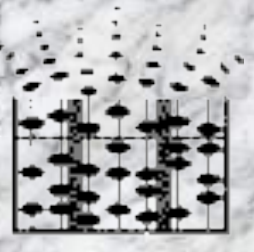
CIS 631
Parallel Processing

Lecture 1: Introduction

Allen D. Malony
malony@cs.uoregon.edu

Department of Computer and Information Science
University of Oregon



Outline

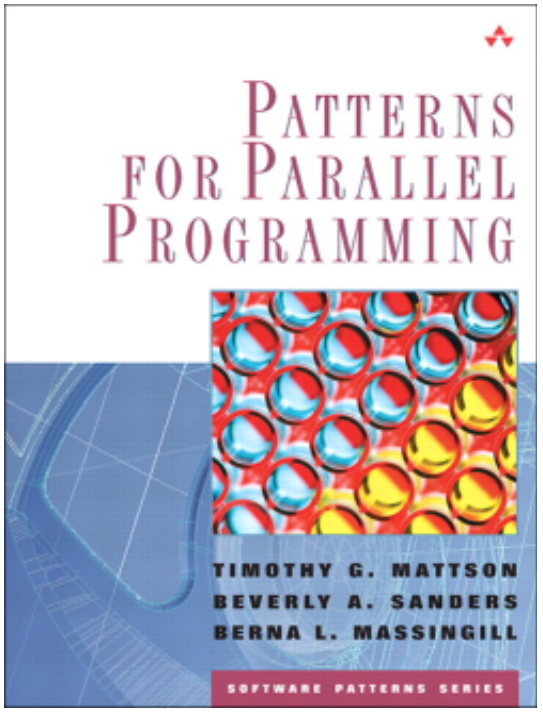
- Course Overview
  - What is CIS 631?
  - What is expected of you?
  - What will you learn in CIS 631?
- Parallel Computing
  - What is it?
  - What motivates it?
  - Trends that shape the field
  - Large-scale problems and high-performance
  - Parallel architecture types
  - Scalable parallel computing and performance

CIS 631 - Parallel Processing (Computing)

- Graduate course
  - Prerequisite: CIS 429/529 (Computer Architecture)
- Lecture time
  - Monday, Wednesday: 12:00 - 13:20 pm, 200 Deschutes
- Final schedule
  - Tuesday, March 19, 10:15, 200 Deschutes
- Course webpage
  - http://www.cs.uoregon.edu/classes/13W/cis631
  - Programming experience
  - C / C++ (required), and Java and/or Fortran (preferred)
  - Linux or Unix background (desired)

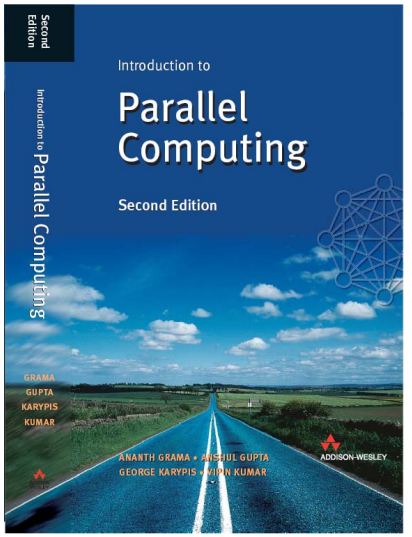
Required Books

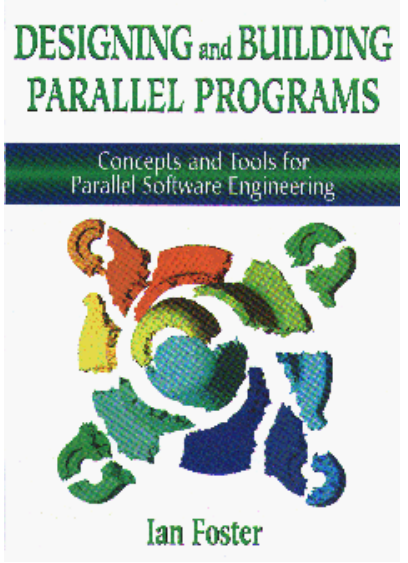
- *Patterns for Parallel Programming*
  T. Mattson, B. Sanders, B. Massingill,
  Addison Wesley, 2005.
  - Targets parallel programming
  - Pattern language approach to parallel program design and development
  - Excellent references
- New version of the book is in the works
  - Unfortunately, it is not yet ready
  - Add GPU and data parallel patterns
  - Supplement book with papers and online resources
- Hope to have guest lecture(s) from Dr. Mattson



Recommended Books

- *Introduction to Parallel Computing*
  A. Grama, A. Gupta, G. Karypis, V. Kumar, Addison Wesley, 2nd Ed., 2003
  - Lecture slides from authors online
  - Excellent reference list at end
  - Used for CIS 631 before
  - Getting old for latest hardware
- Designing and Building Parallel Programs, Ian Foster. Addison Wesley, 1995.
  - Entire book is online!!!
  - Historical book, but very informative





Overview

- Broad/Old field of computer science concerned with:
  - Architecture, HW/SW systems, languages, programming paradigms, algorithms, and theoretical models
  - Computing in parallel
- Performance is the *raison d'être* for parallelism
  - High-performance computing
  - Drives computational science revolution
- Topics of study
  - Parallel architectures
  - Parallel programming
  - Parallel algorithms
  - Parallel performance models and tools
  - Parallel applications

Lectures

- Books, papers, online materials are you main sources for broad and deep background in parallel computing
- Lectures should be more interactive
  - Supplement other sources of information
  - Covers topics of more priority
  - Intended to give you some of my perspective
    - I have been working parallel computing for 25+ years
  - Some topics from previous years and some new
- Will provide online access to lecture slides
- May be some guest lectures throughout the quarter
- May be some extra training

Assignments

- Programming exercises
  - Get familiar with ACISS and parallel programming
  - Not worth much, so don't sweat it, just learn how
- Programming project
  - Individual larger programming exercise
- Team term project
  - Programming, presentation, paper
- Research summary paper
- Midterm exam later in the quarter
- No final exam (project presentations during final period)
- See course webpage for due dates

Programming Exercises

- Shared memory multi-threaded programming
  - OpenMP programming (with C, C++, or Fortran)
  - Other: Pthreads, Thread Building Blocks (TBB)
- Distributed memory message passing programming
  - MPI programming
  - Other: Unified Parallel C (UPC), Co-Array Fortran (CAF)
- Simple programs to be done individually
- Build experience in parallel programming
- Use LiveDVD and ACISS cluster
- May play around with GPU programming
  - CUDA, OpenCL, OpenACC

Individual Programming Project

- More extensive than programming exercises
  - Designed to improve and test your programming skills
- Use whatever parallel programming technology you want
- Use ACISS cluster
- Individual work

Term Project

- Major programming project for the course
  - Non-trivial parallel application
  - Include performance analysis
  - Use ACISS cluster
- Project teams
  - 2-3 person teams, 2-4 teams (depending on enrollment)
  - Will try to balance skills
- Project dates
  - Proposal: due February 11
  - Project talk: March 19
  - Project report: March 19
- See course webpage for system accounts!!!

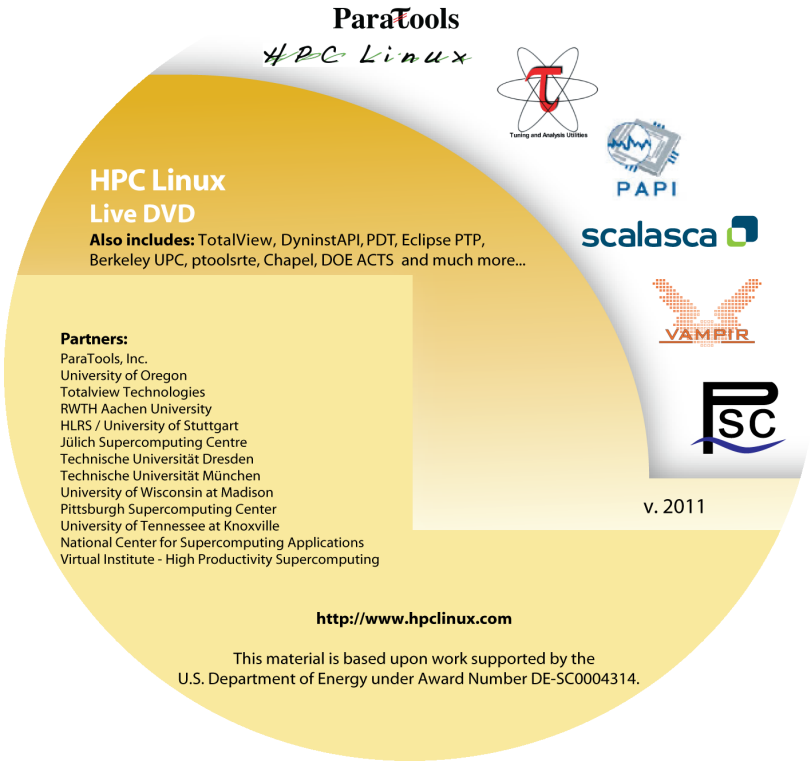
ACISS

- *Applied Computational Instrument for Scientific Synthesis*
  - NSF MRI R^2 award (2010)
- *Basic nodes* (1,536 total cores)
  - 128 ProLiant SL390 G7
  - Two Intel X5650 2.66 GHz 6-core CPUs per node
  - 72GB DDR3 RAM per basic node
- *Fat nodes* (512 total cores)
  - 16 ProLiant DL 580 G7
  - Four Intel X7560 2.266 GHz 8-core CPUs per node
  - 384GB DDR3 per fat node

- *GPU nodes* (624 total cores, 156 GPUs)
  - 52 ProLiant SL390 G7 nodes, 3 NVidia M2070 GPUs (156 total GPUs)
  - Two Intel X5650 2.66 GHz 6-core CPUs per node (624 total cores)
  - 72GB DDR3 per GPU node
- ACISS has 2672 total cores
- ACCIS is located in the UO Computing Center

HPC Linux and LiveDVD

- For parallel programming learning, you can use HPC Linux on a LiveDVD
  - Developed on the NSF POINT project for HPC software training
  - HPC Linux is configured with a complete HPC software development environment
  - LiveDVD can be booted on laptops or run in a VM
- HPC Linux is supported by ParaTools, Inc. http://www.paratools.com/livedvd.php



Term Paper

- Investigate parallel computing topic of interest
  - More in depth review
  - Individual choice
  - Summary of major points
- Requires minimum of ten references
  - Book and other references has a large bibliography
  - Google Scholar, Keywords: parallel computing
  - NEC CiteSeer Scientific Literature Digital Library
- Paper topic abstract and references due February 9
- Final term paper due last class meeting
- Individual work

Grading

- 5% Exercises
- 25% Term paper
- 20% Midterm exam
- 10% Project 1
- 40% Project 2
  - 10% Presentation
  - 30% Report and accomplishments

Schedule

- See course webpage

What will you get out of CIS 631?

- In-depth understanding of parallel computer design
- Knowledge of how to program parallel computer systems
- Understanding of pattern-based parallel programming
- Exposure to different forms parallel algorithms
- Practical experience using a parallel cluster
- Background on parallel performance modeling
- Techniques for empirical performance analysis
- Fun and new friends

Parallel Processing – What is it?

- A *parallel computer* is a computer system that uses multiple processing elements simultaneously in a cooperative manner to solve a computational problem
- *Parallel processing* includes techniques and technologies that make it possible to compute in parallel
  - Hardware, networks, operating systems, parallel libraries, languages, compilers, algorithms, tools, …
- Parallel computing is an evolution of serial computing
  - Parallelism is natural
  - Computing problems differ in level / type of parallelism
- Parallelism is all about performance! Really?

Concurrency

- Consider multiple tasks to be executed in a computer
- Tasks are concurrent with respect to each if
  - They *can* execute at the same time (*concurrent execution*)
  - Implies that there are no dependencies between the tasks
- Dependencies
  - If a task requires results produced by other tasks in order to execute correctly, the task's execution is *dependent*
  - If two tasks are dependent, they are not concurrent
  - Synchronization must be used to satisfy dependencies
- Concurrency is fundamental to computer science
  - Operating systems, databases, networking, …

Concurrency and Parallelism

- Concurrent is not the same as parallel! Why?
- Parallel execution
  - Concurrent tasks *actually* execute at the same time
  - Multiple (processing) resources have to be available
- Parallelism = concurrency + "parallel" hardware
  - Both are required
  - Find concurrent execution opportunities
  - Develop application to execute in parallel
  - Run application on parallel hardware
- Is a parallel application a concurrent application?
- Is a parallel application run with one processor parallel?

Parallelism

- There are granularities of (execution) parallelism
  - Processes, threads, routines, statements, instructions, …
  - What are the software elements that execute concurrently
- These must be supported by hardware resources
  - Processors, cores, … (execution of instructions)
  - Memory, DMA, networks, … (other operations)
  - All aspects of computer architecture offer opportunities for parallel hardware
- Concurrency is a necessary condition for parallelism
  - Where can you find concurrency?
  - How is concurrency expressed to exploit parallel systems?

Why use parallel processing?

- Two primary reasons (both performance related)
  - Faster time to solution (response time)
  - Solve bigger computing problems in same time
- Other factors motivate parallel processing
  - Effective use of machine resources
  - Cost efficiencies
  - Overcoming memory constraints
  - Serial machines have inherent limitations
    - Processor speed, memory bottlenecks, …
- Parallelism = concurrency + parallel HW + performance

Perspectives on Parallelism / Parallel Processing

- Parallel computer architecture
  - Hardware needed for parallel execution?
  - Computer system design
- (Parallel) Operating system
- Parallel programming
  - Libraries (low-level, high-level)
  - Languages
  - Software development environments
- Parallel algorithms
- Parallel performance evaluation
- Parallel tools

Why study parallel computing today?

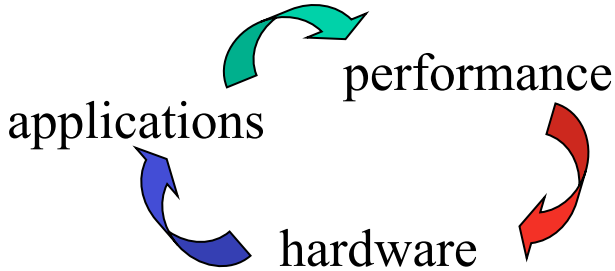
- Computing architecture
  - Innovations often drive to novel programming models
- Technological convergence
  - The "killer micro" is ubiquitous
  - Laptops and supercomputers are fundamentally similar!
  - Trends cause diverse approaches to converge
- Technological trends make parallel computing inevitable
  - Multi-core is here to stay!
- Understand fundamental principles and design tradeoffs
  - Naming, ordering, replication, communication
  - Performance
- Parallelism is the future of computing

Inevitability of Parallel Computing

- Application demands
  - Insatiable need for computing cycles
- Technology trends
  - Processor and memory
- Architecture trends
- Economics
- Current trends:
  - Today's microprocessors have multiprocessor support
  - Servers and workstations available as multiprocessors
  - Tomorrow's microprocessors are multiprocessors
  - Multi-core is here to stay and #cores/processor is growing
  - Accelerators (GPUs, gaming systems)

Application Characteristics

- Application performance demands hardware advances
- Hardware advances generate new applications
- New applications have greater performance requirements
  - Exponential increase in microprocessor performance
  - Innovations in parallel architecture and integration



- Range of performance demands
  - System performance must also improve as a whole
  - Performance requirements require computer engineering
  - Costs addressed through technology advancements

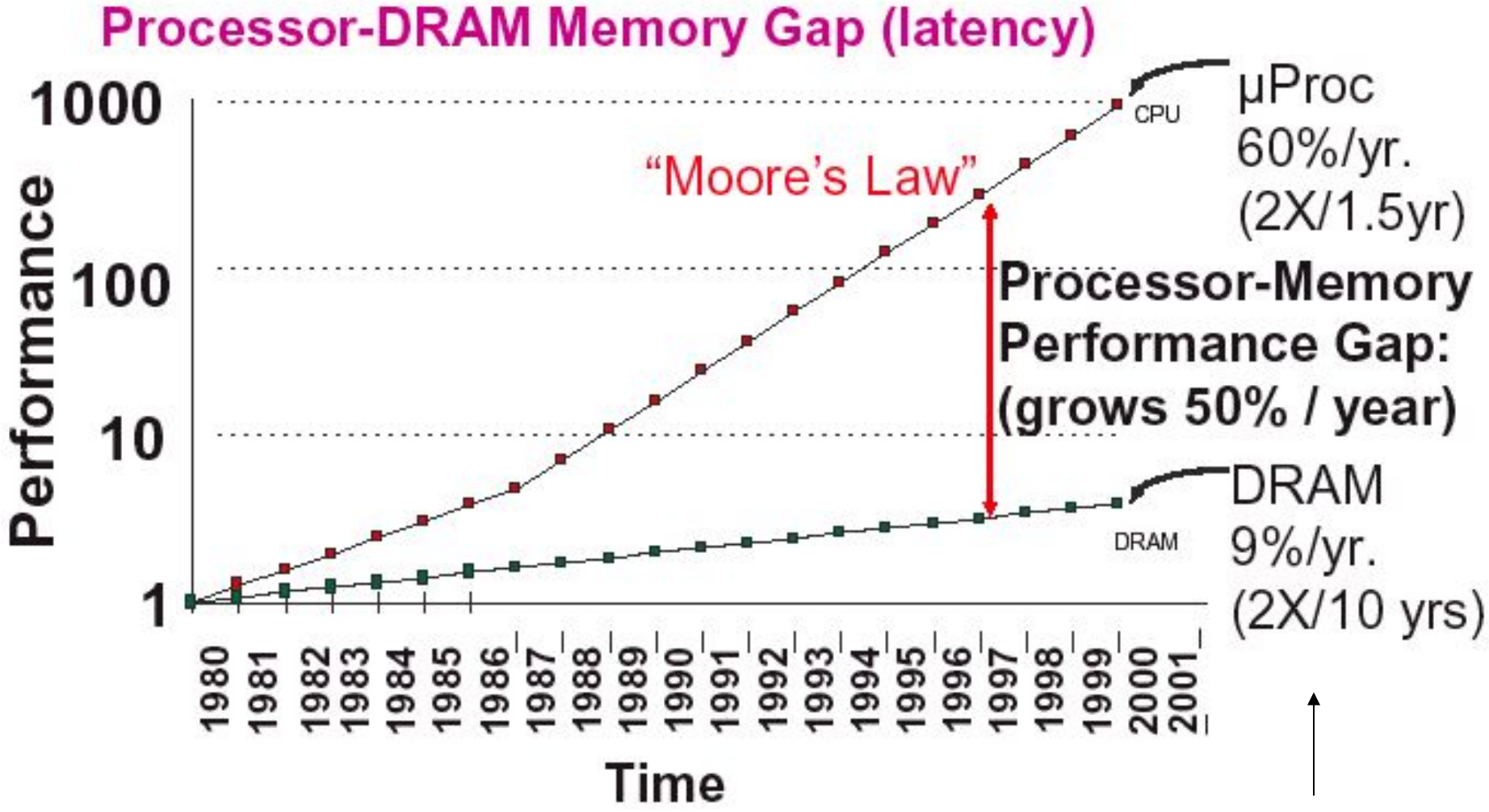
Broad Parallel Architecture Issues

- Resource allocation
  - How many processing elements?
  - How powerful are the elements?
  - How much memory?
- Data access, communication, and synchronization
  - How do the elements cooperate and communicate?
  - How are data transmitted between processors?
  - What are the abstractions and primitives for cooperation?
- Performance and scalability
  - How does it all translate into performance?
  - How does it scale?

Leveraging Moore's Law

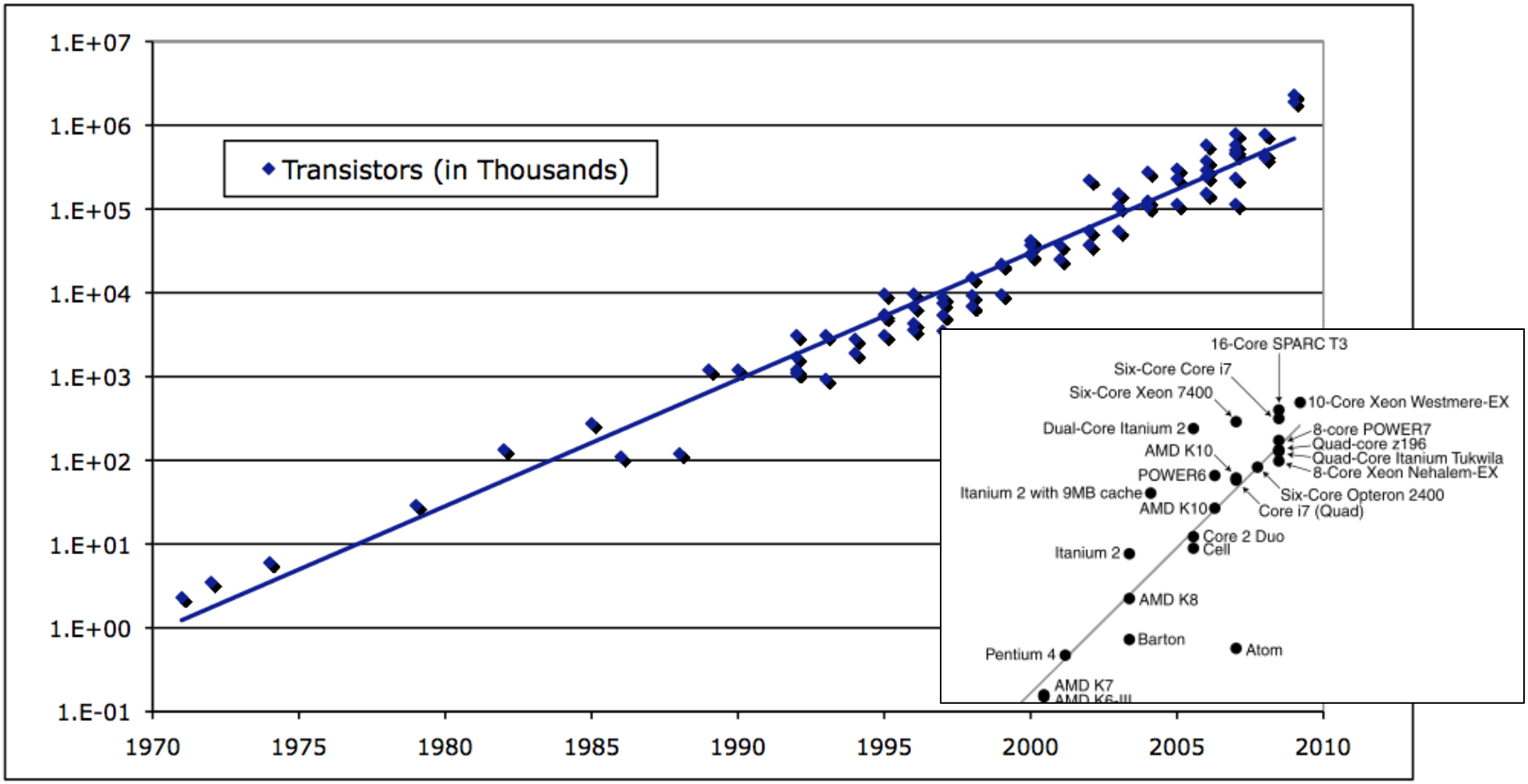
- More transistors = more parallelism opportunities
- Microprocessors
  - Implicit parallelism
    - pipelining
    - multiple functional units
    - superscalar
  - Explicit parallelism
    - SIMD instructions
    - long instruction works

What's Driving Parallel Computing Architecture?



von Neumann bottleneck!!

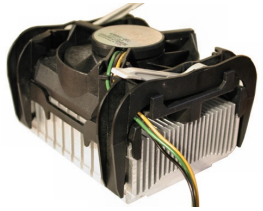
Microprocessor Transitor Counts (1971-2011)



Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanoviç
Slide from Kathy Yelick

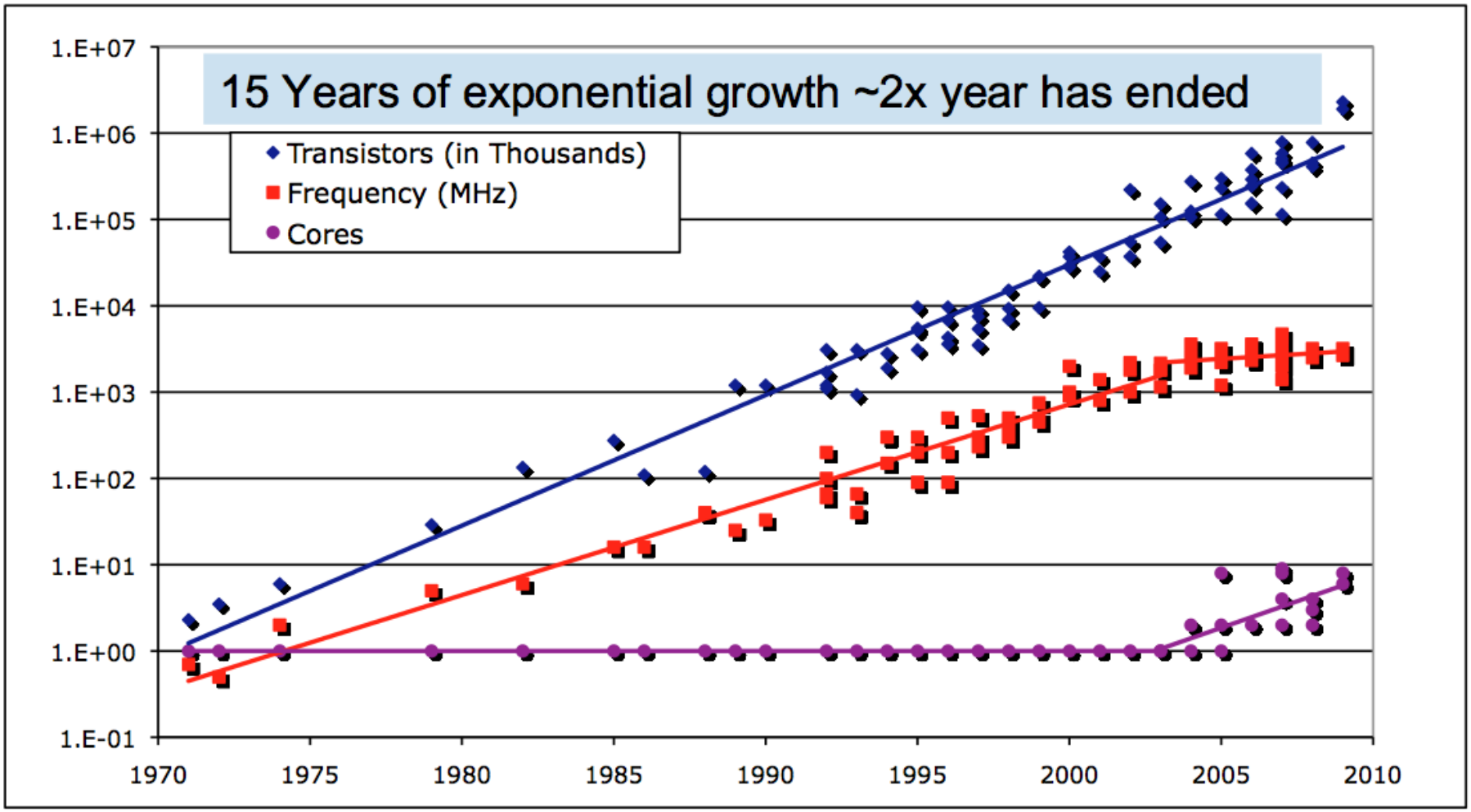
What has happened in the last several years?

- Processing chip manufacturers increased processor performance by increasing CPU clock frequency
  - Riding Moore's law
- Until the chips got too hot!
  - Greater clock frequency ⇒ greater electrical power
  - Pentium 4 heat sink
  - Frying an egg on a Pentium 4



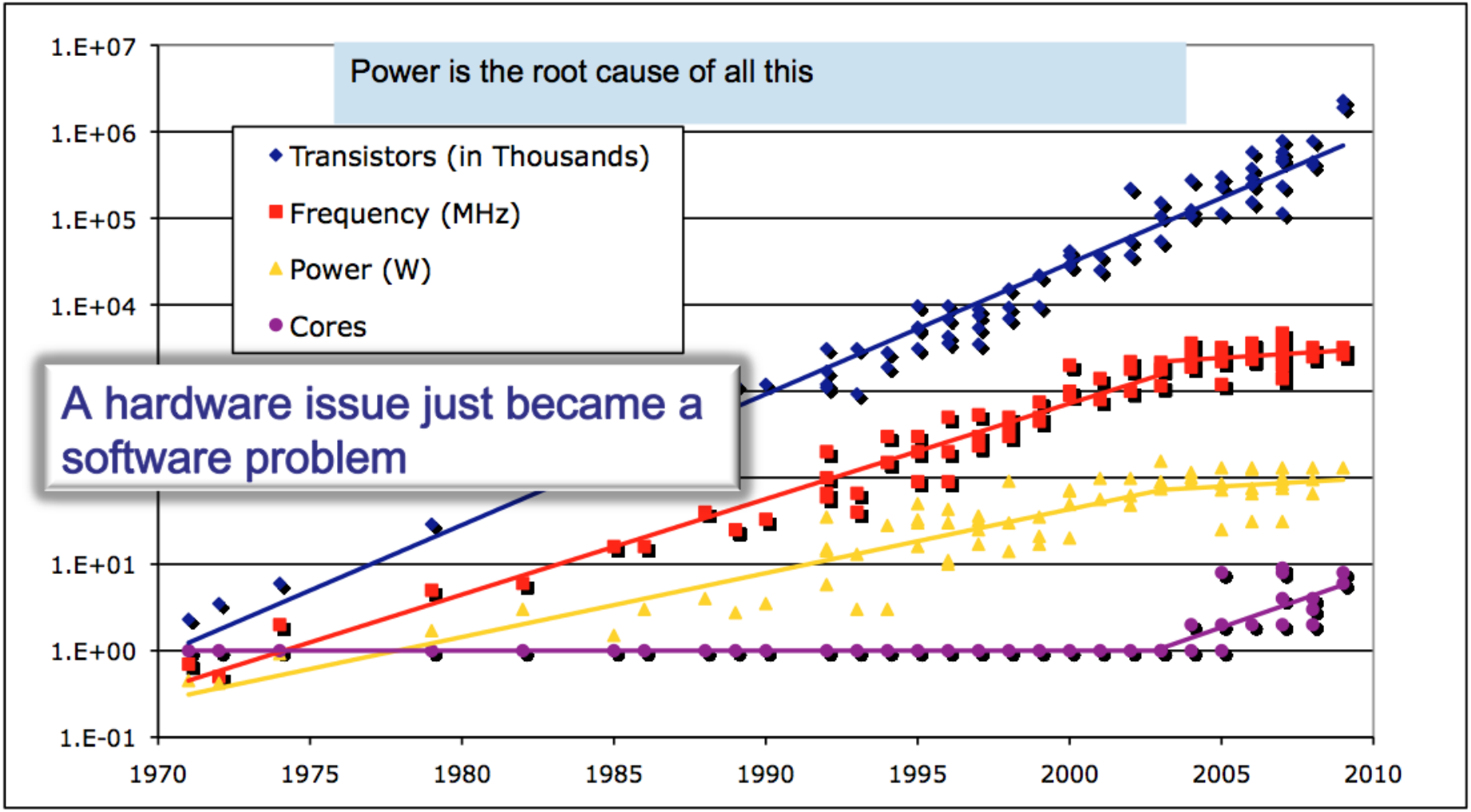
- Add multiple cores to add performance
  - Keep clock frequency same or reduced
  - Keep lid on power requirements

What's Driving Parallel Computing Architecture?



Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanoviç
Slide from Kathy Yelick

What's Driving Parallel Computing Architecture?



Classifying Parallel Systems – Flynn's Taxonomy

- Distinguishes multi-processor computer architectures along the two independent dimensions
  - ***Instruction*** and ***Data***
  - Each dimension can have one state: ***Single*** or ***Multiple***
- SISD: Single Instruction, Single Data
  - Serial (non-parallel) machine
- SIMD: Single Instruction, Multiple Data
  - Processor arrays and vector machines
- MISD: Multiple Instruction, Single Data (weird)
- MIMD: Multiple Instruction, Multiple Data
  - Most common parallel computer systems

Parallel Architecture Types

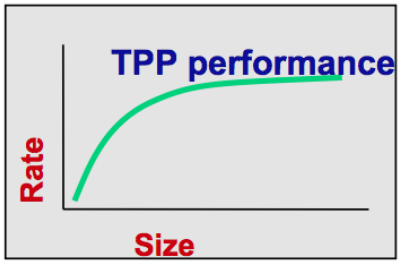
- Instruction-Level Parallelism
  - Parallelism captured in instruction processing
- Vector processors
  - Operations on multiple data stored in vector registers
- Shared-memory Multiprocessor (SMP)
  - Multiple processors sharing memory
  - Symmetric Multiprocessor (SMP)
- Multicomputer
  - Multiple computer connect via network
  - Distributed-memory cluster
- Massively Parallel Processor (MPP)

Phases of Supercomputing (Parallel) Architecture

- Phase 1 (1950s): sequential instruction execution
- Phase 2 (1960s): sequential instruction issue
  - Pipeline execution, reservations stations
  - Instruction Level Parallelism (ILP)
- Phase 3 (1970s): vector processors
  - Pipelined arithmetic units
  - Registers, multi-bank (parallel) memory systems
- Phase 4 (1980s): SIMD and SMPs
- Phase 5 (1990s): MPPs and clusters
  - Communicating sequential processors
- Phase 6 (>2000): many cores, accelerators, scale, …

Top 500 Linpack Benchmarking Methodology

- Listing of the 500 most powerful computers in the world
- Yardstick
  - R_{max} : maximal performance Linpack benchmark
    - Dense linear system of equations ($Ax = b$)
- Data listed
  - R_{peak} : theoretical peak performance
  - N_{max} : problem size needed to achieve R_{max}
  - $N_{1/2}$: problem size needed to achieve 1/2 of R_{max}
  - Manufacturer and computer type
  - Installation site, location, and year
- Updated twice a year: SCxx and ISCxx



Performance Development

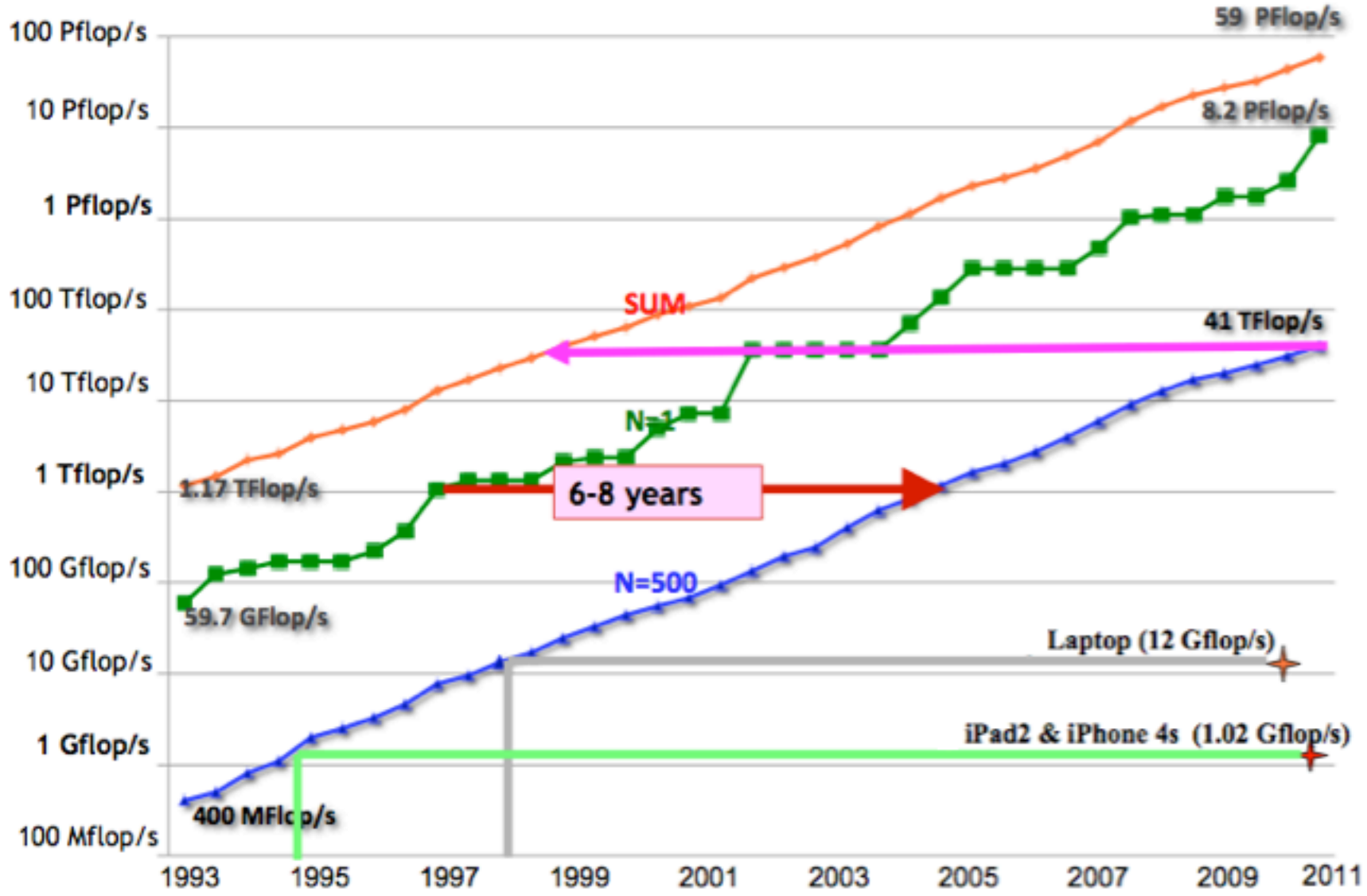


Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/korea-2011.pdf

Top 10 (Top500 List, November 2012)

Rank	Site	Computer	Country	Cores	Rmax [Pflops]	% of Peak	Power [MW]	MFlops /Watt
1	DOE / OS Oak Ridge Nat Lab	Titan, Cray XK7 (16C) + Nvidia Kepler GPU (14c) + custom	USA	560,640	17.6	66	8.3	2120
2	DOE / NNSA L Livermore Nat Lab	Sequoia, BlueGene/Q (16c) + custom	USA	1,572,864	16.3	81	7.9	2063
3	RIKEN Advanced Inst for Comp Sci	K computer Fujitsu SPARC64 VIIIfx (8c) + custom	Japan	705,024	10.5	93	12.7	827
4	DOE / OS Argonne Nat Lab	Mira, BlueGene/Q (16c) + custom	USA	786,432	8.16	81	3.95	2066
5	Forschungszentrum Juelich	JuQUEEN, BlueGene/Q (16c) + custom	Germany	393,216	4.14	82	1.97	2102
6	Leibniz Rechenzentrum	SuperMUC, Intel (8c) + IB	Germany	147,456	2.90	90*	3.42	848
7	Texas Advanced Computing Center	Stampede, Dell Intel (8) + Intel Xeon Phi (61) + IB	USA	204,900	2.66	67	3.3	806
8	Nat. SuperComputer Center in Tianjin	Tianhe-1A, NUDT Intel (6c) + Nvidia Fermi GPU (14c) + custom	China	186,368	2.57	55	4.04	636
9	CINECA	Fermi, BlueGene/Q (16c) + custom	Italy	163,840	1.73	82	.822	2105
10	IBM	DARPA Trial System, Power7 (8C) + custom	USA	63,360	1.51	78	.358	422

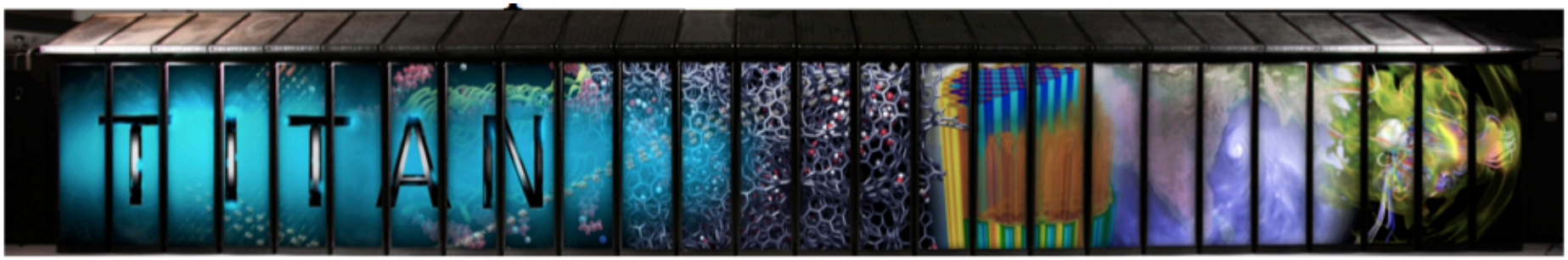
Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/sc2012-nvidia-3.pdf

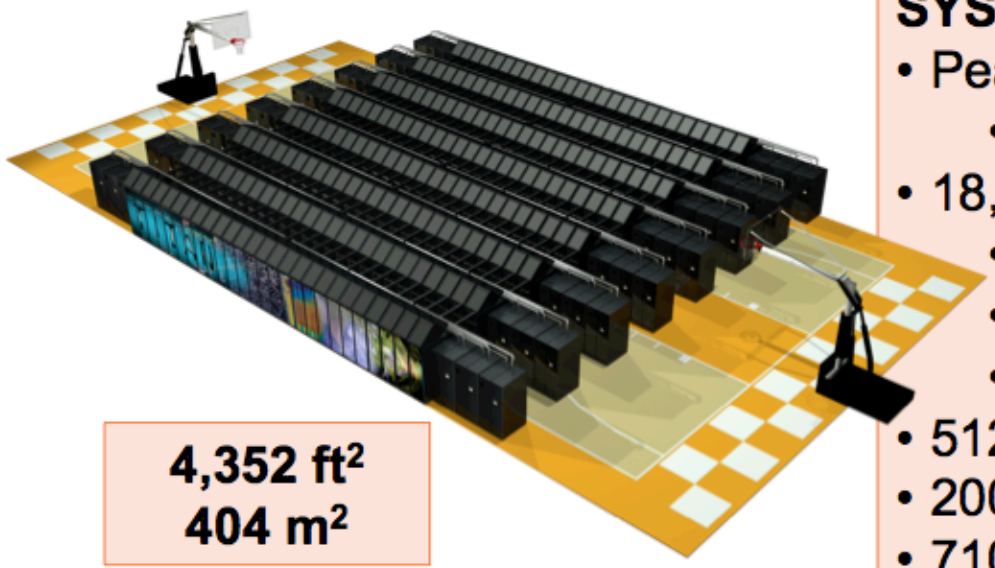
Top 10 (Top500 List, June 2011)

Rank	Site	Computer	Country	Cores	Rmax [Pflops]	% of Peak
1	RIKEN Advanced Inst for Comp Sci	K Computer Fujitsu SPARC64 VIIIfx + custom	Japan	548,352	8.16	93
2	Nat. SuperComputer Center in Tianjin	Tianhe-1A, NUDT Intel + Nvidia GPU + custom	China	186,368	2.57	55
3	DOE / OS Oak Ridge Nat Lab	Jaguar, Cray AMD + custom	USA	224,162	1.76	75
4	Nat. Supercomputer Center in Shenzhen	Nebulea, Dawning Intel + Nvidia GPU + IB	China	120,640	1.27	43
5	GSIC Center, Tokyo Institute of Technology	Tusbame 2.0, HP Intel + Nvidia GPU + IB	Japan	73,278	1.19	52
6	DOE / NNSA LANL & SNL	Cielo, Cray AMD + custom	USA	142,272	1.11	81
7	NASA Ames Research Center/NAS	Plelades SGI Altix ICE 8200EX/8400EX + IB	USA	111,104	1.09	83
8	DOE / OS Lawrence Berkeley Nat Lab	Hopper, Cray AMD + custom	USA	153,408	1.054	82
9	Commissariat a l'Energie Atomique (CEA)	Tera-10, Bull Intel + IB	France	138,368	1.050	84
10	DOE / NNSA Los Alamos Nat Lab	Roadrunner, IBM AMD + Cell GPU + IB	USA	122,400	1.04	76

Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/korea-2011.pdf

Oak Ridge National Lab (ORNL) Titan





4,352 ft²
404 m²

Cray XK7

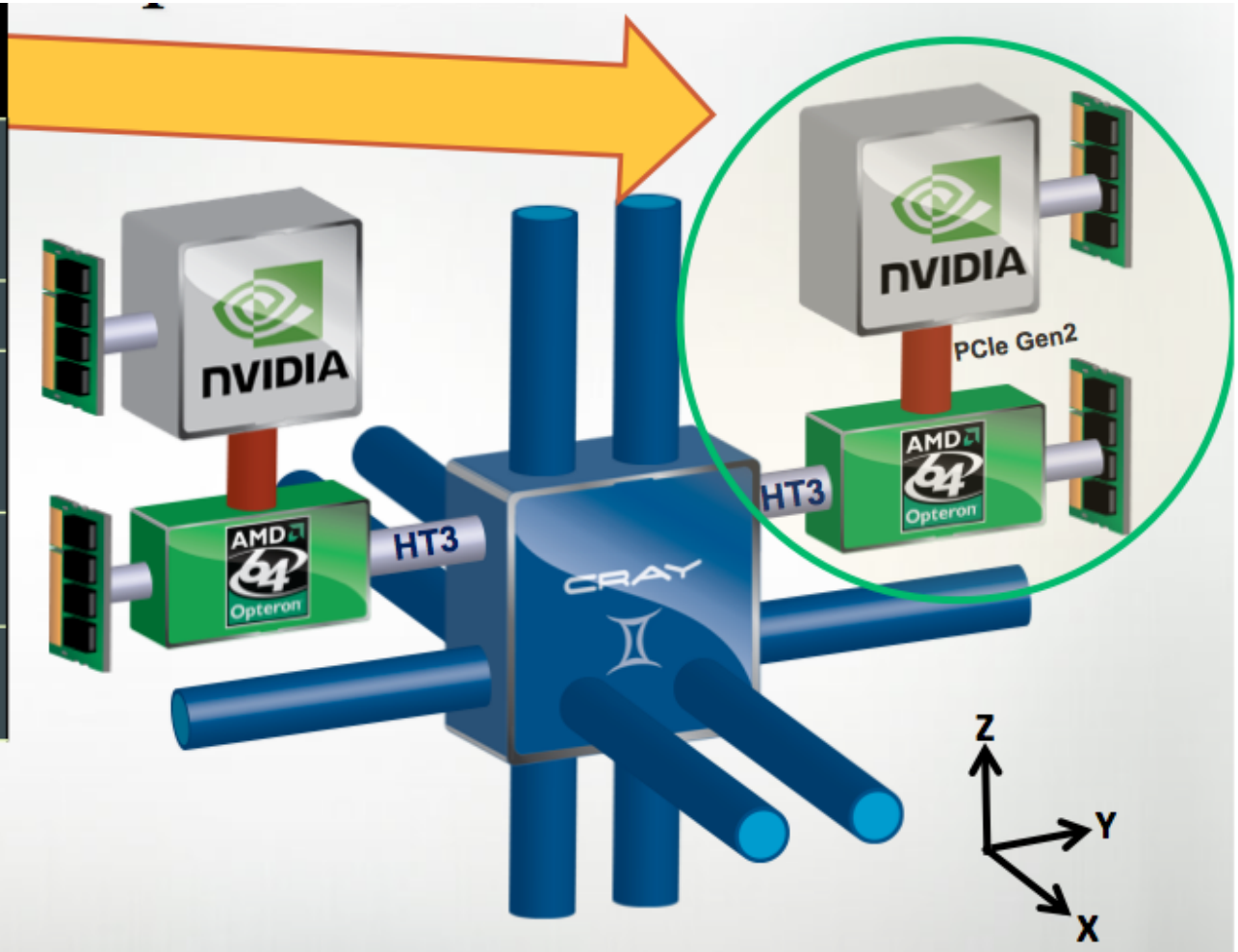
SYSTEM SPECIFICATIONS:

- Peak performance of 27 PF
  - 24.5 Pflop/s GPU + 2.6 Pflop/s AMD
- 18,688 Compute Nodes each with:
  - 16-Core AMD Opteron CPU
  - NVIDIA Tesla "K20x" GPU
  - 32 + 6 GB memory
- 512 Service and I/O nodes
- 200 Cabinets
- 710 TB total system memory
- Cray Gemini 3D Torus Interconnect
- 9 MW peak power

OAK RIDGE National Laboratory

Cray XK7 Node

XK7 Compute Node Characteristics
AMD Opteron 6274 Interlagos 16 core processor
Tesla K20x @ 1311 GF
Host Memory 32GB 1600 MHz DDR3
Tesla K20x Memory 6GB GDDR5
Gemini High Speed Interconnect



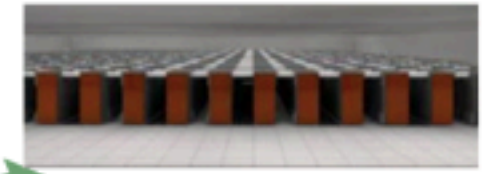
Japanese K Computer

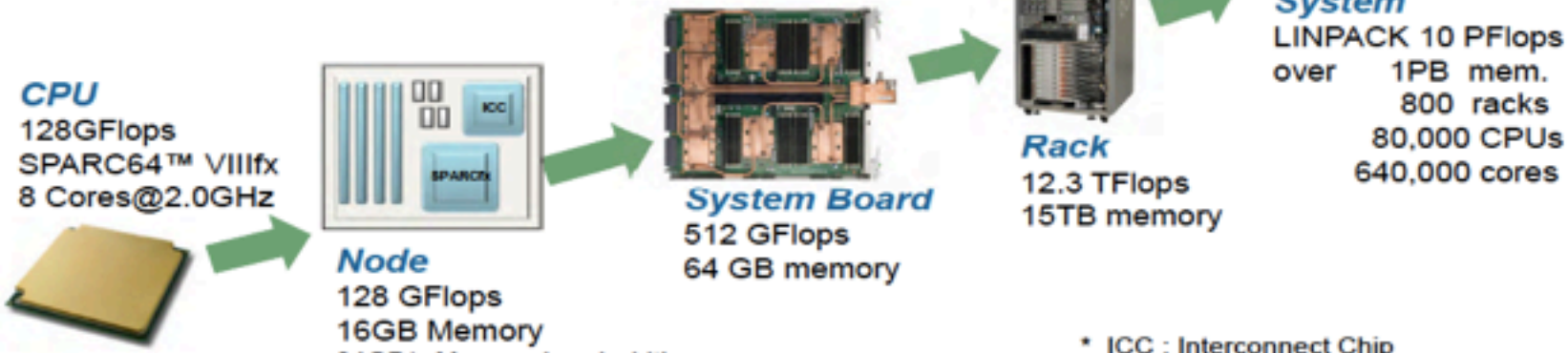
K computer Specifications

FUJITSU

CPU (SPARC64 VIIIfx)	Cores/Node	8 cores (@2GHz)
	Performance	128GFlops
	Architecture	SPARC V9 + HPC extension
	Cache	L1(I/D) Cache : 32KB/32KB L2 Cache : 6MB
	Power	58W (typ. 30 C)
	Mem. bandwidth	64GB/s.
Node	Configuration	1 CPU / Node
	Memory capacity	16GB (2GB/core)
System board(SB)	No. of nodes	4 nodes /SB
Rack	No. of SB	24 SBs/rack
System	Nodes/system	> 80,000

Inter-connect	Topology	6D Mesh/Torus
	Performance	5GB/s. for each link
	No. of link	10 links/ node
	Additional feature	H/W barrier, reduction
	Architecture	Routing chip structure (no outside switch box)
Cooling	CPU, ICC*	Direct water cooling
	Other parts	Air cooling





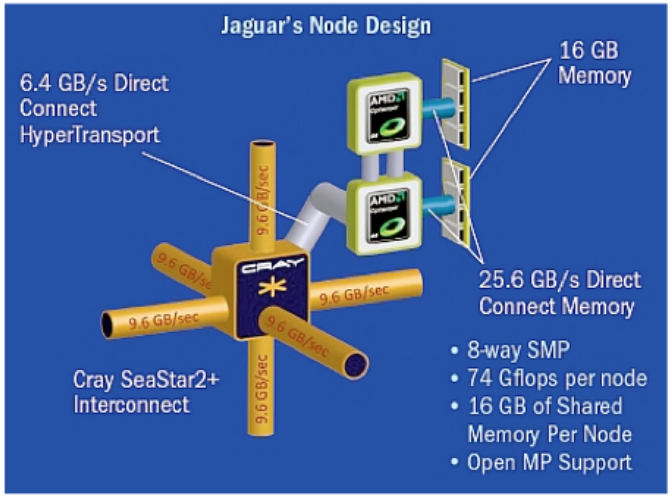
* ICC : Interconnect Chip

New Linpack run with 705,024 cores at 10.51 Pflop/s (88,128 CPUs)

Oak Ridge National Lab (ORNL) Jaguar

Recently upgraded to a 2 Pflop/s system with more than 224K cores using AMD's 6 Core chip.

Peak performance	**2.332 PF**
System memory	300 TB
Disk space	10 PB
Disk bandwidth	240+ GB/s
Interconnect bandwidth	374 TB/s



U.S. DEPARTMENT OF ENERGY | Office of Science

Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/sc09-ms.pdf

Los Alamos National Lab (LANL) Roadrunner

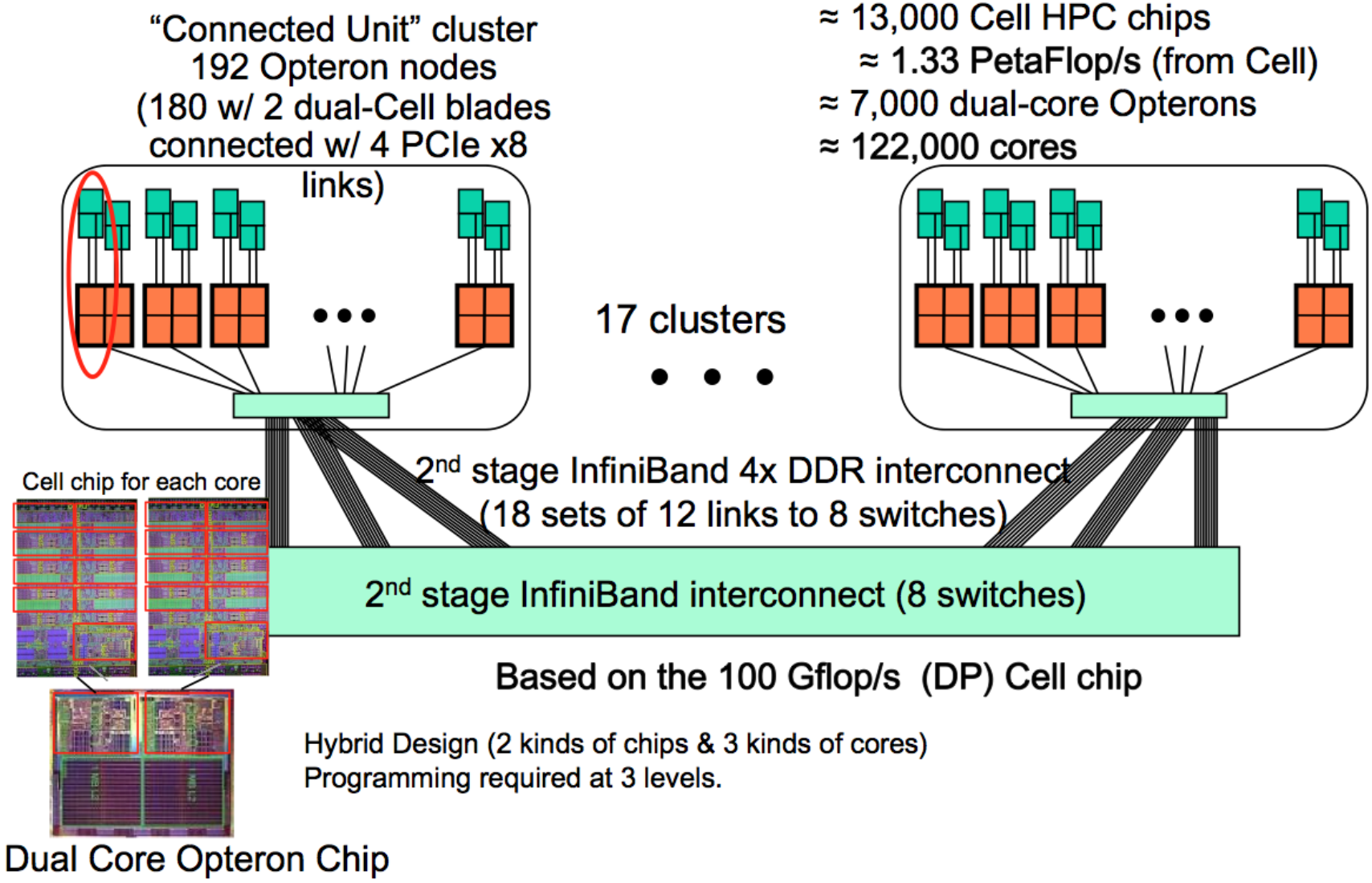


Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/sc09-ms.pdf

Research Centre Juelich Jugene

- IBM BG/P
  - 73,728 nodes
  - 4-way SMP, 32-big PowerPC 450, 850 MHz
  - 294,912 cores

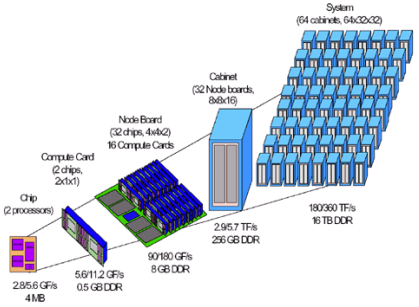


Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/sc09-ms.pdf

Performance of Top20 over 10 Years

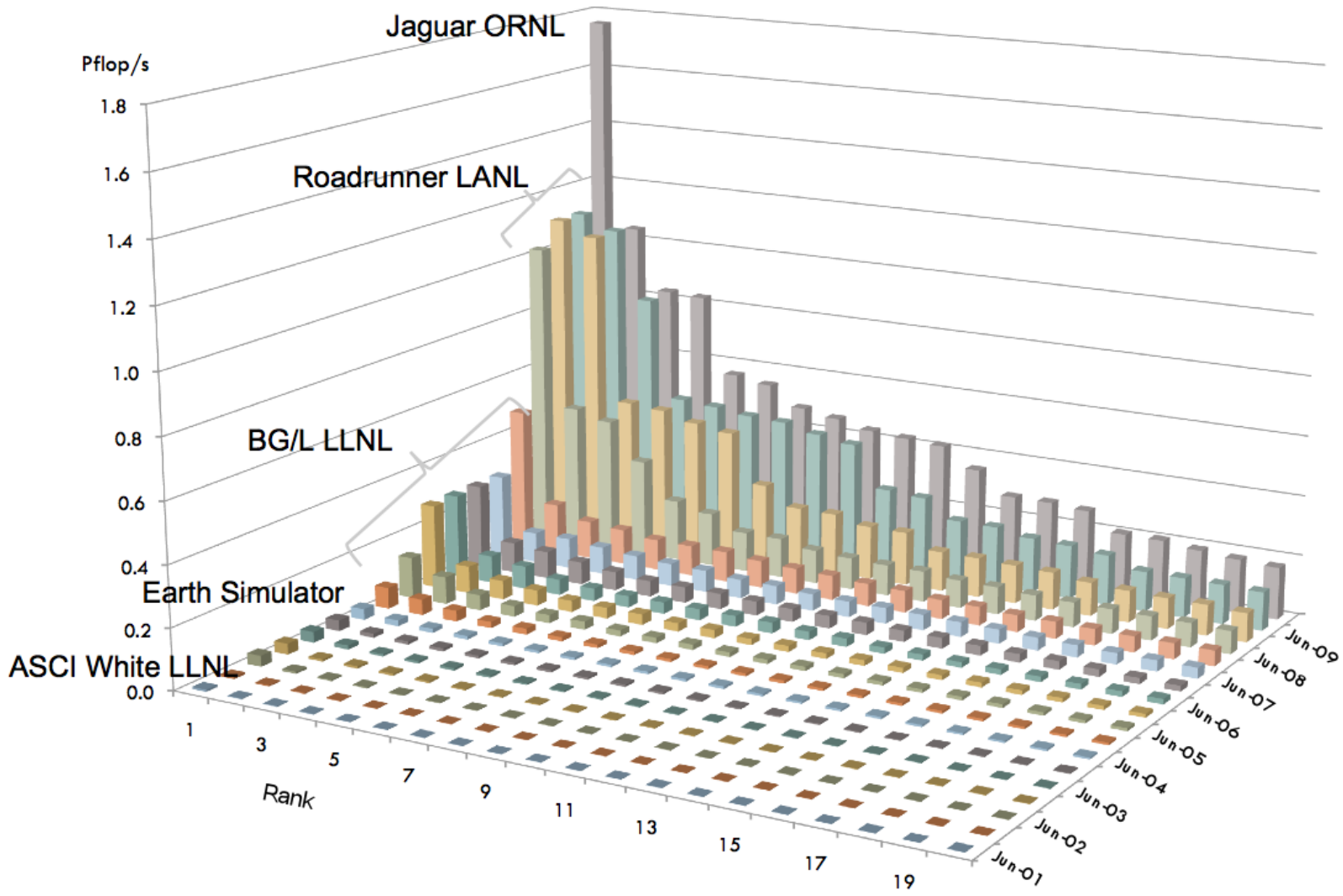


Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/sc09-ms.pdf

Top 500 Top 10 (2006)

	Manufacturer	Computer	Rmax [TF/s]	Installation Site	Country	Year	#Proc
1	IBM	BlueGene/L eServer Blue Gene	280.6	DOE/NNSA/LLNL	USA	2005	131,072
2	Sandia/Cray	Red Storm Cray XT3	101.4	NNSA/Sandia	USA	2006	26,544
3	IBM	BGW eServer Blue Gene	91.29	IBM Thomas Watson	USA	2005	40,960
4	IBM	ASC Purple eServer pSeries p575	75.76	DOE/NNSA/LLNL	USA	2005	12,208
5	IBM	MareNostrum JS21 Cluster, Myrinet	62.63	Barcelona Supercomputing Center	Spain	2006	12,240
6	Dell	Thunderbird PowerEdge 1850, IB	53.00	NNSA/Sandia	USA	2005	9,024
7	Bull	Tera-10 NovaScale 5160, Quadrics	52.84	CEA	France	2006	9,968
8	SGI	Columbia Altix, Infiniband	51.87	NASA Ames	USA	2004	10,160
9	NEC/Sun	Tsubame Fire x4600, ClearSpeed, IB	47.38	GSIC / Tokyo Institute of Technology	Japan	2006	11,088
10	Cray	Jaguar Cray XT3	43.48	ORNL	USA	2006	10,424

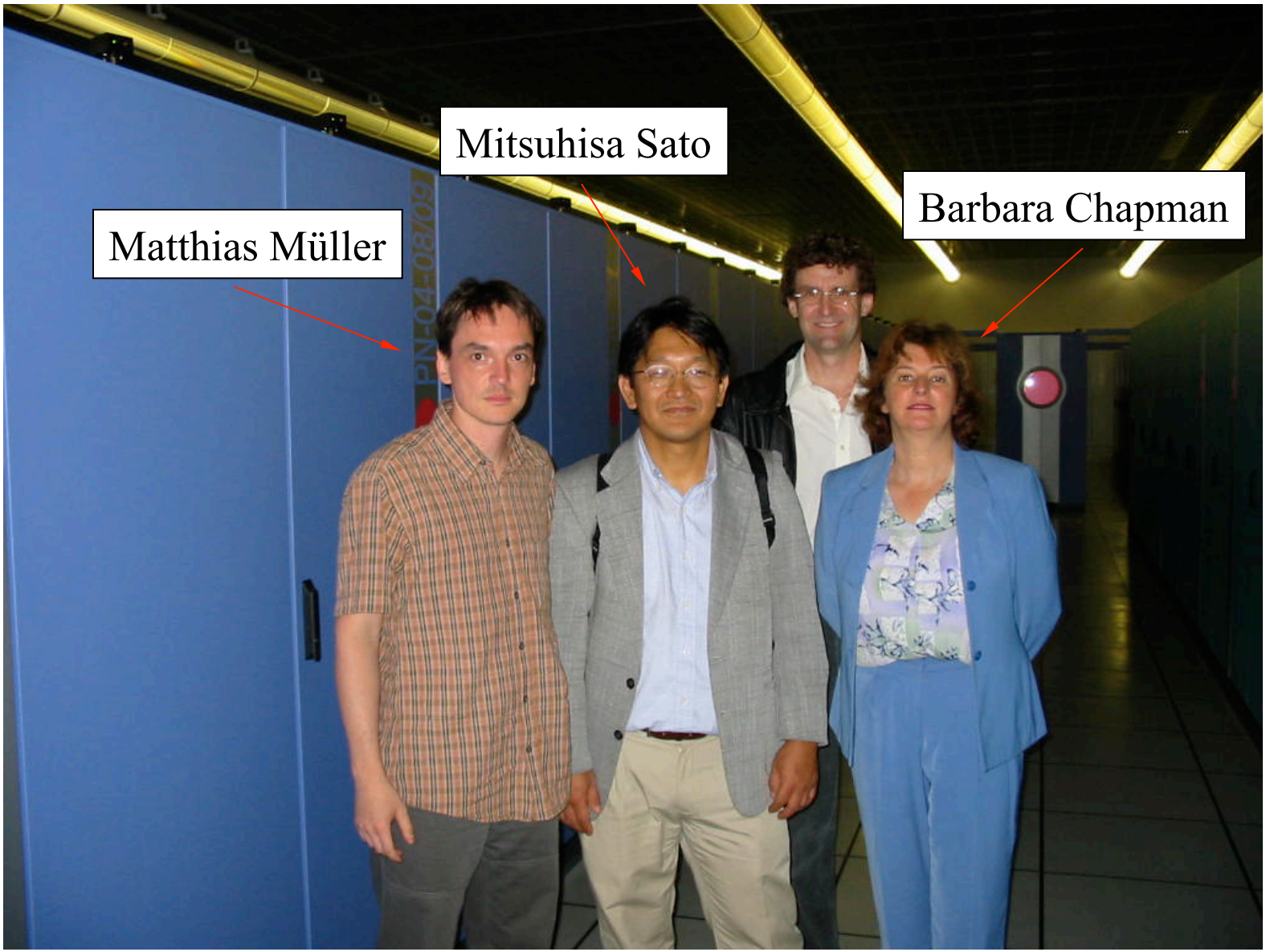
Top 500 Linpack Benchmark List (June 2002)

Computer (Full Precision)		Number of Processors	R_{max} Gflop/s	N_{max} order	$N_{1/2}$ order	R_{peak} Gflop/s
☆Earth Simulator, NEC processors****	esc	5104	35610	1041216	265408	40832
ASCI White-Pacific, IBM SP Power 3(375 MHz)	llnl	8000	7226	518096	179000	12000
☆Compaq AlphaServer SC ES45/EV68 1GHz	psc	3016	4463	280000	85000	6032
Compaq AlphaServer SC ES45/EV68 1GHz	psc	3024	4059	525000	105000	6048
☆Compaq AlphaServer SC ES45/EV68 1GHz	cea	2560	3980	360000	85000	5120
IBM SP Power3 208 nodes 375 MHz	lbnl	3328	3052.	371712		4992
☆Compaq Alphaserver SC ES45/EV68 1GHz	lanl	2048	2916	272000		4096
☆IBM SP Power3 158 nodes 375 MHz	lbnl	2528	2526.	371712	102400	3792
ASCI Red Intel Pentium II Xeon core 333MHz	snl	9632	2379.6	362880	75400	3207
ASCI Blue-Pacific SST, IBM SP 604E(332 MHz)	llnl	5808	2144.	431344	432344	3868
ASCI Red Intel Pentium II Xeon core 333MHz	snl	9472	2121.3	251904	66000	3154
Compaq Alphaserver SC ES45/EV68 1GHz	lanl	1520	2096	390000	71000	3040
☆IBM SP 112 nodes (375 MHz POWER3 High)	ibm	1792	1791	275000	275000	2688
HITACHI SR8000/MPP/1152(450MHz)	u toyko	1152	1709.1	141000	16000	2074
☆HITACHI SR8000-F1/168(375MHz)	leibniz	168	1653.	160000	19560	2016
ASCI Red Intel Pentium II Xeon core 333Mhz	snl	6720	1633.3	306720	52500	2238
SGI ASCI Blue Mountain	lanl	5040	1608.	374400	138000	2520
IBM SP 328 nodes (375 MHz POWER3 Thin)	noo	1312	1417.	374000	374000	1968
Intel ASCI Option Red (200 MHz Pentium Pro)	snl	9152	1338.	235000	63000	1830
NEC SX-5/128M8(3.2ns)	osaka	128	1192.0	129536	10240	1280
CRAY T3E-1200 (600 MHz)	us government	1488	1127.	148800	28272	1786
HITACHI SR8000-F1/112(375MHz)	leibniz	112	1035.0	120000	15160	1344

Japanese Earth Simulator

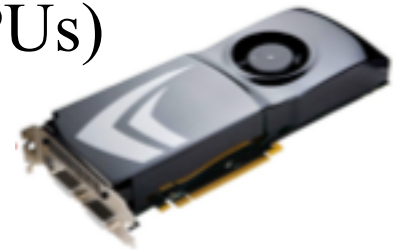
- World's fastest supercomputer!!! (2002)
  - 640 NEC SX-6 nodes
    - 8 vector processors
  - 5104 total processors
  - Single stage crossbar
    - ~2900 meters of cables
  - 10 TB memory
  - 700 TB disk space
  - 1.6 PB mass storage
  - 40 Tflops peak performance
  - 35.6 Tflops Linpack performance

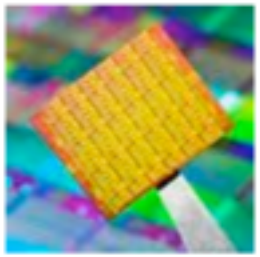
Prof. Malony and colleagues at Japanese ES



Future Computer Systems

- Mostly likely will be a hybrid design
  - Standard multicore chips and accelerators (GPUs)
- Today accelerators are attached
- Next generation will be integrated
- Intel's MIC architecture
  - "Knights Ferry" and "Knights Corner"
  - 48 x86 cores
- AMD's Fusion
  - Multicore with embedded graphics (ATI)
- NVIDIA's Project Denver
  - Integrated chip using ARM architecture (2013)





AMD
The future is fusion

Performance Development in Top 500

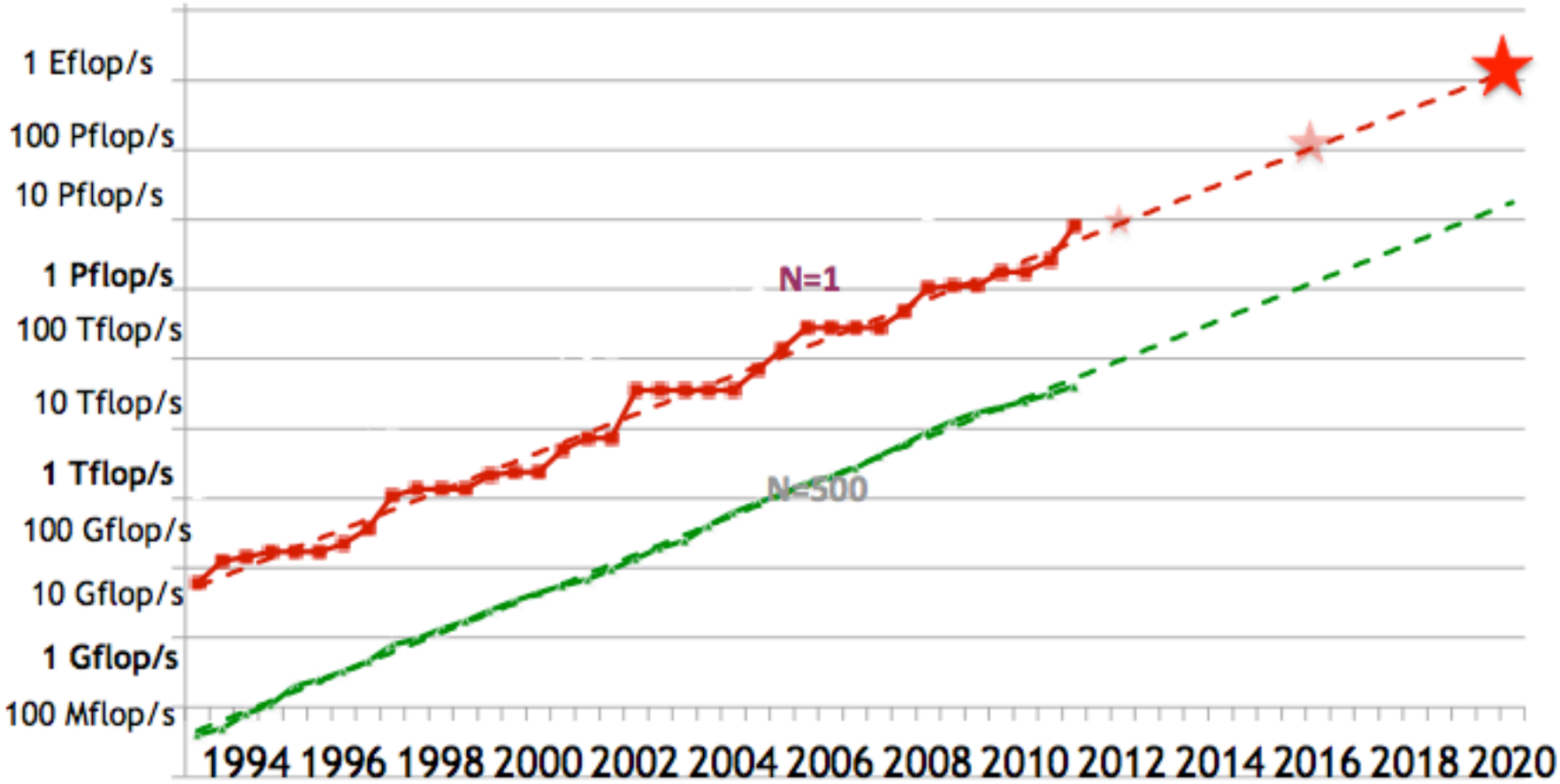


Figure credit: http://www.netlib.org/utk/people/JackDongarra/SLIDES/korea-2011.pdf

Exascale Initiative

- DOE Exascale initiative (since 2007)
  http://science.energy.gov/ascr/news-and-resources/program-documents
- Exascale machines are targeted for the 2019 timeframe
- What are the potential differences and problems?

Systems	2011 K Computer	2019	Difference Today & 2019
System peak	8.7 Pflop/s	1 Eflop/s	O(100)
Power	10 MW	~20 MW	???
System memory	1.6 PB	32 - 64 PB	O(10)
Node performance	128 GF	1,2 or 15TF	O(10) - O(100)
Node memory BW	64 GB/s	2 - 4TB/s	O(100)
Node concurrency	8	O(1k) or 10k	O(100) - O(1000)
Total Node Interconnect BW	20 GB/s	200-400GB/s	O(10)
System size (nodes)	68,544	O(100,000) or O(1M)	O(10) - O(100)
Total concurrency	548,352	O(billion)	O(1,000)
MTTI	days	O(1 day)	- O(10)

Major Changes to Software and Algorithms

- What were we concerned about before and now?
- Must rethink the design for exascale
  - Data movement is expensive (Why?)
  - Flops per second are cheap (Why?)
- Need to reduce communication and sychronization
- Need to develop fault-resilient algorithms
- How do with deal with massive parallelism?
- Software must adapt to the hardware (autotuning)

International Exascale Software Project (IESP)

- Develop a plan for producing a software infrastructure capable of supporting exascale applications
- International effort to engage the HPC community and vendors
- What to build?
- How to build?
- Strategic plan

IESP

China | Japan SDHPC | EU-EESI | US-DOE | US-NSF

www.exascale.org

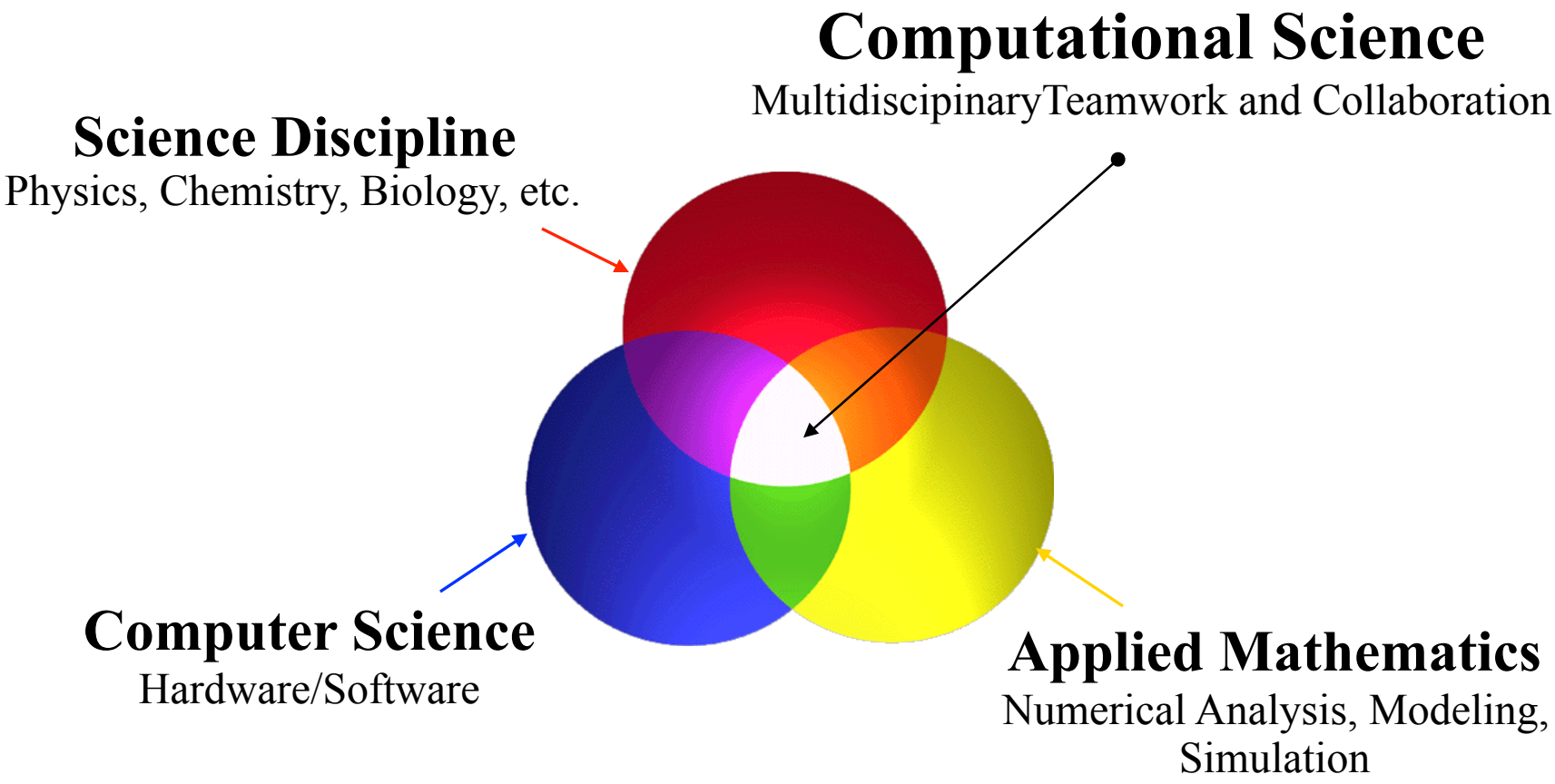
Supercomputing and Computational Science

- By definition, a supercomputer is of a class of computer systems that are the most powerful computing platforms at that time
- Computational science has always lived at the leading (and bleeding) edge of supercomputing technology
- "Most powerful" depends on performance criteria
  - Performance metrics related to computational algorithms
  - Benchmark "real" application codes
- Where does the performance come from?
  - More powerful processors
  - More processors (cores)
  - Better algorithms

Computational Science

- Traditional scientific methodology
  - Theoretical science
    - Formal systems and theoretical models
    - Insight through abstraction, reasoning through proofs
  - Experimental science
    - Real system and empirical models
    - Insight from observation, reasoning from experiment design
- Computational science
  - Emerging as a principal means of scientific research
  - Use of computational methods to model scientific problems
    - Numerical analysis plus simulation methods
    - Computer science tools
  - Study and application of these solution techniques

What is Computational Science?



21st Century Science and Engineering Practice

- Three-fold science
  - theory
  - experiment
  - computational simulation
- Supported by
  - multimodal collaboration systems
  - distributed, multi-petabyte data archives
  - leading edge computing systems (*)
  - distributed experimental facilities
  - internationally distributed multidisciplinary teams
- Collectively defining a new future
  - creation of 21st century IT infrastructure
  - sustainable, multidisciplinary communities

Computational Challenges

- Computational science thrives on computer power
  - Faster solutions
  - Finer resolution
  - Bigger problems
  - Improved interaction
  - ⇒ BETTER SCIENCE!!!
- How to get more computer power?
  - *Scalable parallel computing*
- Computational science also thrives better integration
  - Couple computational resources
  - Grid computing

Amdahl's Law

- T_{seq} : sequential execution time that cannot be parallelized
- T_{par} : sequential execution time that can be parallelized
- $T_1 = T_{seq} + T_{par} \Rightarrow T_{par} = T_1 - T_{seq}$
- $T_p = T_{seq} + T_{par} / p$ (assume fully parallelized)
- As $p \rightarrow \infty$, $T_p \rightarrow T_{seq}$
- Let f_{seq} be the fraction T_{seq} / T_1 and $S_p = T_1 / T_p$
- $Speedup = S_p = T_1 / T_p = T_1 / (T_{seq} + T_{par} / p)$
  $= 1 / (f_{seq} + T_{par} / pT_1) = 1 / (f_{seq} + (1 - f_{seq}) / p)$
  - As $p \rightarrow \infty$, $S_p = S_\infty \rightarrow 1 / f_{seq}$
- Speedup bound is determined by the degree of sequential execution time in the computation, not # processors!!!

Amdahl's Law and Scaled Speedup

- Amdahl's Law makes it hard to obtain good speedup

f_{seq} * 100%	10%	5%	2%	1%	.1%
S_∞	10	20	50	100	1000

- Why? Problem size remains fixed (*strong scaling*)
- Scale problem size as # processors scale (*weak scaling*)
- T_{seq} : sequential execution time (1 and p processors)
- T_{par} : execution time in parallel mode on p processors
- $T_p = T_{seq} + T_{par}$, $T_1 = T_{seq} + pT_{par}$
- Let f_{par} be the fraction T_{par} / T_p
- *Scaled speedup* $= S_p = 1 + (p\text{-}1)T_{par} / T_p = 1 + (p\text{-}1)f_{par}$

Scalable Parallel Computing

- Scalability in parallel architecture
  - Processor numbers
  - Memory architecture
  - Interconnection network
  - Avoid critical architecture bottlenecks
- Scalability in computational problem
  - Problem size
  - Computational algorithms
    - Computation to memory access ratio
    - Computation to communication ration
- Parallel programming models and tools
- Performance scalability

Next Lecture

- Parallel computer architectures
- Parallel performance models