Introduction to all forms of parallel computing

Prof Alan Edelman

(18.s192/16.s098)

- All material on https://github.com/mitmath/Parallel-Computing-Spoke
  - Suggest bookmarking
- Canvas only used for submitting hws, nothing else

What will you learn in this class?

use Julia to write
-portable parallel programs on GPUs
-use multithreading on your own laptops
-and distributed computing on multiprocessors.

Get a good feel for what is and is not possible in gaining speedups and performance,

Have a sense of possible research directions including applications and the use of AI, LLMs, and ML, and perhaps most valuably, real progress towards making parallel computing easier.

What should you learn in another class?

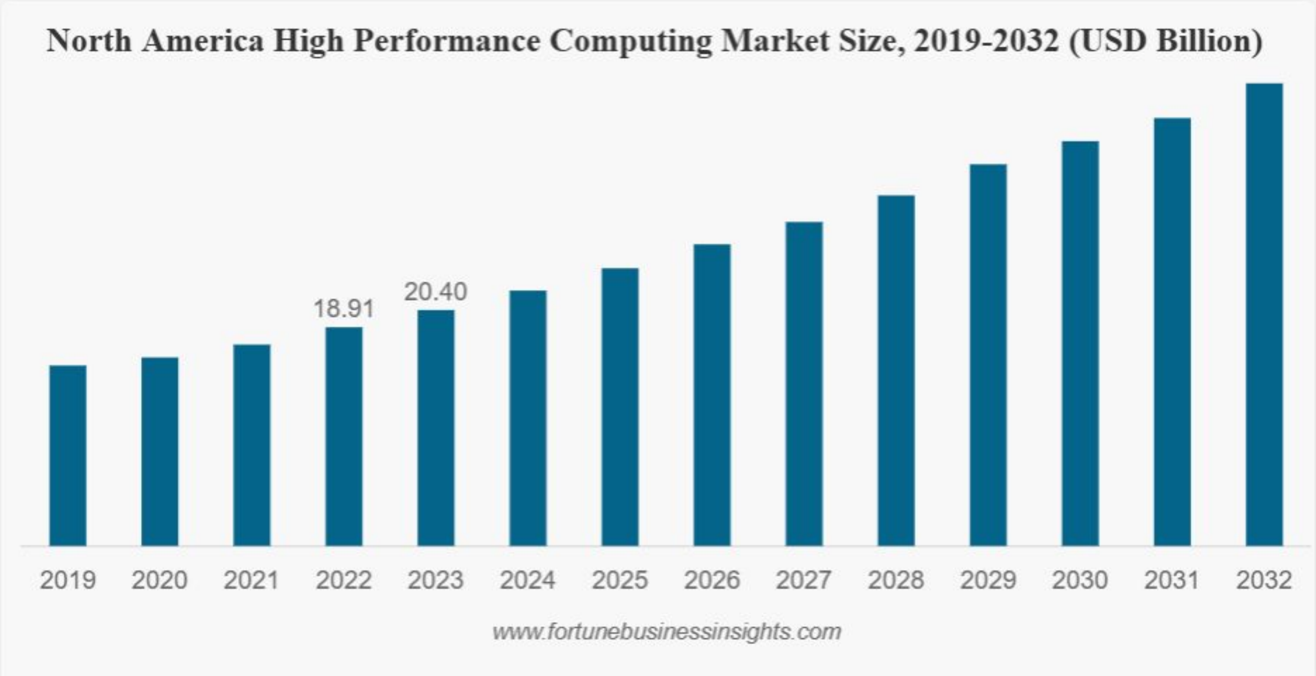
6.1060 performance engineering - low-level optimization

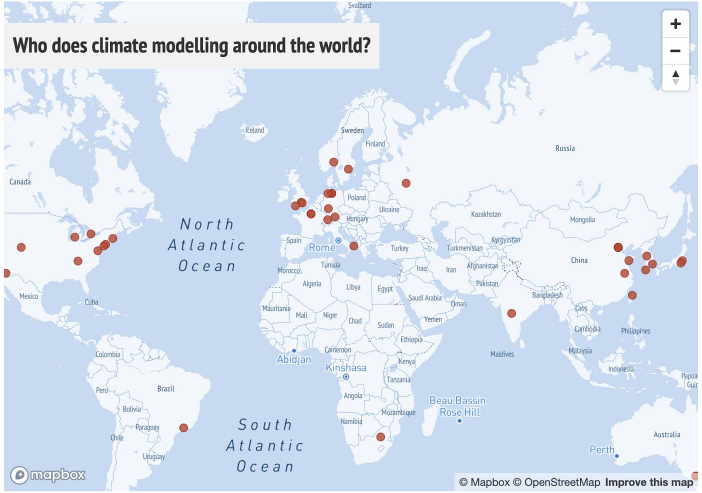
6.S894 low-level performance engineering for GPUs

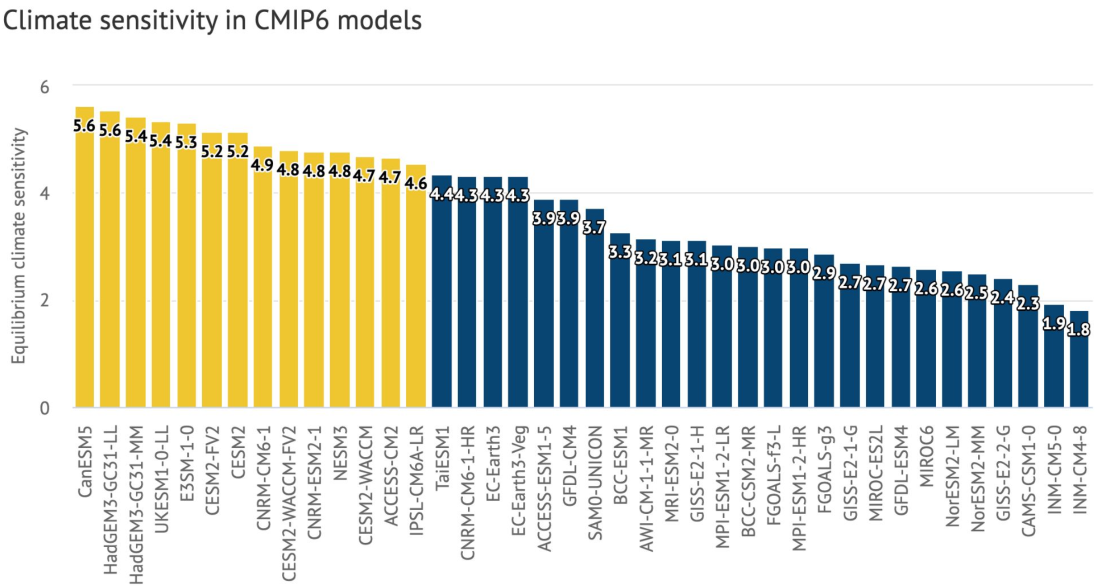
6.1100 computer language engineering - compiler tinkering

Why High-Performance Computing and Parallelism

1. You cannot escape: HPC is will be a >100B market in less than a decade







2. Large data make faster larger computers necessary

Large Climate models run on supercomputers at big centers

Source:
https://www.carbonbrief.org/cmip6-the-next-generation-of-climate-models-explained/

3. Because it’s fun!

Parallelism

Biggest challenge to HPC (my opinion)

- Software situation is really not great (understatement?)
- Too many kinds of parallelism, not enough coherence (yet?)
- Research community tends to prefer performance boasting to usable software
- Same problem gets solved over and over again

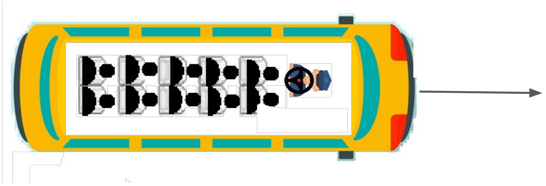
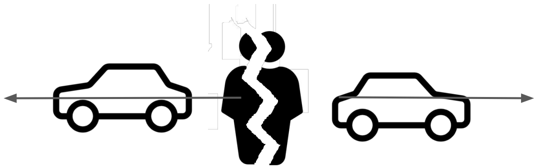
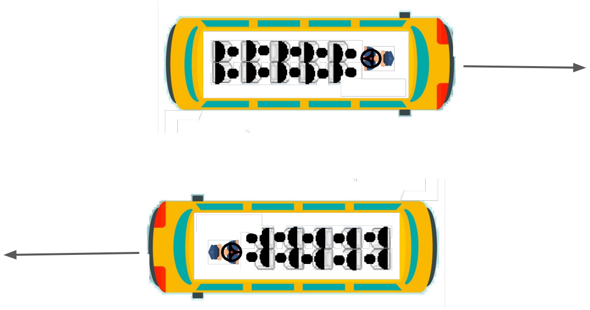
Intro to concepts of parallel computing

Flops

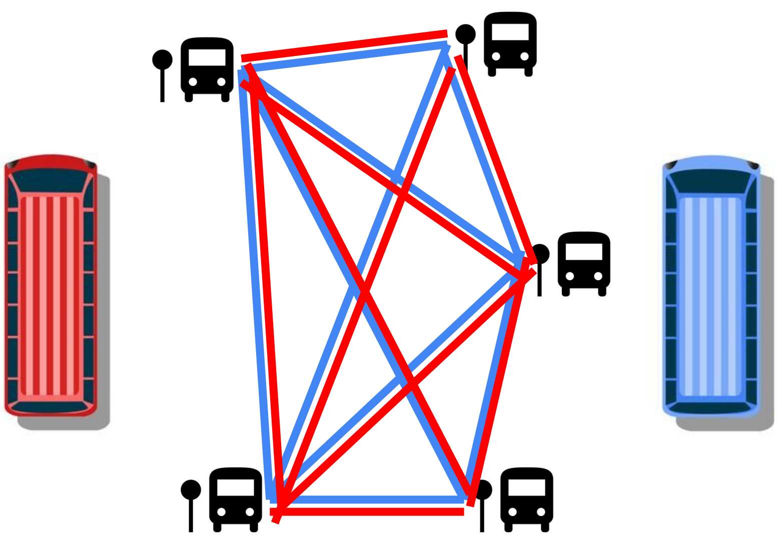
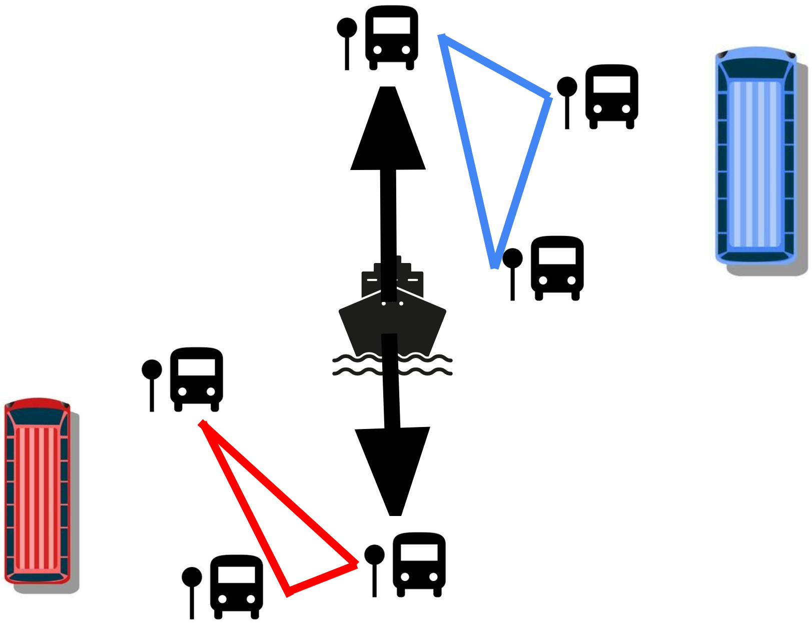
Used as #floating point ops and also floating ops/second

- Discussion of top500.org including exaflop machines
- Comparing the rate of matrix multiply computation vs the rate of matrix addition computation.
  - Note: rate means we are counting the speed of each operation,normalizing by the number of operations.

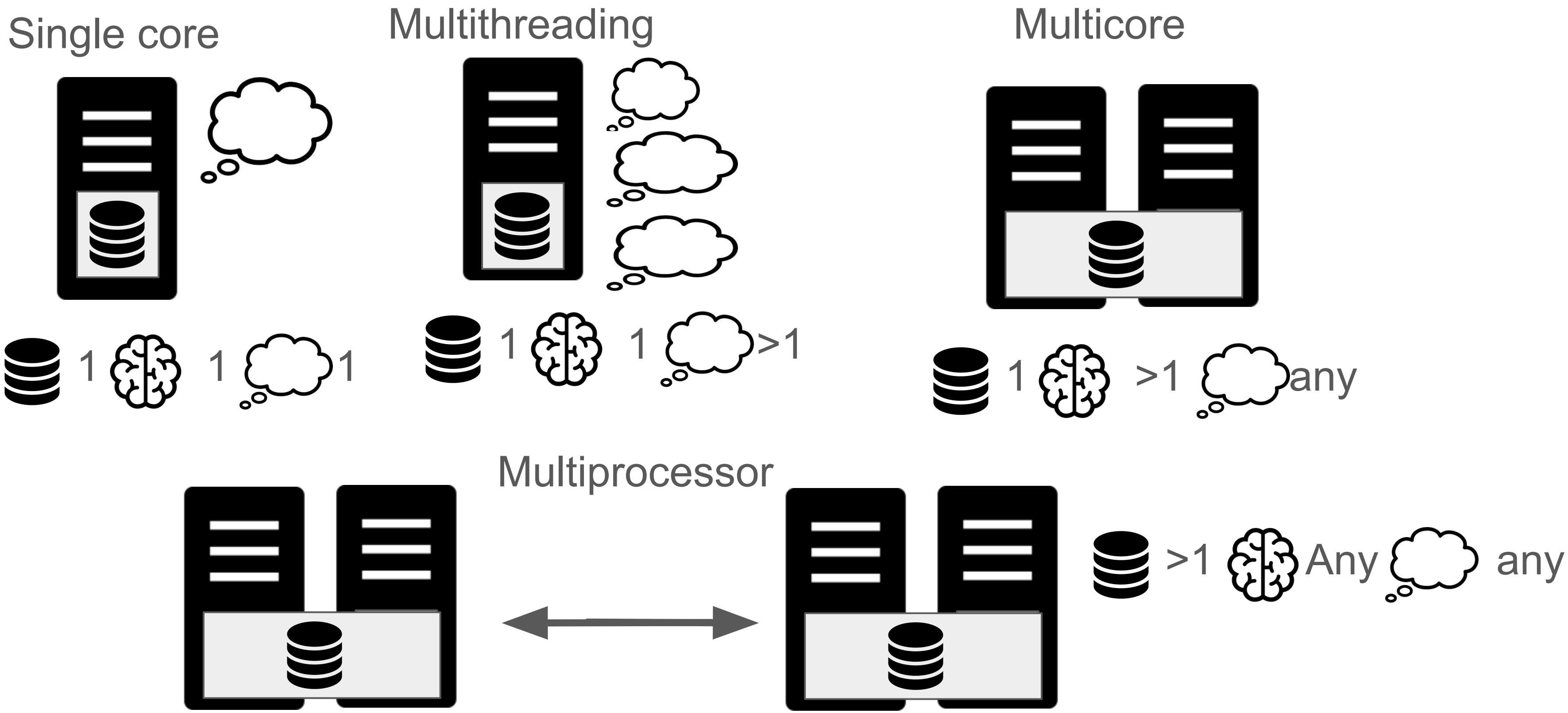
The concept of parallelism

	Single data	Multiple data
Single instruction		
Multiple instruction		

The concept of parallelism

Shared memory	Distributed memory
	
Two bus services serve all stops	Each bus service has their own hubs; and there is one slower connection between them

Levels of parallelism



Levels of parallelism - the technical and all the options

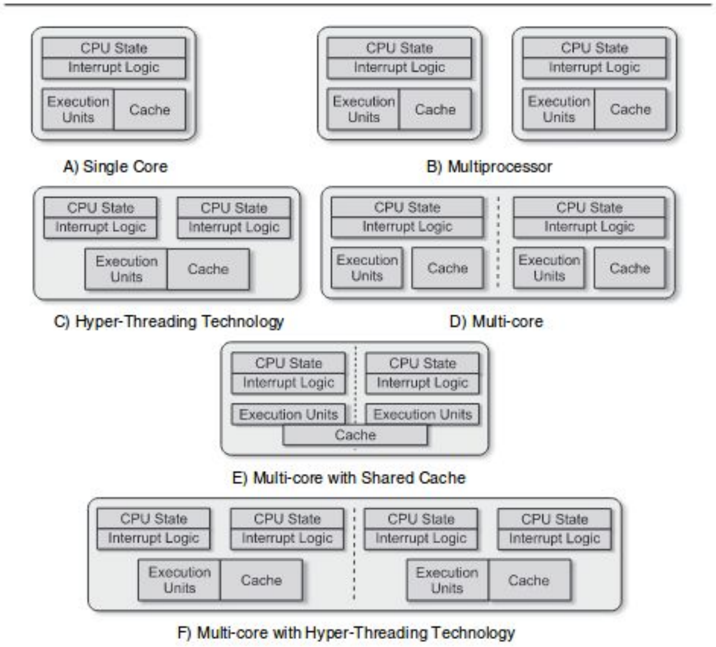


Figure 1.4 Simple Comparison of Single-core, Multi-processor, and Multi-Core Architectures

In julia, multithreading is achieved with tasks

source: https://superuser.com/questions/122536/what-is-hyper-threading-and-how-does-it-work