CS267
Applications of Parallel Computers

www.cs.berkeley.edu/~demmel/cs267_Spr15/

Lecture 1: Introduction

Jim Demmel

EECS & Math Departments

demmel@berkeley.edu

Outline

- Why ~~powerful~~ all computers must be parallel processors
  - Including your laptops and handhelds
- Large Computational Science and Engineering (CSE) problems require powerful computers
  - Commercial problems too
- Why writing (fast) parallel programs is hard
  - But things are improving
- Structure of the course

Units of Measure

- **High Performance Computing (HPC) units are:**
  - Flop: floating point operation, usually double precision unless noted
  - Flop/s: floating point operations per second
  - Bytes: size of data (a double precision floating point number is 8 bytes)
- **Typical sizes are millions, billions, trillions…**

Mega	Mflop/s = 10^6 flop/sec	Mbyte = 2^{20} = 1048576 ~ 10^6 bytes
Giga	Gflop/s = 10^9 flop/sec	Gbyte = 2^{30} ~ 10^9 bytes
Tera	Tflop/s = 10^{12} flop/sec	Tbyte = 2^{40} ~ 10^{12} bytes
Peta	Pflop/s = 10^{15} flop/sec	Pbyte = 2^{50} ~ 10^{15} bytes
Exa	Eflop/s = 10^{18} flop/sec	Ebyte = 2^{60} ~ 10^{18} bytes
Zetta	Zflop/s = 10^{21} flop/sec	Zbyte = 2^{70} ~ 10^{21} bytes
Yotta	Yflop/s = 10^{24} flop/sec	Ybyte = 2^{80} ~ 10^{24} bytes

- **Current fastest (public) machine ~ 55 Pflop/s, 3.1M cores**
  - Up-to-date list at www.top500.org

Why ~~powerful~~ all computers are parallel

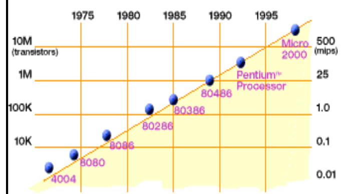
(2007)

circa 1991-2006

Tunnel Vision by Experts

- "I think there is a world market for maybe five computers."
  - Thomas Watson, chairman of IBM, 1943.
- "There is no reason for any individual to have a computer in their home"
  - Ken Olson, president and founder of Digital Equipment Corporation, 1977.
- "640K [of memory] ought to be enough for anybody."
  - Bill Gates, chairman of Microsoft,1981.
- "On several recent occasions, I have been asked whether parallel computing will soon be relegated to the trash heap reserved for promising technologies that never quite make it."
  - Ken Kennedy, CRPC Directory, 1994

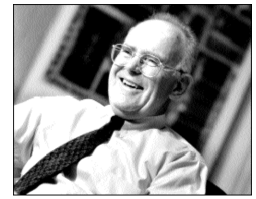
Technology Trends: Microprocessor Capacity



2X transistors/Chip Every 1.5 years

Called "Moore's Law"

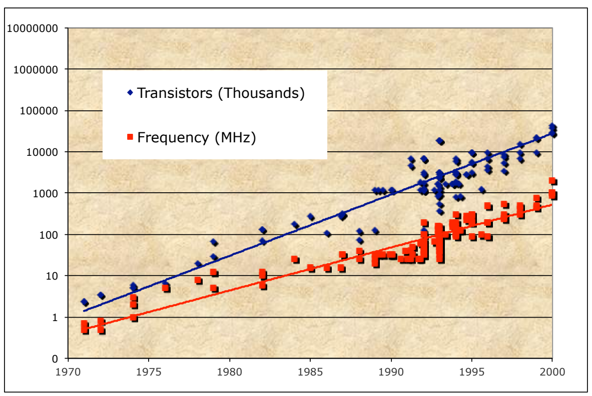
Microprocessors have become smaller, denser, and more powerful.



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

Slide source: Jack Dongarra

Microprocessor Transistors / Clock (1970-2000)

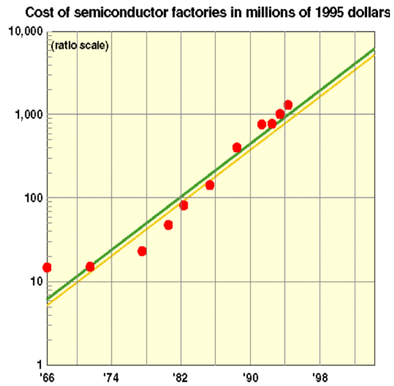


Impact of Device Shrinkage

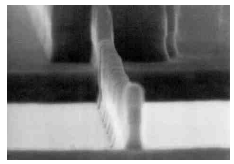
- What happens when the feature size (transistor size) shrinks by a factor of x ?
- Clock rate goes up by x because wires are shorter
  - actually less than x, because of power consumption
- Transistors per unit area goes up by x^2
- Die size also tends to increase
  - typically another factor of ~x
- Raw computing power of the chip goes up by ~ x^4 !
  - typically x^3 is devoted to either on-chip
    - parallelism: hidden parallelism such as ILP
    - locality: caches
- So most programs x^3 times faster, without changing them

Manufacturing Issues Limit Performance

Manufacturing costs and yield problems limit use of density



- Moore's 2nd law (Rock's law): costs go up



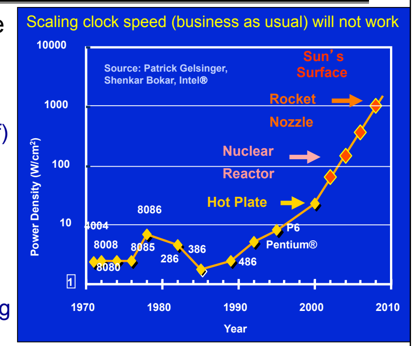
Demo of 0.06 micron CMOS

Source: Forbes Magazine

- **Yield**
  - -What percentage of the chips are usable?
  - -E.g., Cell processor (PS3) was sold with 7 out of 8 "on" to improve yield

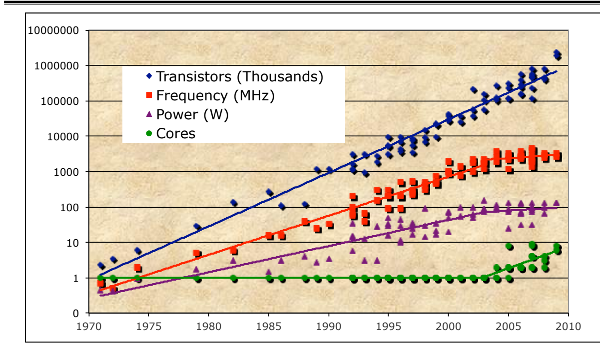
Power Density Limits Serial Performance

- Concurrent systems are more power efficient
  - Dynamic power is proportional to V^2fC
  - Increasing frequency (f) also increases supply voltage (V) → cubic effect
  - Increasing cores increases capacitance (C) but only linearly
  - Save power by lowering clock speed



- High performance serial processors waste power
  - Speculation, dynamic dependence checking, etc. burn power
  - Implicit parallelism discovery
- More transistors, but not faster serial processors

Revolution in Processors



- Chip density is continuing increase ~2x every 2 years
- Clock speed is not
- Number of processor cores may double instead
- Power is under control, no longer growing

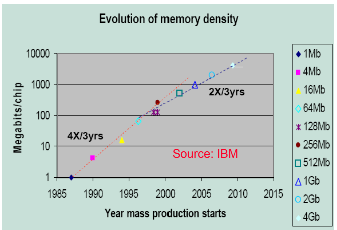
Parallelism in 2015?

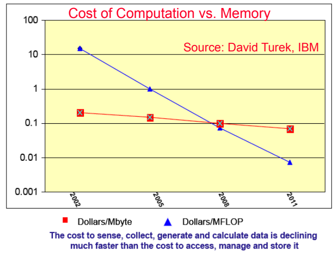
- These arguments are no longer theoretical
- All major processor vendors are producing *multicore* chips
  - Every machine will soon be a parallel machine
  - To keep doubling performance, parallelism must double
- Which (commercial) applications can use this parallelism?
  - Do they have to be rewritten from scratch?
- Will all programmers have to be parallel programmers?
  - New software model needed
  - Try to hide complexity from most programmers – eventually
  - In the meantime, need to understand it
- Computer industry betting on this big change, but does not have all the answers
  - Berkeley ParLab, then ASPIRE, established to work on this

Memory is Not Keeping Pace

Technology trends against a constant or increasing memory per core

- Memory density is doubling every three years; processor logic is every two
- Storage costs (dollars/Mbyte) are dropping gradually compared to logic costs





Question: Can you double concurrency without doubling memory?

- **Strong scaling:** fixed problem size, increase number of processors
- **Weak scaling:** grow problem size proportionally to number of processors

The TOP500 Project

- Listing the 500 most powerful computers in the world
- Yardstick: Rmax of Linpack
  - Solve Ax=b, dense problem, matrix is random
  - Dominated by dense matrix-matrix multiply
- Updated twice a year:
  - ISC'xy in June in Germany
  - SCxy in November in the U.S.
- All information available from the TOP500 web site at: www.top500.org

The TOP10 in November 2014

#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	National University of Defense Technology	NUDT	Tianhe-2 NUDT TH-IVB-FEP, Xeon 12C 2.2GHz, IntelXeon Phi	China	3,120,000	33.9	17.8
2	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.6	8.21
3	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	17.2	7.89
4	RIKEN Advanced Institute for Computational Science	Fujitsu	K Computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	795,024	10.5	12.7
5	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.59	3.95
6	Swiss National Supercomputing Centre (CSCS)	Cray	Piz Daint Cray XC30, Xeon E5 8C 2.6GHz, Aries, NVIDIA K20x	Switzer-land	115,984	6.27	2.33
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	462,462	5.17	4.51
8	Forschungszentrum Juelich (FZJ)	IBM	JuQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	458,752	5.01	2.30
9	Lawrence Livermore National Laboratory	IBM	Vulcan BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	393,216	4.29	1.97
10	Government	Cray	Cray CS-Storm, Xeon E5 10C 2.2GHz, I-FDR, NVIDIDA K40	USA	72,800	3.58	1.50

The TOP10 in November 2014

#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	National University of Defense Technology	NUDT	Tianhe-2 NUDT TH-IVB-FEP, Xeon 12C 2.2GHz, IntelXeon Phi	China	3,120,000	33.9	17.8
2	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.6	8.21
3	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	17.2	7.89
4	RIKEN Advanced Institute for Computational Science	Fujitsu	K Computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	795,024	10.5	12.7
5	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.59	3.95
6	Swiss National Supercomputing Centre (CSCS)	Cray	Piz Daint Cray XC30, Xeon E5 8C 2.6GHz, Aries, NVIDIA K20x	Switzer-land	115,984	6.27	2.33
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	462,462	5.17	4.51
8	Forschungszentrum Juelich (FZJ)	IBM	JuQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	458,752	5.01	2.30
24	Lawrence Berkeley National Laboratory	Cray	Edison Cray XC30, Intel Xeon E5-2695v2, 2.4GHz	USA	133,824	1.65	
44	Lawrence Berkeley National Laboratory	Cray	Hopper Cray XE6, Opteron 12C 2.1 GHZ, Gemini	USA	153,408	1.05	2.90

42st List: The TOP10 in November 2013

#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	National University of Defense Technology	NUDT	Tianhe-2 NUDT TH-IVB-FEP, Xeon 12C 2.2GHz, IntelXeon Phi	China	3,120,000	33.9	17.8
2	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.6	8.21
3	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	17.2	7.89
4	RIKEN Advanced Institute for Computational Science	Fujitsu	K Computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	795,024	10.5	12.7
5	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.59	3.95
6	Swiss National Supercomputing Centre (CSCS)	Cray	Piz Daint Cray XC30, Xeon E5 8C 2.6GHz, Aries, NVIDIA K20x	Switzer-land	115,984	6.27	2.33
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	462,462	5.17	4.51
8	Forschungszentrum Juelich (FZJ)	IBM	JuQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	458,752	5.01	2.30
9	Lawrence Livermore National Laboratory	IBM	Vulcan BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	393,216	4.29	1.97
10	Leibniz Rechenzentrum	IBM	SuperMUC iDataPlex DX360M4, Xeon E5 8C 2.7GHz, Infiniband FDR	Germany	147,456	2.90	3.52

42st List: The TOP9 + the one you'll use

#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	National University of Defense Technology	NUDT	Tianhe-2 NUDT TH-IVB-FEP, Xeon 12C 2.2GHz, IntelXeon Phi	China	3,120,000	33.9	17.8
2	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.6	8.21
3	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	17.2	7.89
4	RIKEN Advanced Institute for Computational Science	Fujitsu	K Computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	795,024	10.5	12.7
5	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.59	3.95
6	Swiss National Supercomputing Centre (CSCS)	Cray	Piz Daint Cray XC30, Xeon E5 8C 2.6GHz, Aries, NVIDIA K20x	Switzer-land	115,984	6.27	2.33
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	462,462	5.17	4.51
8	Forschungszentrum Juelich (FZJ)	IBM	JuQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	458,752	5.01	2.30
9	Lawrence Livermore National Laboratory	IBM	Vulcan BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	393,216	4.29	1.97
28	Lawrence Berkeley National Laboratory	Cray	Hopper Cray XE6, Opteron 12C 2.1 GHz, Gemini	USA	153,408	1.05	2.91

The TOP10 in November 2012

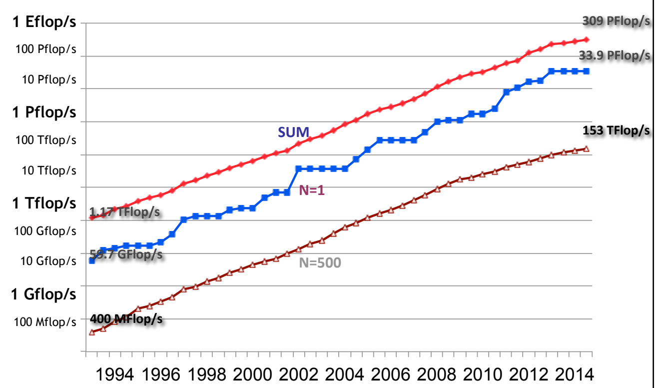
#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.59	8.21
2	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	16.32	7.89
3	RIKEN Advanced Institute for Computational Science	Fujitsu	K computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	705,024	10.51	12.66
4	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.16	3.95
5	Forschungszentrum Juelich (FZJ)	IBM	JUQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	393,216	4.14	1.97
6	Leibniz Rechenzentrum	IBM	SuperMUC iDataPlex DX360M4, Xeon E5 8C 2.7GHz, Infiniband FDR	Germany	147,456	2.90	3.42
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	204,900	2.66	
8	National SuperComputer Center in Tianjin	NUDT	Tianhe-1A NUDT TH MPP, Xeon 6C, NVidia, FT-1000 8C	China	186,368	2.57	4.04
9	CINECA	IBM	Fermi BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Italy	163,840	1.73	.82
10	IBM	IBM	DARPA Trial Subset Power 775, Power7 8C 3.84GHz, Custom	USA	63,360	1.52	3.58

01/20/2015 CS267 - Lecture 1

The TOP10 in November 2012, plus one

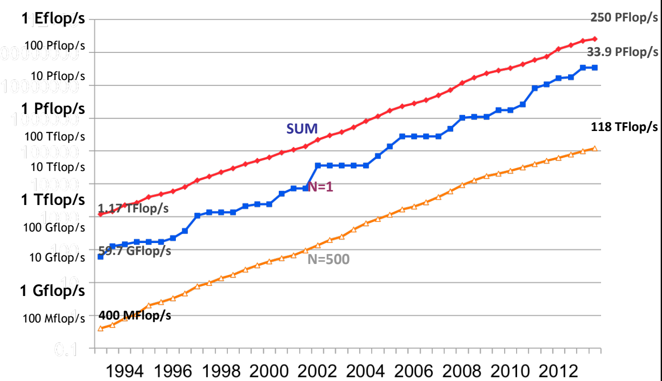
#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.59	8.21
2	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	16.32	7.89
3	RIKEN Advanced Institute for Computational Science	Fujitsu	K computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	705,024	10.51	12.66
4	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.16	3.95
5	Forschungszentrum Juelich (FZJ)	IBM	JUQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	393,216	4.14	1.97
6	Leibniz Rechenzentrum	IBM	SuperMUC iDataPlex DX360M4, Xeon E5 8C 2.7GHz, Infiniband FDR	Germany	147,456	2.90	3.42
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	204,900	2.66	
8	National SuperComputer Center in Tianjin	NUDT	Tianhe-1A NUDT TH MPP, Xeon 6C, NVidia, FT-1000 8C	China	186,368	2.57	4.04
9	CINECA	IBM	Fermi BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Italy	163,840	1.73	.82
10	IBM	IBM	DARPA Trial Subset Power 775, Power7 8C 3.84GHz, Custom	USA	63,360	1.52	3.58
19	Lawrence Berkeley National Laboratory	Cray	Hopper Cray XE6, 6C 2.1 GHz	USA	153,408	1.054	2.91

Performance Development (Nov 2014)



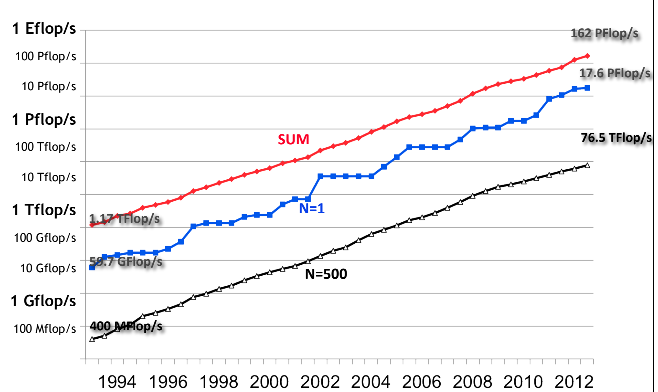
01/20/2015 CS267 - Lecture 1

Performance Development (Nov 2013)



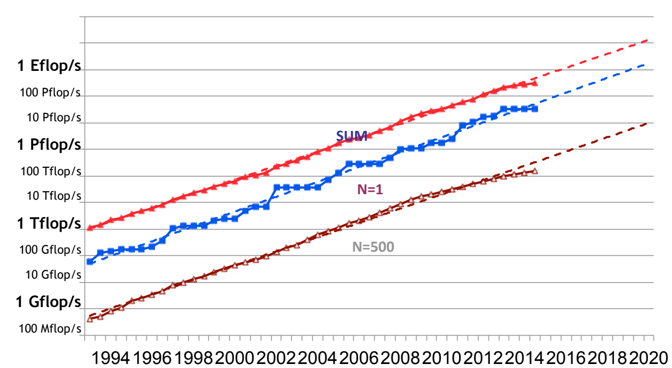
01/20/2015 CS267 - Lecture 1

Performance Development (Nov 2012)



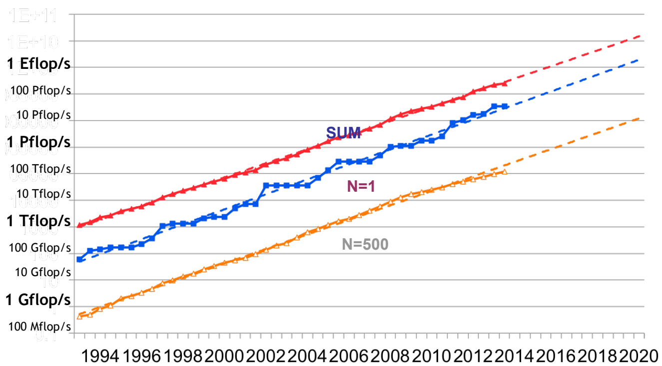
01/20/2015 CS267 - Lecture 1

Projected Performance Development (Nov 2104)

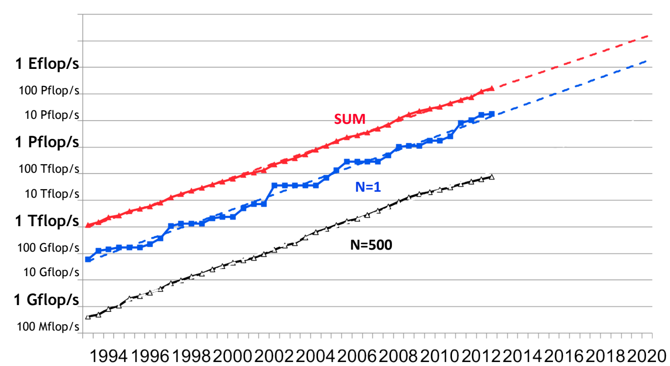


01/20/2015 CS267 - Lecture 1

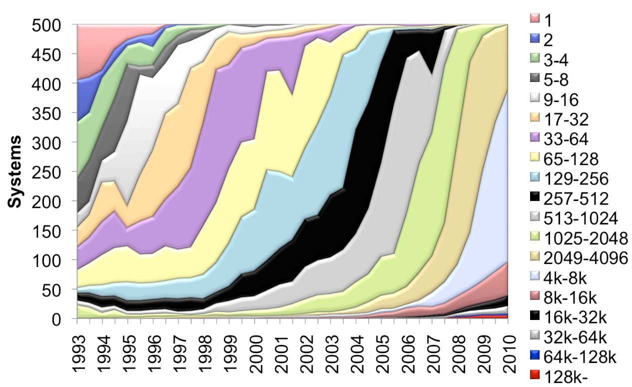
Projected Performance Development (Nov 2013)



Projected Performance Development (Nov 2012)



Core Count



Moore's Law reinterpreted

- **Number of cores per chip can double every two years**
- **Clock speed will not increase (possibly decrease)**
- **Need to deal with systems with millions of concurrent threads**
- **Need to deal with inter-chip parallelism as well as intra-chip parallelism**

Outline

- Why ~~powerful~~ all computers must be parallel processors
  - Including your laptops and handhelds
- Large CSE problems require powerful computers
  - Commercial problems too
- Why writing (fast) parallel programs is hard
  - But things are improving
- Structure of the course

Computational Science - News

"An important development in sciences is occurring at the intersection of computer science and the sciences that has the potential to have a profound impact on science. It is a leap from the application of computing … to the *integration of computer science concepts, tools, and theorems* into the very fabric of science." -*Science* 2020 Report, March 2006

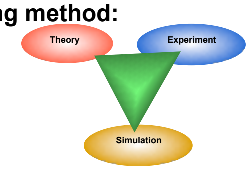
Nature, March 23, 2006

Drivers for Change

- **Continued exponential increase in computational power**
  - **Can simulate what theory and experiment can't do**
- **Continued exponential increase in experimental data**
  - **Moore's Law applies to sensors too**
  - **Need to analyze all that data**

Simulation: The Third Pillar of Science

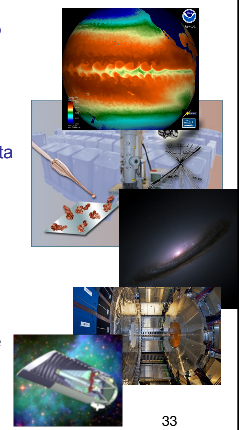
- **Traditional scientific and engineering method:**
  - **(1) Do theory or paper design**
  - **(2) Perform experiments or build system**



- **Limitations:**
  - **–Too difficult—build large wind tunnels**
  - **–Too expensive—build a throw-away passenger jet**
  - **–Too slow—wait for climate or galactic evolution**
  - **–Too dangerous—weapons, drug design, climate experimentation**
- **Computational science and engineering paradigm:**
  - **(3) Use computers to simulate and analyze the phenomenon**
    - **Based on known physical laws and efficient numerical methods**
    - **Analyze simulation results with computational tools and methods beyond what is possible manually**

Data Driven Science

- Scientific data sets are growing exponentially
  - Ability to generate data is exceeding our ability to store and analyze
  - Simulation systems and some observational devices grow in capability with Moore's Law
- Petabyte (PB) data sets will soon be common:
  - *Climate modeling:* estimates of the next IPCC data is in 10s of petabytes
  - *Genome:* JGI alone will have .5 petabyte of data this year and double each year
  - *Particle physics*: LHC is projected to produce 16 petabytes of data per year
  - *Astrophysics*: LSST and others will produce 5 petabytes/year (via 3.2 Gigapixel camera)
- Create scientific communities with "Science Gateways" to data



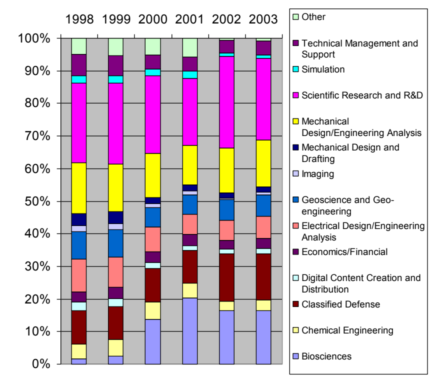
Some Particularly Challenging Computations

- **Science**
  - Global climate modeling
  - Biology: genomics; protein folding; drug design
  - Astrophysical modeling
  - Computational Chemistry
  - Computational Material Sciences and Nanosciences
- **Engineering**
  - Semiconductor design
  - Earthquake and structural modeling
  - Computation fluid dynamics (airplane design)
  - Combustion (engine design)
  - Crash simulation
- **Business**
  - Financial and economic modeling
  - Transaction processing, web services and search engines
- **Defense**
  - Nuclear weapons -- test by simulations
  - Cryptography

Economic Impact of HPC

- **Airlines:**
  - System-wide logistics optimization systems on parallel systems.
  - Savings: approx. $100 million per airline per year.
- **Automotive design:**
  - Major automotive companies use large systems (500+ CPUs) for:
    - CAD-CAM, crash testing, structural integrity and aerodynamics.
    - One company has 500+ CPU parallel system.
  - Savings: approx. $1 billion per company per year.
- **Semiconductor industry:**
  - Semiconductor firms use large systems (500+ CPUs) for
    - device electronics simulation and logic validation
  - Savings: approx. $1 billion per company per year.
- **Energy**
  - Computational modeling improved performance of current nuclear power plants, equivalent to building two new power plants.

$5B World Market in Technical Computing in 2004



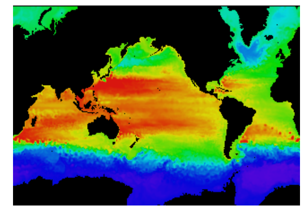
Source: IDC 2004, from NRC Future of Supercomputing Report

What Supercomputers Do – Two Examples

- **Climate modeling**
  - **simulation replacing experiment that is too slow**
- **Cosmic microwave background radition**
  - **analyzing massive amounts of data with new tools**

Global Climate Modeling Problem

- Problem is to compute:
  f(latitude, longitude, elevation, time) → "weather" =
  (temperature, pressure, humidity, wind velocity)
- Approach:
  - *Discretize* the domain, e.g., a measurement point every 10 km
  - Devise an algorithm to predict weather at time t+δt given t
- Uses:
  - Predict major events, e.g., El Nino
  - Use in setting air emissions standards
  - Evaluate global warming scenarios



Source: http://www.epm.ornl.gov/chammp/chammp.html

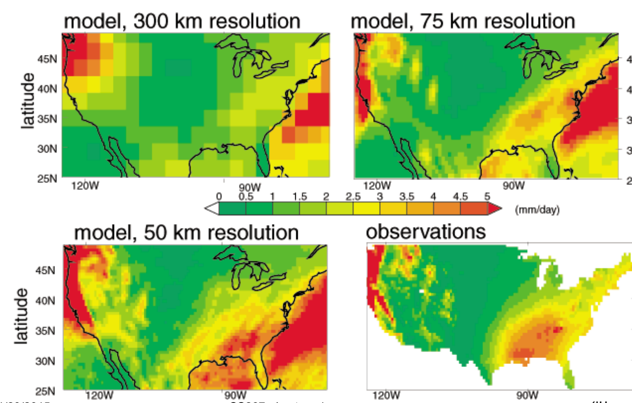
Global Climate Modeling Computation

- One piece is modeling the fluid flow in the atmosphere
  - Solve Navier-Stokes equations
    - Roughly 100 Flops per grid point with 1 minute timestep
- Computational requirements:
  - To match real-time, need 5 x 10^{11} flops in 60 seconds = 8 Gflop/s
  - Weather prediction (7 days in 24 hours) → 56 Gflop/s
  - Climate prediction (50 years in 30 days) → 4.8 Tflop/s
  - To use in policy negotiations (50 years in 12 hours) → 288 Tflop/s
- To double the grid resolution, computation is 8x to 16x
- State of the art models require integration of atmosphere, clouds, ocean, sea-ice, land models, plus possibly carbon cycle, geochemistry and more
- Current models are coarser than this

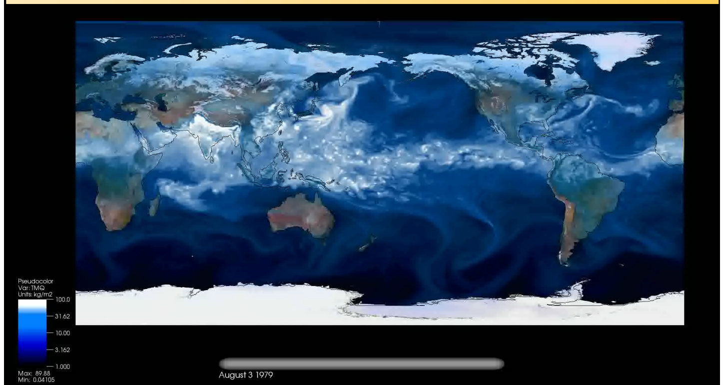
High Resolution Climate Modeling on NERSC-3 – P. Duffy, et al., LLNL

Wintertime Precipitation (millimeters/day)

As model resolution becomes finer, results converge towards observations

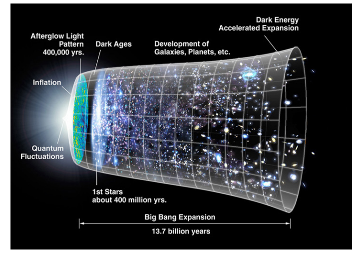


U.S.A. Hurricane



Source: Data from M.Wehner, visualization by Prabhat, LBNL

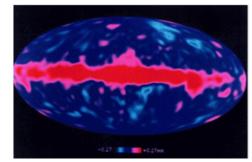
NERSC User George Smoot wins 2006 Nobel Prize in Physics



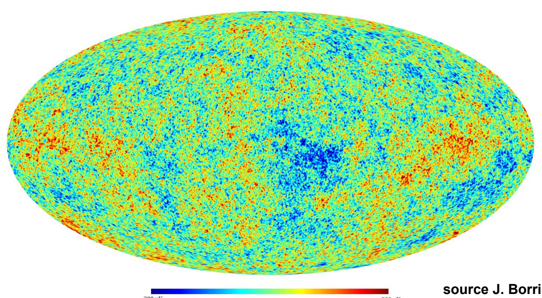
Cosmic Microwave Background Radiation (CMB): an image of the universe at 400,000 years

Smoot and Mather 1992

COBE Experiment showed anisotropy of CMB



The Current CMB Map



source J. Borrill, LBNL

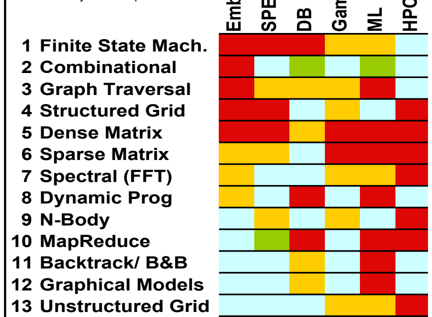
- Unique imprint of primordial physics through the tiny anisotropies in temperature and polarization.
- Extracting these μKelvin fluctuations from inherently noisy data is a serious computational challenge.

Evolution Of CMB Data Sets: Cost > O(Np^3)

Experiment	N_t	N_p	N_b	Limiting Data	Notes
COBE (1989)	$2x10^9$	$6x10^3$	$3x10^1$	Time	Satellite, Workstation
BOOMERanG (1998)	$3x10^8$	$5x10^5$	$3x10^1$	Pixel	Balloon, 1st HPC/NERSC
(4yr) WMAP (2001)	$7x10^{10}$	$4x10^7$	$1x10^3$	?	Satellite, Analysis-bound
Planck (2007)	$5x10^{11}$	$6x10^8$	$6x10^3$	Time/ Pixel	Satellite, Major HPC/DA effort
POLARBEAR (2007)	$8x10^{12}$	$6x10^6$	$1x10^3$	Time	Ground, NG-multiplexing
CMBPol (~2020)	10^{14}	10^9	10^4	Time/ Pixel	Satellite, Early planning/design
data compression →					

Which commercial applications *require* parallelism?

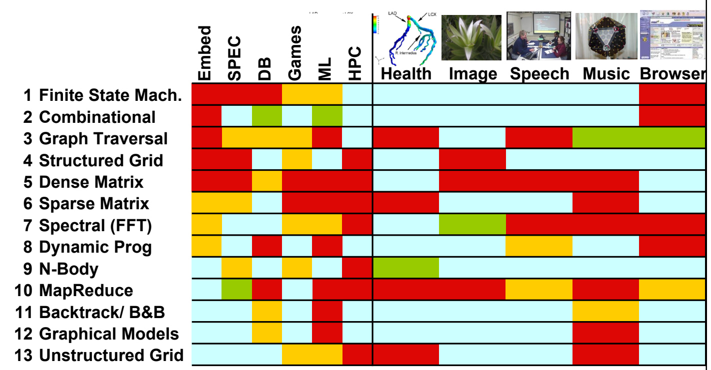
Analyzed in detail in "Berkeley View" report



Analyzed in detail in "Berkeley View" report www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html

What do commercial and CSE applications have in common?

Motif/Dwarf: Common Computational Methods
(Red Hot → Blue Cool)



Outline

- Why ~~powerful~~ all computers must be parallel processors
  - Including your laptops and handhelds
- Large CSE problems require powerful computers
  - Commercial problems too
- Why writing (fast) parallel programs is hard
  - But things are improving
- Structure of the course

Principles of Parallel Computing

- Finding enough parallelism (Amdahl's Law)
- Granularity – how big should each parallel task be
- Locality – moving data costs more than arithmetic
- Load balance – don't want 1K processors to wait for one slow one
- Coordination and synchronization – sharing data safely
- Performance modeling/debugging/tuning

➡ All of these things makes parallel programming even harder than sequential programming.

"Automatic" Parallelism in Modern Machines

- Bit level parallelism
  - within floating point operations, etc.
- Instruction level parallelism (ILP)
  - multiple instructions execute per clock cycle
- Memory system parallelism
  - overlap of memory operations with computation
- OS parallelism
  - multiple jobs run in parallel on commodity SMPs

Limits to all of these -- for very high performance, need user to identify, schedule and coordinate parallel tasks

Finding Enough Parallelism

- Suppose only part of an application seems parallel
- Amdahl's law
  - let s be the fraction of work done sequentially, so (1-s) is fraction parallelizable
  - P = number of processors

Speedup(P) = Time(1)/Time(P)

<= 1/(s + (1-s)/P)

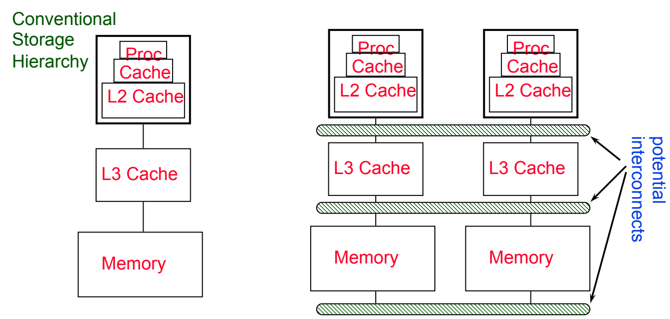
<= 1/s

- Even if the parallel part speeds up perfectly performance is limited by the sequential part
- Top500 list: currently fastest machine has P~3.1M; 2nd fastest has ~560K

Overhead of Parallelism

- Given enough parallel work, this is the biggest barrier to getting desired speedup
- Parallelism overheads include:
  - cost of starting a thread or process
  - cost of communicating shared data
  - cost of synchronizing
  - extra (redundant) computation
- Each of these can be in the range of milliseconds (=millions of flops) on some systems
- Tradeoff: Algorithm needs sufficiently large units of work to run fast in parallel (i.e. large granularity), but not so large that there is not enough parallel work

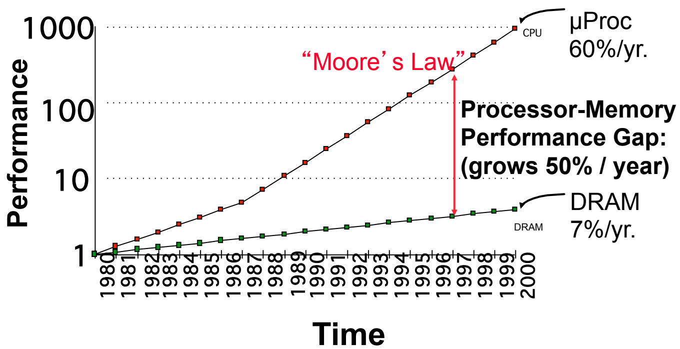
Locality and Parallelism



- Large memories are slow, fast memories are small
- Storage hierarchies are large and fast on average
- Parallel processors, collectively, have large, fast cache
  - the slow accesses to "remote" data we call "communication"
- Algorithm should do most work on local data

Processor-DRAM Gap (latency)

Goal: find algorithms that minimize communication, not necessarily arithmetic



Load Imbalance

- Load imbalance is the time that some processors in the system are idle due to
  - insufficient parallelism (during that phase)
  - unequal size tasks
- Examples of the latter
  - adapting to "interesting parts of a domain"
  - tree-structured computations
  - fundamentally unstructured problems
- Algorithm needs to balance load
  - Sometimes can determine work load, divide up evenly, before starting
    - "Static Load Balancing"
  - Sometimes work load changes dynamically, need to rebalance dynamically
    - "Dynamic Load Balancing," eg work-stealing

Parallel Software Eventually – ParLab view

- 2 types of programmers ➔ 2 layers of software
- **Efficiency Layer** (10% of programmers)
  - Expert programmers build Libraries implementing kernels, "Frameworks", OS, ….
  - Highest fraction of peak performance possible
- **Productivity Layer** (90% of programmers)
  - Domain experts / Non-expert programmers productively build parallel applications by composing frameworks & libraries
  - Hide as many details of machine, parallelism as possible
  - Willing to sacrifice some performance for productive programming
- Expect students may want to work at either level
  - In the meantime, we all need to understand enough of the efficiency layer to use parallelism effectively

Outline

- Why ~~powerful~~ all computers must be parallel processors
  - Including your laptops and handhelds
- Large CSE problems require powerful computers
  - Commercial problems too
- Why writing (fast) parallel programs is hard
  - But things are improving
- Structure of the course

Course Mechanics

- Web page:
  - http://www.cs.berkeley.edu/~demmel/cs267_Spr15/
- Normally a mix of CS, EE, and other engineering and science students
- **Please fill out survey on web page (posted)**
- Grading:
  - Warmup assignment (homework 0 on the web)
    - **Build a web page on an interest of yours in CSE**
  - Three programming assignments in first half of semester
    - We will team up CS/nonCS students for HW1
  - Final projects
    - Could be parallelizing an application, building or evaluating a tool, etc.
    - We encourage interdisciplinary teams, since this is the way parallel scientific software is generally built
- Class computer accounts on Edison, Hopper at NERSC
  - Fill out forms next time

Instructors

- Jim Demmel, EECS & Mathematics
- GSIs:
  - Evangelos Georganas, EECS
  - Forrest Iandola, EECS
  - Penporn Koanantakool, EECS
- Contact information on web page

Students

- 118 registered or on the waitlist (100 grad, 18 undergrad)
- 78 CS or EECS students, rest from

Applied Math	Math
Business Administration	Mechanical Engineering
Civil & Environmental Engineering	Neuroscience
Industrial Engineering and Operations Research	Nuclear Engineering
Information Management and Systems	Physics
	Psychology
	Statistics

Remote instruction – preparing an experiment

- Lectures will be webcast, archived, as in past semesters
  - See class webpage for details
- XSEDE is nationwide project supporting users of NSF supercomputer facilities
  - XSEDE offering CS267 to students nationwide, starting 2013
  - Based on Videos from Spring 2012 offering
  - Free accounts on NSF supercomputer
  - This year: local instructors at 10+ universities to give real grades
  - Challenges to "scaling up" education
    - Q&A – piazza for CS267, moodle for XSEDE
    - Autograding
      - For correctness – run test cases (not as easy as it sounds)
      - For performance – timing on suitable platform
  - Ditto for Kurt Keutzer's CS194 class

Rough List of Topics

- Basics of computer architecture, memory hierarchies, performance
- Parallel Programming Models and Machines
  - Shared Memory and Multithreading
  - Distributed Memory and Message Passing
  - Data parallelism, GPUs
  - Cloud computing
- Parallel languages and libraries
  - Shared memory threads and OpenMP
  - MPI
  - Other Languages , frameworks (UPC, CUDA, PETSC, "Pattern Language", …)
- "Seven Dwarfs" of Scientific Computing
  - Dense & Sparse Linear Algebra
  - Structured and Unstructured Grids
  - Spectral methods (FFTs) and Particle Methods
- 6 additional motifs
  - Graph algorithms, Graphical models, Dynamic Programming, Branch & Bound, FSM, Logic
- General techniques
  - Autotuning, Load balancing, performance tools
- Applications: climate modeling, materials science, astrophysics … (guest lecturers)

Reading Materials

- Pointers on class web page
- Must read:
  - "The Landscape of Parallel Processing Research: The View from Berkeley"
    - http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf
- Some on-line texts:
  - Demmel's notes from CS267 Spring 1999, which are similar to 2000 and 2001. However, they contain links to html notes from 1996.
    - http://www.cs.berkeley.edu/~demmel/cs267_Spr99/
  - Ian Foster's book, "Designing and Building Parallel Programming".
    - http://www-unix.mcs.anl.gov/dbpp/
- Potentially useful texts:
  - "Sourcebook for Parallel Computing", by Dongarra, Foster, Fox, ..
    - A general overview of parallel computing methods
  - "Performance Optimization of Numerically Intensive Codes" by Stefan Goedecker and Adolfy Hoisie
    - This is a practical guide to optimization, mostly for those of you who have never done any optimization

Reading Materials (cont.)

- Recent books with papers about the current state of the art
  - David Bader (ed.), "Petascale Computing, Algorithms and Applications", Chapman & Hall/CRC, 2007
  - Michael Heroux, Padma Ragahvan, Horst Simon (ed.),"Parallel Processing for Scientific Computing", SIAM, 2006.
  - M. Sottile, T. Mattson, C. Rasmussen, Introduction to Concurrency in Programming Languages, Chapman & Hall/CRC, 2009.
- More pointers on the web page

Students

- ~~73~~ 75 registered or on the waitlist (61 grad, 12 undergrad)
- 28 CS or EECS grad students, rest from

Applied Math
Applied Science & Technology
Astrophysics
Bioengineering
Business Administration
Chemical Engineering
Chemistry
Civil & Environmental Engineering
Geography
Geography
Industrial Engineering and Operations Research
Information Management and Systems
Math
Mechanical Engineering
Music
Nuclear Engineering
Physics
Statistics

- 8 CS or EECS undergrads, 4 double

What you should get out of the course

In depth understanding of:

- When is parallel computing useful?
- Understanding of parallel computing hardware options
- Overview of programming models (software) and tools, and experience using some of them
- Some important parallel applications and the algorithms
- Performance analysis and tuning
- Exposure to various open research questions

Extra slides