

---

# **CS267/E233**

# **Applications of Parallel Computers**

## **Lecture 1: Introduction**

Kathy Yelick

yelick@cs.berkeley.edu

<http://inst.eecs.berkeley.edu/~cs267>

# Outline

---

- Introduction
- Large important problems require powerful computers
- Why powerful computers must be parallel processors
- Principles of parallel computing performance
- Structure of the course

---

# Why we need powerful computers

# Simulation: The Third Pillar of Science

---

- Traditional scientific and engineering paradigm:
  - 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 paradigm:
  - 3) Use high performance computer systems to **simulate** the phenomenon
    - Base on known physical laws and efficient numerical methods.

# Some Particularly Challenging Computations

- **Science**

- Global climate modeling
- Astrophysical modeling
- Biology: genomics; protein folding; drug design
- Computational Chemistry
- Computational Material Sciences and Nanosciences

- **Engineering**

- Crash simulation
- Semiconductor design
- Earthquake and structural modeling
- Computation fluid dynamics (airplane design)
- Combustion (engine design)

- **Business**

- Financial and economic modeling
- Transaction processing, web services and search engines

- **Defense**

- Nuclear weapons -- test by simulations
- Cryptography

# Units of Measure in HPC

---

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

Mega	Mflop/s = $10^6$ flop/sec	Mbyte = $10^6$ byte (also $2^{20} = 1048576$ )
Giga	Gflop/s = $10^9$ flop/sec	Gbyte = $10^9$ byte (also $2^{30} = 1073741824$ )
Tera	Tflop/s = $10^{12}$ flop/sec	Tbyte = $10^{12}$ byte (also $2^{40} = 10995211627776$ )
Peta	Pflop/s = $10^{15}$ flop/sec	Pbyte = $10^{15}$ byte (also $2^{50} = 1125899906842624$ )
Exa	Eflop/s = $10^{18}$ flop/sec	Ebyte = $10^{18}$ byte

# Economic Impact of HPC

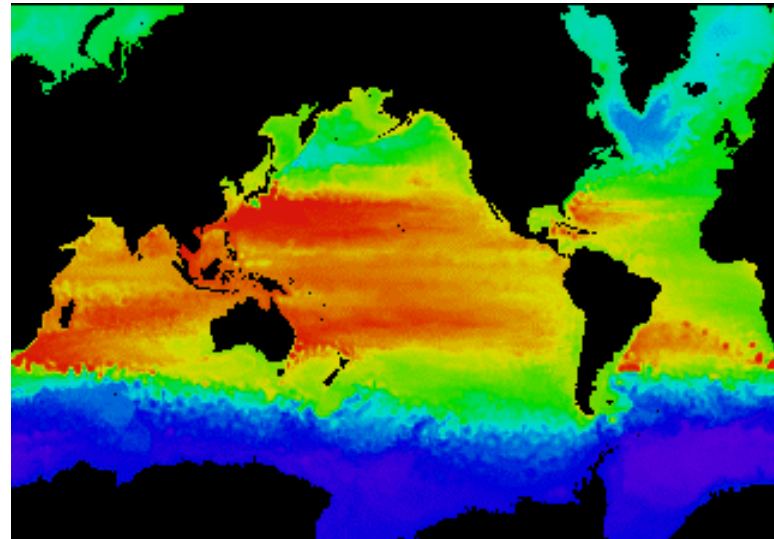
---

- **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.
- **Securities industry:**
  - Savings: approx. \$15 billion per year for U.S. home mortgages.

# Global Climate Modeling Problem

---

- Problem is to compute:  
     $f(\text{latitude, longitude, elevation, time}) \rightarrow$   
    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  $t+1$  given  $t$
- Uses:
  - Predict major events, e.g., El Nino
  - Use in setting air emissions standards



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

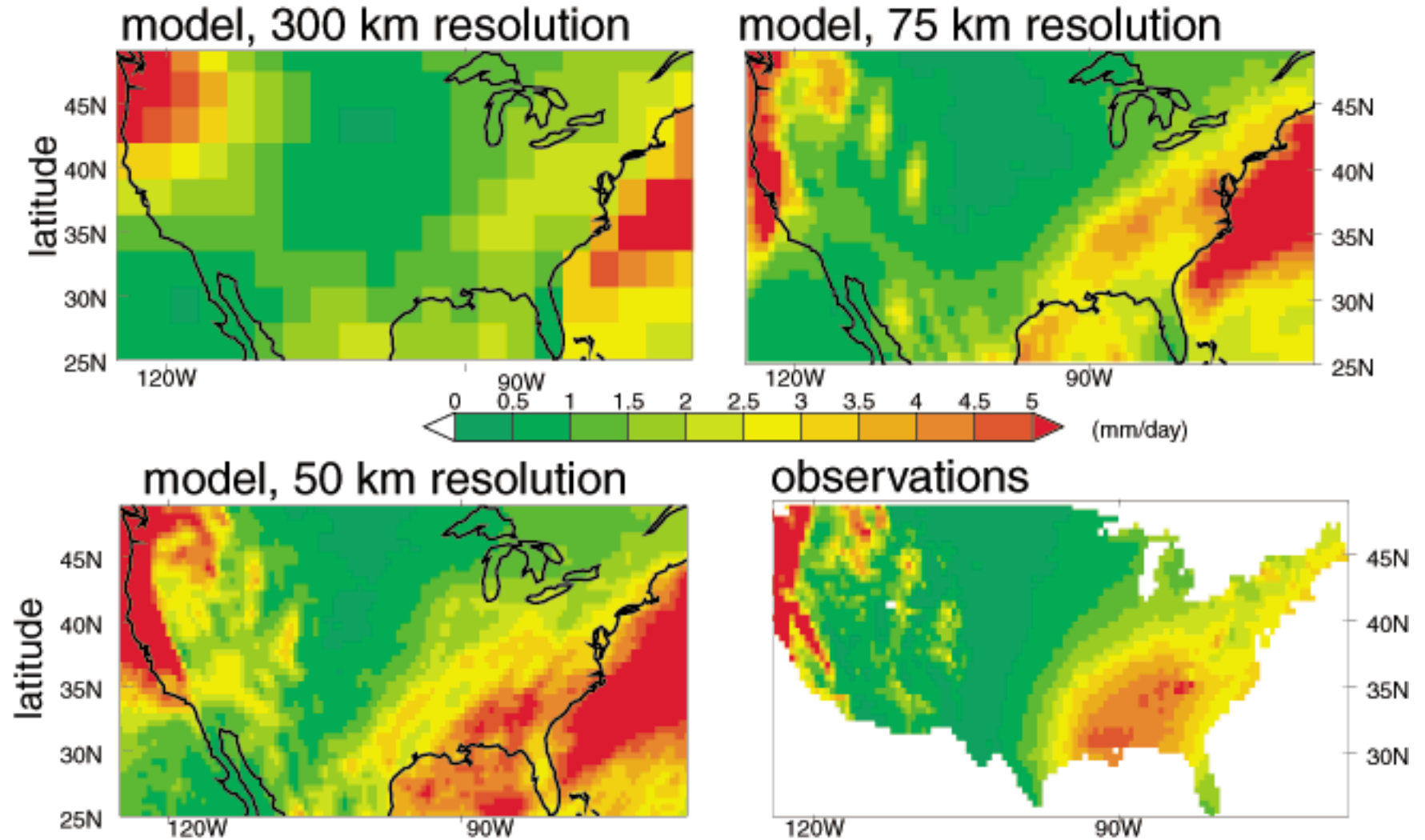
# Global Climate Modeling Computation

- One piece is modeling the fluid flow in the atmosphere
  - Solve Navier-Stokes problem
    - Roughly 100 Flops per grid point with 1 minute timestep
- Computational requirements:
  - To match real-time, need  $5 \times 10^{11}$  flops in 60 seconds = 8 Gflop/s
  - Weather prediction (7 days in 24 hours)  $\rightarrow$  56 Gflop/s
  - Climate prediction (50 years in 30 days)  $\rightarrow$  4.8 Tflop/s
  - To use in policy negotiations (50 years in 12 hours)  $\rightarrow$  288 Tflop/s
- To double the grid resolution, computation is at least 8x
- State of the art models require integration of atmosphere, 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**

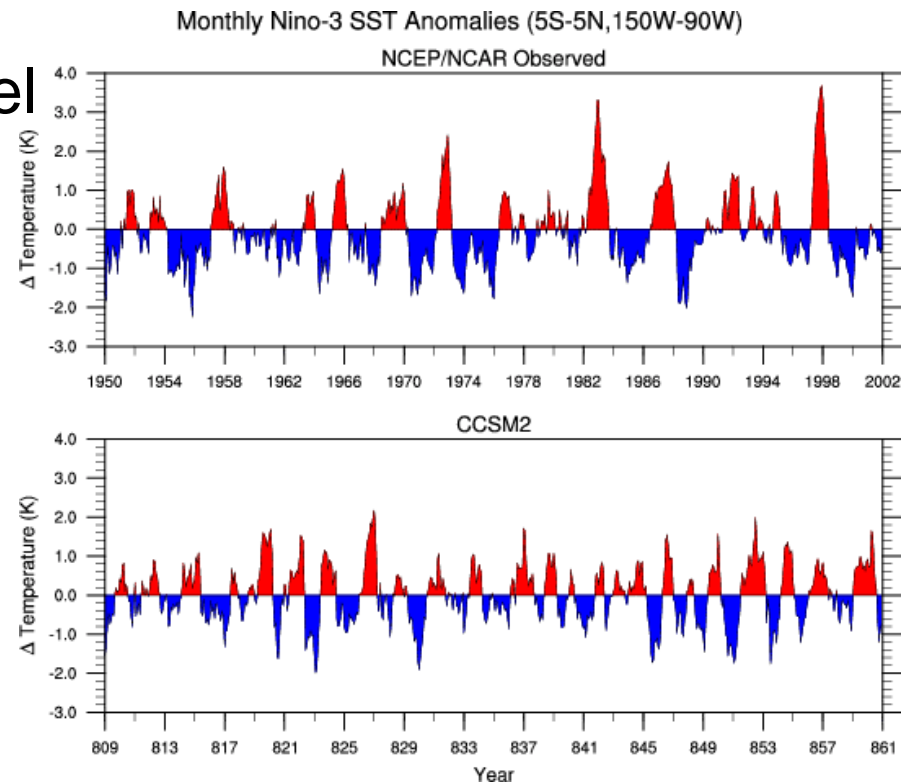
## Wintertime Precipitation

As model resolution becomes finer, results converge towards observations



# A 1000 Year Climate Simulation

- Demonstration of the Community Climate Model (CCSM2)
- A 1000-year simulation shows long-term, stable representation of the earth's climate.
- 760,000 processor hours used
- Temperature change shown
- *Warren Washington and Jerry Meehl, National Center for Atmospheric Research; Bert Semtner, Naval Postgraduate School; John Weatherly, U.S. Army Cold Regions Research and Engineering Lab Laboratory et al.*
- <http://www.neresc.gov/aboutneresc/pubs/bigsplash.pdf>



# Climate Modeling on the Earth Simulator System

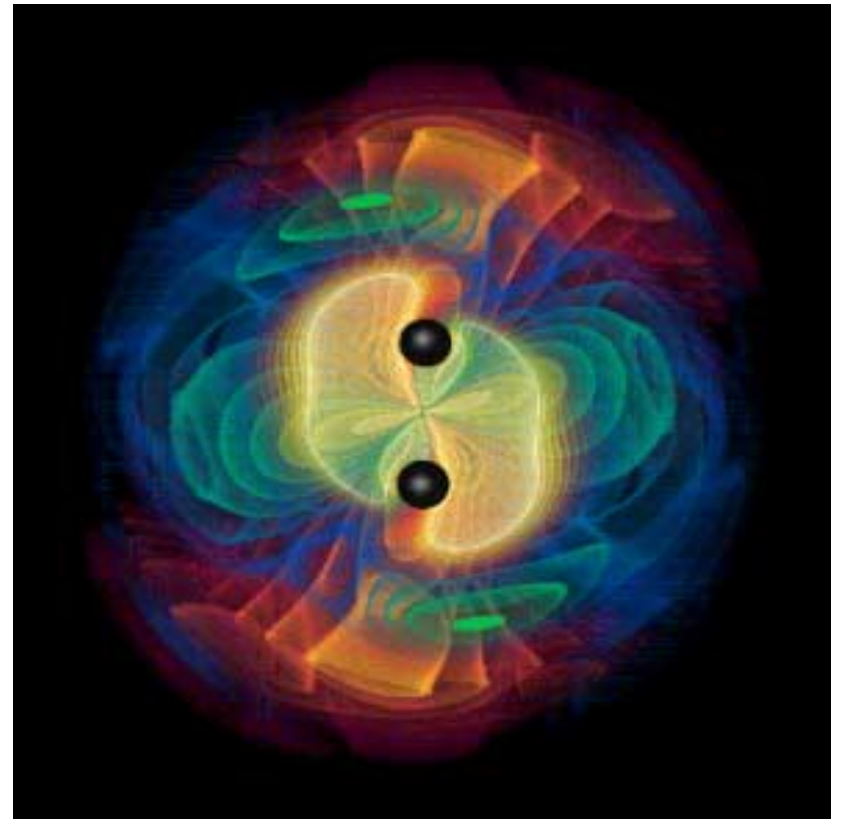
---

- Development of ES started in 1997 in order to make a comprehensive understanding of global environmental changes such as global warming.
- Its construction was completed at the end of February, 2002 and the practical operation started from March 1, 2002
- 35.86Tflops (87.5% of the peak performance) is achieved in the Linpack benchmark.
- 26.58Tflops was obtained by a global atmospheric circulation code.

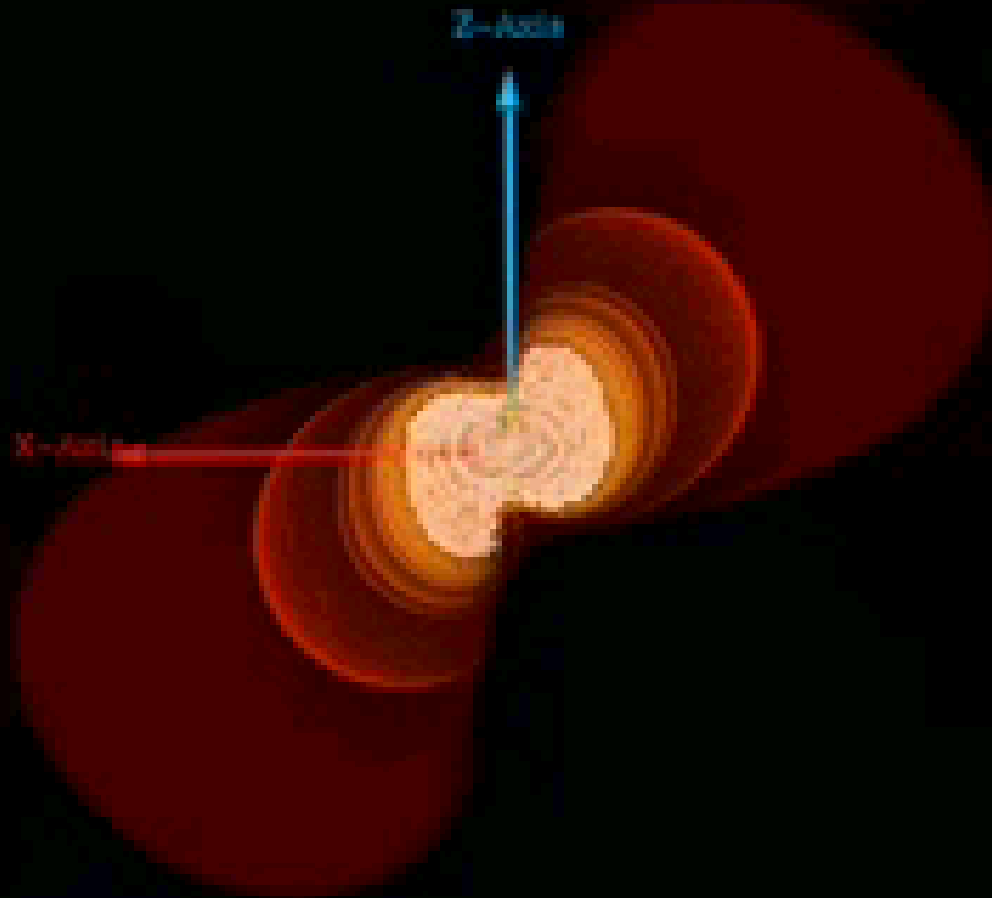
# Astrophysics: Binary Black Hole Dynamics

---

- Massive supernova cores collapse to black holes.
- At black hole center spacetime breaks down.
- Critical test of theories of gravity – General Relativity to Quantum Gravity.
- Indirect observation – most galaxies have a black hole at their center.
- Gravity waves show black hole directly including detailed parameters.
- Binary black holes most powerful sources of gravity waves.
- Simulation extraordinarily complex – evolution disrupts the spacetime !



$T = 0$



# Heart Simulation

---

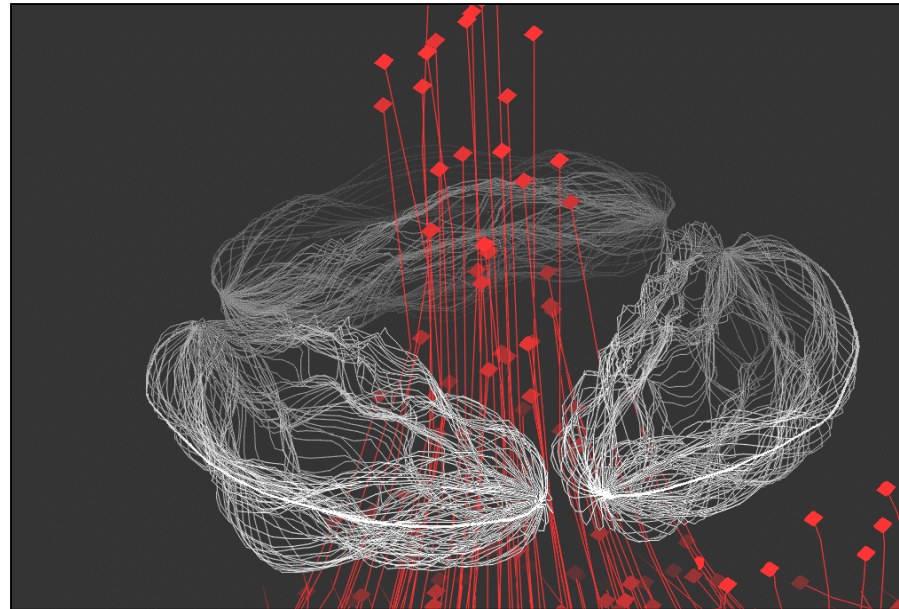
- Problem is to compute blood flow in the heart
- Approach:
  - Modeled as an elastic structure in an incompressible fluid.
    - The “immersed boundary method” due to Peskin and McQueen.
    - 20 years of development in model
    - Many applications other than the heart: blood clotting, inner ear, paper making, embryo growth, and others
  - Use a regularly spaced mesh (set of points) for evaluating the fluid
- Uses
  - Current model can be used to design artificial heart valves
  - Can help in understand effects of disease (leaky valves)
  - Related projects look at the behavior of the heart during a heart attack
  - Ultimately: real-time clinical work

# Heart Simulation Calculation

---

The involves solving Navier-Stokes equations

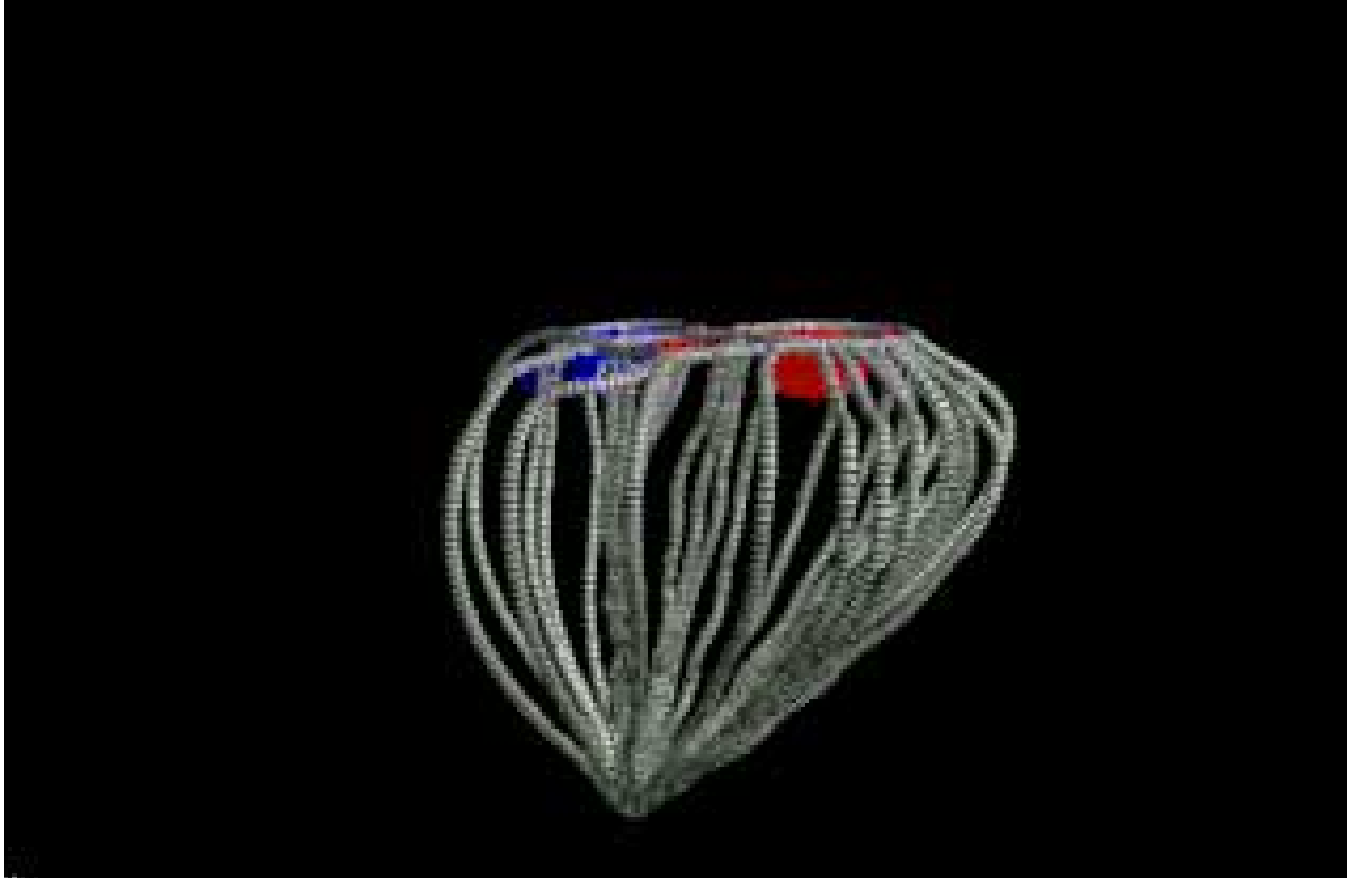
- $64^3$  was possible on Cray YMP, but  $128^3$  required for accurate model (would have taken 3 years).
  - Done on a Cray C90 -- 100x faster and 100x more memory
  - Until recently, limited to vector machines
- 
- Needs more features:
    - Electrical model of the heart, and details of muscles, E.g.,
      - Chris Johnson
      - Andrew McCulloch
    - Lungs, circulatory systems



# Heart Simulation

---

Animation of lower portion of the heart

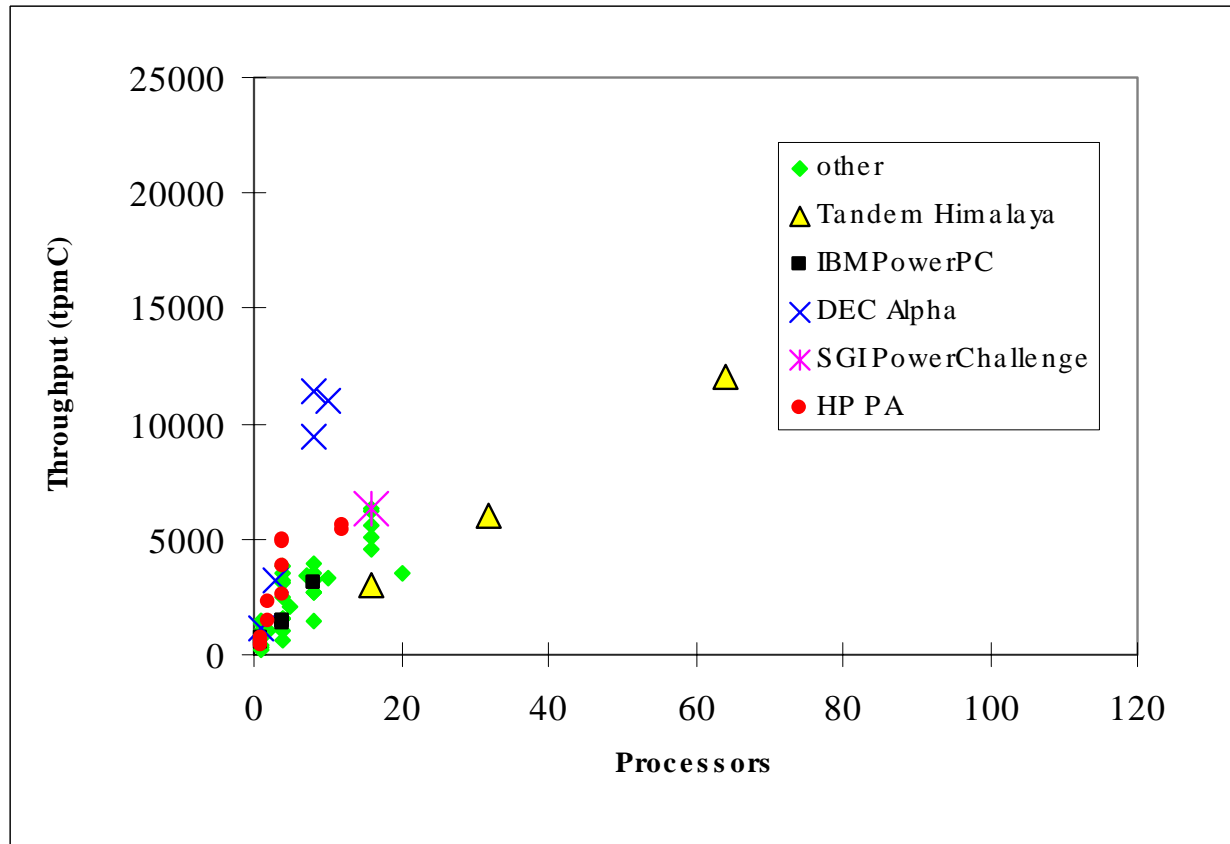


# Parallel Computing in Data Analysis

- Finding information amidst large quantities of data
- General themes of sifting through large, unstructured data sets:
  - Has there been an outbreak of some medical condition in a community?
  - Which doctors are most likely involved in fraudulent charging to medicare?
  - When should white socks go on sale?
  - What advertisements should be sent to you?
- Data collected and stored at enormous speeds (Gbyte/hour)
  - remote sensor on a satellite
  - telescope scanning the skies
  - microarrays generating gene expression data
  - scientific simulations generating terabytes of data

# Transaction Processing

(mar. 15, 1996)



- Parallelism is natural in relational operators: select, join, etc.
- Many difficult issues: data partitioning, locking, threading.

---

# Why powerful computers are parallel

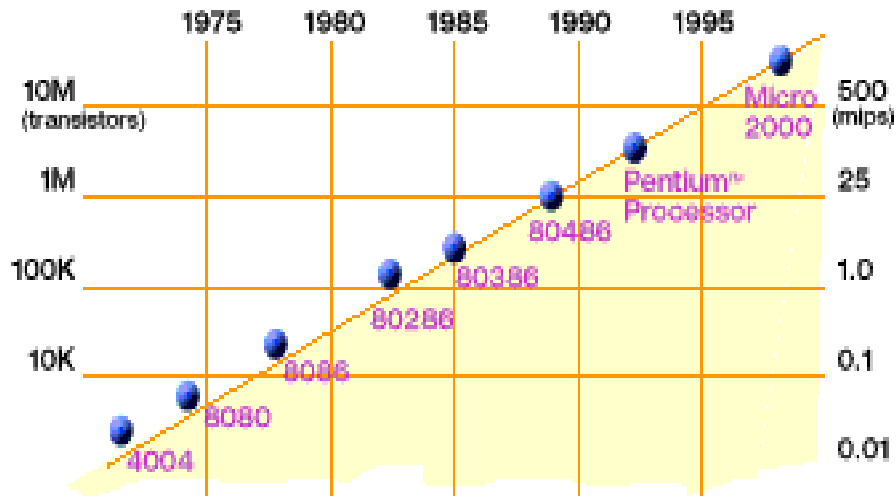
# Tunnel Vision by Experts

---

- “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.

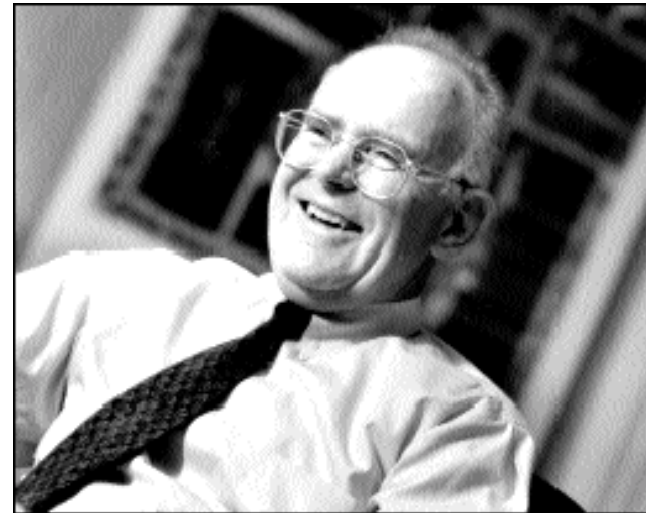
Slide source: Warfield et al.

# 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.

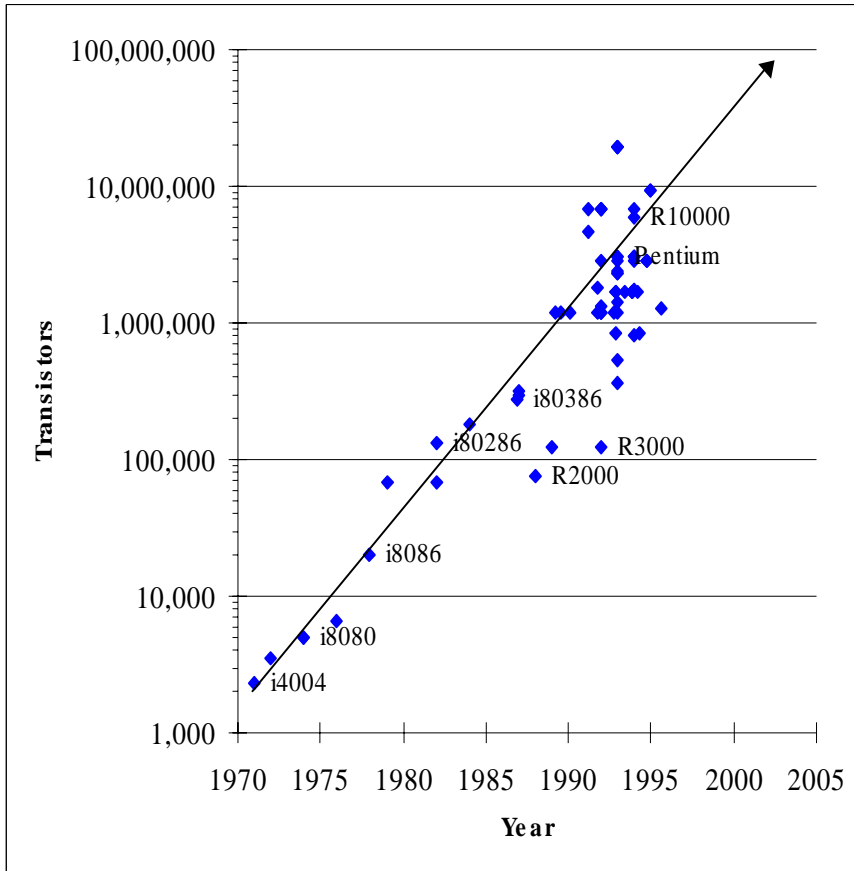
Slide source: Jack Dongarra

# Impact of Device Shrinkage

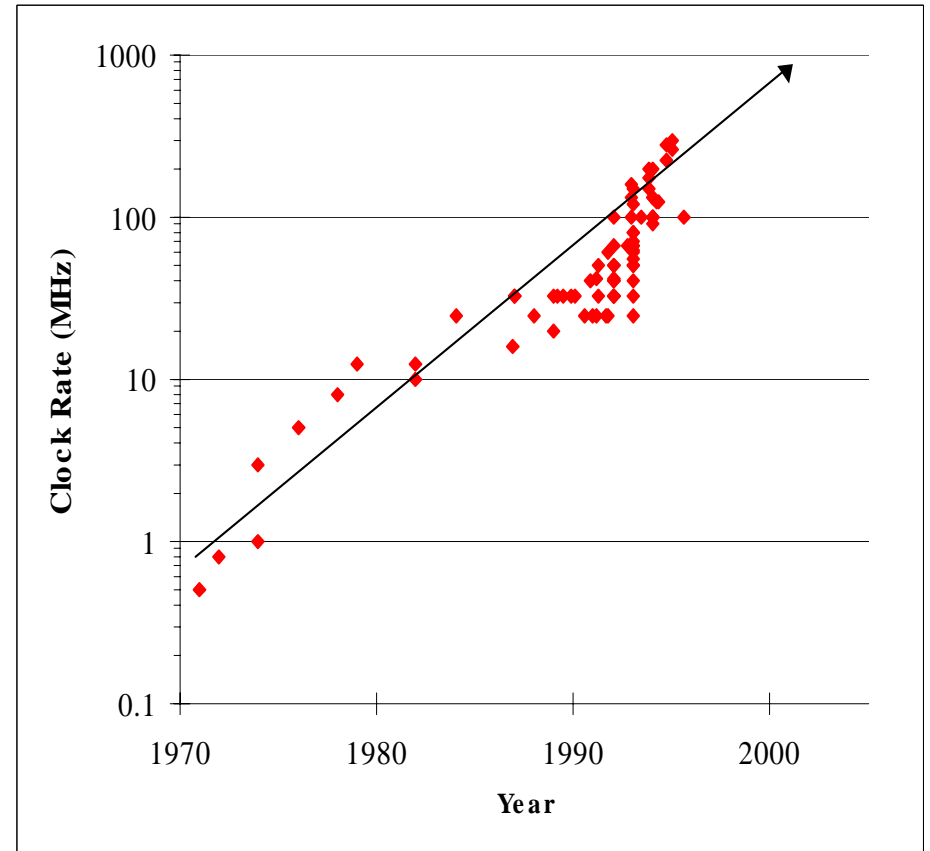
- What happens when the feature size (transistor size) shrinks by a factor of  $x$  ?
- Clock rate goes up by  $x$  because wires are shorter
  - actually less than  $x$ , because of power consumption
- Transistors per unit area goes up by  $x^2$
- Die size also tends to increase
  - typically another factor of  $\sim x$
- Raw computing power of the chip goes up by  $\sim x^4$  !
  - of which  $x^3$  is devoted either to parallelism or locality

# Microprocessor Transistors per Chip

- Growth in transistors per chip



- Increase in clock rate

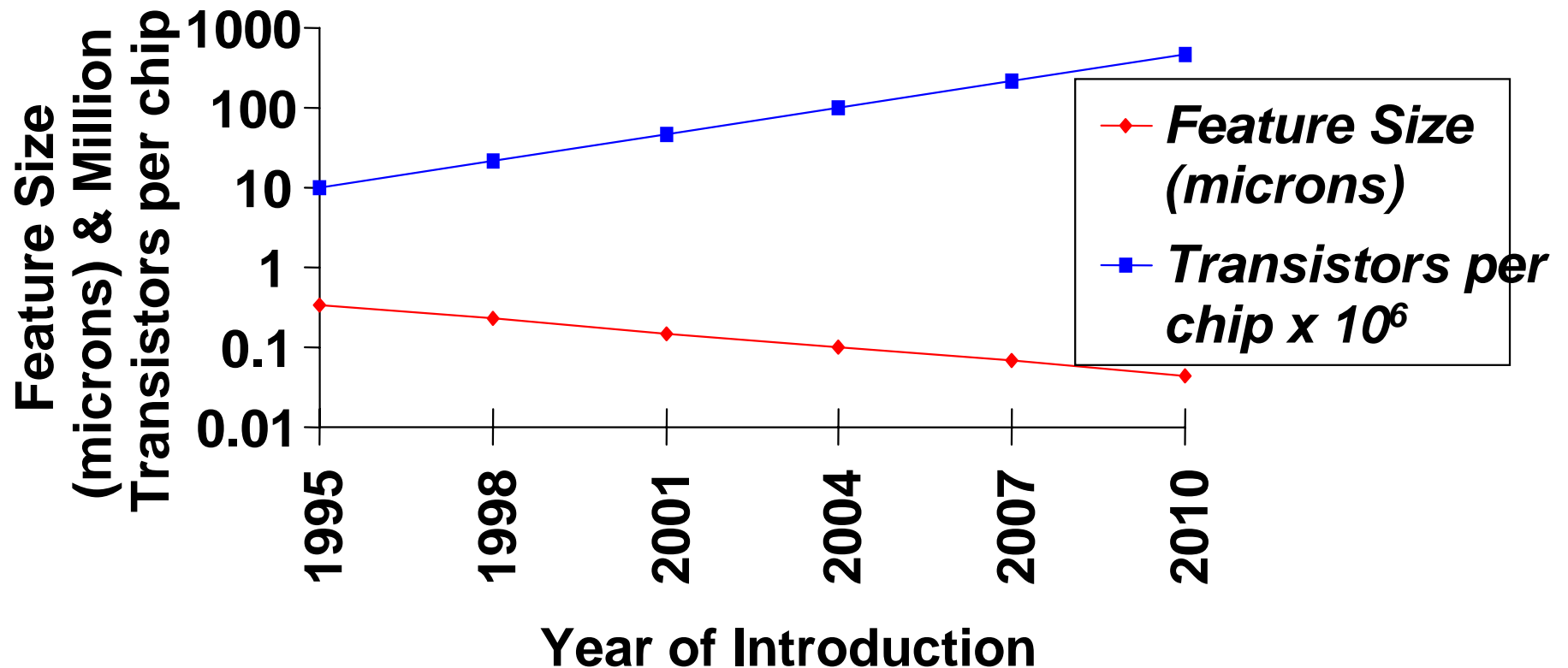


# Performance on Linpack Benchmark

System

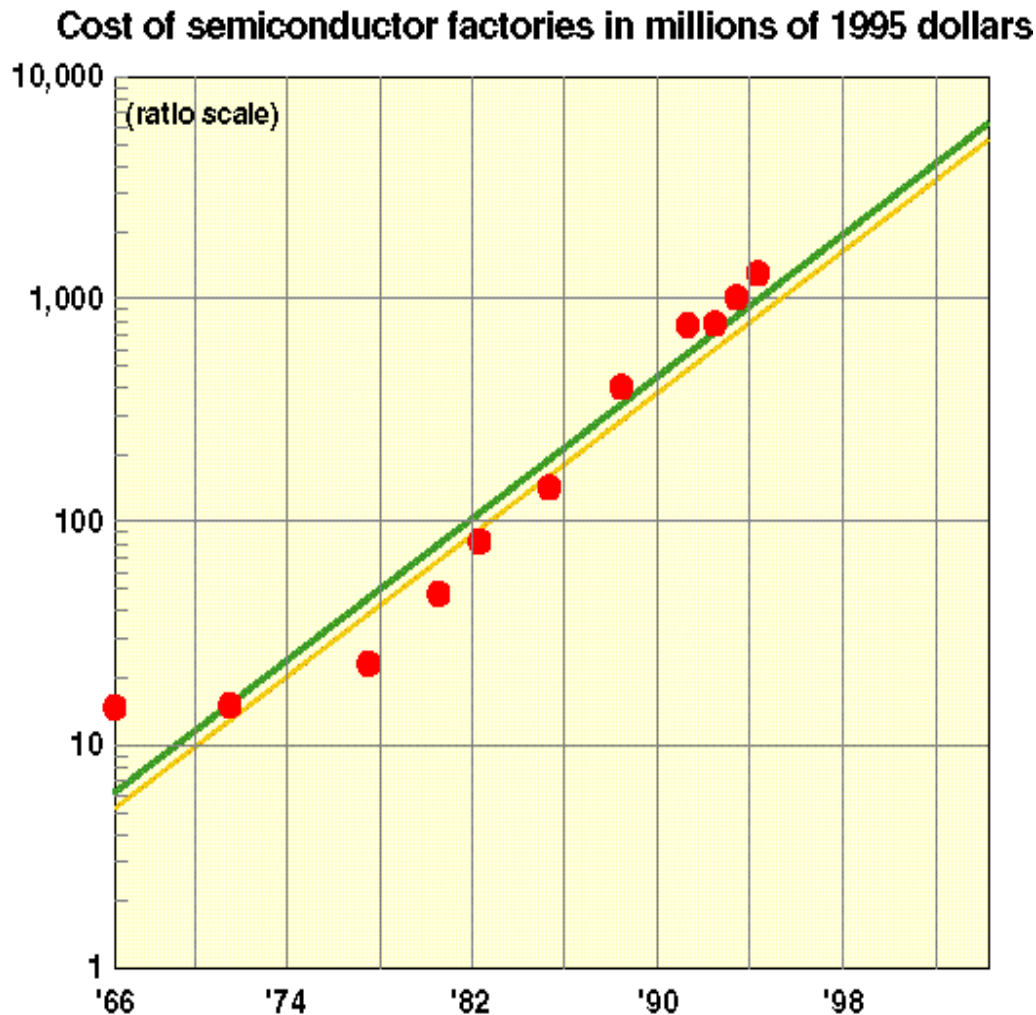
# SIA Projections for Microprocessors

*Compute power  $\sim 1/(\text{Feature Size})^3$*

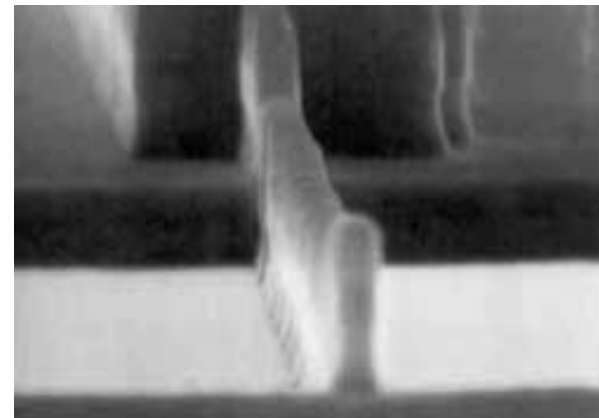


based on F.S.Preston, 1997

# But there are limiting forces: Increased cost and difficulty of manufacturing



- **Moore's 2<sup>nd</sup> law (Rock's law)**

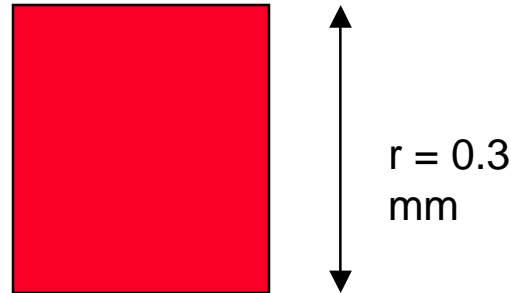


Demo of  
0.06  
micron  
CMOS

# How fast can a serial computer be?

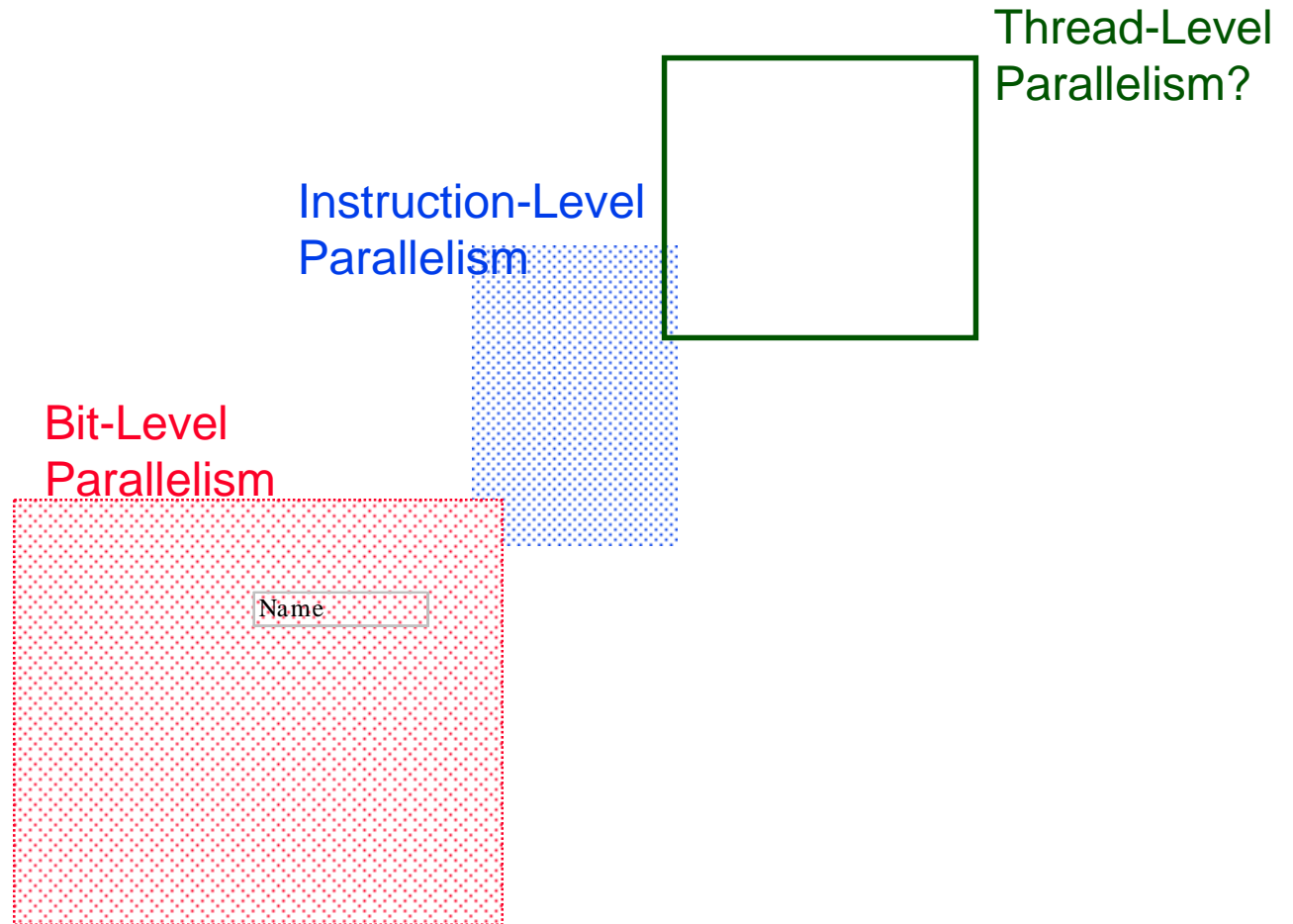
---

1 Tflop/s, 1  
Tbyte sequential  
machine



- Consider the 1 Tflop/s sequential machine:
  - Data must travel some distance,  $r$ , to get from memory to CPU.
  - To get 1 data element per cycle, this means  $10^{12}$  times per second at the speed of light,  $c = 3 \times 10^8$  m/s. Thus  $r < c/10^{12} = 0.3$  mm.
- Now put 1 Tbyte of storage in a 0.3 mm x 0.3 mm area:
  - Each word occupies about 3 square Angstroms, or the size of a small atom.

# Much of the Performance is from Parallelism



# “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.

There are limitations to all of these!

Thus to achieve high performance, the programmer needs to identify, schedule and coordinate parallel tasks and data.

---

# Measuring Performance

# Improving Real Performance

---

## Peak Performance is skyrocketing

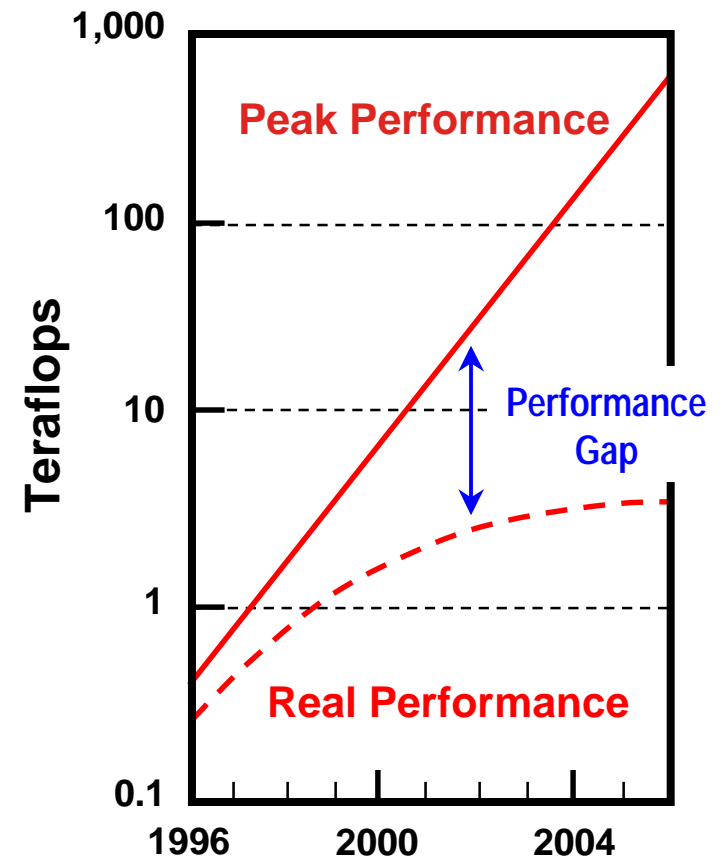
- In 1990's, peak performance increased 100x; in 2000's, it will increase 1000x

## But efficiency (the performance relative to the hardware peak) has declined

- was 40-50% on the vector supercomputers of 1990s
- now as little as 5-10% on parallel supercomputers of today

## Close the gap through ...

- Mathematical methods and algorithms that achieve high performance on a single processor and scale to thousands of processors
- More efficient programming models and tools for massively parallel supercomputers



# Performance Levels

---

- Peak advertised performance (PAP)
  - You can't possibly compute faster than this speed
- LINPACK
  - The "hello world" program for parallel computing
- Gordon Bell Prize winning applications performance
  - The right application/algorithm/platform combination plus years of work
- Average sustained applications performance
  - What one reasonable can expect for standard applications

When reporting performance results, these levels are often confused, even in reviewed publications

## Performance Levels (for example on NERSC-3)

- Peak advertised performance (PAP): 5 Tflop/s
- LINPACK (TPP): 3.05 Tflop/s
- Gordon Bell Prize winning applications performance :  
2.46 Tflop/s
  - Material Science application at SC01
- Average sustained applications performance: ~0.4 Tflop/s
  - Less than 10% peak!

---

# Course Organization

## Who is in the class?

---

- This class is listed as both a CS and Engineering class
- Normally a mix of CS, EE, and other engineering/science students
- This class seems to be about:
  - 45% CS
  - 30% EE
  - 20% Other (Civil, Mechanical, ...)
- For final projects we encourage interdisciplinary teams
  - This is the way parallel scientific software is generally built

# First Assignment

---

- See home page for details.
- Find an application of parallel computing and build a web page describing it.
  - Choose something from your research area.
  - Or from the web or elsewhere.
- Create a web page describing the application.
  - Describe the application and provide a reference (or link)
  - Describe the platform where this application was run
  - Find peak and LINPACK performance for the platform and its rank on the TOP500 list
  - Find performance of your selected application
  - What ratio of sustained to peak performance is reported?
  - Evaluate project: How did the application scale? What were the major difficulties in obtaining good performance? What tools and algorithms were used?
- Send us (Kathy and Jason) the link (we will publish a list online)
- Due next week, Wednesday (1/28)

# Schedule of Topics

---

- Introduction
- Parallel Programming Models and Machines
  - Shared Memory and Multithreading
  - Distributed Memory and Message Passing
  - Data parallelism
- Sources of Parallelism in Simulation
- Tools
  - Languages (UPC)
  - Performance Tools
  - Visualization
  - Environments
- Algorithms
  - Dense Linear Algebra
  - Partial Differential Equations (PDEs)
  - Particle methods
  - Load balancing, synchronization techniques
  - Sparse matrices
- Applications: biology, climate, combustion, astrophysics
- Project Reports

08/27/2002

CS267-Lecture 1

38

# Reading Materials

---

- 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/](http://www.cs.berkeley.edu/~demmel/cs267_Spr99/)
  - Simon's notes from Fall
    - <http://www.nersc.gov/~simon/cs267/>
  - 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

# Requirements

---

- Fill out on-line account request for Millennium machine.
  - See course web page for pointer
- Fill out request for NERSC account
  - Form available in class
- Build a web page
  - Every week or two students will report explorations, ideas, proposed work, and work to the TA via an organized webpage
  - There will be 3-4 programming assignments geared towards hands-on experience, interdisciplinary teams.
  - SIAM PP conference in late February
  - There will be a Final Project
    - Teams of 2-3, interdisciplinary is best.
    - Interesting applications or advance of systems.
    - Presentation (poster session)
    - Conference quality paper

# 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.
- Some important parallel applications and the algorithms
- Performance analysis and tuning

# Administrative Information

---

- Instructors:
  - Kathy Yelick, 777 Soda, [yelick@cs.berkeley.edu](mailto:yelick@cs.berkeley.edu)
  - TA: Jason Riedy, 515 Soda, [ejr@cs.berkeley.edu](mailto:ejr@cs.berkeley.edu)
- Accounts – **fill out forms**
  - Submit online form for Millennium account:
    - <http://www.millennium.berkeley.edu/accounts>
  - Fill out a paper form for NERSC and give to Jason:
    - <http://hpcf.nersc.gov/policy/usage.php>
- Lecture notes are based on previous semester notes:
  - Jim Demmel, David Culler, David Bailey, Bob Lucas, Kathy Yelick and Horst Simon
- Most class material and lecture notes are at:
  - <http://www.cs.berkeley.edu/~yelick/cs267>