UCIrvine | The Henry Samueli School of Engineering

Introduction and Course Overview

EECS 221: Intro to High-Performance Computing

Aparna Chandramowlishwaran

April 4, 2017

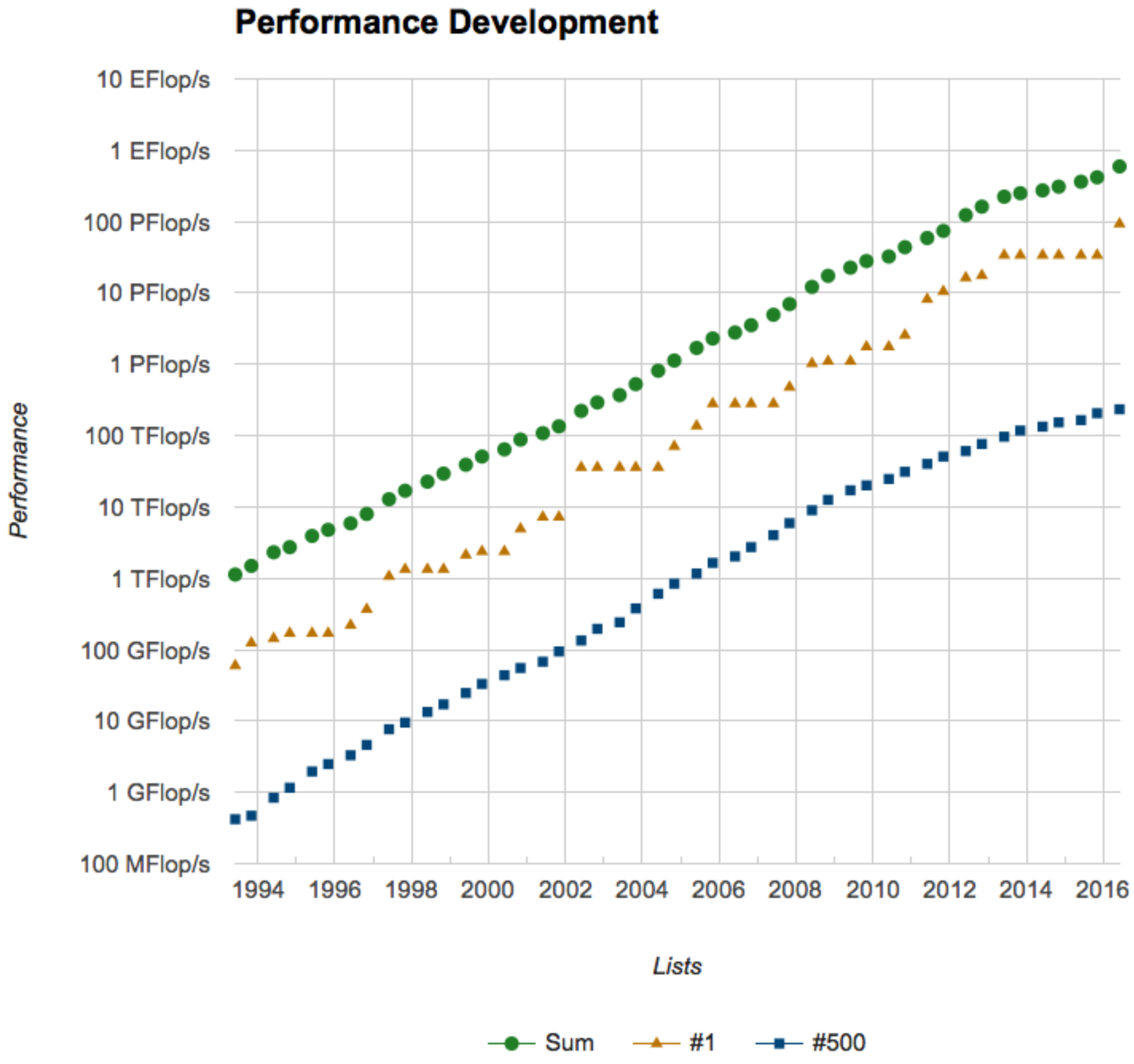
Course Goal

- Learn fundamentals of parallel computing
  - Principles of parallel algorithm design
  - State-of-the-art programming models
  - Performance analysis of parallel programs
- Learn practical performance engineering
  - Software fundamentals to tackle *grand challenge* problems
  - Ideally, able to program supercomputers

Course Goal

- Learn fundamentals of parallel computing
  - Principles of parallel algorithm design
  - State-of-the-art programming models
  - Performance analysis of parallel programs
- Learn practical performance engineering
  - Software fundamentals to tackle *grand challenge* problems
  - Ideally, able to program supercomputers

EECS 221: Theory + Practice



http://www.top500.org/statistics/perfdevel/

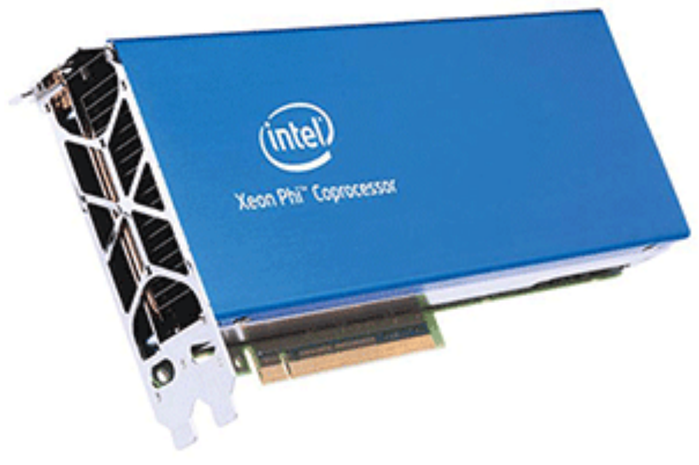
Sunway TaihuLight supercomputer
10.6 million cores
93 x 10^{15} floating-point op/s (93 Pflop/s)
15.4 Megawatts

https://www.top500.org/resources/top-systems/sunway-taihulight-national-supercomputing-center-i

http://hexus.net/tech/news/systems/93845-worlds-fastest-supercomputer-93-petaflop-sunway-taihulight

Intel Xeon *IvyBridge*
12 cores @ 2.2 GHz

+



Tianhe-2 *Milkyway-2* supercomputer
3.1 million cores
34 x 10^{15} floating-point op/s (34 Pflop/s)
17.8 Megawatts (~ $17 M / yr or ~ 10,000 homes)

http://www.geeky-gadgets.com/wp-content/uploads/2013/06/Tianhe-2.jpg

Titan supercomputer
560,640 cores
17.6 x 10^{15} floating-point op/s (17.6 Pflop/s)
8.2 Megawatts

AMD Operon *Interlagos*
16 cores @ 2.2 GHz

+

https://www.olcf.ornl.gov/wp-content/themes/olcf/titan/images/gallery/titan2-tn.jpg
http://images.anandtech.com/doci/6446/TeslaK20.jpg

Why is parallelism everywhere?

Tunnel Vision

Tunnel Vision

“I think there is a world market for maybe five computers.”

Thomas Watson, chairman of IBM, 1943.

Tunnel Vision

“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.

Tunnel Vision

“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.

Tunnel Vision

“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”

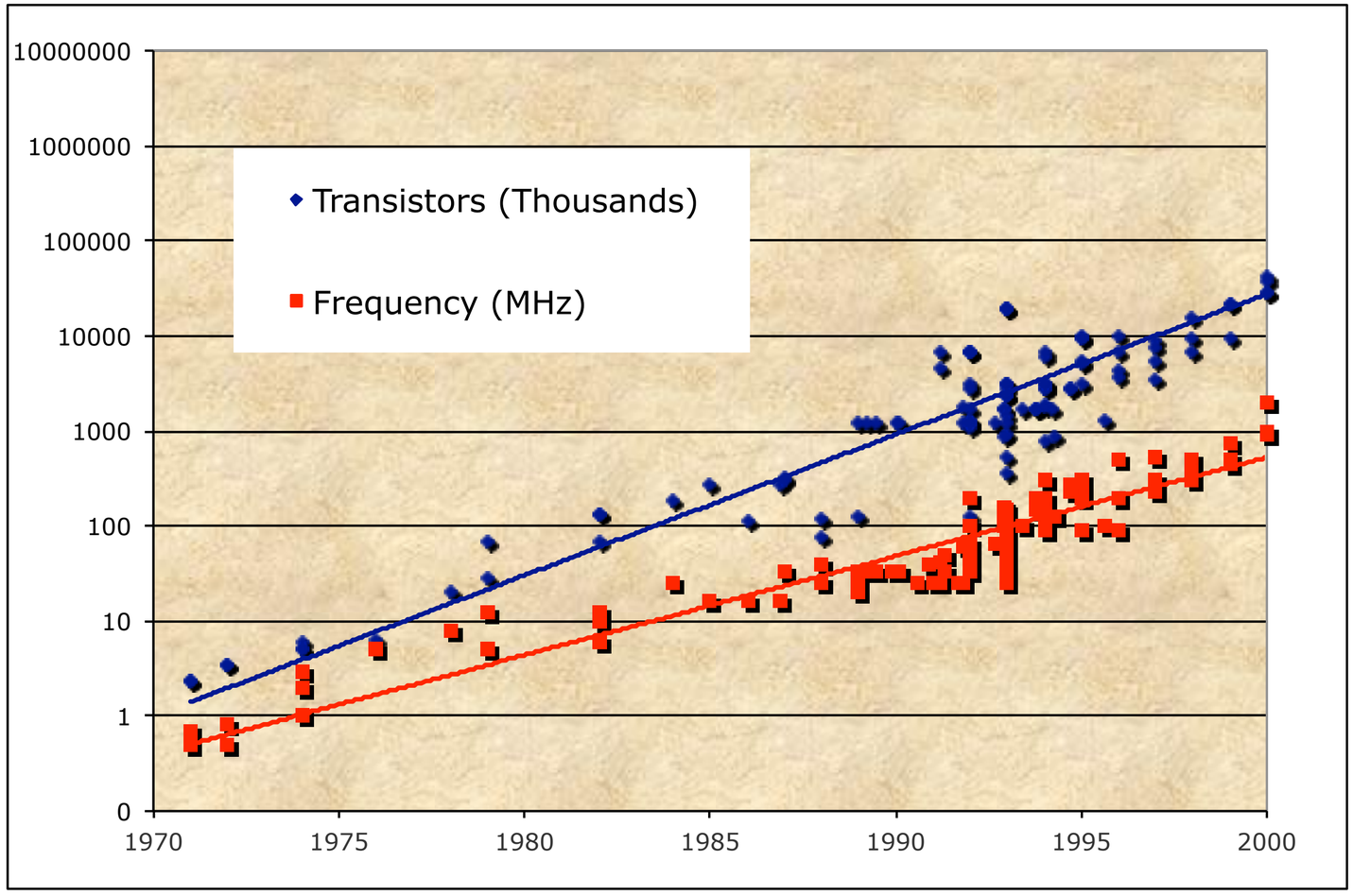
Ken Olson, president and founder of Digital Equipment Corporation, 1977.

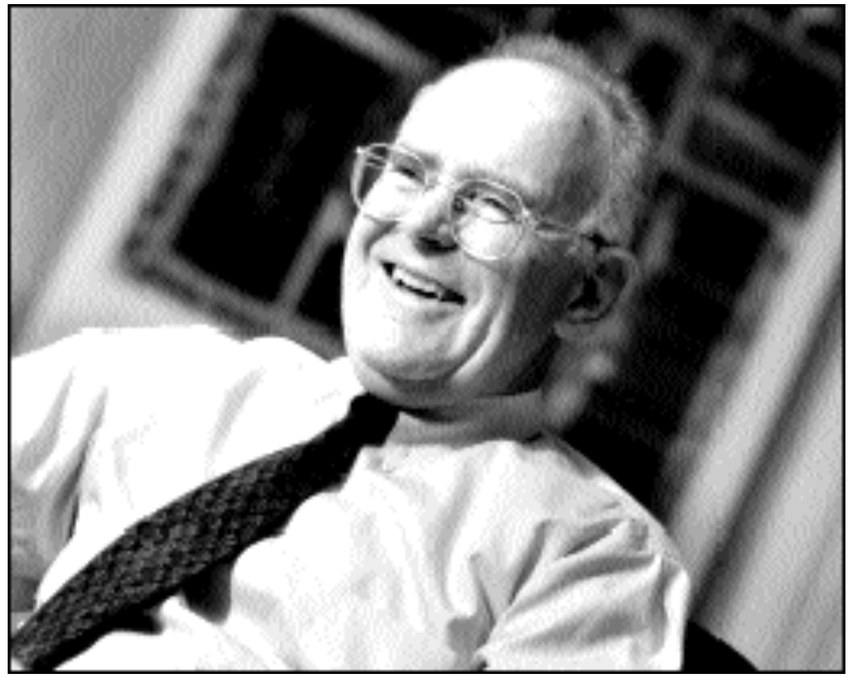
“640K [of memory] ought to be enough for anybody.”

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





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

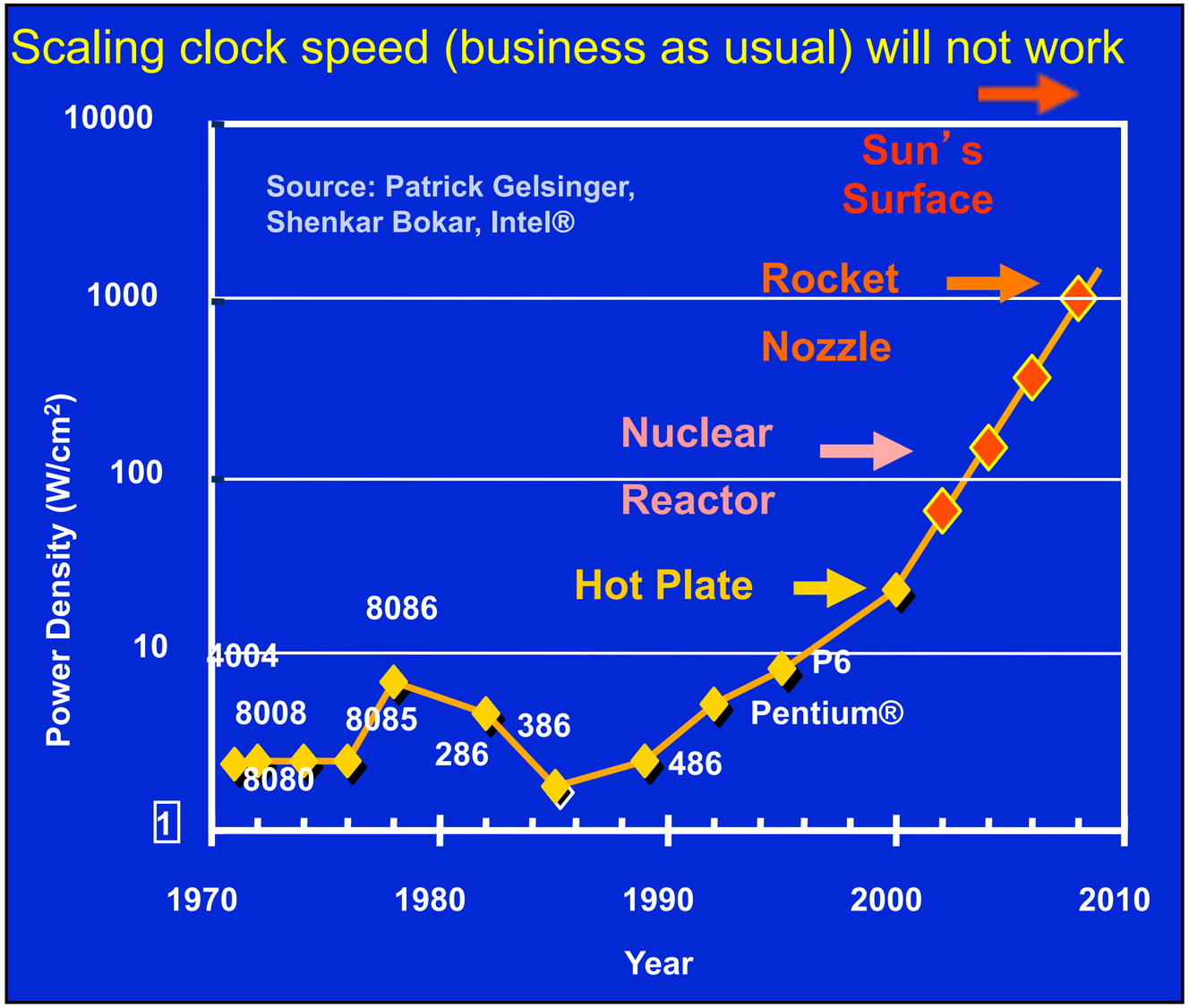
Microprocessor Transistors/Clock

Microprocessors have become smaller, denser, and more powerful.

Why is parallelism everywhere?

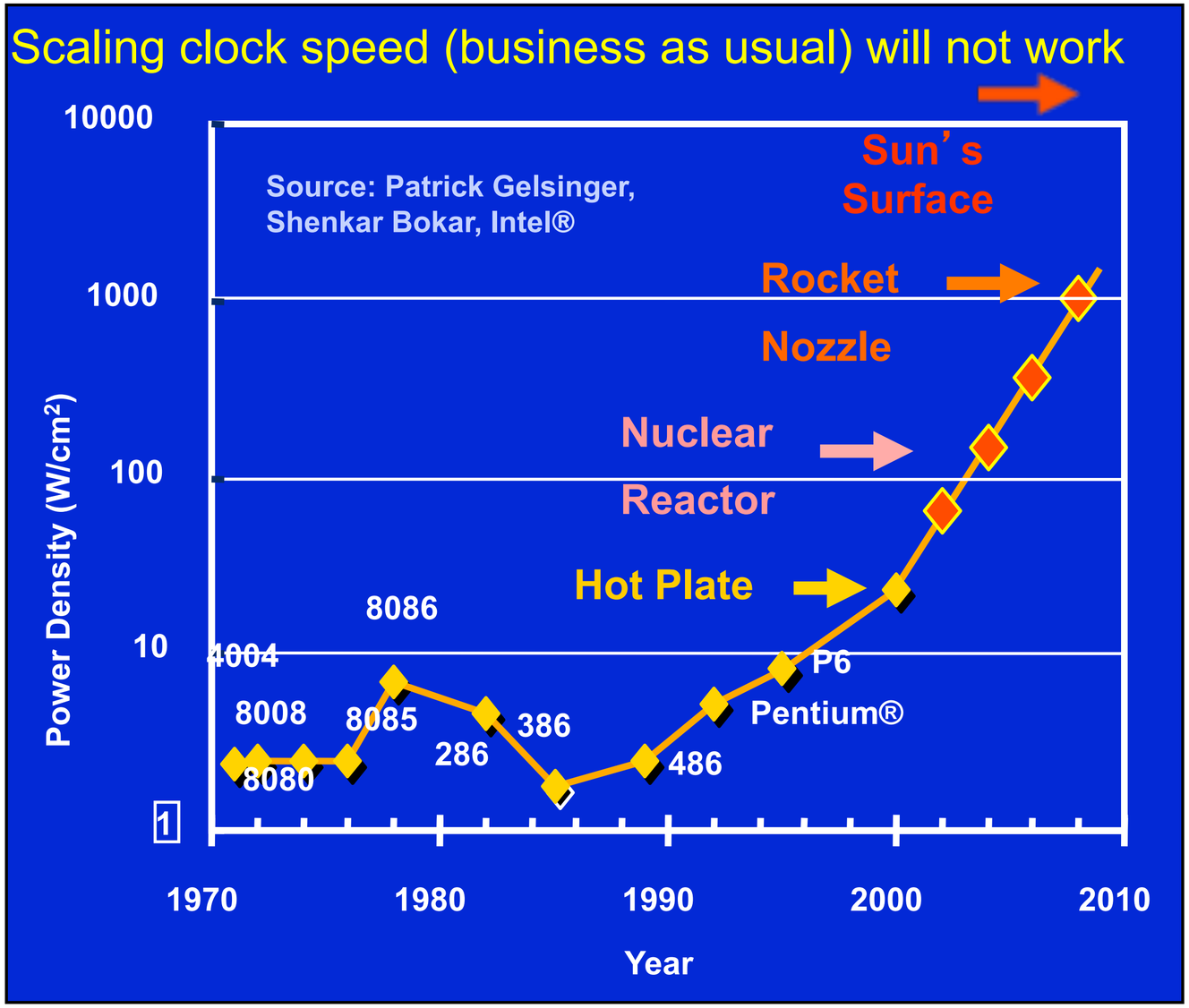
Why is parallelism everywhere?

Driven by power and memory constraints



POWER WALL

Is it better to increase performance by doubling frequency or cores?

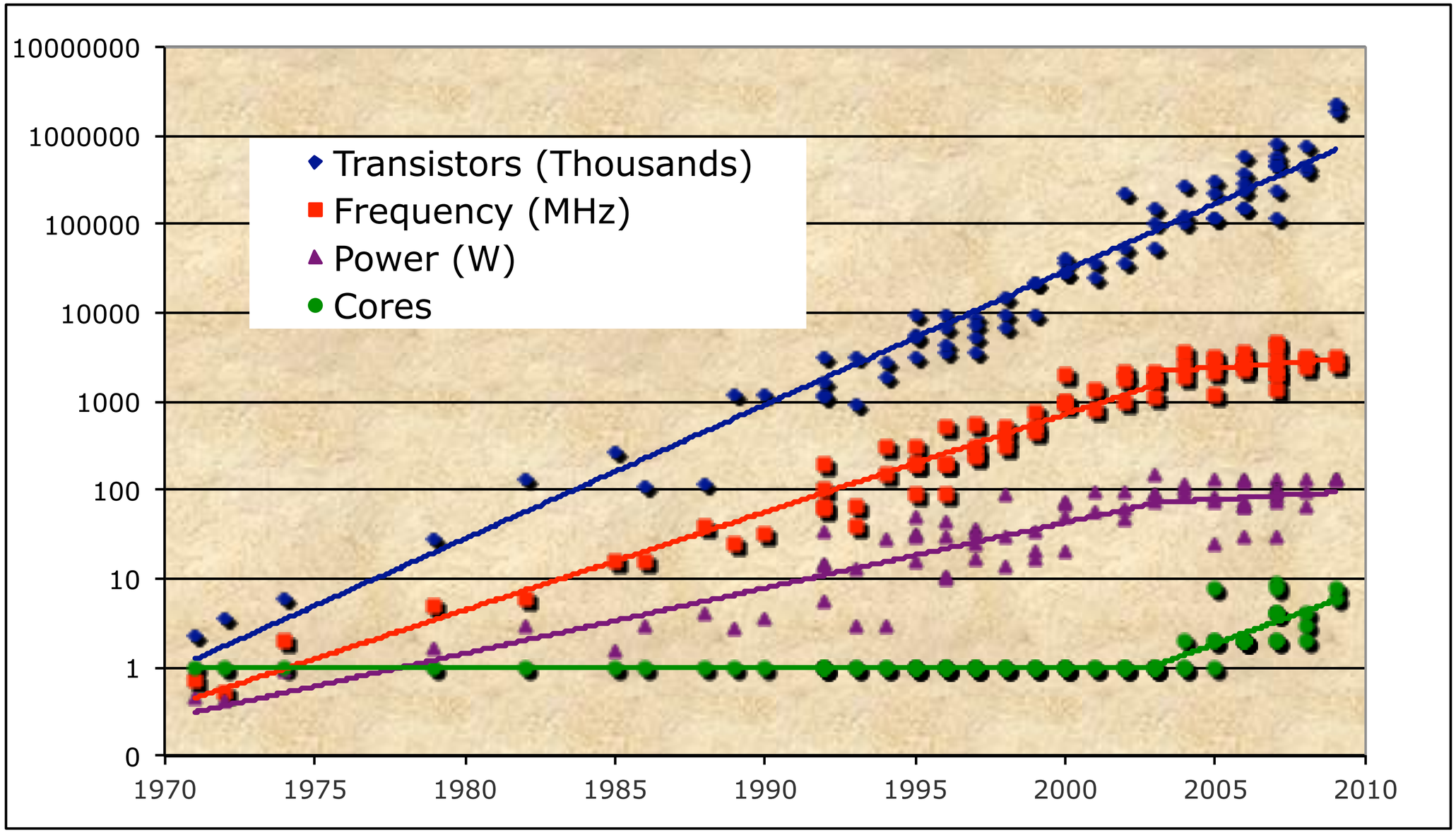


Performance ~ cores x freq

Power ~ V^2 C freq

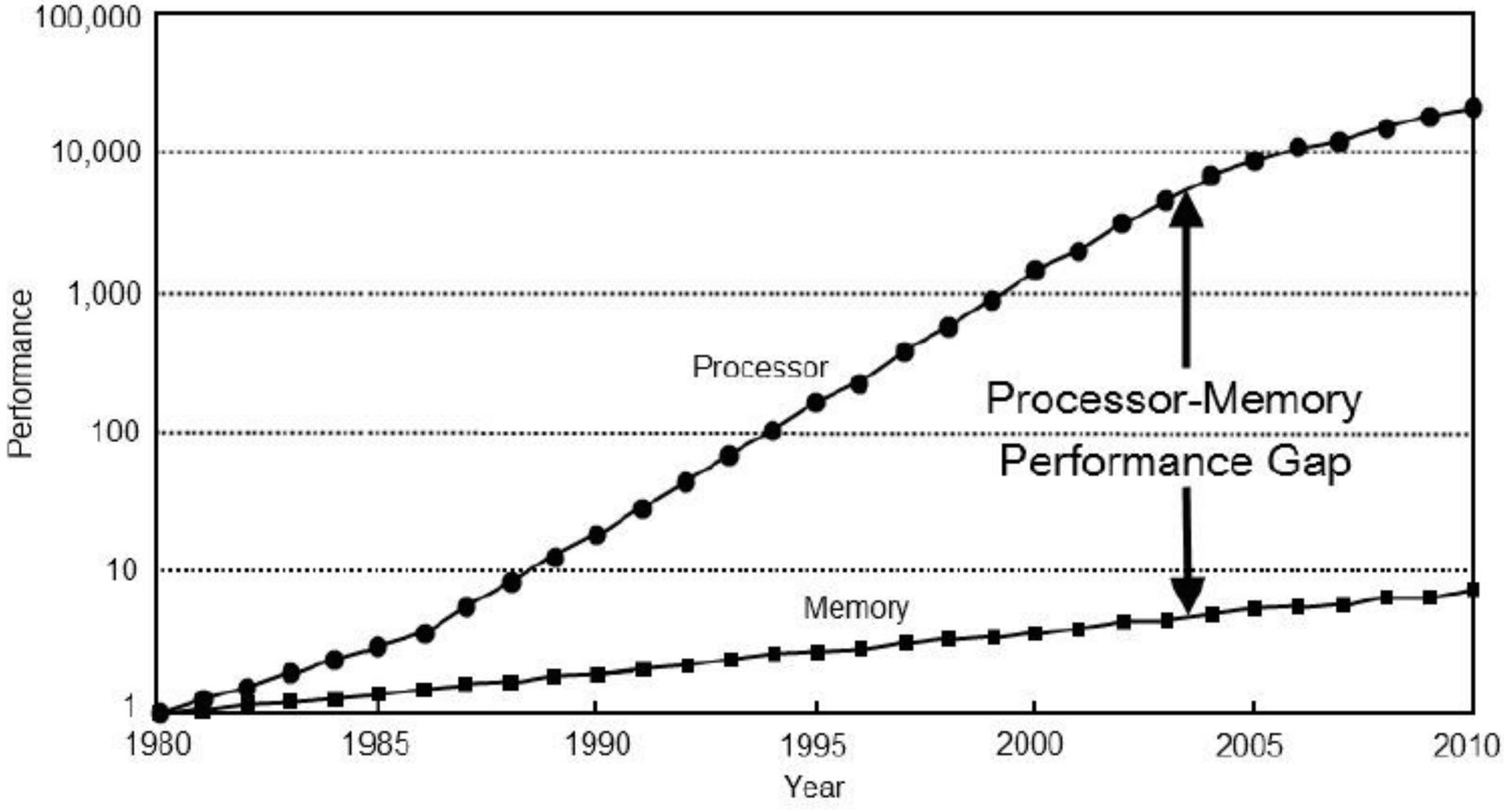
POWER WALL

Is it better to increase performance by doubling frequency or cores?



MULTICORE REVOLUTION

Power and frequency are no longer growing, but chip density continues to double every 2 years.



Processor Memory Gap

Cost of accessing data from memory is exponentially more than the cost of a floating point operation.

Little's Law in HPC

- **Latency**: Every operation takes time to execute (i.e. instruction, memory or network).
- **Bandwidth**: Rate at which operations can be completed
- **Concurrency**: Total number of operations executing simultaneously

Little's Law in HPC

- **Latency**: Every operation takes time to execute (i.e. instruction, memory or network).
- **Bandwidth**: Rate at which operations can be completed
- **Concurrency**: Total number of operations executing simultaneously

Concurrency = Latency x **Bandwidth**

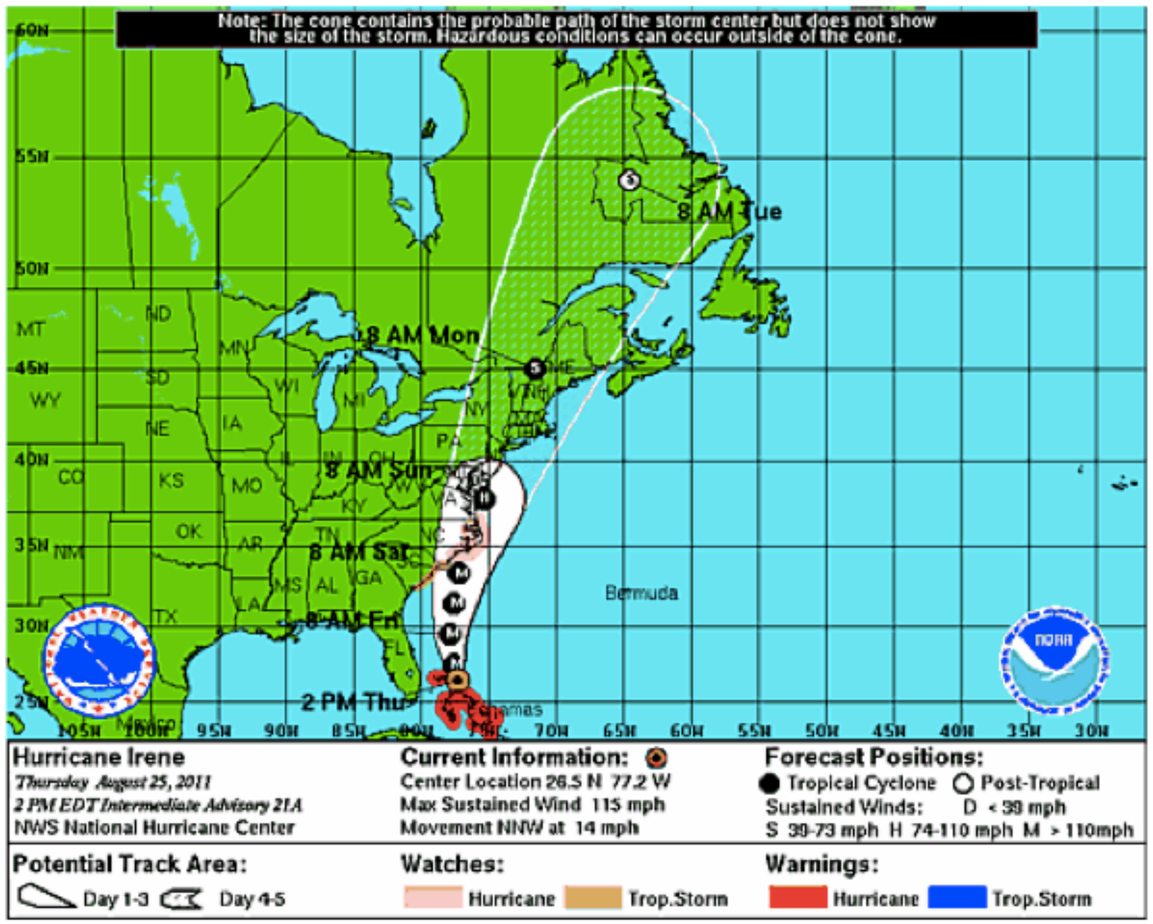
(or)

Effective Throughput = **Concurrency** / Latency

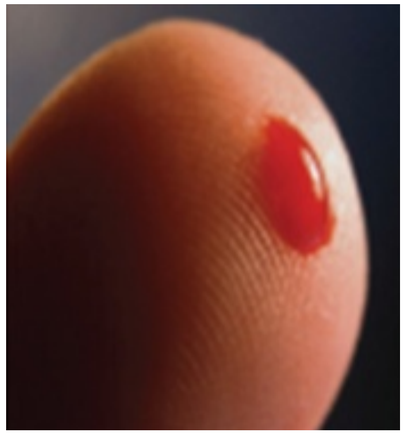
Need for Speed: Motivating applications in science and engineering

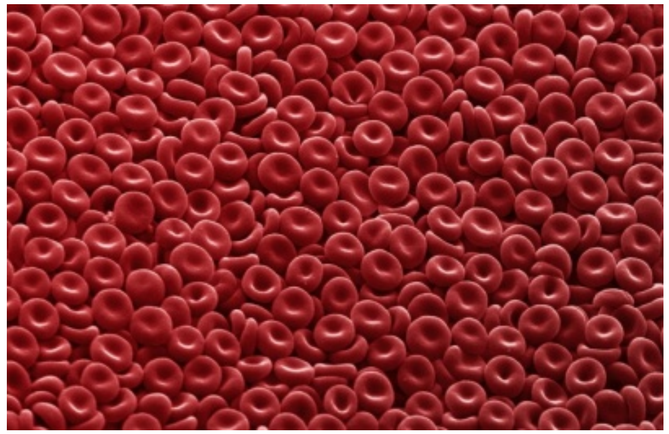
Predicting Irene

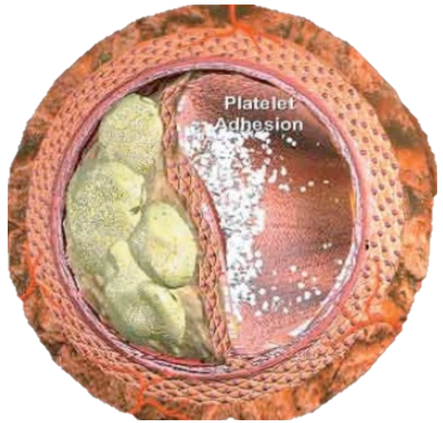
POSTED BY: TEKLA PERRY / THU, AUGUST 25, 2011

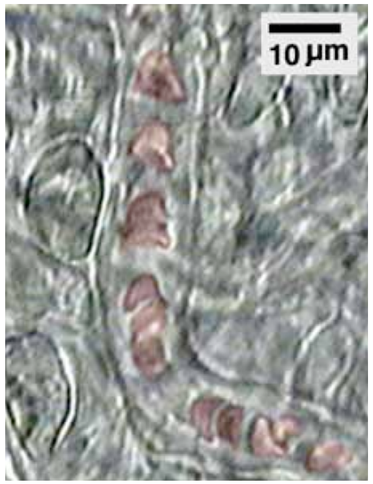


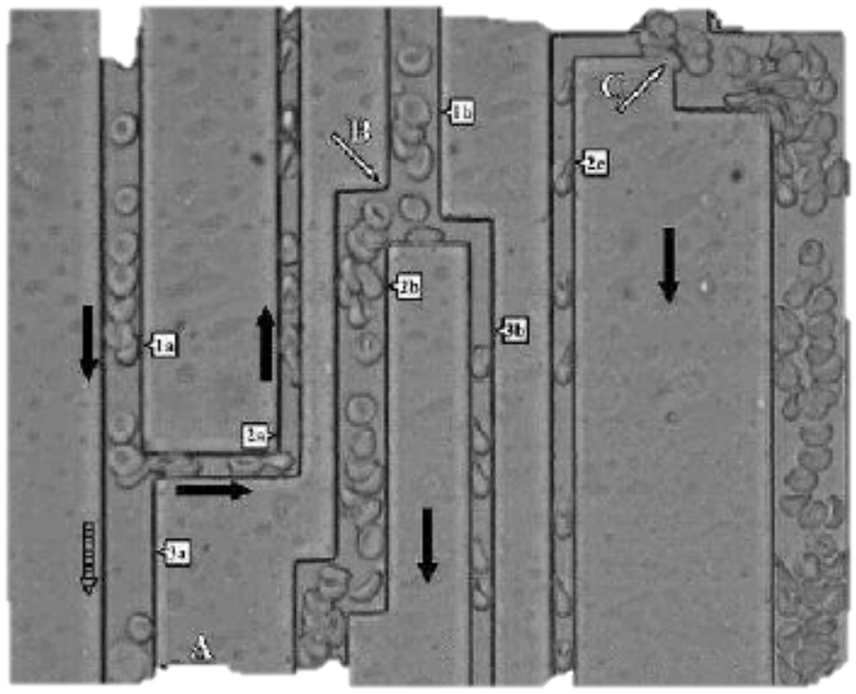
Hurricane prediction is by no means a perfect science. Supercomputers crunch vast amounts of data to model a hurricane's path, taking information from sensors around the world, including balloons, aircraft, ships, and satellites, but it's a patchwork collection at best. And there's a practical limit to how much data the models can consider—a simulation that takes more time to calculate than the storm takes to advance isn't very practical.

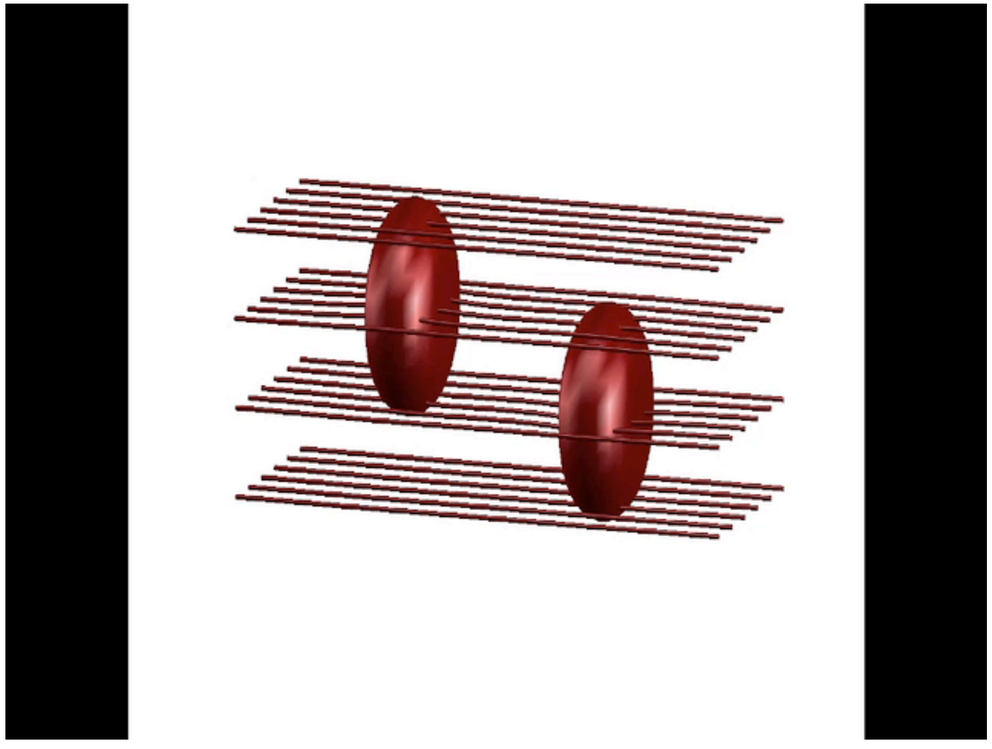








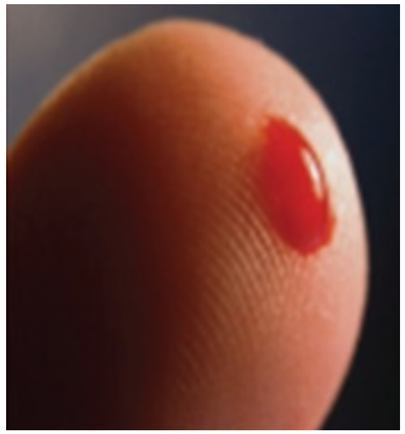


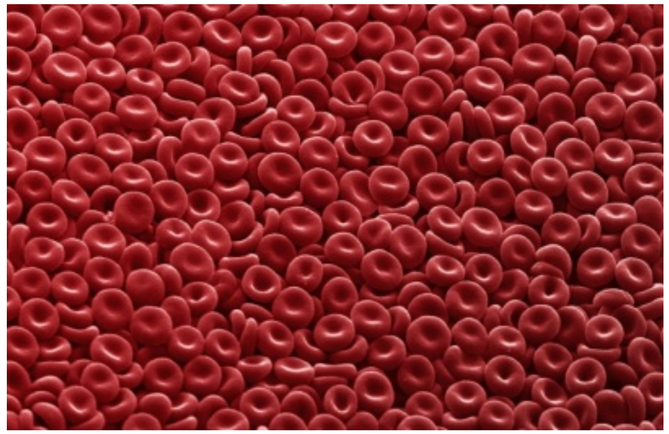


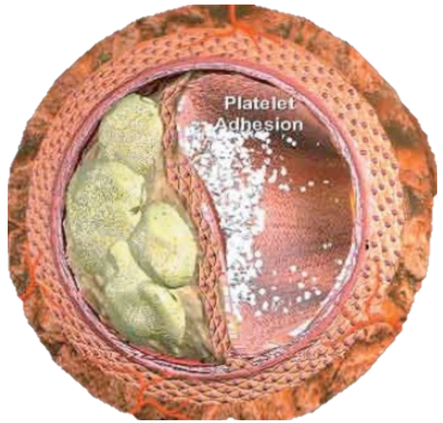
Blood Flow Simulation: MoBo

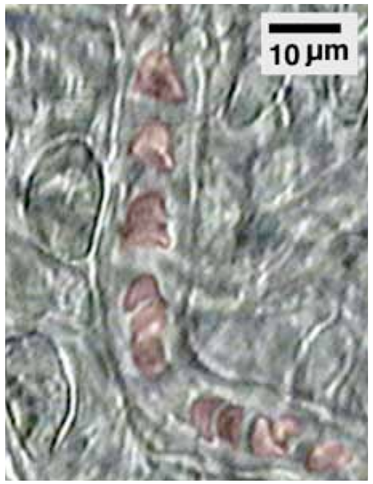
("Moving Boundaries")

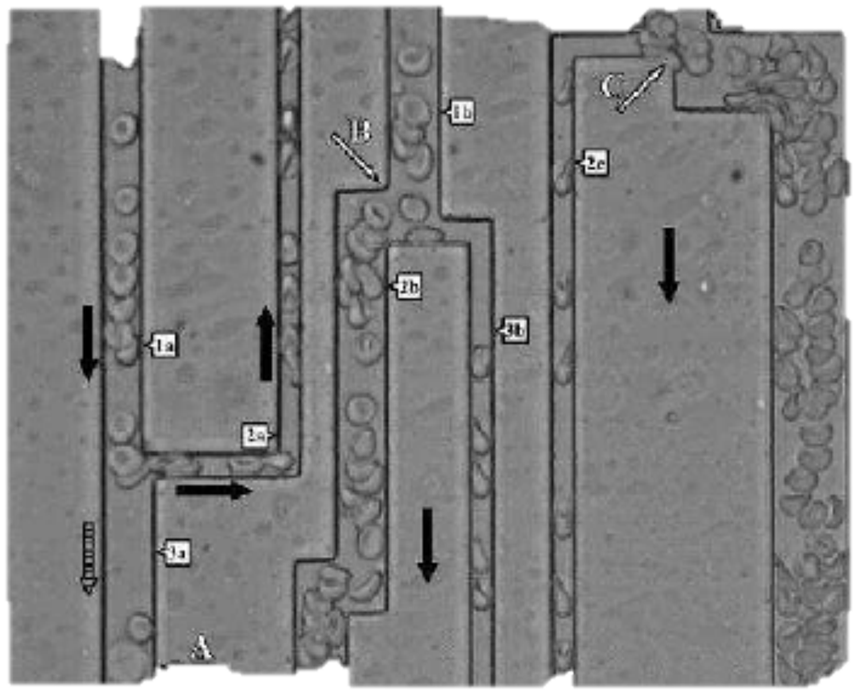
Citation: A. Rahimian, I. Lashuk, A. Chandramowlishwaran, D. Malhotra, L. Moon, R. Sampath, A. Shringarpure, S. Veerapaneni, J. Vetter, R. Vuduc, D. Zorin, and G. Biros. "Petascale direct numerical simulation of blood flow on 200k cores and heterogeneous architectures." In *SC'10.* **Winner, Gordon Bell Prize**. http://dx.doi.org/10.1109/SC.2010.42

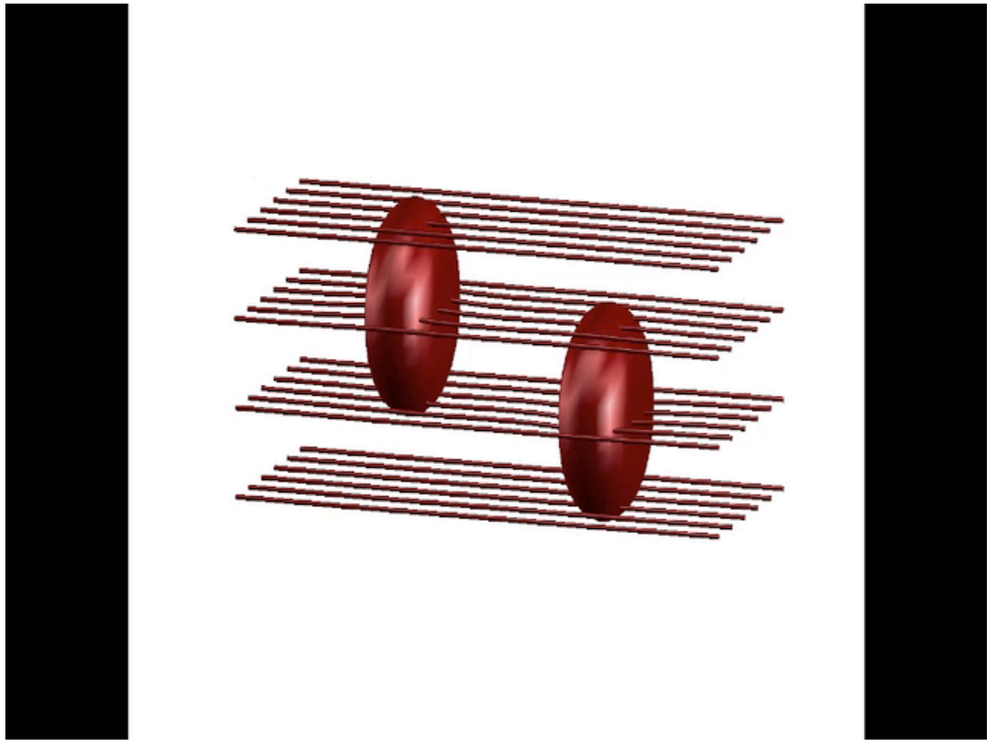












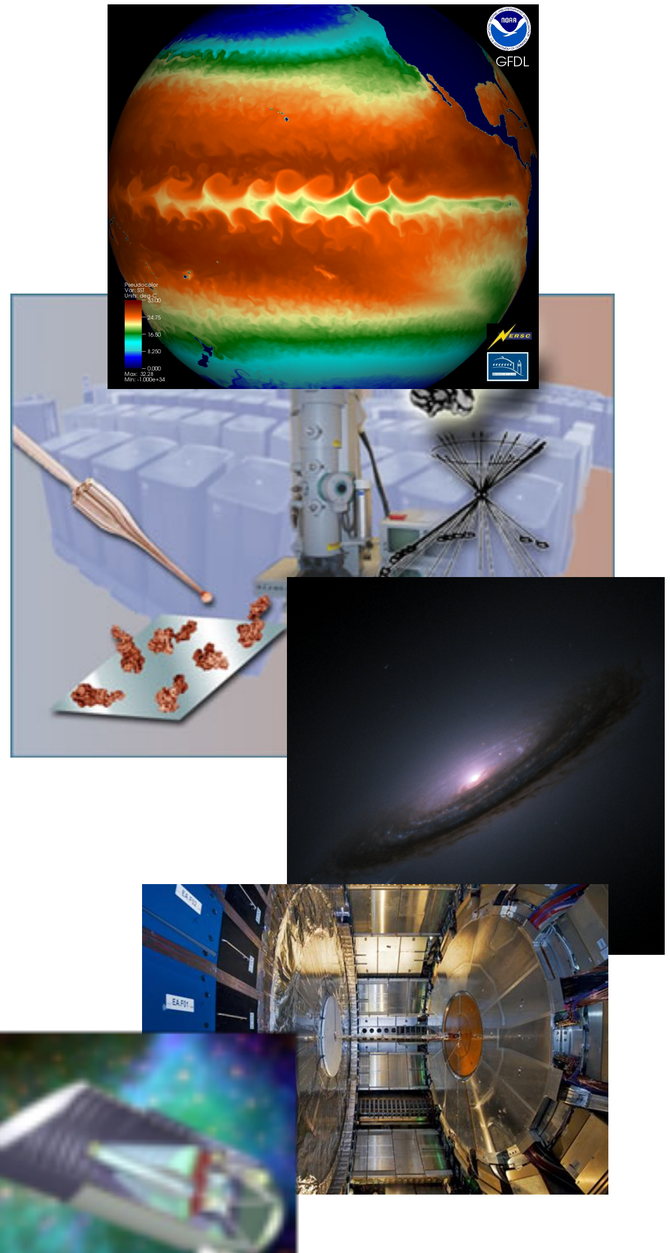
Blood Flow Simulation: MoBo

("Moving Boundaries")

Citation: A. Rahimian, I. Lashuk, A. Chandramowlishwaran, D. Malhotra, L. Moon, R. Sampath, A. Shringarpure, S. Veerapaneni, J. Vetter, R. Vuduc, D. Zorin, and G. Biros. "Petascale direct numerical simulation of blood flow on 200k cores and heterogeneous architectures." In *SC'10.* **Winner, Gordon Bell Prize**. http://dx.doi.org/10.1109/SC.2010.42

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



Source: J. Demmel @ UCB – http://www.cs.berkeley.edu/~demmel/cs267_Spr14

Why is parallel programming hard?

ieee spectrum
INSIDE TECHNOLOGY

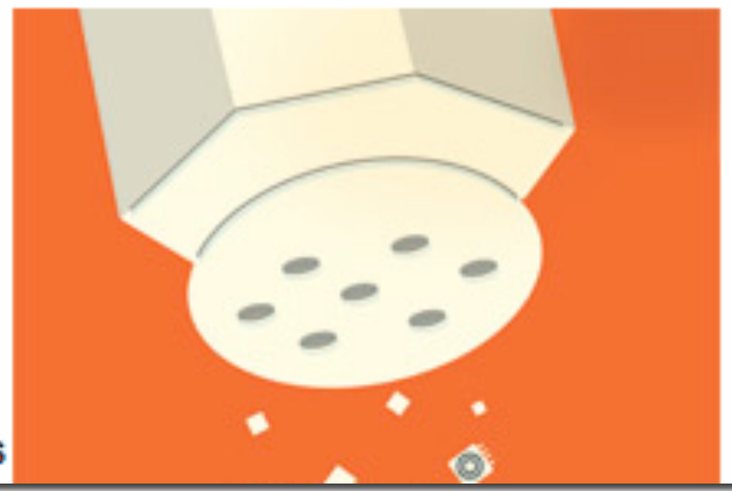
FEATURE

The Trouble With Multicore

Chipmakers are busy designing microprocessors that most programmers can't handle

By DAVID PATTERSON / JULY 2010

In 1975, future Hall of Famer Roger Staubach had the football but little else in a playoff game against the Minnesota Vikings. Behind by four points at midfield with 24 seconds to go, the Dallas Cowboys quarterback closed his eyes, threw the ball as hard as he could, and said a Hail Mary. (For you soccer fans, this



Course Logistics (EECS 221)

Class Website

http://newport.eecs.uci.edu/~amowli/teaching/eecs221_s17

EECS 221: Intro to High-Performance Computing

Prof. Aparna Chandramowlishwaran

Spring 2017 – T&Th 11-12:20pm in MSTB 120

Syllabus	Logistics	Piazza

Format and Grading

- Class forum on Piazza: https://piazza.com/uci/spring2017/eecs221
- Office hours: Thursday 1:30-3:30pm in EH 4410
- Grading
  - Three Homework's [30%]: No late assignments
  - Midterm [30%]: May 11th
  - Final project [40%]: Your choice
  - Participation : Online forum (Piazza), reading assignments, in-class participation
  - No final exam, instead **project presentations**

This is a tentative schedule subject to change.

WEEK	TUESDAY	THURSDAY
Apr 3-Apr 7	Parallel computing introduction; Parallel programing models	Quantifying concurrency: DAG, work-depth, Brent's theorem; First parallel algorithms: mergesort & merge
Apr 10-14	Amdahl's law; strong/weak scaling; parallel prefix-sums/scan	Shared memory: OpenMP & NUMA
Apr 17-21	Hands on lab 1: OpenMP	Distributed memory model: alpha-beta; 1-D distributed matrix multiply
Apr 24-28	2-D matrix multiply (SUMMA); topology	MPI & Collective communication
May 1-May 5	Hands-on lab 2: MPI	GPU architecture
May 8-12	Midterm review; HW solutions	Midterm
May 15-19	CUDA	GPU reductions and tuning
May 22-26	Hands-on lab 3: CUDA	Memory hierarchy/Caches; computational intensity
May 29-June 2	SIMD vectorization; Low-level tuning; ILP	Measuring performance and identifying bottlenecks: roofline model
Jun 5-9	Project Presentations	Project Presentations (contd)

EECS 221 Spring '17: Calibration Quiz

This quiz is mandatory but you will not be penalized for incorrect answers. It is mainly to help gauge your current skill level and for the instructor to get a sense of the overall class.

*Required

Section 1: About you

What is your name? Use: <last name>, <first name> (<nickname>) *

Your answer

What degree program? Use: <program>-<degree> *

Example: "ME-PhD", "EE-MS", "CE-BS"

Your answer

EECS 221 Spring '17: Calibration Quiz

This quiz is mandatory but you will not be penalized for incorrect answers. It is mainly to help gauge your current skill level and for the instructor to get a sense of the overall class.

*Required

Section 1: About you

What is your name? Use: <last name>, <first name> (<nickname>) *

Your answer

What degree program? Use: <program>-<degree> *
Example: "ME-PhD", "EE-MS", "CE-BS"

Your answer

Section 2: Programming language familiarity

Please specify your familiarity with the following languages in approximate lines of code written.

Assembly (any ISA) *

Your answer

C *

Your answer

C++ *

Your answer

Java *

Your answer

EECS 221 Spring '17: Calibration Quiz

This quiz is mandatory but you will not be penalized for incorrect answers. It is mainly to help gauge your current skill level and for the instructor to get a sense of the overall class.

*Required

Section 1: About you

What is your name? Use: <last name>, <first name> (<nickname>) *

Your answer

What degree program? Use
Example: "ME-PhD", "EE-MS", "CE-BS"

Your answer

Section 2: Programming language familiarity

Please specify your familiarity with the following languages in approximate lines of code written.

Assembly (any ISA) *

Your answer

Section 3: Quiz

Which of these big-Oh statements is true? *
"O(.)" is big-Oh notation; "^" means exponentiation, and "sqrt" means square-root.

- [ ] O(2^n) < O(n!)
- [ ] O(n * log(n)) < O(n^1.1)
- [ ] O(n^2) < O(2^n)
- [ ] O(log(n)) < O(sqrt(n))
- [ ] None of the above

In any programming language of your choice, write a function that computes the n-th Fibonacci number, for n >= 0. *
Use this definition: The n-th Fibonacci number, fib(n), is fib(n-1) + fib(n-2). Assume fib(0) = 0 and fib(1) = 1.

Your answer

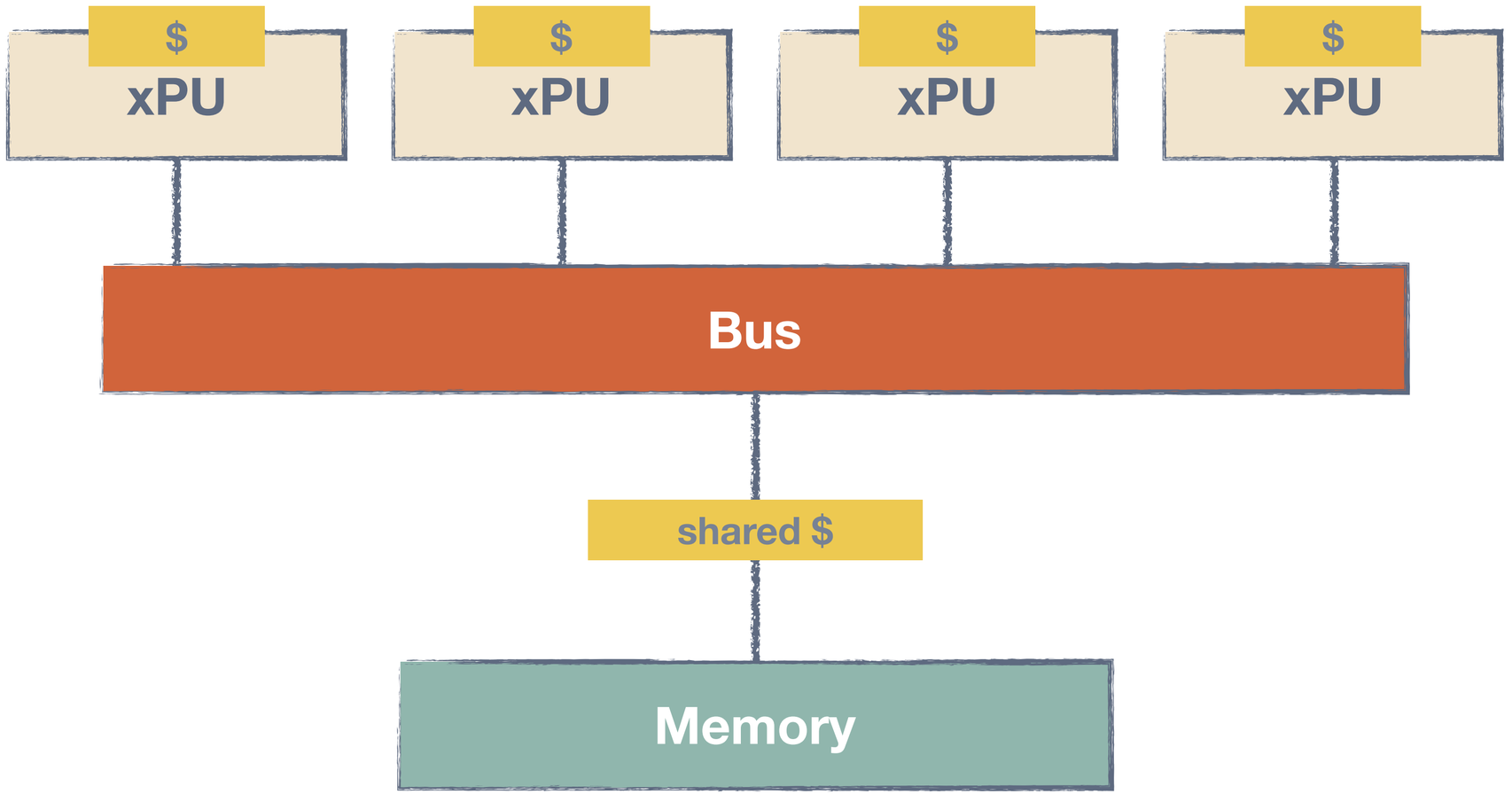
Introduction to parallel programming models

Variety of programming models

- Shared memory
- Shared address space
- Message passing
- Data parallel
- Clusters

Cost Factors

- [*Control*] How to control parallelism?
- [*Data*] Where is the data located (e.g, private, shared)?
- [*Synchronization*] How to synchronize between different parallel tasks happening simultaneously?
- How to compute the cost of the above? (Often cost more than arithmetic)

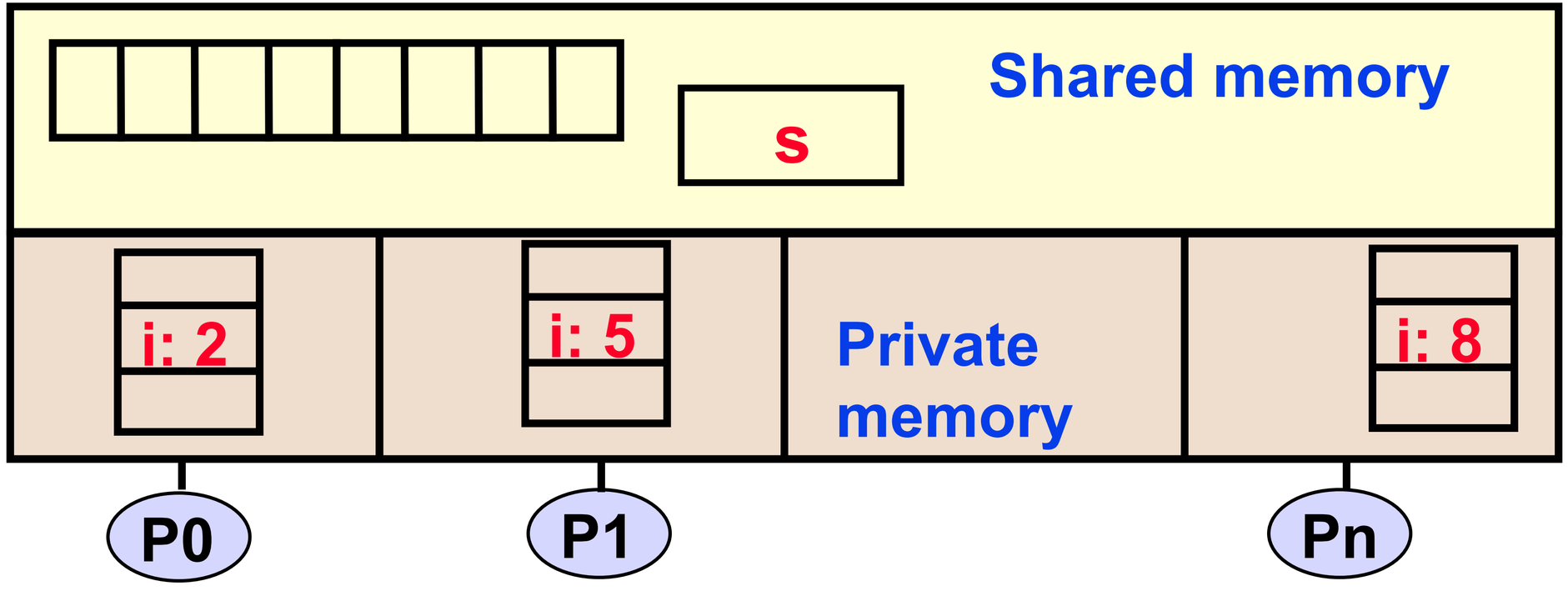


Shared Memory

All processors are connected to a shared memory.
Advantage: Uniform memory access (UMA)
Disadvantage: Difficult to scale to large number of processors

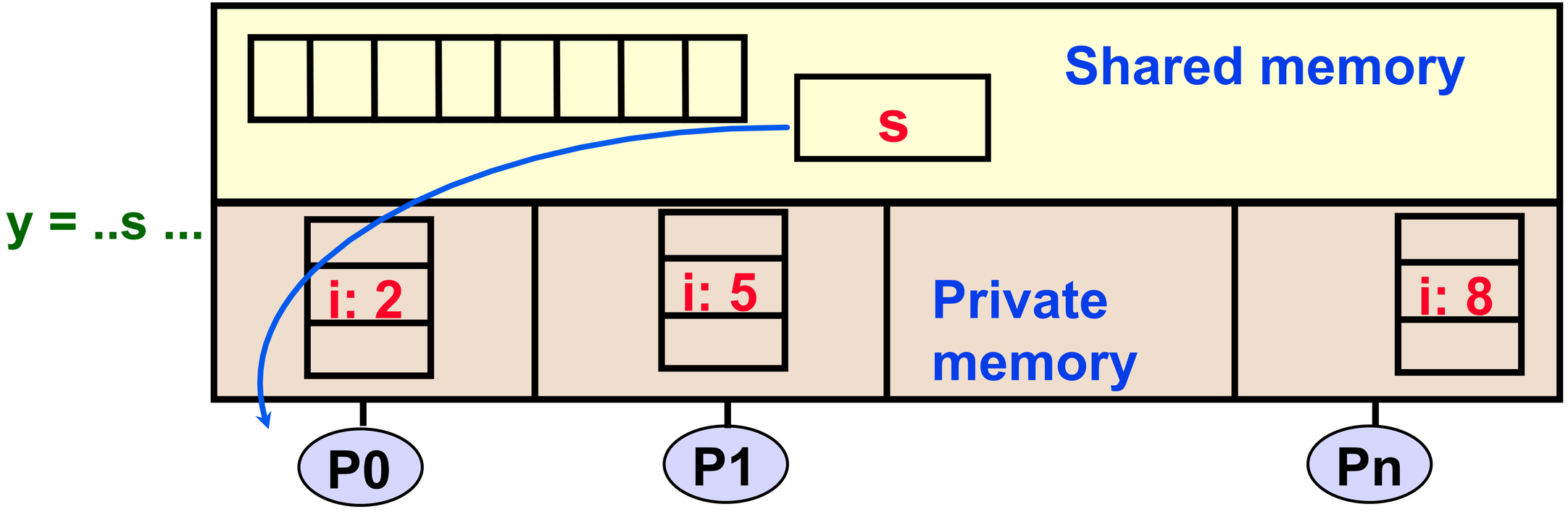
Shared Memory Model

- Program is a collection of threads of control
- Threads communicate implicitly by reading and writing shared variables
- Threads coordinate by synchronizing on shared variables



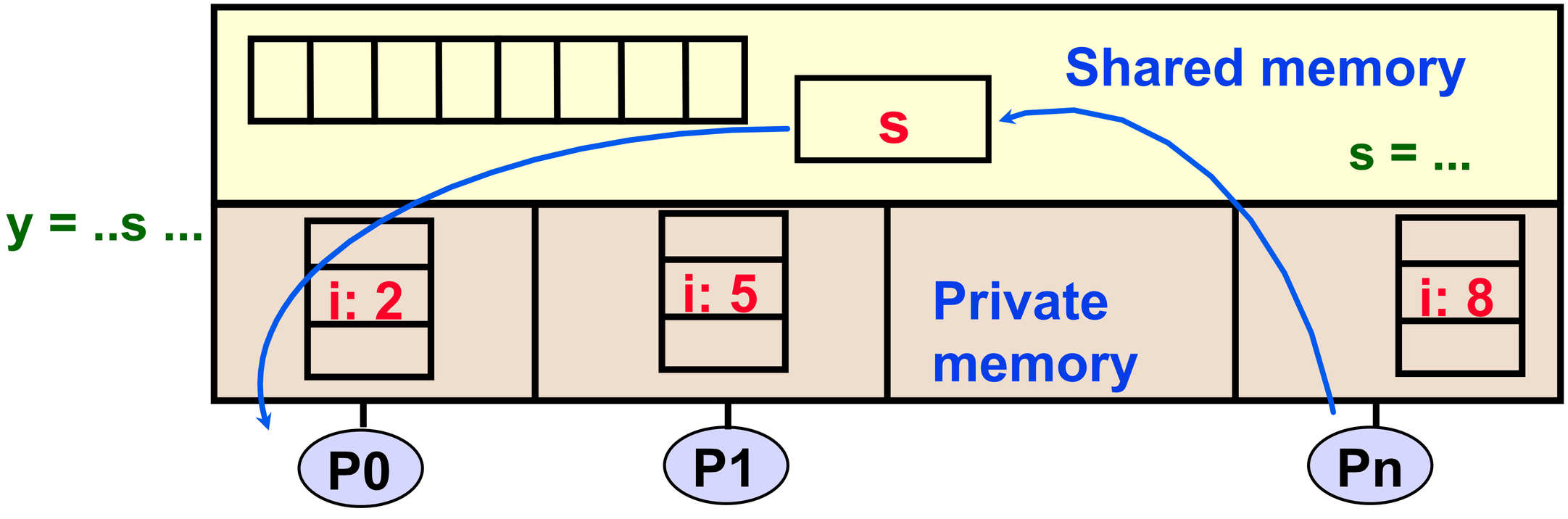
Shared Memory Model

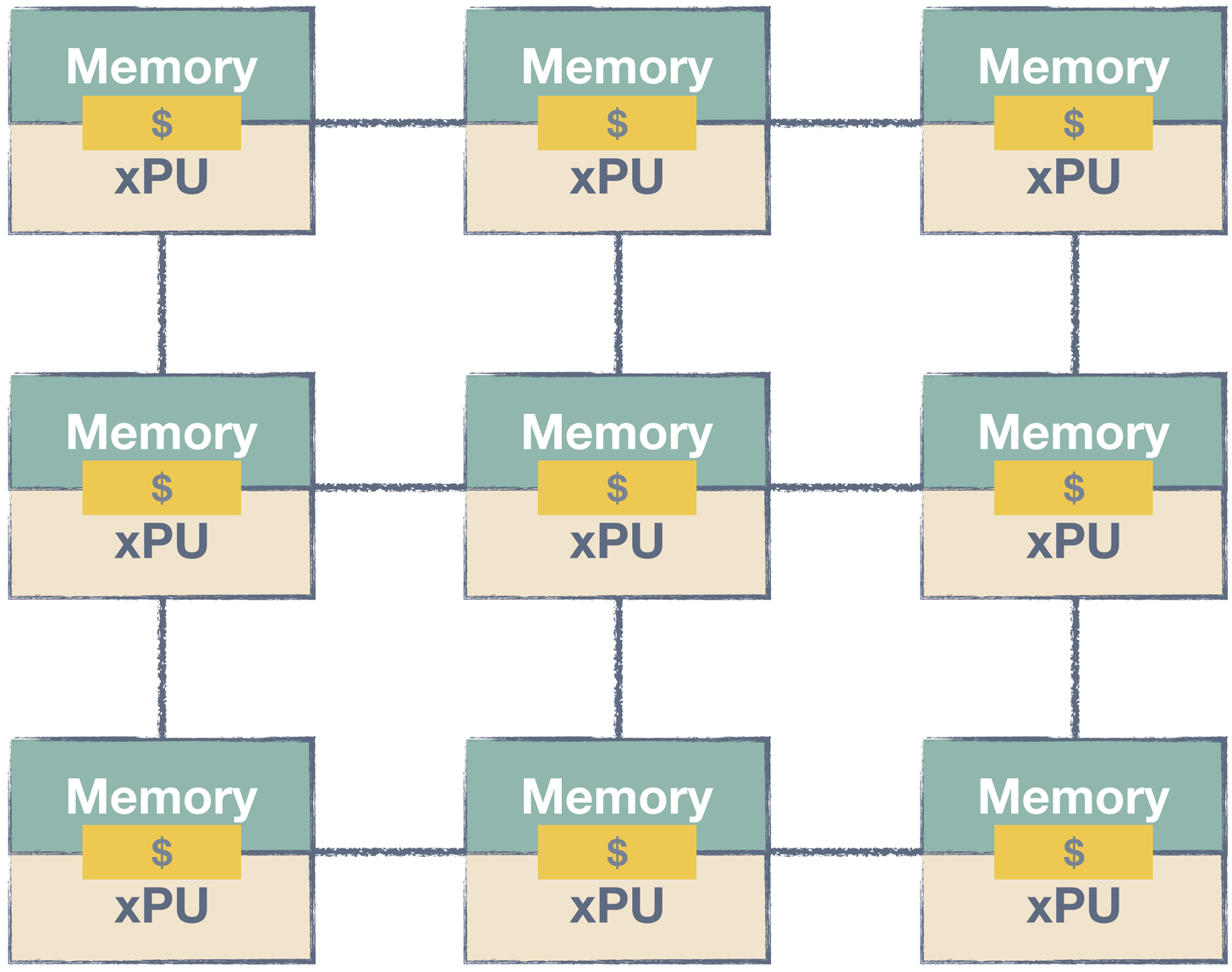
- Program is a collection of threads of control
- Threads communicate implicitly by reading and writing shared variables
- Threads coordinate by synchronizing on shared variables



Shared Memory Model

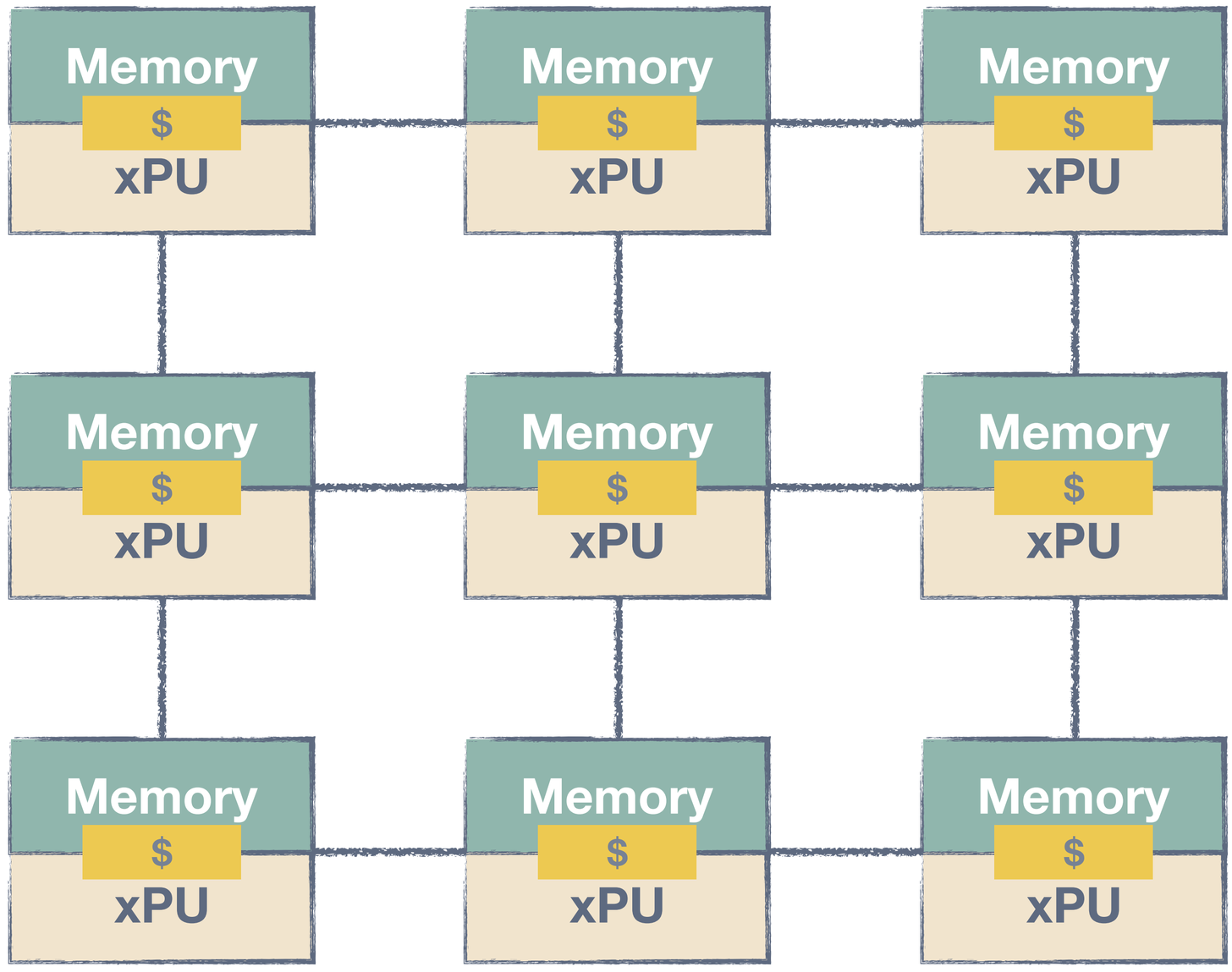
- Program is a collection of threads of control
- Threads communicate implicitly by reading and writing shared variables
- Threads coordinate by synchronizing on shared variables



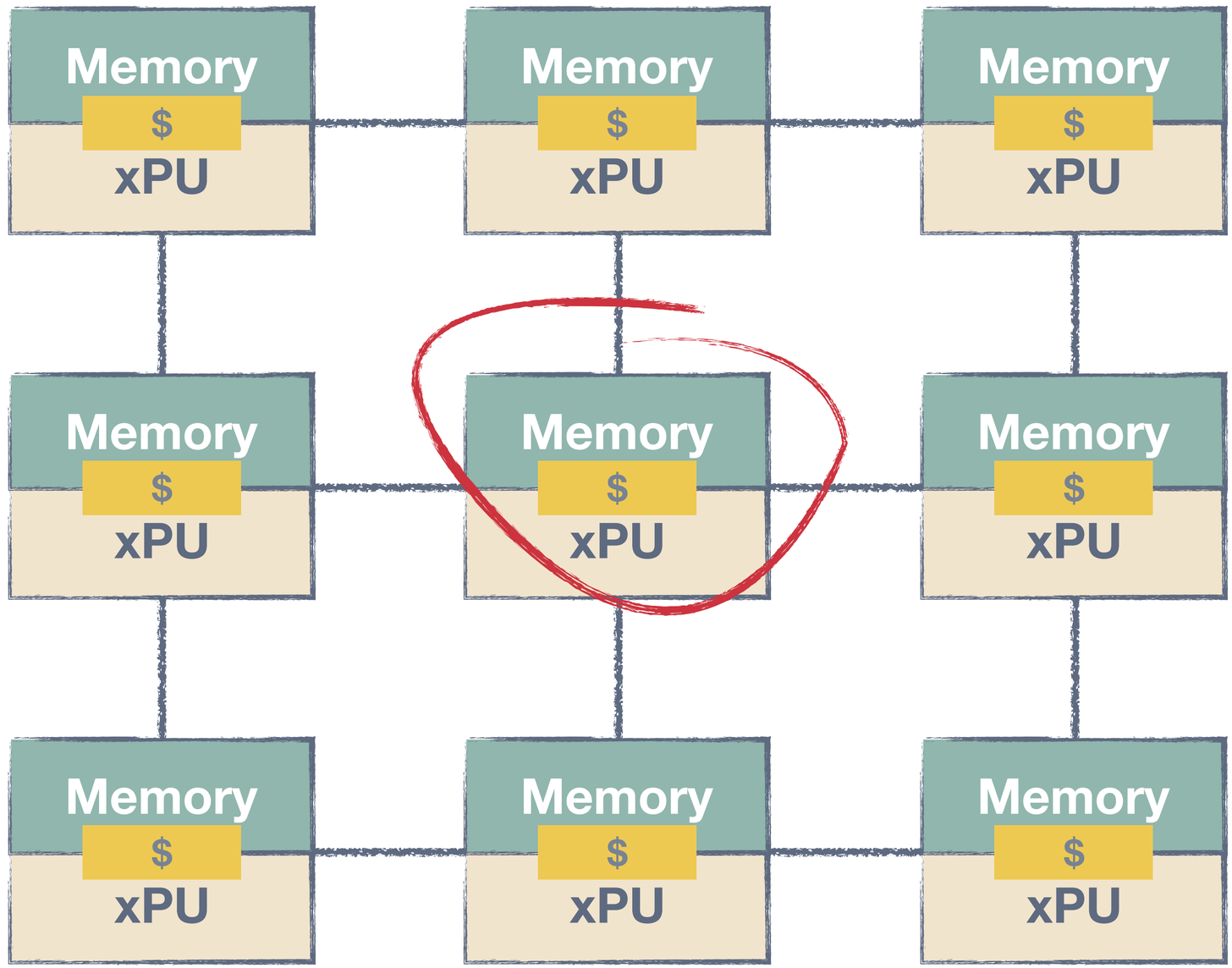


Distributed Memory

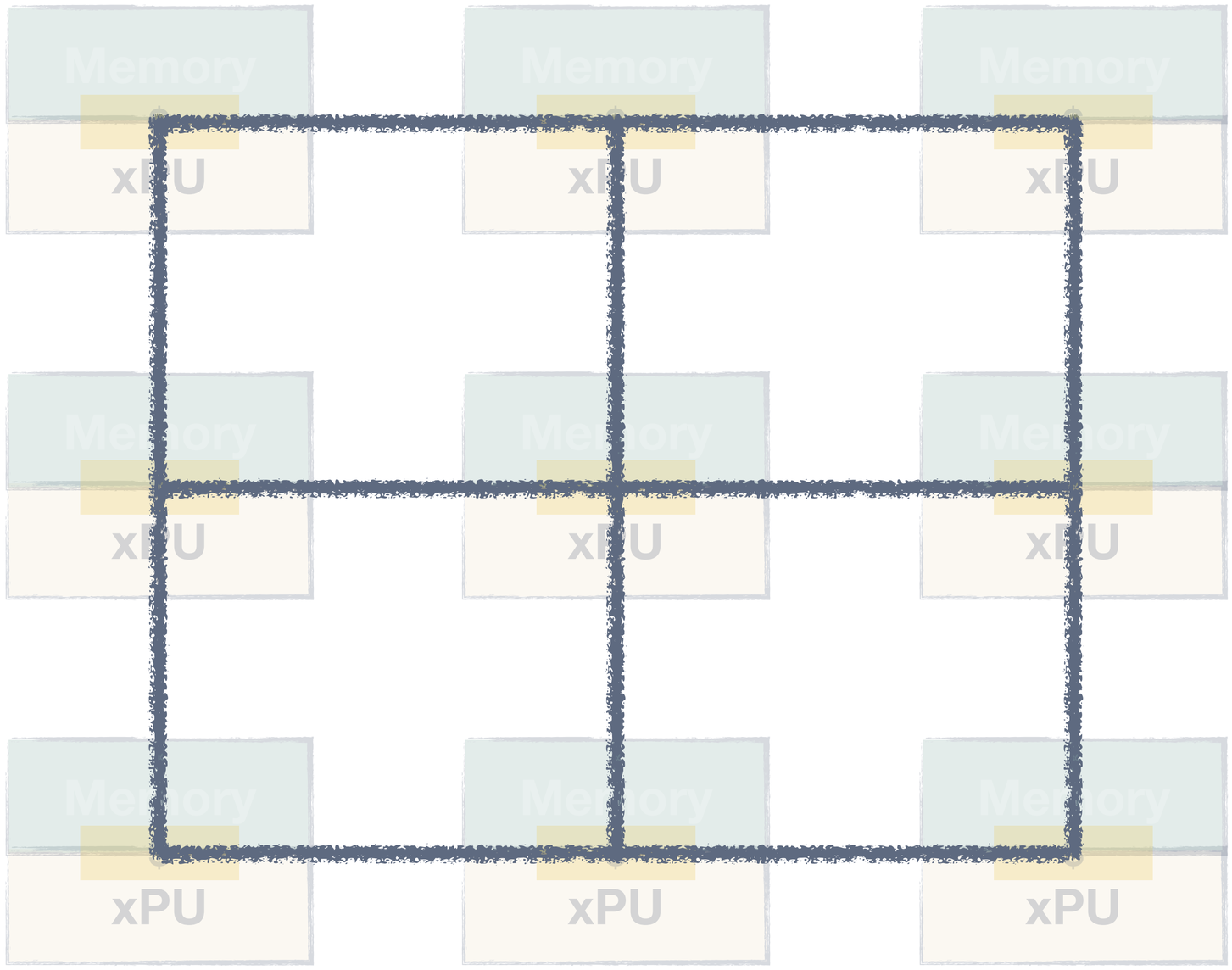
Each processor has its own memory and cache but cannot directly access another processor's memory.



Two costs: $T_{\text{network}} + T_{\text{memory}}$



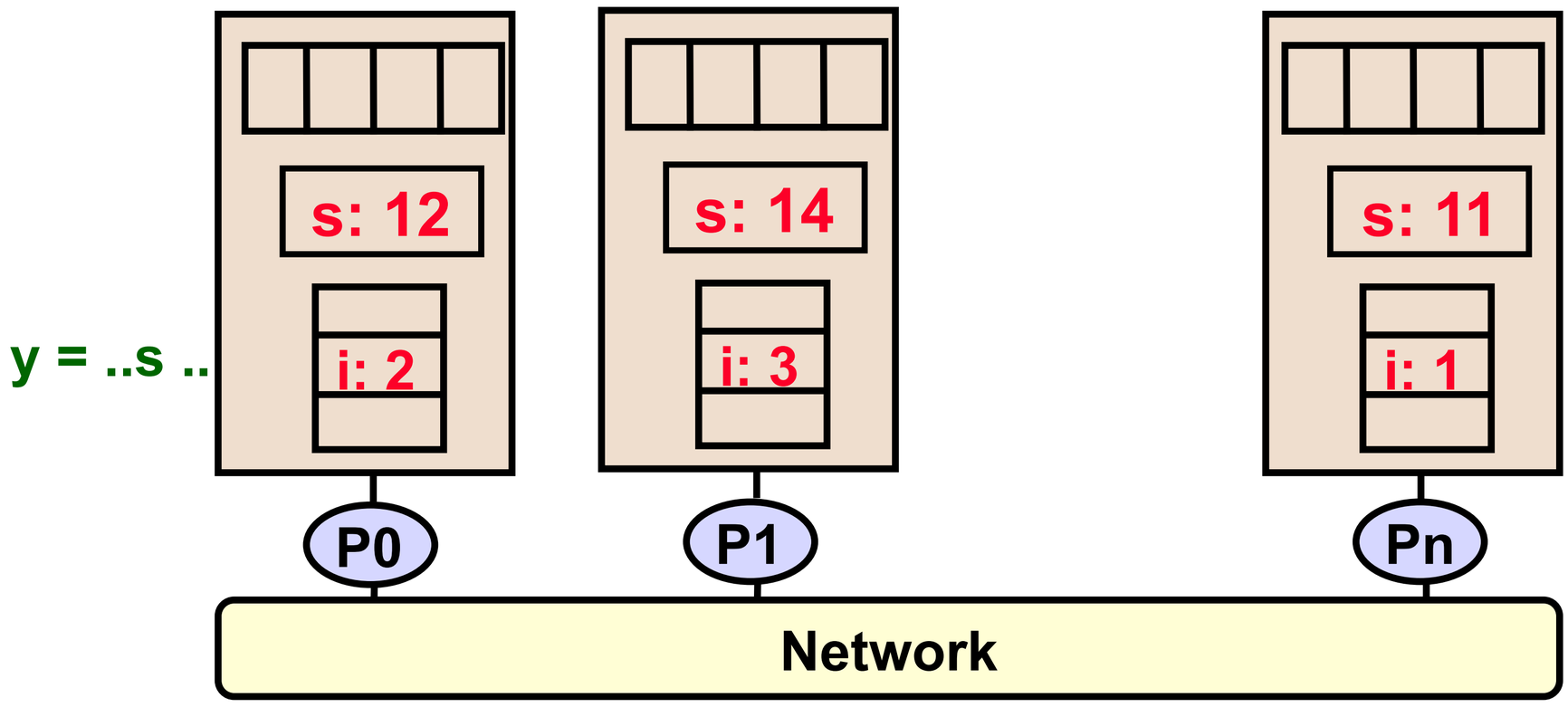
Two costs: $T_{network}$ + T_{memory}



Two costs: $T_{network}$ + T_{memory}

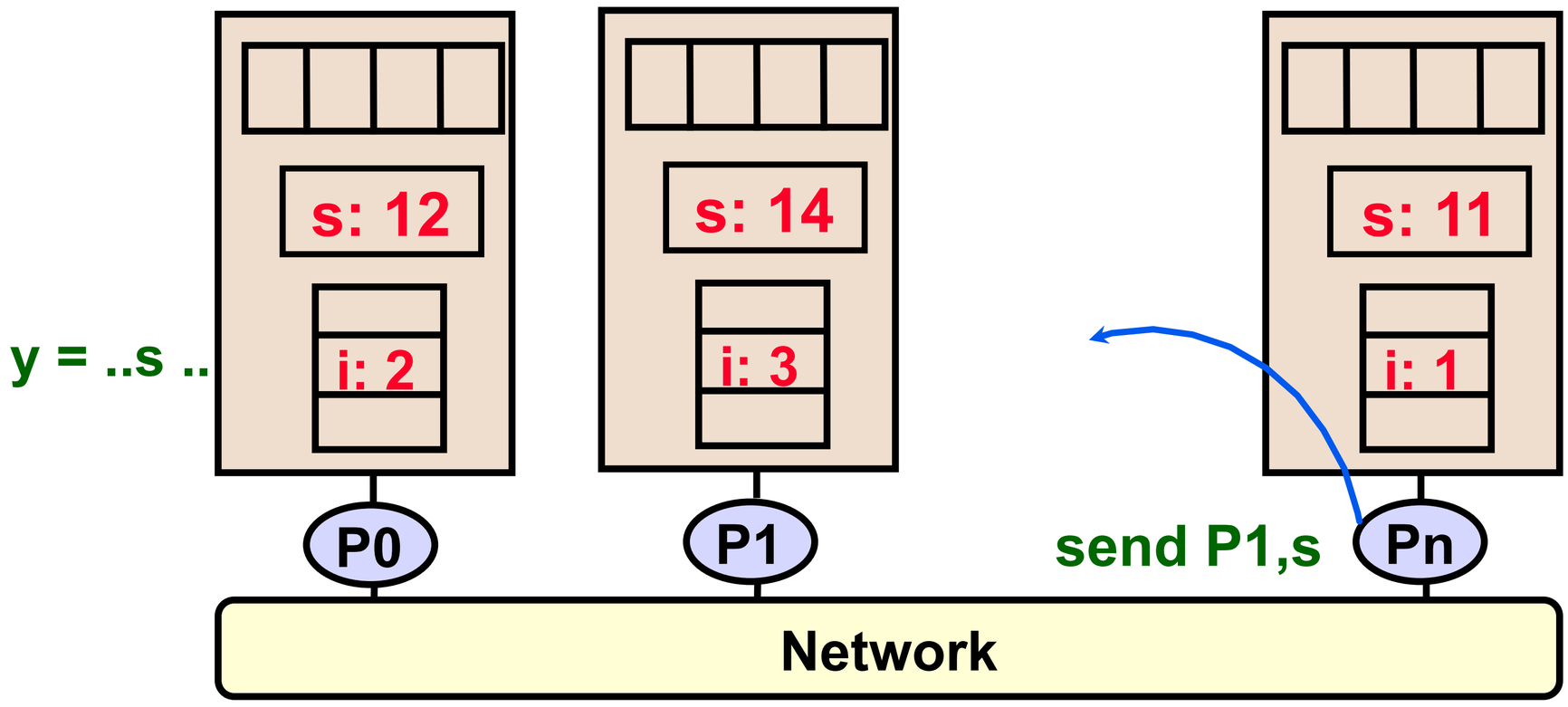
Message Passing Model

- Program consists of a collection of named processes
- Processes communicate by explicit send/receive messages
- Coordination is implicit in every communication
- MPI has become the de facto standard



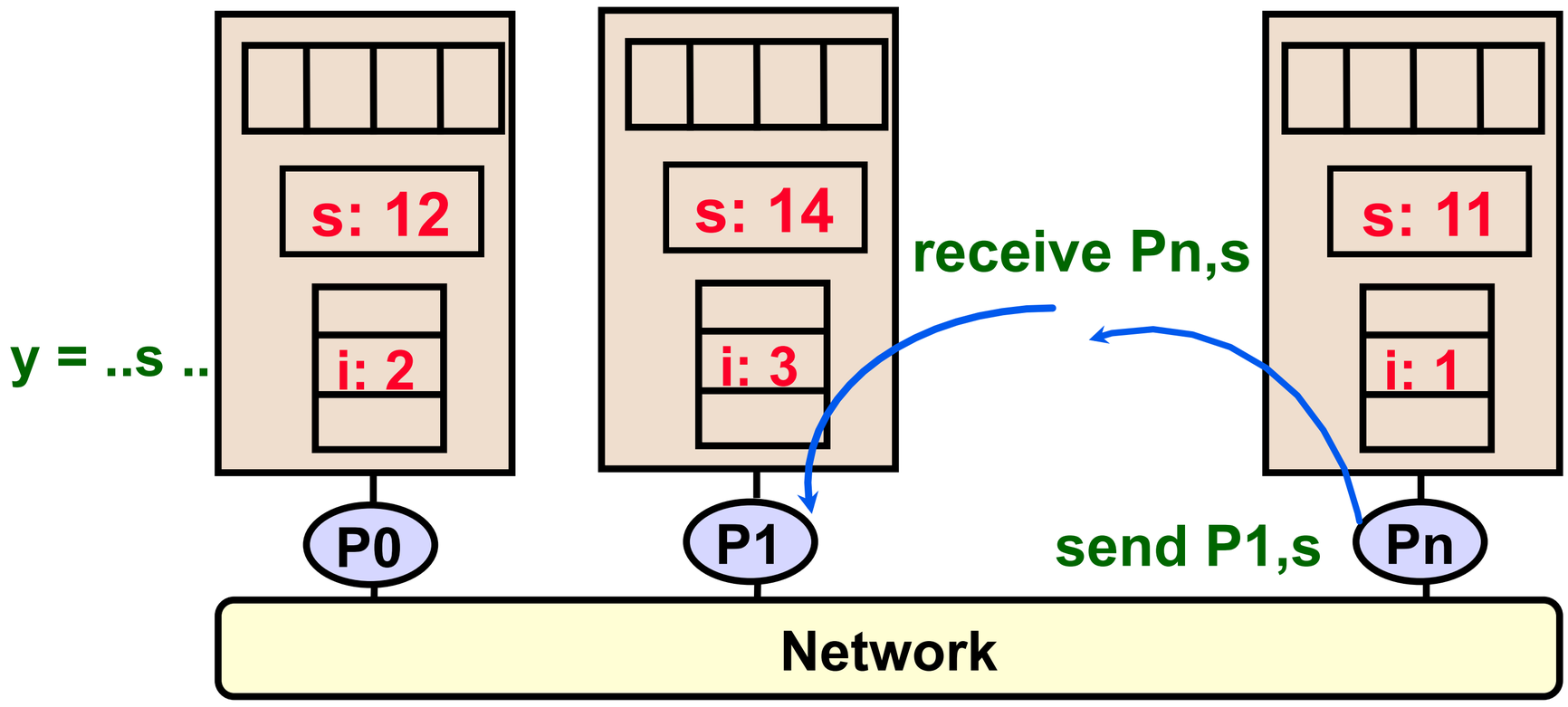
Message Passing Model

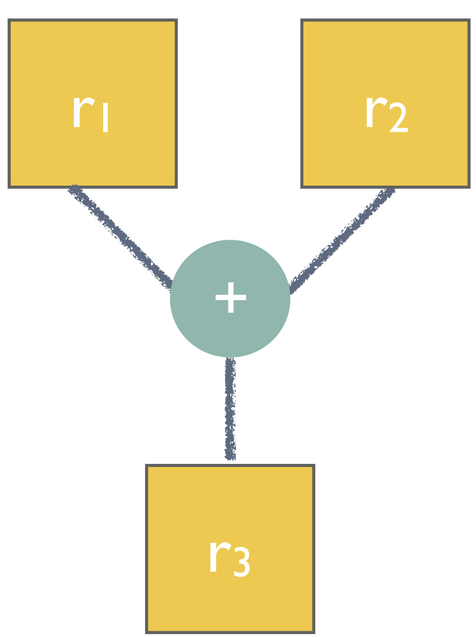
- Program consists of a collection of named processes
- Processes communicate by explicit send/receive messages
- Coordination is implicit in every communication
- MPI has become the de facto standard

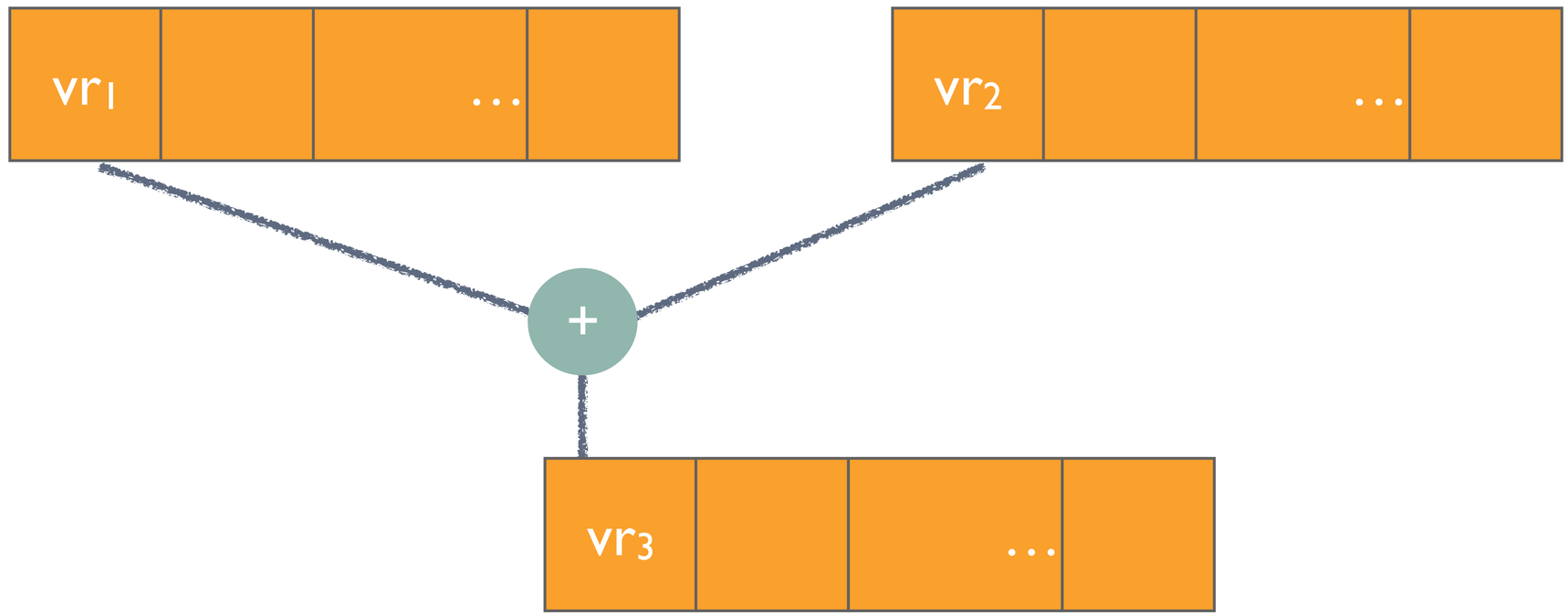


Message Passing Model

- Program consists of a collection of named processes
- Processes communicate by explicit send/receive messages
- Coordination is implicit in every communication
- MPI has become the de facto standard

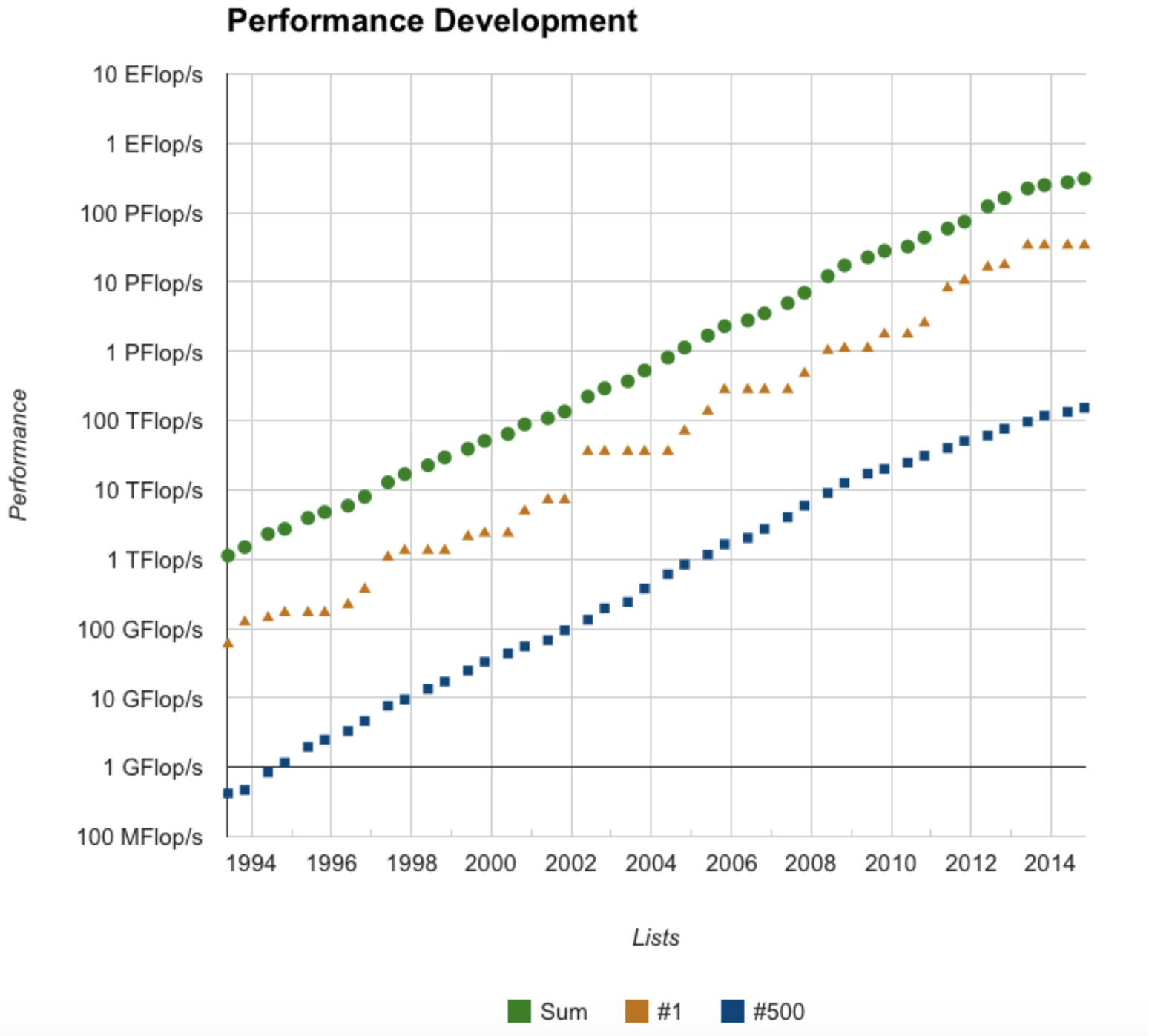




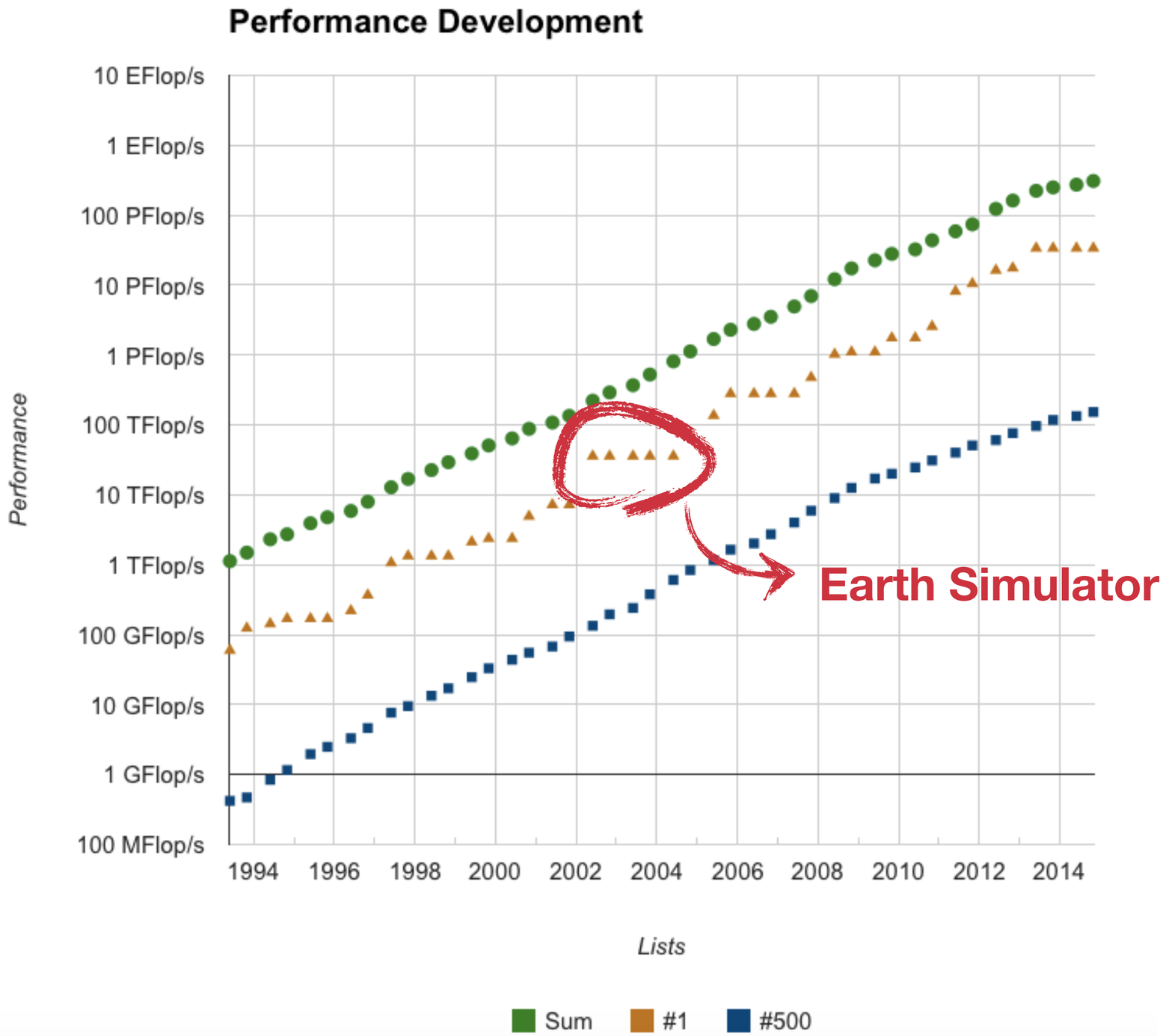


VECTOR MACHINES

The hardware performs a full vector operation in number of elements per vector register / number of lanes



http://www.top500.org/statistics/perfdevel/



http://www.top500.org/statistics/perfdevel/

Data Parallel Model

- Single thread of control consisting of parallel operations.
- Communication is implicit in parallel operators
- Coordination is implicit – statements executed synchronously

- **Drawbacks**
  - Not all problems fit this model
  - Difficult to map onto coarse-grained machines

Summary

- Three basic conceptual models
  - Shared memory
  - Distributed memory
  - Data parallel

  and hybrids of these machines
- Key to performance
  - Mostly independent (little synchronization)
  - About the same granularity (load balanced)
  - Good locality (little communication)