Self Adapting Numerical Software (SANS) – Effort and Fault Tolerance in Linear Algebra Algorithms

Jack Dongarra
University of Tennessee
and
Oak Ridge National Laboratory

Overview

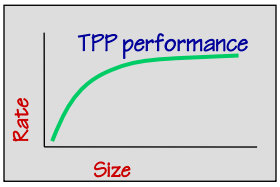
- **Quick look at fastest computers**
  - **From the Top500**
- **Self Adapting Numerical Software (SANS) Effort**
  - **Techniques for fault tolerant computations for iterative methods**
    - **Strategies when we start to using 10's of thousands of processors**

TOP500 SUPERCOMPUTER

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

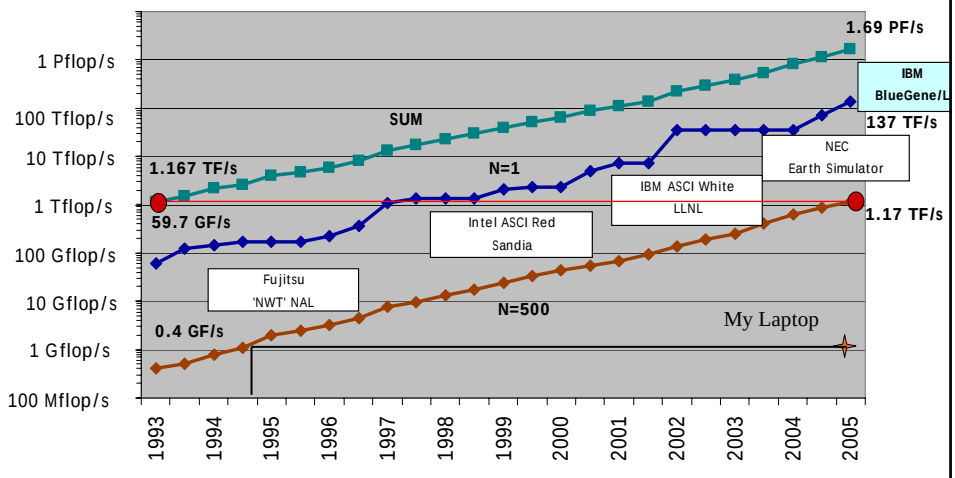
- Listing of the 500 most powerful Computers in the World
- Yardstick: Rmax from LINPACK MPP

  $Ax=b$, *dense problem*



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

TOP500 Performance Trend – June 2005



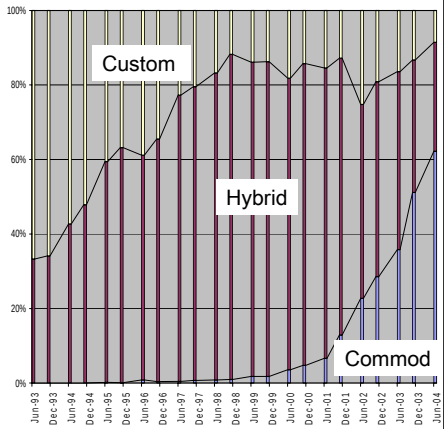
All systems > 1Tflop, last was at 300 last time; top100 starts at 3.4 tfs; 304 clusters

Architecture/Systems Continuum

Tightly Coupled

- **Custom processor with custom interconnect**
  - Cray X1
  - NEC SX-8
  - IBM Regatta
  - IBM Blue Gene/L
- **Commodity processor with custom interconnect**
  - SGI Altix
    - Intel Itanium 2
  - Cray XT3, XD1
    - AMD Opteron
- **Commodity processor with commodity interconnect**
  - Clusters
    - Pentium, Itanium, Opteron, Alpha
    - GigE, Infiniband, Myrinet, Quadrics
  - NEC TX7
  - IBM eServer
  - Dawning

Loosely Coupled



24th List: The TOP10

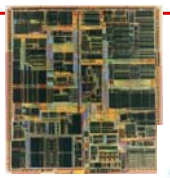
TOP500 SUPERCOMPUTER SITES

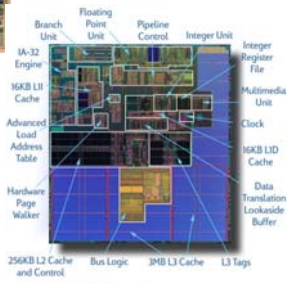
	Manufacturer	Computer	Rmax [TF/s]	Installation Site	Country	Type Year	#Proc
1	IBM	BlueGene/L β-System	136.8	Lawrence Livermore National Laboratory	USA	Custom 2005	65536
2	IBM	BGW - eServer Blue Gene Solution	91.29	IBM Thomas J. Watson Research Center	USA	Custom 2005	40960
3	SGI	Columbia Altix + Infiniband	51.87	NASA Ames	USA	Hybrid 2004	10160
4	NEC	Earth-Simulator	35.86	Earth Simulator Center	Japan	Custom 2002	5120
5	IBM	MareNostrum BladeCenter JS20, Myrinet	27.91	Barcelona Supercomputer Center	Spain	Commod 2004	4800
6	IBM	eServer BG Solution	27.45	University Groningen	NL	Custom 2005	12288
7	CCD	Thunder Itanium2, Quadrics	19.94	Lawrence Livermore National Laboratory	USA	Commod 2004	4096
8	IBM	eServer BG Solution	18.20	EPFL	Swiss	Custom 2005	8192
9	IBM	eServer BG Solution	18.20	Japan Adv. Inst. of Science and Technology	Japan	Custom 2005	8192
10	Cray Inc	Red Storm, Cray XT3	15.25	Sandia National Lab	USA	Hybrid 2005	5000

All 500 systems > 1 TFlop/s; 304 machines clusters; top10 average 16K proc; US has 294
IBM 259. HP 131, SGI 24, Dell 21 and Cray 16 (6 of 10 IBM, 5 in USA); > 4800 processors

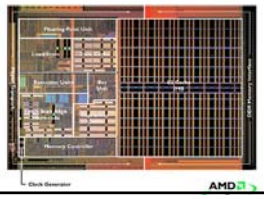
Commodity Processors

- **Intel Pentium Nocona**
  - 3.6 GHz, peak = 7.2 Gflop/s
  - Linpack 100 = 1.8 Gflop/s
  - Linpack 1000 = 4.2 Gflop/s
- **Intel Itanium 2**
  - 1.6 GHz, peak = 6.4 Gflop/s
  - Linpack 100 = 1.7 Gflop/s
  - Linpack 1000 = 5.7 Gflop/s
- **AMD Opteron**
  - 2.6 GHz, peak = 5.2 Gflop/s
  - Linpack 100 = 1.6 Gflop/s
  - Linpack 1000 = 3.9 Gflop/s

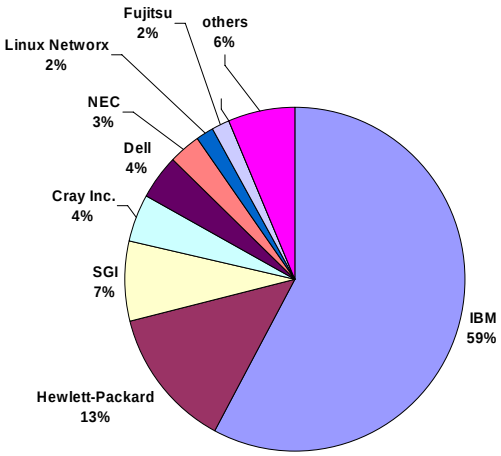




McKinley microprocessor



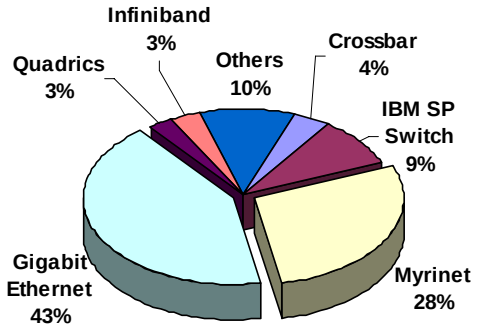
Manufacturers/Integraters of the Top500 by Performance



er

Today Intel is inside 2/3 of the Top500 machines

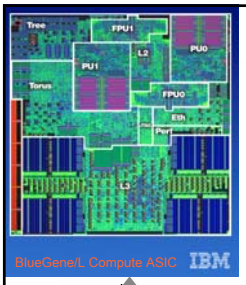
Interconnects – June 2005 Top500



05 9

IBM BlueGene/L

131,072 Processors (#1-64K and #2-40K)



Full system total of
131,072 processors

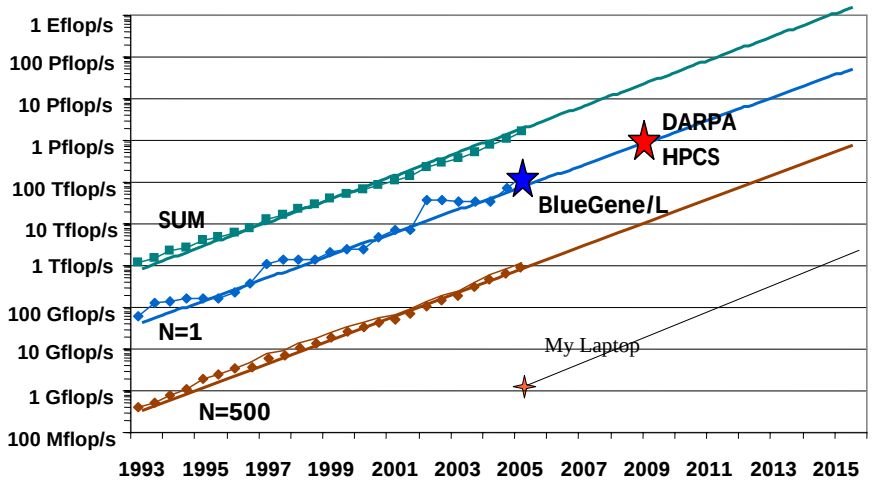
The compute node ASICs include all networking and processor functionality. Each compute ASIC includes two 32-bit superscalar PowerPC 440 embedded cores (note that L1 cache coherence is not maintained between these cores).

(10K sec about 3 hours; n=1.2M

"Fastest Computer"
BG/L 700 MHz 64K proc
32 racks
Peak: 184 Tflop/s
Linpack: 135 Tflop/s
73% of peak

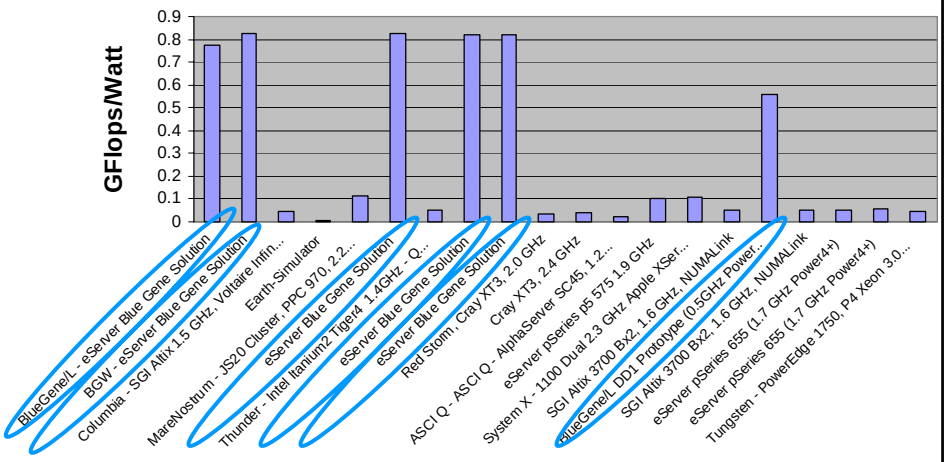
05 10

Performance Projection



05 11

Fuel Efficiency: Gflops/Watt



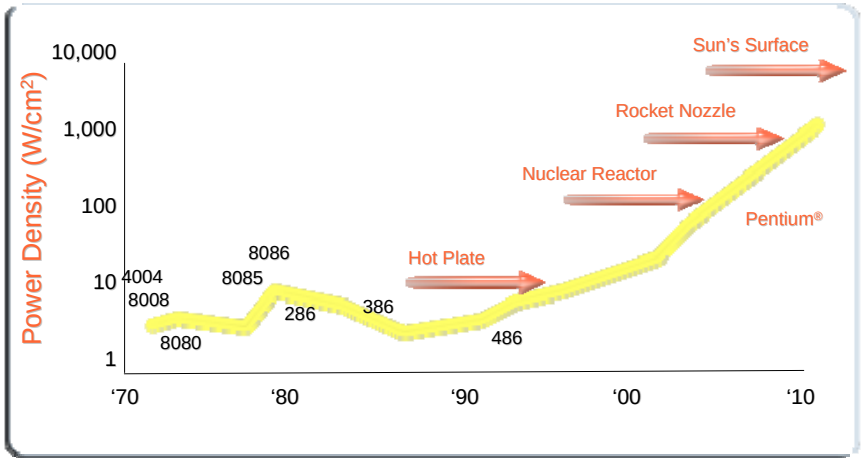
05

Top 20 systems
Based on processor power rating only (3,>100,>800)

12

Today's CPU Architecture:

Heat becoming an unmanageable problem



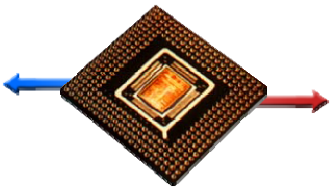
Intel Developer Forum, Spring 2004 - Pat Gelsinger (Pentium at 90 W)

Cube relationship between the cycle time and power

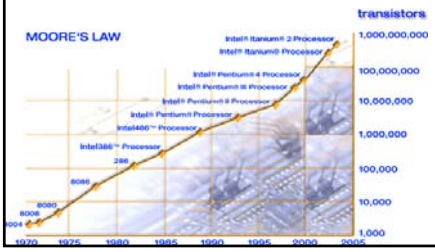
Increasing CPU Performance:

A Delicate Balancing Act

Lower Voltage



Increase Clock Rate & Transistor Density



We have seen increasing number of gates on a chip and increasing clock speed.

Heat becoming an unmanageable problem, Intel Processors > 100 Watts

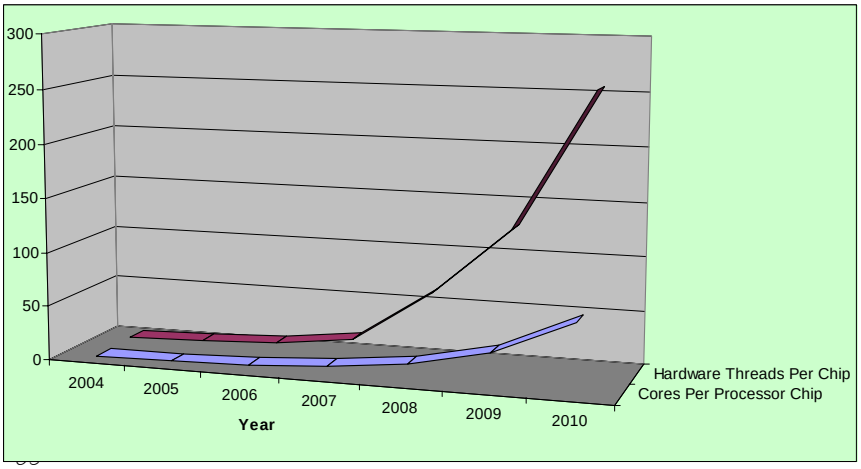
We will not see the dramatic increases in clock speeds in the future.

However, the number of gates on a chip will continue to increase.

Intel Yonah will double the processing power on a per watt basis.

CPU Desktop Trends 2004-2010

- Relative processing power will continue to double every 18 months
- 256 logical processors per chip in late 2010



15

Commodity Processor Trends

Bandwidth/Latency is the Critical Issue, not FLOPS

Got Bandwidth?

	Annual increase	Typical value in 2005
Single-chip floating-point performance	59%	4 GFLOP/s
Front-side bus bandwidth	23%	1 GWord/s = 0.25 word/flop
DRAM latency	(5.5%)	70 ns = 280 FP ops = 70 loads

05 Source: *Getting Up to Speed: The Future of Supercomputing*, National Research Council, 222 pages, 2004, National Academies Press, Washington DC, ISBN 0-309-09502-6.

16

A PetaFlop Computer by the End of the Decade

- 10 Companies working on a building a Petaflop system by the end of the decade.
  - Cray
  - IBM
  - Sun

  } HPCS
  - Dawning
  - Galactic
  - Lenovo

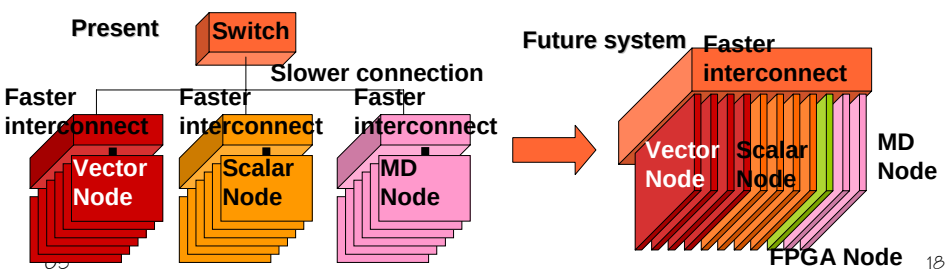
  } Chinese Companies
  - Hitachi
  - NEC

  } Japanese "Life Simulator" (10 Pflop/s)
  - Fujitsu
  - Bull

05 17

Japanese: Tightly-Coupled Heterogeneous System

- Would like to get to 10 PetaFlop/s by 2011
- Scalable, fits any computer center
  - Size, cost, ratio of components
- Easy and low-cost to develop new component
- Scale merit of components



18

Reliability of Leading-Edge HPC Systems

System	CPUs	Reliability
LANL ASCI Q	8,192	MTBI: 6.5 hours. Leading outage sources: storage, CPU, memory.
LLNL ASCI White	8,192	MTBF: 5.0 hours ('01) and 40 hours ('03). Leading outage sources: storage, CPU, 3rd-party HW.
Pittsburgh Lemieux	3,016	MTBI: 9.7 hours.

MTBI: mean time between interrupts = wall clock hours / # downtime periods
MTBF: mean time between failures (measured)

- 100K processor systems
  - are in construction
  - we have fundamental challenges in dealing with machines of this size
  - … and little in the way of programming support

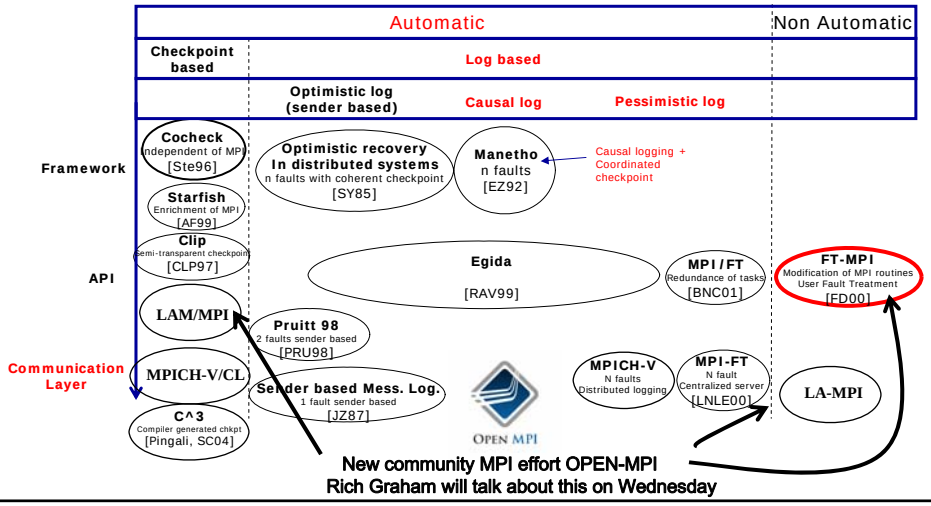
Fault Tolerance: Motivation

- Trends in HPC:
  - High end systems with thousand of processors
- Increased probability of a node failure
  - Most systems nowadays are robust
- MPI widely accepted in scientific computing
  - Process faults not tolerated in MPI model

Mismatch between hardware and (non fault-tolerant) programming paradigm of MPI.

Related work

A classification of fault tolerant message passing environments considering
A) level in the software stack where fault tolerance is managed and
B) fault tolerance techniques.



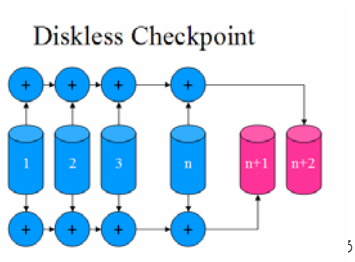
Three Ideas for Fault Tolerant Linear Algebra Algorithms

- **Lossless diskless check-pointing for iterative methods**
  - Checksum maintained in active processors
  - On failure, roll back to checkpoint and continue
  - No lost data
- **Lossy approach for iterative methods**
  - No checkpoint maintained
  - On failure, approximate missing data and carry on
  - Lost data but use approximation to recover
- **Check-pointless methods for dense algorithms**
  - Checksum maintained as part of computation
  - No roll back needed; No lost data

05

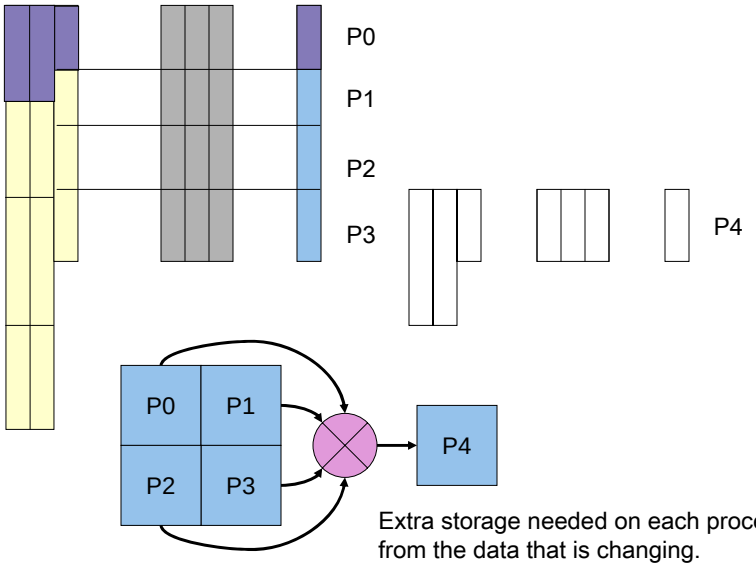
First Approach

- **Lossless diskless check pointing for iterative methods**
  - **We do a diskless checkpoint**
    - **Checksum written to an additional processor, not to disk**
    - **Similar to RAID for disks**
  - **Roll back to where the checkpoint was taken and proceed**
  - **Simultaneous-multiple processor failures can occur if have extra processors to checkpoint**
  - **Similar to Reed-Solomon encoding**



05

Diskless Version



Extra storage needed on each process from the data that is changing.
Actually don't do XOR, add the information.

05

24

FT PCG Algorithm Analysis

Compute $r^{(0)} = b - Ax^{(0)}$ for some initial guess $x^{(0)}$
for $i = 1, 2, \ldots$
  solve $Mz^{(i-1)} = r^{(i-1)}$
  $\rho_{i-1} = r^{(i-1)^T} z^{(i-1)}$
  if $i = 1$
    $p^{(1)} = z^{(0)}$
  else
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $p^{(i)} = z^{(i-1)} + \beta_{i-1} p^{(i-1)}$
  endif
  $q^{(i)} = Ap^{(i)}$
  $\alpha_i = \rho_{i-1}/p^{(i)^T} q^{(i)}$
  $x^{(i)} = x^{(i-1)} + \alpha_i p^{(i)}$
  $r^{(i)} = r^{(i-1)} - \alpha_i q^{(i)}$
  check convergence; continue if necessary
end

Global Operations

Global operation in PCG: three dot product, one preconditioning, and one matrix vector multiplication.

05 Global operation in Checkpoint: encoding the local checkpoint. 25

FT PCG Algorithm Analysis

Compute $r^{(0)} = b - Ax^{(0)}$ for some initial guess $x^{(0)}$
for $i = 1, 2, \ldots$
  solve $Mz^{(i-1)} = r^{(i-1)}$
  $\rho_{i-1} = r^{(i-1)^T} z^{(i-1)}$
  if $i = 1$
    $p^{(1)} = z^{(0)}$
  else
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $p^{(i)} = z^{(i-1)} + \beta_{i-1} p^{(i-1)}$
  endif
  $q^{(i)} = Ap^{(i)}$
  $\alpha_i = \rho_{i-1}/p^{(i)^T} q^{(i)}$
  $x^{(i)} = x^{(i-1)} + \alpha_i p^{(i)}$
  $r^{(i)} = r^{(i-1)} - \alpha_i q^{(i)}$
  check convergence; continue if necessary
end

Checkpoint *x, r,* and *p* every k iterations

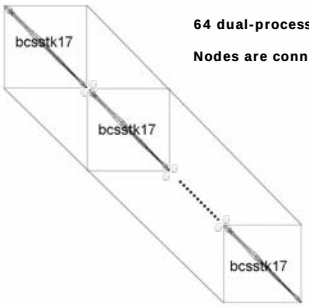
Global Operations

Global operation in PCG: three dot product, one preconditioning, and one matrix vector multiplication.

05 Global operation in Checkpoint: encoding the local checkpoint. 26
Global operation in checkpoint can be localized by sub-group.

PCG: Performance with Different MPI Implementations

bcsstk17:
The size is:
10974 x 10974
Non-zeros:
428650
Sparsity:
39 non-zeros per row on average
Source:
Linear equation from elevated pressure vessel



64 dual-processor 2.4 GHz AMD Opteron nodes

Nodes are connected with a Gigabit Ethernet.

N	Procs	LAM-7.0.4	MPICH2-1.0	FT-MPI	FT-MPI ckpt /2000 iters	FT-MPI exit 1 proc @10000 iters
165K	15	522.5	536.3	517.8	518.9	521.7
329K	30	532.9	542.9	532.2	533.3	537.5
658K	60	545.5	553.0	546.5	547.8	554.2
1317K	120	674.3	624.4	622.9	624.4	637.1

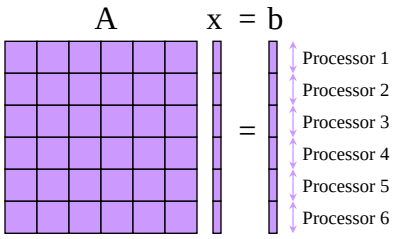
http://icl.cs.utk.edu/ft-mpi/

Second Approach

- **Lossy approach for iterative methods**
  - **Here there is only a checkpoint of the primary data**
    - **Continuous checkpointing is not done during the iteration.**
  - **When the failure occurs we will approximate the missing data and continue**
    - **No guarantee here; may or may not work**

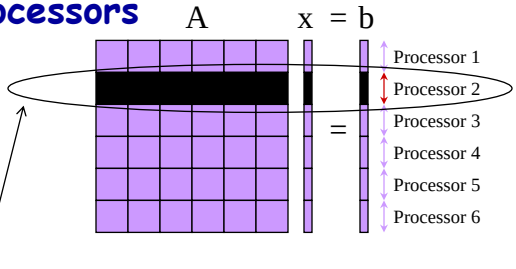
Lossy algorithm : basic idea

- Let us assume that the exact solution of the system Ax=b is stored on different processors by rows



Lossy algorithm : basic idea

- Let us assume that the exact solution of the system Ax=b is stored on different processors

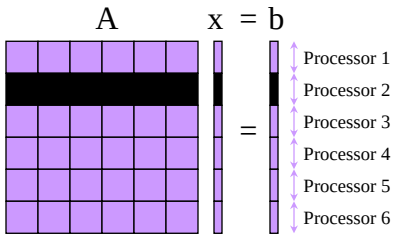


Processor 2 (e.g.) fails, all its data is lost.

How to recover the lost part of x in this case?

Lossy algorithm : basic idea

- **Let us assume that the exact solution of the system Ax=b is stored on different processors**

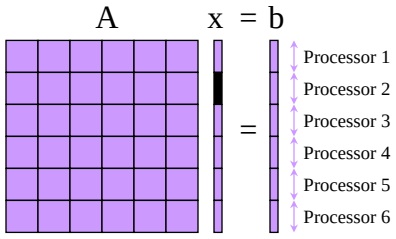


3 steps

Step 1: recover a processor and a running parallel environment (the job of the FT-MPI library)

Lossy algorithm : basic idea

- **Let us assume that the exact solution of the system Ax=b is stored on different processors**



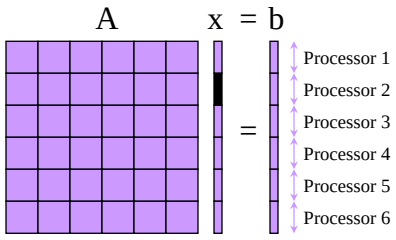
3 steps

Step 1: revover a processor and a running parallel environement (the job of the FT-MPI library)

Step 2: recover A_{21} A_{22}, …, A_{n2} and b_2 (the original data) on the failed processor

Lossy algorithm : basic idea

- **Let us assume that the exact solution of the system Ax=b is stored on different processors**



3 steps

Step 1: recover a processor and a running parallel environment (the job of the FT-MPI library)

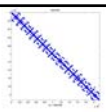
Step 2: recover A_{21} A_{22}, …, A_{n2} and b_2 (the original data) on the failed processor

Step 3: Notice that

$A_{21} x_1 + A_{22} x_2 + \ldots + A_{2n} x_n = b_2)$

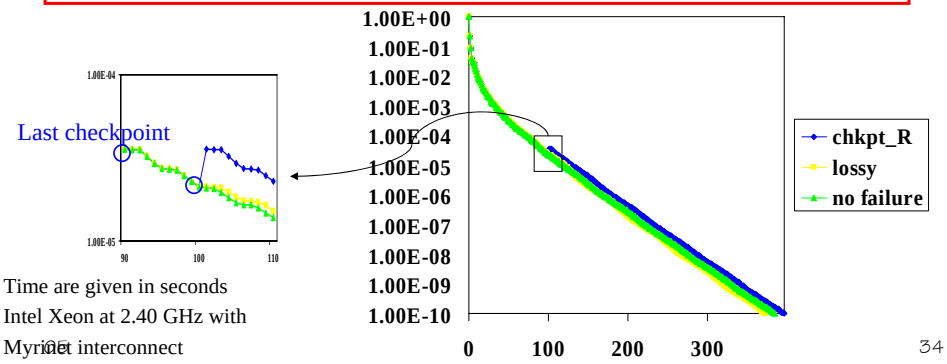
$x_2 = A_{22}^{-1} (b_2 - \Sigma_{i \neq 2} A_{2i} x_i)$

Using GMRES(30) Non Symetric Matrix



stomach: n=213,360; nnz=3,021,648; tol=10^{-10}; #procs=16; n_f=13,335; nnz=185,541

recovery	$iter_f$	#iter	T_{Wall}	T_{Chkpt}	T_{Roll}	T_{Recov}	T_I	$T_{II,a,b}$	T_{III}
lossy	no	385	38.89						
$chkpt_D$	no	385	41.04	1.92					
lossy	100	372	42.38		1.56	5.38	1.03	0.33	3.91
$chkpt_R$	100	395	45.49	1.92	2.40	1.68	1.02	0.32	0.20



Time are given in seconds
Intel Xeon at 2.40 GHz with Myrinet interconnect

Third Approach

- **Checkpointless methods for dense algorithms**
  - **We need extra processors to participate in the computation**
    - **The extra processors carry the active checksum**
  - **No roll back needed; just compute what's missing and carry on.**

Matrix-Vector Multiplication with Checksum Matrix

$$M^r = \begin{pmatrix} M_{11} & M_{12} & \cdots & M_{1q} \\ \vdots & \vdots & \cdots & \vdots \\ M_{p1} & M_{p2} & \cdots & M_{pq} \\ \sum_{i=1}^{p} M_{i1} & \sum_{i=1}^{p} M_{i2} & \cdots & \sum_{i=1}^{p} M_{iq} \end{pmatrix} \qquad v = \begin{pmatrix} v_1 \\ \cdots \\ v_q \\ \sum_{q} v_i \end{pmatrix}$$

Let $$b = M^r v = \begin{pmatrix} \sum_{j=1}^{q} M_{1j} v_j \\ \vdots \\ \sum_{j=1}^{q} M_{pj} v_j \\ \sum_{i=1}^{p} \sum_{j=1}^{q} M_{ij} v_j \end{pmatrix}$$

Then $b_1 + \cdots + b_p = b_{p+1}$

Conclusion: Any singular failure in the result b can be corrected

Fault Tolerent Dense Matrix Computations

- Assume the original matrix M is distributed into a *p* by *q* processor grid with a 2D block cyclic distribution. Then from processor point of view, the distributed matrix is

$$M = \begin{pmatrix} M_{11} & \cdots & M_{1q} \\ \vdots & \cdots & \vdots \\ M_{p1} & \cdots & M_{pq} \end{pmatrix}$$

, where M_{ij} is the local matrix on processor (*i* , *j*).

- Define the *full distributed checksum* matrix of M as:

$$M^{f} = \begin{pmatrix} M_{11} & \cdots & M_{1q} & \sum_{j=1}^{q} M_{1j} \\ \vdots & \cdots & \vdots & \vdots \\ M_{p1} & \cdots & M_{pq} & \sum_{j=1}^{q} M_{pj} \\ \sum_{i=1}^{p} M_{i1} & \cdots & \sum_{k=1}^{p} M_{iq} & \sum_{i=1}^{p}\sum_{j=1}^{q} M_{ij} \end{pmatrix}$$

- For p x q processors need extra p + q + 1 processors to maintain the checksum.

Overhead for Recovery

Size of Matrix	12,800	19,200	25,600
Process Grid w/o FT	2 by 2	3 by 3	4 by 4
Process Grid w/ FT	3 by 3	4 by 4	5 by 5
Execution time w/o FT	453.9	699.1	965.3
Execution time w/ Recvr	543.4	800.6	1078.4
Time for Recovery	31.6	33.4	34.9
Overhead for Recovery	7.0	4.8	3.6

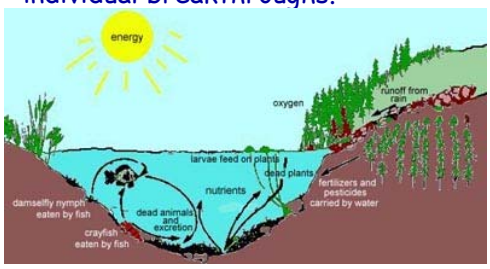
Next Steps

- Software to determine the checkpointing interval and number of checkpoint processors from the machine characteristics.
  - Perhaps use historical information.
  - Monitoring
  - Migration of task if potential problem
- Local checkpoint and restart algorithm.
  - Coordination of local checkpoints.
  - Processors hold backups of neighbors.
- Have the checkpoint processes participate in the computation and do data rearrangement when a failure occurs.
  - Use p processors for the computation and have k of them hold checkpoint.
- Generalize the ideas to provide a library of routines to do the diskless check pointing.
- Look at "real applications" and investigate "Lossy" algorithms.

Future Challenge: Developing a New Ecosystem for HPC

From the NRC Report on "The Future of Supercomputing":

- Hardware, software, institutions, applications, and people who solve supercomputing applications can be thought of collectively as an ecosystem
- Research investment in HPC should be informed by the ecosystem point of view - progress must come on a broad front of interrelated technologies, rather than in the form of individual breakthroughs.



A supercomputer ecosystem is a continuum of computing platforms, system software, and the people who know how to exploit them to solve computational science applications.

Real Crisis With HPC Is With The Software

- Our ability to configure a hardware system capable of 1 PetaFlop (10^{15} ops/s) is without question just a matter of time and $$.
- A supercomputer application and software are usually much more long-lived than a hardware
  - Hardware life typically five years at most…. Apps 20-30 years
  - Fortran and C are the main programming models (still!!)
- The REAL CHALLENGE is Software
  - Programming hasn't changed since the 70's
  - HUGE manpower investment
    - MPI… is that all there is?
  - Often requires HERO programming
  - Investments in the entire software stack is required (OS, libs, etc.)
- Software is a major cost component of modern technologies.
  - The tradition in HPC system procurement is to assume that the software is free… SOFTWARE COSTS (over and over)

Some Current Unmet Needs

- Performance / Portability
- Fault tolerance
- Better programming models
  - Global shared address space
  - Visible locality
- Maybe coming soon (incremental, yet offering real benefits):
  - Global Address Space (GAS) languages: UPC, Co-Array Fortran, Titanium, Chapel)
    - "Minor" extensions to existing languages
    - More convenient than MPI
    - Have performance transparency via explicit remote memory references

Collaborators / Support

- **Top500 Team**
  - Erich Strohmaier, NERSC
  - Hans Meuer, Mannheim
  - Horst Simon, NERSC
- **Fault Tolerant Work**
  - Julien Langou, UTK
  - Jeffery Chen, UTK
- **FT-MPI**

  http://icl.cs.utk.edu/ft-mpi/

  - Graham Fagg, UTK
  - Edgar Gabriel, UH
  - Thara Angskun, UTK
  - George Bosilca, UTK
  - Jelena Pjesivac-Grbovic, UTK

05