m^i. The lowest energy eigensolutions of the $O(L)$ Hamiltonian matrix must be calculated—a formidable task even for a Petaflops computer handling a CI that scales as m^i because L is still greater than 10^8.

The discussion presented here clearly infers that the advent of Petaflops computing will lead to an unprecedented expansion in the scope of *ab initio* quantum chemistry. Biological systems such as HIV protease, which are currently impossible to study at any level of theory using today's state-of-the-art computers, will become rote using HFR methods on a Petaflops machine. And despite the fact that highly accurate full-CI calculations will remain out of reach, the feasibility of post-Hartree-Fock methods will no longer be an unattainable goal. Consequently, the advent of Petaflops supercomputing will result in the birth of *ab initio* quantum biochemistry and the coming of age of *ab initio* quantum chemistry.

4.4.5 Petaflops or PetaOPS Requirements from Genome Projects

The Human Genome project should produce the entire sequence of all 3×10^9 bases of human DNA in the next decades. It is reasonable to expect that in that same time frame it may be possible to do genomes of similar size, such as individual humans, experimental and domestic animals, crops, and others at the rate of one per year. Rates of 100 bases per second may be routine. Full, element-by-element comparisons of new sequences with existing data bases would require more than 10^{11} to 10^{12} elementary comparisons per second, compounded as the database grows. Clever heuristics can reduce the essential operations, but a newly discovered method could require complete re-analysis of all existing data. Full comparison of human and mouse genomes would need 10^{18}–10^{19} operations. This kind of comparative sequence analysis is already one of the most powerful sources of knowledge about genes, and shows promise of becoming more important as the data and knowledge bases grow. These discoveries will drive the need for even more sophisticated analyses.

Genomic information completely determines the characteristics of the protein and nucleic acid molecules that express a living organism's form and function. One of the greatest challenges, in which computation is playing a major role, is the prediction of higher-order structure from the one-dimensional sequence of genes. Rules for prediction of macromolecule folding are beginning to emerge. In the case of RNA, there are some simple rules that partially predict the secondary interactions of distant parts of the polymer chain. A deterministic, dynamic programming code is in wide use. It scales as $\sim N^3$ in operations and $\sim N^2$ in memory. Other non-deterministic methods also are available. More complex methods for predicting three-dimensional structure are appearing. Preliminary secondary structure predictions for sequences of HIV RNA (9,218 nucleotides) can be done on a 16 K processor SIMD Maspar or an 8-processor CRAY Y-MP in about six hours. There are 100,000s of sequences of potential interest ranging in size from 50 to 10,000 nucleotides.

Protein structure prediction is even of greater interest since proteins are the principal agents of expression for genetic information. Rules for prediction are more complex that for RNA, and are a research area of major concern. In its extreme, the problem could be viewed as of n^N complexity, where reasonable values for n, the number of conformations amino acids may take, could be dozens. Exhaustive conformational search would not be feasible for many proteins, even with Petaflops computers. However, even now there are a number of strategies to explore the problem. Exhaustive search on highly simplified lattice models, using simplified potential functions is partially successful. Statistics-based assignments of structure from sequence similarities is another. In all cases, refinement of final structures requires molecular mechanical and computational chemistry tools.

Much experimental work, involving heavy computation, is still needed in algorithm development for both the RNA and protein-folding problem.

Problem match to three categories of Petaflops computer: Almost every sort of high-performance architecture is easily adaptable to sequence comparison. Workstations are very competitive at the present time.

Class I machines are seldom used for sequence matching. High-precision word length is not needed. Lattice models are explored with supercomputers for their raw, scalar speed. Three-dimensional modeling is most frequently done on scalar plus vector, and limited multiprocessor shared memory.

Class II machines will do well on sequence comparison. Some HPCC efforts are adapting the molecular mechanical and electronic methods.

Class III machines are beginning to appear for sequence matching in commercial products. Some experimental adaptations of molecular mechanics calculations on heterogeneous architectures have been done.

4.4.6 Drug Design

Current costs of developing a new drug are several hundred million dollars. Good candidates are scarce. For example, by screening 10s to 100s of thousands of natural products and other chemicals, new anticancer agents emerge at less than one per year. Computation is anticipated as a powerful aid to designing and prescreening new candidates. This approach requires fundamental structural data on potential targets and the agents that may affect them. High-performance computer models are essential for analysis of experimental structural data to produce detailed molecular models, to molecular mechanical/dynamic exploration of target structures, to chemical structure of small drugs, and to interaction between drugs and targets. All of these methods will require Petaflops capability to achieve reliable prediction, design, and testing.

Structure determination requires computation in solution of the phase problems of x-ray crystallography and in distance geometry calculation for magnetic resonance. Computational chemistry is needed to understand the structure of drugs, how they bind to targets at the atomic level, and for details of electronic structure.

As examples, consider the protease of HIV and its inhibitors. This enzyme is required for mature, infectious virus to be produced, and is thus a candidate for drug targeting. It is being studied extensively. At this stage, computing speeds are on the order of 10^{12} fold slower than desired. Typically, 10^{-9}s of real time requires 100 CRAY Y-MP hours. We want to cover 1 ms of chemical time.

Petaflops performance would permit electronic calculations for the entire protease (at moderate levels of theory) and very detailed calculations for drug-sized molecules.

The calculations currently scale as N^2 for number of atoms in molecular mechanics calculations, with various heuristics to reduce from this upper bound. Electronic structure requirements vary depending on the level of theory, but range from N^4 upward. Memory requirements scale as N^2 for molecular mechanisms with additional offline storage for molecular dynamics trajectories. Electronic structure codes vary in memory requirement depending on whether storage and re-use of intermediate values or recomputation is chosen.

Problem match to three categories of Petaflops computers:

Class I—These codes traditionally are heavy users of Class I machines. Vectorization and multiprocessing are not high. Memory use is significantly less than 1 MW/MF for molecular mechanics and can be less or more than 1 MW/MF for electronic calculations.

Class II—Current codes are being ported to predecessor machines of this class. Managing communication is a significant effort.

Class III—A few demonstrations have been done on SIMD machines that may be analogous to Class III architectures.

4.4.7 Three-Dimensional Heart

The challenge is to develop a realistic three-dimensional model of the heart for improved design of prosthetic heart valves, modeling of cardiac diseases, and understanding the functional anatomy of the heart. Petaflops are needed to allow current promising models to be scaled to realistic levels of detail.

The present level of development successfully models some portions of the fluid dynamics of a heartbeat. It models the heart as geodesic fiber paths on surfaces in three dimensions, based on painstaking anatomical dissection of mammalian hearts. Fiber forces are transmitted to the blood by a special weighting function. Blood is represented currently by $128 \times 128 \times 128$ three-dimensional lattice of points on which fluid dynamics are calculated using versions of the Navier-Stokes equations. The problem scales in memory as slightly less than the grid size cubed, and in computational complexity as more than N^4. Present requirements are a CRAY C90 cpu-week and 50 Megawords of memory for a single beat. Realistic improvements would require a Petaflops computer, and could be utilized immediately to refine the many parameters, to achieve steady-state dynamics, and to introduce new features such as electrical activity. Methods developed for this work are applicable to problems of sperm motility, platelet aggregation, and other problems with flexible boundaries, such as blood vessels of the lung and heart.

Problem match to three categories of Petaflops computer:

Class I machines could be utilized immediately. Very large shared memory, vectorizable, multiprocessor codes are in use. The ratio of Megawords to Megaflops is less than one and a high fraction of theoretical peak speed is attained.

Class II machines are likely to be useful with modification of the codes currently being developed on clustered microprocessor machines.

Class III machines are likely to be applicable as well, since they appear to be successful for other fluid dynamics codes.

4.4.8 Global Surface Database

The goal of this application is to create and maintain a database of the surface coverage of the land mass of the Earth at a resolution of 10×10 square meters with samples acquired a few times per days. This database would be able to answer questions like:

- What is the expected quality and yield of this coming year's French wine? This estimate is to be based on the size of the fields being cultivated for grapes, the health of the vines, how they are being fertilized and irrigated, and so on.
- What is the likelihood of food riots in Mongolia this winter? In addition to crop health, need to determine the levels of water reservoirs, the quality of the water, how much fertilizer is being applied, and so on.
- Which forested areas are most susceptible to fires? How dry are the trees, and so on.
- How is energy use distributed in a city? How does usage vary with time of day, changes in weather, and so on?

To answer questions of this type, we propose a database of the surface coverage of the Earth. The land area of the Earth is 1.5×10^{14} meter^2. Dividing the land area in to 10×10 meter^2 grids, yields 1.5×10^{12} grid points.

We assume that each grid point is represented by a sample consisting of a $10^3 \times 10^3$ pixel images, where each pixel is contains intensity information (8 levels) at each of eight different frequencies. (Each pixel corresponds to a single square centimeter.) The image associated with each grid point consists of 6.4×10^7 bits. Further assume that the data can be compressed by a factor of 100; therefore, in compressed form, the data associated with each grid point will be 6.4×10^5 bits.

Satellite downlink characteristics. If 100 satellites are used for image gathering, with downlinks of 10^9 bits per second, we can transmit 1.6×10^5 images per second, or about 1.35×10^{10} images per day. Each grid point will be sampled approximately once every three to four months.

A typical wheat farm (10^4 acres or 4×10^7 square meters) consists of 4×10^5 grid points. Thus we would collect, on average, 3,668 images per day that pertain to the entire farm. In order to determine properly the state of the crops, it will be necessary to look at samples at different times during the day. Thus, the data we will have available for this farm is the equivalent of approximately 600 grid points sampled once every four hours, and from this data we are to determine the health of a field of 10,000 acres.

Data processing requirements. The compressed data stream produced by satellites is 8.6×10^{15} bits per day. Two steps need to be taken to convert this information into a form appropriate for queries: (1) some signal processing of the image data needs to be performed, and (2) the image at each grid point needs to be converted into information indicating the type of ground cover (wheat, rice, rock, etc.), and the state of the ground cover (dormant, germinating, ready for harvest, etc.). This final form we call a *model*, which consists of an identifier and a state. It is this form that is of most use to those making queries of the database. Since perfect matches with models will be rare, the database will need to associate with each grid point several "closest matching models." For simplicity, we assume that there will be 10^5 different identifier/state combinations, and that 32 closest matching models will be retained for each grid point.

The resulting data rates generated by the satellites are summarized in Figure 4.2. Notice that if the down link rates per satellite were increased to 10^{11} bits per second, nearly every grid point could be sampled each day.

The signal processing typically will require computing the Fourier transform of the image and performing some filtering operations on the resulting

	Downlink rate	Grid points	percent	Modeled bits
per second	1.0×10^{11}	1.56×10^{5}	0.00%	8.30×10^{7}
per day	8.64×10^{15}	1.35×10^{10}	0.91%	7.18×10^{12}
per year	3.16×10^{18}	4.93×10^{12}	331%	2.62×10^{15}

Figure 4.2: Data Rates in Surface Model

set of spectral coefficients. Using some distance metric, these filtered spectral coefficients then are compared against a table of spectral coefficients for different identifier/state combinations and "closest matching models" are selected.

For simplicity, we assume that the signal processing required is a small integer multiple of the cost of performing a Fourier transform of the image attached to each grid point. Each of these images is a $10^3 \times 10^3 \times 8$ array of intensities. Assuming an n point fast Fourier transform takes $n \log n$ floating point operations, the signal processing required per grid point is 8×10^7 flops. Since the satellites can transmit 1.56×10^5 images per second, we will need roughly 1.25×10^{13} floating point operations per second just to perform the basic signal processing. This computation, however, is highly parallelizable and requires little inter-processor communication, since each image can be processed independently.

For template matching, we need to match against 10^5 total models. Each grid point consists of 8×10^6 spectral coefficients, and the distance between these coefficients and the corresponding coefficients of each model must be computed. Using the simple minded approach of computing the distance between the image and each of the models will require a total of 8×10^{11} operations for each grid point. Since 1.56×10^5 images are provided by the satellites each second, 1.25×10^{17} comparisons will be required per second. Without improving the comparison algorithm, this dominates the signal processing cost. Again this computation is highly parallelizable. Storing the data required by the 32 closest matching models will require 532 bits.

A database consisting of one set of 32 models per grid point would require 7.92×10^{14} bits. The satellites will produce 7.18×10^{12} model bits per day (since not every grid point is modeled each day). Thirty years of modeled

	N cost	$N \log N$ cost	N^2 cost
All archived data	1.14×10^{17}	6.04×10^{18}	1.12×10^{33}
U.S. data, one sample/grid	5.73×10^{14}	2.61×10^{16}	1.12×10^{28}
Farm 10^4 acres	4.68×10^6	8.72×10^7	1.90×10^{12}

Figure 4.3: Query processing time in operations per second

data will require 9.8×10^{15} bits of modeled data, or one Petabyte.

It is not sufficient to preserve the transformed (modeled) versions of the data. Over time, improved templates will be developed and it then will be useful to re-do the template matching. Each day, the data produced by the satellites in compressed grid point images is 8.4×10^{15} bits. The raw data produced by the satellites over a 30-year period is at least 1.18×10^{19} bits, or one exabyte.

Database Queries. Queries of the global surface database will be made by agricultural planners, economic planners, city planners, climate modelers, sociological studies, political studies, etc. We anticipate upwards of 10^6 queries per day or 12 queries per second, each query requiring correlations over varying regions of the database.

The processed database is approximately 10^{15} bytes. Some queries will use substantial portions of the database and the queries will need to be answered on the order minutes. (For reference, the United States is 1/16 of the land mass of the planet.) A query that needs access to the entire database in five minutes would require a bandwidth of 32.8 Terabytes per second. Accessing a single model set for each grid point in the United States in five minutes requires bandwidth of 165 Gigabytes per second.

The table in Figure 4.3 estimates the computational cost required for different classes of queries. It is assumed that 10^6 queries per day are to be processed and each is identical. Three different size queries are estimated: (1) one that uses the entire data contained in the database, (2) one that uses the data corresponding to one sample for each grid point in the United States, and (3) one that uses the data corresponding to one sample for each grid point on a 10^4 acre farm. In addition, it is assumed that the computation required to handle the query is either linear, $n \log n$, or quadratic.

Clearly, Peta-operation machines will have difficulties with these query rates if a query requires more than linear processing. However, even large-scale queries that deal with all of the archived information can (barely) be handled by Peta-machines if efficient (linear) algorithms can be developed.

4.4.9 Video Image Fusion

Description: This application combines real-time video or other image sensor data with solids models of an object or complex physical system to create a CyberScene. This combined dataset provides users an immersive virtual environment that can be explored in real time. Typical applications include exploration via telepresence of dangerous, small or large scale or otherwise inaccessible physical locations for example, reactor cores, deep sea vents, manufacturing floors, inside of human body, etc. A system with 10^3 high-resolution (10^6 pixel) input video streams (these video or other arrayed sensor data most likely will come from highly integrated array cameras) requires a Terabit per second (Tbps) of input bandwidth and would be used to computationally reconstruct a 10^{12} voxel 3D CyberScene object. This object will be merged with any available CAD or solid model of the physical system. These CAD or solids models may be based on 3D laser scans or other active scanning databases but we do not consider the data or computational requirements for active scanning here. We estimate real-time voxel reconstruction to require on the order of a 10^5–10^6 floating point operations per voxel per second and 10–10^2 communications events. Communications events are needed for merging multiple pixels into a single 3D pixel (voxel). Existing 3D image reconstruction systems are limited to a small number of depth planes; therefore, we increase the number of input sources to improve the number of depth planes in the final reconstructed 3D image to provide uniform spatial resolution of the CyberScene. Output bandwidth requirements depend on the number of users. We estimate 100 users would require a Tbps of output bandwidth to provide for 100 high-resolution 3D immersive displays (CAVE or high resolution head mounted displays).

System Requirements: This application requires a Petaflops computer system with between 10^3 and 10^5 processors. Global communication is required since voxel reconstruction requires parallel FFTs and/or wavelets and correlation and registration of multiple image planes. Global communication is required for image registration; global sums are needed for FFTs, correlation, etc. Total primary storage of $\sim 10^{15}$ bytes is required. Multiple seconds of image data are required to be in primary storage for motion estimation while motion parallax extraction requires $O(10^{14})$ bytes, and underlying 3D solids model will require $O(10^{14})$ bytes, and additional primary memory is needed for buffer space for secondary store. The Petaflops system requires significant real-time I/O capability both for user data streams but also for archival storage of cyberspace events. Secondary storage will be used for both archiving input directly, but also for recording and playback of events (e.g., multiple Tbps to secondary store is needed). Secondary store capacity needs to be in the range of 10–100 Petabytes. The algorithms for CyberScene reconstruction can use computer systems with rather deep memory hierarchies; however, it is not clear that this application can make use of SIMD architectures.

Impact: CyberScene capability can have a broad economic impact. The potential uses include: telepresence for dangerous environment work (reactors, contamination zones, biological hazards and deep sea construction, mining), simulation and training of workers for complex manufacturing environments (highly automated assembly lines), entertainment and education, mass participation in remote space exploration activities (space station, lunar and mars based environments), 3D high-end telecommuting, and merging of VR with RR.

4.5 Algorithmic Issues for the Petaflops Computer

Algorithms sit between the application and the machine and interact with both. Advances in the performance and architecture of the machine may have a serious impact on the choice of algorithms and vice versa.

The main algorithmic issue in going from a Teraflops machine to a Petaflops machine is *scalability* (because the problems to be solved on the latter will be necessarily much larger), in terms of both the algorithmic efficiency (i.e., the serial complexity of the algorithm) and the parallelization efficiency. Unfortunately, these are often two conflicting goals.

We list, in Table 4.5, three different types of algorithms viewed in these two terms:

We expect a Petaflops machine to have two essential features that will influence critically the proper choice of algorithms: the memory will be hierarchical (in terms of access time) and the processor interconnect will be hierarchical (e.g., a MIMD network of SIMD machines). We expect that

Table 4.5: Characteristics of General Algorithm Classes

Type of Algorithm	Example	Algorithm Scalability	Parallel Scalability
Local/regular	PDE time marching	poor	good
	Monte Carlo	poor	good
global	Pseudo time stepping	poor	good
	Sparse matrix inversion	poor	poor
	Multigrid	good	not optimal
dynamic/irregular	Local mesh refinement	good	poor

these architectures will favor algorithms that are also hierarchical in nature, because there will be a better match in terms of the algorithm's demand for synchronization/communication and the efficiency with which the machine can deliver them. For example, for elliptic PDEs, a hierarchical architecture should favor hierarchical algorithms such as multigrid and domain decomposition. These types of algorithms require both local and global data interaction but the amount of interaction decreases with the distance between the processors. Such an architecture may also favor a preconditioned iterative approach where the preconditioner can be chosen to better match the architecture.

On the other hand, a potential problem is that it is often difficult to implement hierarchical algorithms to run at perfect parallel efficiency. A Petaflops machine will necessarily involve a large degree of parallelism and it may increase the percentage of time an algorithm spends on synchronization and communication (the Amdahl's Law effect). For example, on the coarser grids, there are fewer data to keep the machine busy. Much more research is needed, both in new algorithms that parallelize across grid levels and also in architectures that do not impose a heavy penalty for these kinds of multilevel communication patterns.

Another issue is problem scalability. For some problems, the scale-up in the size of the problem does not give rise to increased parallelism. For example, in certain molecular dynamics applications, one may want to fix the number of molecules but increase the number of time steps. Unfortunately, it is difficult to extract parallelism in the time direction. We may need to develop better algorithms to deal with these kinds of problems.

A final issue is precision. We may need to increase the precision as the problem size grows. For example, for many elliptic PDE problems, the conditioning grows like $O(n^2)$ (2nd-order problems) or $O(n^4)$ (4th-order problems), where n is the number of grid points in each co-ordinate direction. Therefore, for $n = 10^4$, a 4th-order elliptic solver may not have any accurate significant digits on current 64-bit word machines. A Petaflops machine may need to have increased precision. Alternatively, more stable and better conditioned discretizations can be employed. For example, Monte Carlo methods are less sensitive to precision than those based on solving differential equations. Further, such statistical methods are often straightforward to parallelize. Thus, we anticipate that further investigation of Monte Carlo algorithms may be important for Petaflops scale applications.

Bibliography

[Covick:90a] Covick, L. A., and Sando, M. K. *J. Comput. Chem.*, 11:1151, 1990.

[Harrison:93a] Harrison, R. J., and Kendall, R. A. *Theoret. Chim. Acta*, 84:489, 1993.

[Mulliken:81a] Mulliken, R. S., and Ermler, W. C. *Polyatomic Molecules: Results of ab Initio Calculations.* Academic Press, New York, 1981.

[Ross:94a] Ross, R. B., Ermler, W. C., Kern, C. W., and Pitzer, R. M. "Modelling bulk berylium," *Supercomputer*, 1994. Accepted for Publication.