COMP 4901Q: High Performance Computing (HPC)

Lecture 2: Introduction to Parallel Computer Architecture

Instructor: Shaohuai SHI (shaohuais@cse.ust.hk)
Teaching assistants: Mingkai TANG (mtangag@connect.ust.hk)
Yazhou XING (yxingag@connect.ust.hk)

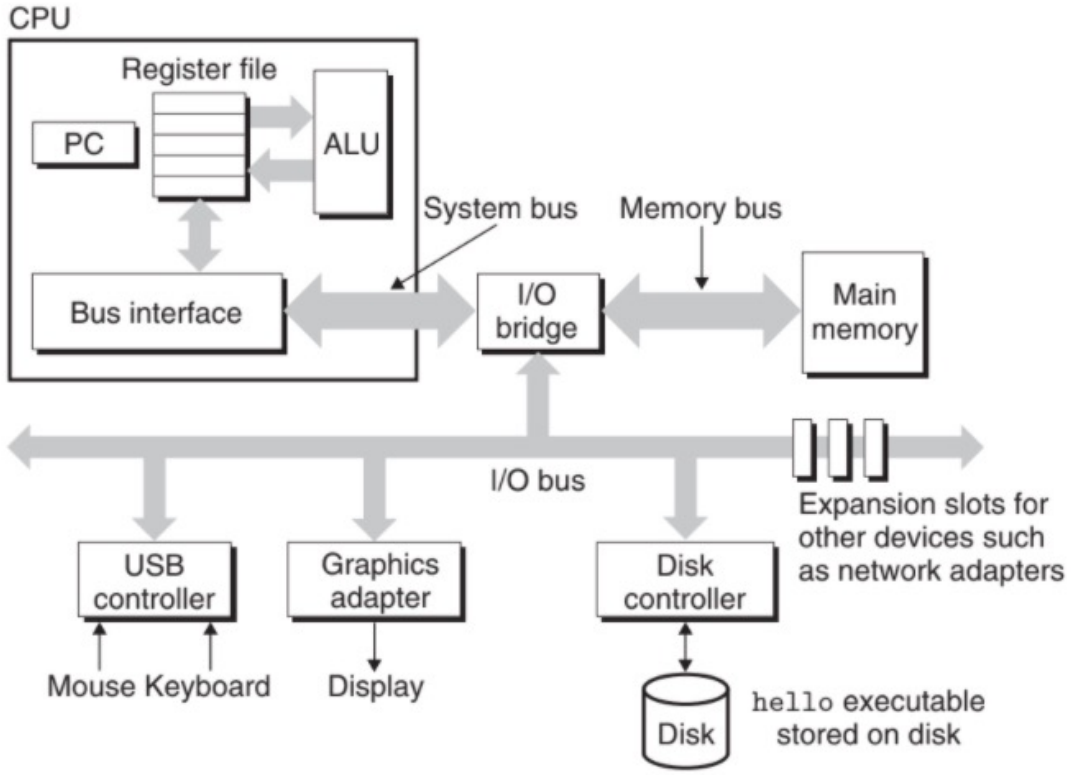
Course website: https://course.cse.ust.hk/comp4901q/

Outline

- Computer Architecture
- Parallel Computer Architecture
- Performance Measurement in Parallel Computing

Computer Architecture

- CPU
  - Central Processing Unit
  - Responsible for running programs
- Main Memory (or RAM)
  - Random access memory
  - Temporary storage for program and data
- Buses
  - Data movement
- I/O Devices
  - Disk: Long-term storage
  - Mouse/Keyboard/Display



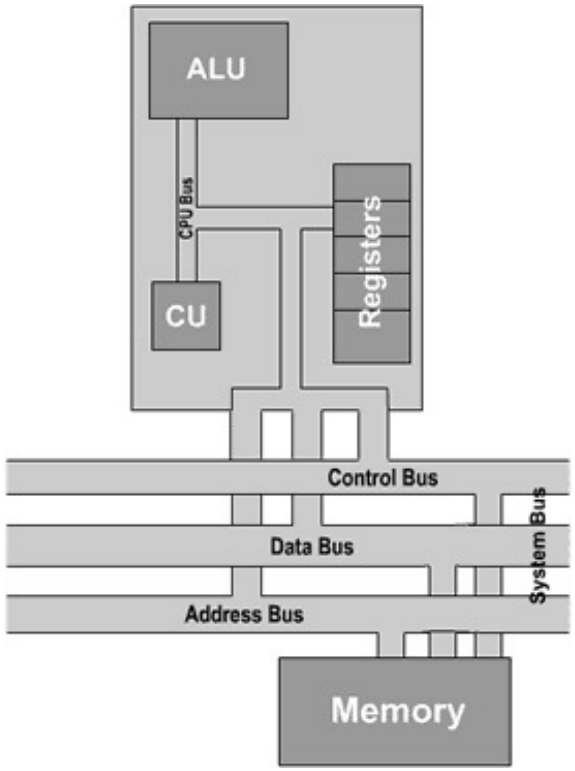
Hardware organization of a typical system [1]

[1] Bryant, Randal E., O'Hallaron David Richard, and O'Hallaron David Richard. Computer systems: a programmer's perspective. Vol. 2. Upper Saddle River: Prentice Hall, 2003.

The Processor

- Arithmetic logic unit (ALU)
  - Performs mathematical and logic operations
- Control unit (CU)
  - Directs the movement of instructions in and out of the processor
  - Sends control signals to the ALU
- Registers
  - Instruction register
  - Program counter (PC)
  - General purpose registers

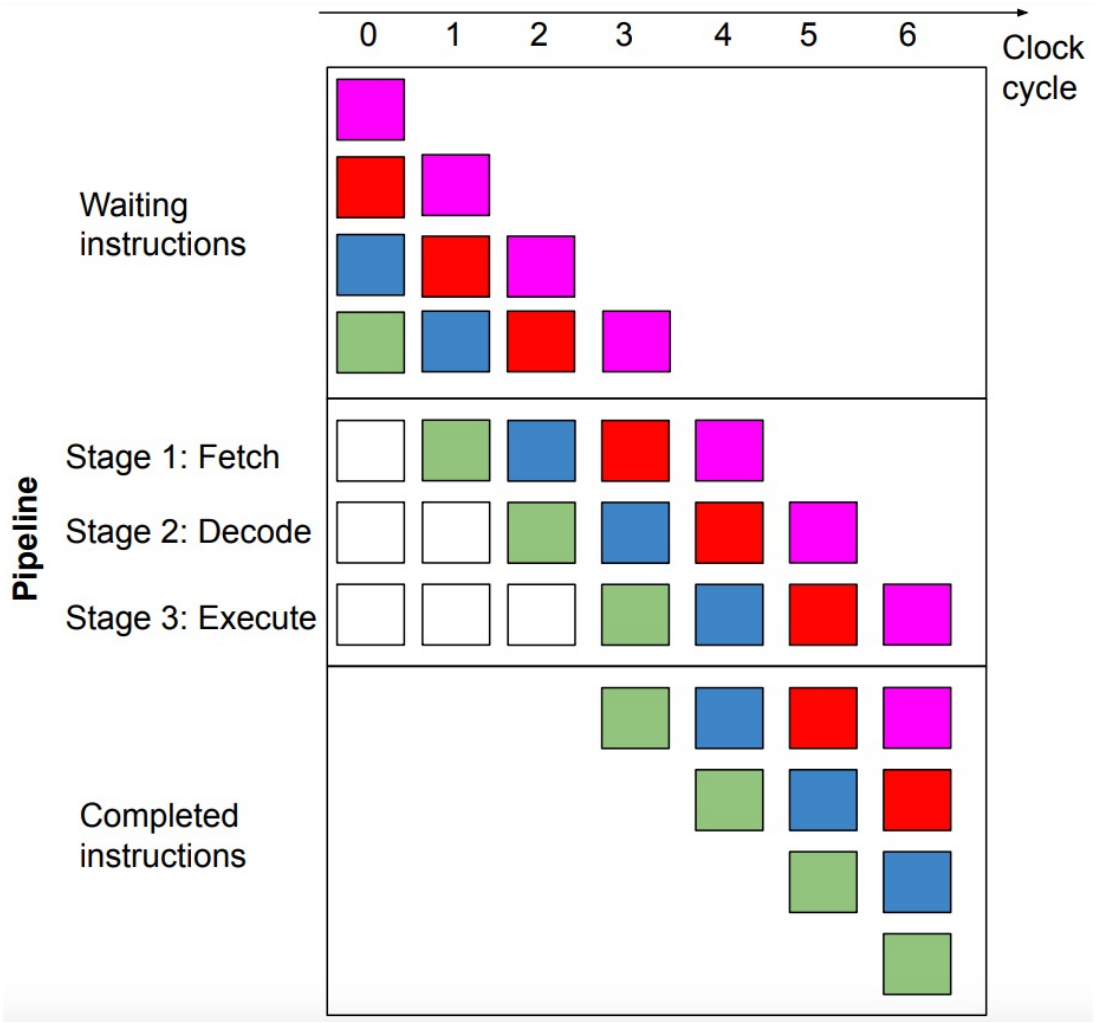
von Neumann architecture



Source: http://computerscience.chemeketa.edu/cs160Reader/ComputerArchitecture/Processor.html

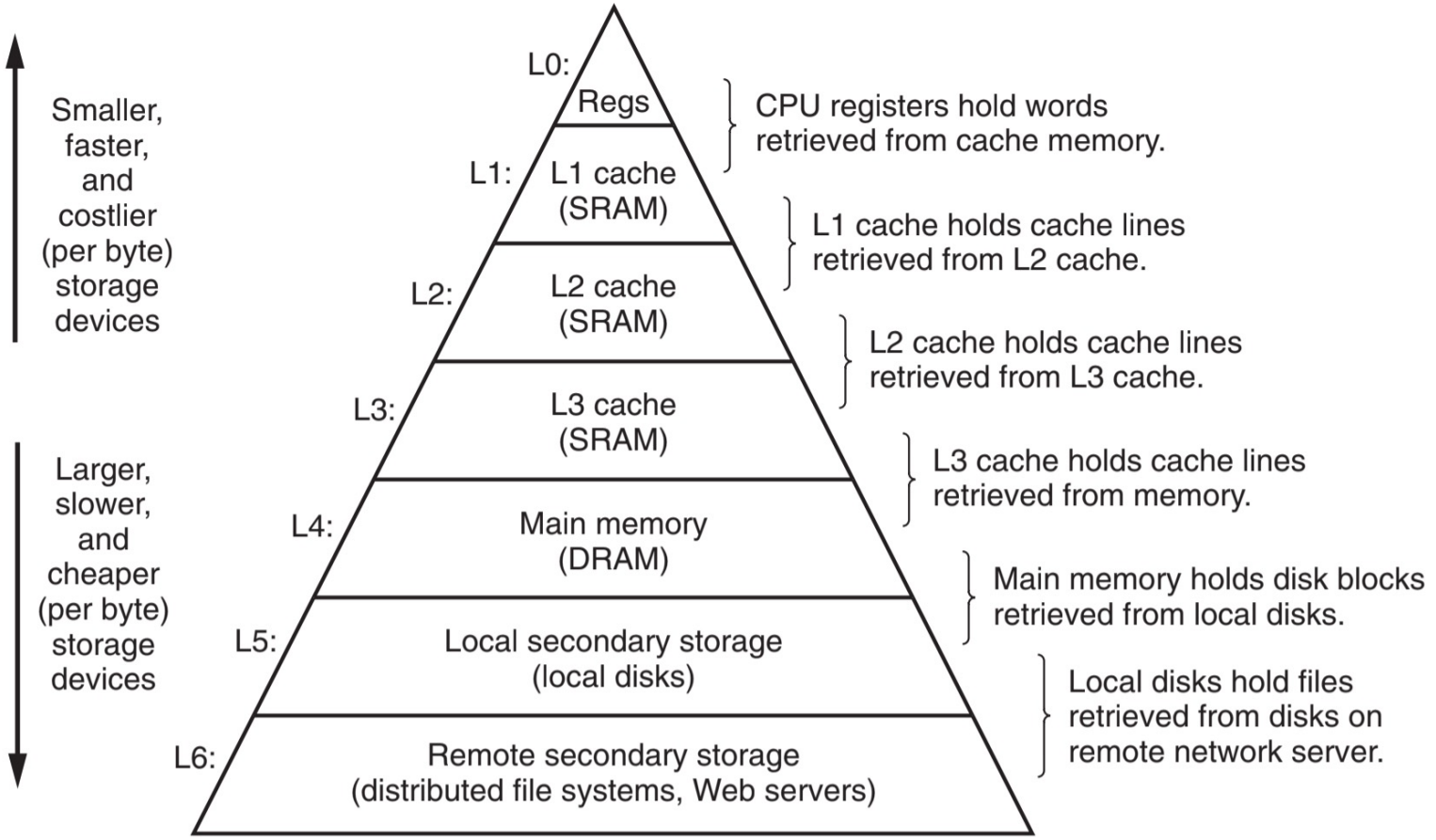
Instruction Cycle and Pipelining

- Each instruction has the following cycle
  - Fetch Stage
    - the next instruction (from PC) is fetched from the memory address
  - Decode Stage
    - instruction presented in the instruction register is interpreted by CU
  - Execute Stage
    - ALU to perform mathematical or logic functions
- Pipeline
  - Different stages from different instructions can be processed in parallel



Memory Hierarchy

- Keep processor busy by reducing data movement
- Fast storage is expensive
  - L0 (Registers): 1ns, KB
  - L1, L2, L3: 10ns, MB
  - Main memory: 100ns, GB
  - Disk: 10ms, TB
  - Remote: 10sec, PB
- DRAM
  - Double Data Rate (DDR)
  - High Bandwidth Memory (HBM)

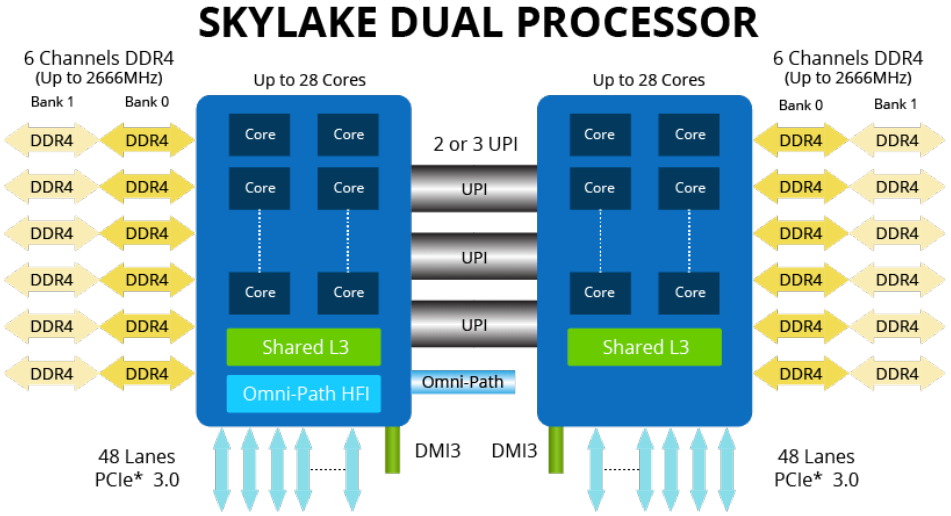


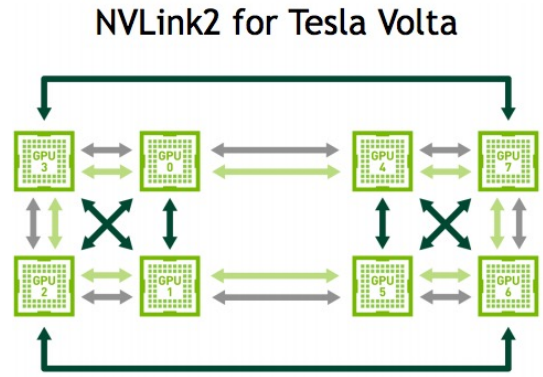
Memory hierarchy of modern computers [1]

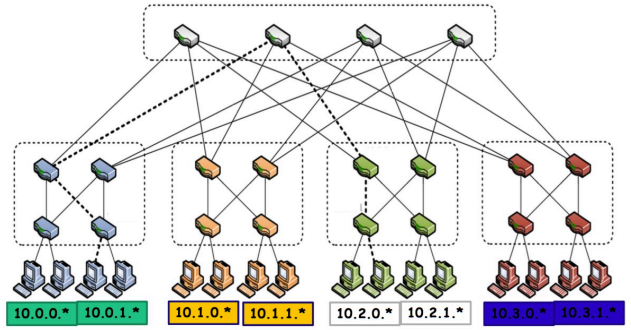
[1] Bryant, Randal E. and O'Hallaron David Richard. Computer systems: a programmer's perspective. Vol. 2. Upper Saddle River: Prentice Hall, 2003.

All Computers Become Parallel

- From mobile phones to supercomputers
  - Mobile phones: e.g., Apple A9, ARMv8-A dual-core
  - Desktop computer: e.g., Intel Core i3, 2 Cores
  - Servers
    - Multiple Cores per CPU: e.g., Intel(R) Xeon(R) Gold 6230 CPU: 20 Cores
    - Multiple CPUs: e.g., Intel(R) Xeon(R) Gold 6230, 20
      - QPI: Intel QuickPath Interconnect (Unidirectional Speed: 6.4 GT/s)
      - UPI (starting at 2017): Intel Ultra Path Interconnect (Unidirectional Speed: 10.4 GT/s)
    - Multiple GPUs
      - PCIe for GPU or other external devices: e.g., PCIe3.0x16 (8 GT/s per lane, ~1GB/s per lane)
      - NVLink for GPUs: e.g., NVLink 3.0 (can be up to 96 lanes) for Nvidia A100 GPUs (50 Gbit/s per lane)
  - Clusters
    - Multiple servers connected with high-speed interconnects (e.g., Mellanox 100 Gbit/s EDR InfiniBand ConnectX-5)
- Parallel Computers
  - Make use of multiple cores to finish one task







Parallel Hardware: Flynn's Taxonomy, 1966

- Michael J. Flynn: an American professor emeritus at Stanford University

SISD Single instruction stream Single data stream No parallelism!	(SIMD) Single instruction stream Multiple data stream Popular: SSE, AVX, GPU
MISD Multiple instruction stream Single data stream Uncommon	(MIMD) Multiple instruction stream Multiple data stream Popular: multi-core, cluster

SISD

- Both the instruction and data streams are sequentially executed
- Single control unit (CU) fetches single instruction stream (IS) from memory
- The CU then generates appropriate control signals to direct single processing element (PE) to operate on single data stream (DS)

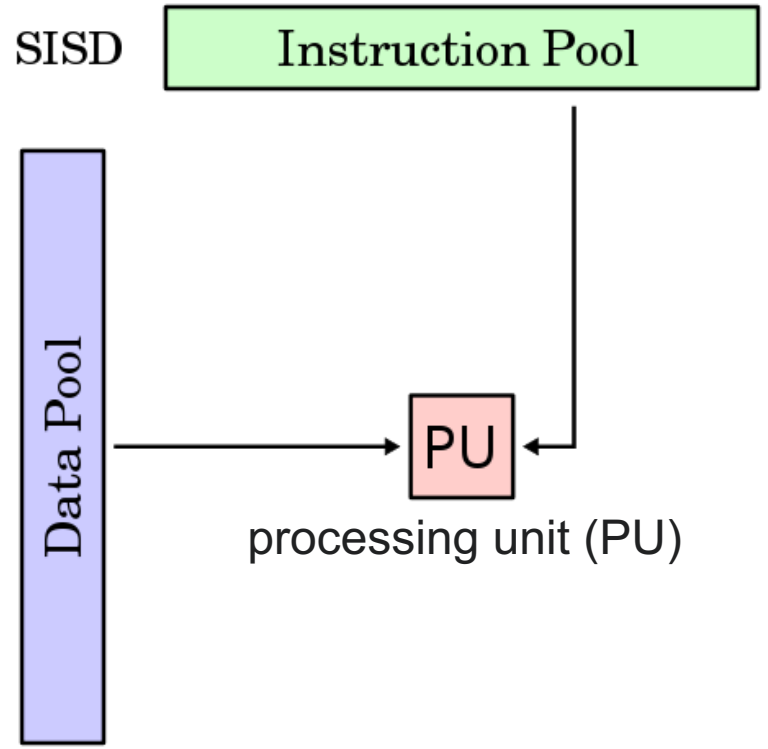


Image credit: https://en.wikipedia.org/wiki/Flynn%27s_taxonomy

SIMD

- Parallelism achieved by dividing data among the processors with data parallelism.
- Applies the same instruction to multiple data items.
- Examples of SIMD architectures
  - Intel x86 CPUs: MMX, SSE, and AVX
    - MMX: MultiMedia eXtension, introduced in 1997
      - A single instruction can be applied to two 32-bit integers, four 16-bit integers, or eight 8-bit integers at once
    - SSE: Streaming SIMD Extensions, since 1999
      - One SSE instruction can perform 4 single-precision or 2 double-precision operations
    - AVX (Advanced Vector Extensions), AVX2, AVX-512, since 2008
      - One AVX-256 instruction can perform 8 single-precision or 4 double-precision operations
      - One AVX-512 instruction can perform 16 single-precision or 8 double-precision operations
  - GPU: Graphics Processing Unit

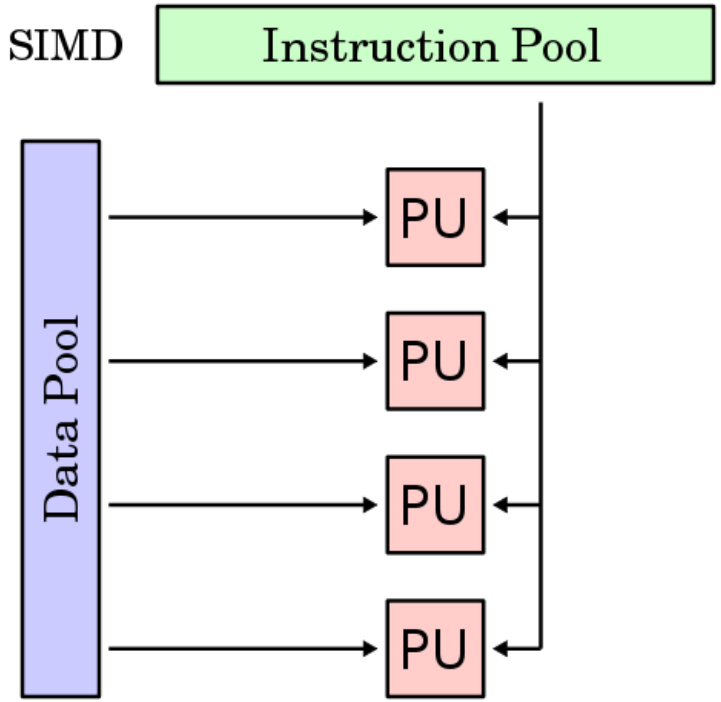
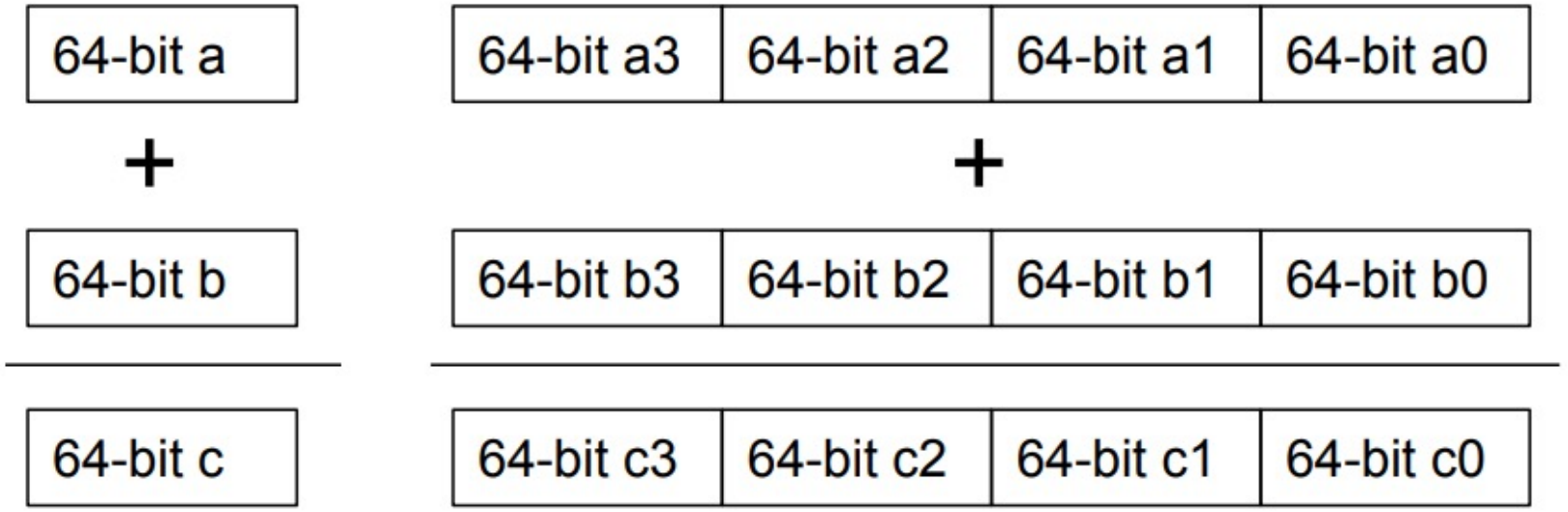


Image credit: https://en.wikipedia.org/wiki/Flynn%27s_taxonomy

Example of SISD vs. SIMD

- SISD
  - Traditional mode
  - One operation produces one result
- SIMD
  - E.g., Sandy Bridge: AVX (256 bits) or Cascade Lake: AVX-512 (512 bits)
  - One operation (256-bit AVX) produces 256/64 = 4 results (double-precision, 64bits)



MIMD

- Supports multiple simultaneous instruction streams operating on multiple data streams
- A collection of fully independent processing units or cores, each of which has its own control unit and its own ALU.
- Shared-memory systems
  - Each processor can access each memory location.
- Distributed-memory systems
  - Connected by a commodity interconnection network.

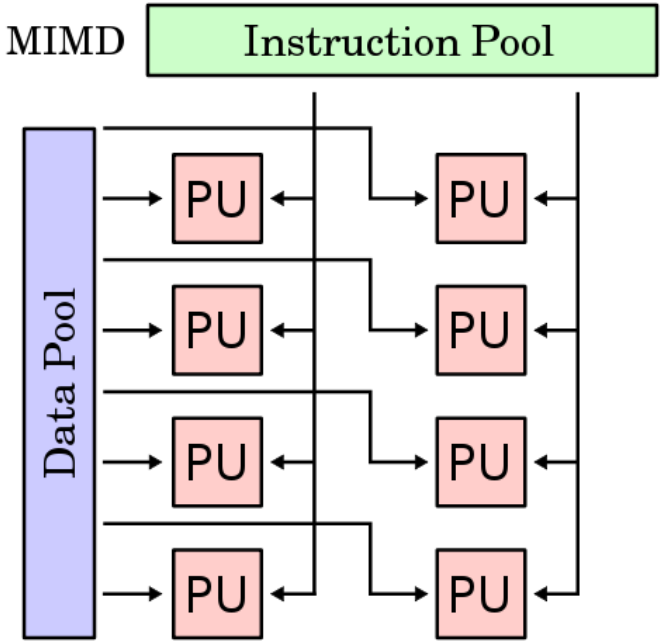


Image credit: https://en.wikipedia.org/wiki/Flynn%27s_taxonomy

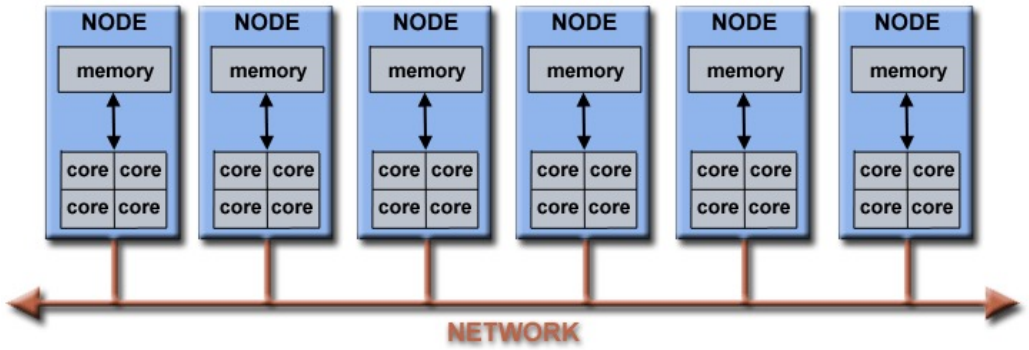


Image credit: https://hpc.llnl.gov/training/tutorials/introduction-parallel-computing-tutorial

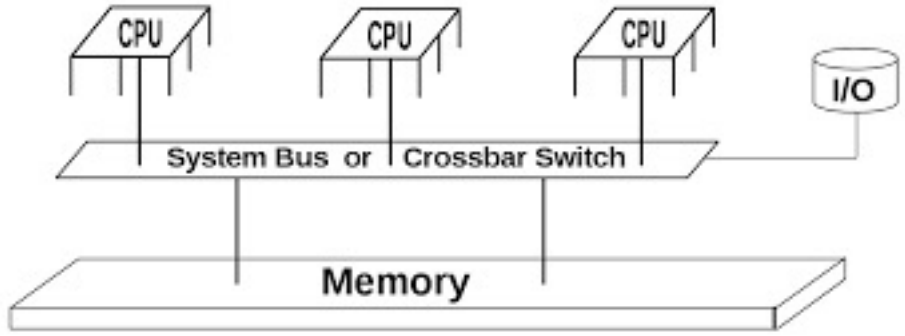
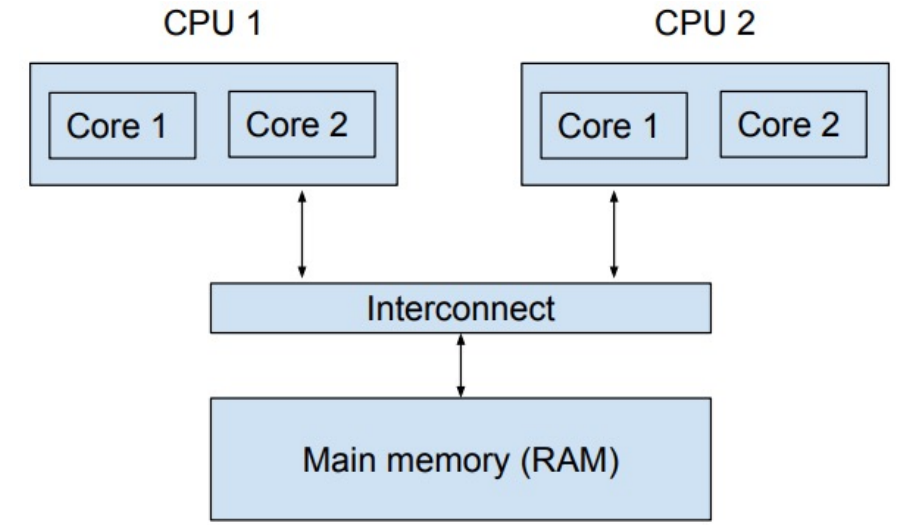


Image credit: https://en.wikipedia.org/wiki/File:Shared_memory.svg

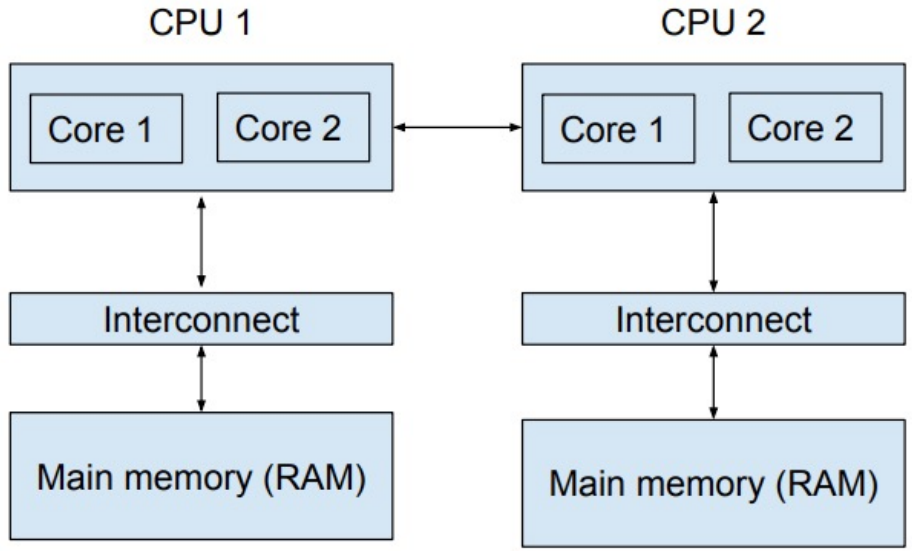
Shared-memory systems

- Uniform memory access (UMA)
  - all processors can directly access main memory
  - each processor has equal memory accessing time (latency) and access speed
  - E.g., Sun Starfire servers, Compaq alpha server and HP v series
  - less scalable
    - Increasingly difficult for hardware to provide shared memory behavior to increasing CPU cores

- Non-uniform memory access (NUMA)
  - the processors can access each others' blocks of main memory through special hardware (e.g., QPI, UPI)
  - the access time of the memory relies on the distance where the processor is placed
  - E.g., BBN, TC-2000, SGI Origin 3000, Cray



UMA



NUMA

Distributed-memory systems

- A set of processing nodes interconnected by a network
  - Components
    - Links
    - Switches
    - Network interface cards (NIC)
  - Network communication speed matters
    - Link speed
    - Routing
    - Network topology
      - Fat tree, Bcube, Torus, …

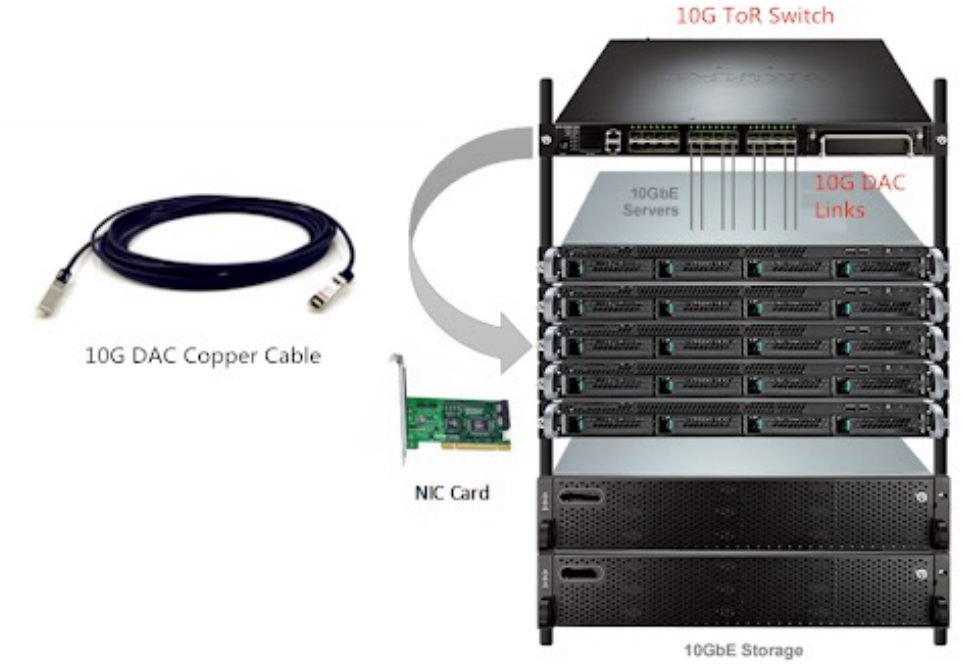
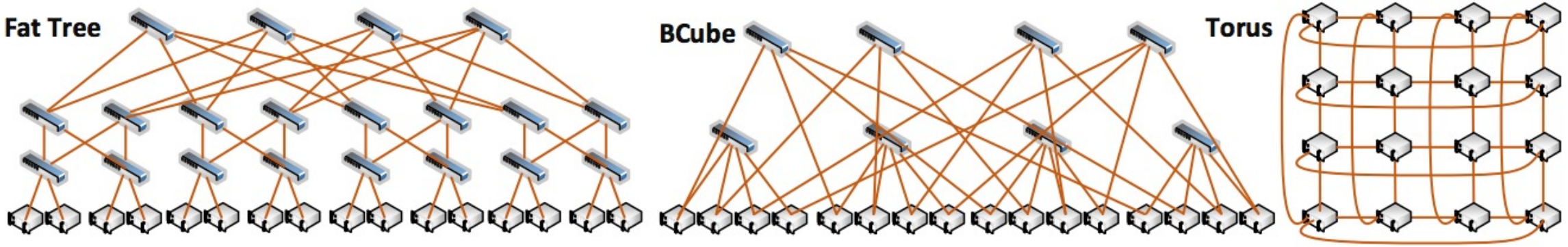


Image credit [1]



[1] http://www.cables-solutions.com/connectivity-options-comparison-10g-serversswitches-networking.html

Performance Measurement

- Define Performance (Effective FLOPS) = FLOP/Execution_Time
- "Processor X is *n* time faster than Processor Y on running a program"

$$\text{Performance}_X / \text{Performance}_Y$$
$$= \text{Execution\_time}_Y / \text{Execution\_time}_X = n$$

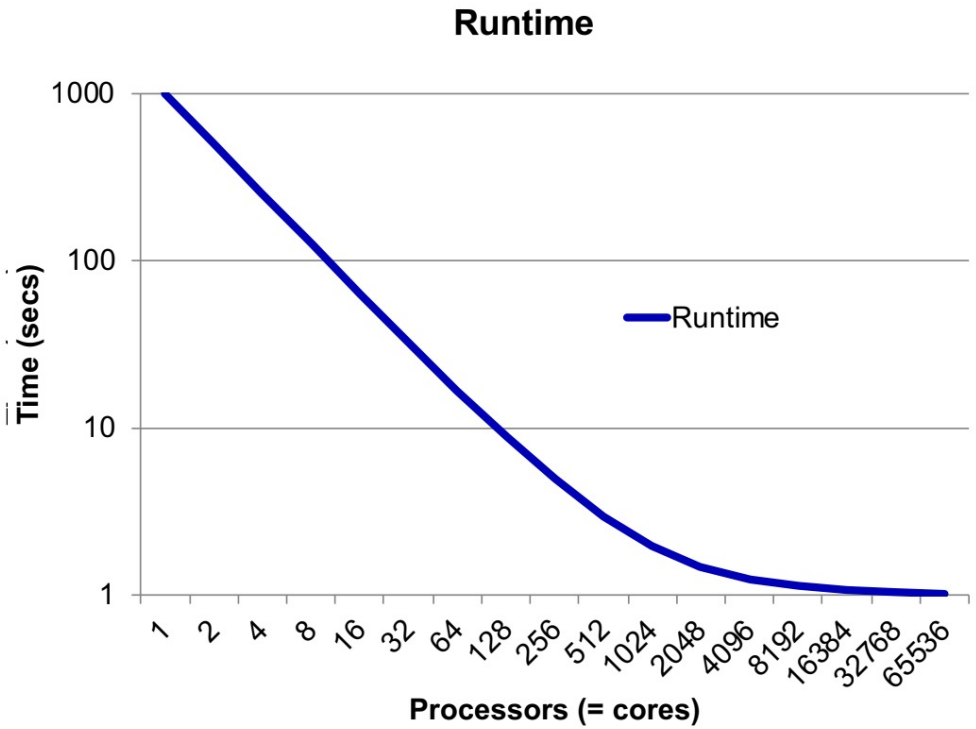
- Example: time taken to run a program
  - 10s on X, 15s on Y
  - Execution_Time_Y / Execution_Time_X = 15s / 10s = 1.5
  - So X is 1.5 times faster than Y

Goal of Parallelism

- Parallel program: instructions are executed in parallel by multiple processors (single server or clusters) to reduce the execution time of the program
  - Serial run-time = Tserial
  - Parallel run-time = Tparallel<Tserial

$$\text{speedup} = \frac{T_{serial}}{T_{parallel}}$$

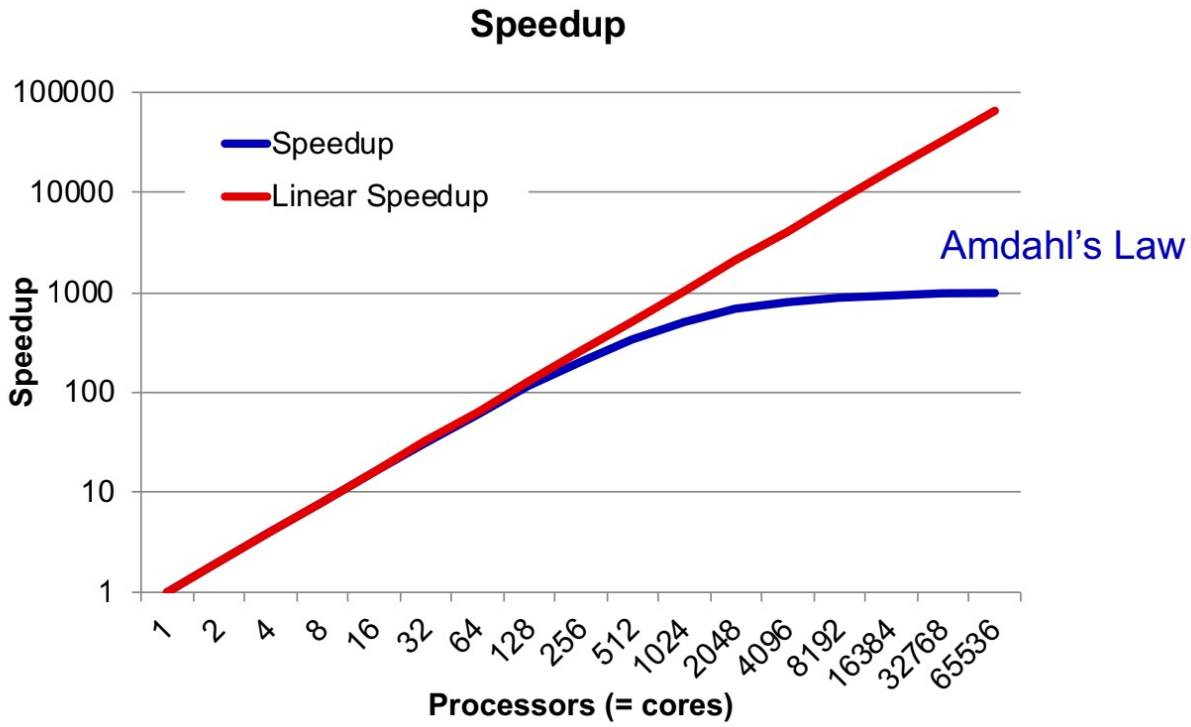
  - Ideal: linear scaling with increased number of cores
  - Fact: limited by Amdahl's Law



Amdahl's Law

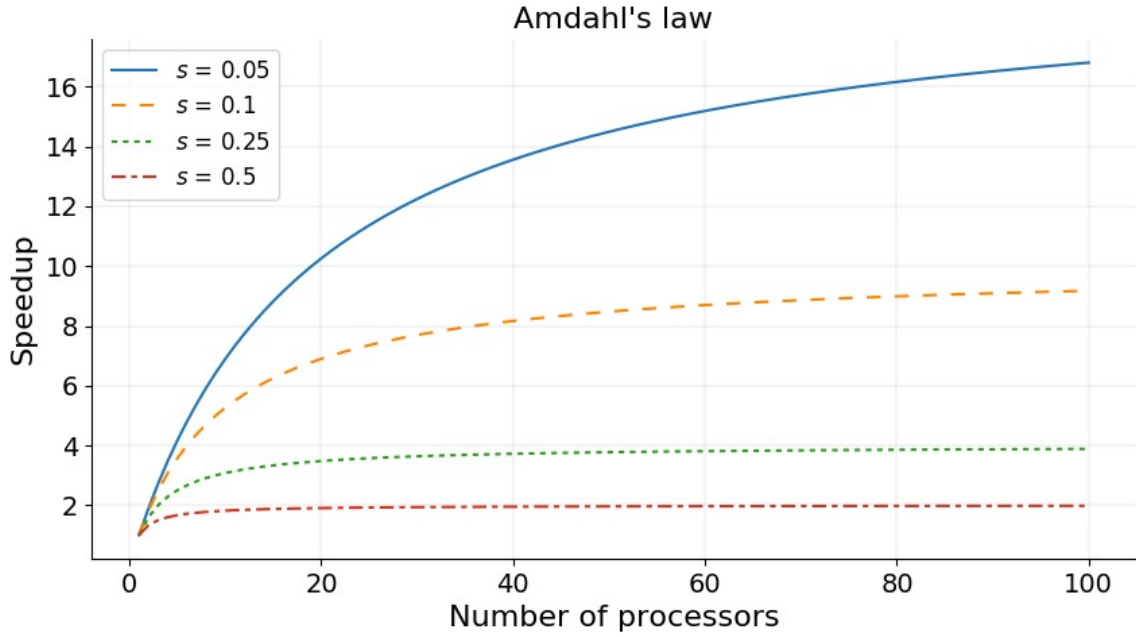
- Any computing task involves some part that can be parallelized (Tp denotes its time), and some part that cannot be parallelized (Ts denotes its time).
- The theoretical speedup is limited by the part of the task that cannot benefit from parallelism.
- For a serial program:
  - Tserial = Tp + Ts=(1-s)+(s),
  - s is the Amdahl fraction
    - fraction of work done sequentially
- It is parallelized with p processors
  - Tparallel ≥ Tp / p + Ts=(1-s)/p+(s)
- speedup = Tserial / Tparallel

$$\leq 1/((1-s)/p+(s)) \leq 1/s$$



Strong-scaling vs. Weak-scaling

- Strong-scaling
  - The total problem size (W) is fixed for p processors
  - Every processor has the problem size: W/p
  - Limited by Amdahl's law
    - speedup ≤ 1/s
- Weak-scaling
  - The total problem size is scaled for p processors: W × p
  - Every processor has the same problem size: W
  - Limited by Gustafson's law
    - speedup ≤ s + (1-s) × p



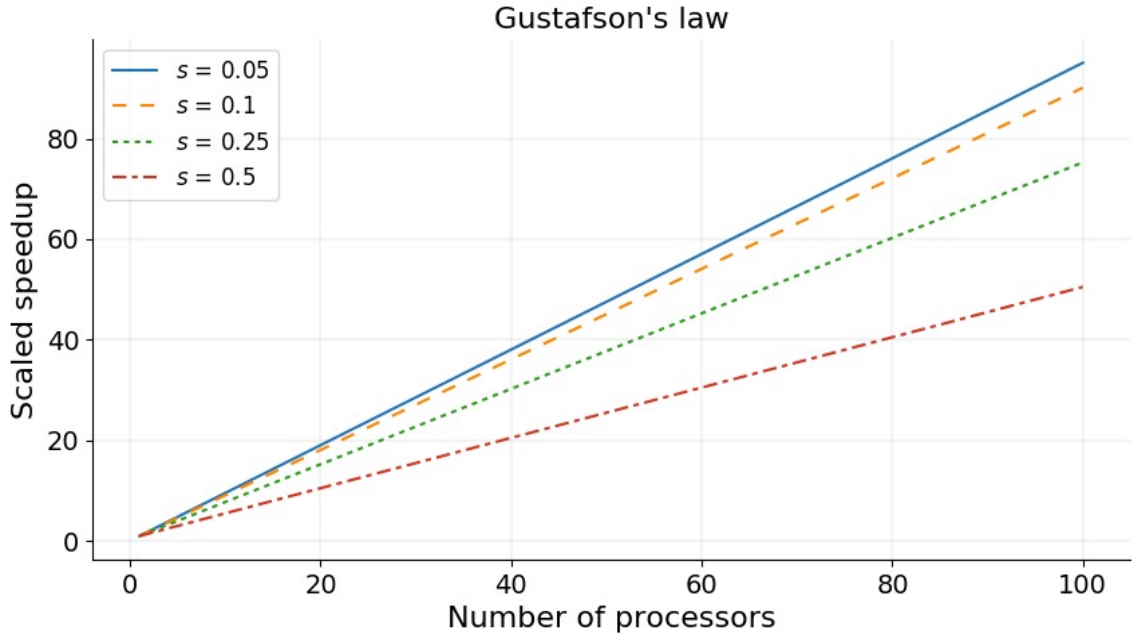


Image credit: http://kth.se/blogs/pdc/2018/11/scalability-strong-and-weak-scaling

Scaling Efficiency

- Consider a parallel system using p processors, and achieves speedup of

$$S = T_{serial} / T_{parallel}$$

- Scaling Efficiency = S/p : it measures how efficient a parallel solution is.
  - For example, by using 4 CPU cores, we decrease the running time from 1 hour to 20 minutes. Then the efficiency is 75%.

- Linear scaling => Scaling Efficiency=100%.

Reading List

- Thomas Sterling, Matthew Anderson and Maciej Brodowicz (2018), “High Performance Computing: Modern Systems and Practices,” Morgan Kaufmann, **Chapter 2**. [PDF: https://www.sciencedirect.com/book/9780124201583/high-performance-computing]

- Duncan, R. (1990). “A survey of parallel computer architectures”. *Computer*, *23*(2), 5-16. [PDF: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=44900]

Summary

- Computer architecture
  - CPU
  - Memory
  - Instruction cycle
  - Instruction pipelining
- Parallel computer architecture
  - Flynn's Taxonomy
    - SISD, SIMD, MISD, and MIMD
  - Shared-memory and distributed-memory systems
  - Network topology
- Performance measurement in parallel computing
  - Speedup
  - Amdahl's law and Gustafson's law