Seminario EEUU/Venezuela de Computación de Alto Rendimiento 2000
US/Venezuela Workshop on High Performance Computing 2000 - WoHPC 2000
4 al 6 de Abril del 2000
Puerto La Cruz Venezuela

Trends in High Performance Computing, Enhancing Performance and the Computational Grid

Jack Dongarra
University of Tennessee
and
Oak Ridge National Laboratory
http://www.cs.utk.edu/~dongarra/

Outline

- Look at trends in HPC
  - Top500 statistics
- NetSolve
  - Example of grid middleware
- Performance on today's architecture
  - ATLAS effort
- Tools for performance evaluation
  - Performance API (PAPI)

Background Information

- Started in 6/93 by JD,Hans W. Meuer and Erich Strohmaier

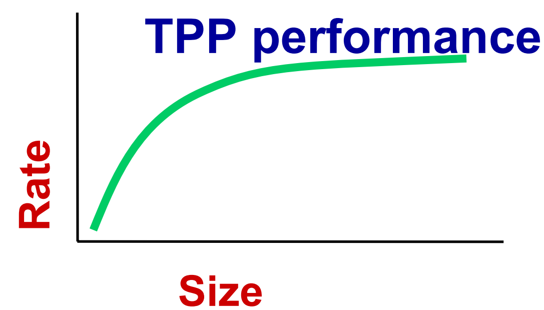
TOP500 Motivation

- Basis for analyzing the HCP market
- Quantify observations
- Detection of trends (market, architecture, technology)

TOP500 Procedure

- 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 November
  Meeting in Mannheim in June
- All data available from **www.top500.org**

TOP-500 List

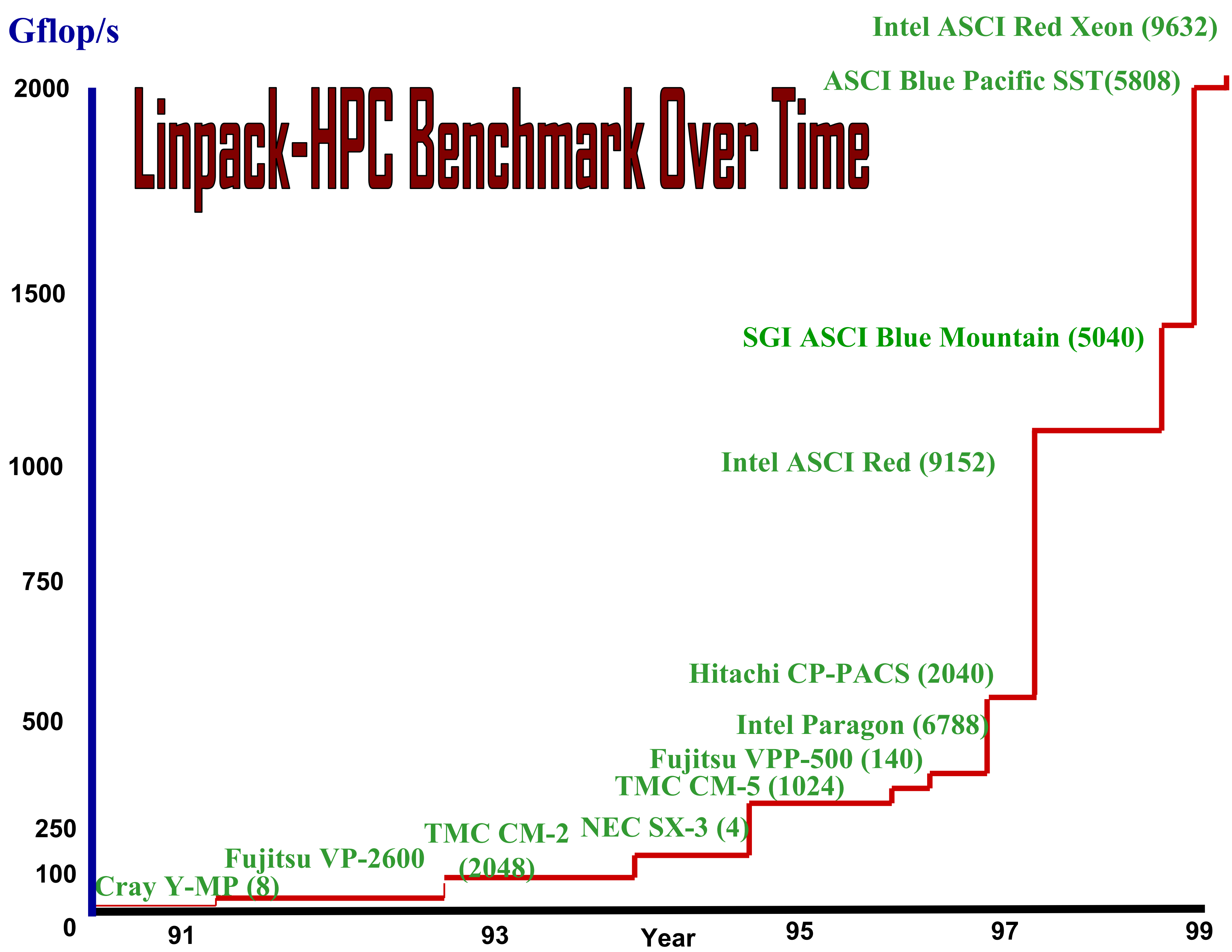
- A way for tracking trends
  - in performance
  - in market
  - in classes of HPC systems
    - Architecture
    - Technology
- Original classes of machines
  - Sequential
  - SMPs
  - MPPs
  - SIMDs
- Two new classes
  - Beowulf-class systems
  - Clustering of SMPs and DSMs
    - Requires additional terminology
      - "Constellation"

"Constellation" Cluster of Clusters

- An ensemble of N nodes each comprising p computing elements
- The p elements are tightly bound shared memory (e.g. smp, dsm)
- The N nodes are loosely coupled, i.e.: distributed memory
- p is greater than N
- Distinction is which layer gives us the most power through parallelism

4TF Blue Pacific SST

3 x 480 4-way SMP nodes
3.9 TF peak performance
2.6 TB memory
2.5 Tb/s bisectional bandwidth
62 TB disk
6.4 GB/s delivered I/O bandwidth

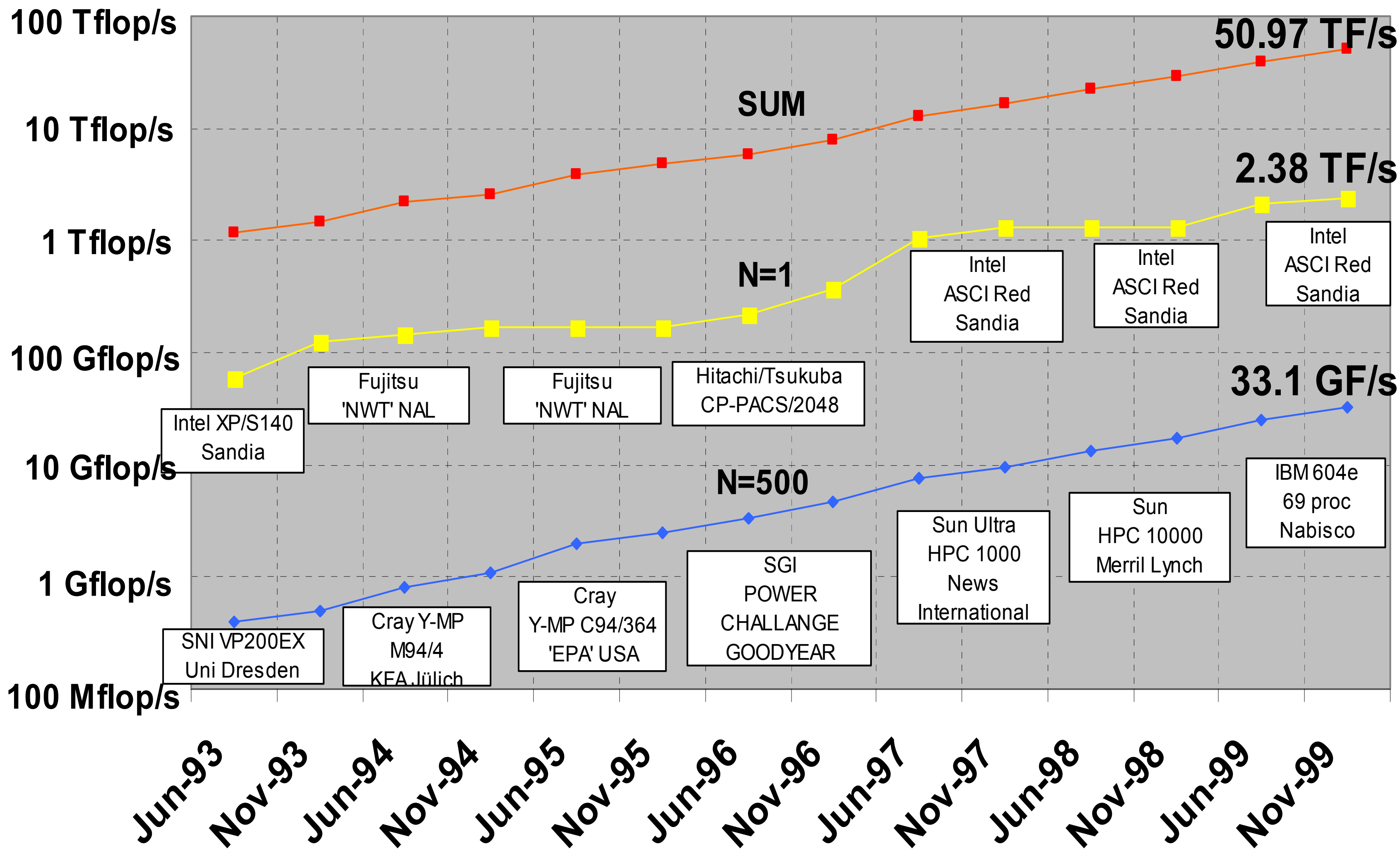


Gflop/s
Linpack-HPC Benchmark Over Time
2000
1500
1000
750
500
250
100
0
91
93
Year
95
97
99
Intel ASCI Red Xeon (9632)
ASCI Blue Pacific SST(5808)
SGI ASCI Blue Mountain (5040)
Intel ASCI Red (9152)
Hitachi CP-PACS (2040)
Intel Paragon (6788)
Fujitsu VPP-500 (140)
TMC CM-5 (1024)
NEC SX-3 (4)
TMC CM-2
(2048)
Fujitsu VP-2600
Cray Y-MP (8)

TOP10 11/99

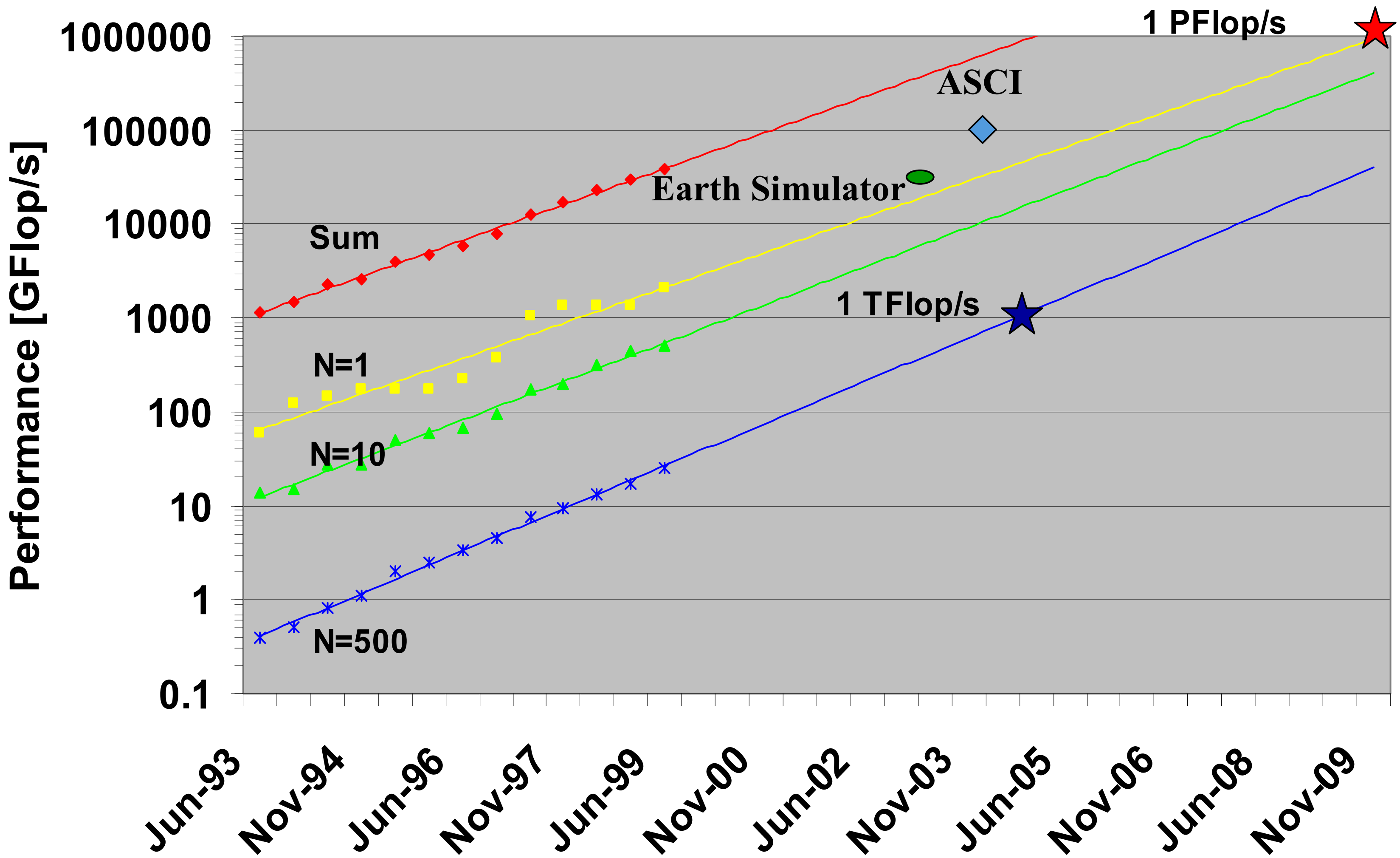
RANK	MANU-FACTURER	COMPUTER	RMAX [GF/S]	INSTALLATION SITE	COUNTRY	YEAR	AREA OF INSTALLATION	# PROC
1	Intel	ASCI Red	2379.6	Sandia National Labs Albuquerque	USA	1999	Research	9632
2	IBM	ASCI Blue-Pacific SST, IBM SP 604E	2144	Lawrence Livermore National Laboratory	USA	1999	Research	5808
3	SGI	ASCI Blue Mountain	1608	Los Alamos National Lab	USA	1998	Research	6144
4	SGI	T3E 1200	891.5	Government	USA	1998	Classified	1084
5	Hitachi	SR8000	873.6	University of Tokyo	Japan	1999	Academic	128
6	SGI	T3E 900	815.1	Government	USA	1997	Classified	1324
7	SGI	Orgin 2000	690.9	Los Alamos National Lab /ACL	USA	1999	Research	2048
8	Cray/SGI	T3E 900	675.7	Naval Oceanographic Office, Bay Saint Louis	USA	1999	Research Weather	1084
9	SGI	T3E 1200	671.2	Deutscher Wetterdienst	Germany	1999	Research Weather	812
10	IBM	SP Power3	558.13	UCSD/San Diego Supercomputer Center, IBM/Poughkeepsie	USA	1999	Research	1024

Performance Development



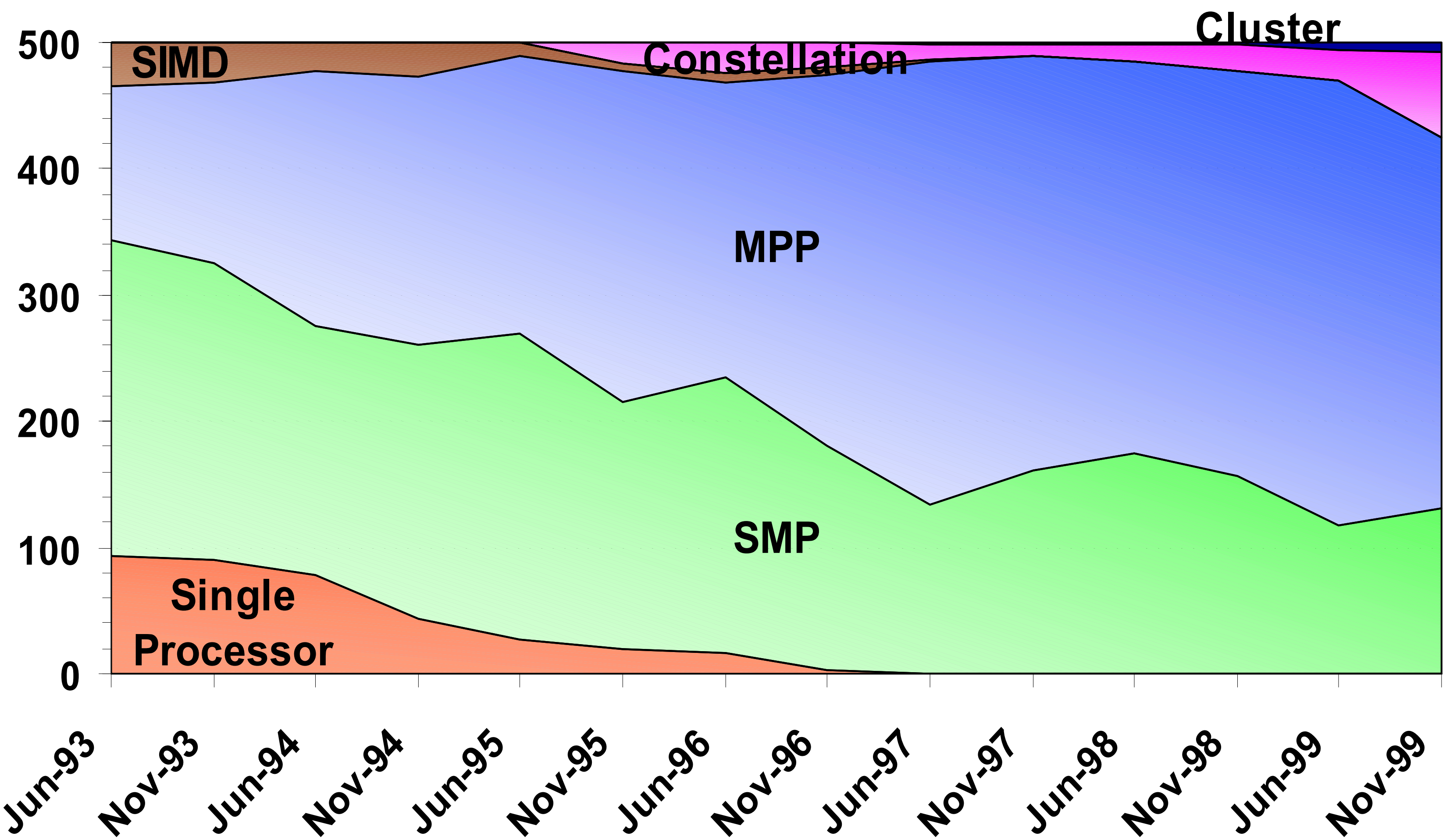
[60G - 400 M][2.4 Tflop/s 33 Gflop/s],[1 40 Tf] Schwab #12, 1/2 each year, 100 > 100 Gf, faster than Moore’s law

Performance Development

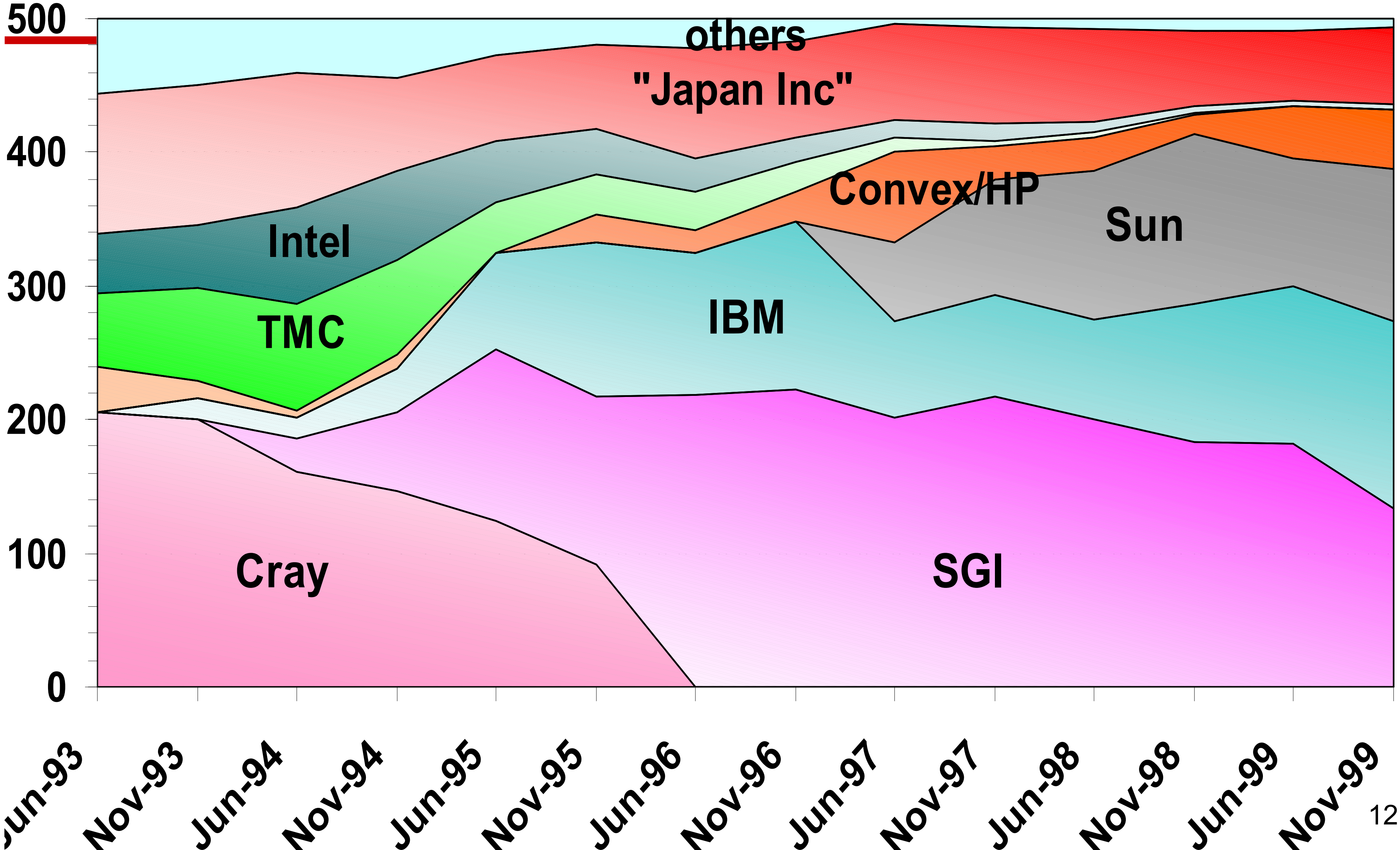


Entry 1 T 2005 and 1 P 2010

Architectures

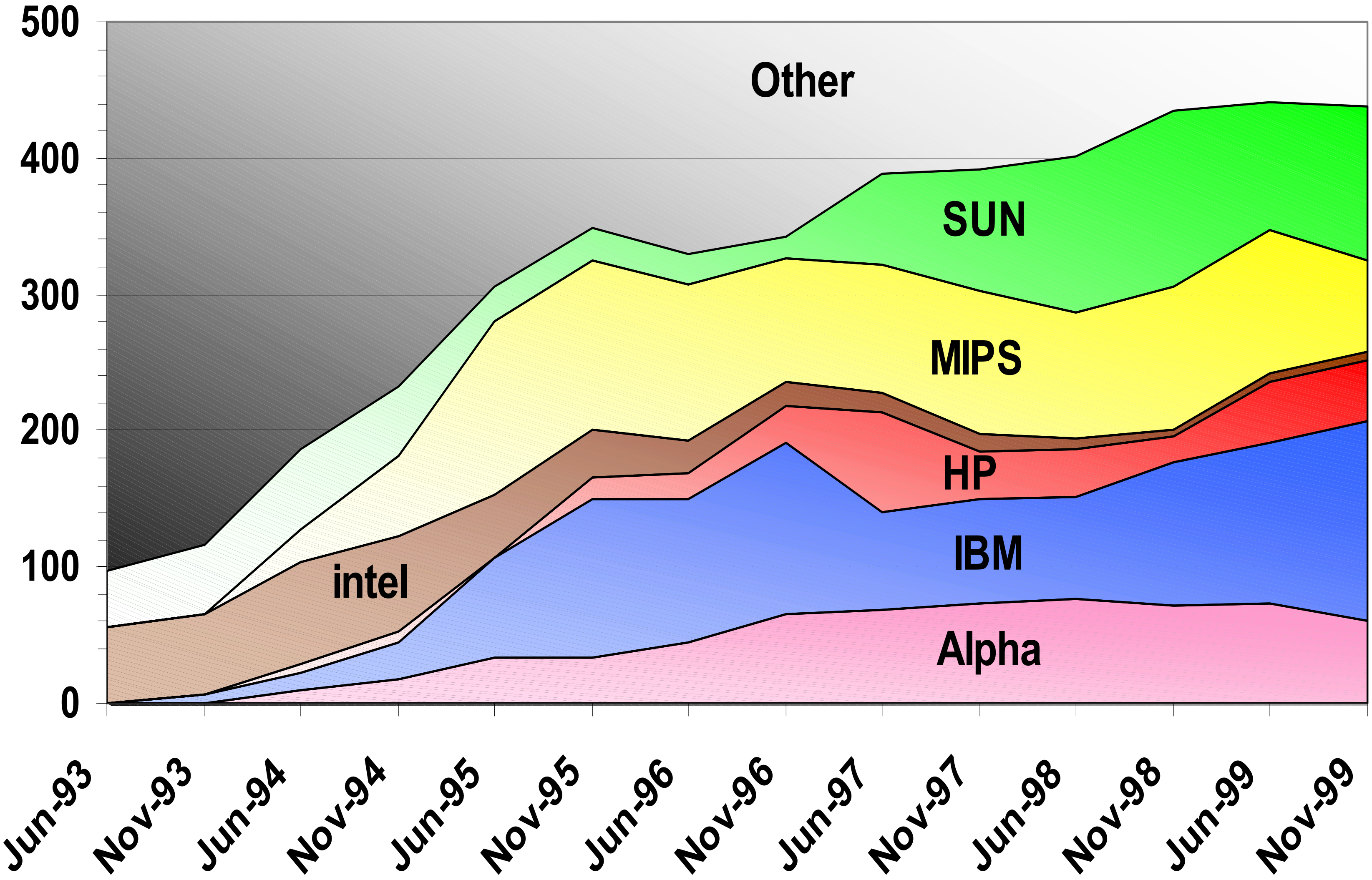


Manufacturer

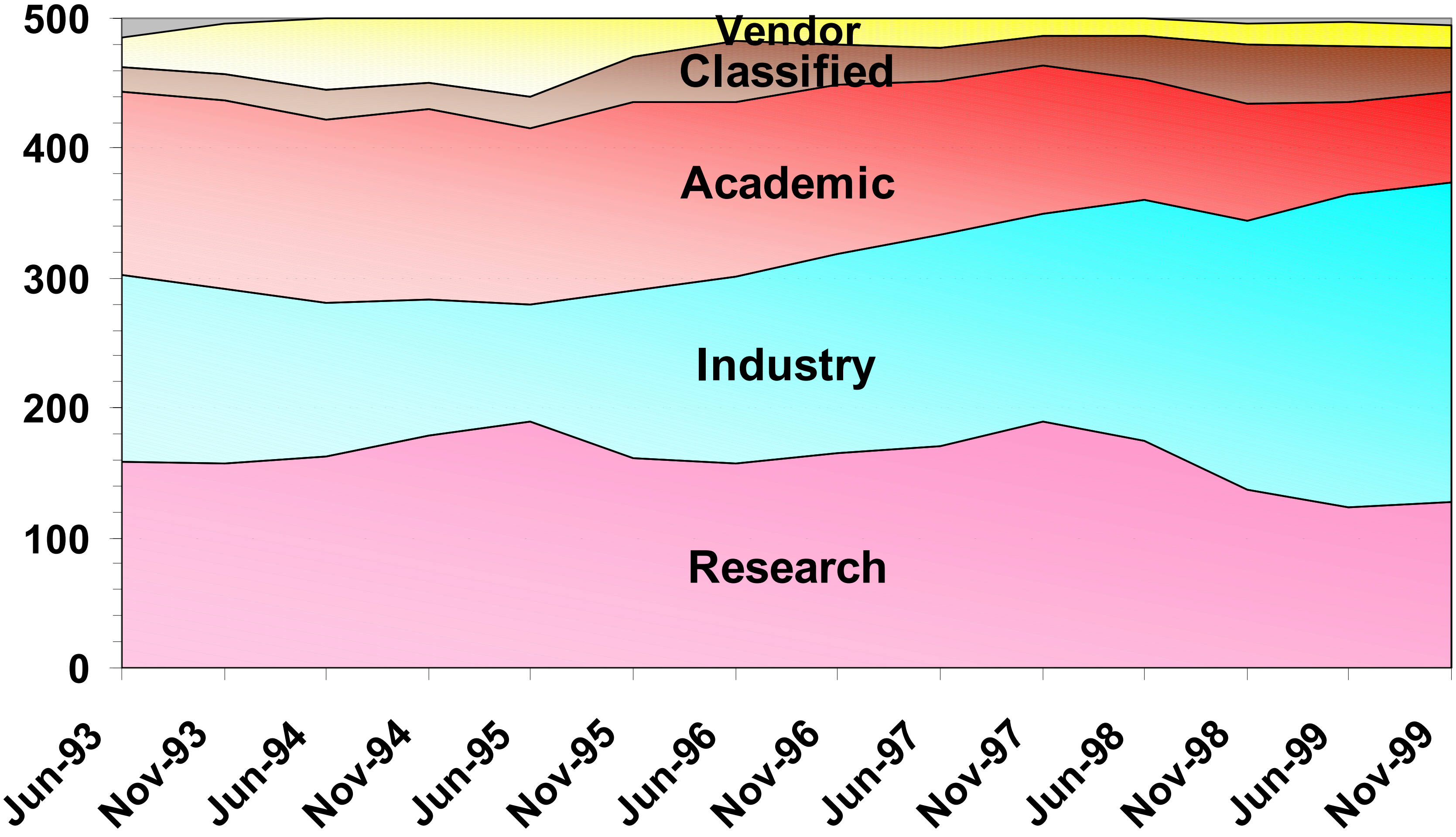


Tara -> Cray Inc $15M

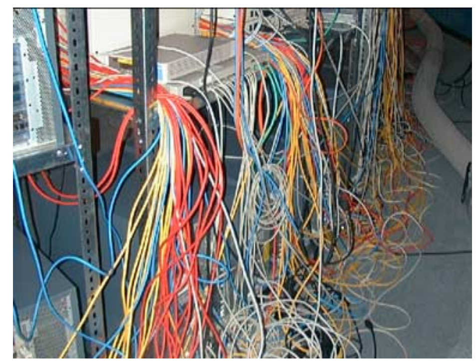
Chip Technology



Customer Type



High-Performance Computing Directions



- Clustering of shared memory machines for scalability
  - Emergence of PC commodity systems
    - Pentium/Alpha based, Linux or NT driven
    - "Supercomputer performance at mail-order prices"
  - Beowulf-Class Systems (Linux+PC)
  - Distributed Shared Memory (clusters of processors connected)
  - Shared address space w/deep memory hierarchy
- Efficiency of message passing and data parallel programming
  - Helped by standards efforts such as PVM, MPI, Open-MP and HPF
- Many of the machines as a single user environments
- Pure COTS

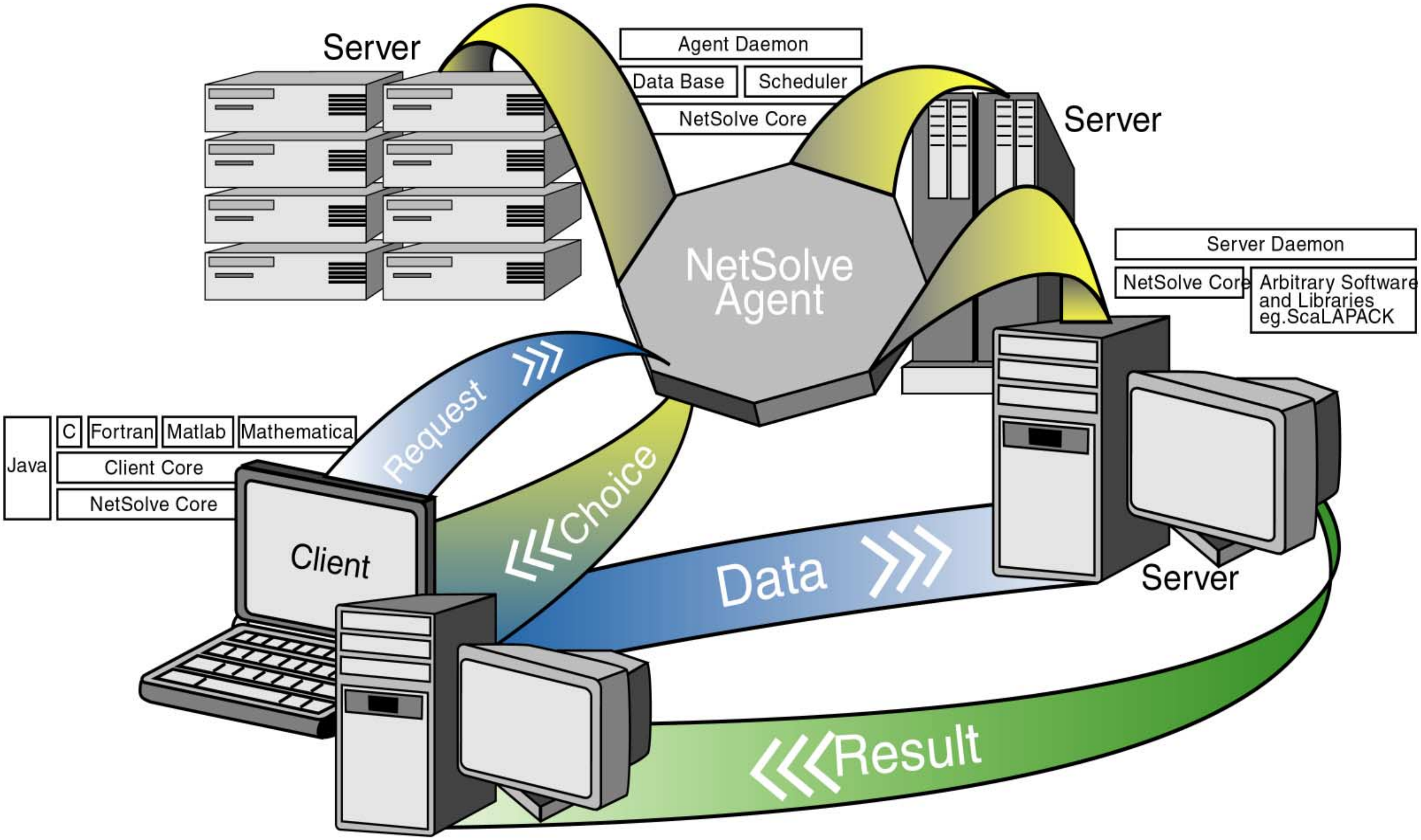
Clusters on the TOP500

RANK	MANU-FACTURER	COMPUTER	RMAX	INSTALLATION SITE	COUNTRY	YEAR	AREA OF INSTALLATION	# PROC
33	Sun	HPC 450 Cluster	272.1	Sun, Burlington	USA	1999	Vendor	720
34	Compaq	Alpha Server SC	271.4	Compaq Computer Corp. Littleton	USA	1999	Vendor	512
…	…	…	…	…	…	…	…	…
44	Self-made	Cplant Cluster	232.6	Sandia National Laboratories	USA	1999	Research	580
…	…	…	…	…	…	…	…	…
169	Self-made	Alphleet Cluster	61.3	Institute of Physical and Chemical Res. (RIKEN)	Japan	1999	Research	140
…	…	…	…	…	…	…	…	…
265	Self-made	Avalon Cluster	48.6	Los Alamos National Lab/ CNLS	USA	1998	Research	140
…	…	…	…	…	…	…	…	…
351	Siemens	hpcLine Cluster	41.45	Universitaet Paderborn/PC2	Germany	1999	Academic	192
…	…	…	…	…	…	…	…	…
454	Self-made	Parnass2 Cluster	34.23	University of Bonn/ Applied Mathematic	Germany	1999	Academic	128

NetSolve - Network Enabled Servers

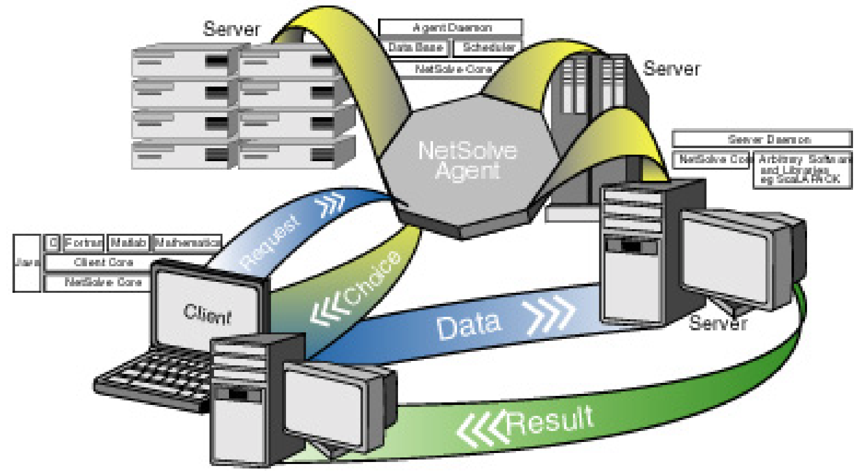
- Allow networked resources to be integrated into the desktop.
- Not just hardware, but also make available software resources.
- Locate and "deliver" software or solutions to the user in a directly usable and "conventional" form.
- Part of the motivation - software maintenance

NetSolve



Server
Agent Daemon
Data Base
Scheduler
NetSolve Core
Server
NetSolve
Agent
Server Daemon
NetSolve Core
Arbitrary Software
and Libraries
eg.ScaLAPACK
Java
C
Fortran
Matlab
Mathematica
Client Core
NetSolve Core
Request
Choice
Client
Data
Server
Result

NetSolve



- **Three basic scenarios:**
  - Client, servers and agents anywhere on Internet (3(10)-150(80-ws/mpp)-Mcell)
  - Client, servers and agents on an Intranet
  - Client, server and agent on the same machine
- **"Blue Collar" Grid Based Computing**
  - User can set things up, no "su" required
  - Doesn't require deep knowledge of network programming
- **Focus on Matlab users**
  - OO language, objects are matrices (pse, eg os)
  - One of the most popular desktop systems for numerical computing, 400K Users

NetSolve - The Client

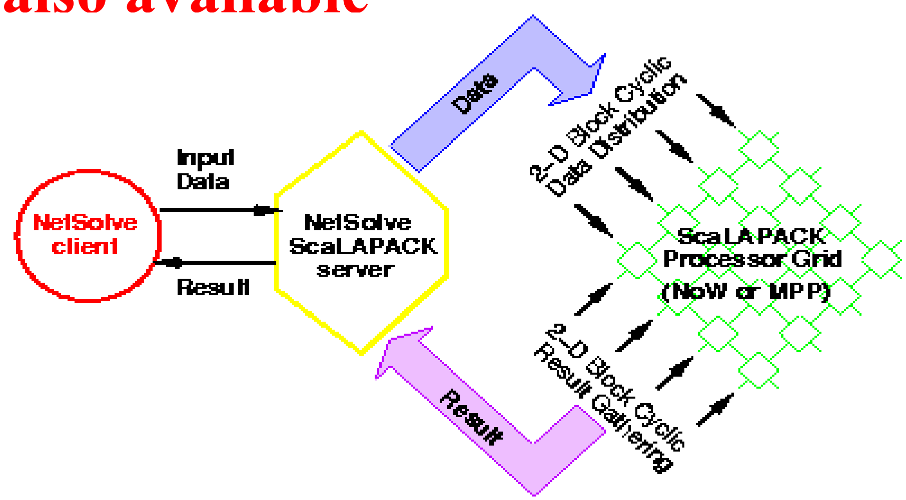
- No knowledge of networking involved
- Hide complexity of numerical software
- Computation **location transparency**
- Provides access to **Virtual Libraries** :
  - **Component grid-based framework**
  - **Central management of library resources**
  - **User not concerned with most up-to-date version**
  - **Automatic tie to Netlib repository in project**
- Provides synchronous or asynchronous calls
  (User level parallelism)

NetSolve - Interface

>> define sparse matrix A
>> define rhs
>> [x, its] = netsolve('itmeth','petsc', A, rhs, 1.e-6
...

call NETSL('DGESV()',NSINFO,
N,1,A,MAX,IPIV,B,MAX,INFO)

Asynchronous Calls also available



NetSolve - The Server Side

Computational Server :

- Various **Software** resources installed on various **Hardware** Resources
- **Configurable and Extensible :**
  - Framework to easily add arbitrary software ...
  - Many numerical libraries being integrated by the NetSolve team
  - Many software being integrated by users

Agent :

- Gateway to the computational servers
- Performs **Load Balancing** among the resources

http://www.cs.utk.edu/netsolve

NetSolve - Load Balancing

NetSolve agent :

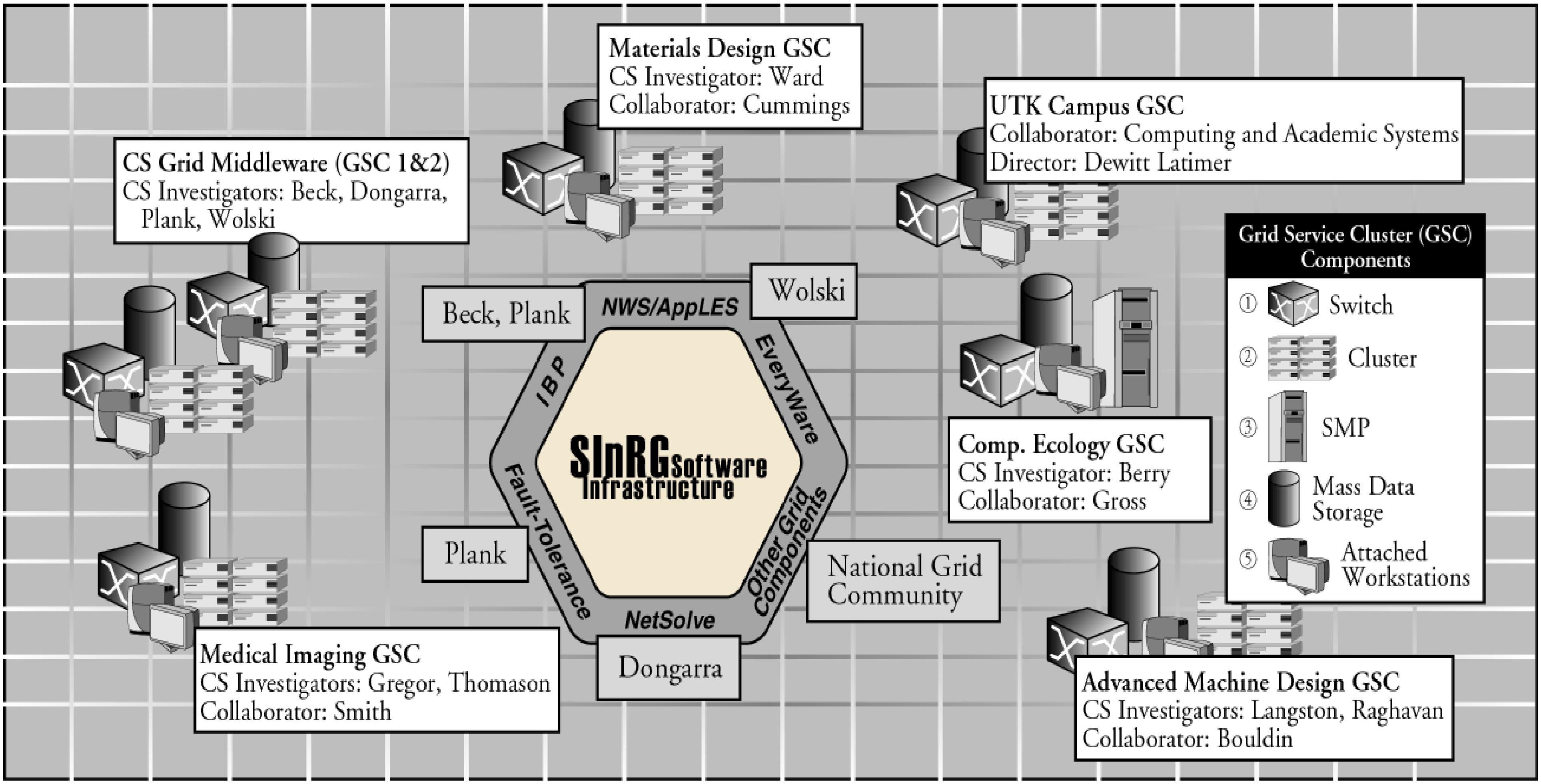
predicts the execution times and sorts the servers

Prediction for a server based on :

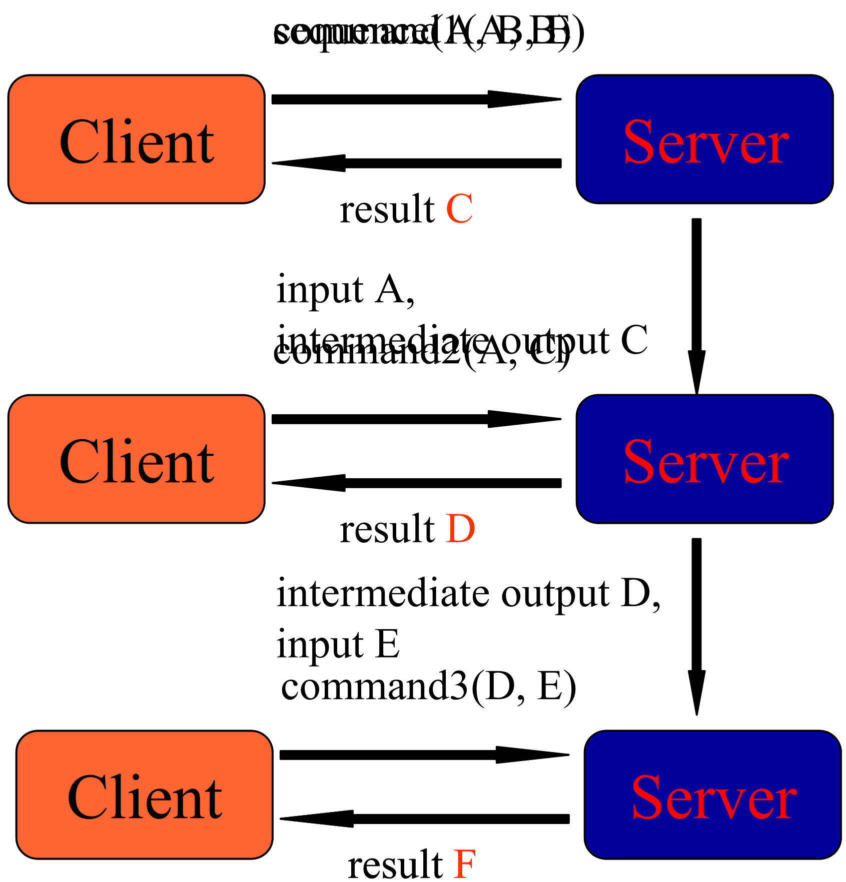
- **Its distance over the network**
  - **- Latency and Bandwidth**
  - **- Statistical Averaging**
- **Its performance (LINPACK benchmark)**
- **Its workload**
- **The problem size and the algorithm complexity**

Cached data → **Quick estimate** / **workload out of date ?**

University of Tennessee's Grid Prototype: Scalable Intracampus Research Grid: SInRG

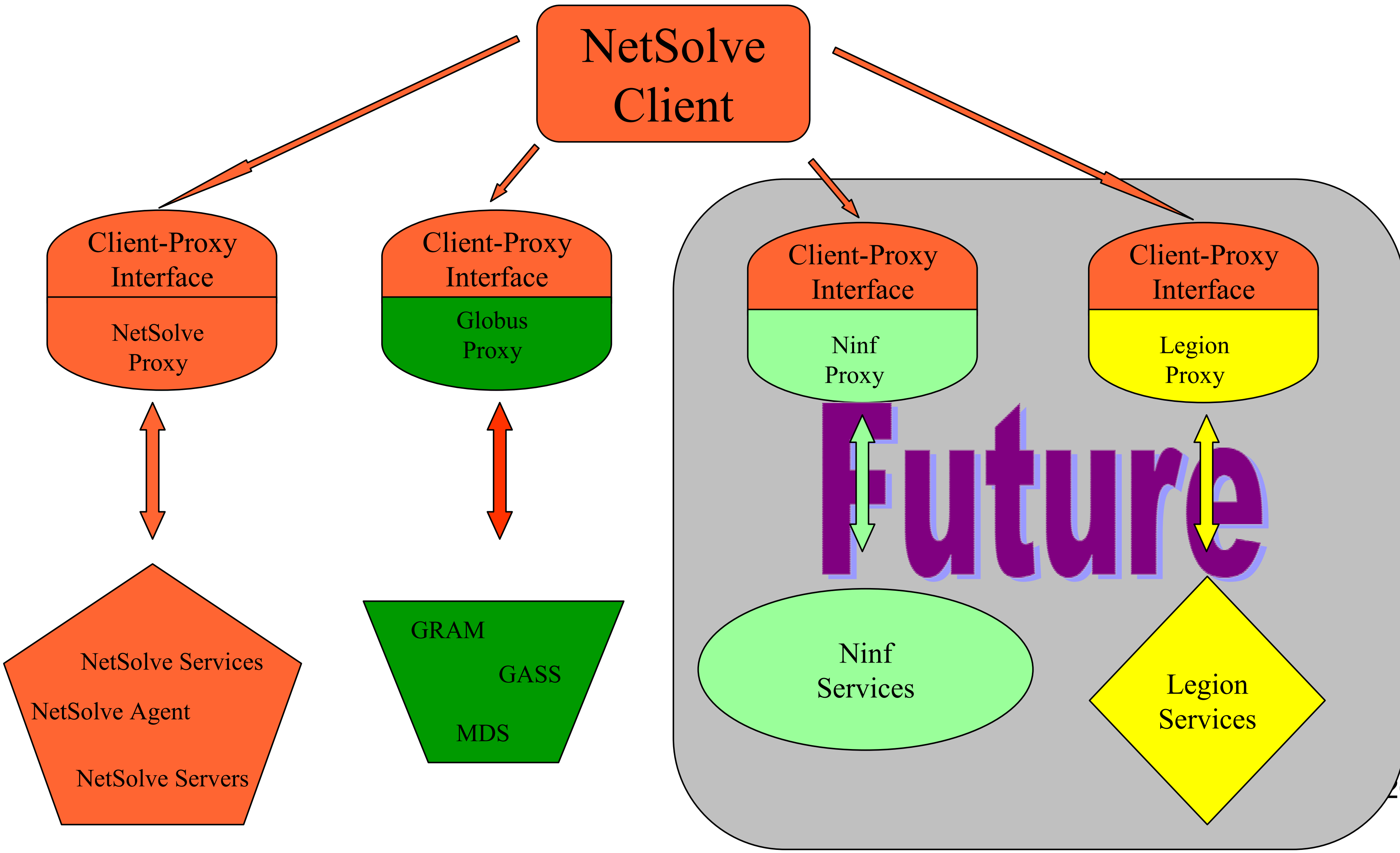


Data Persistence (cont'd)

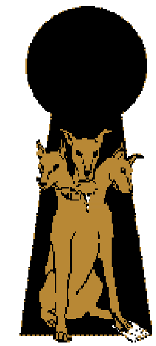



```
netsl_begin_sequence( );
netsl("command1", A, B, C);
netsl("command2", A, C, D);
netsl("command3", D, E, F);
netsl_end_sequence(C, D);
```

Developing Client Proxies Interfaces



NetSolve Authentication with Kerberos

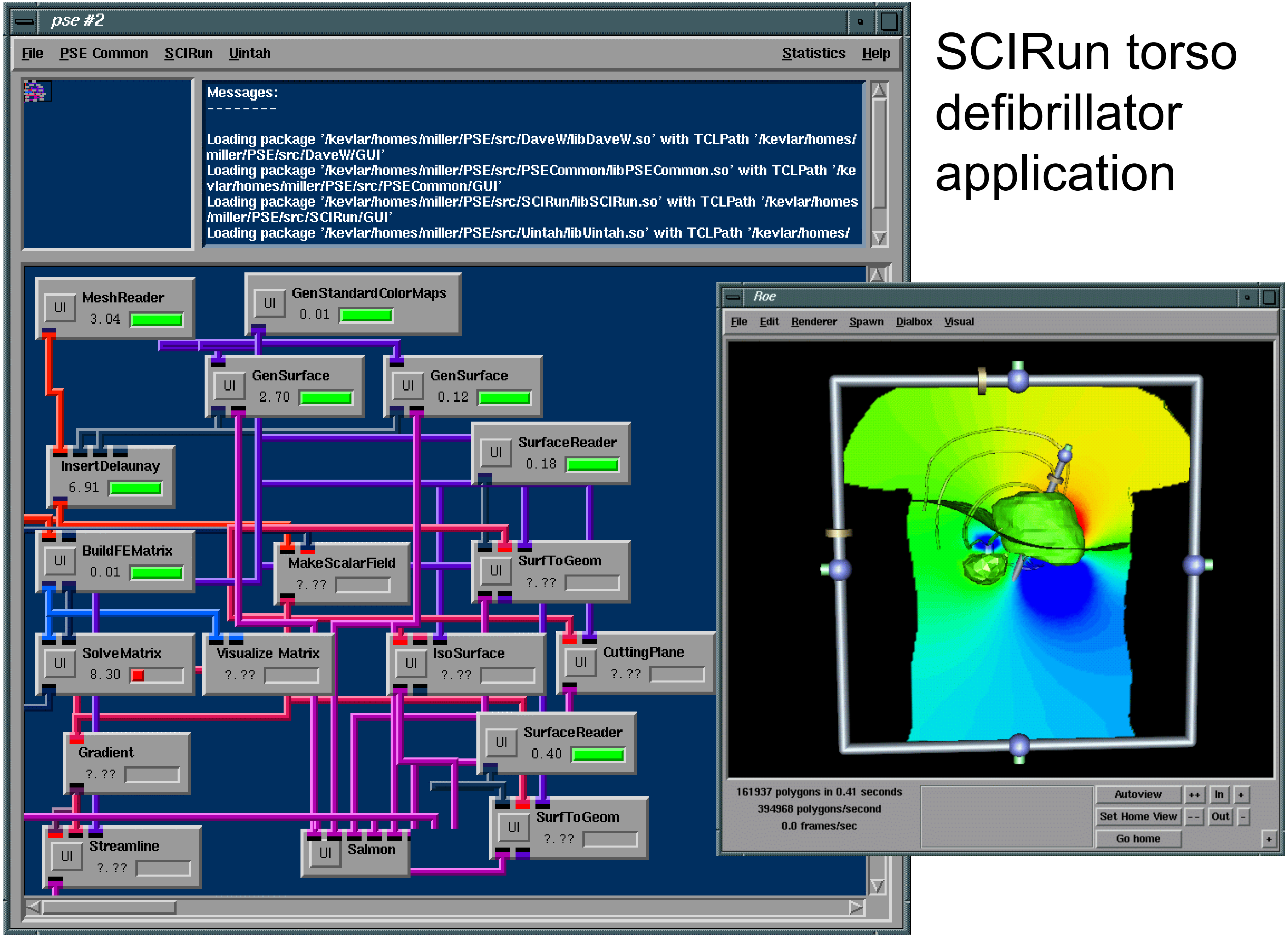


- Kerberos used to maintain Access Control Lists and manage access to computational resources.
- NetSolve properly handles authorized and non-authorized components together in the same system.
- In use by DOD Modernization program at Army Research Lab

Task Farming

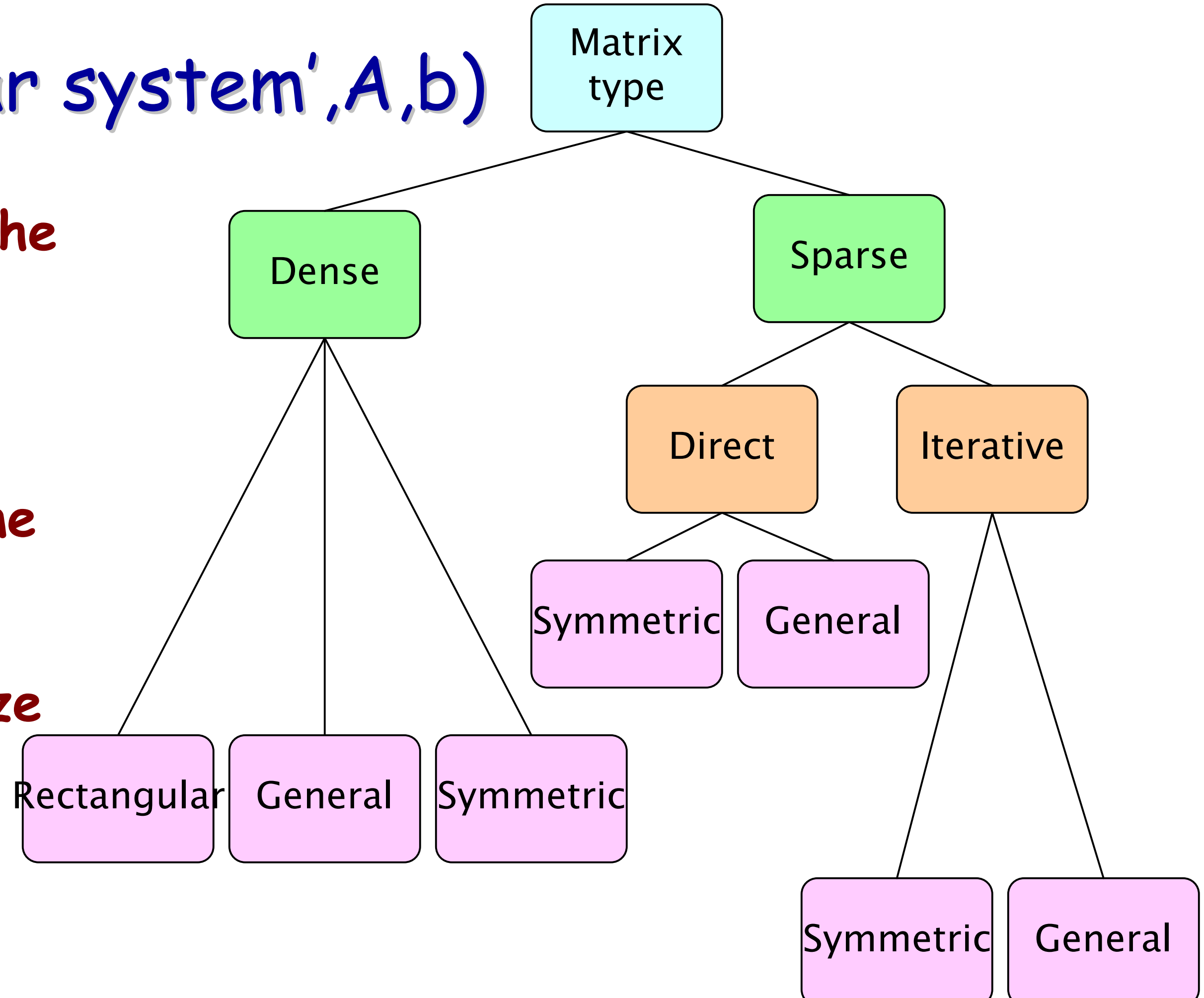
- **Multiple requests to single problem.**
- **Previous Solution:**
  - Many calls to netslnb( ); /* non-blocking */
- **New Solution:**
  - **Single call to netsl_farm( );**

SCIRun torso defibrillator application

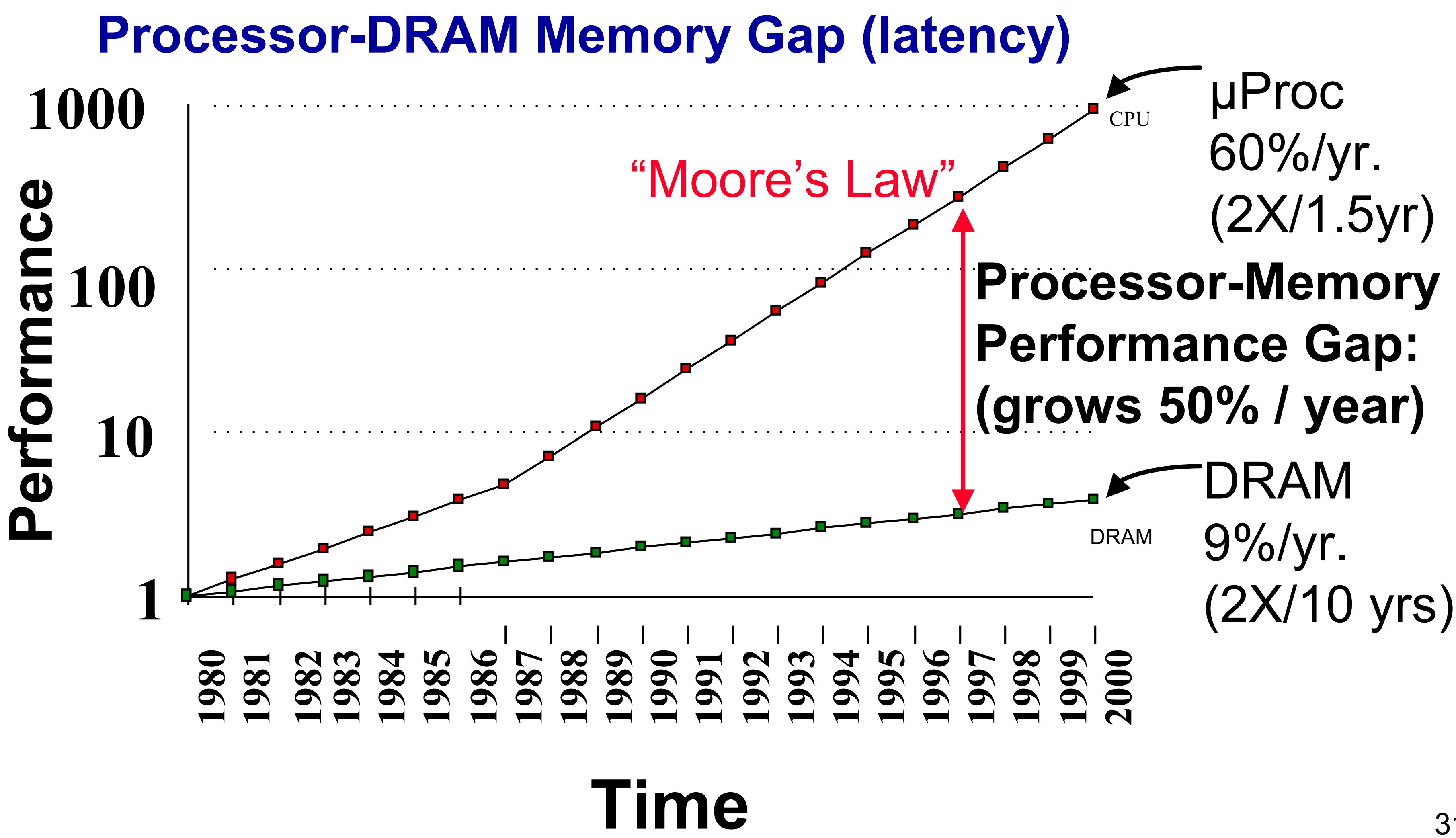


NetSolve to Determine the Best Algorithm/Software (A bit in the future)

- x = netsolve('linear system',A,b)
  - NetSolve examines the user's data together with the resources available (in the grid sense) and makes decisions on best time to solution dynamically.
  - Decision based on size of problem, hardware/software, network connection etc.
  - Method and Placement.

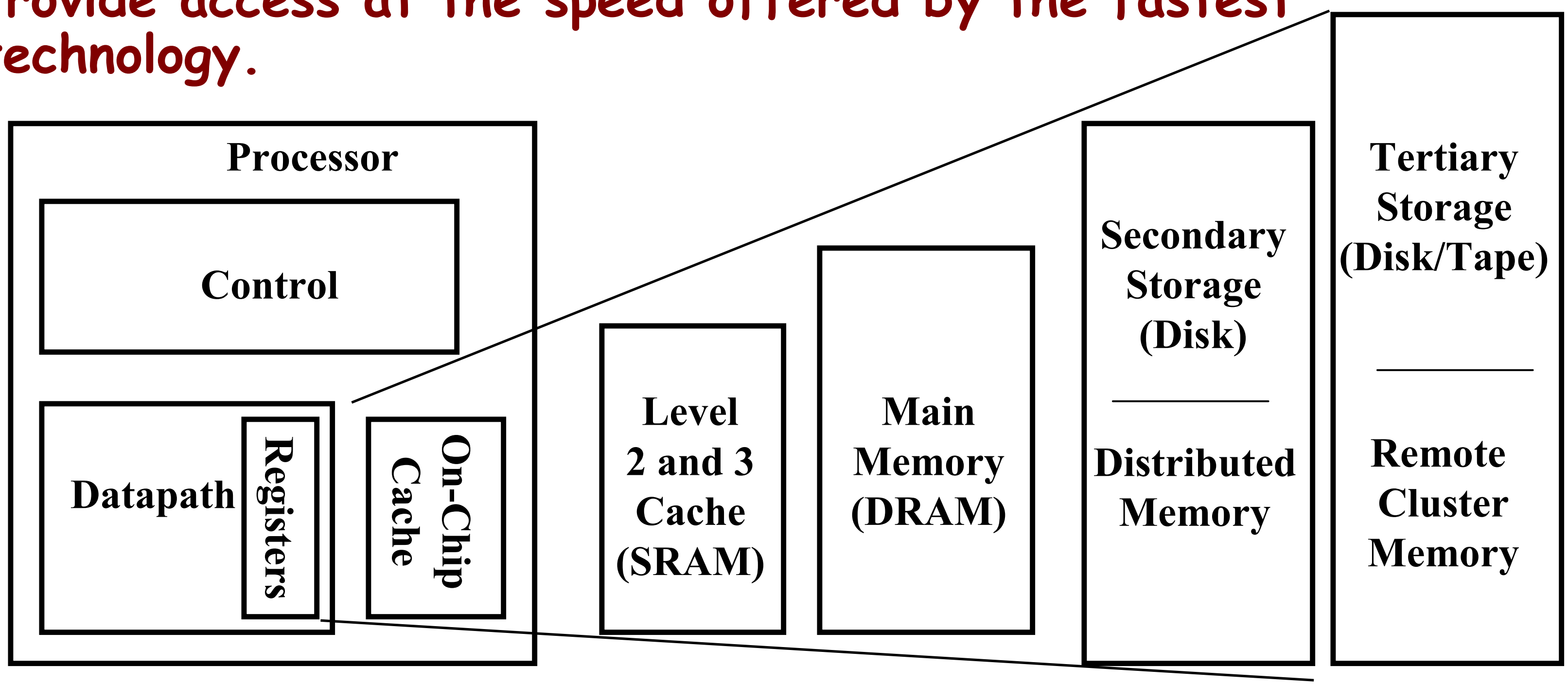


Where Does the Performance Go? or Why Should I Cares About the Memory Hierarchy?



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.



Speed (ns):	1s	10s	100s	10,000,000s (10s ms) 100,000 s (.1s ms)	10,000,000,000s (10s sec) 10,000,000 s (10s ms)
Size (bytes):	100s	Ks	Ms	Gs	Ts

How To Get Performance From Commodity Processors?

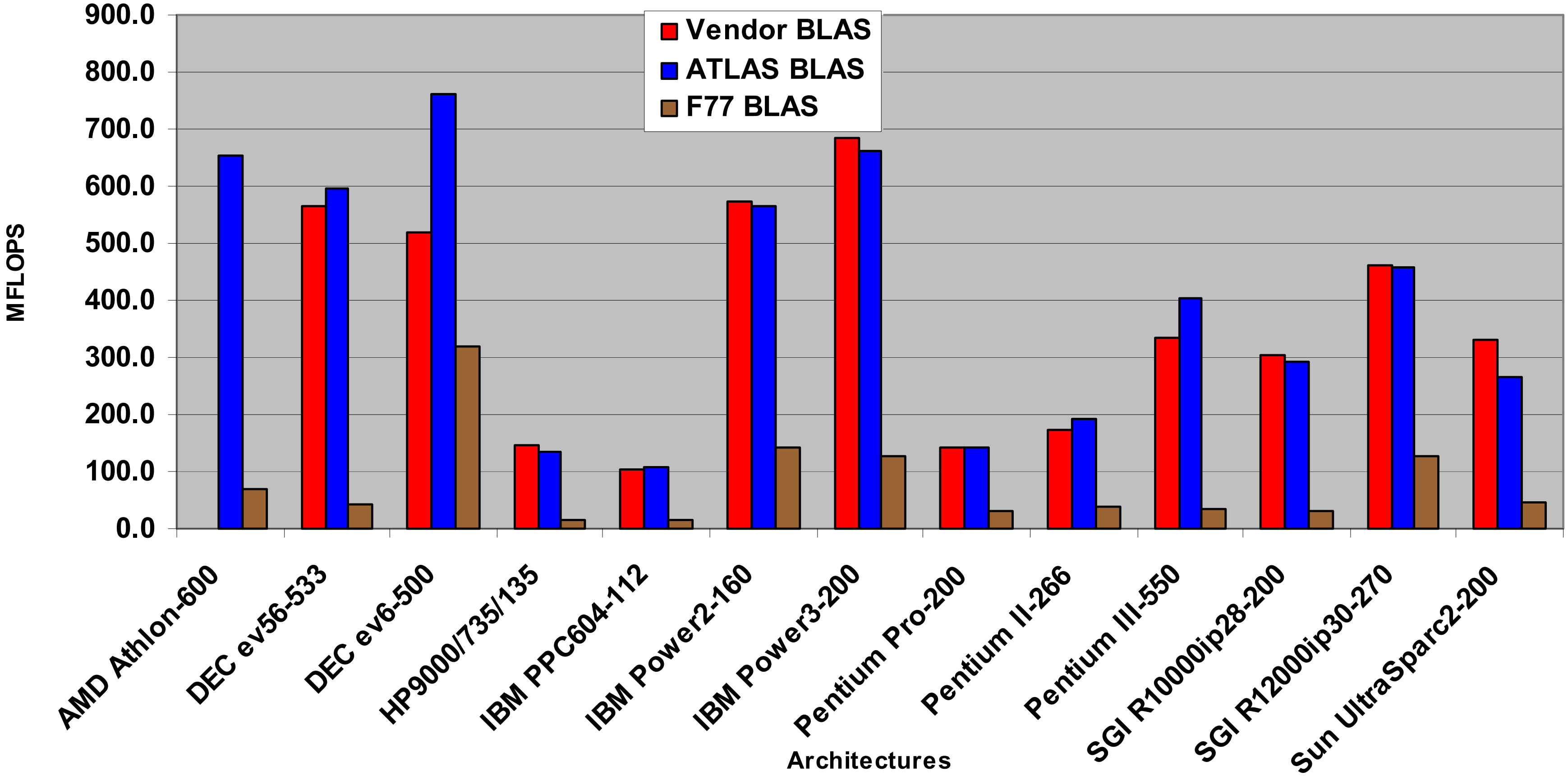
- Today's processors can achieve high-performance, but this requires extensive machine-specific hand tuning.
- Hardware and software have a large design space w/many parameters
  - Blocking sizes, loop nesting permutations, loop unrolling depths, software pipelining strategies, register allocations, and instruction schedules.
  - Complicated interactions with the increasingly sophisticated micro-architectures of new microprocessors.
- About a year ago no tuned BLAS for Pentium for Linux.
- Need for quick/dynamic deployment of optimized routines.
- ATLAS - Automatic Tuned Linear Algebra Software
  - PhiPac from Berkeley
  - FFTW from MIT (http://www.fftw.org)

ATLAS

- **An adaptive software architecture**
  - **High-performance**
  - **Portability**
  - **Elegance**

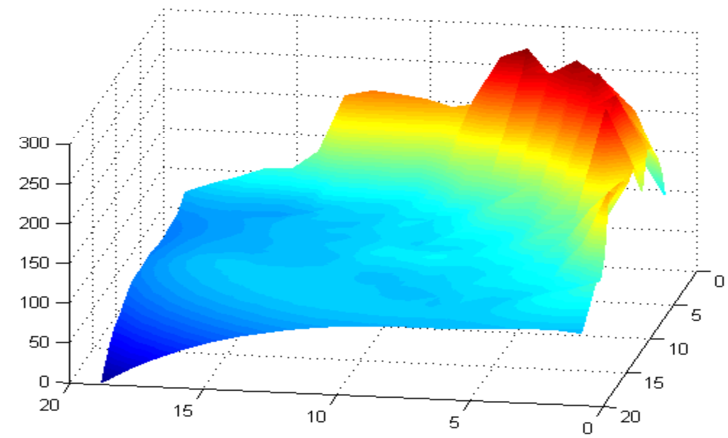
- **ATLAS is faster than all other portable BLAS implementations and it is comparable with machine-specific libraries provided by the vendor.**

ATLAS (DGEMM n = 500)



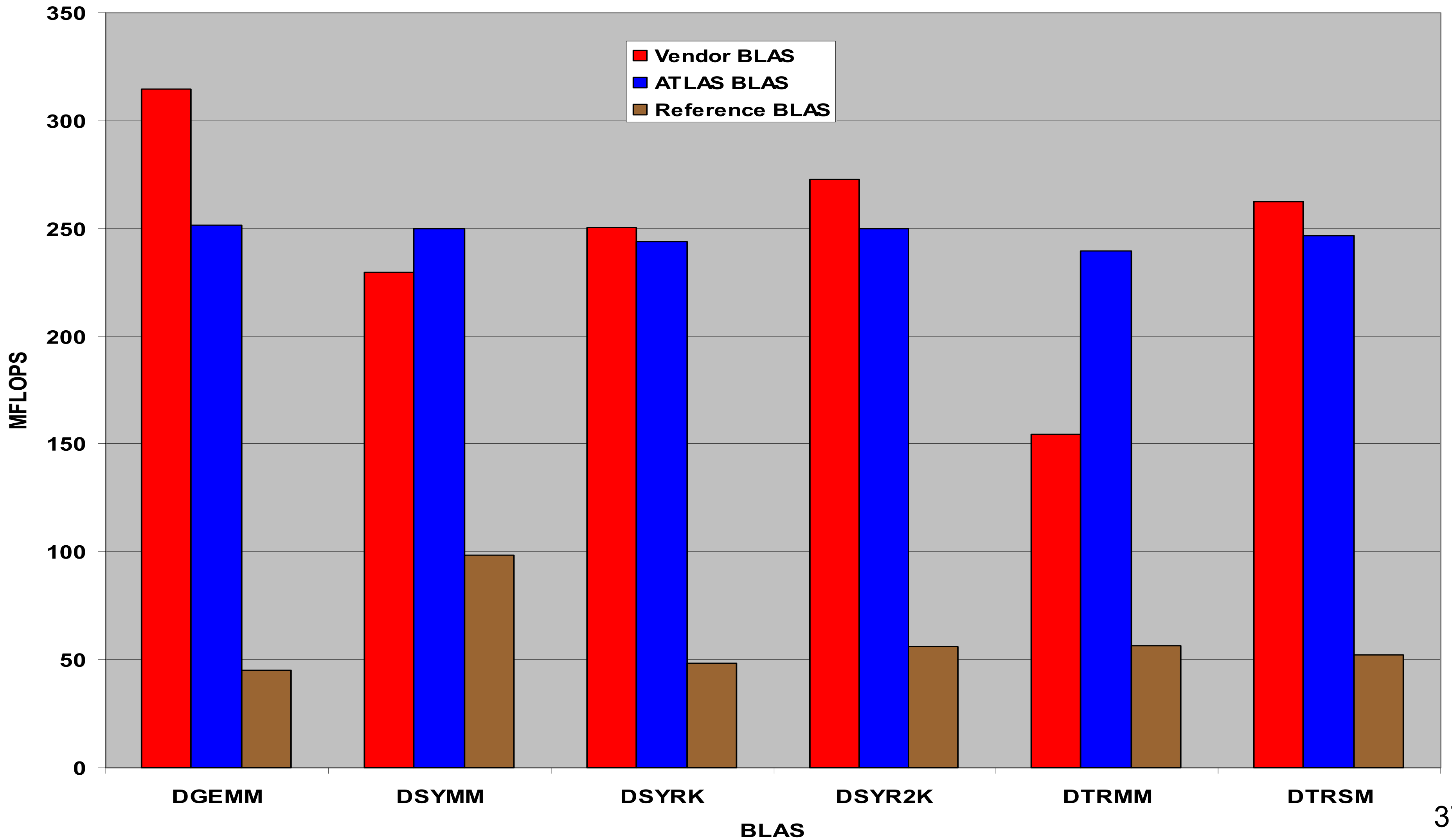
- **ATLAS is faster than all other portable BLAS implementations and it is comparable with machine-specific libraries provided by the vendor.**

Code Generation Strategy

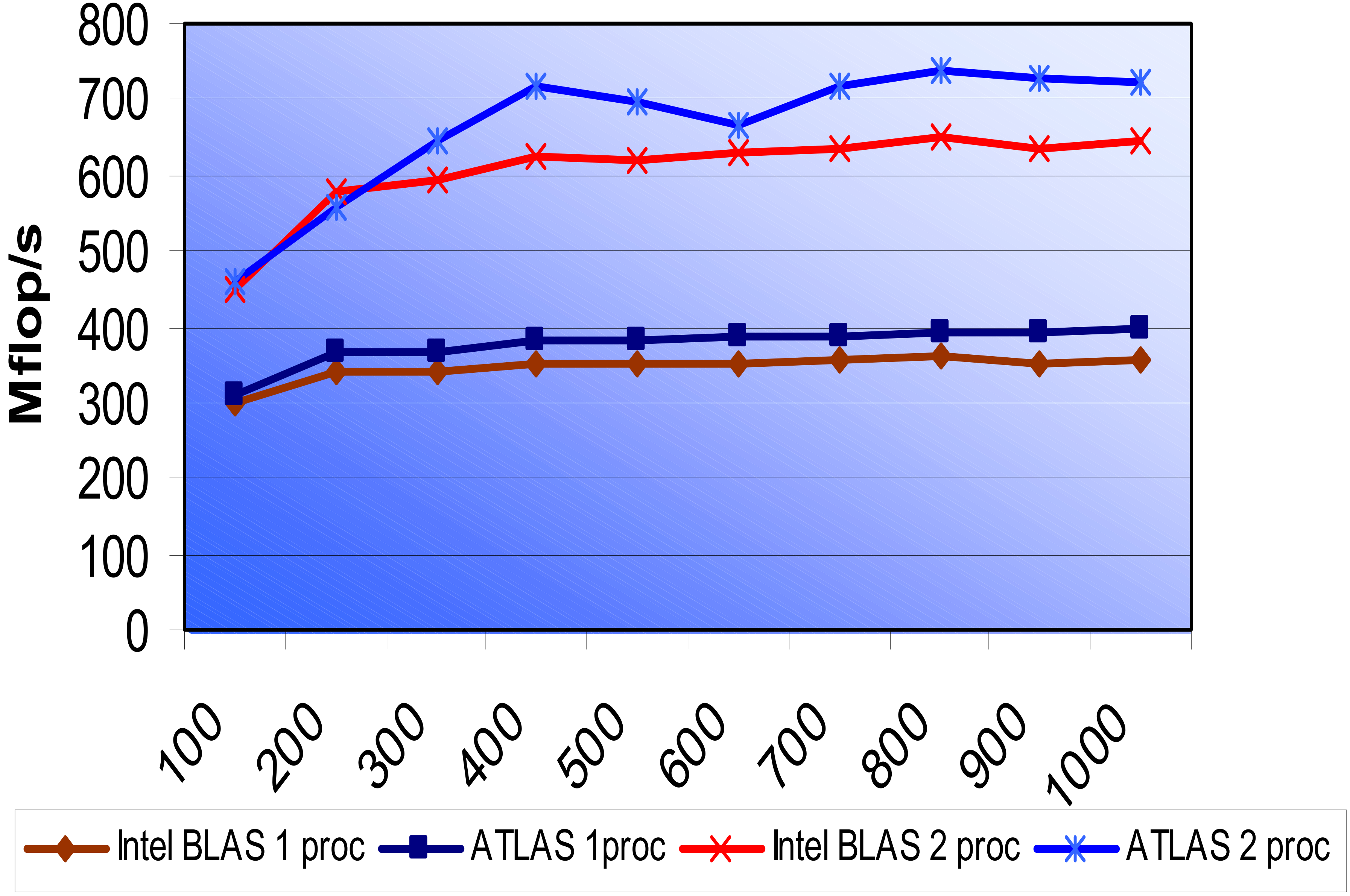


- Two phases:
  - Probes the systems for system features
  - Does a parameter study
- On-chip multiply optimizes for:
  - TLB access
  - L1 cache reuse
  - FP unit usage
  - Memory fetch
  - Register reuse
  - Loop overhead minimization
- New model of HP programming where critical code is machine generated using parameter optimization.
- Code is iteratively generated & timed until optimal case is found. We try:
  - Differing NBs
  - Breaking false dependencies
  - M, N and K loop unrolling
- Designed for RISC arch
  - Super Scalar
  - Need reasonable C compiler
- Takes ~20 minutes to run

500x500 Recursive Level 3 BLAS on UltraSparc 2 200



Multi-Threaded DGEMM
Intel PIII 550 MHz



Plans for ATLAS

- Software Release, available today:
  - Level 1, 2, and 3 BLAS implementations
  - See: www.netlib.org/atlas/
- Near Future:
  - Multi-treading
  - Optimize message passing system
  - Extend these ideas to Java directly
  - Sparse Matrix-Vector ops
- Futures:
  - Runtime adaptation
    - Sparsity analysis
    - Iterative code improvement
  - Specialization for user applications
  - Adaptive libraries

Tools for Performance Evaluation

- Timing and performance evaluation has been an art
  - Resolution of the clock
  - Issues about cache effects
  - Different systems
- Situation about to change
  - Today's processors have internal counters

Performance Counters

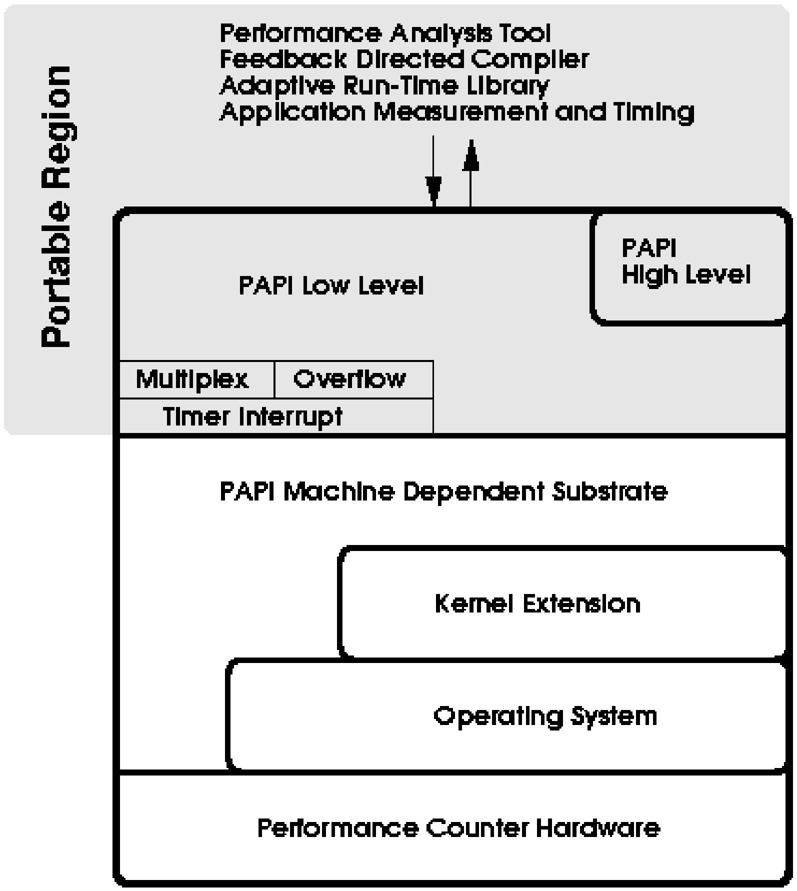
- Hidden from users.
- On most platforms the APIs, if they exist, are not appropriate for a common user, functional or well documented.
- Existing performance counter APIs
  - Cray T3E
  - SGI MIPS R10000
  - IBM Power series
  - DEC Alpha pfm pseudo-device interface
  - Windows 95, NT and Linux

Performance Data (cont.)

- Cycle count
- Floating point instruction count
- Integer instruction count
- Instruction count
- Load/store count
- Branch taken / not taken count
- Branch mispredictions
- Pipeline stalls due to memory subsystem
- Pipeline stalls due to resource conflicts
- I/D cache misses for different levels
- Cache invalidations
- TLB misses
- TLB invalidations

PAPI Implementation

- Performance Application Programming Interface
- The purpose of PAPI is to design, standardize and implement a portable and efficient API to access the hardware performance monitor counters found on most modern microprocessors
- Used by Tau (A. Malony) and SvPablo (D. Reed)

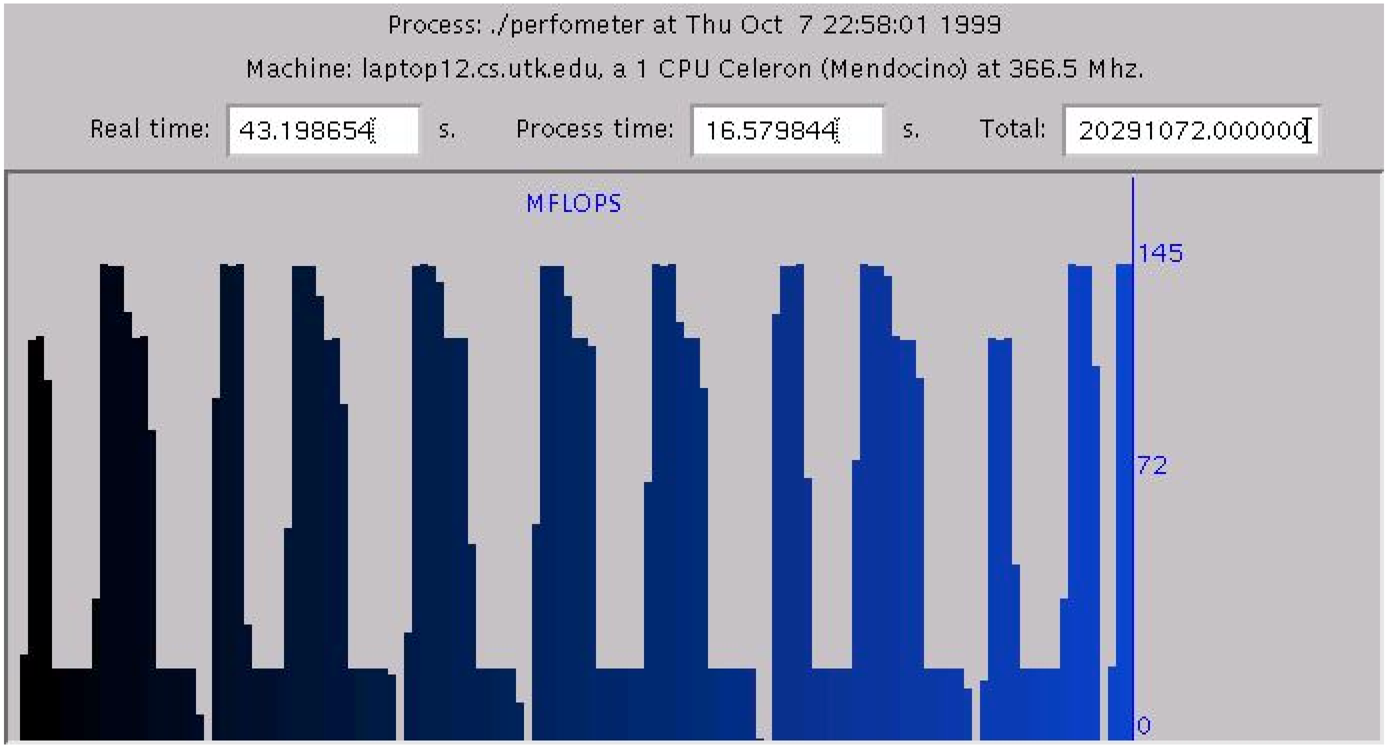


Perfometer Usage

- Application is instrumented with PAPI
  - One simple "call"
- Will be layered over the best existing vendor-specific APIs for these platforms
- Sections of code that are of interest are designated with specific colors
  - Using a call to mark_perfometer('color')
- Application is started and a Java window containing the Perfometer application is also started

Perfometer Screenshot

Call Perfometer()



Contributors to These Ideas

- Top500
  - Hans W. Meuer, Mannheim U
  - Erich Strohmaier, UTK
- NetSolve
  - Dorian Arnold, UTK
  - Susan Blackford, UTK
  - Henri Casanova, UCSD
  - Michelle Miller, UTK
  - Ganapathy Raman, UTK
  - Sathish Vadhiyar, UTK
- ATLAS
  - Clint Whaley, UTK
  - Antoine Petitet, UTK
- PAPI
  - Shirley Browne, UTK
  - Nathan Garner, UTK
  - Kevin London, UTK
  - Phil Mucci, UTK

For additional information see…

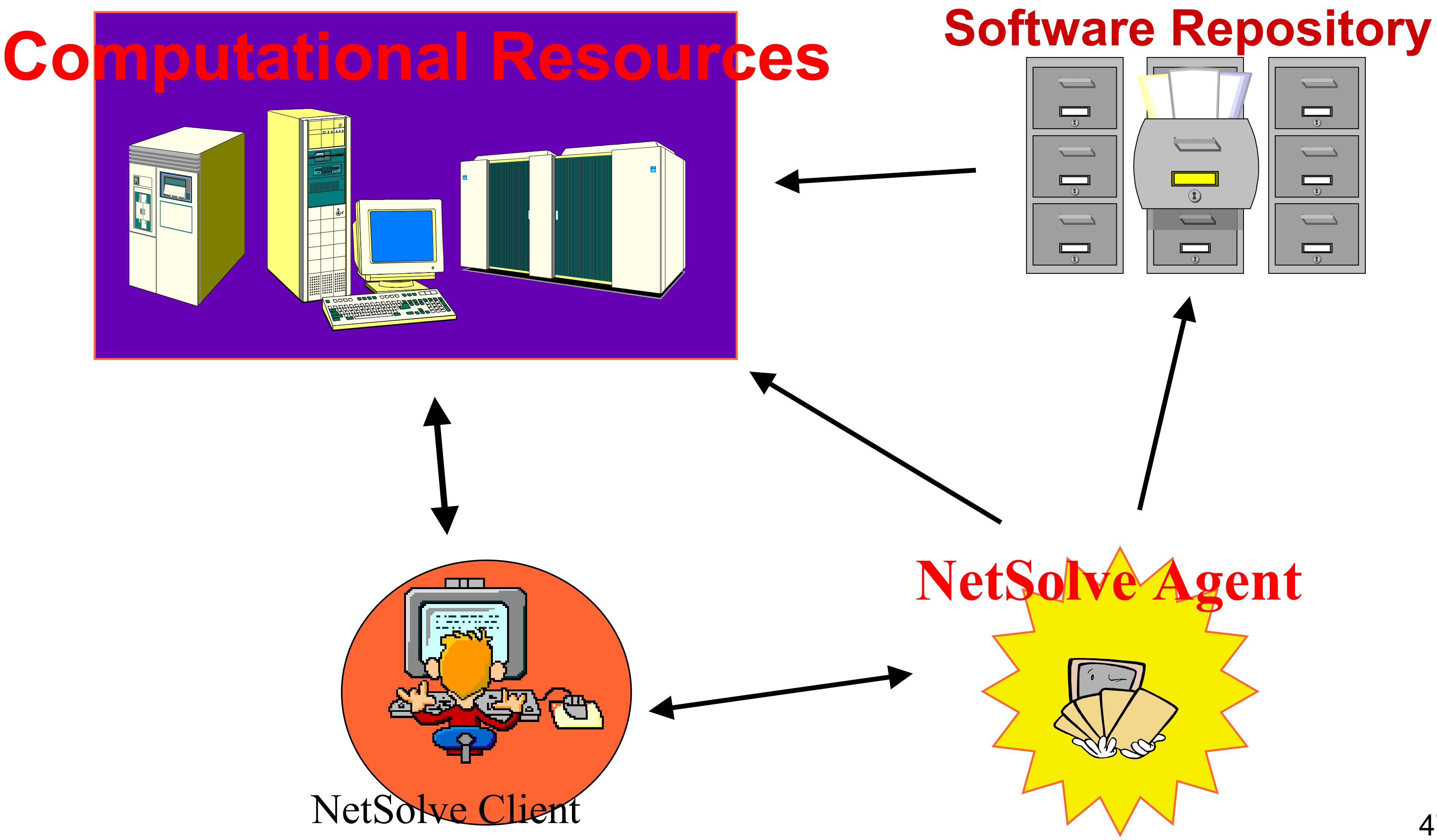
www.top500.org

www.netlib.org/atlas/

www.netlib.org/netsolve/

www.cs.utk.edu/~dongarra/

Next Step:
Hardware & Software Servers



http://www.cs.utk.edu/netsolve/

Futures

- List of Top 100 Clusters
  - IEEE Task Force on Cluster Computing
  - Interested in assembling a list of the Top n Clusters
  - Based on current metric
  - Starting to put together software to facilitate running and collection of data.
- Sparse Benchmark
  - Look at the performance in terms of sparse matrix operations
  - Iterative solvers
  - Beginning to collect data

NetSolve Applications and Interactions

- Tool integration
  - Globus - Middleware infrastructure (ANL/SSI)
  - Condor - Workstation farm (U Wisconsin)
  - NWS - Network Weather Service (U Tennessee)
  - SCIRun - Computational steering (U Utah)
  - Ninf - NetSolve-like system, (Tsukuba U)
- Library usage
  - LAPACK/ScaLAPACK - Parallel dense linear solvers
  - SuperLU/MA28 - Parallel sparse direct linear solvers(UCB/RAL)
  - PETSc/Aztec - Parallel iterative solvers (ANL/SNL)
  - Other areas as well (not just linear algebra)
- Applications
  - MCell - Microcellular physiology (UCSD/Salk)
  - IPARS - Reservoir Simulator (UTexas, Austin)
  - Virtual Human - Pulmonary System Model (ORNL)
  - RSICC - Radiation Safety sw/simulation (ORNL)
  - LUCAS - Land usage modeling (U Tennessee)
  - ImageVision - Computer Graphics and Vision (Graz U)

Sparse Matrices/Solvers

- Iterative and direct solvers: PETSc, Aztec, SuperLU, Ma28, …
- Support for compressed row/column sparse matrix storage -- significantly reduces network data transmission
- Sequential and parallel implementations available

SPOOLES

PETSc

SuperLU

MA28