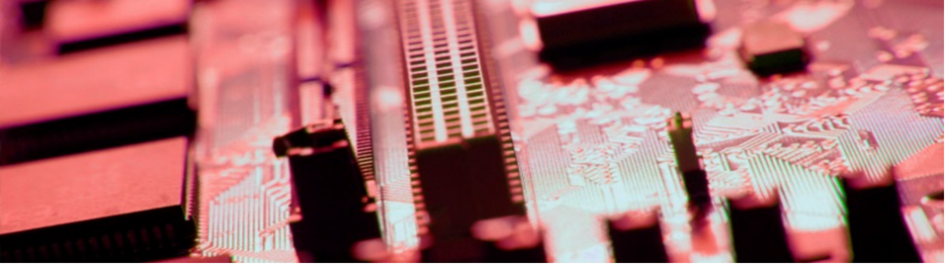


LCI workshop
on high performance clustered computing

Linux Clusters Institute: Introduction to High Performance Computing

Henry Neeman, University of Oklahoma

Assistant Vice President, Information Technology - Research Strategy Advisor

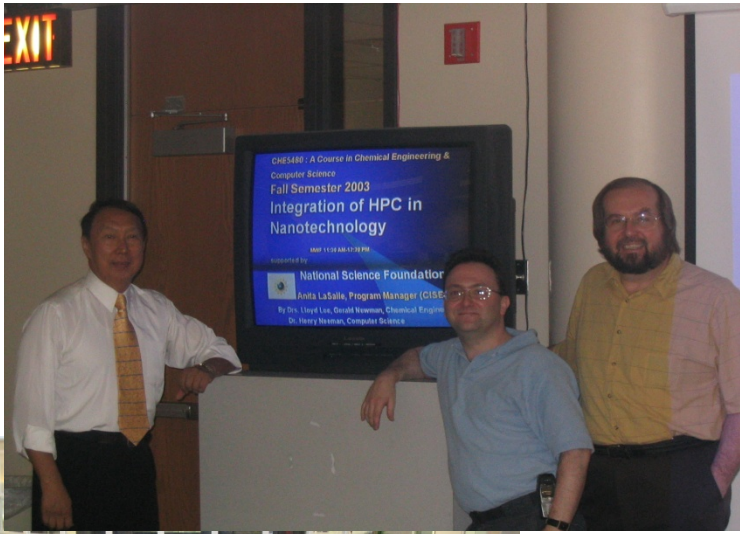
Director, OU Supercomputing Center for Education & Research (OSCER)

Associate Professor, College of Engineering

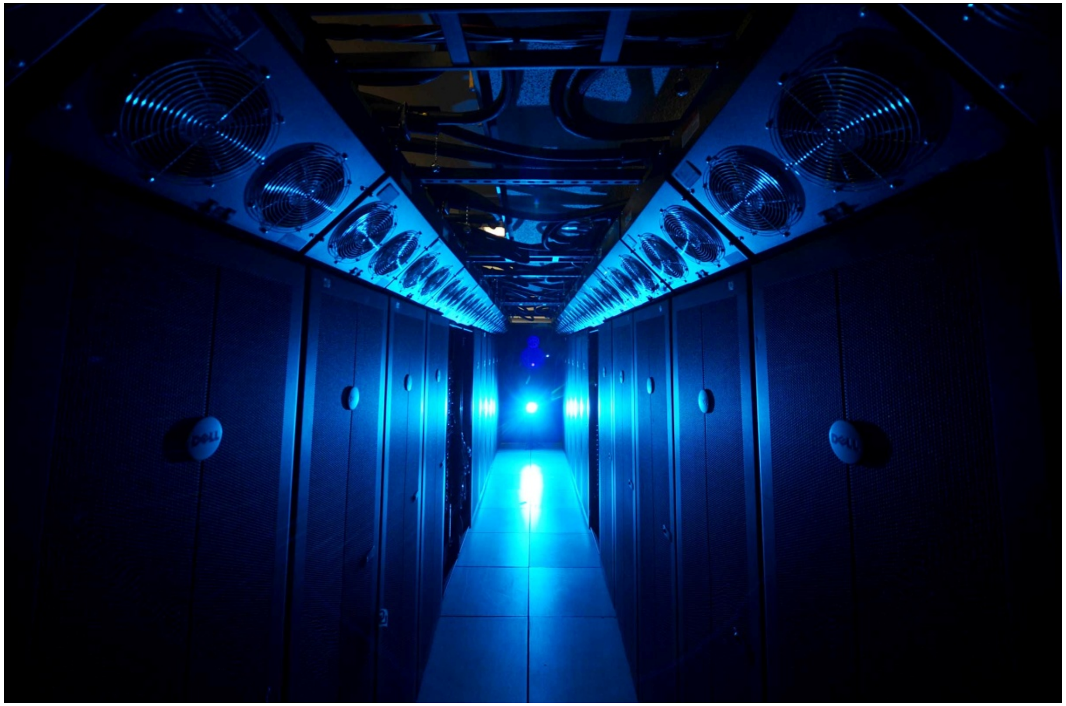
Adjunct Faculty, School of Computer Science

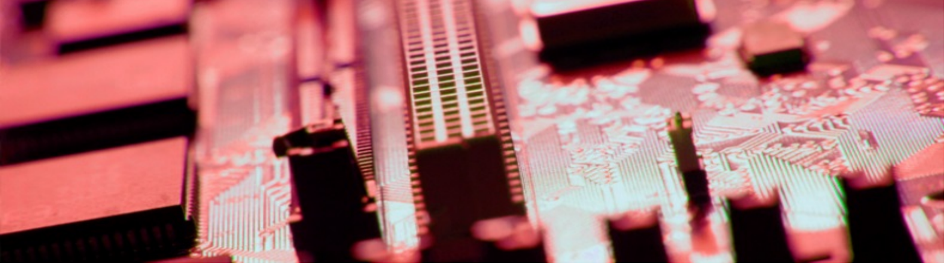
LCI workshop
on high performance clustered computing

People



Things





Thanks for your attention!

Questions?

www.oscer.ou.edu

What is Supercomputing?

Supercomputing is the **biggest, fastest computing right this minute**.

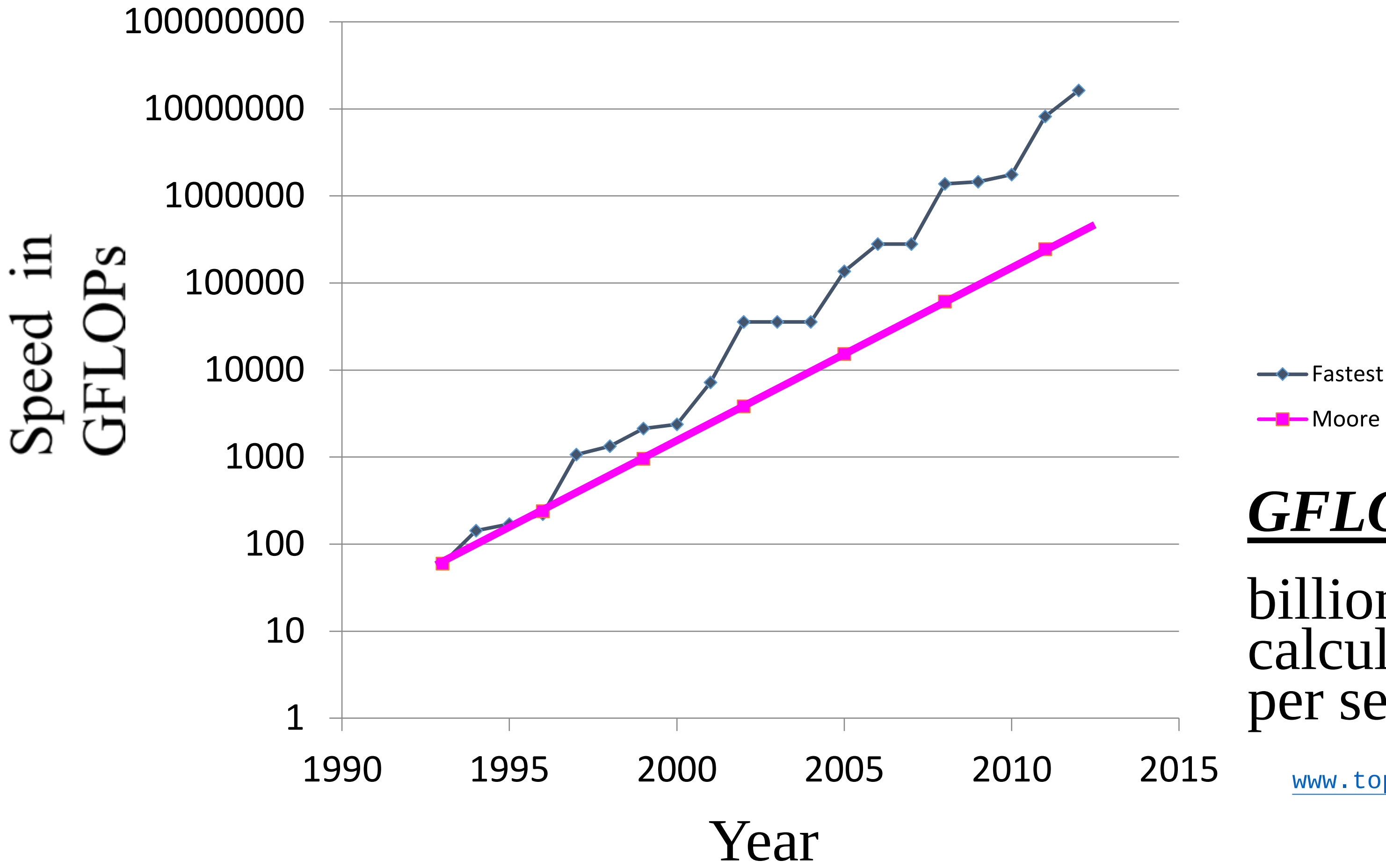
Likewise, a ***supercomputer*** is one of the biggest, fastest computers right this minute.

So, the definition of supercomputing is **constantly changing**.

Rule of Thumb: A supercomputer is typically at least 100 times as powerful as a PC.

Jargon: Supercomputing is also known as
High Performance Computing **(HPC)** or
High End Computing **(HEC)** or
Cyberinfrastructure **(CI).**

Fastest Supercomputer vs. Moore



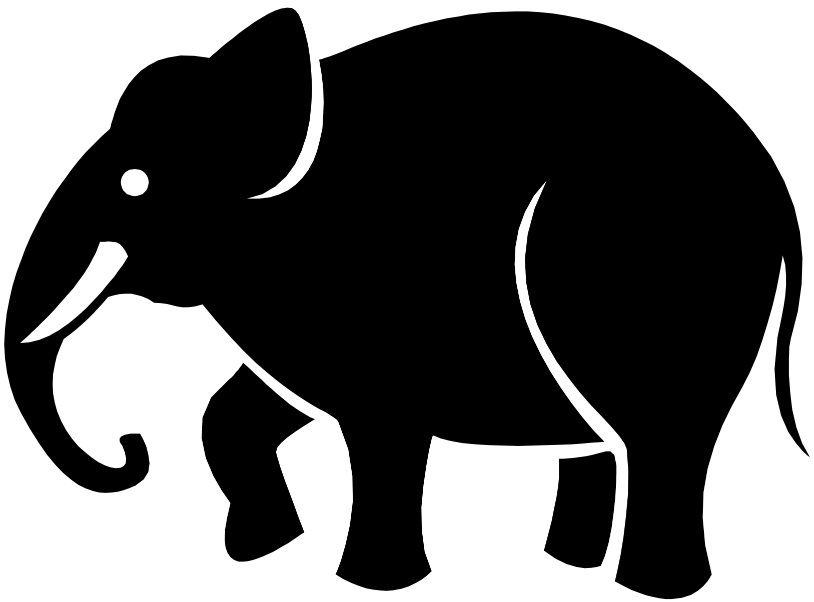
GFLOPs:

billions of calculations per second

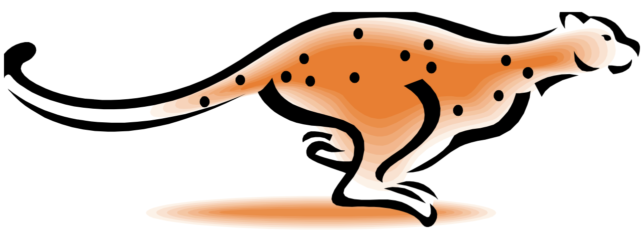
www.top500.org

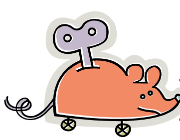
What is Supercomputing About?

Size



Speed

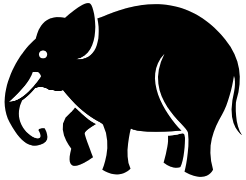




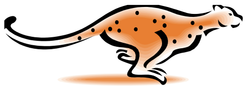
Laptop

What is Supercomputing About?

- **Size**: Many problems that are interesting to scientists and engineers **can't fit on a PC** – usually because they need more than a few GB of RAM, or more than a few 100 GB of disk.



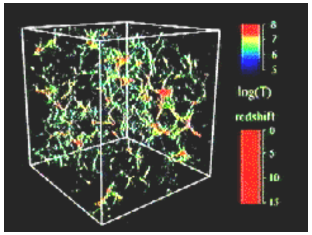
- **Speed**: Many problems that are interesting to scientists and engineers would take a very very long time to run on a PC: months or even years. But a problem that would take **a month on a PC** might take only **an hour on a supercomputer**.

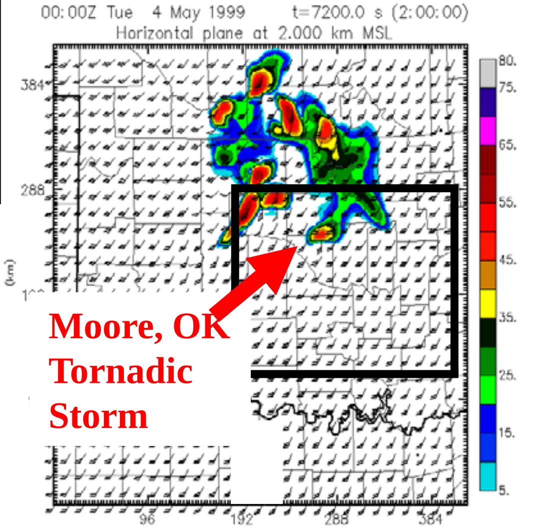


What Is HPC Used For?

- ***Simulation*** of physical phenomena, such as
  - Weather forecasting
  - Galaxy formation
  - Oil reservoir management
- ***Data mining***: finding **needles** in a **haystack** of data, such as
  - Gene sequencing
  - Signal processing
  - Detecting storms that might produce
- ***Visualization***: turning a vast sea of **data** into **pictures** that a scientist can understand

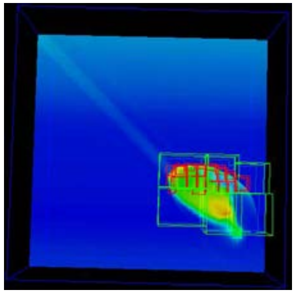
[1]





May 3 1999[2]

[3]



Supercomputing Issues

- The tyranny of the ***storage hierarchy***
- ***Parallelism***: doing multiple things at the same time

What is a Cluster Supercomputer?

"… [W]hat a ship is … It's not just a keel and hull and a deck and sails. That's what a ship needs. But what a ship is ... is freedom."

– Captain Jack Sparrow

"Pirates of the Caribbean"

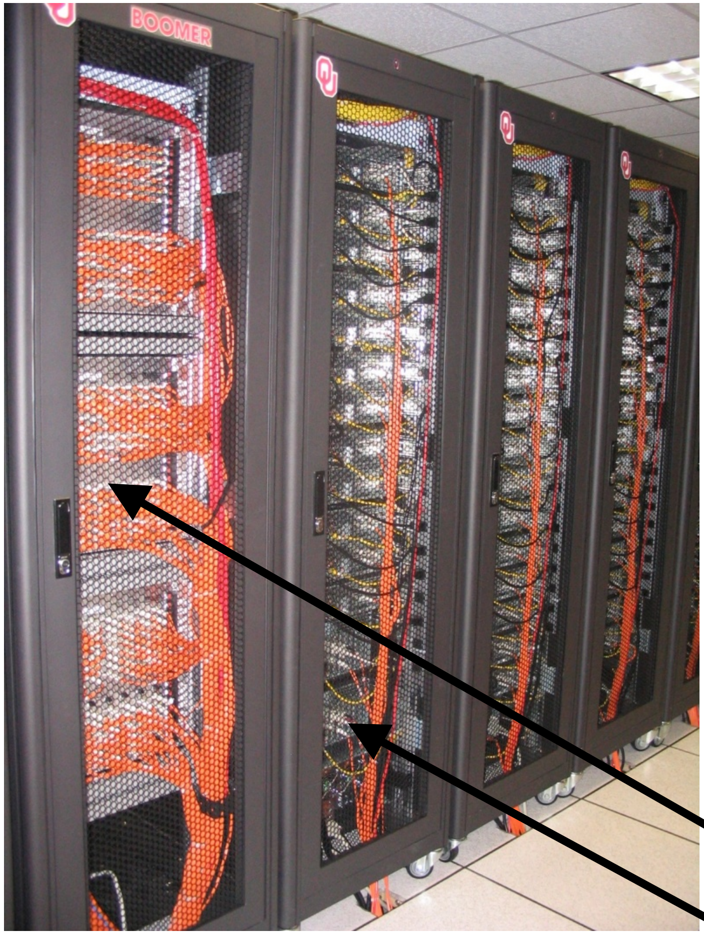
What a Cluster is ....

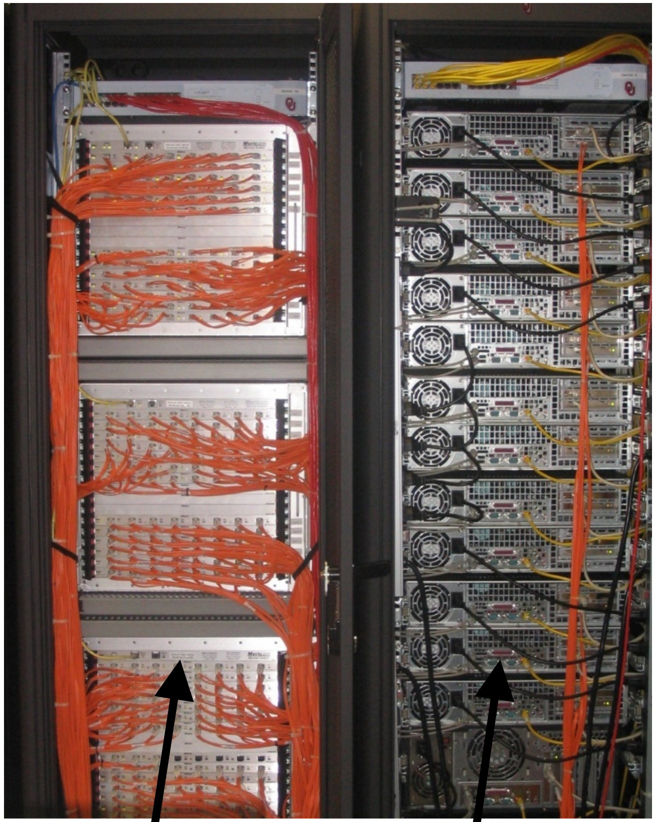
A cluster **needs** of a collection of small computers, called ***nodes***, hooked together by an ***interconnection network*** (or ***interconnect*** for short).

It also **needs** software that allows the nodes to communicate over the interconnect.

But what a cluster **is** … is all of these components working together as if they're one big computer ... a **super** computer.

An Actual Cluster

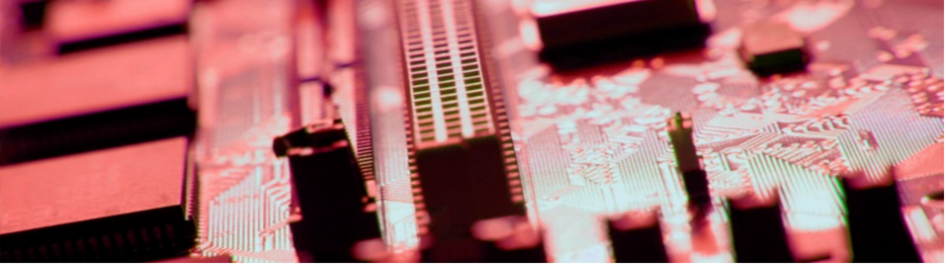




Interconnect

Nodes

Boomer (in service 2002-5)



A Quick Primer on Hardware

Henry's Laptop

Lenovo B570[4]

- Intel Pentium B940
  2.0 GHz w/2 MB L2 Cache

  GB 1333 MHz DDR3 SDRAM

  500 GB SATA 5400 RPM Hard Drive

  DVD±RW/CD-RW Drive

  1 Gbps Ethernet Adapter

Typical Computer Hardware

- Central Processing Unit
- Primary storage
- Secondary storage
- Input devices
- Output devices

Central Processing Unit

Also called ***CPU*** or ***processor***: the “brain”

Components

- ***Control Unit***: figures out what to do next – for example, whether to load data from memory, or to add two values together, or to store data into memory, or to decide which of two possible actions to perform (***branching***)
- ***Arithmetic/Logic Unit***: performs calculations – for example, adding, multiplying, checking whether two values are equal
- ***Registers***: where data reside that are **being used right now**

Primary Storage

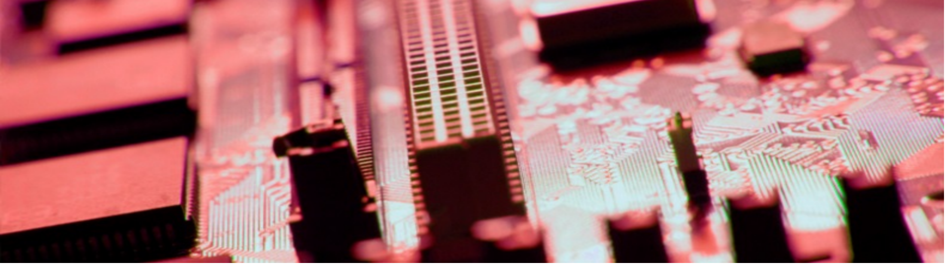
- ***Main Memory***
  - Also called ***RAM*** ("Random Access Memory")
  - Where data reside when they're **being used by a program that's currently running**
- ***Cache***
  - Small area of much faster memory
  - Where data reside when they're **about to be used** and/or **have been used recently**
- Primary storage is ***volatile***: values in primary storage disappear when the power is turned off.

Secondary Storage

- Where data and programs reside that are going to be used **in the future**
- Secondary storage is ***non-volatile***: values **don't** disappear when power is turned off.
- Examples: hard disk, CD, DVD, Blu-ray, magnetic tape, floppy disk
- Many are ***portable***: can pop out the CD/DVD/tape/floppy and take it with you

Input/Output

- Input devices – for example, keyboard, mouse, touchpad, joystick, scanner
- Output devices – for example, monitor, printer, speakers



The Tyranny of the Storage Hierarchy

The Storage Hierarchy

Fast, expensive, few

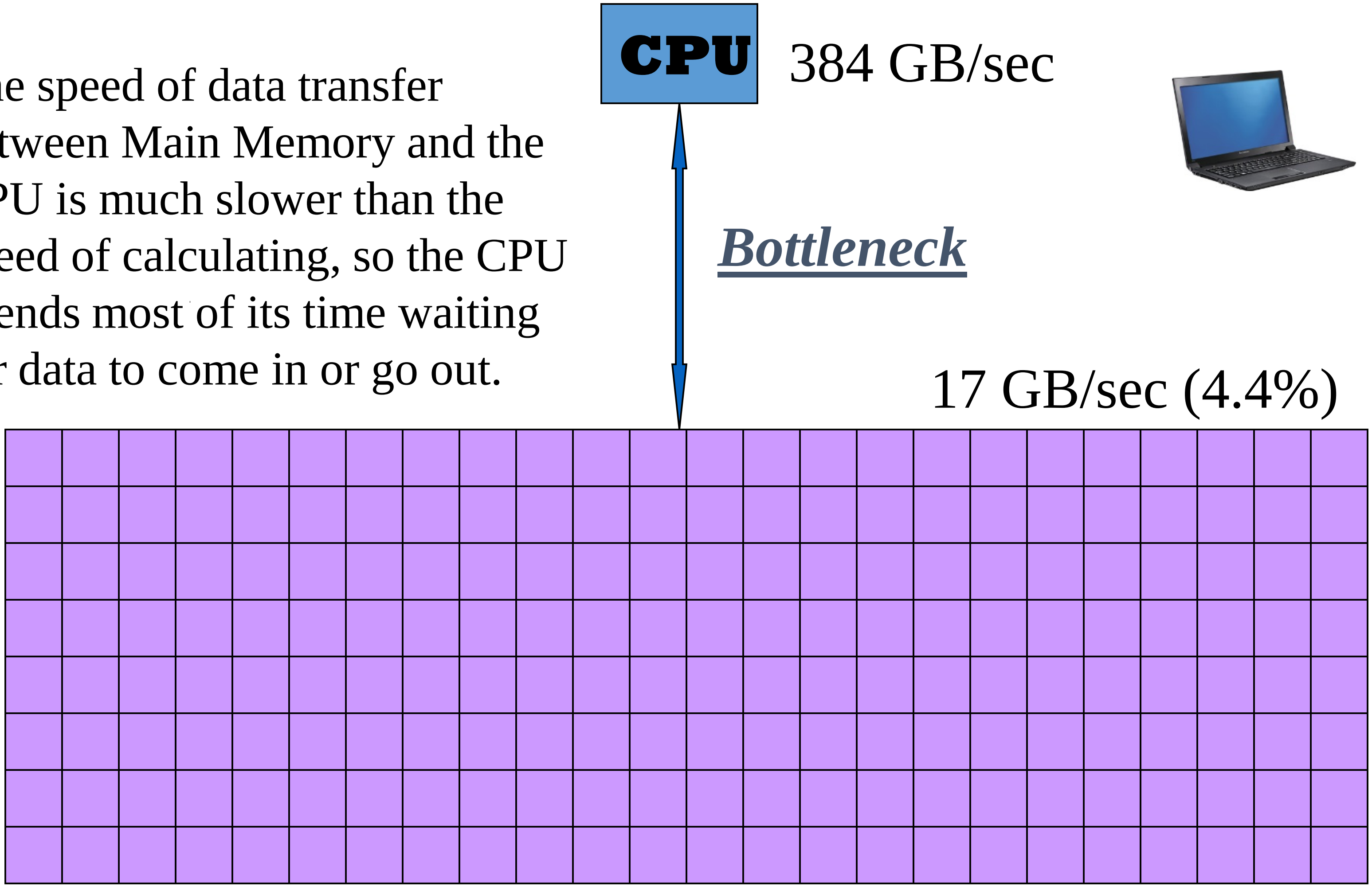
Slow, cheap, a lot

[5]

- Registers
- Cache memory
- Main memory (RAM)
- Hard disk
- Removable media (CD, DVD etc)
- Internet

RAM is Slow

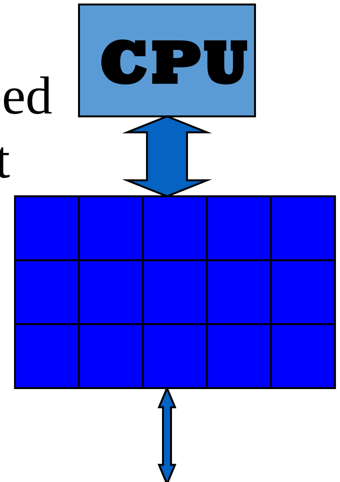
The speed of data transfer between Main Memory and the CPU is much slower than the speed of calculating, so the CPU spends most of its time waiting for data to come in or go out.

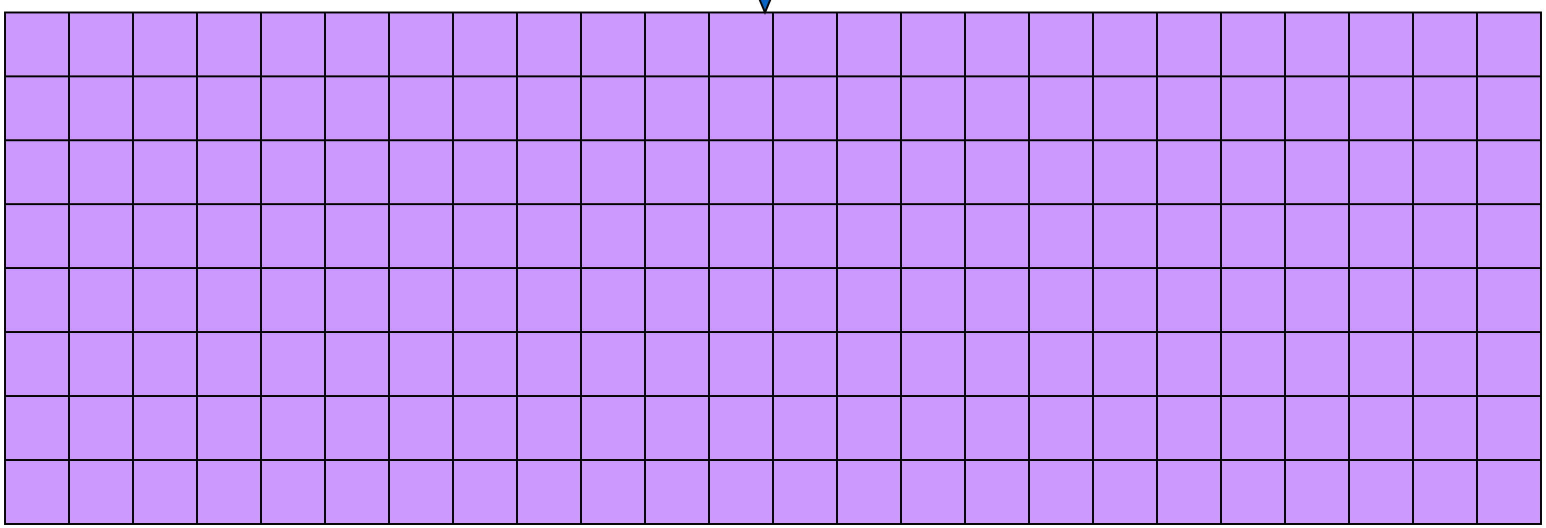


LCI workshop
on high performance clustered computing

Why Have Cache?

Cache is much closer to the speed of the CPU, so the CPU doesn't have to wait nearly as long for stuff that's already in cache: it can do more operations per second!





Henry’s Laptop

Lenovo B570[4]

- Intel Pentium B940
  2.0 GHz w/2 MB L2 Cache

  GB 1333 MHz DDR3 SDRAM

  500 GB SATA 5400 RPM Hard Drive

  DVD+RW/CD-RW Drive

  1 Gbps Ethernet Adapter

Storage Speed, Size, Cost

Henry's Laptop	Registers (Intel Core2 Duo 1.6 GHz)	Cache Memory (L2)	Main Memory (1333MHz DDR3 SDRAM)	Hard Drive	Ethernet (1000 Mbps)	DVD+R (16x)	Phone Modem (56 Kbps)
Speed (MB/sec) [peak]	393,216[6] (16 GFLOP/s*)	30,720	17,400 [7]	25[9]	125	22 [10]	0.007
Size (MB)	464 bytes** [11]	3	4096	500,000	unlimited	unlimited	unlimited
Cost ($/MB)	–	$32 [12]	$0.007 [12]	$0.00003 [12]	charged per month (typically)	$0.00005 [12]	charged per month (typically)

* GFLOP/s: billions of floating point operations per second

** 16 64-bit general purpose registers, 8 80-bit floating point registers, 16 128-bit floating point vector registers

Why the Storage Hierarchy?

Why does the Storage Hierarchy always work? Why are faster forms of storage more expensive and slower forms cheaper?

Proof by contradiction:

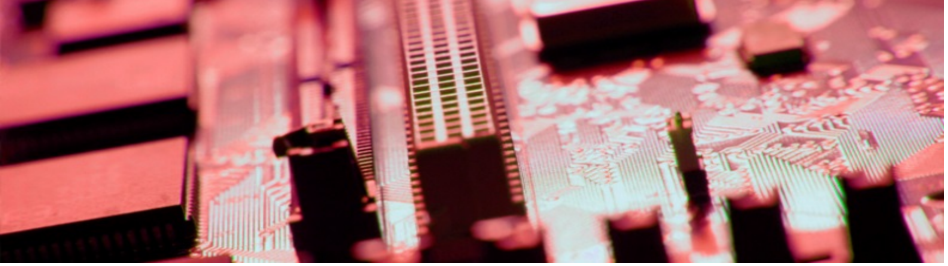
Suppose there were a storage technology that was **slow** and **expensive**.

How much of it would you buy?

Comparison

- Zip: Cartridge $7.25 (2.9 cents per MB), speed 2.4 MB/sec
- Blu-Ray: Disk $4 ($0.00005 per MB), speed 19 MB/sec

Not surprisingly, no one buys Zip drives any more.



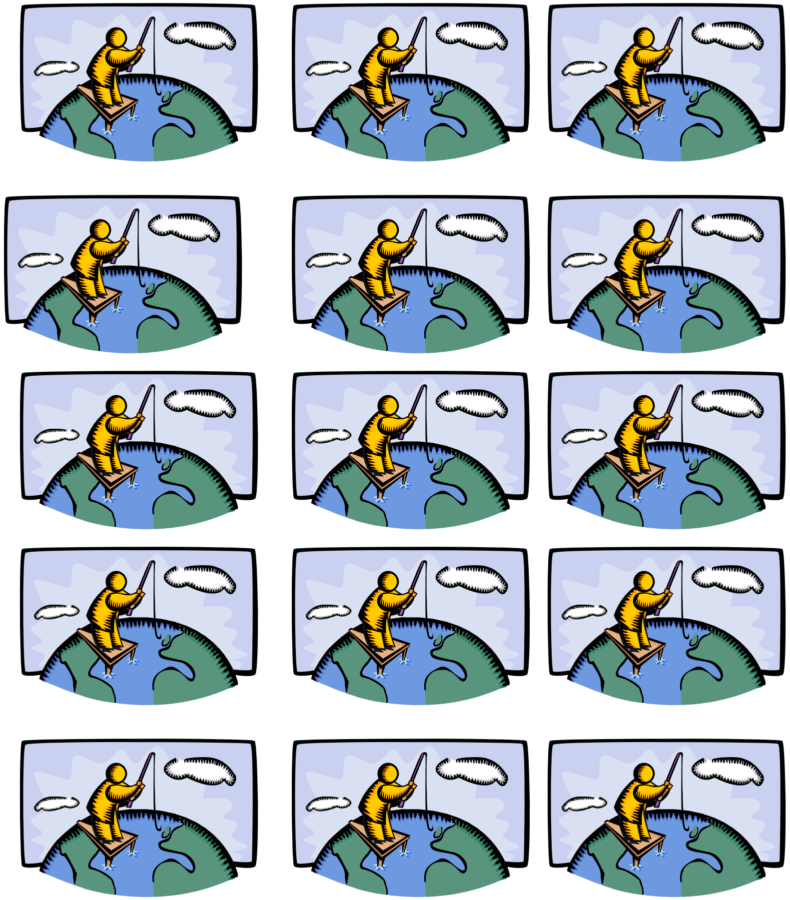
Parallelism

Parallelism

LCI workshop on high performance clustered computing

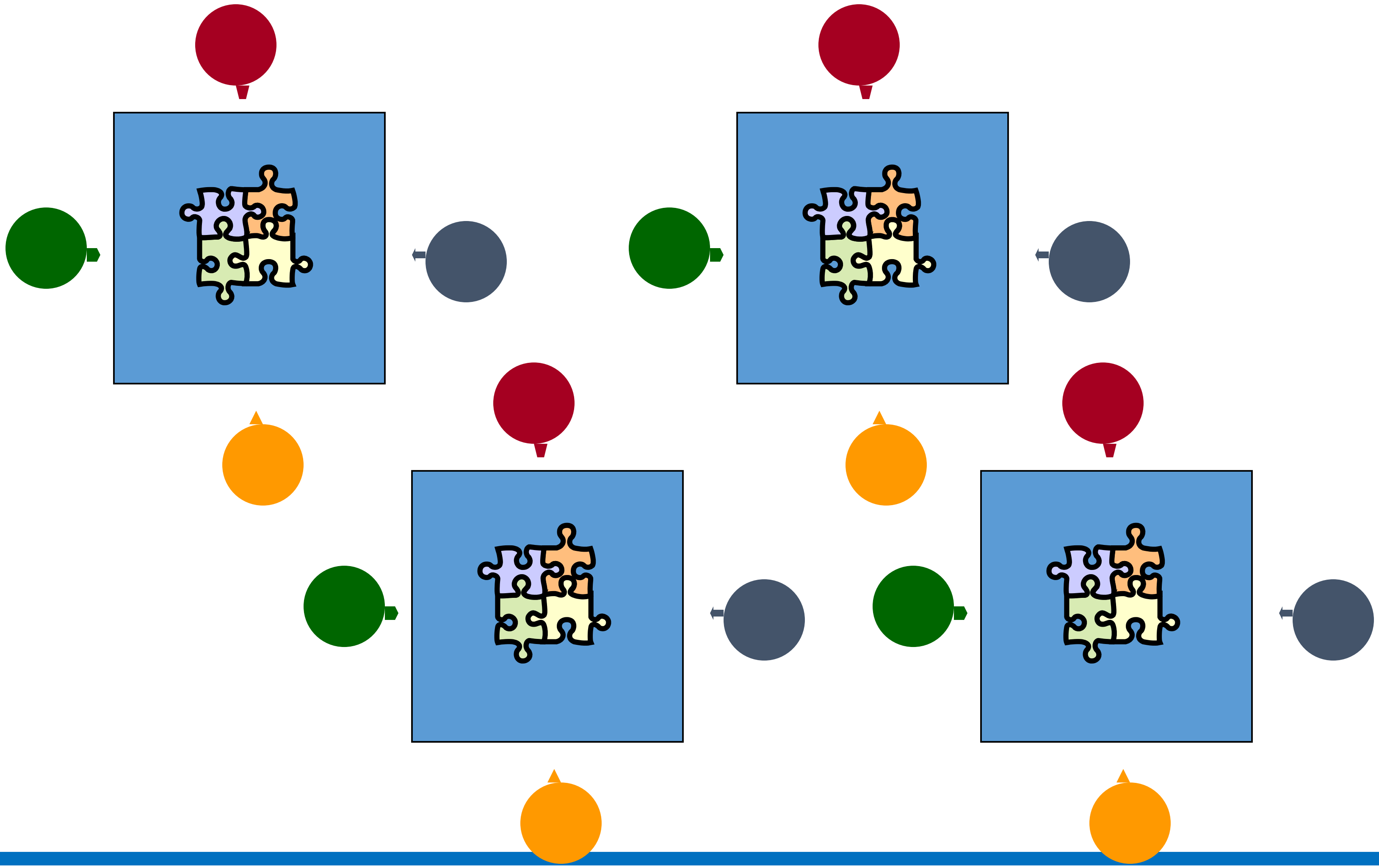
Parallelism means doing multiple things at the same time: you can get more work done in the same time.

Less fish …

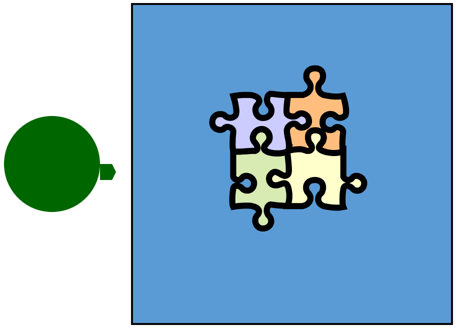


More fish!

The Jigsaw Puzzle Analogy



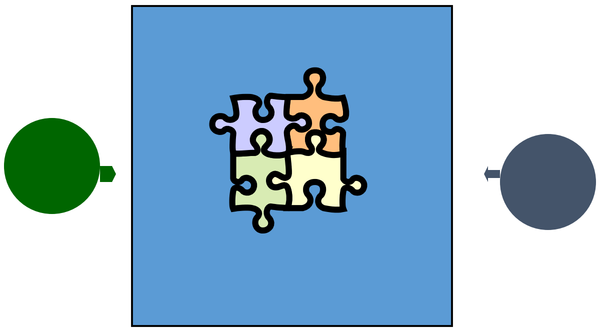
Serial Computing



Suppose you want to do a jigsaw puzzle that has, say, a thousand pieces.

We can imagine that it'll take you a certain amount of time. Let's say that you can put the puzzle together in an hour.

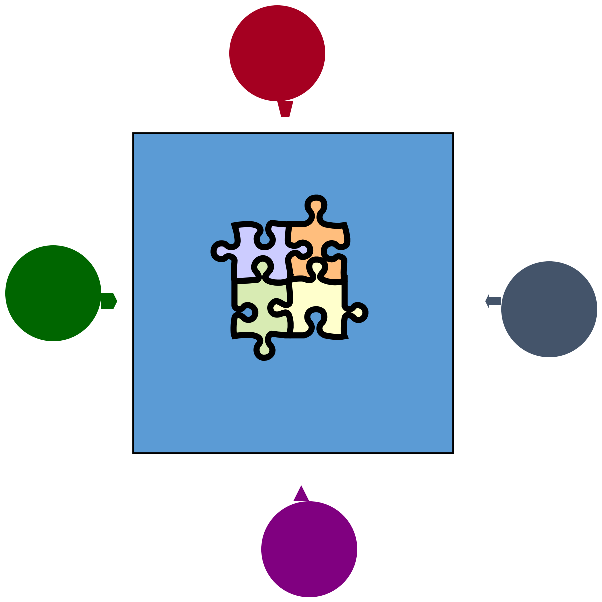
Shared Memory Parallelism



If Scott sits across the table from you, then he can work on his half of the puzzle and you can work on yours. Once in a while, you'll both reach into the pile of pieces at the same time (you'll ***contend*** for the same resource), which will cause a little bit of slowdown. And from time to time you'll have to work together (***communicate***) at the interface between his half and yours. The speedup will be nearly 2-to-1: y'all might take 35 minutes instead of 30.

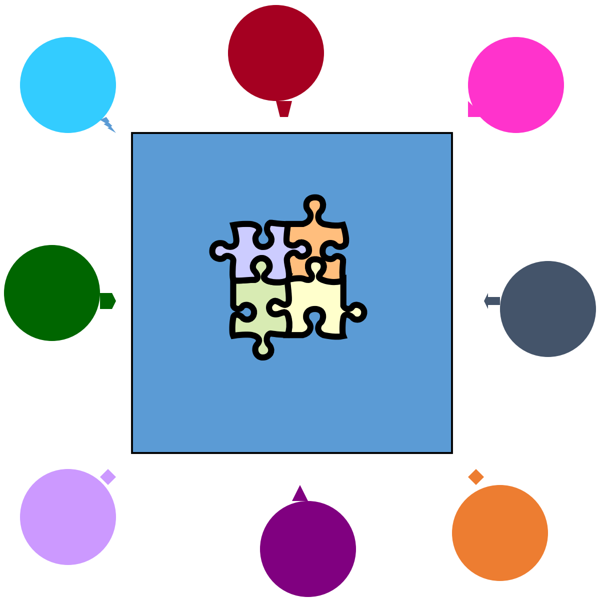
LCI workshop on high performance clustered computing

The More the Merrier?



Now let’s put Paul and Charlie on the other two sides of the table. Each of you can work on a part of the puzzle, but there’ll be a lot more contention for the shared resource (the pile of puzzle pieces) and a lot more communication at the interfaces. So y’all will get noticeably less than a 4-to-1 speedup, but you’ll still have an improvement, maybe something like 3-to-1: the four of you can get it done in 20 minutes instead of an hour.

Diminishing Returns



If we now put Dave and Tom and Horst and Brandon on the corners of the table, there's going to be a whole lot of contention for the shared resource, and a lot of communication at the many interfaces. So the speedup y'all get will be much less than we'd like; you'll be lucky to get 5-to-1.

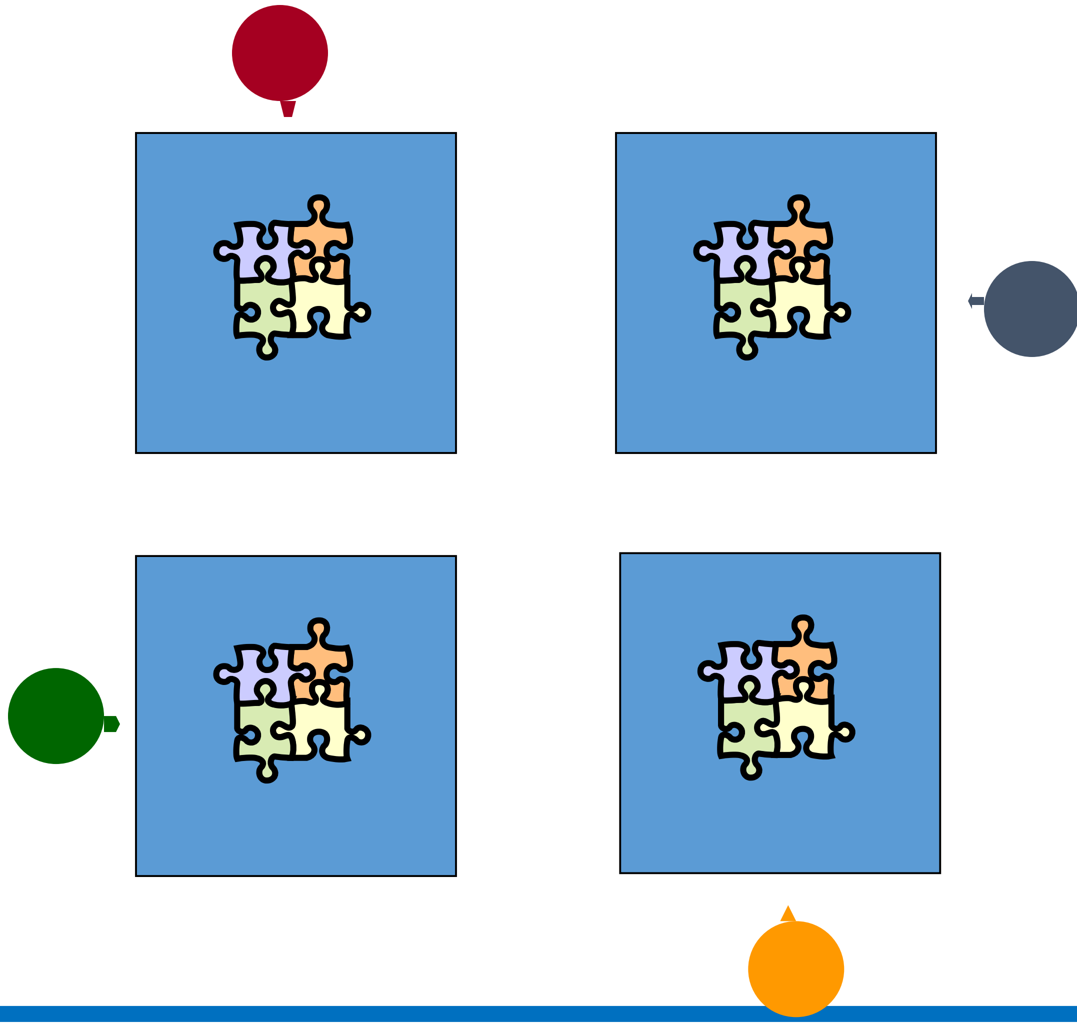
So we can see that adding more and more workers onto a shared resource is eventually going to have a diminishing return.

Distributed Parallelism

Now let’s try something a little different. Let’s set up two tables, and let’s put you at one of them and Scott at the other. Let’s put half of the puzzle pieces on your table and the other half of the pieces on Scott’s. Now y’all can work completely independently, without any contention for a shared resource. **BUT**, the cost per communication is **MUCH** higher (you have to scootch your tables together), and you need the ability to split up (***decompose***) the puzzle pieces reasonably evenly, which may be tricky to do for some puzzles.

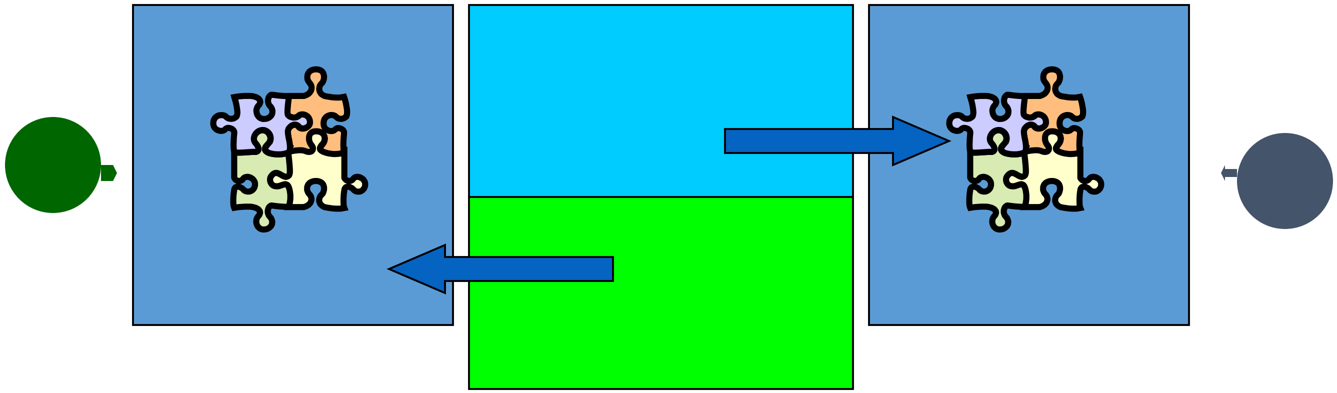
LCI workshop
on high performance clustered computing

More Distributed Processors



It’s a lot easier to add more processors in distributed parallelism. But, you always have to be aware of the need to decompose the problem and to communicate among the processors. Also, as you add more processors, it may be harder to ***load balance*** the amount of work that each processor gets.

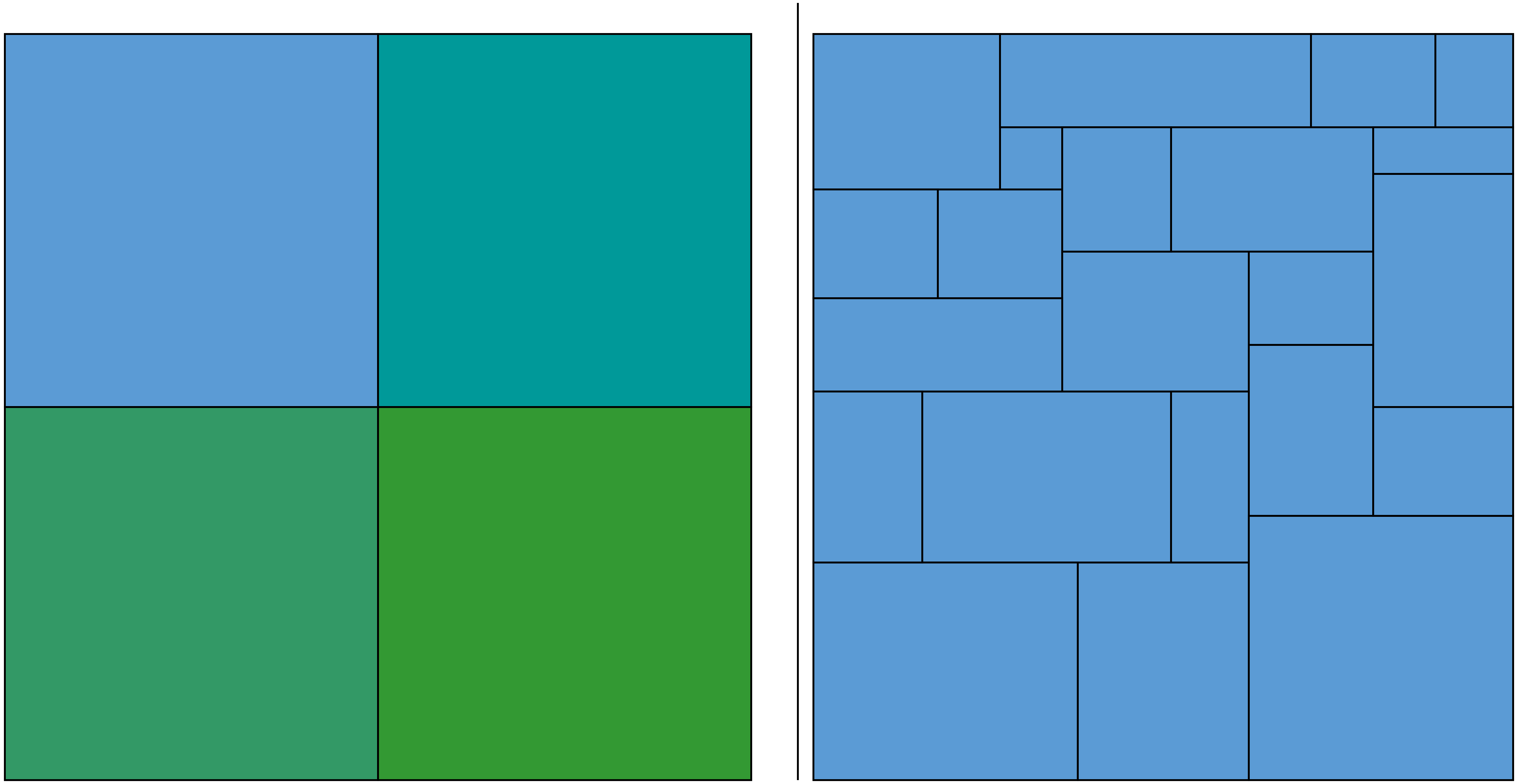
Load Balancing



Load balancing means ensuring that everyone completes their workload at roughly the same time.

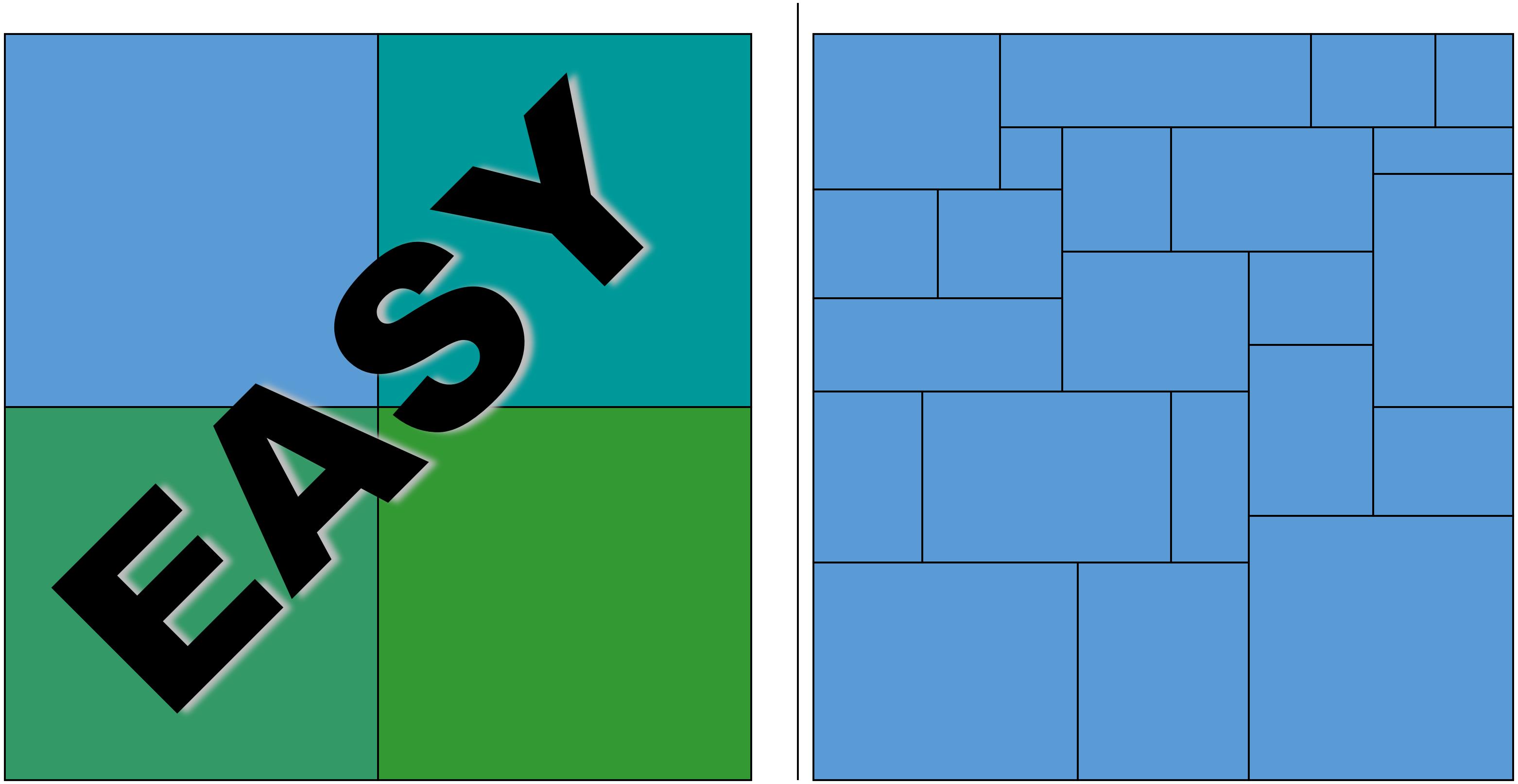
For example, if the jigsaw puzzle is half grass and half sky, then you can do the grass and Scott can do the sky, and then y'all only have to communicate at the horizon – and the amount of work that each of you does on your own is roughly equal. So you'll get pretty good speedup.

Load Balancing



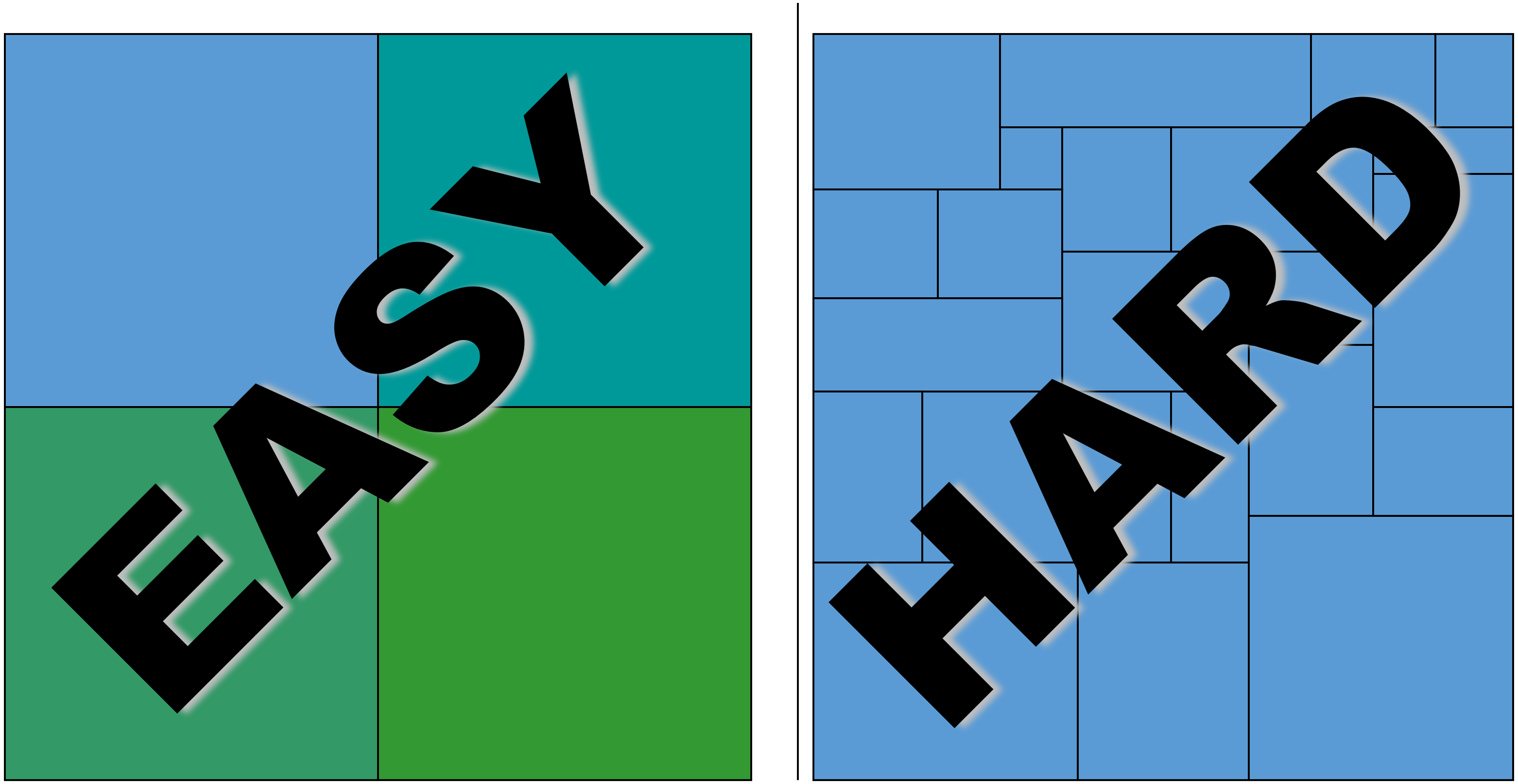
Load balancing can be easy, if the problem splits up into chunks of roughly equal size, with one chunk per processor. Or load balancing can be very hard.

Load Balancing

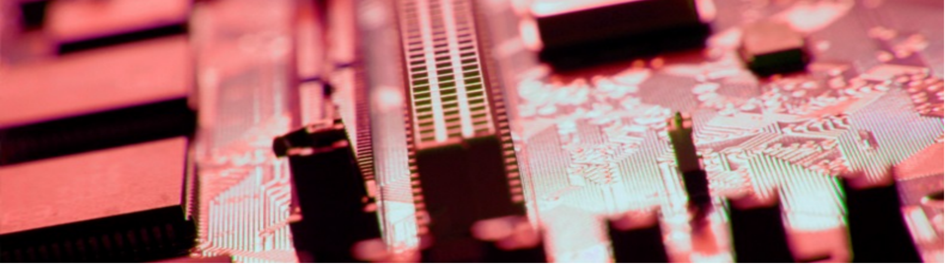


Load balancing can be easy, if the problem splits up into chunks of roughly equal size, with one chunk per processor. Or load balancing can be very hard.

Load Balancing



Load balancing can be easy, if the problem splits up into chunks of roughly equal size, with one chunk per processor. Or load balancing can be very hard.



Moore’s Law

Moore's Law

In 1965, Gordon Moore was an engineer at Fairchild Semiconductor.

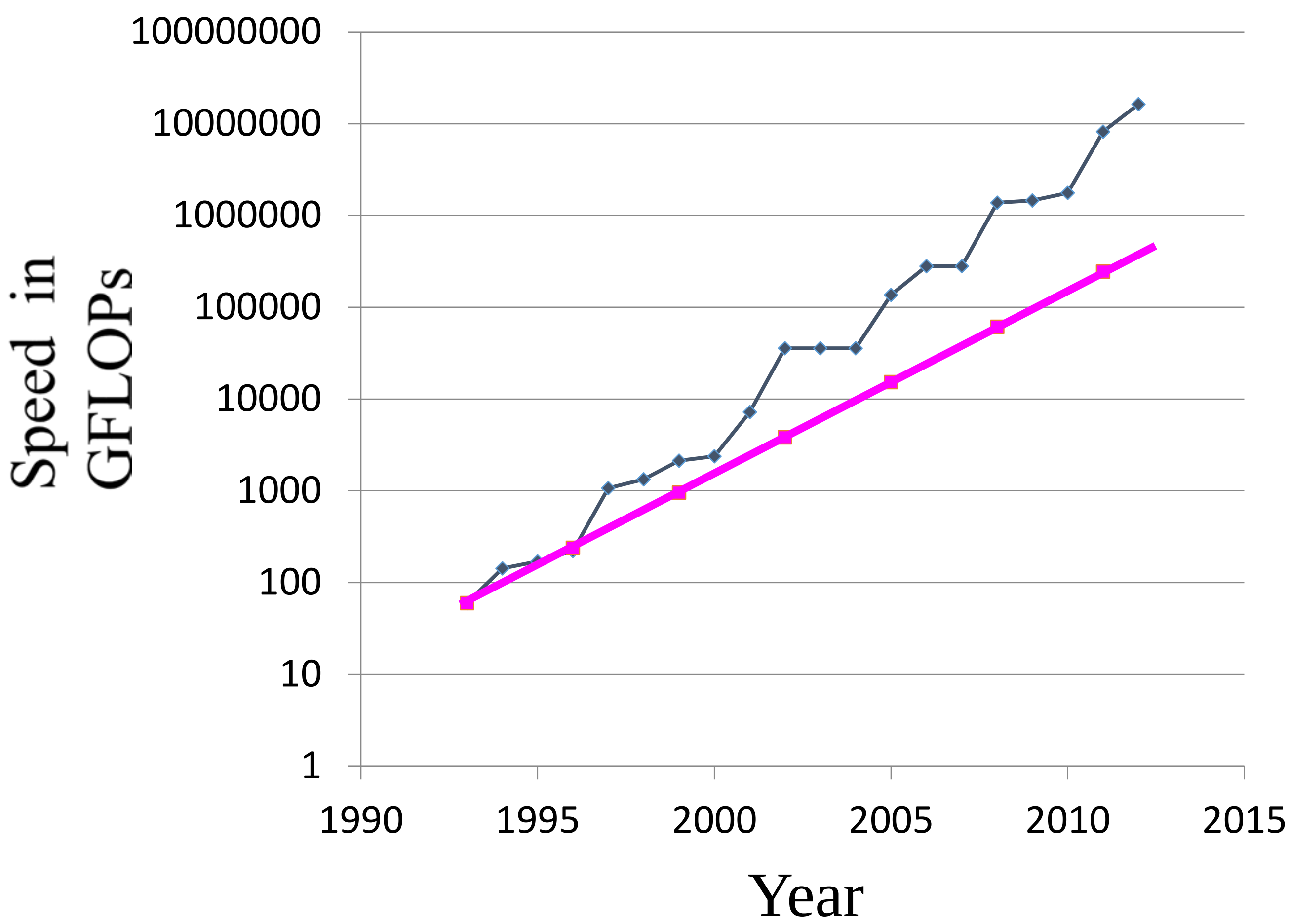
He noticed that the number of transistors that could be squeezed onto a chip was doubling about every 2 years.

It turns out that computer speed is roughly proportional to the number of transistors per unit area.

Moore wrote a paper about this concept, which became known as ***"Moore's Law."***

![LCI workshop on high performance clustered computing]

Fastest Supercomputer vs. Moore



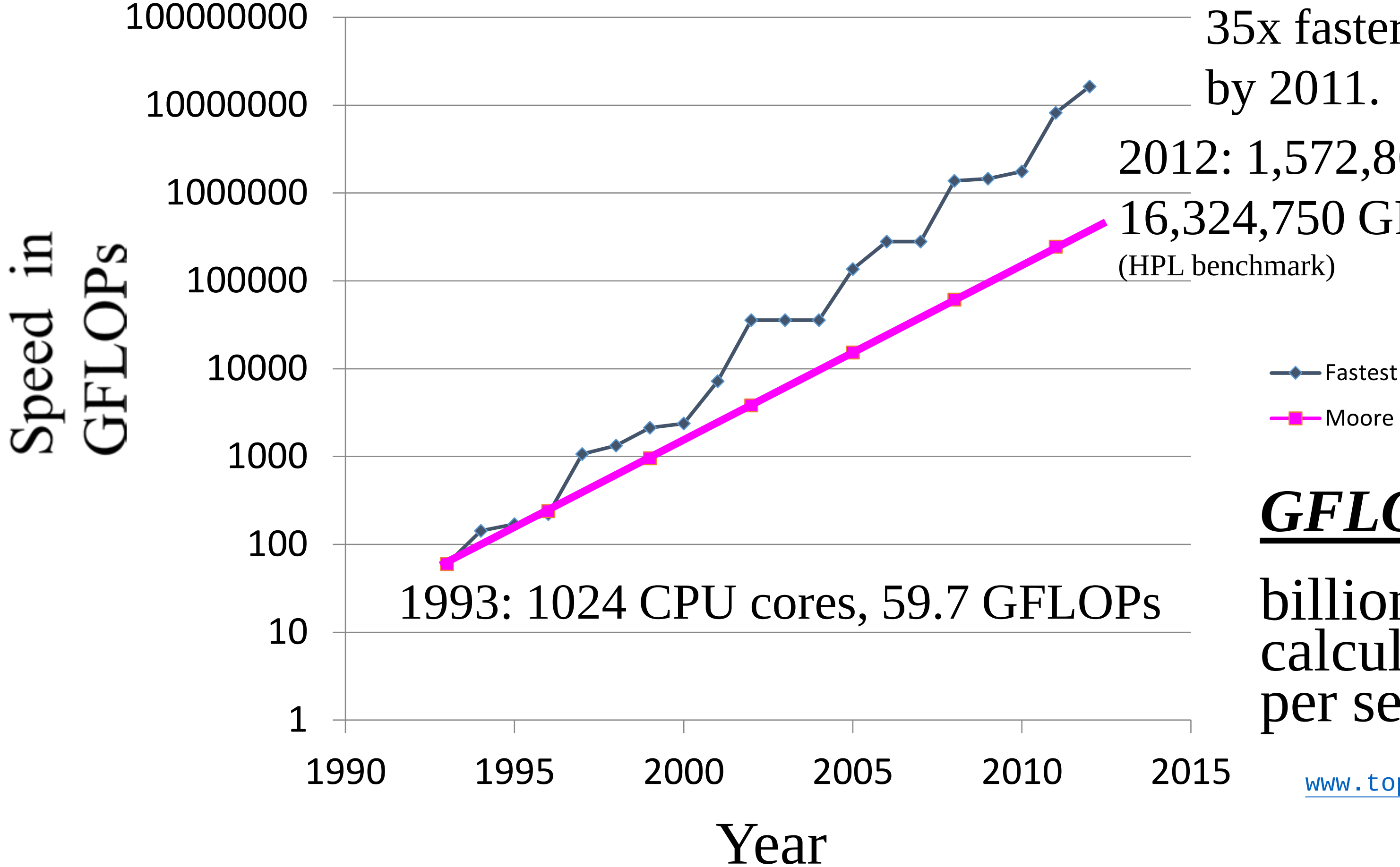
GFLOPs:

billions of calculations per second

www.top500.org

Fastest Supercomputer vs. Moore

Gap: Supercomputers had improved 35x faster than Moore by 2011.



2012: 1,572,864 CPU cores, 16,324,750 GFLOPs
(HPL benchmark)

GFLOPs: billions of calculations per second

www.top500.org

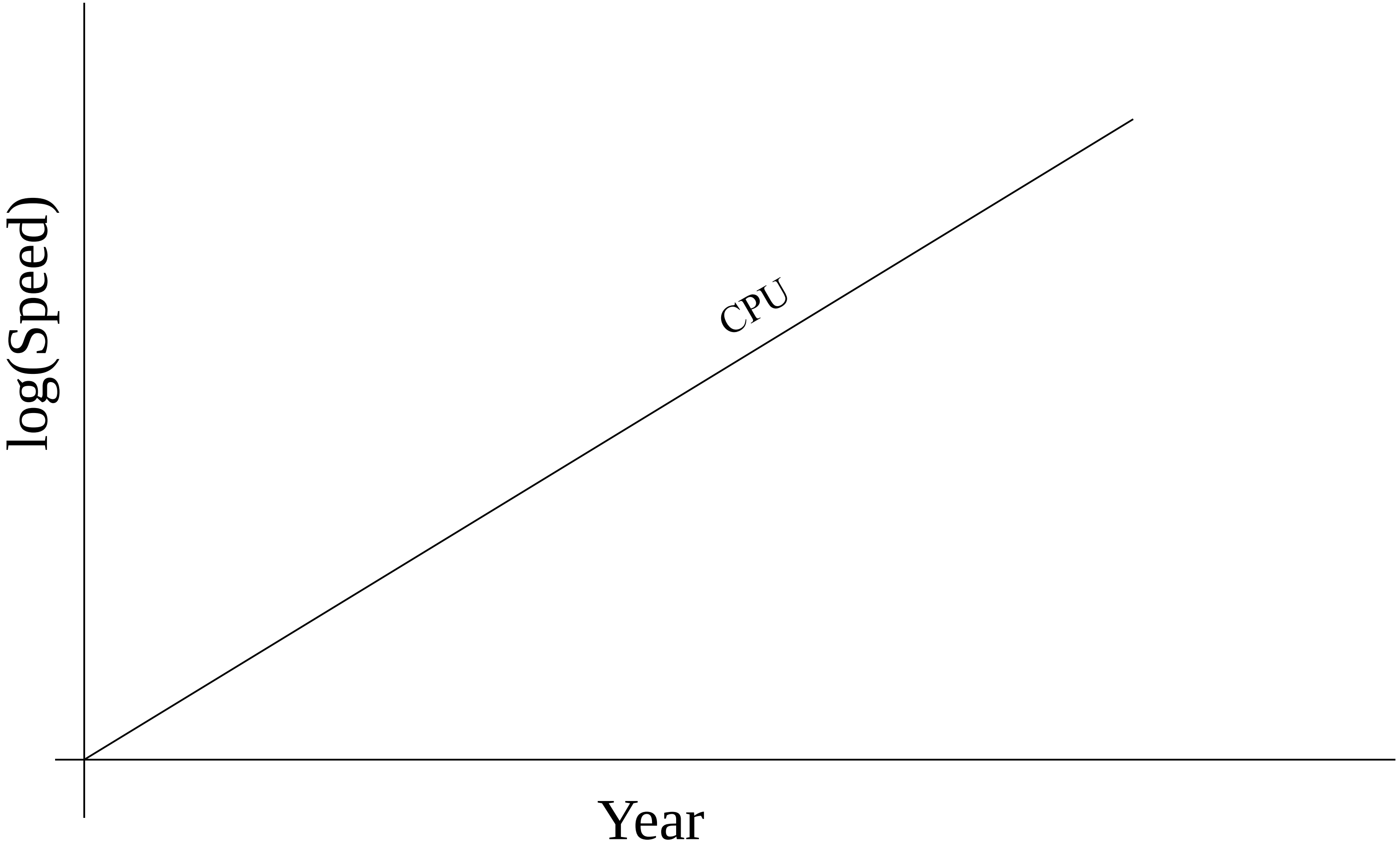
Moore: Uncanny!

- Nov 1971: Intel 4004 – 2300 transistors
- March 2010: Intel Nehalem Beckton – 2.3 billion transistors
- Factor of 1M improvement in 38 1/3 years
- $2^{(38.33 \text{ years} / 1.9232455)} = 1{,}000{,}000$

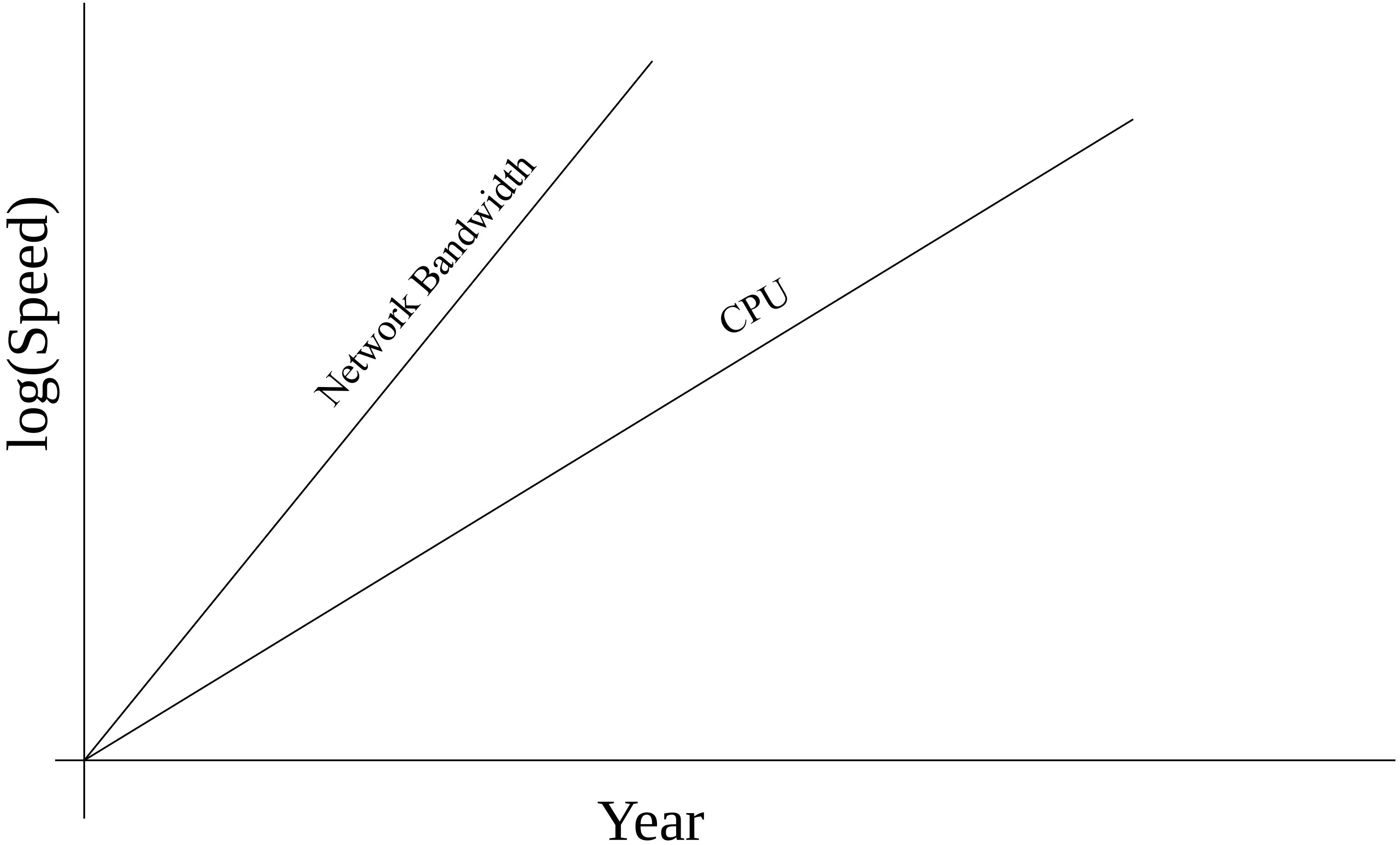
So, transistor density has doubled every 23 months:

UNCANNILY ACCURATE PREDICTION!

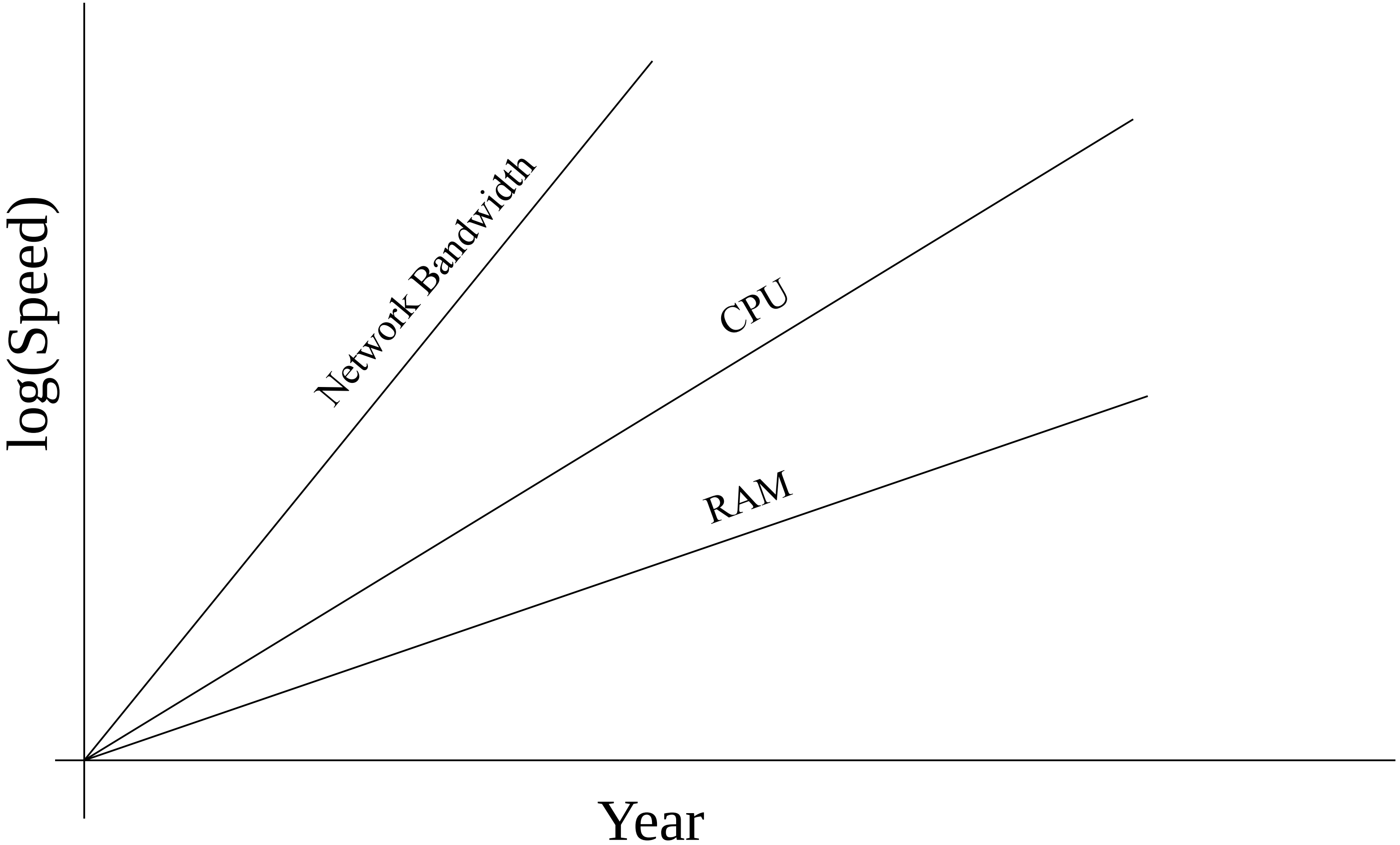
Moore's Law in Practice



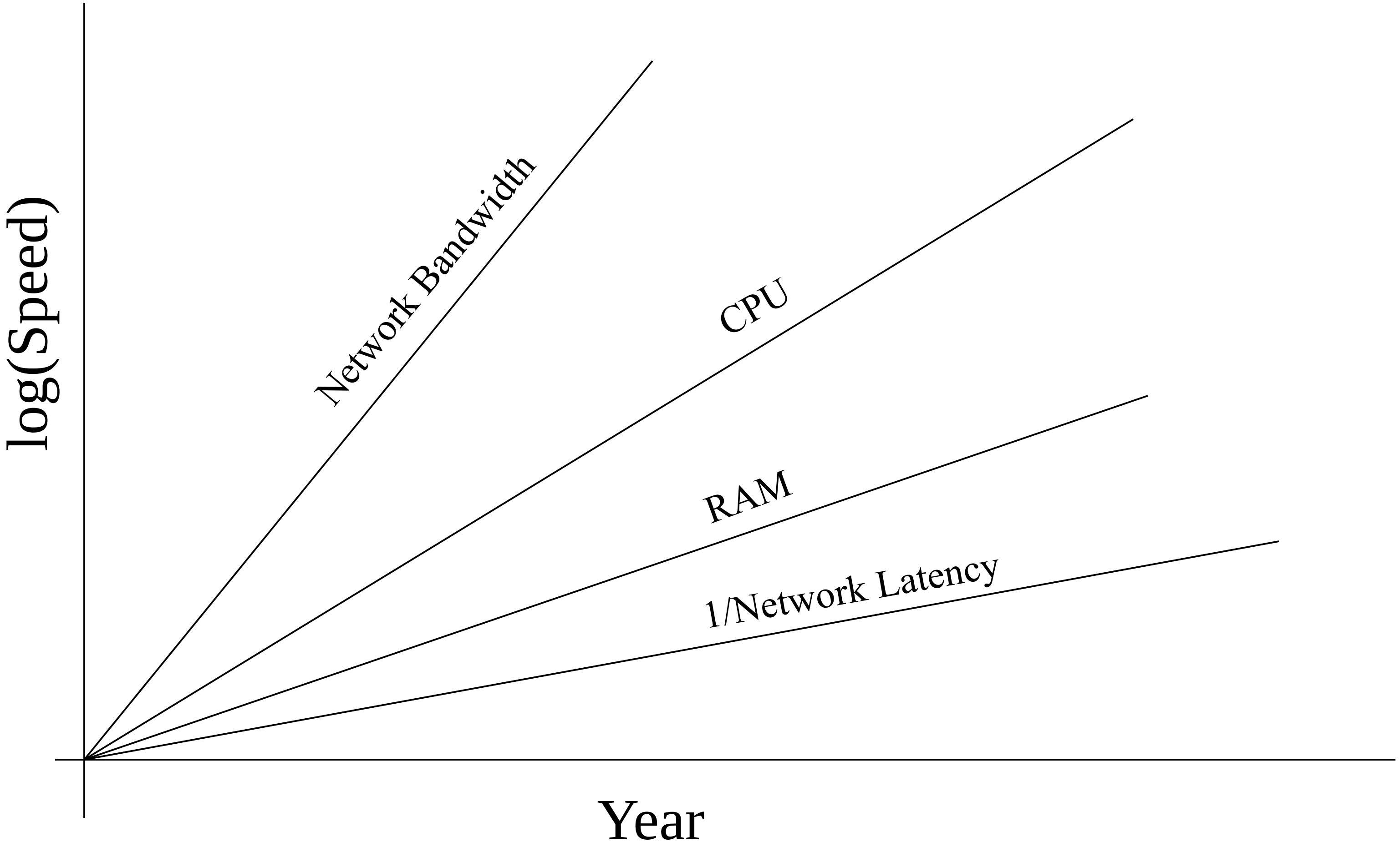
Moore's Law in Practice



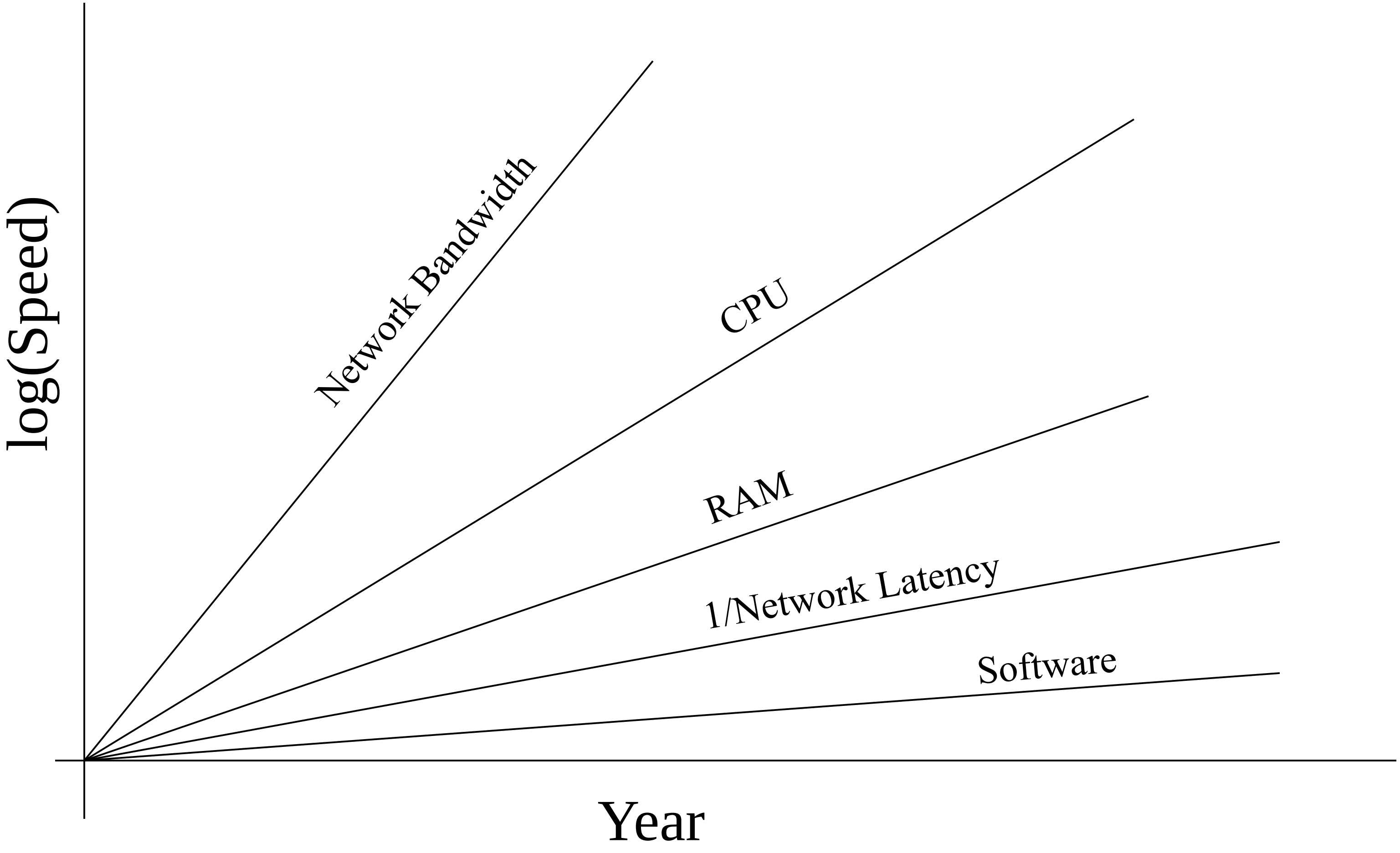
Moore's Law in Practice



Moore's Law in Practice

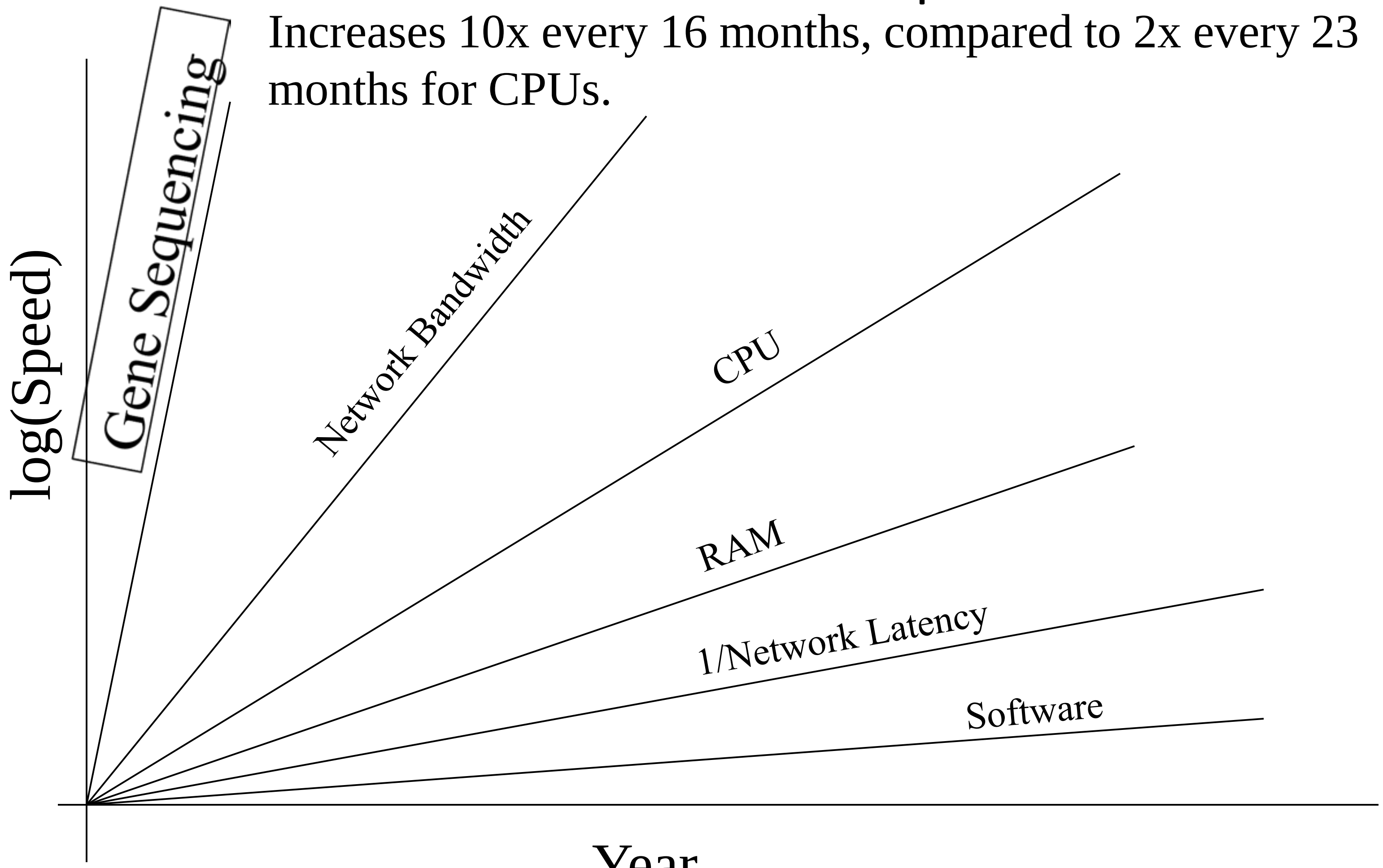


Moore's Law in Practice



Moore's Law on Gene Sequencers

Increases 10x every 16 months, compared to 2x every 23 months for CPUs.



LCI workshop
on high performance clustered computing

What does 1 TFLOPs Look Like?

1997: Room

ASCI RED[13]
Sandia National Lab

2002: Row

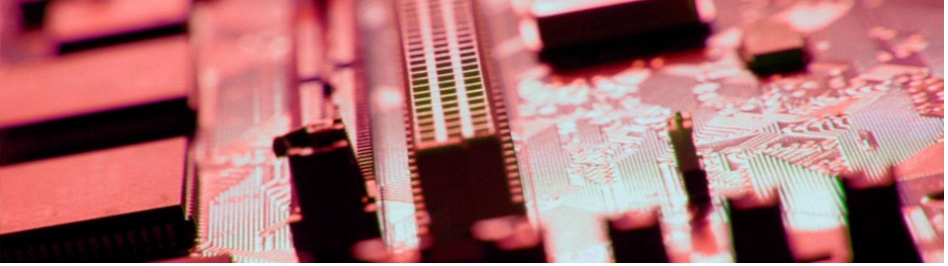
`boomer.oscer.ou.edu`
In service 2002-5: 11 racks

2012: Card

AMD FirePro W9000[14]

NVIDIA Kepler K20[15]

Intel MIC Xeon Phi[16]



Why Bother?

Why Bother with HPC at All?

It's clear that making effective use of HPC takes quite a bit of effort, both learning how and developing software.

That seems like a lot of trouble to go to just to get your code to run faster.

It's nice to have a code that used to take a day, now run in an hour. But if you can afford to wait a day, what's the point of HPC?

Why go to all that trouble just to get your code to run faster?

Why HPC is Worth the Bother

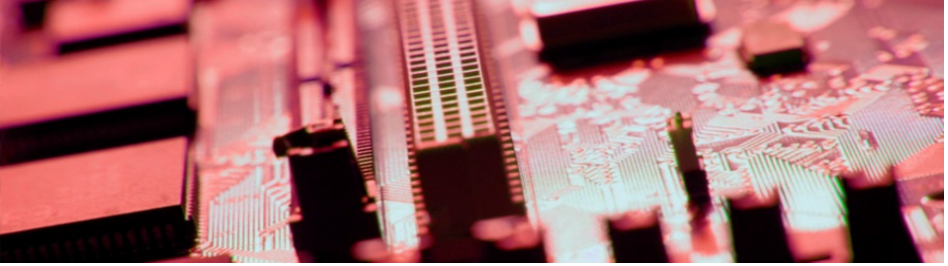
- What HPC gives you that you won't get elsewhere is the ability to do **bigger, better, more exciting science**. If your code can run faster, that means that you can tackle much bigger problems in the same amount of time that you used to need for smaller problems.
- HPC is important not only for its own sake, but also because what happens in HPC today will be on your desktop in about 10 to 15 years and on your cell phone in 25 years:
  it puts you **ahead of the curve**.

The Future is Now

Historically, this has always been true:

Whatever happens in supercomputing today will be on your desktop in 10 – 15 years.

So, if you have experience with supercomputing, you'll be ahead of the curve when things get to the desktop.



Thanks for your attention!

Questions?

www.oscer.ou.edu

References

[1] Image by Greg Bryan, Columbia U.
[2] "Update on the Collaborative Radar Acquisition Field Test (CRAFT): Planning for the Next Steps." Presented to NWS Headquarters August 30 2001.
[3] See http://hneeman.oscer.ou.edu/hamr.html for details.
[4] http://www.dell.com/
[5] http://www.vw.com/newbeetle/
[6] Richard Gerber, The Software Optimization Cookbook: High-performance Recipes for the Intel Architecture. Intel Press, 2002, pp. 161-168.
[7] RightMark Memory Analyzer. http://cpu.rightmark.org/
[8] ftp://download.intel.com/design/Pentium4/papers/24943801.pdf
[9] http://www.samsungssd.com/meetssd/techspecs
[10] http://www.samsung.com/Products/OpticalDiscDrive/SlimDrive/OpticalDiscDrive_SlimDrive_SN_S082D.asp?page=Specifications
[11] ftp://download.intel.com/design/Pentium4/manuals/24896606.pdf
[12] http://www.pricewatch.com/