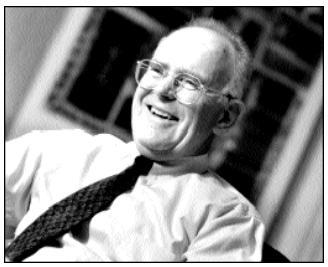
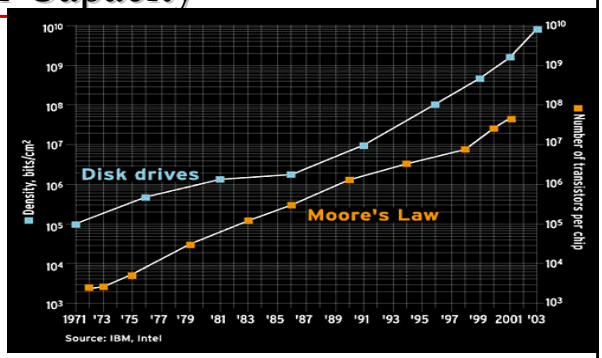
Trends in High Performance Computing and Using Numerical Libraries on Clusters

(Supercomputer and Clusters and Grids, Oh My!)

Jack Dongarra
University of Tennessee
Oak Ridge National Laboratory

Technology Trends: Microprocessor Capacity



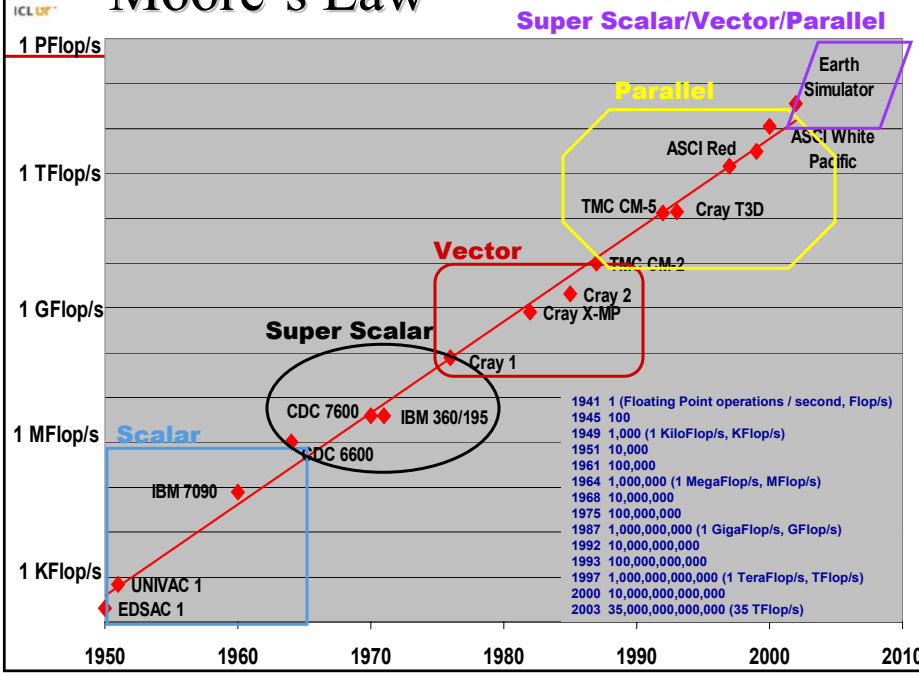


Gordon Moore (co-founder of Intel) predicted in 1965 that the transistor density of semiconductor chips would double roughly every 18 months.

2X transistors/Chip Every 1.5 years
Called "Moore's Law"

Microprocessors have become smaller, denser, and more powerful.
Not just processors, bandwidth, storage, etc.
2X memory and processor speed and ½ size, cost, & power every 18 months.

Moore's Law

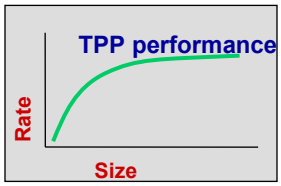


TOP500 supercomputer

H. Meuer, H. Simon, E. Strohmaier, & JD

- Listing of the 500 most powerful Computers in the World
- Yardstick: Rmax from LINPACK MPP

  $Ax=b$, *dense problem*

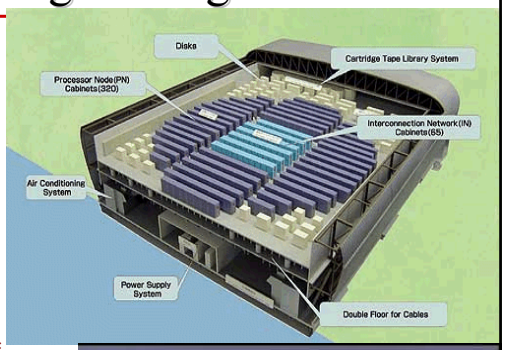


- Updated twice a year
  SC'xy in the States in November
  Meeting in Mannheim, Germany in June
- All data available from **www.top500.org**

A Tour de Force in Engineering

- Homogeneous, Centralized, Proprietary, Expensive!
- Target Application: CFD-Weather, Climate, Earthquakes
- 640 NEC SX/6 Nodes (mod)
  - 5120 CPUs which have vector ops
  - Each CPU 8 Gflop/s Peak
- 40 TFlop/s (peak)
- $1/2 Billion for machine & building
- Footprint of 4 tennis courts
- 7 MWatts
  - Say 10 cent/KWhr - $16.8K/day = $6M/year!
- Expect to be on top of Top500 until 60-100 TFlop ASCI machine arrives

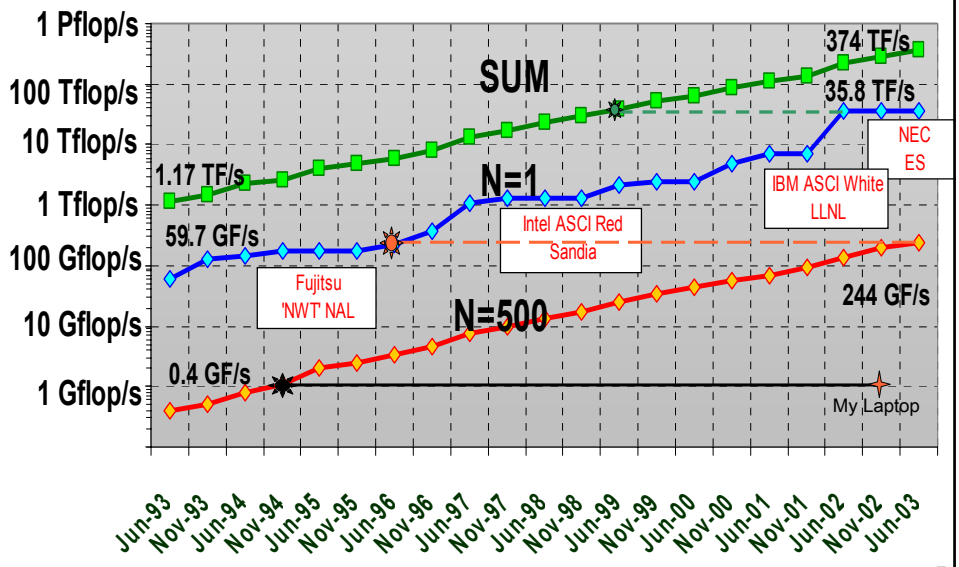
- From the Top500 (June 2003)
  - Performance of ESC ≈ Σ Next Top 4 Computers
  - ~ 10% of performance of all the Top500 machines



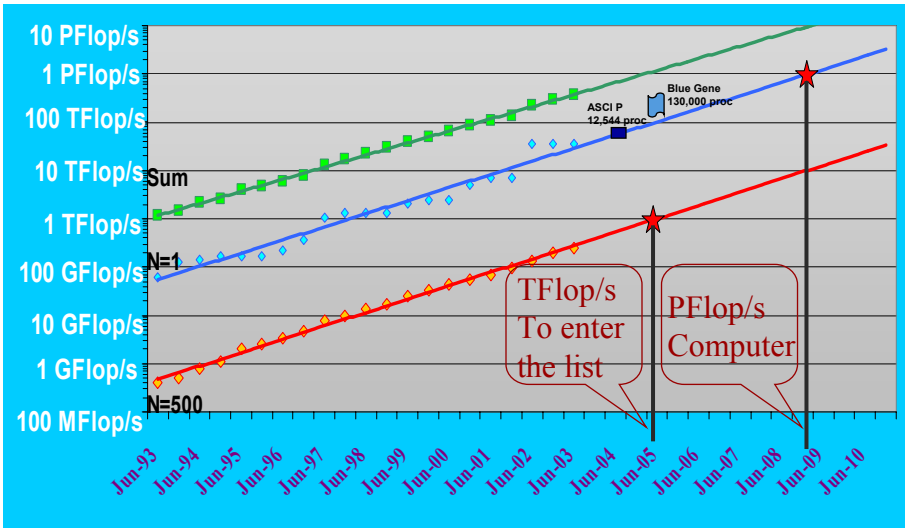
June 2003

	Manufacturer	Computer	Rmax	Installation Site	Year	# Proc	Rpeak
1	NEC	Earth-Simulator	35860	Earth Simulator Center Yokohama	2002	5120	40960
2	Hewlett-Packard	ASCI Q - AlphaServer SC ES45/1.25 GHz	13880	Los Alamos National Laboratory Los Alamos	2002	8192	20480
3	Linux NetworX Quadrics	MCR Linux Cluster Xeon 2.4 GHz - Quadrics	7634	Lawrence Livermore National Laboratory Livermore	2002	2304	11060
4	IBM	ASCI White, SP Power3 375 MHz	7304	Lawrence Livermore National Laboratory Livermore	2000	8192	12288
5	IBM	SP Power3 375 MHz 16 way	7304	NERSC/LBNL Berkeley	2002	6656	9984
6	IBM/Quadrics	xSeries Cluster Xeon 2.4 GHz - Quadrics	6586	Lawrence Livermore National Laboratory Livermore	2003	1920	9216
7	Fujitsu	PRIMEPOWER HPC2500 (1.3 GHz)	5406	National Aerospace Lab Tokyo	2002	2304	11980
8	Hewlett-Packard	rx2600 Itanium2 1 GHz Cluster - Quadrics	4881	Pacific Northwest National Laboratory Richland	2003	1540	6160
9	Hewlett-Packard	AlphaServer SC ES45/1 GHz	4463	Pittsburgh Supercomputing Center Pittsburgh	2001	3016	6032
10	Hewlett-Packard	AlphaServer SC ES45/1 GHz	3980	Commissariat a l'Energie Atomique (CEA) Bruyeres-le-Chatel	2001	2560	5120

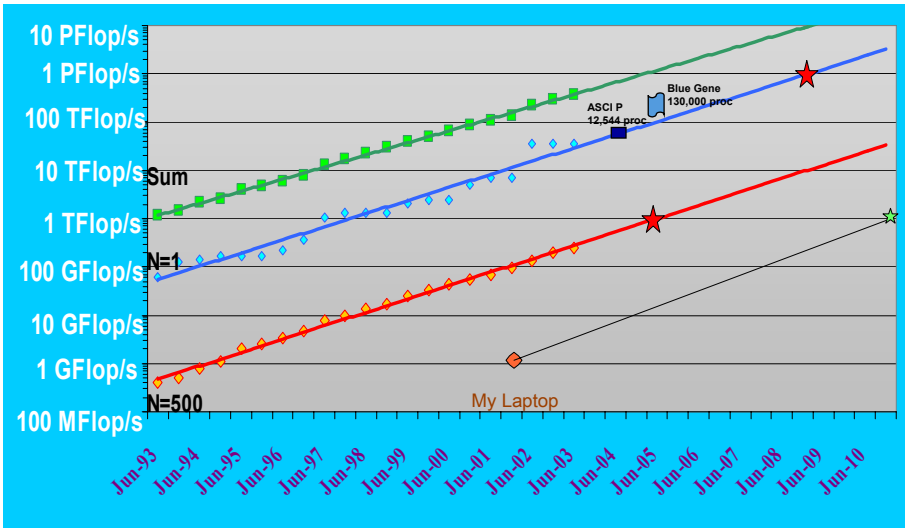
TOP500 – Performance - June 2003



Performance Extrapolation



Performance Extrapolation



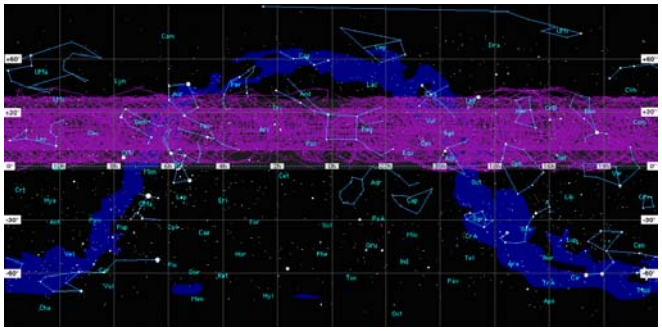
Excerpt from the Top500 - 21th list

Rank	Manufacturer	Computer	Rmax [TF/s]	Installation Site	Country	# Proc
	…	…	…	…	…	…
3	Linux Networx	MCR Linux Cluster Xeon - Quadrics	7.634	Lawrence Livermore National Laboratory	USA	2304
6	IBM	xSeries Cluster Xeon – Quadrics	6.586	Lawrence Livermore National Laboratory	USA	1920
8	Hewlett-Packard	rx2600 Itanium2 - Quadrics	4.881	Pacific Northwest National Laboratory	USA	1540
11	HPTi	Aspen Systems, Xeon – Myrinet2000	3.337	Forecast Systems Laboratory – NOAA	USA	1536
19	Atipa Technology	P4 Xeon Cluster - Myrinet	2.207	Louisiana State University	USA	1024
25	Dell	PowerEdge 2650 P4 Xeon – Myrinet	2.004	University at Buffalo, SUNY, CCR	USA	600
31	IBM	Titan Cluster Itanium2 – Myrinet	1.593	NCSA	USA	512
39	Self-made	PowerRACK-HX Xeon GigE	1.202	University of Toronto	Canada	512
…	…	…	…	…	…	…

- Not "bottom feeders"
- 149 Clusters on the Top500
- 119 are Intel based
- A substantial part of these are installed at industrial customers especially in the oil-industry.
- 23 of these clusters are labeled as 'Self-Made'.

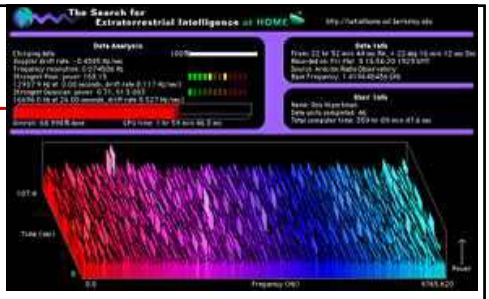
SETI@home: Global Distributed Computing

- Running on 500,000 PCs, ~1300 CPU Years per Day
  - 1.3M CPU Years so far
- Sophisticated Data & Signal Processing Analysis
- Distributes Datasets from Arecibo Radio Telescope



SETI@home

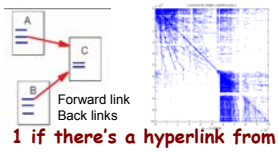
- Use thousands of Internet-connected PCs to help in the search for extraterrestrial intelligence.
- When their computer is idle or being wasted this software will download ~ half a MB chunk of data for analysis. Performs about 3 Tflops for each client in 15 hours.
- The results of this analysis are sent back to the SETI team, combined with thousands of other participants.



- **Largest distributed computation project in existence**
  - Averaging 55 Tflop/s

Google™

- Google query attributes
  - 150M queries/day (2000/second)
  - 100 countries
  - 3B documents in the index
- Data centers
  - 15,000 Linux systems in 6 data centers
    - 15 TFlop/s and 1000 TB total capability
    - 40-80 1U/2U servers/cabinet
    - 100 MB Ethernet switches/cabinet with gigabit Ethernet uplink
  - growth from 4,000 systems (June 2000)
    - 18M queries then
- Performance and operation
  - simple reissue of failed commands to new servers
  - no performance debugging
    - problems are not reproducible
- Eigenvalue problem
  - $n=2.7 \times 10^9$ (see: MathWorks Cleve's Corner)

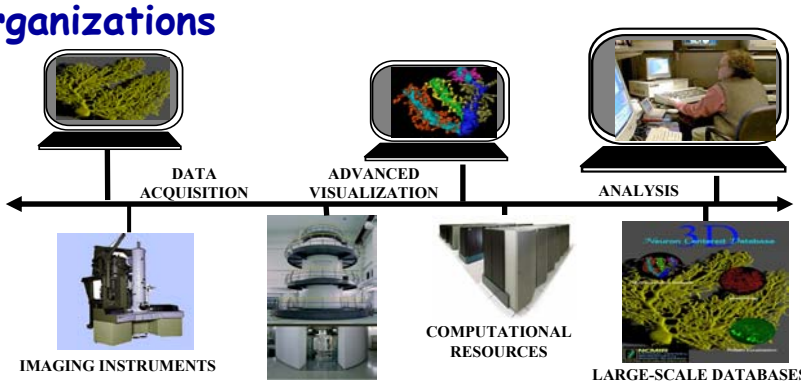


  - 1 if there's a hyperlink from page *i* to *j*
- Form a transition probability matrix of the Markov chain
  - Matrix is not sparse, but it is a rank one modification of a sparse matrix
- Largest eigenvalue is equal to one; want the corresponding eigenvector (the state vector of the Markov chain).
  - The elements of eigenvector are Google's PageRank (Larry Page).
- When you search: They have an inverted index of the web pages
  - Words and links that have those words
- Your query of words: find links then order lists of pages by their PageRank.

Source: Monika Henzinger, Google & Cleve Moler

Grid Computing is About …

Resource sharing & coordinated problem solving in dynamic, multi-institutional virtual organizations



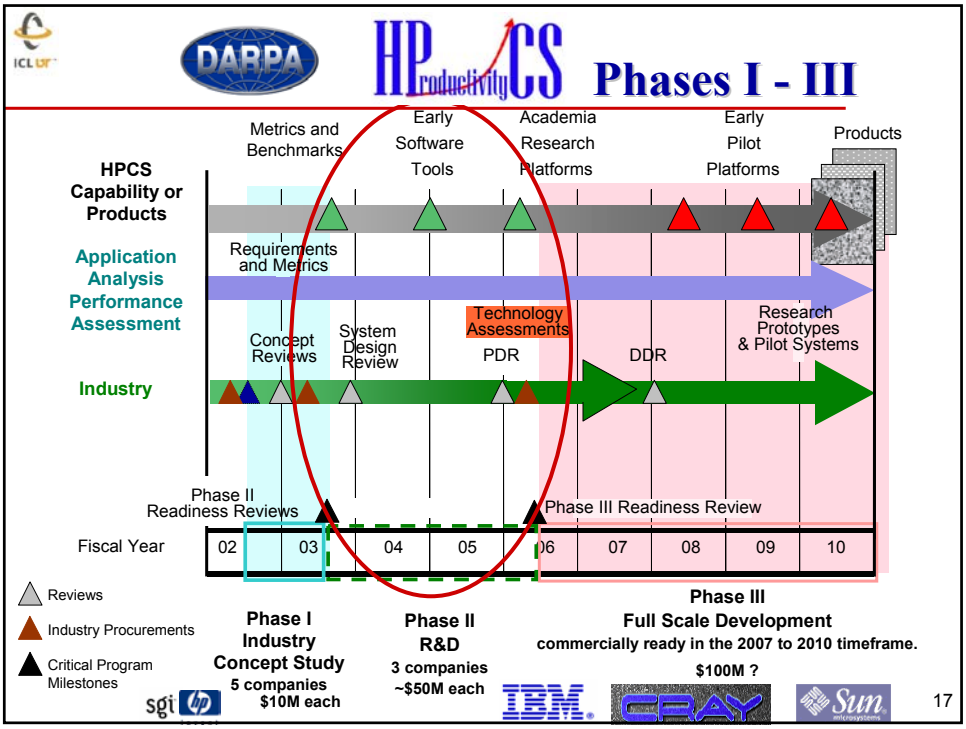
"Telescience Grid", Courtesy of Mark Ellisman

The Computing Continuum

- Each strikes a different balance
  - computation/communication coupling
- Implications for execution efficiency
- *Applications for diverse needs*
  - *computing is only one part of the story!*

Selected System Characteristics

		Earth Simulator (NEC)	Cray X1 (Cray)	ASCI Q (HP ES45)	MCR (Dual Xeon)
Year of Introduction		2002	2003	2003	2002
Node Architecture		Vector SMP	Vector SMP	Alpha micro SMP	Xeon micro SMP
System Topology		NEC single-stage Crossbar	2D Torus Interconnect	Quadrics QsNet Fat-tree	Quadrics QsNet Fat-tree
Number of Nodes		640	32	2048	1152
Processors	- per node	8	4	4	2
	- system total	5120	128	8192	2304
Processor Speed		500 MHz	800 MHz	1.25 GHz	2.4 GHz
Peak Speed	- per processor	8 Gflops	12.8 Gflops	2.5 Gflops	4.8 Gflops
	- per node	64 Gflops	51.2 Gflops	10 Gflops	9.6 Gflops
	- system total	40 Tflops	1.6 Tflops	30 Tflops	10.8 Tflops
Memory	- per node	16 GB	8-64 GB	16 GB	16 GB
	- per processor	2 GB	2-16 GB	4 GB	2 GB
	- system total	10.24 TB		48 TB	4.6 TB
Memory Bandwidth (peak)					
	- L1 Cache	N/A	76.8 GB/s	20 GB/s	20 GB/s
	- L2 Cache	N/A		13 GB/s	1.5 GB/s
	Main (per proc)	32 GB/s	34.1 GB/s	2 GB/s	2 GB/s
Inter-node MPI					
	- Latency	8.6 µsec	8.6 µsec	5 µsec	4.75 µsec
	- Bandwidth	11.8 GB/s	11.9 GB/s	300 MB/s	315 MB/s
Bytes/flop to main memory		4	3	0.8	0.4
Bytes/flop interconnect		1.5	1	0.12	0.07



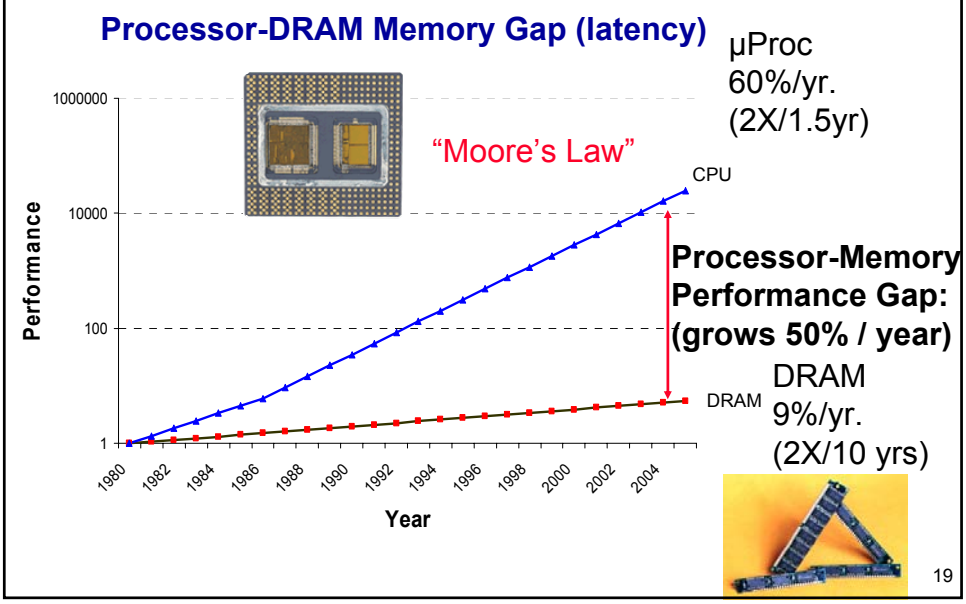
Linpack (100x100) Analysis

- Compaq 386/SX20 SX with FPA - .16 Mflop/s
- Pentium IV – 2.8 GHz – 1317 Mflop/s
- 12 years ➔ we see a factor of ~ 8231
  - Doubling in less than a year, for 12 years
- Moore's Law gives us a factor of 256 (factor of 2 18 months).
- How
  - Clock speed increase = 128x
  - External Bus Width & Caching –
    - 16 vs. 64 bits = 4x
  - Floating Point –
    - 4/8 bits multi vs. 64 bits (1 clock) = 8x
  - Compiler Technology = 2x
- However the potential for that Pentium 4 is 5.6 Gflop/s and here we are getting 1.32 Gflop/s
  - Still a factor of 4.25 off of peak

Complex set of interaction between
- Users' applications
- Algorithm
- Programming language
- Compiler
- Machine instruction
- Hardware

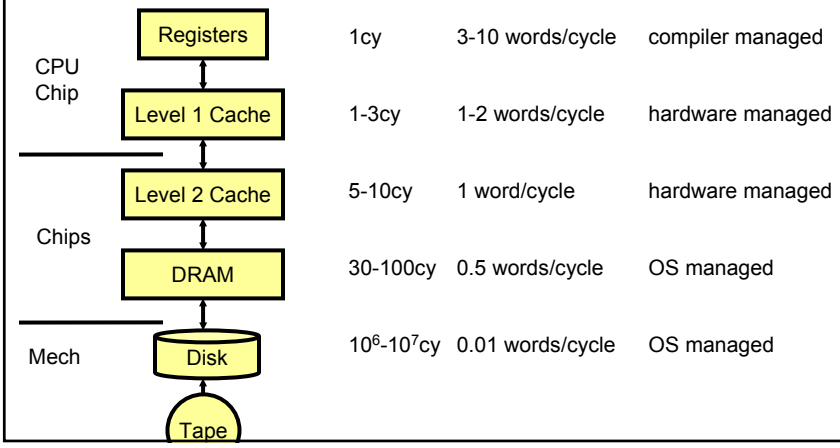
Many layers of translation from the application to the hardware
Changing with each generation

Where Does the Performance Go? or Why Should I Care About the Memory Hierarchy?



The Memory Hierarchy

- **By taking advantage of the principle of locality:**
  - **Present the user with as much memory as is available in the cheapest technology.**
  - **Provide access at the speed offered by the fastest technology.**



	1cy	3-10 words/cycle	compiler managed
	1-3cy	1-2 words/cycle	hardware managed
	5-10cy	1 word/cycle	hardware managed
	30-100cy	0.5 words/cycle	OS managed
	10^6-10^7cy	0.01 words/cycle	OS managed

Challenges in Achieving High Performance on Today's Systems

- **Diversity of execution environments**
  - Growing complexity of modern microprocessors.
    - Deep memory hierarchies
    - Out-of-order execution
    - Instruction level parallelism
  - Growing diversity of platform characteristics
    - SMPs
    - Clusters (employing a range of interconnect technologies)
    - Highly parallel systems (> 100K processors)
    - Grids (heterogeneity, wide range of characteristics)
- **Wide range of application needs**
  - Dimensionality and sizes
  - Data structures and data types
  - Languages and programming paradigms

Software Technology & Performance

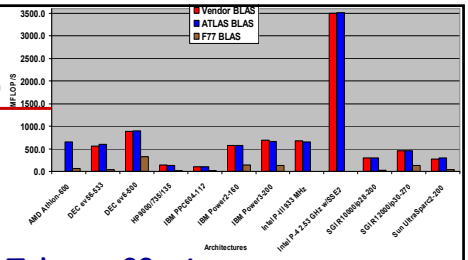
- **Tendency to focus on hardware**
- **Software required to bridge an ever widening gap**
- **Gaps between usable and deliverable performance is very steep**
  - Performance only if the data and controls are setup just right
    - Otherwise, dramatic performance degradations, very unstable situation
    - Will become more unstable
- **Challenge of Numerical Libraries, PSEs and Tools is formidable with Tflop/s level, even greater with Pflops, some might say insurmountable.**

Motivation Self Adapting Numerical Software (SANS) Effort

- Optimizing software to exploit the features of a given system has historically been an exercise in hand customization.
  - Time consuming and tedious
  - Hard to predict performance from source code
  - Must be redone for every architecture and compiler
    - Software technology often lags architecture
    - Best algorithm may depend on input, so some tuning may be needed at run-time.
  - Need for quick/dynamic deployment of optimized routines.

Software Generation Strategy - ATLAS BLAS

- Parameter study of the hw
- Generate multiple versions of code, w/difference values of key performance parameters
- Run and measure the performance for various versions
- Pick best and generate library
- Level 1 cache multiply optimizes for:
  - TLB access
  - L1 cache reuse
  - FP unit usage
  - Memory fetch
  - Register reuse
  - Loop overhead minimization

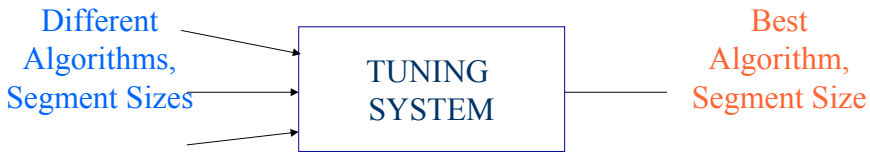


- Takes ~ 20 minutes to run, generates Level 1, 2, & 3 BLAS
- "New" model of high performance programming where critical code is machine generated using parameter optimization.
- Designed for modern architectures
  - Need reasonable C compiler
- Today ATLAS in used within various ASCI and SciDAC activities and by Matlab, Mathematica, Octave, Maple, Debian, Scyld Beowulf, SuSE,…

See: http://icl.cs.utk.edu/atlas/ for the ATLAS software

Self Adapting Numerical Software - SANS Effort

- Provide software technology to aid in high performance on commodity processors, clusters, and grids.
- Pre-run time (library building stage) and run time optimization.
- Integrated performance modeling and analysis
- Automatic algorithm selection – polyalgorithmic functions
- Automated installation process
- Can be expanded to areas such as communication software and selection of numerical algorithms

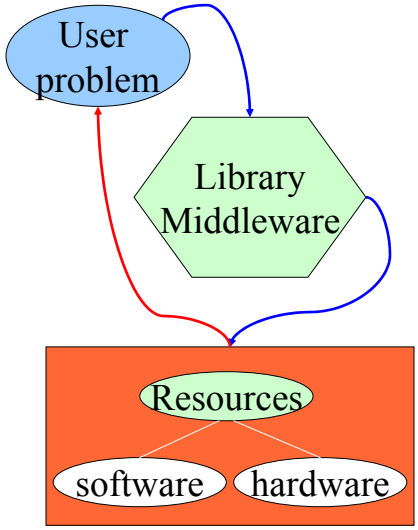


Self Adaptive Software

- Software can adapt its workings to the environment in (at least) 3 ways
  - Kernels, optimized for platform (Atlas, Sparsity): static determination
  - Scheduling, taking network conditions into account (LFC): dynamic, but data-independent
  - Algorithm choice (Salsa): dynamic, strongly dependent on user data.

Cluster Library

- Want to relieve the user of some of the tasks via Cluster Middleware
- Make decisions on the number of processors to use based on the user's problem and the state of the system
  - Optimize for the best time to solution
  - Distribute the data on the processors and collections of results
  - Start the SPMD library routine on all the platforms



LAPACK for Clusters

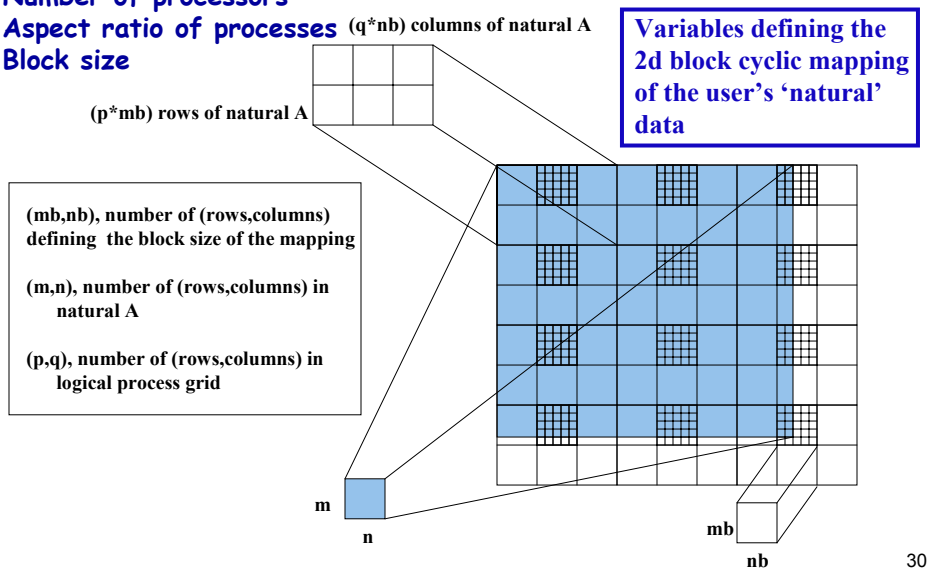
- Numerical software for dense linear algebra intended for cluster computing environments
- Descendant of LAPACK and ScaLAPACK
- Partly derived from Grid environment. GrADS (Grid Application Development Software)
- In the class of SANS (Self Adaptive Numerical Software)
- Part of the NSF NPACI's NPACkage

Cluster Numerical Library

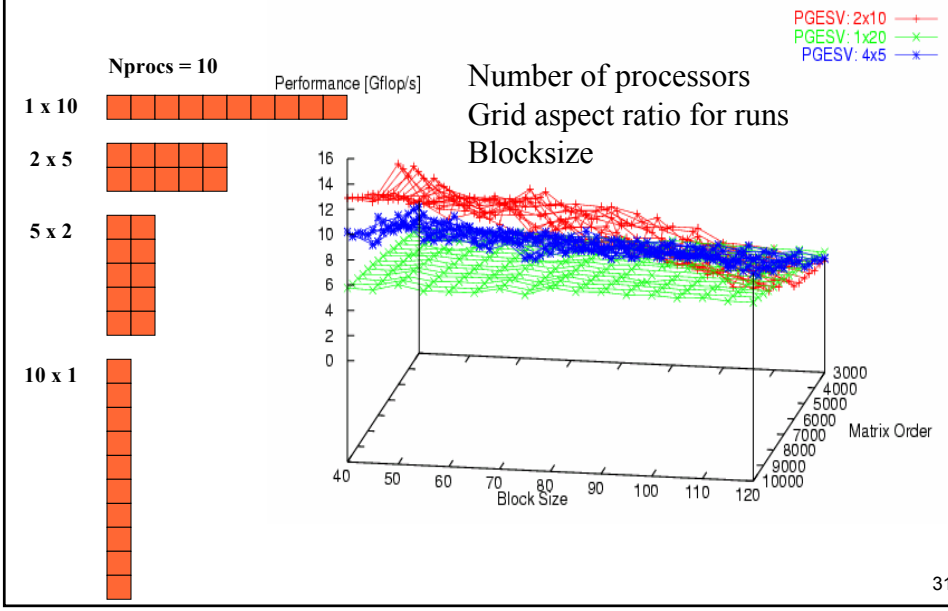
- Want to relieve the user of some of the tasks
- Make decisions on which machines to use based on the user's problem and the state of the system
  - Determinate set of procs that should be used
  - Optimize for the best time to solution
  - Distribute the data on the processors and collections of results
  - Start the SPMD library routine on all the platforms
  - Check to see if the computation is proceeding as planned
    - If not perhaps migrate application

With ScaLAPACK Data Layout Critical for Performance

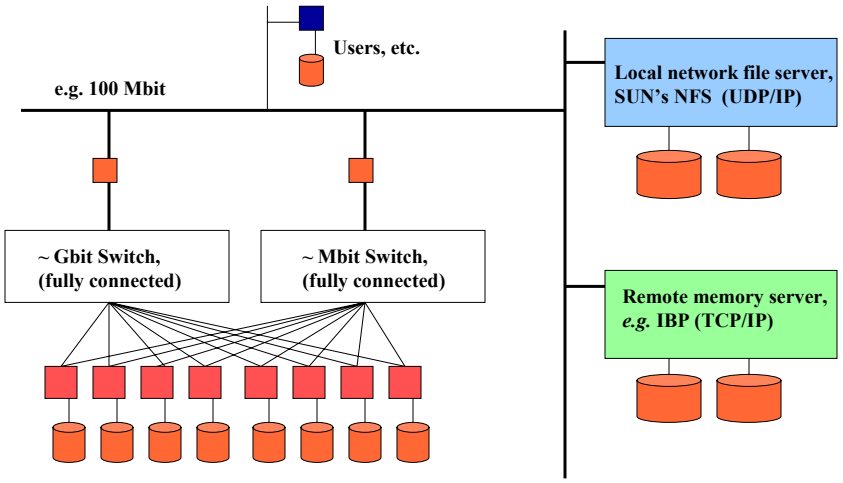
Number of processors
Aspect ratio of processes
Block size



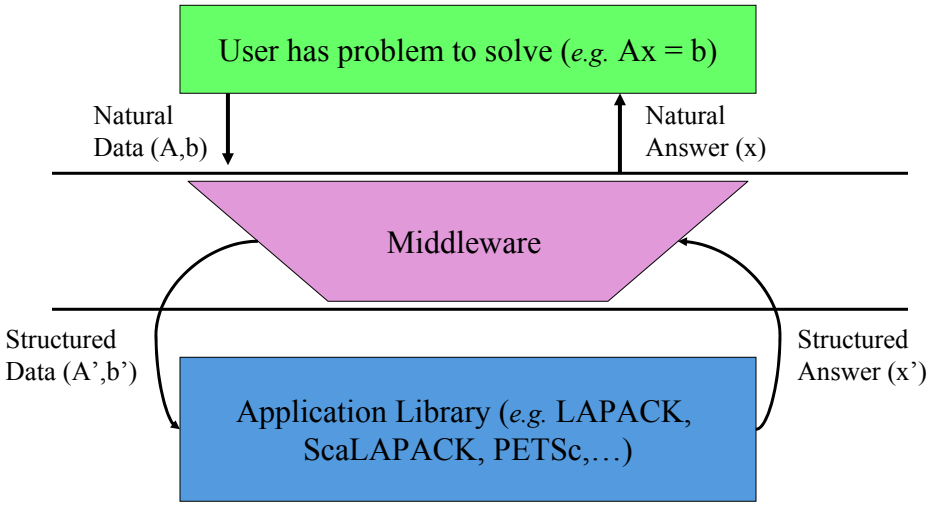
Needs An Expert To Do The Tuning



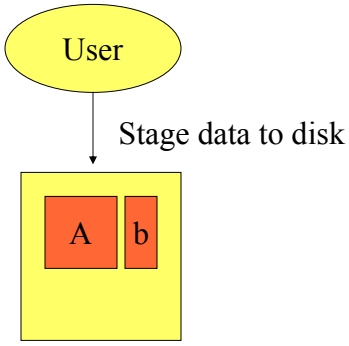
LFC Sample Computing Environment:



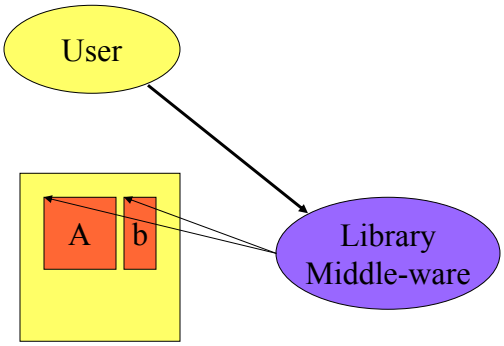
User Interface/Middleware



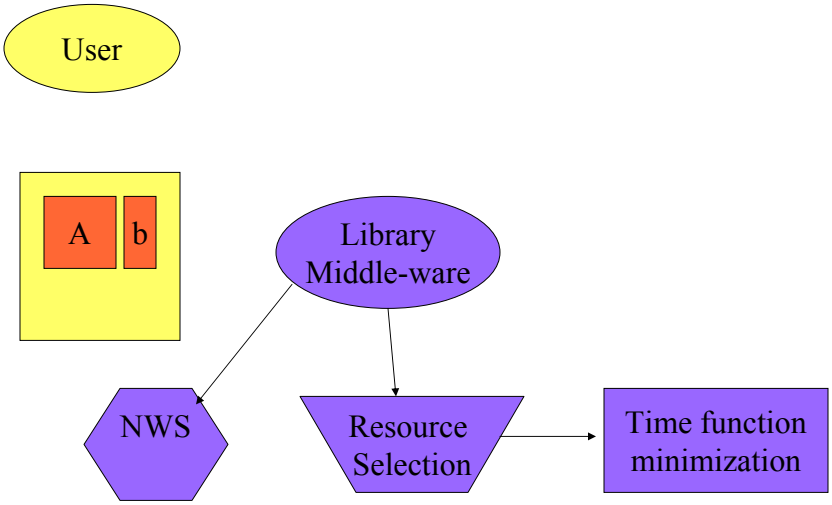
File System -based



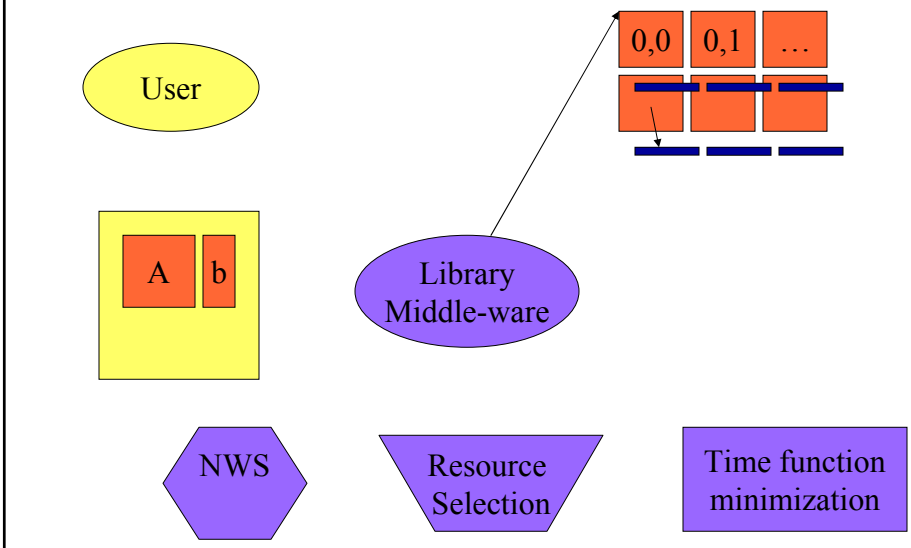
File System -based



File System -based

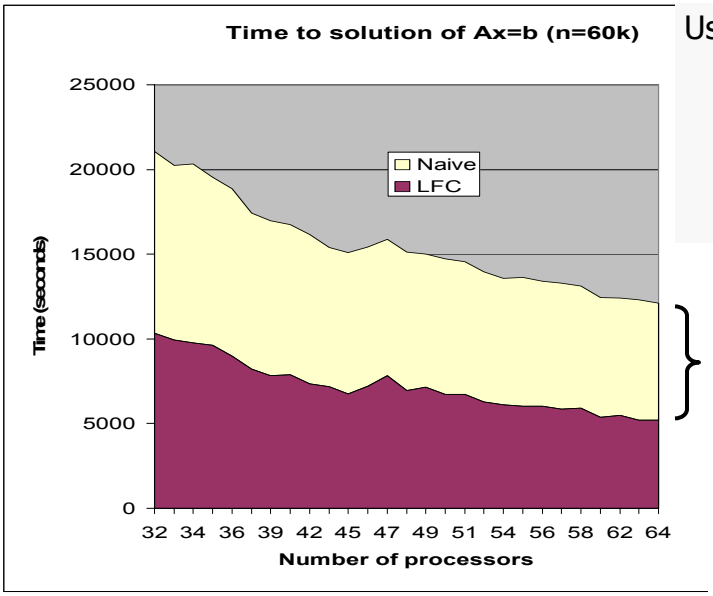


File System -based



Has been applied to Grid infrastructure, i.e.Globus/NWS, but doesn't have to.

LFC Performance Results



Using up to 64 of AMD 1.4 GHz processors at Ohio Supercomputer Center

Increasing margin of potential user error

Executing Matlab Programs on a Cluster

> mpirun -np 128 lfc_server port=35000 &

> Matlab

Cluster

```
server_connect(35000);
A = lfc_fread(…);
b = lfc_fread(…);
x = A \ b;
r = b – A * x;
z = A \ r;
x = x + z;
…
```

- **Arrays will live on the server and execution takes place there via LFC / ScaLAPACK.**
- **Debug on laptop, run on cluster**

Plans for Python, Mathematica, Maple … as well

Grids vs. Capability vs. Cluster Computing

- **Not an "either/or" question**
  - Each addresses different needs
  - Each are part of an integrated solution
- **Grid strengths**
  - Coupling necessarily distributed resources
    - instruments, software, hardware, archives, and people
  - Eliminating time and space barriers
    - remote resource access and capacity computing
  - Grids are not a cheap substitute for capability HPC
- **Capability computing strengths**
  - Supporting foundational computations
    - terascale and petascale "nation scale" problems
  - Engaging tightly coupled computations and teams
- **Clusters**
  - Low cost, group solution
  - Potential hidden costs

Collaborators / Support

- TOP500
  - H. Meuer, Mannheim U
  - H. Simon, NERSC
  - E. Strohmaier, NERSC
- SANS
  - Kenny Roche, UTK
  - Piotr Luszczek, UTK
  - Jeffery Chen, UTK
  - Victor Eijkhout, UTK
  - Antoine Petitet, Sun Micro
  - Clint Whaley, U of Florida