24 Advanced Processors Organization
CSC 230

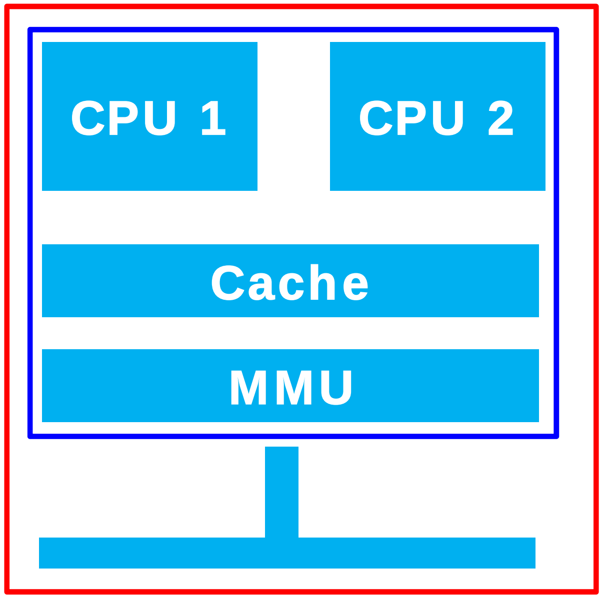
Department of Computer Science
University of Victoria

M&H: 10.1 to 10.4 (not everything!)

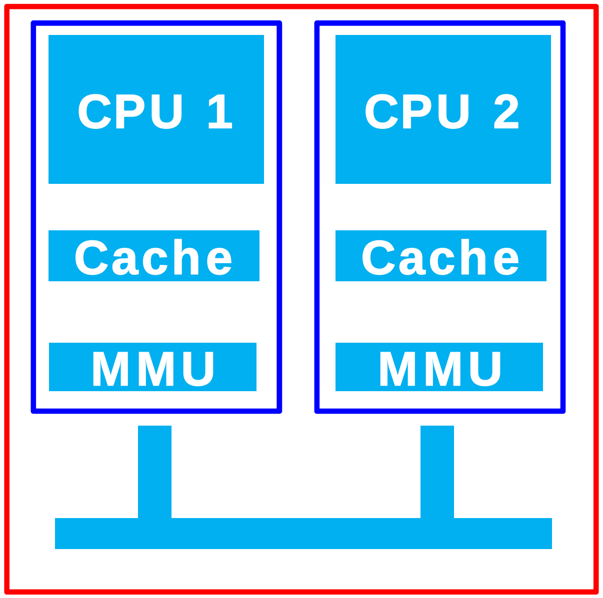
Stallings: 2.6, 18.1, 18.2, 18.3, 18.5 (not everything!)

Multicore Computers *(seen before)*

- Two or more cores (CPUs) as a single integrated circuit.
- May have independent or shared on-board caches.
- Each core independently implements optimizations such as superscalar execution and pipelining.
- Number of cores is 2 for Intel Duo, 4 for Intel Core 2 Quad, 8 for PS3, … 32 for Intel Larrabee.

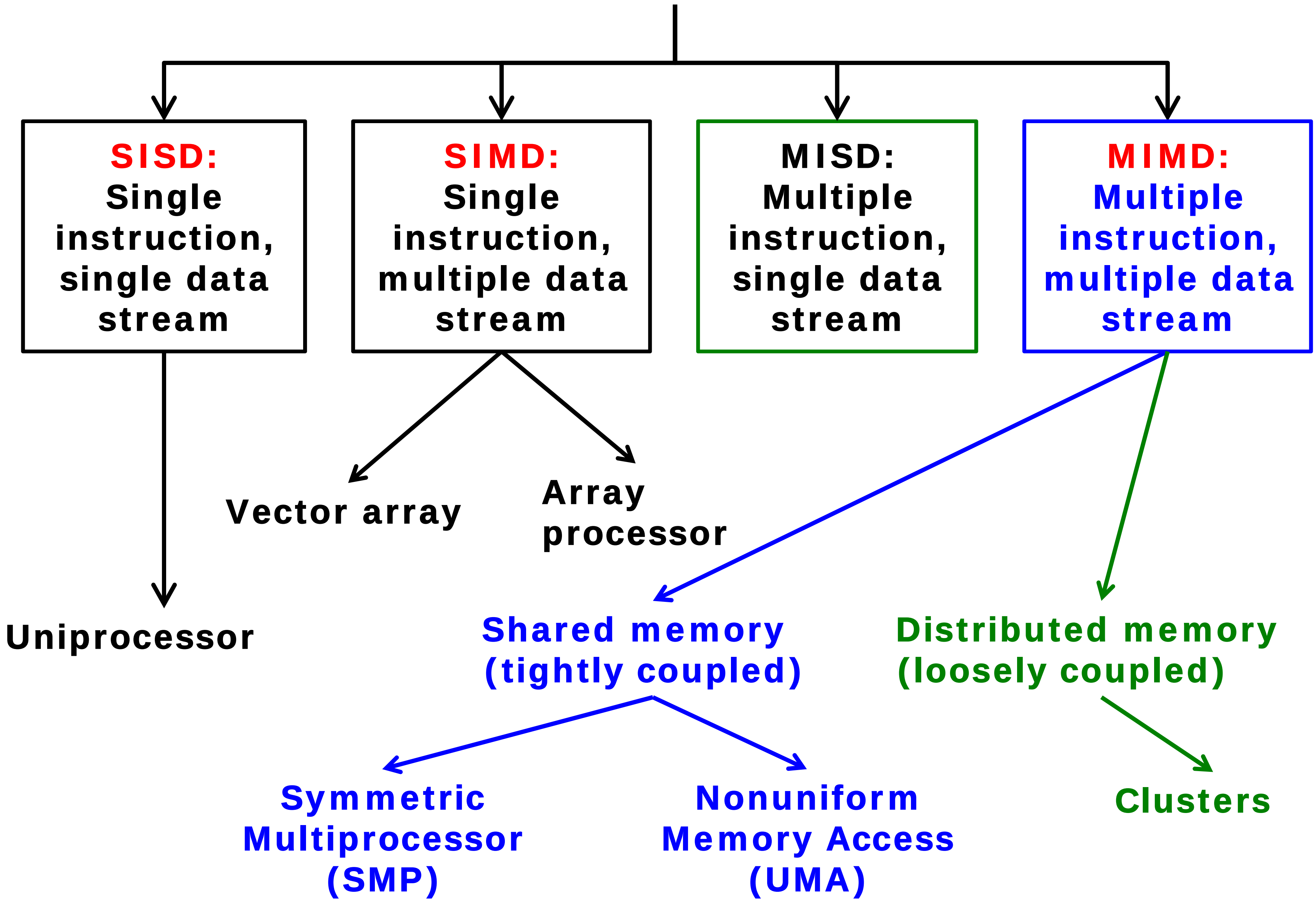


Multicore Design



Multiprocessor Design

Processor Organization: the Flynn Taxonomy



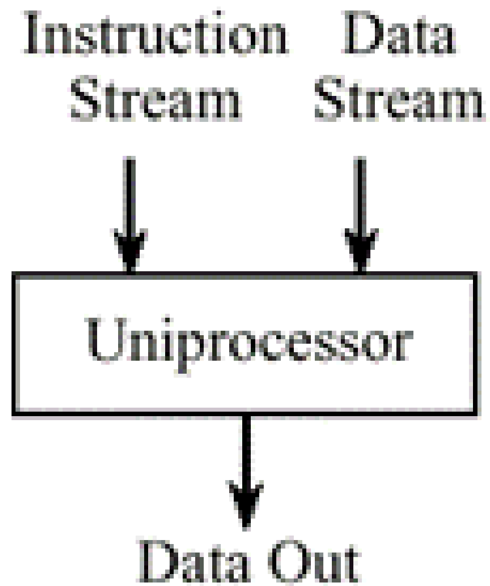
Flynn Taxonomy: Classification of architectures

(a) SISD = Single instruction, single data stream

(b) SIMD;

(c) MISD;

(d) MIMD.



(a)

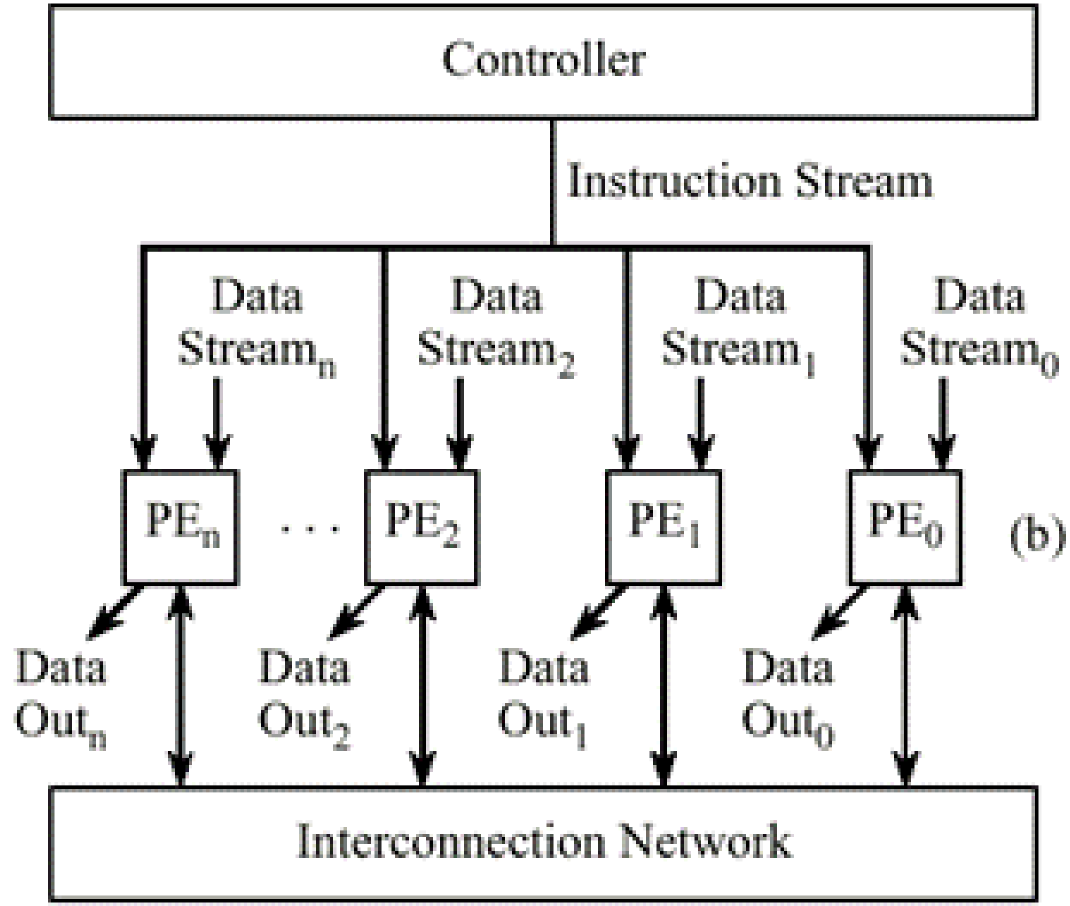
Flynn Taxonomy: Classification of architectures

(a) SISD;

(b) SIMD= Single instruction, multiple data

(c) MISD;

(d) MIMD.



Flynn Taxonomy: Classification of architectures

Not used much at all

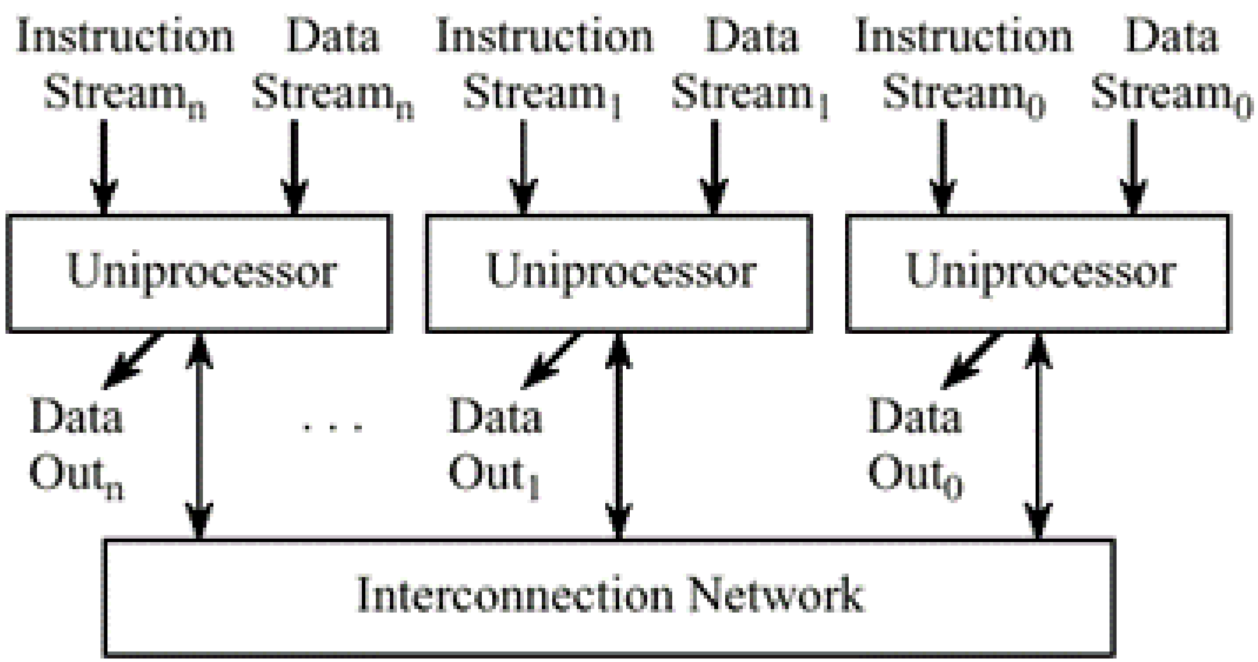
(a) SISD;

(b) SIMD;

(c) MISD= Multiple instruction, single data stream

(d) MIMD.

(c)



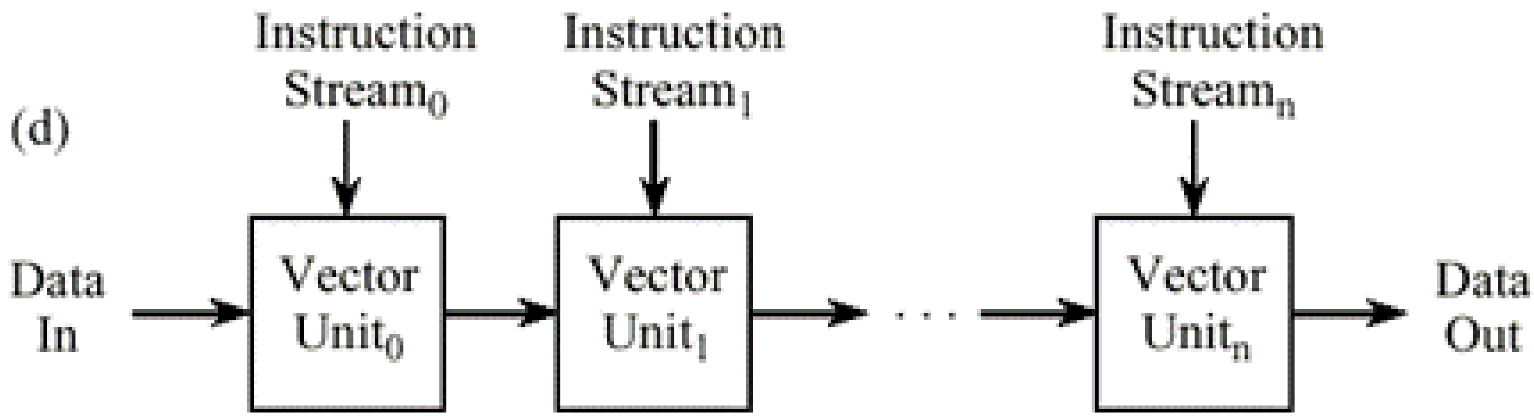
Flynn Taxonomy: Classification of architectures

(a) SISD;

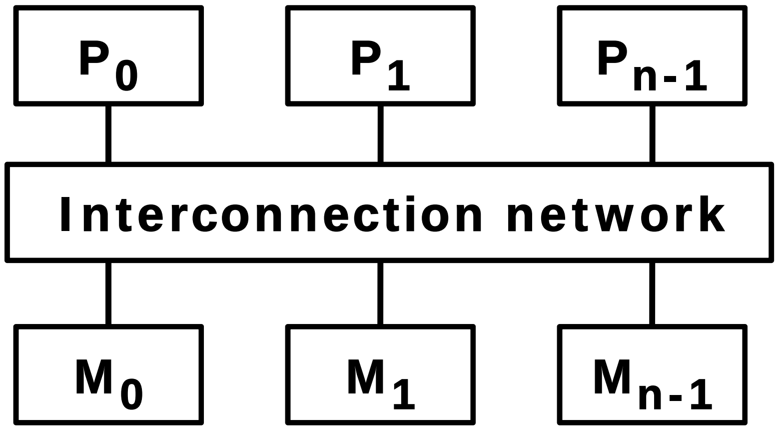
(b) SIMD;

(c) MISD;

(d) MIMD = Multiple instruction, multiple data stream

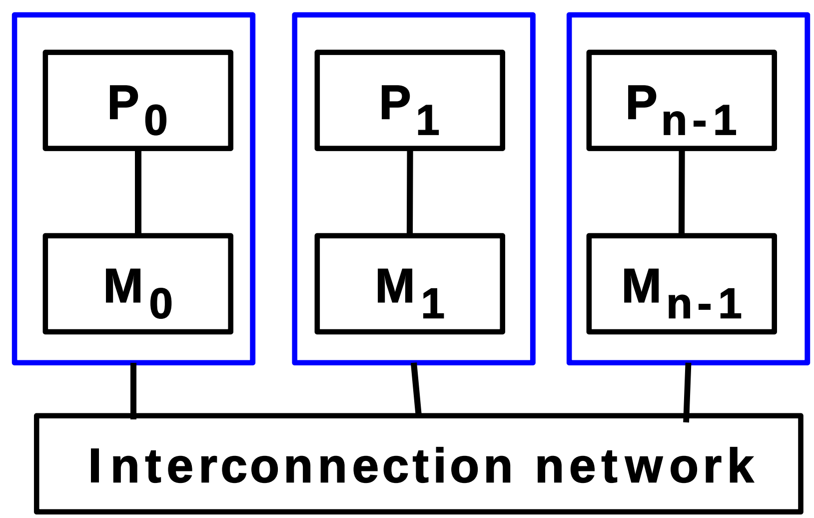


Shared Memory or Message Passing?



shared memory

message passing



Why SIMD?

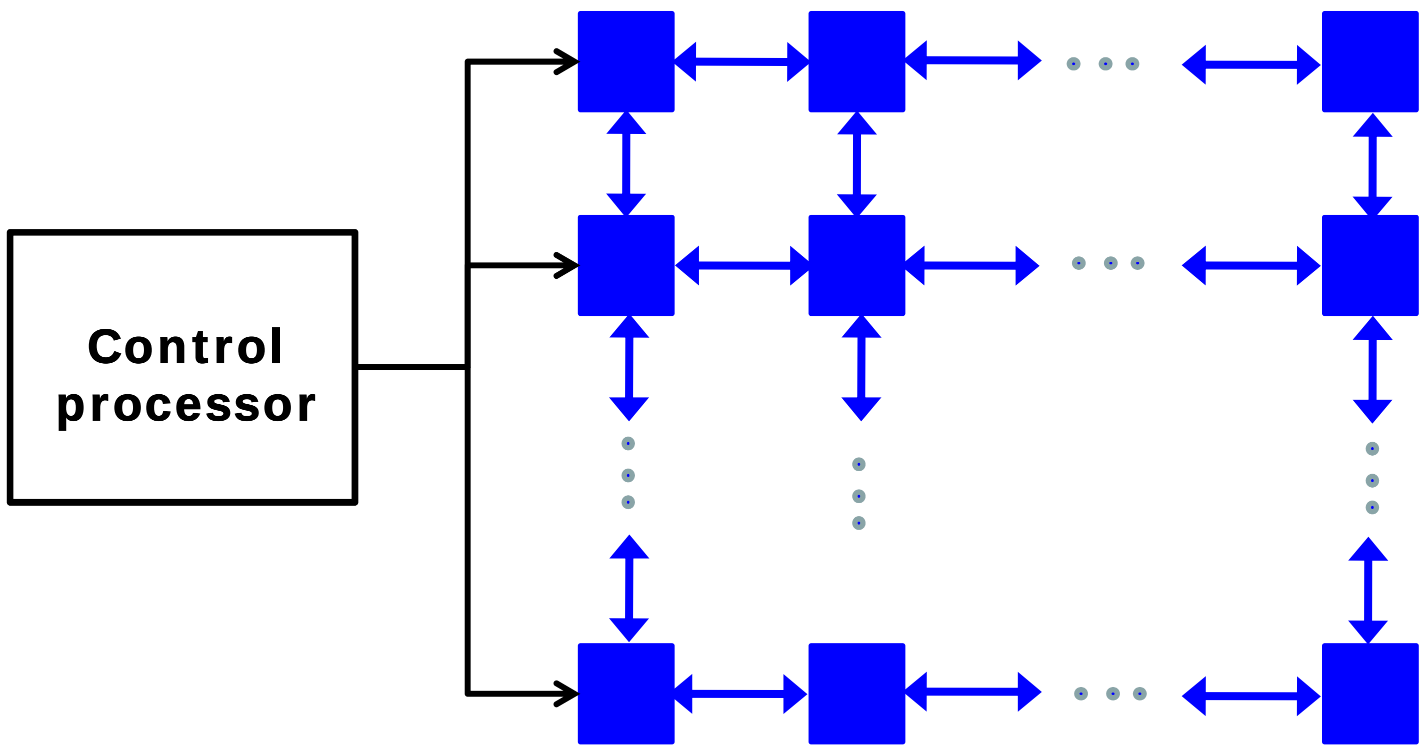
SIMD = Single Instruction / Multiple Data

Multimedia and communication applications often use repetitive loops

> **While occupying < 10 % of the overall application code, they can account for up to 90 % of the execution time.**

SIMD enables one instruction to perform the same function on multiple pieces of data.

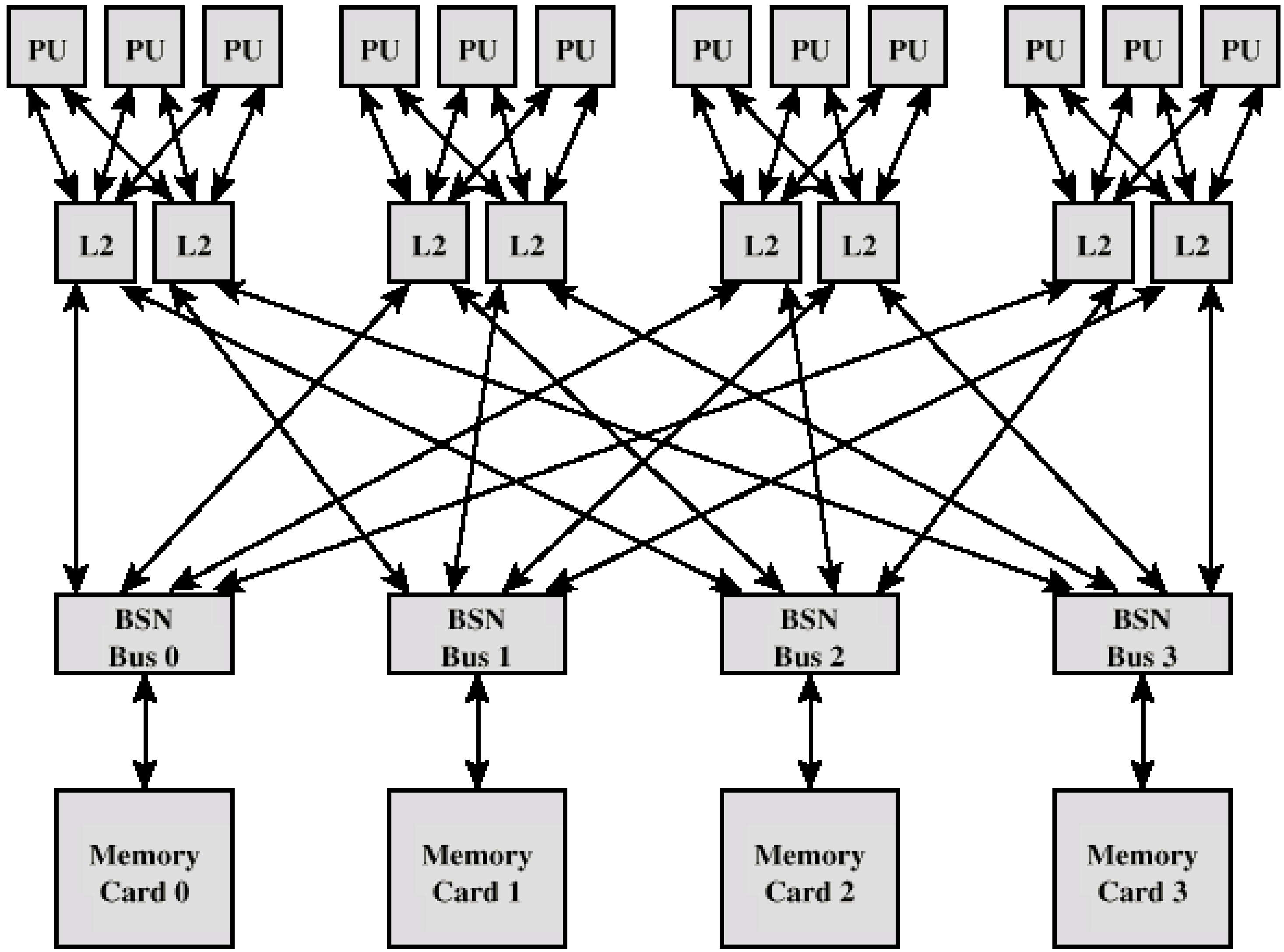
An array processor



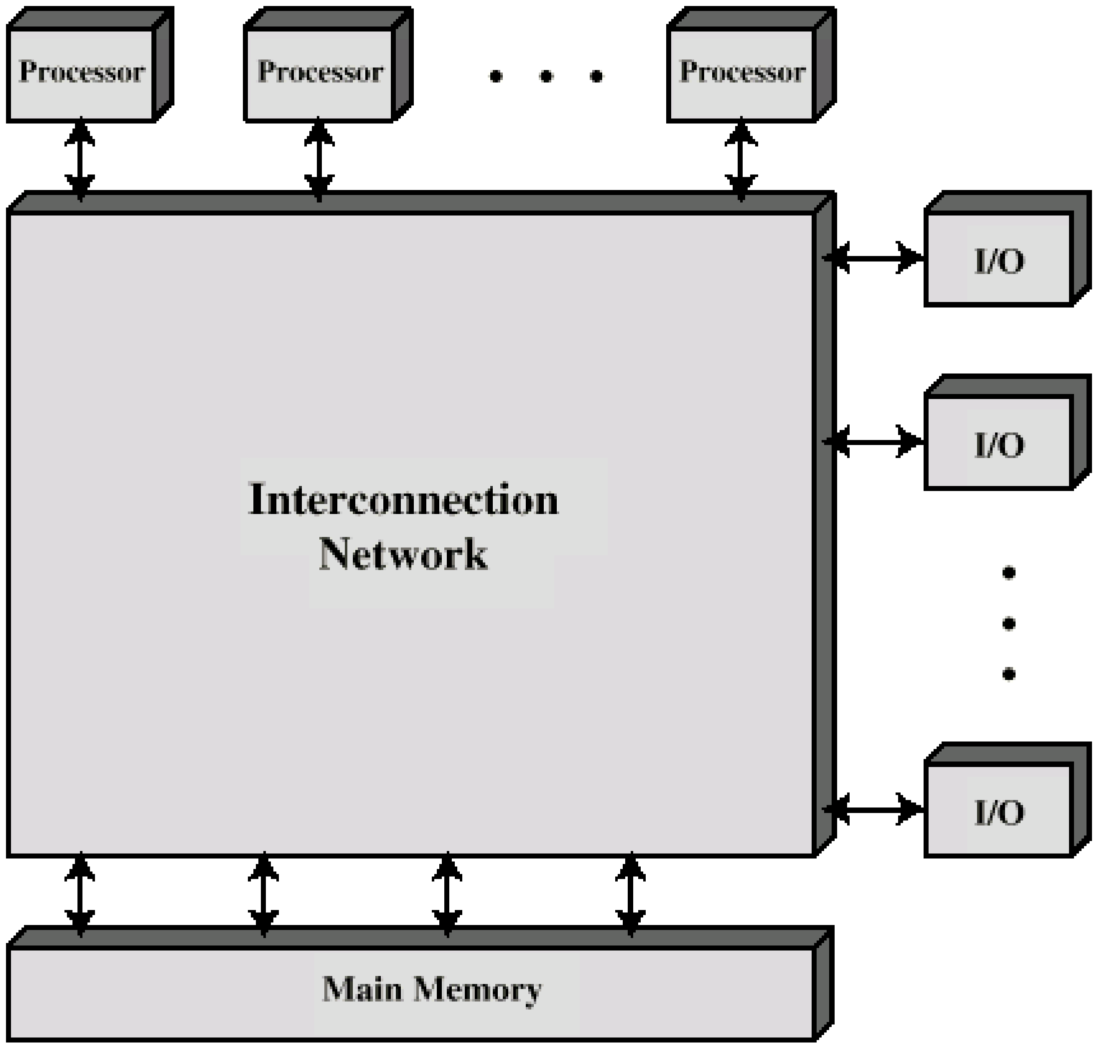
Grid of processing elements

MIMD: Symmetric Multiprocessor (SMP)

IBM 390 SMP

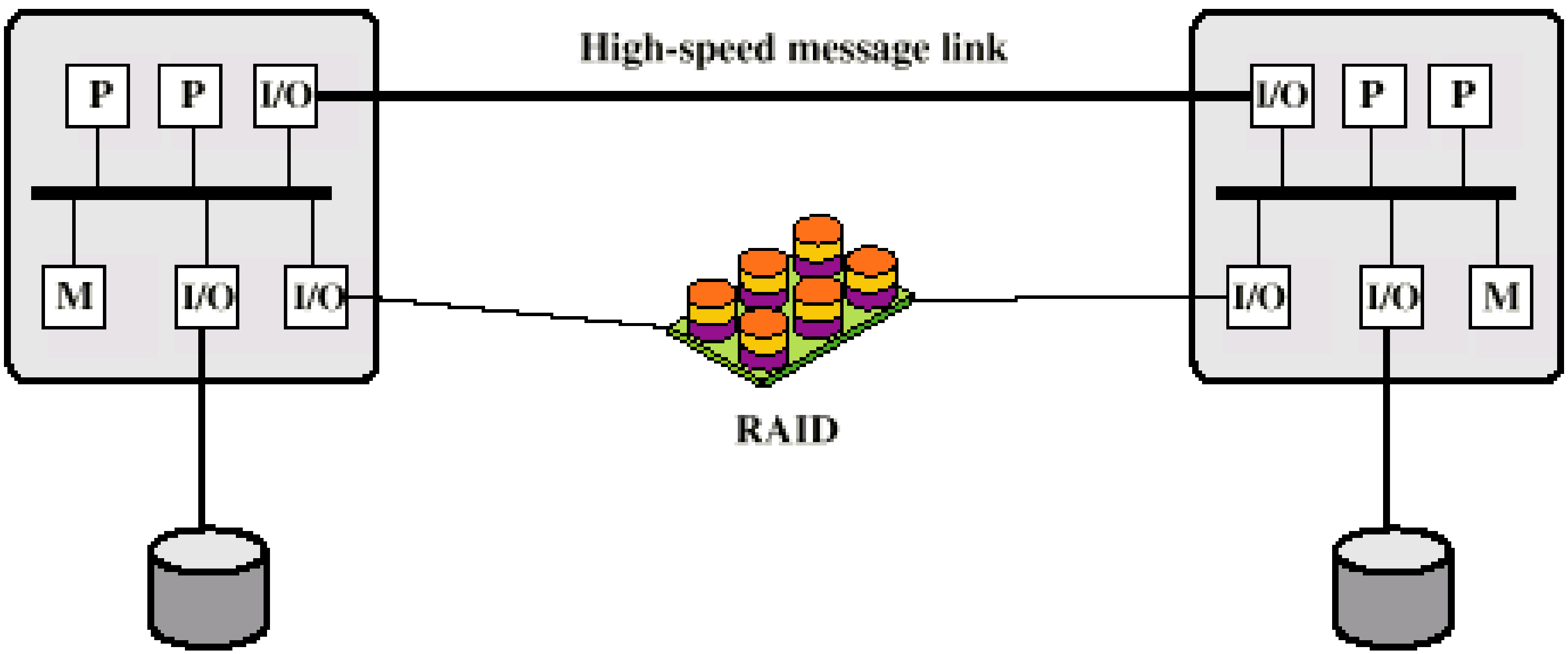


MIMD: Tightly Coupled Multi-Processor
(e.g. MINERVA)



MIMD: Distributed Memory (loosely coupled)

Cluster Configurations



Some Definitions.1

- ***Parallel processing*:**

  the *simultaneous execution* of instructions, as in:

  - **two or more sequences of instructions or one sequence of instructions operating on two or more sets of data, by a computer having multiple arithmetic or logic units or both**

- ***Multiprocessing*:**
  - **the *simultaneous execution* of two or more programs or sequences of instructions by a machine consisting of *two or more processors***

- ***Massively parallel processing*:**
  - **computers with more than 100 processors**

Some Definitions.2

- ***Processor farm*:**
  - ***a parallel machine consisting of multiple processors*, where tasks are distributed (or "farmed out") by one farmer to several worker processors which send results back to the farmer (processor farms are suited for applications that can be partitioned into several independent tasks); also: “master/slave”.**

- ***Tightly coupled multiprocessing systems*:**
  - **have a high degree of interactions among the processors or whole computers**

- ***Loosely coupled multiprocessing systems*:**
  - **have a low degree of interactions among the processors or whole computers**

The Multiprocessor Challenge

Find applications that can take advantage of many processors

1. Success has been made through developing a parallel subsystem that presents a sequential interface
   - Examples: databases, file servers, CAD packages, multiprocessing operating systems

2. Some key questions that drive multiprocessors design:
   - ➔ How do parallel processors share data?
   - ➔ How do parallel processors coordinate?
   - ➔ How many processors?

3. The three technologies available to parallel processor designers :
   - fast microprocessors
   - high-capacity DRAMs (capacity)
   - increasing network bandwidth

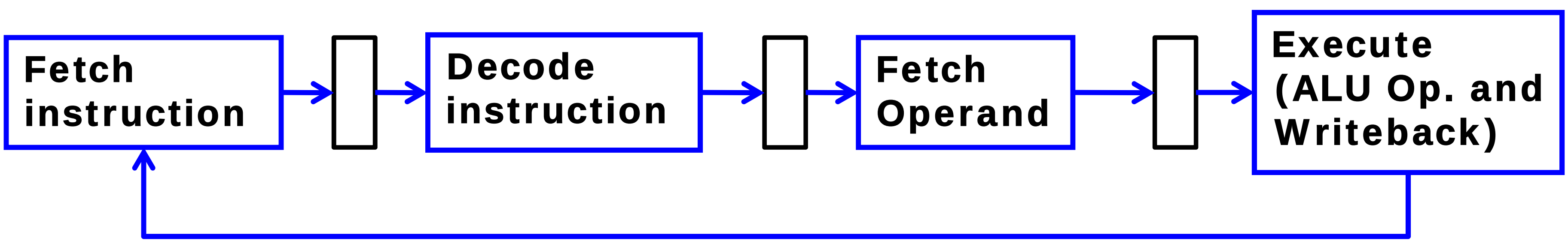
Parallel processing Characteristics and Performance

3 main parameters:

- **Number of processing elements (PEs)**
- **Interconnection network among (PEs)**
- **Memory organization**

Remember example from a 4-stage instruction pipeline:

- ***4 PEs***
- ***interconnection network is a ring***
- ***memory is external RAM***



What about Performance?

4 main measures:

- **Parallel Time** = absolute time to execute on a parallel processor
- **Speedup** = ratio

$$S_{peedup} = \frac{T_{sequential}}{T_{parallel}}$$

- **Efficiency** = ratio of speedup to number of PEs
- **Throughput** = how much computation done over time

Speedup (with parallelism)

Sequential and Parallel algorithms are different and are coded differently!

- **Want speed up of 100, then use 100 PEs ➔ NO**
- **Must decompose the portions of the computation between the sequential and the parallel parts, then analyze**

Amdahl's Law

$$S_{peedup} = \frac{1}{f + \frac{1-f}{p}}$$

sequential portion

# of PEs

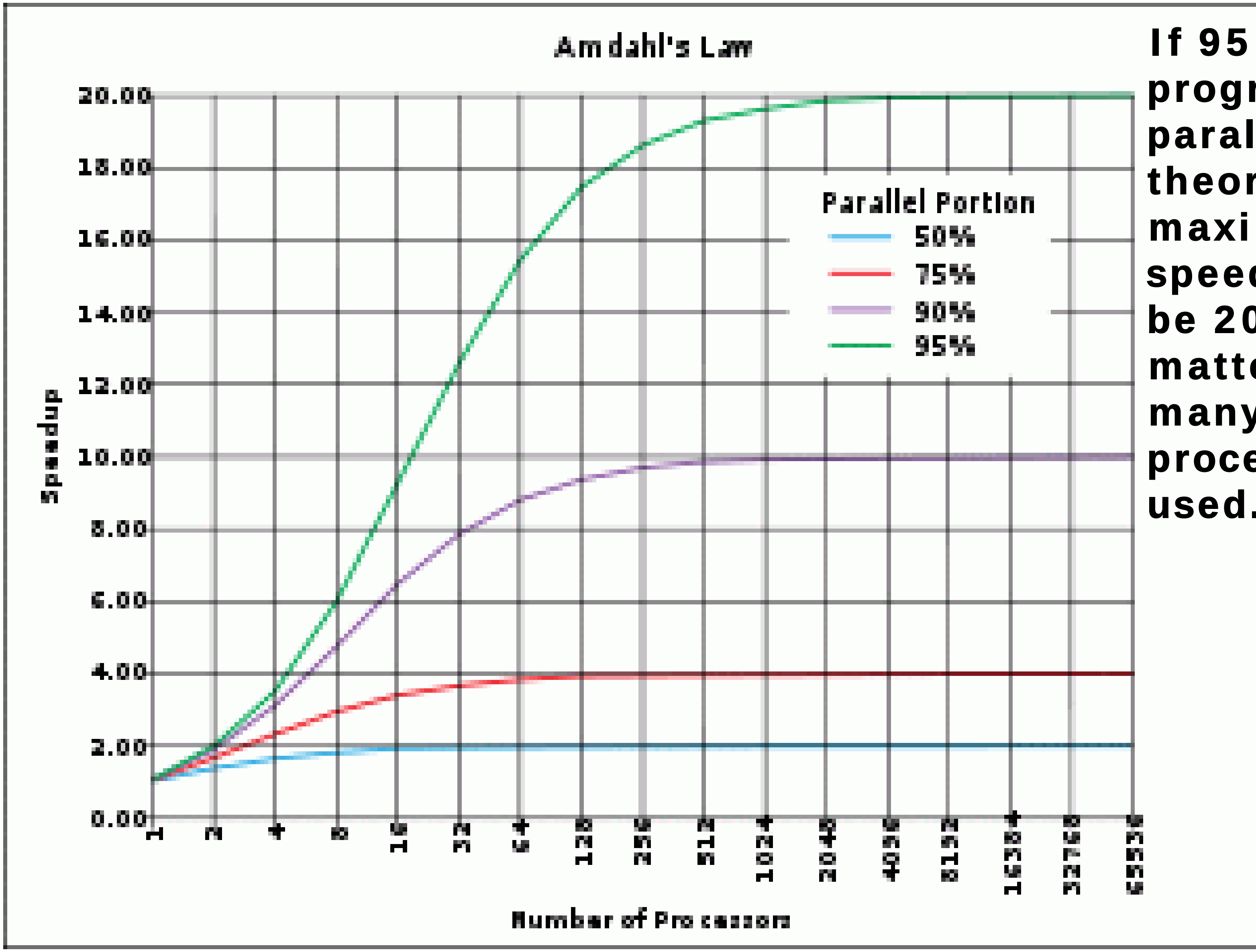
Ex. 1: Let f = 20% and PEs = 20

$$S = \frac{1}{0.2 + \frac{0.8}{20}} = 4.1$$

Ex. 2: Let f = 20% and PEs = infinite

$$S = \frac{1}{0.2 + \frac{0.8}{\infty}} = 5$$

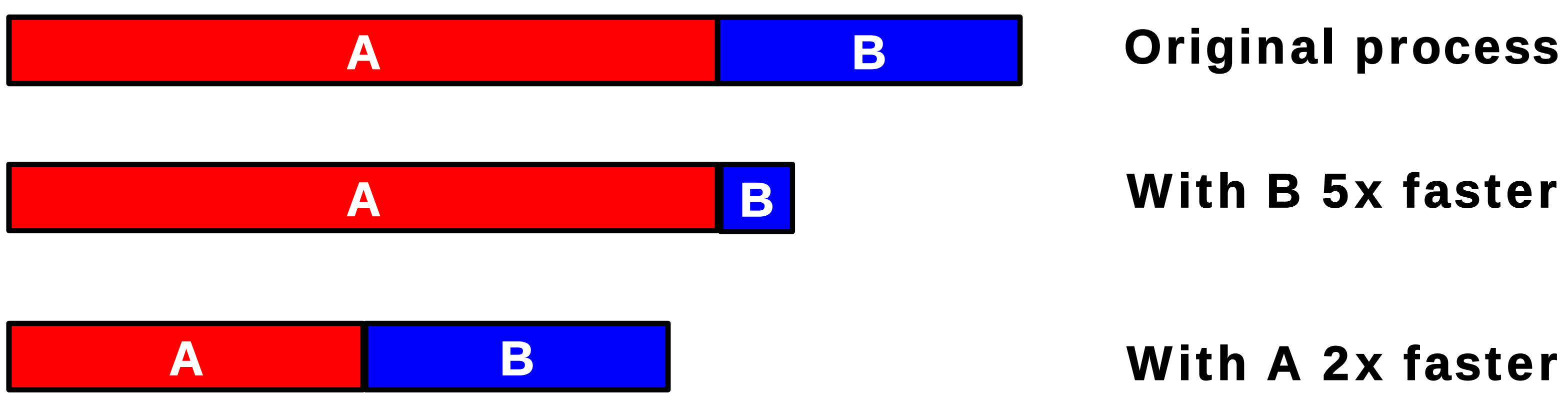
From Wikipedia (Amdahl's law)



If 95% of the program can be parallelized, the theoretical maximum speedup would be 20, no matter how many processors are used.

From Wikipedia (Amdahl's law)

- Consider a task with two independent parts, A and B.
- B takes roughly 25% of the time.
  - With *hard* work, one may accelerate B to 5 times faster
    - ➔Overall time reduced by little.
  - With less work, one may accelerate A to 2 times faster
    - ➔Overall time reduced by a lot
  - Different possible speedup of A and B do not give the whole answer



Efficiency (with parallelism)

$$efficiency = \frac{speedup}{\#\ of\ PEs}$$

From previous example:

Speedup = 4.1

 # PEs = 20

➔ Efficiency = 4.1 / 20 = 0.205 or 20.5%

Consider the following:

If #PEs = 40 then

(with previous

f = 20%)

$$S = \frac{1}{0.2 + \frac{0.8}{40}} = 4.54$$

However :

$$efficiency = \frac{4.54}{40} = 11\%$$

Parallelizing improves performance but it is limited by the sequential portion

Throughput (with parallelism)

Quantity of computation done over time

Especially important for:

- **I / O bound applications**
- **design of pipelines**

$$speedup = \frac{T_{serial}}{T_{pipeline}} = \frac{m \times N}{m + N - 1}$$

REVIEW

$$T = \frac{N \times S}{R}$$

THROUGHPUT Ps ➔

$$P_s = \frac{R}{S}$$

T = performance (time)
N = actual number of executed instructions
S = number of 1-clock steps needed for 1 instruction
R = clock rate (cycles per second)

Mapping an Algorithm onto a Parallel Architecture

Carefully! Non trivial!

Read the example in section 10.1.4 (just read and enjoy!)

Highlights of conclusions:

- ***Worst-case path for matrix multiplication has speedup of 9.3***

Bigger, better, EPIC architectures

Intel IA-64 Itanium

Read about the Itanium in section 10.3.1 (just read and enjoy!)

EPIC = Explicitly Parallel Instruction Computing