Challenges of Future High-End Computing

David H. Bailey

NERSC, Lawrence Berkeley Lab
Mail Stop 50B-2239
Berkeley, CA 94720
dhb@nersc.gov

A Petaflops Computer System

- 1 Pflop/s (10^{15} flop/s) in computing power.
- Between 10,000 and 1,000,000 processors.
- Between 10 Tbyte and 1 Pbyte main memory (1 Pbyte =100 times the capacity of the U. C. Berkeley library).
- Between 1 Pbyte and 100 Pbyte on-line mass storage.
- Between 100 Pbyte and 10 Ebyte archival storage.
- Commensurate I/O bandwidth, etc.
- If built today, would cost $50 billion and consume 1,000 Mwatts of electric power.
- May be feasible and “affordable” by the year 2010 or sooner.

Petaflops Computers: Who Needs Them?

xpert predictions:

- (c. 1950) Thomas J. Watson: only about six computers are needed worldwide.
- (c. 1977) Seymour Cray: there are only about 100 potential customers worldwide for a Cray-1.
- (c. 1980) IBM study: only about 50 Cray-class computers will be sold per year.

resent reality:

- Some private homes now have six Cray-1-class computers!

Applications for Petaflops Systems

- Nuclear weapons stewardship.
- Cryptology and digital signal processing.
- Satellite data processing.
- Climate and environmental modeling.
- Design of advanced aircraft and spacecraft.
- Design of practical fusion energy systems.
- Pattern matching in DNA sequences.
- 3-D protein molecule simulations.
- Global-scale economic modeling.
- Virtual reality design tools for molecular nanotechnology.

lus numerous novel applications that can only be dimly envisioned now.

SIA Semiconductor Technology Roadmap

haracteristic	1999	2001	2003	2006	2009
eature size (micron)	0.18	0.15	0.13	0.10	0.07
RAM size (Mbit)	256	1024	1024	4096	16K
ISC processor (MHz)	1200	1400	1600	2000	2500
ransistors (millions)	21	39	77	203	521
ost per transistor (ucents)	1735	1000	580	255	100

bservations:

- Moore's Law of increasing density will continue until at least 2009.
- Clock rates of RISC processors and DRAM memories are **not** expecte to be more than about twice today's rates.

onclusion: Future high-end systems will feature tens of thousands of processors, with deeply hierarchical memories.

Designs for a Petaflops System

Commodity technology design:

- 100,000 nodes, each of which is a 10 Gflkop/s processor.
- Clock rate = 2.5 GHz; each processor can do four flop per clock.
- Multi-stage switched network.

Hybrid technology, multi-threaded (HTMT) design:

- 10,000 nodes, each with one superconducting RSFQ processor.
- Clock rate = 100 GHz; each processor sustains 100 Gflop/s.
- Multi-threaded processor design handles a large number of outstandin memory references.
- Multi-level memory hierarchy (CRAM, SRAM, DRAM, etc.).
- Optical interconnection network.

Little's Law of Queuing Theory

ittle's Law:

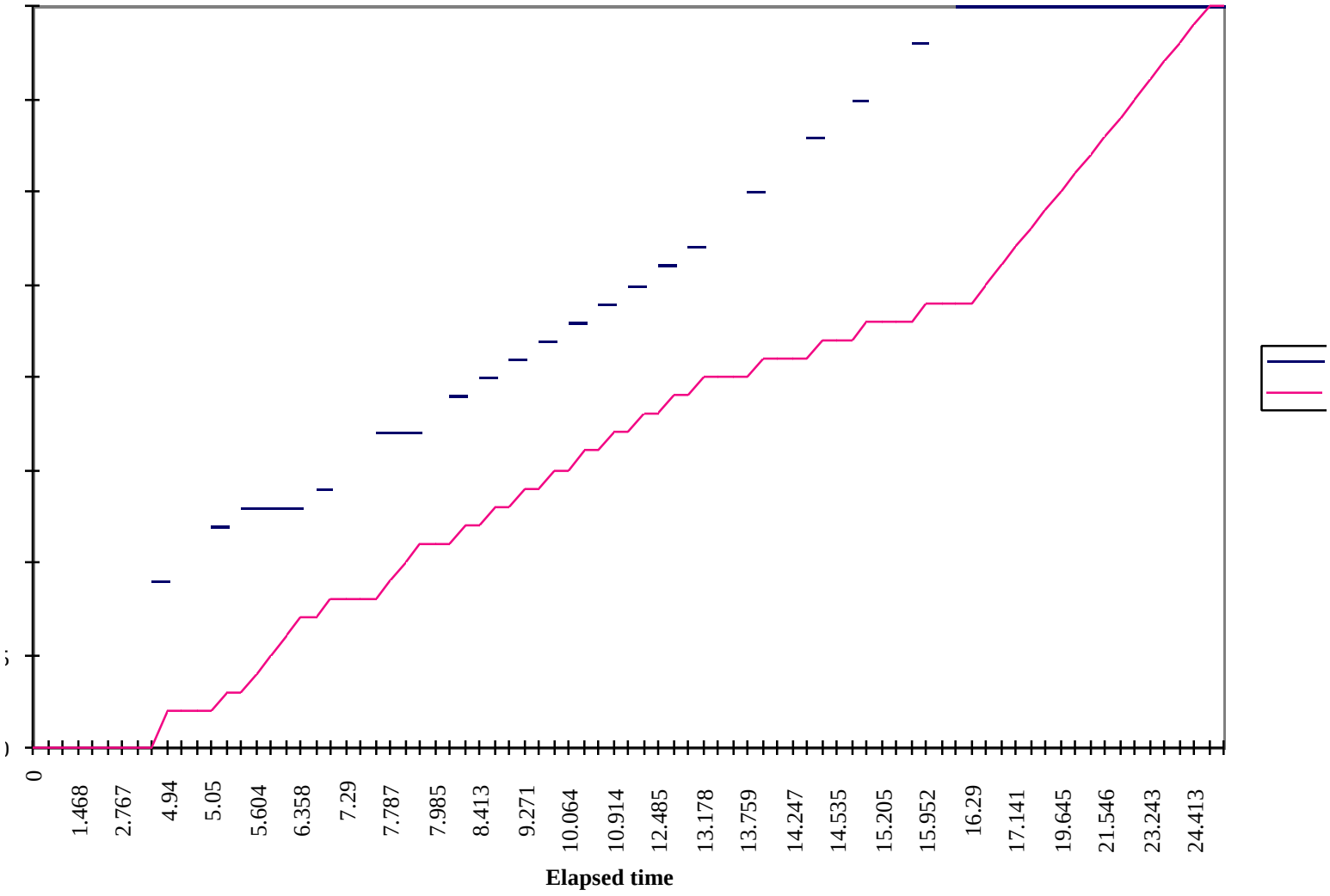
verage number of waiting customers =

average arrival rate x average wait time per customer.

roof:

)efine f(t) = cumulative number of arrived customers, and g(t) = cumulative number of departed customers. Assume f(0) = g(0) = 0, ar f(T) = g(T) = N. Consider the region between f(t) and g(t). By Fubini theorem of measure theory, one can evaluate this area by integration along either axis. Thus Q T = D N, where Q is average length of queue, and D is average delay per customer. In other words, Q = (N/T D.

Little's Law


0
1.468
2.767
4.94
5.05
5.604
6.358
7.29
7.787
7.985
8.413
9.271
10.064
10.914
12.485
13.178
13.759
14.247
14.535
15.205
15.952
16.29
17.141
19.645
21.546
23.243
24.413
Elapsed time

Little's Law of High Performance Computing

Assume:

- Single processor-memory system.
- Computation deals with data in local main memory.
- Pipeline between main memory and processor is fully utilized.

Then by Little's Law, the number of words in transit between CPU and memory (i.e. length of vector pipe, size of cache lines, etc.)

= memory latency x bandwidth.

This observation generalizes to multiprocessor systems:

concurrency = latency x bandwidth,

where "concurrency" is aggregate system concurrency, and "bandwidth" is aggregate system memory bandwidth.

This form of Little's Law was first noted by Burton Smith of Tera.

Little's Law and Petaflops Computing

Assume:

- DRAM memory latency = 100 ns.
- There is a 1-1 ratio between memory bandwidth (word/s) and sustaine performance (flop/s).
- Cache and/or processor system can maintain sufficient outstanding memory references to cover latency.

Commodity design:

- Clock rate = 2.5 GHz, so latency = 250 CP. Then system concurrency = 100,000 x 4 x 250 = 10^8.

HTMT design:

- Clock rate = 100 GHz, so latency = 10,000 CP. Then system concurrency = 10,000 x 10,000 = 10^8.

But by Little's Law, system concurrency = 10^{-7} x 10^{15} = 10^8 in each case.

Amdahl's Law and Petaflops Computing

ssume:

- Commodity petaflops system -- 100,000 CPUs, each of which can sustain 10 Gflop/s.
- 90% of operations can fully utilize 100,000 CPUs.
- 10% can only utilize 1,000 or fewer processors.

hen by Amdahl's Law,

Sustained performance $< 10^{15} / [0.9/10^{5} + 0.1/10^{3}]$

$= 9.2 \times 10^{12}$ flop/s,

which is only about 1% of the system's presumed achievable performance.

Concurrency and Petaflops Computing

- Conclusion: No matter what type of processor technology is used, applications on petaflops computer systems must exhibit roughly 100 million way concurrency at virtually every step of the computation, or else performance will be disappointing.
  - This assumes that most computations access data from local DRAM memory, with little or no cache re-use (typical of many applications).
  - If substantial long-distance communication is required, the concurrenc requirement may be even higher!

- Key question: Can applications for future systems be structured to exhibi these enormous levels of concurrency?

Latency and Data Locality

	Latency	
ystem	**Sec.**	**Clocks**
GI O2, local DRAM	320 ns	62
GI Origin, remote DRAM	1us	200
3M SP2, remote node	40 us	3,000
[TMT system, local DRAM	50 ns	5,000
[TMT system, remote memory	200 ns	20,000
GI cluster, remote memory	3 ms	300,000

Algorithms and Data Locality

- Can we quantify the inherent data locality of key algorithms?
- Do there exist “hierarchical” variants of key algorithms?
- Do there exist “latency tolerant” variants of key algorithms?
- Can bandwidth-intensive algorithms be substituted for latency-sensitiv algorithms?
- Can Little’s Law be “beaten” by formulating algorithms that access data lower in the memory hierarchy? If so, then systems such as HTMT can be used effectively.

A Hierarchical, Latency Tolerant Algorithm for Large 1-D FFTs

- Regard input data of length n = p q as a p x q complex matrix, distributed so that each node contains a block of columns.
- Transpose to q x p matrix.
- Perform q-point FFTs on each of the p columns.
- Multiply resulting matrix by exp (-2 pi i j k / n), where j and k are row and column indices of matrix.
- Transpose to p x q matrix.
- Perform p-point FFTs on each of the q columns.
- Transpose to a q x p matrix.

eatures:

- Computational steps are embarrassingly parallel -- no communication.
- Transpose operations can be done as latency tolerant block transfers.
- This scheme can be recursively employed for each level of hierarchy.

Numerical Scalability

- or the solvers used in most of today's codes, condition numbers of the linear systems increase linearly or quadratically with grid resolution.
- 'he number of iterations required for convergence is directly proportiona to the condition number.

:onclusions:

- Solvers used in most of today's applications are not numerically scalable.
- Research in novel techniques now being studied in the academic worl especially domain decomposition and multigrid, may yield fundamentally more efficient methods.

System Performance Modeling

tudies must be made of future computer system and network designs, years before they are constructed.

calability assessments must be made of future algorithms and applications, years before they are implemented on real computers.

pproach:

- Detailed cost models derived from analysis of codes.
- Statistical fits to analytic models.
- Detailed system and algorithm simulations, using discrete event simulation programs.

Performance Model of the NAS LU Benchmark

'otal run time T per iteration is given by

$$T = 485\ F\ N^3\ 2^{-2K} + 320\ B\ N^2\ 2^{-K} + 4\ L + [1 + 2\ (2^K - 1) / (N - 2)]$$
$$\{2\ (N - 2)\ [279\ F\ N^2\ 2^{-2K} + 80\ B\ N\ 2^{-K} + L)] + 953\ F\ (N - 2)\ N^2\ 2^{-2k}\}$$

/here

= node-to-node latency (assumed not to degrade with large K)

= node-to-node bandwidth (assumed not to degrade with large K)

= floating point rate

I = grid size

$= 2^{2K}$ = number of processors

.cknowledgment: Maurice Yarrow and Rob Van der Wijngaart, NASA

Hardware and Architecture Issues

- Commodity technology or advanced technology?
- How can the huge projected power consumption and heat dissipation requirements of future systems be brought under control?
- Conventional RISC or multi-threaded processors?
- Distributed memory or distributed shared memory?
- How many levels of memory hierarchy?
- How will cache coherence be handled?
- What design will best manage latency and hierarchical memories?

How Much Main Memory?

- 5-10 years ago: One word (8 byte) per sustained flop/s.
- Today: One byte per sustained flop/s.
- 5-10 years from now: 1/8 byte per sustained flop/s may be adequate.

/4 rule: For many 3-D computational physics problems, main memory scales as d^3, while computational cost scales as d^4, where d is linea dimension.

[owever:

- Advances in algorithms, such as domain decomposition and multigrid, may overturn the 3/4 rule.
- Some data-intensive applications will still require one byte per flop/s o more.

Programming Languages and Models

MPI, PVM, etc.

- Difficult to learn, use and debug.
- Not a natural model for any notable body of applications.
- Inappropriate for distributed shared memory (DSM) systems.
- The software layer may be an impediment to performance.

HPF, HPC, etc.

- Performance significantly lags behind MPI for most applications.
- Inappropriate for a number of emerging applications, which feature large numbers of asynchronous tasks.

Java, SISAL, Linda, etc.

- Each has its advocates, but none has yet proved its superiority for a large class of highly parallel scientific applications.

Towards a Petaflops Language

- High-level features for application scientists.
- Low-level features for performance programmers.
- Handles both data and task parallelism, and both synchronous and asynchronous tasks.
- Scalable for systems with up to 1,000,000 processors.
- Appropriate for parallel clusters of distributed shared memory nodes.
- Permits both automatic and explicit data communication.
- Designed with a hierarchical memory system in mind.
- Permits the memory hierarchy to be explicitly controlled by performance programmers.

System Software

- How can tens or hundreds of thousands of processors, running possibl thousands of separate user jobs, be managed?
- How can hardware and software faults be detected and rectified?
- How can run-time performance phenomena be monitored?
- How should the mass storage system be organized?
- How can real-time visualization be supported?

xotic techniques, such as expert systems and neural nets, may be needed to manage future systems.

Faith, Hope and Charity

Until recently, the high performance computing field was sustained by

- Faith in highly parallel computing technology.
- Hope that current faults will be rectified in the next generation.
- Charity of federal government(s).

Results:

- Numerous firms have gone out of business.
- Government funding has been cut.
- Many scientists and lab managers have become cynical.

Where do we go from here?

Time to Get Quantitative

- Quantitative assessments of architecture scalability.
- Quantitative measurements of latency and bandwidth.
- Quantitative analyses of multi-level memory hierarchies.
- Quantitative analyses of algorithm and application scalability.
- Quantitative assessments of programming languages.
- Quantitative assessments of system software and tools.

et the analyses begin!