Very Brief Review of Parallel Computing

Bin ZHOU @ NVIDIA & USTC

Jan. 2015

Acknowledgements

- Florent NOLOT, Univ-reims, france, HPC
- Introduction to Parallel Processing. Shantanu Dutt. University of Illinois at Chicago

Contents

- Why do we need parallel computing?
- How?
- Some concepts and ideas

Parallel Processing is common in real life

- Task:
- Moving
- A Pile of
- Bricks

Serial Processing Strategy

CPU way of processing?

Parallel Processing Strategy

Moore’s Law

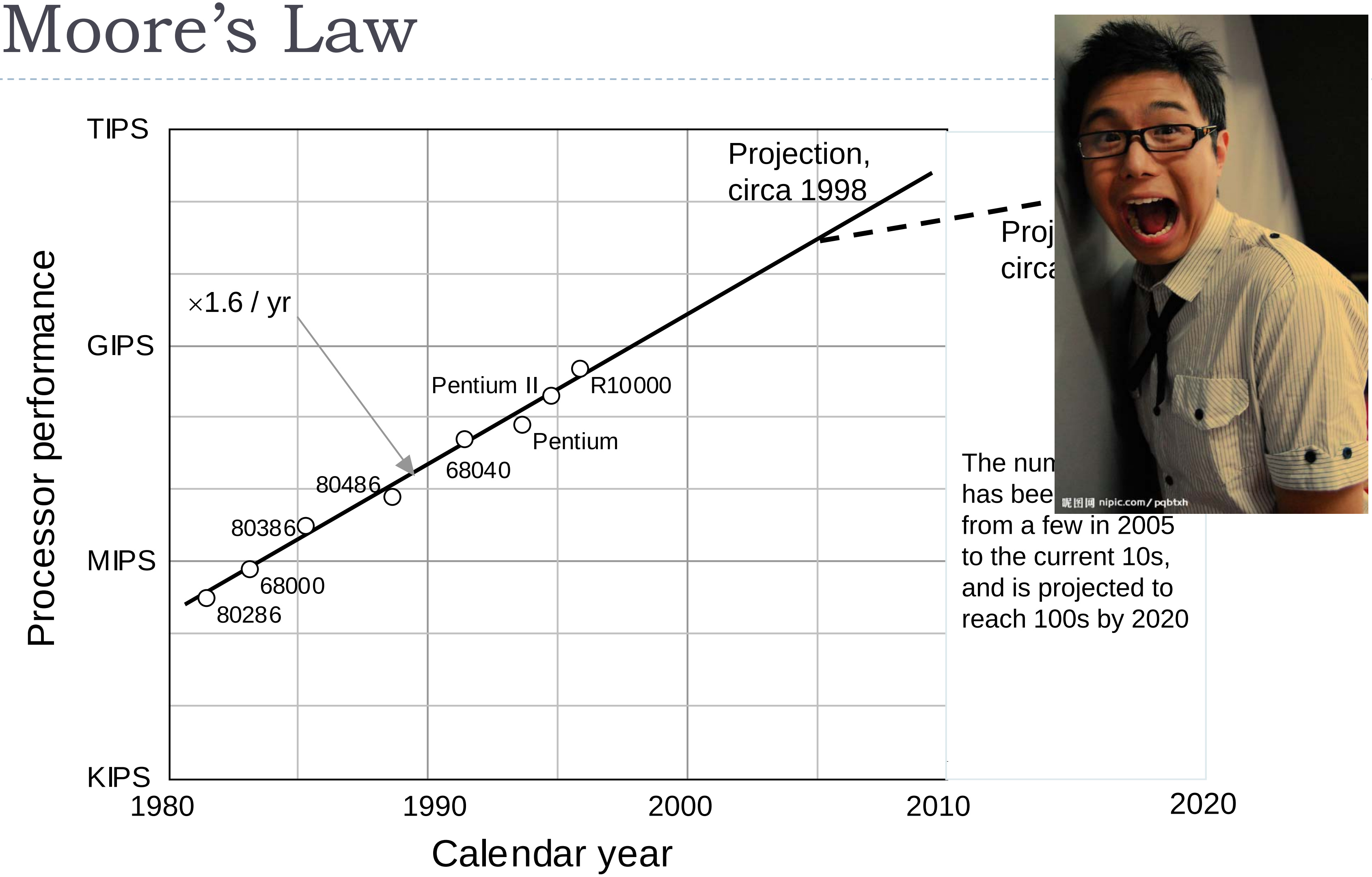
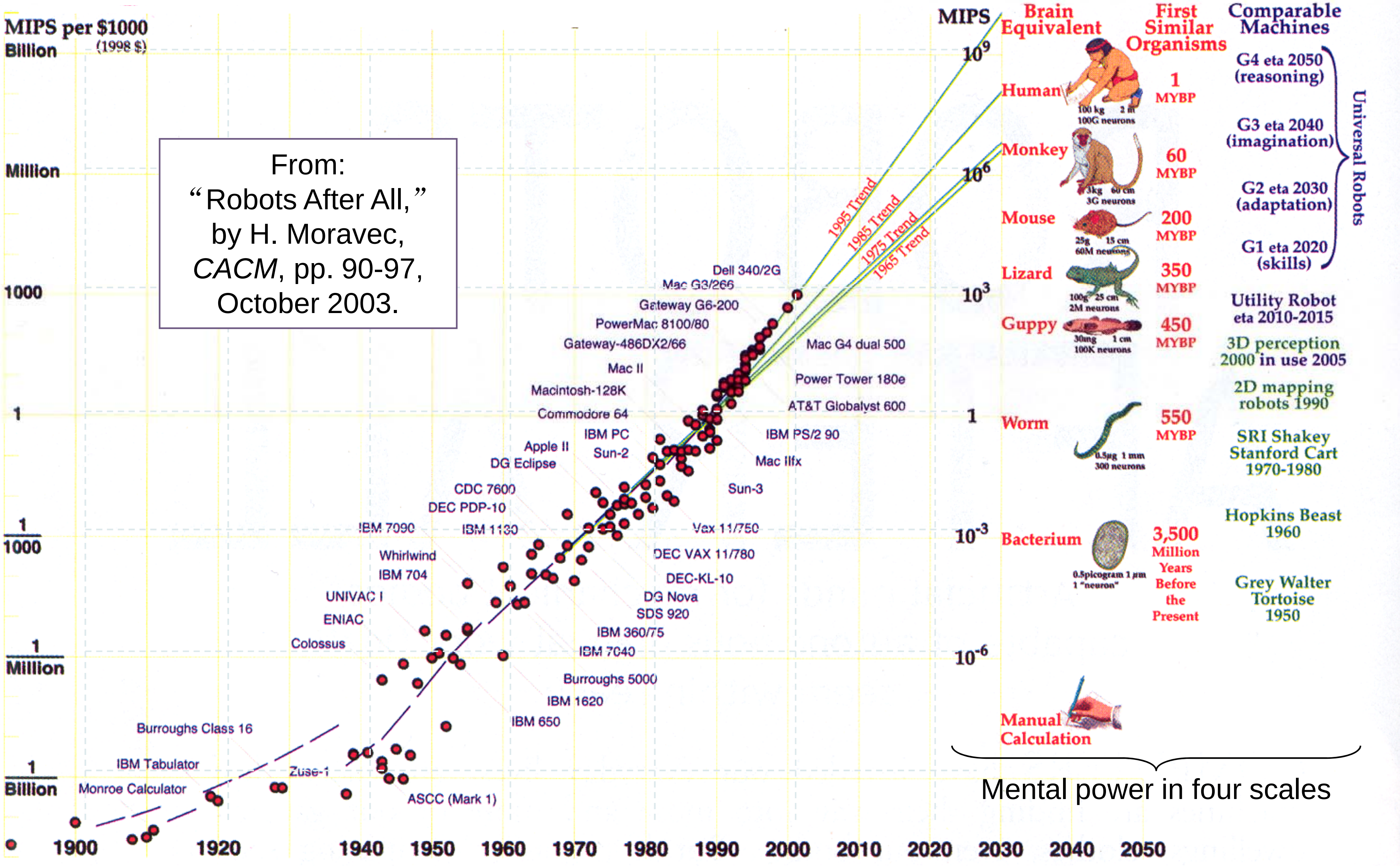


Fig. 1.1 Moore’s Law again(extrapolated).

Performance/Cost



From:
"Robots After All,"
by H. Moravec,
CACM, pp. 90-97,
October 2003.

Mental power in four scales

Semi-conductor Industry Roadmap

Year →	2001	2004	2007	2010	2013	2016
width (nm)	140	90	65	45	32	22
frequency. (GHz)	2	~~4~~	~~7~~	3.6 ~~12~~	4.1 ~~20~~	4.6 ~~30~~
Wiring levels	7	8	9	10	10	10
Voltage (V)	1.1	1.0	0.8	0.7	0.6	0.5
Power (W)	130	160	190	220	250	290

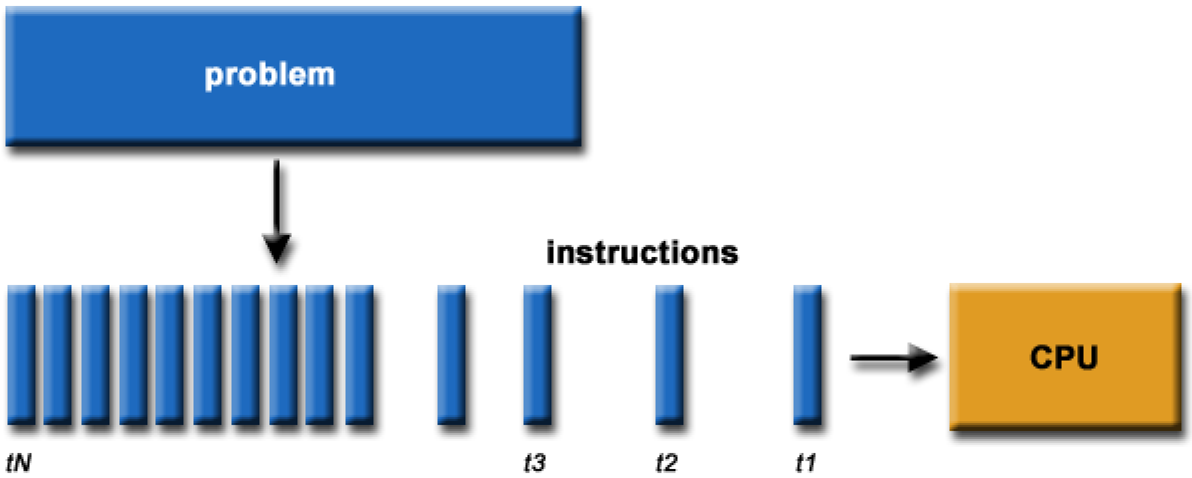
From the 2001 edition of the roadmap [Alla02]

2015	2020	2025
19	12	8
4.4	5.3	6.5
		0.6

From the 2011 edition (Executive Summary)

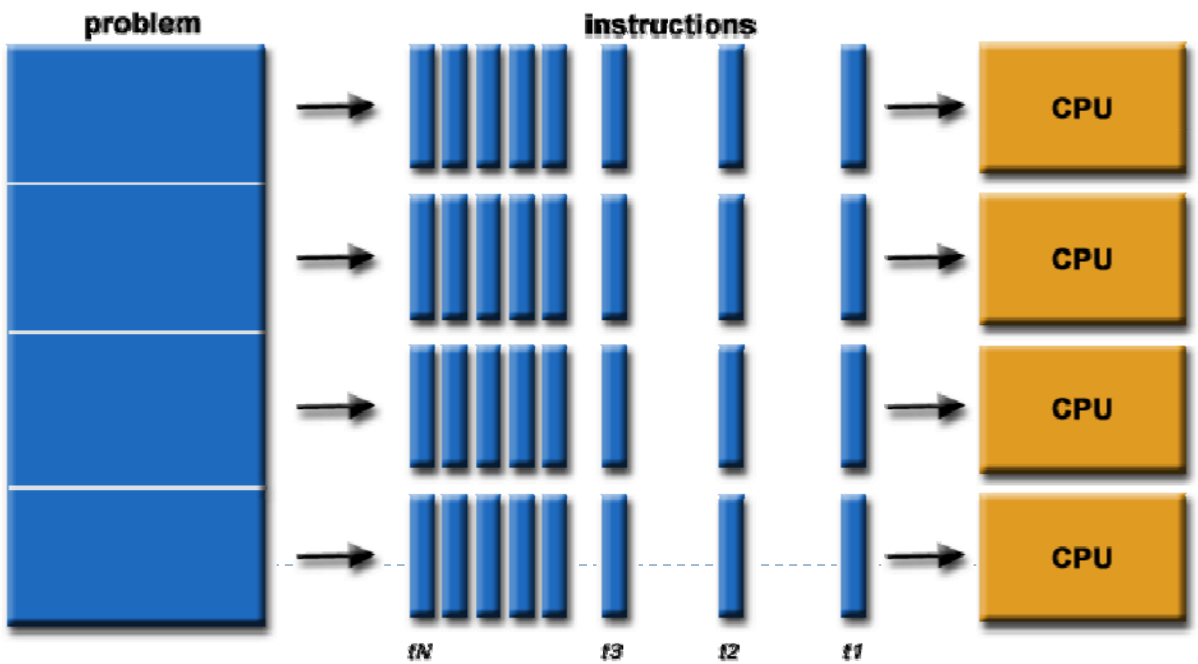
Serial Processing Model

- Traditionally, software has been written for serial computation:
  - To be run on a single computer having a single Central Processing Unit (CPU);
  - A problem is broken into a discrete series of instructions.
  - Instructions are executed one after another.
  - Only one instruction may execute at any moment in



Parallel Processing Model

- In the simplest sense, parallel computing is the simultaneous use of multiple compute resources to solve one computational problem.
  - To be run using multiple CPUs
  - A problem is broken into discrete parts that can be solved concurrently
  - Each part is further broken down to a series of instructions
- Instructions from each part execute simultaneously on different CPUs



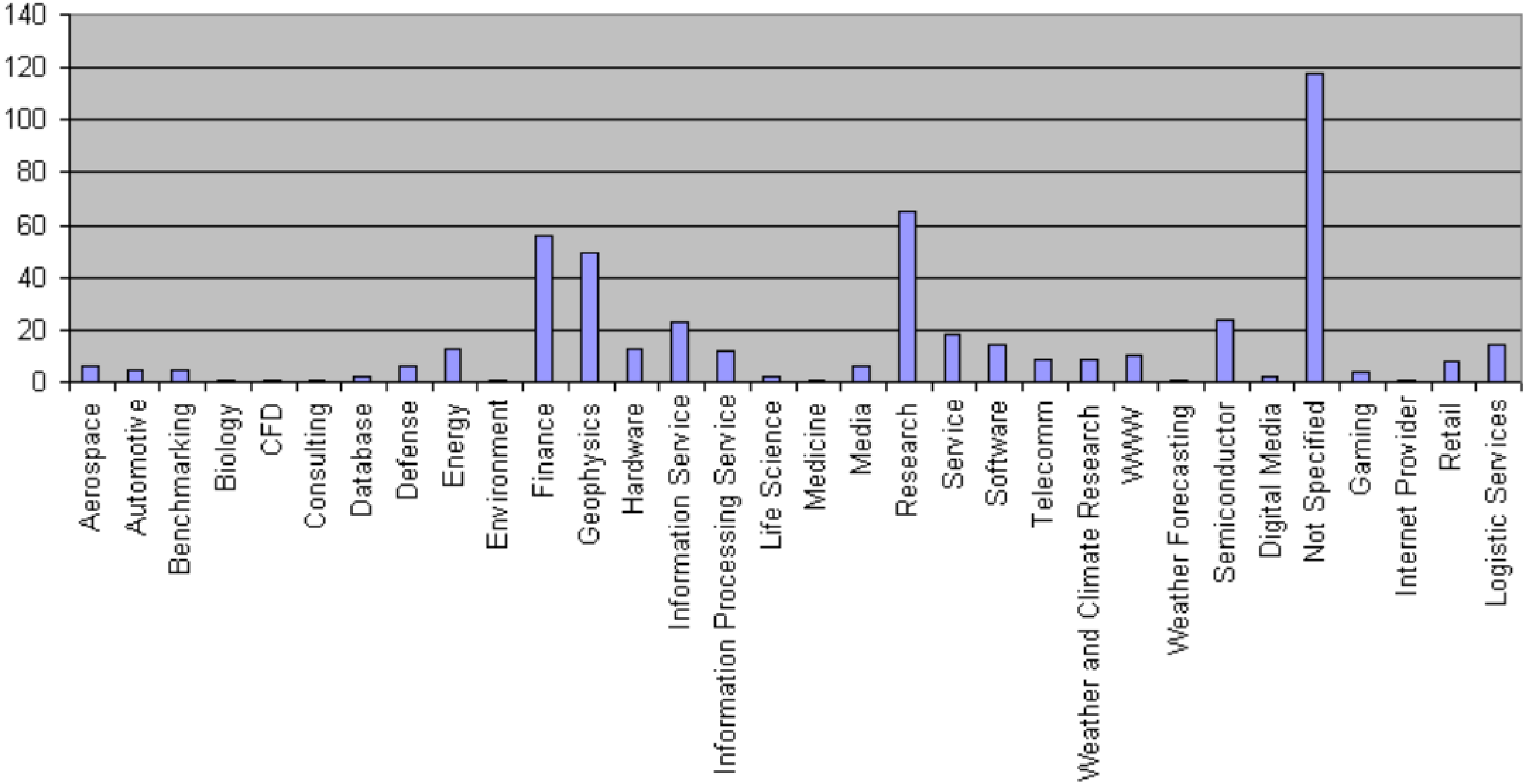
What problems can parallel computing solve?

- So many……
- x
- xx
- xxx
- xxxx

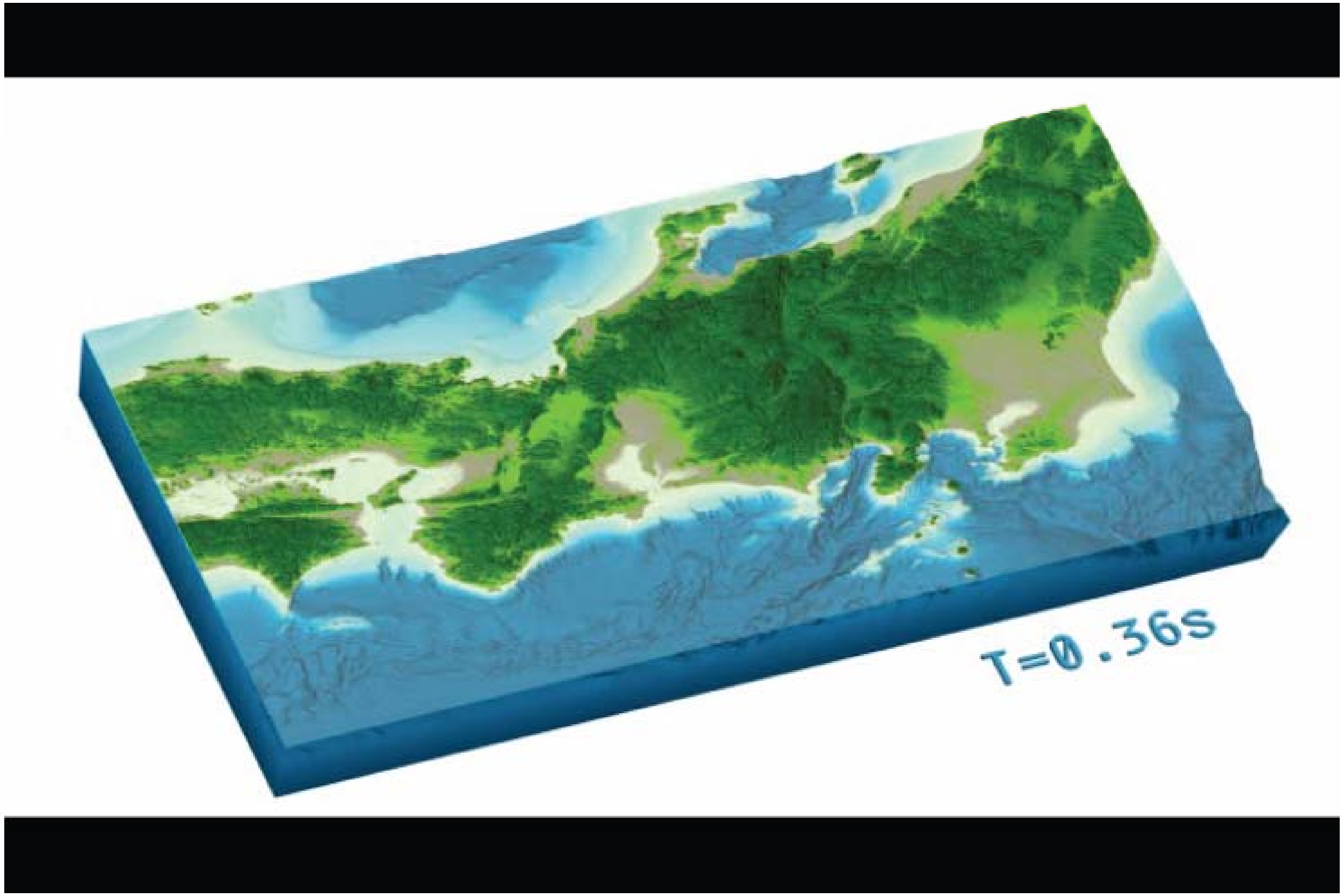
The Need for Speed: Complex Problems

- Science
  - —understanding matter from elementary particles to cosmology
  - —storm forecasting and climate prediction
  - —understanding biochemical processes of living organisms
- Engineering
  - —combustion and engine design
  - —computational fluid dynamics and airplane design
  - —earthquake and structural modeling
  - —pollution modeling and remediation planning
  - —molecular nanotechnology
- Business
  - —computational finance - high frequency trading
  - —information retrieval
  - —data mining
- Defense
  - —nuclear weapons stewardship
  - —cryptology

Applications of Parallel Processing



Earthquake Simulation

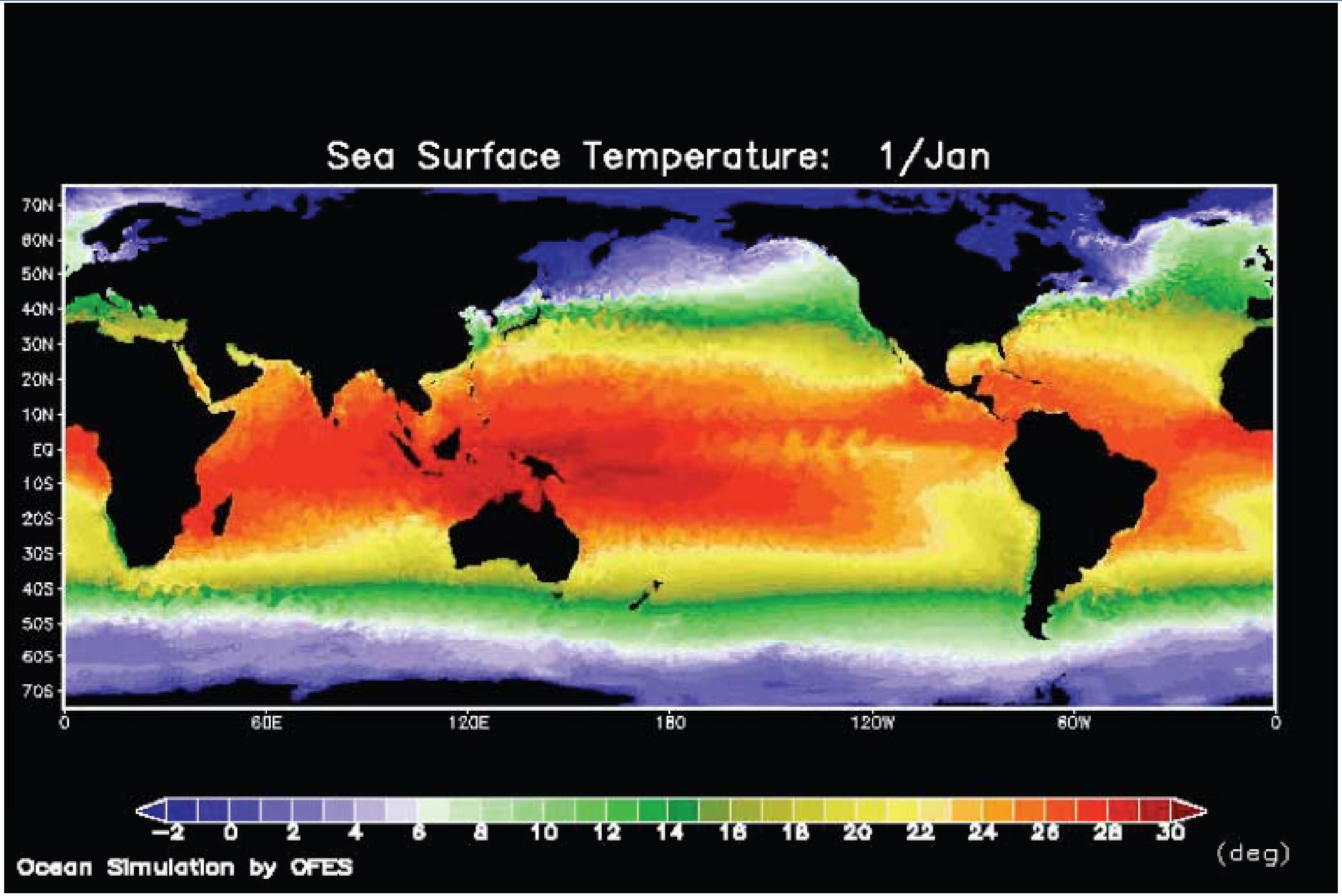


Earthquake Research Institute, University of Tokyo

Tonankai-Tokai Earthquake Scenario

Photo Credit: The Earth Simulator Art Gallery, CD-ROM, March 2004

Ocean Circulation Simulation



Ocean Global Circulation Model for the Earth Simulator

Seasonal Variation of Ocean Temperature

Photo Credit: The Earth Simulator Art Gallery, CD-ROM, March 2004

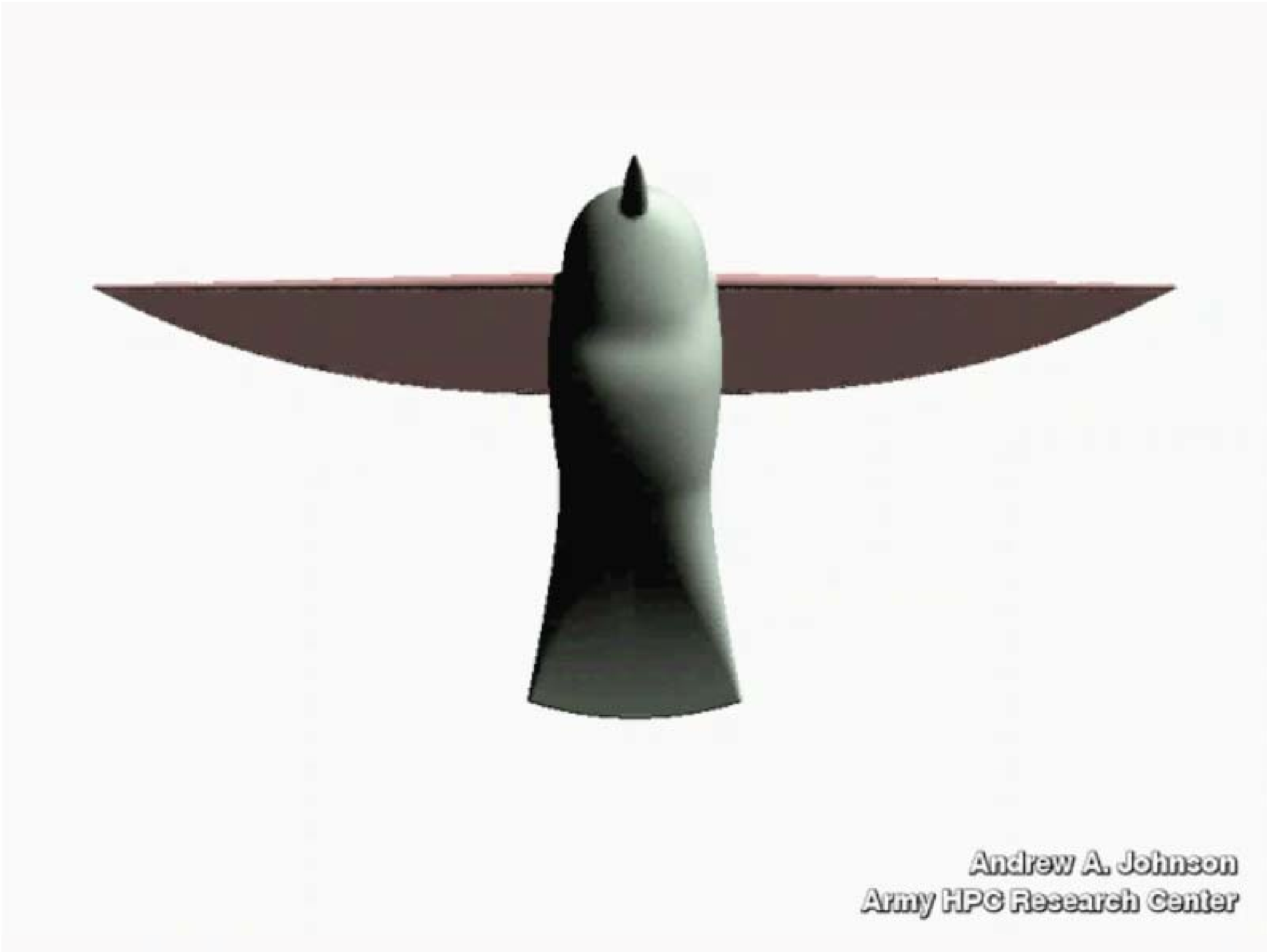
Fluid-Structure Interactions

- Simulate …
  —rotational geometries (e.g. engines, pumps), flapping wings
- Traditionally, such simulations have used a fixed mesh
  —drawback: solution quality is only as good as initial mesh
- Dynamic mesh computational fluid dynamics
  —integrate automatic mesh generation within parallel flow solver
    - nodes added in response to user-specified refinement criteria
    - nodes deleted when no longer needed
    - element connectivity changes to maintain minimum energy mesh

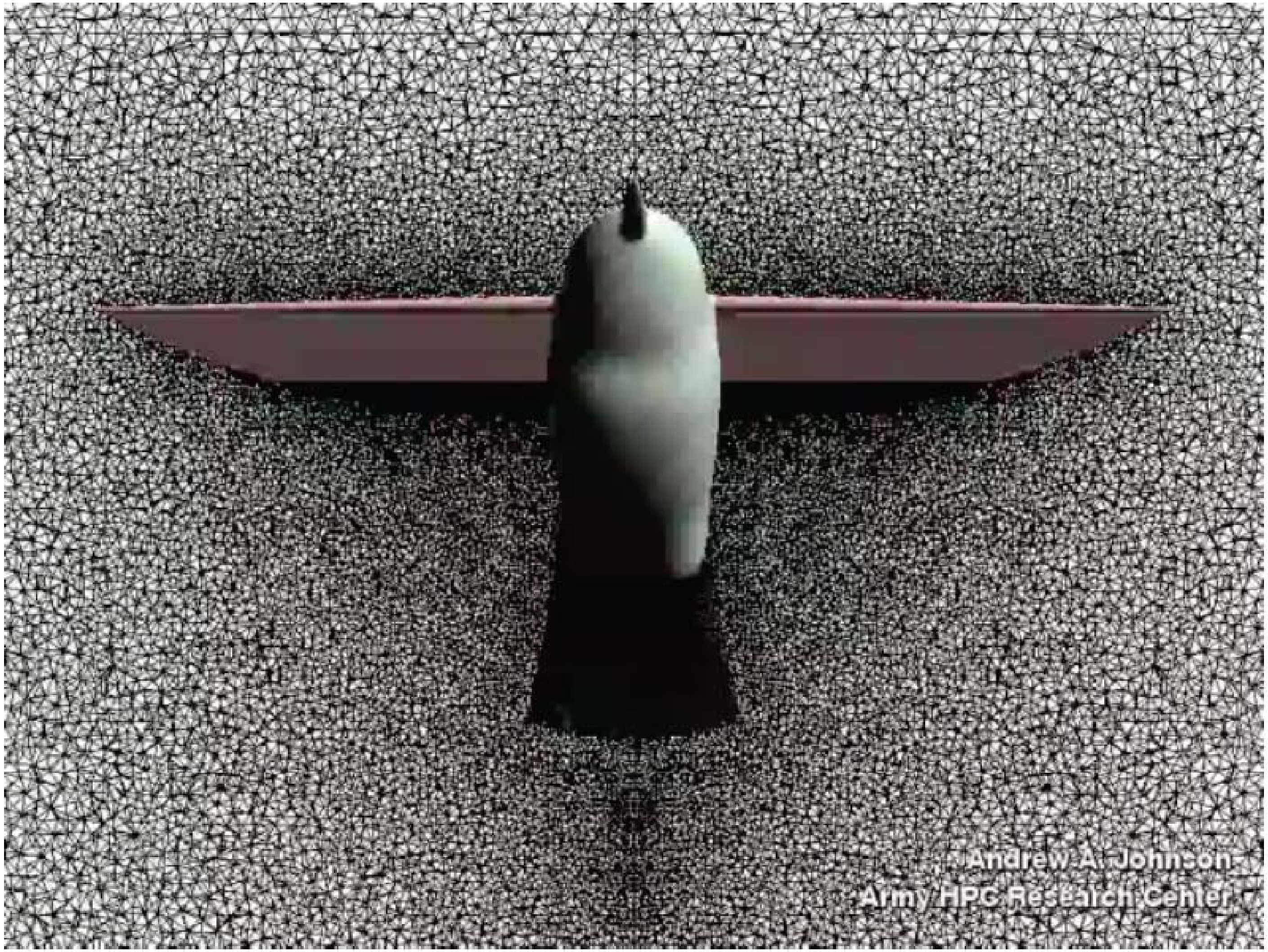
  —mesh changes continuously as geometry + solution changes
- Example: 3D simulation of a hummingbird's flight

[Andrew Johnson, AHPCRC 2005]

Air Velocity (Front)



Mesh Adaptation (front)



Some lists

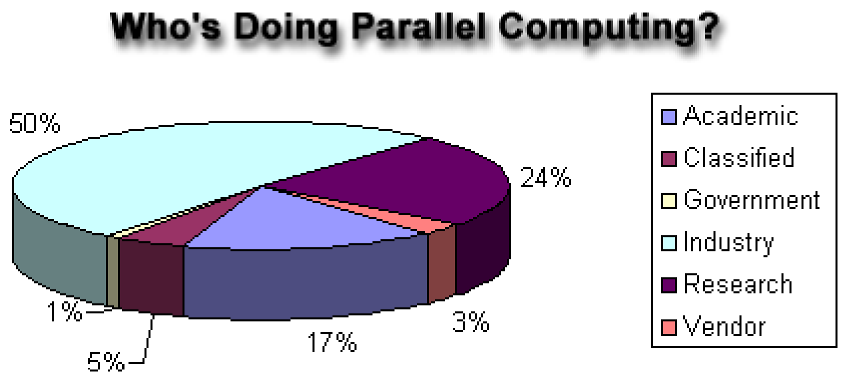
- weather and climate
- chemical and nuclear reactions
- biological, human genome
- geological, seismic activity
- mechanical devices - from prosthetics to spacecraft
- electronic circuits
- manufacturing processes

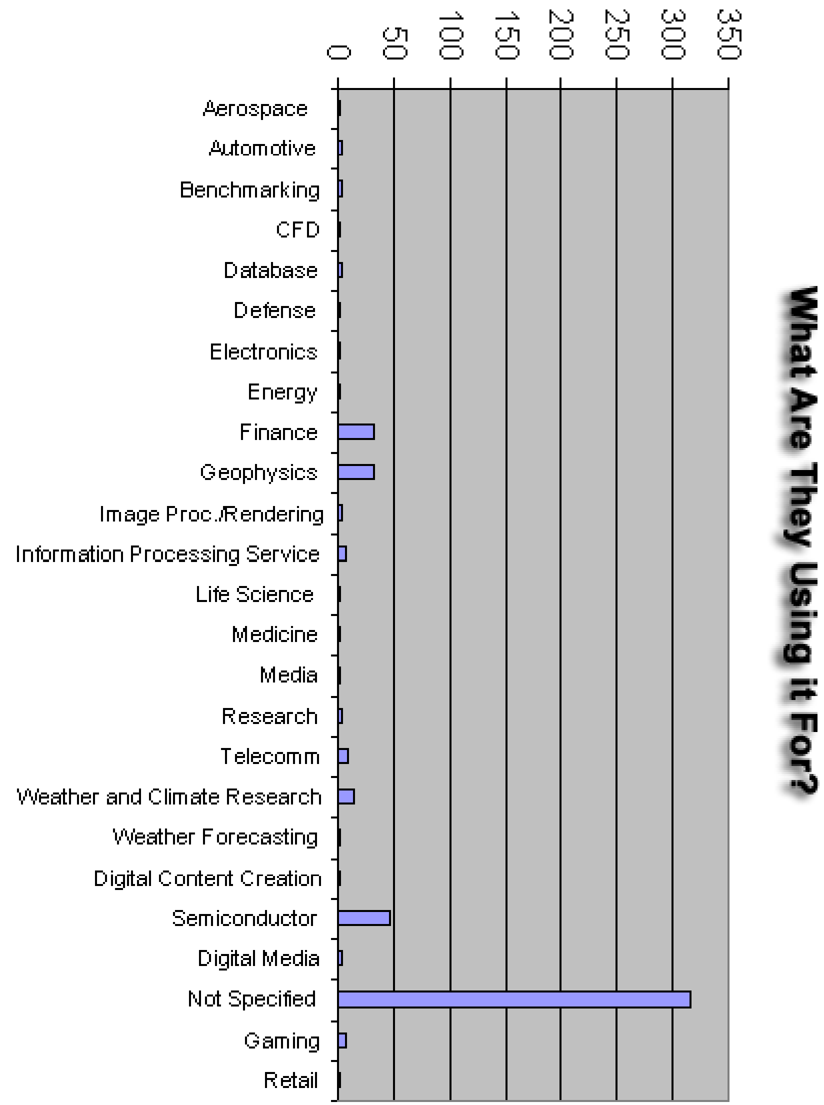
Daily life related

- Data Mining
- Search Engine,
- Online Shopping
- Medicine
- Game
- Video, Virtual Reality
- Even Cellphones

- Ultimately, parallel computing is an attempt to maximize the infinite but seemingly scarce commodity called time.

Parallel Computing Usage





Data obtained from top500.org, June 2006

Basic Concepts and Terms

Flynn Matrix

- The matrix below defines the 4 possible classifications according to Flynn

S I S D **Single Instruction, Single Data**	**S I M D** **Single Instruction, Multiple Data**
M I S D **Multiple Instruction, Single Data**	**M I M D** **Multiple Instruction, Multiple Data**

Terms in Parallel Computing

- Task
- Parallel Task
- Serial Execution
- Parallel Execution
- Shared Memory
- Distributed Memory
- Communications
- Synchronization
- Granularity
- Observed Speedup
- Parallel Overhead
- Scalability

Parallel Computer Memory Architectures

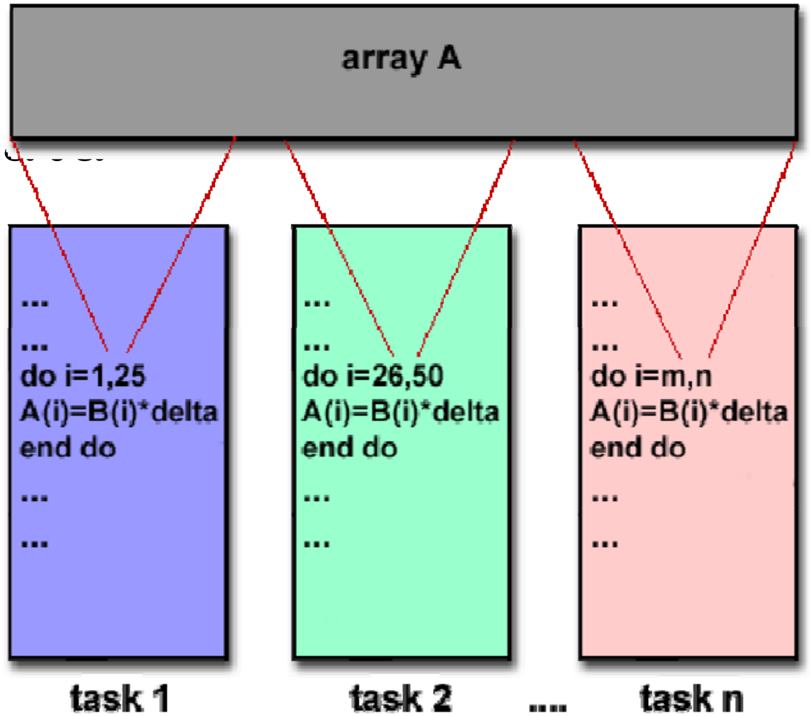
- Shared Memory
- Distributed Memory
- Hybrid Distributed-Shared Memory

Parallel Programming Models

- Shared Memory Model
- Threads Model
- Message Passing Model
- Data Parallel Model

Some Implementations

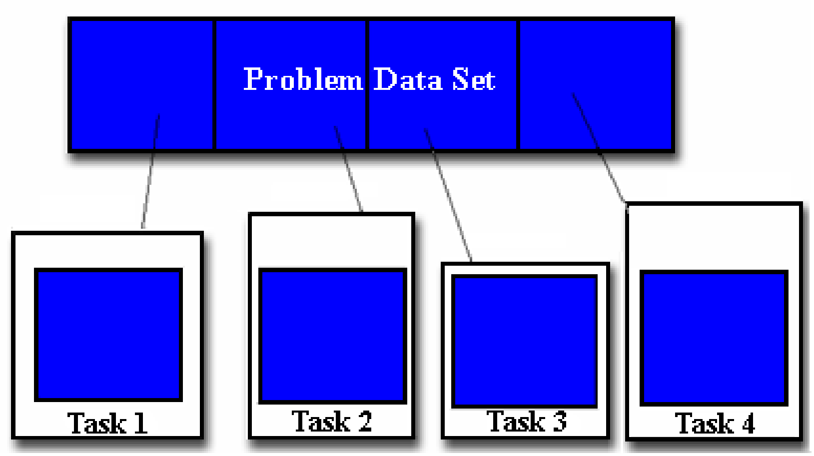
- OpenMP
- MPI
- Single Program Multiple Data
- (SPMD)
- Multiple Program Multiple
- (MPMD)

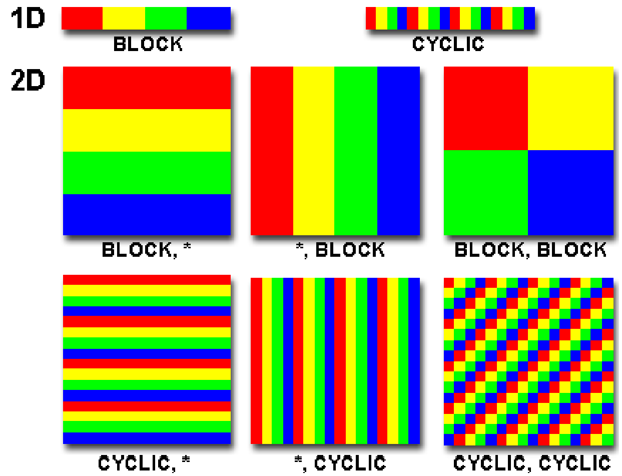


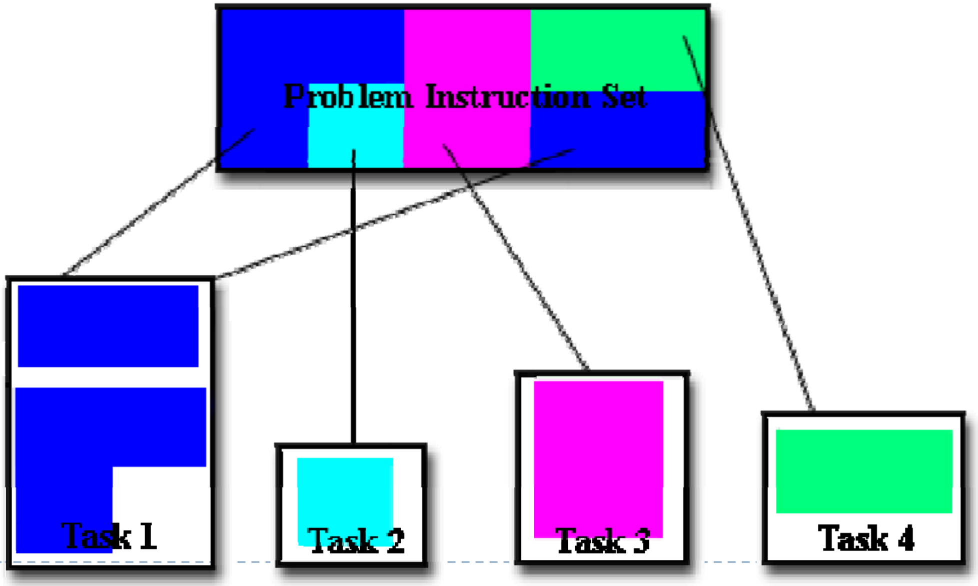
Designing Parallel Programs

- Automatic vs. Manual Parallelization
- Understand the Problem and the Program
- Partitioning
- Communications
- Synchronization
- Data Dependencies
- Load Balancing
- Granularity
- I/O
- Limits and Costs of Parallel Programming
- Performance Analysis and Tuning

Domain Decomposition

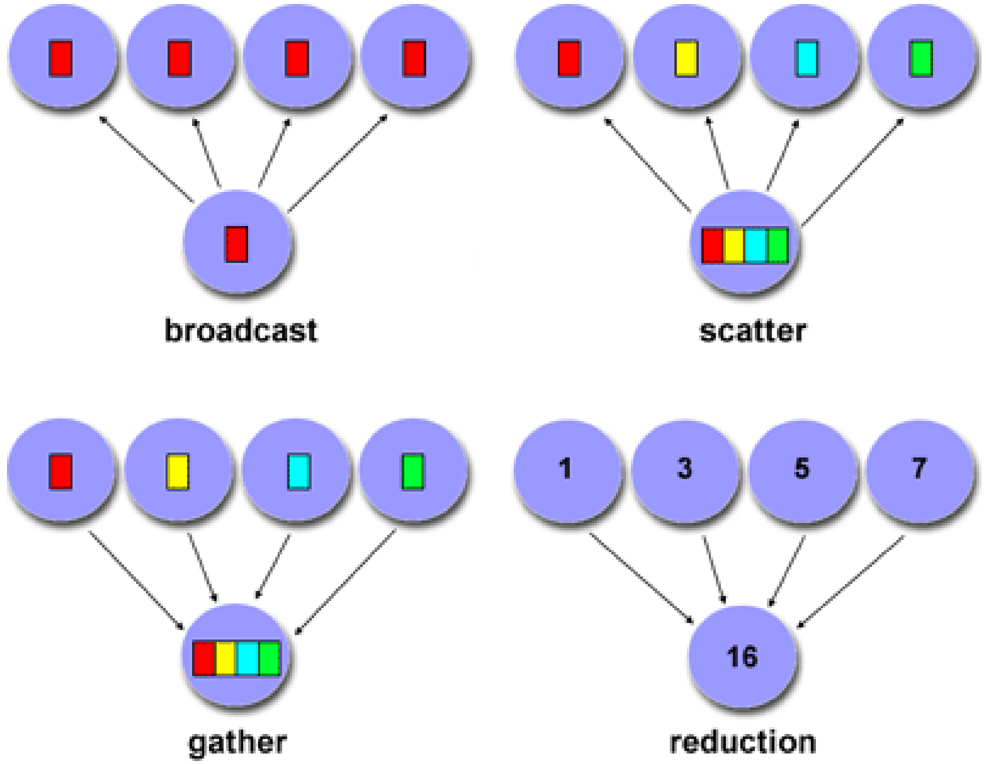






Communication and Synchronization

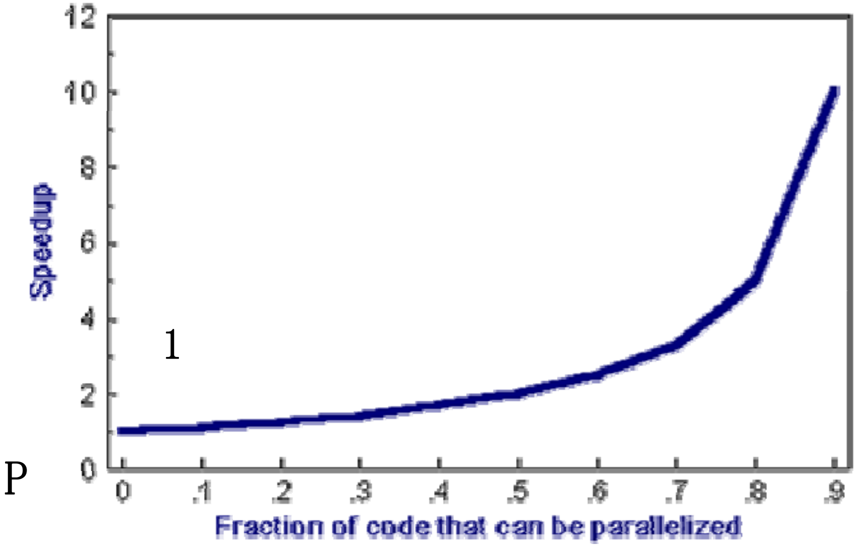
- Considerations:
  - Barrier
  - Lock / semaphore
  - Synchronous communication



Amdahl's Law

- Amdahl's Law states that potential program speedup is defined by the fraction of code (P) that can be parallelized:

$$\text{speedup} = \frac{1}{1 - P}$$



- If none of the code can be parallelized, P = 0 and the speedup = 1 (no speedup). If all of the code is parallelized, P = 1 and the speedup is infinite (in theory).
- If 50% of the code can be parallelized, maximum speedup = 2, meaning the code will run twice as fast

Amdahl's Law

- Introducing the number of processors performing the parallel fraction of work, the relationship can be modeled by

```
speedup   =      1
              -----------
               P   +  S
              ---
               N
```

- where P = parallel fraction, N = number of processors and S = serial fraction

Amdahl's Law

- It soon becomes obvious that there are limits to the scalability of parallelism. For example, at P = .50, .90 and .99 (50%, 90% and 99% of the code is parallelizable)

	speedup		
N	P = .50	P = .90	P = .99
10	1.82	5.26	9.17
100	1.98	9.17	50.25
1000	1.99	9.91	90.99
10000	1.99	9.91	99.02