High-Performance Computing

12.1: Concurrent Processing

High-Performance Computing

- A fancy term for computers significantly faster than your average desktop machine (Dell, Mac)

- For most computational modelling, High *Productivity* Computing (C. Moler) is more important (human time more costly than machine time).

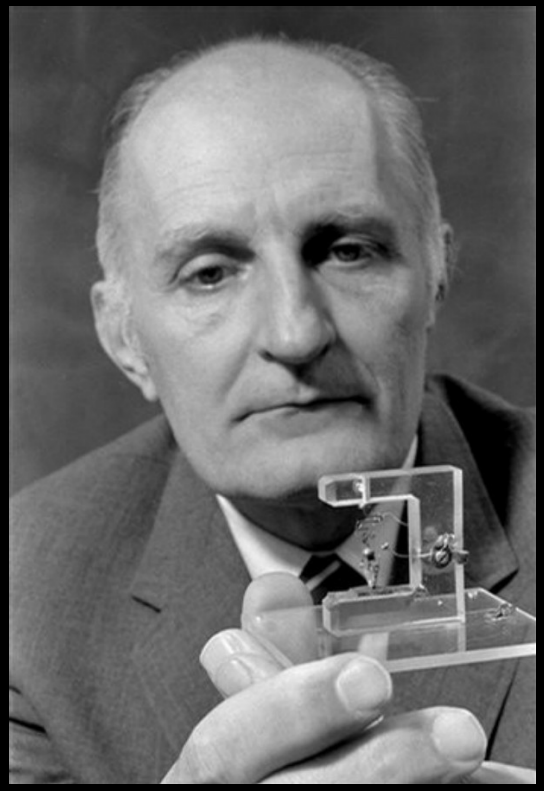
- But there will always be applications for computers that maximize performance, so HPC is worth knowing about

Background: Moore’s Law

- Moore’s Law: Computing power (number of transistors, or switches, basic unit of computation) available at a given price doubles roughly every 18 months

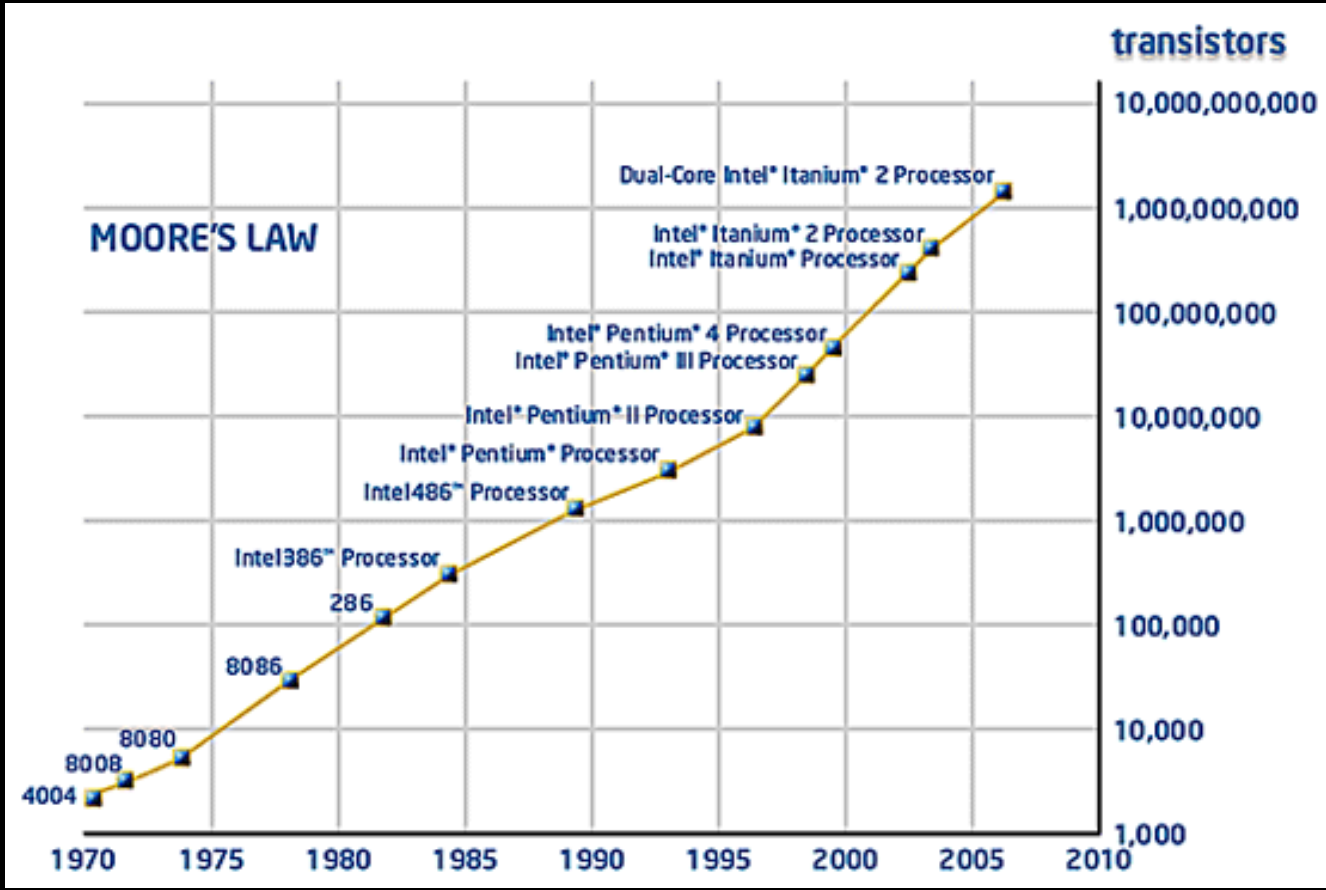
- (So why don’t we have (super)human machine intelligence by now?)

Background: Moore's Law



Morgan Sparks (1916-2008)
with an early transistor

Background: Moore's Law



Computer Architecture Basics

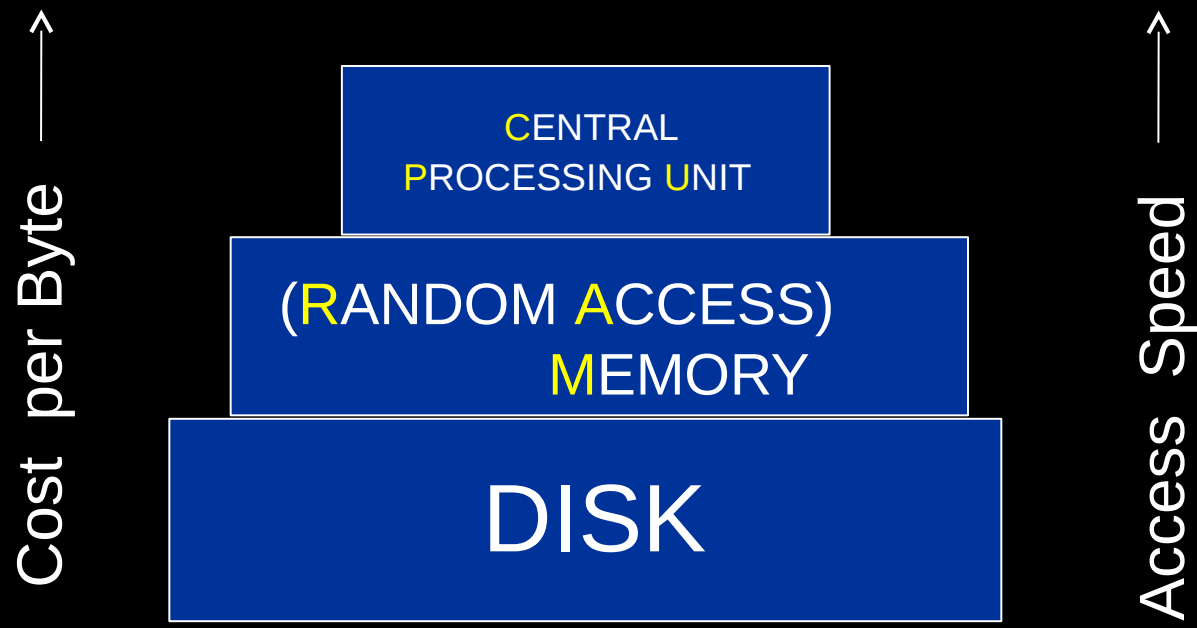
- Architecture is used in two different senses in computer science

  1. Processor Architecture (Pentium architecture, RISC architecture, etc.): the basic instruction set (operations) provided by a given “chip”

  2. Layout of CPU & Memory (& disk)

- We will use the latter (more common) sense

Computer Architecture Basics



Spreadsheet Example

- Double-click on (open) document: loads spreadsheet data and program (Excel) from disk into memory

- Type a formula (= A1*B3 > C2) and hit return:

  1. Numbers are loaded into CPU's registers from memory
  2. CPU performs arithmetic & logic to compute answer (ALU = Arithmetic / Logic Unit)
  3. Answer is copied out to memory (& displayed)

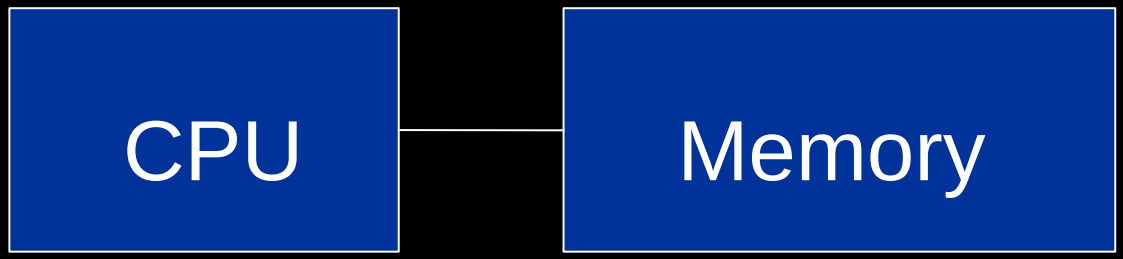
- Frequently accessed memory areas may be stored in CPU's cache

- Hit *Save*: memory is copied back to disk

Sequential Processing

- From an HPC perspective, the important things are CPU, memory, and how they are connected.

- Standard desktop machine is (until recently!) sequential: one CPU, one memory, one task at a time:



Concurrent Processing

- The dream has always been to break through the von Neumann bottleneck and do more than one computation at a given time

- Two basic varieties

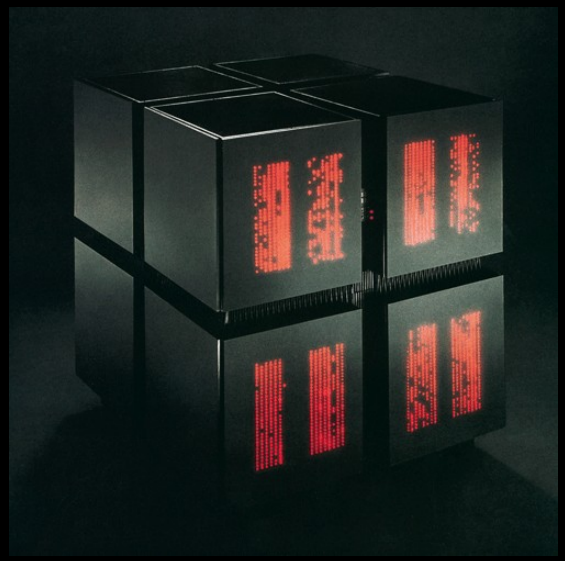
    - Parallel Processing: several CPUs inside the same hardware “box”

    - Distributed Processing: multiple CPUs connected over a network

Parallel Processing: A Brief History

- In general, the lesson is that it is nearly impossible to make money from special-purpose parallel hardware “boxes”
- 1980’s - 1990’s: “Yesterday’s HPC is tomorrow’s doorstop”
  - Connection Machine
  - MasPar
  - Japan’s “Fifth Generation”
- “The revenge of Moore’s Law” by the time you finish building the $$$$ supercomputer, the $$ computer is fast enough (though there was always a market for supercomputers like the Cray)

Supercomputers of Yesteryear



Connection Machine
CM-1 (1985)

Cray YM-P (1988)

MasPar MP-1 (1990)

Distributed Processing: A Brief(er) History

- 1990’s - 2000’s: “Age of the cluster”
- Beowulf: lots of commodity (inexpensive) desktop machines (Dell) wired together in a rack with fast connections, running Linux (free, open-source OS)
- Cloud Computing: “The internet is the computer” (like Gmail, but for computing services)

Today: Back to Parallel Processing

- Clusters take up lots of room, require lots of air conditioning, and require experts to build, maintain, & program
- Cloud Computing sabotaged by industry hype (Larry Ellison comment)
- Sustaining Moore's Law requires increasingly sophisticated advances in semiconductor physics

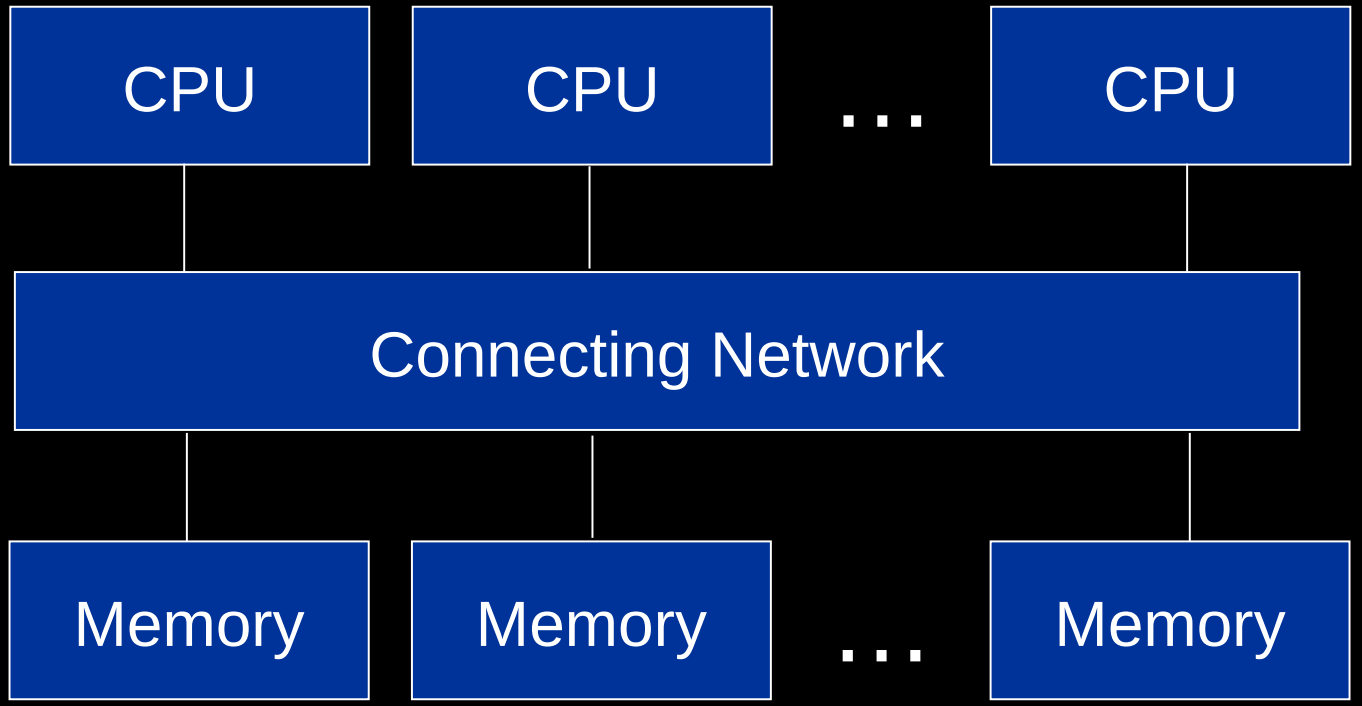
Today: Back to Parallel Processing

- Two basic directions
  - Multicore / multiprocessor machines : lots of little CPUs inside your desktop/laptop computer (and now, cellphone!)
  - Inexpensive special-purpose hardware like Graphics Processing Units

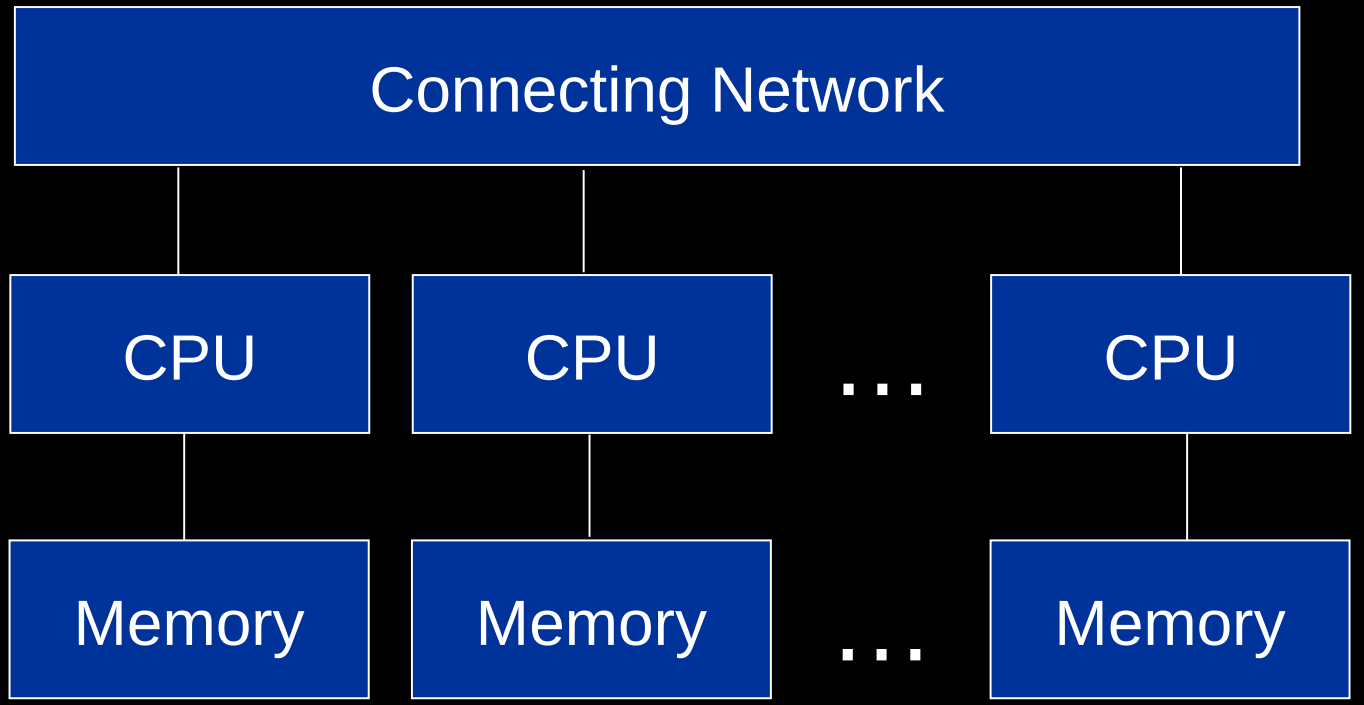
Multiprocessor Architectures

- Two basic designs
  - Shared memory multiprocessor: all processors can access all memory modules
  - Message-passing multiprocessor
    - Each CPU has its own memory
    - CPU's pass "messages" around to request/provide computation

Shared Memory Multiprocessor



Message-Passing Multiprocessor



Scalability is Everything

- Which is better?
  - $1000 today
  - $100 today, plus a way of making $100 more every day in the future?
- Scalability is the central question not just for hardware, but also for software and algorithms (think “economy of scale”)

Processes & Streams

- Process: an executing instance of a program (J. Plank)
- Instruction stream: sequence of instructions coming from a single process
- Data stream: sequence of data items on which to perform computation

Flynn's Four-Way Classification

1. SISD: Single Instruction stream, Single Data stream. You rarely hear this term, because it's the default (though this is changing)

2. MIMD: Multiple Instruction streams, Multiple Data streams

    - Thread (of execution): "lightweight" process executing on some part of a multiprocessor

    - GPU is probably best current exemplar

Flynn's Four-Way Classification

3. SIMD: Single Instruction stream, Multiple Data streams -- same operation on all data at once (recall Matlab, though it's not (yet) truly SIMD)

4. MISD: "Disagreement exists on whether this category has any systems"

    - Pipelining is perhaps an example: think of breaking weekly laundry into two loads, drying first load while washing second

Communication

- “Pure parallelism”: like physics without friction
  - It’s useful as a first approximation to pretend that processors don’t have to communicate results
  - But then you have to deal with the real issues

Granularity & Speedup

- Granularity: ratio of computation time to communication time
  - Lots of “tiny little computers” (grains) means small granularity (because they have to communicate a lot)
- Speedup: how much faster is it to execute the program on *n* processors vs. 1 processor?

Linear Speedup

- In principle, maximum speedup is linear: *n* times faster on *n* processors
- This gives a decaying (k/*n*) exponential curve of execution time vs. processors
- “Super-linear” speedup is sometimes possible, if each of the processors can access memory more efficiently than a single processor (c.f. cache concept for web pages)