LAU
الجامعة اللبنانية الأميركية
Lebanese American University

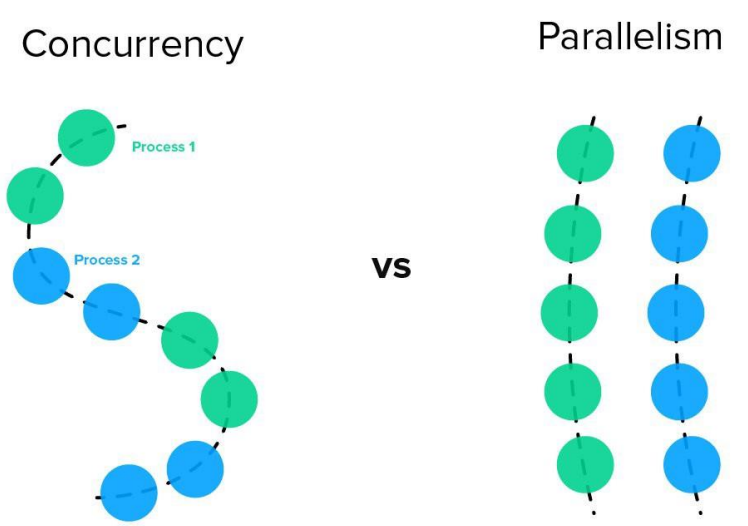
LAU
School of Engineering

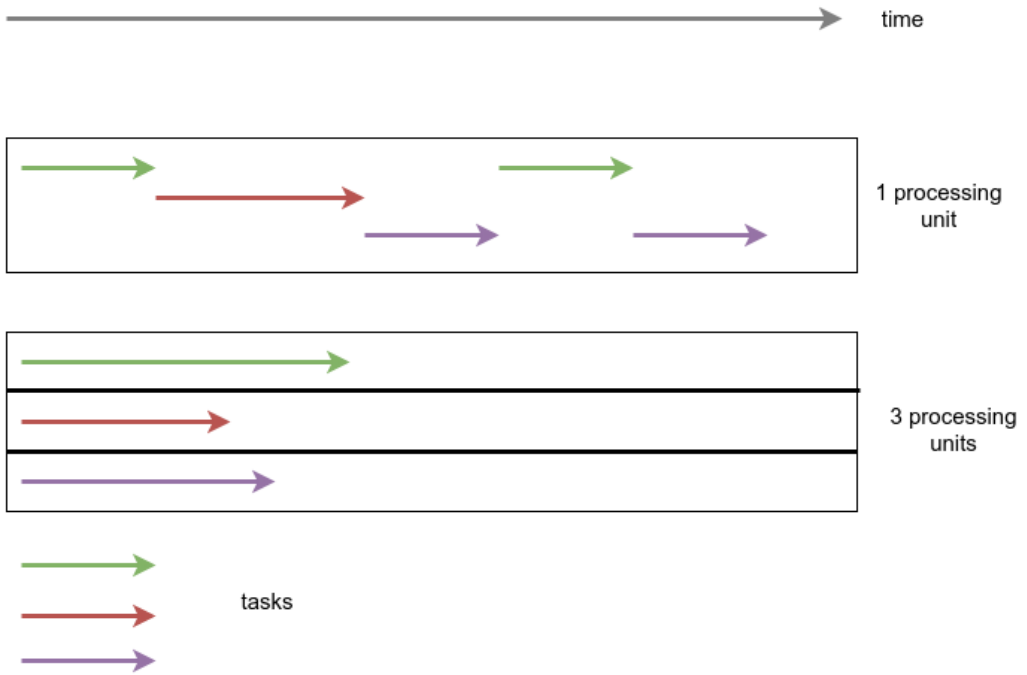
CSC 447: Parallel Programming For Multicore & Cluster Systems

Dr. Hamdan Abdellatef

Parallel Programming vs. Concurrent Programming

- **Parallel computing** is the idea that large problems can be split into **smaller tasks**, and these tasks are **independent** of each other running simultaneously on **more than one processor**.
- **Concurrent programming**, which is the composition of multiple processes that may begin and end at different times, but are managed by the host system's **task scheduler** which frequently switches between them.





Parallelism = concurrency + "parallel" hardware

LAU
الجامعة اللبنانية الأميركية
Lebanese American University

LAU
School of Engineering

Introduction to Parallel Computing

Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers

Chapter 1

Dr. Hamdan Abdellatef

Demand for Computational Speed

- Continual demand for greater computational speed from a computer system than is currently possible

- Areas requiring great computational speed include numerical modeling and simulation of scientific and engineering problems.

- Computations must be completed within a “reasonable” time period.

Grand Challenge Problems

- One that cannot be solved in a reasonable amount of time with today's computers. Obviously, an execution time of 10 years is always unreasonable.

- Examples

  - Modeling large DNA structures
  - Global weather forecasting
  - Modeling motion of astronomical bodies.
  - Many AI applications

Weather Forecasting

- Atmosphere modeled by dividing it into 3-dimensional cells.
- Calculations of each cell repeated many times to model passage of time.

Global Weather Forecasting Example

- Suppose whole global atmosphere divided into cells of size 1 mile $\times$ 1 mile $\times$ 1 mile to a height of 10 miles (10 cells high) - about 5×10^8 cells.

- Suppose each calculation requires 200 floating point operations. In one time step, 10^{11} floating point operations necessary.

- To forecast the weather over 7 days using 1-minute intervals, a computer operating at 1Gflops (10^9 floating point operations/s) takes 10^6 seconds or over 10 days.

- To perform calculation in 5 minutes requires computer operating at 3.4 Tflops (3.4×10^{12} floating point operations/sec).

Modeling Motion of Astronomical Bodies

- Each body attracted to each other body by gravitational forces. Movement of each body predicted by calculating total force on each body.
- With *N* bodies, *N* - 1 forces to calculate for each body, or approx. N^2 calculations. (N $\log_2$ N for an efficient approx. algorithm.)
- After determining new positions of bodies, calculations repeated.
- A galaxy might have, say, 10^{11} stars.
- Even if each calculation done in 1 ms (extremely optimistic figure), it takes 10^9 years for one iteration using N^2 algorithm and almost a year for one iteration using an efficient N $\log_2$ N approximate algorithm.

Parallel Computing

- Using more than one computer, or a computer with more than one processor, to solve a problem.

Motives:

- Usually faster computation - very simple idea - that n computers operating simultaneously can achieve the result n times faster - it will not be n times faster for various reasons.
- Other motives include: fault tolerance, larger amount of memory available, ...

Background

Parallel computers - computers with more than one processor - and their programming - parallel programming - has been around for more than 40 years.

Speedup Factor

where t_s is execution time on a single processor and t_p is execution time on a multiprocessor.

$$S(p) = \frac{\text{Execution time using one processor (best sequential algorithm)}}{\text{Execution time using a multiprocessor with } p \text{ processors}} \quad \frac{t_s}{t_p}$$

$S(p)$ gives increase in speed by using multiprocessor.

Use best sequential algorithm with single processor system. Underlying algorithm for parallel implementation might be (and is usually) different.

Speedup Factor

Speedup factor can also be cast in terms of computational steps:

$$S(p) = \frac{\text{Number of computational steps using one processor}}{\text{Number of parallel computational steps with } p \text{ processors}}$$

Can also extend time complexity to parallel computations.

Efficiency

It is sometimes useful to know how long processors are being used on the computation

$$E = \frac{\text{Execution time using one processor}}{\text{Execution time using a multiprocessor} \times \text{number of processors}}$$

$$= \frac{t_s}{t_p \times p}$$

$$E = \frac{S(p)}{p} \times 100\%$$

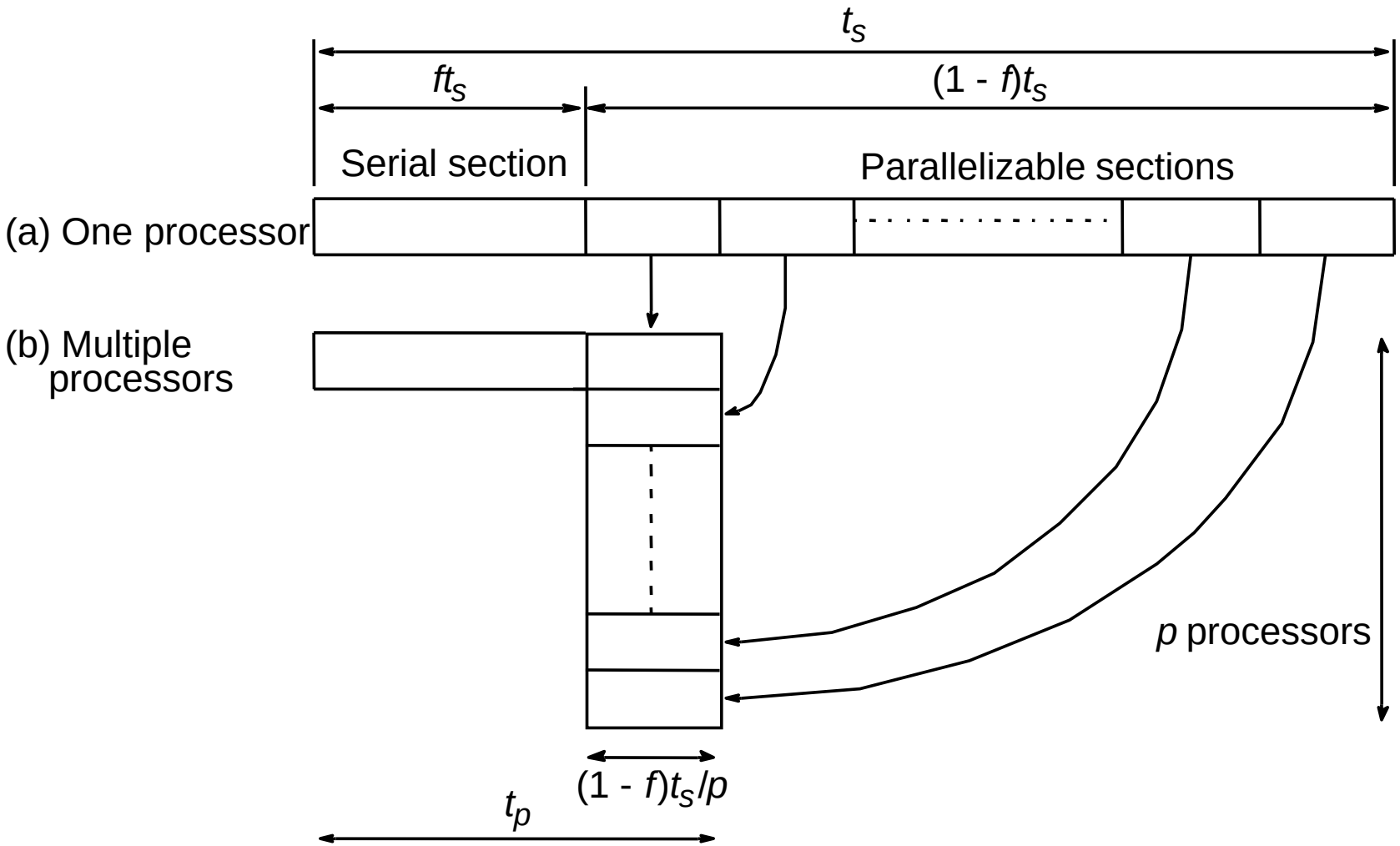
Maximum Speedup

Maximum speedup is usually *p* with *p* processors (linear speedup).

Possible to get superlinear speedup (greater than *p*) but usually a specific reason such as:

- Extra memory in multiprocessor system
- Nondeterministic algorithm

Maximum Speedup - Amdahl's law



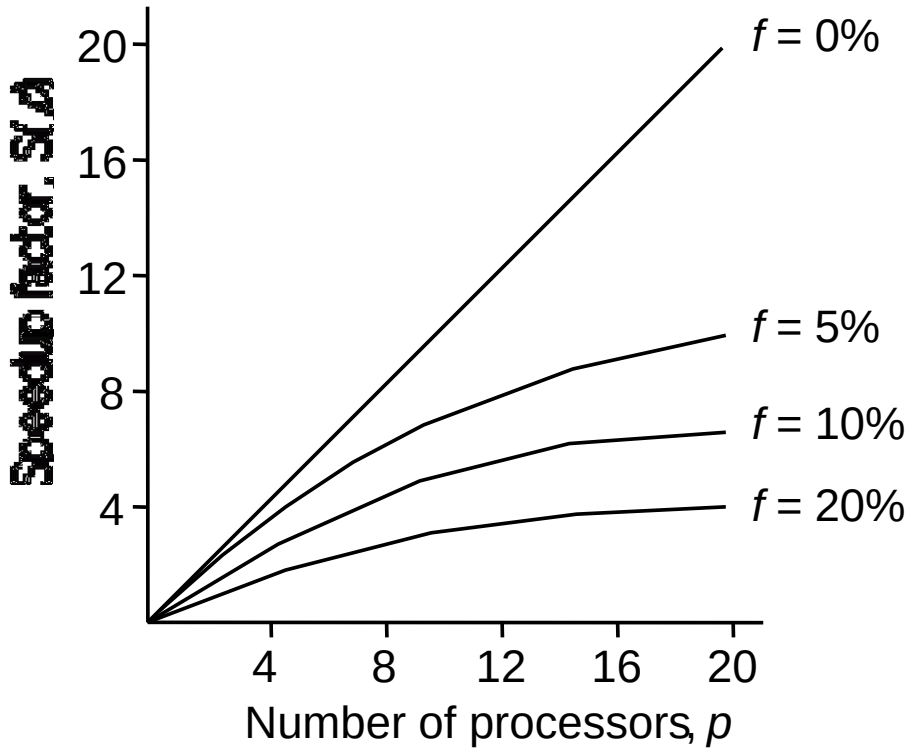
Amdahl's law

Speedup factor is given by:

$$S(p) = \frac{t_s}{ft_s + (1-f)t_s/p} = \frac{p}{1+(p-1)f}$$

This equation is known as Amdahl's law

Speedup against number of processors

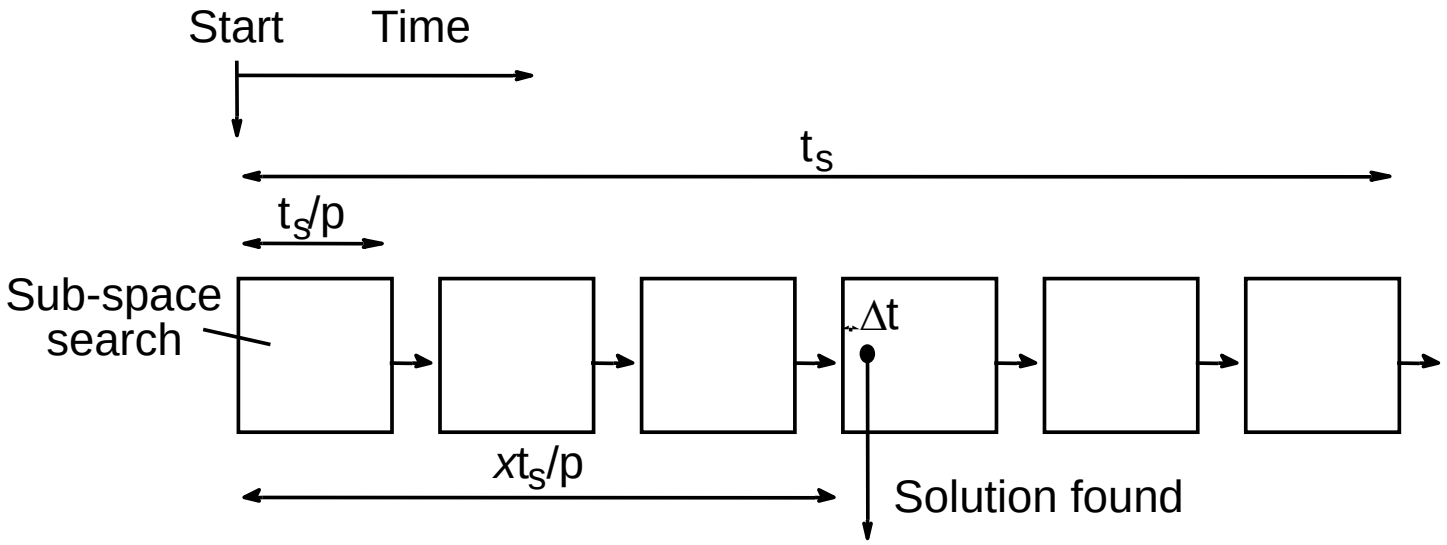


$$S(p) = \frac{t_s}{ft_s + (1-f)t_s/p} = \frac{p}{1+(p-1)f}$$

- Even with infinite number of processors, maximum speedup limited to 1/f.
- Example
- With only 5% of computation being serial, maximum speedup is 20, irrespective of number of processors.

Superlinear Speedup example - Searching

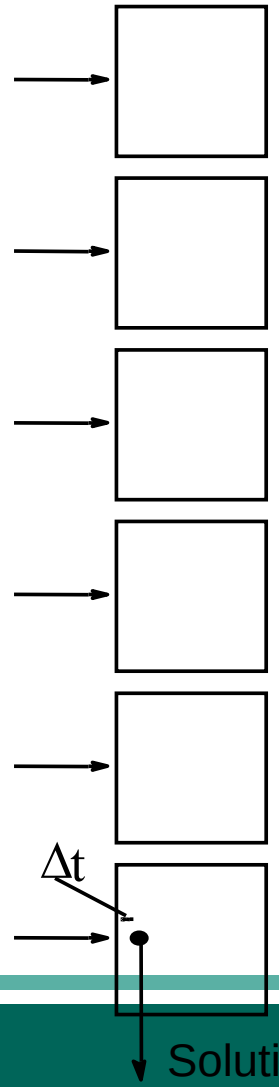
Searching each sub-space sequentially



x indeterminate

$$S(p) = \frac{\left[x \times \frac{t_s}{p} \right] + \Delta t}{\Delta t}$$

Searching each sub-space in parallel



Superlinear Speedup example - Searching

Worst case for sequential search when solution found in last sub-space search. Then parallel version offers greatest benefit, i.e.

$$S(p) = \frac{\left[\frac{p-1}{p}\right] \times t_s + \Delta t}{\Delta t} \rightarrow \infty$$

as Δt tends to zero

Superlinear Speedup example - Searching

Least advantage for parallel version when solution found in first sub-space search of the sequential search, i.e.

$$S(p) = \frac{\Delta t}{\Delta t} = 1$$

Actual speed-up depends upon which subspace holds solution but could be extremely large.

Types of Parallel Computers

Two principal types:

- Shared memory multiprocessor
- Distributed memory multicomputer

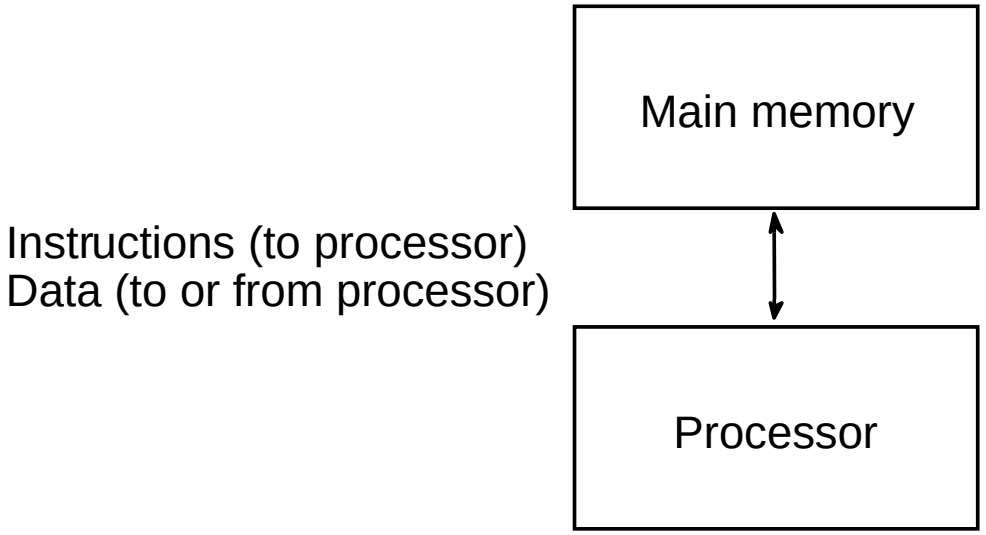
LAU
الجامعة اللبنانية الأميركية
Lebanese American University

LAU
School of Engineering

Shared Memory Multiprocessor

Conventional Computer

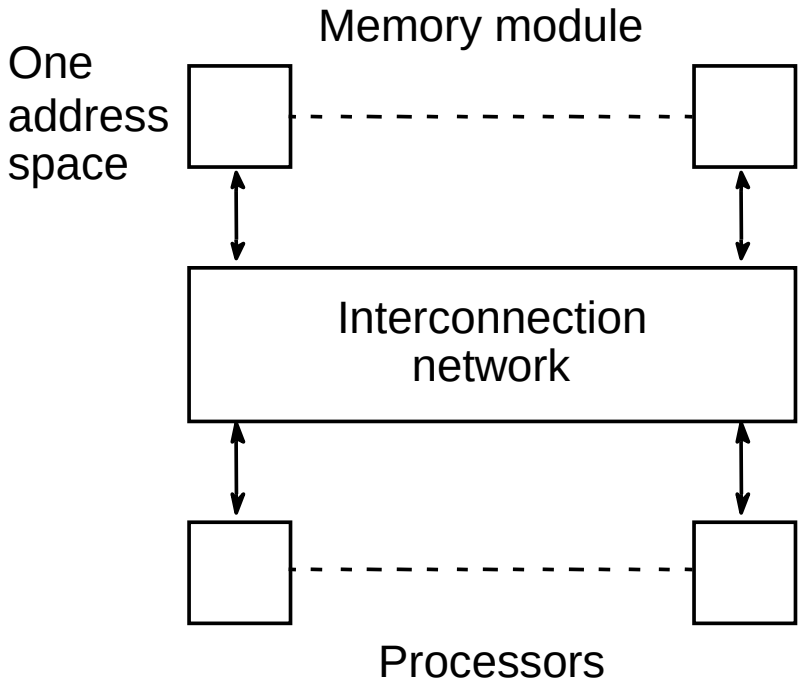
Consists of a processor executing a program stored in a (main) memory:



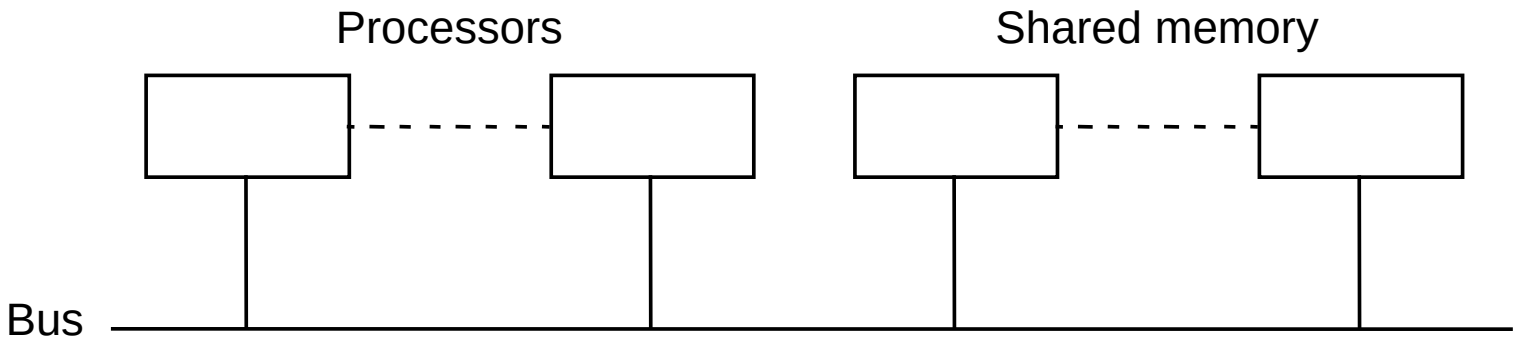
Each main memory location located by its address. Addresses start at 0 and extend to 2^b - 1 when there are *b* bits (binary digits) in address.

Shared Memory Multiprocessor System

Natural way to extend single processor model - have multiple processors connected to multiple memory modules, such that each processor can access any memory module :



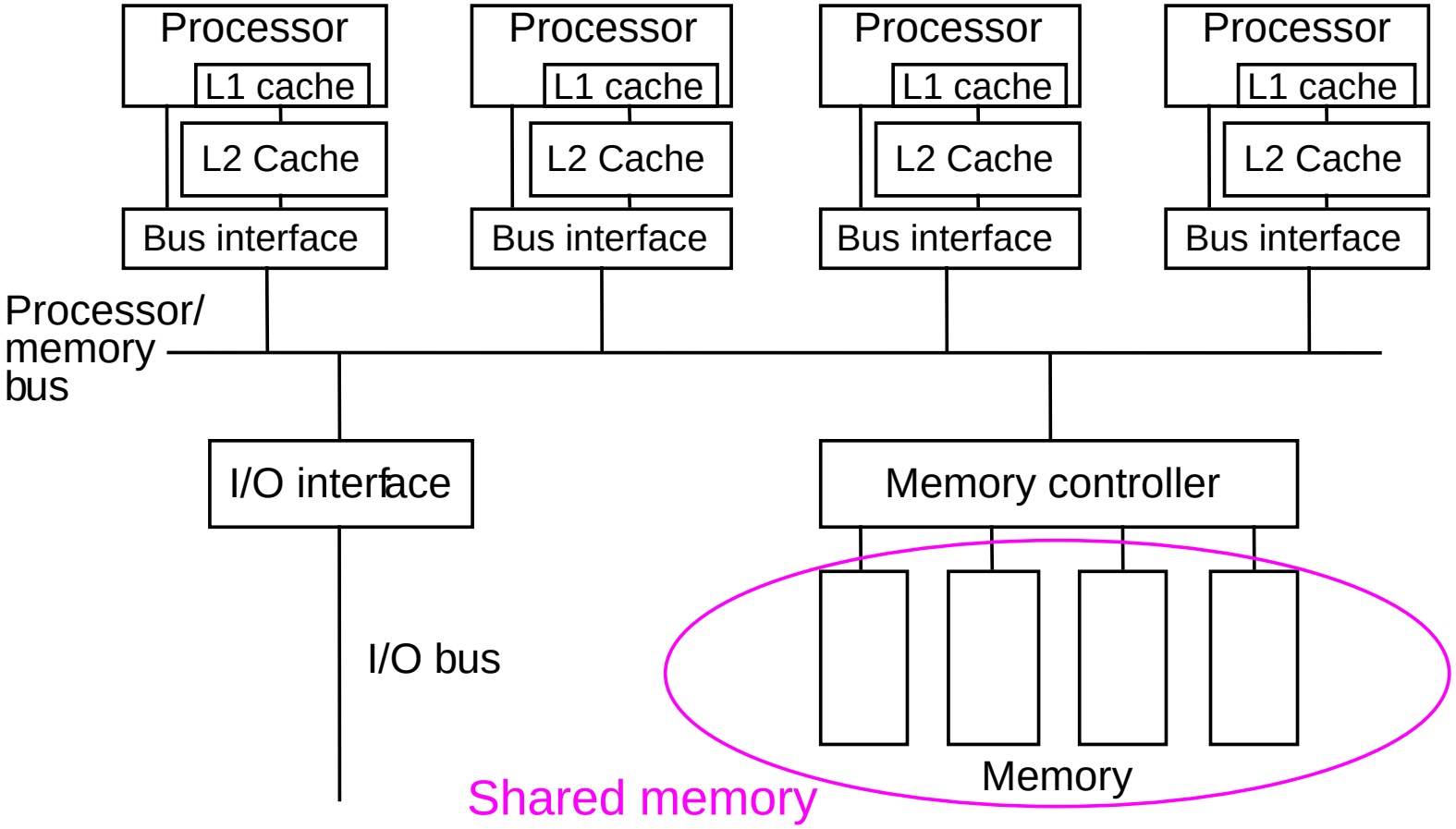
Simplistic view of a small shared memory multiprocessor



Examples:

- Dual Pentiums
- Quad Pentiums

Quad Pentium Shared Memory Multiprocessor



Programming Shared Memory Multiprocessors

- Threads - programmer decomposes program into individual parallel sequences, (threads), each being able to access variables declared outside threads.

  Example Pthreads

- Sequential programming language with preprocessor compiler directives to declare shared variables and specify parallelism.

  Example OpenMP - industry standard - needs OpenMP compiler

Programming Shared Memory Multiprocessors

- Sequential programming language with added syntax to declare shared variables and specify parallelism.

  Example UPC (Unified Parallel C) - needs a UPC compiler.

- Parallel programming language with syntax to express parallelism - compiler creates executable code for each processor (not now common)

- Sequential programming language and ask parallelizing compiler to convert it into parallel executable code. - also not now common

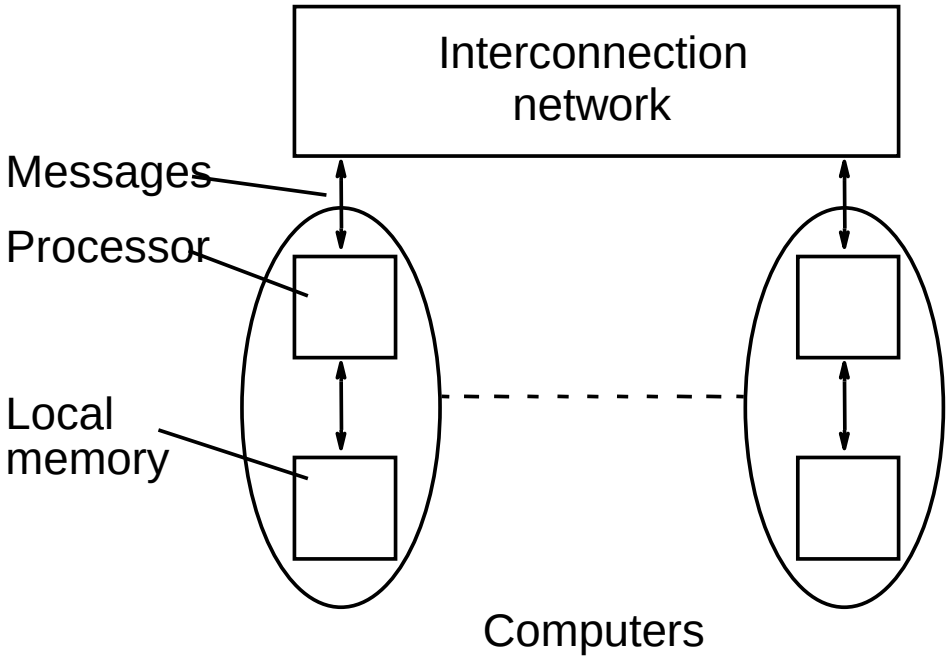
LAU
الجامعة اللبنانية الأميركية
Lebanese American University

LAU
School of Engineering

Message-Passing Multicomputer

Message-Passing Multicomputer

Complete computers connected through an interconnection network:



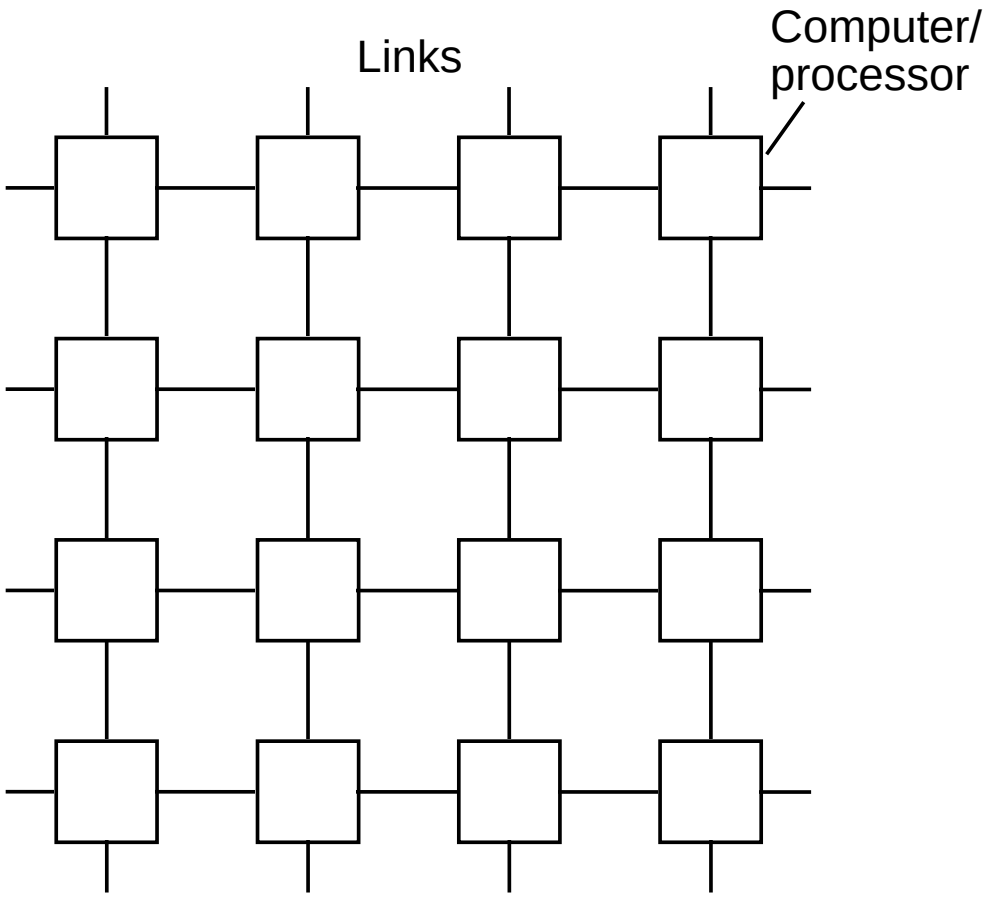
Key issues in network design:

Bandwidth (bits/s)
Latency (network, comm, message)
Cost

Interconnection Networks

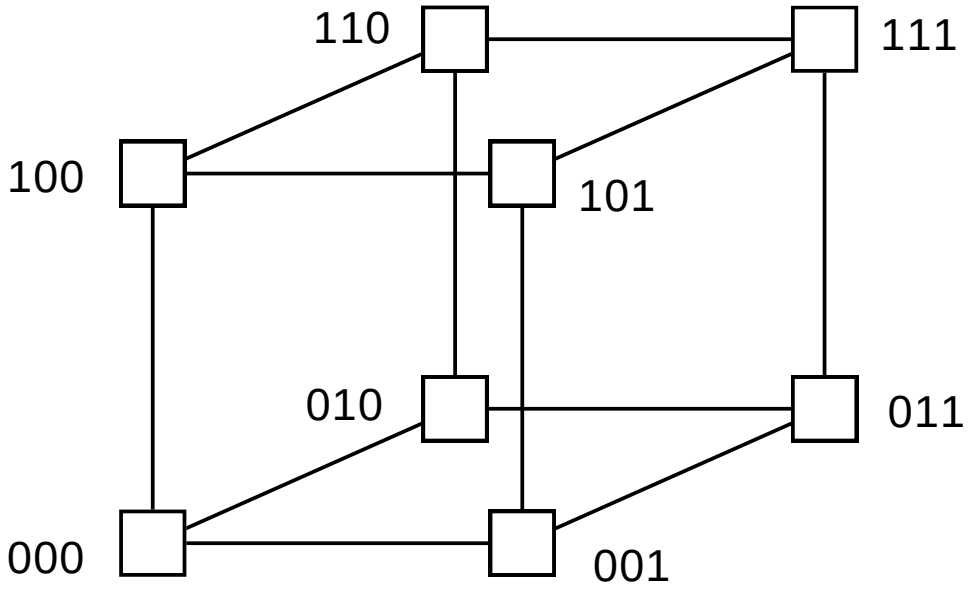
- Limited and exhaustive interconnections
- 2- and 3-dimensional meshes
- Hypercube (not now common)
- Using Switches:
  - Crossbar
  - Trees
  - Multistage interconnection networks

Two-dimensional array (mesh)

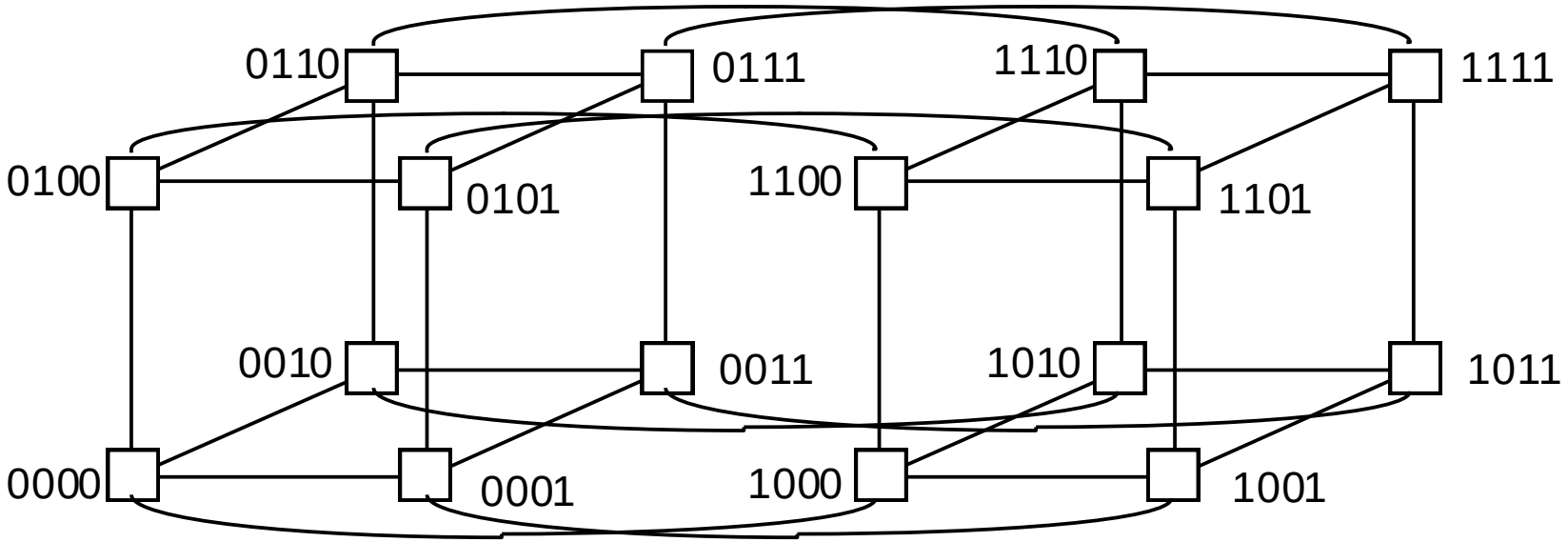


Also three-dimensional - used in some large high performance systems.

Three-dimensional hypercube

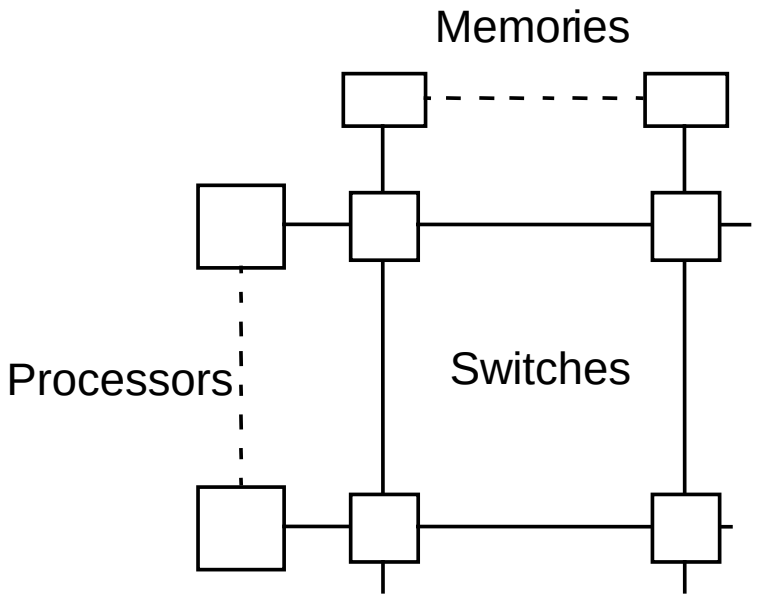


Four-dimensional hypercube

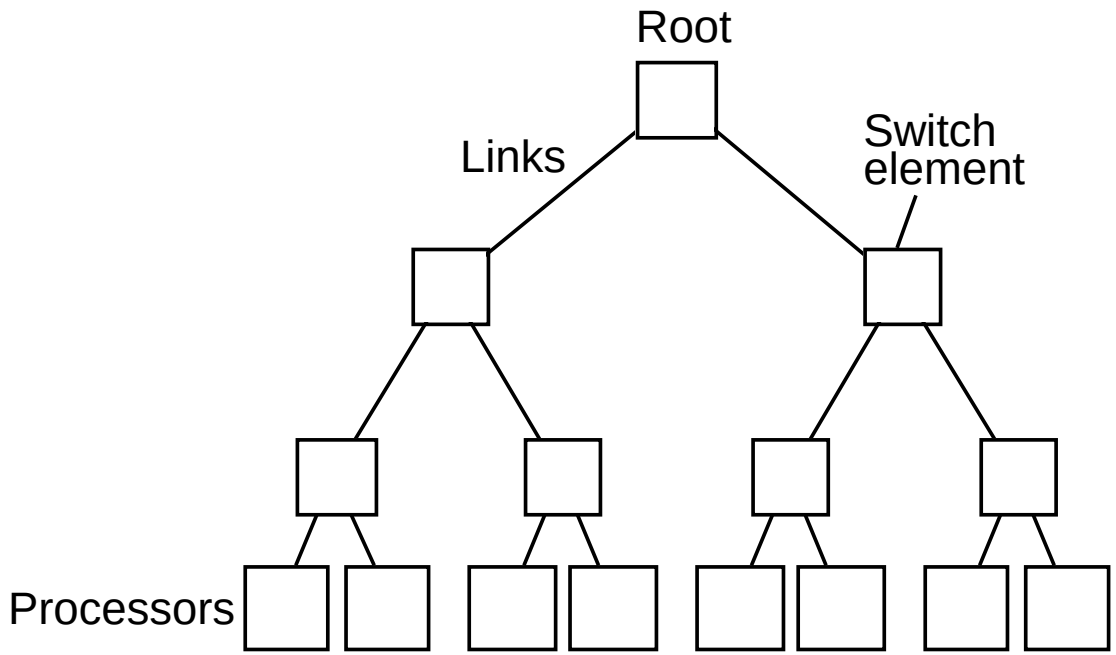


Hypercubes popular in 1980's - not now

Crossbar switch

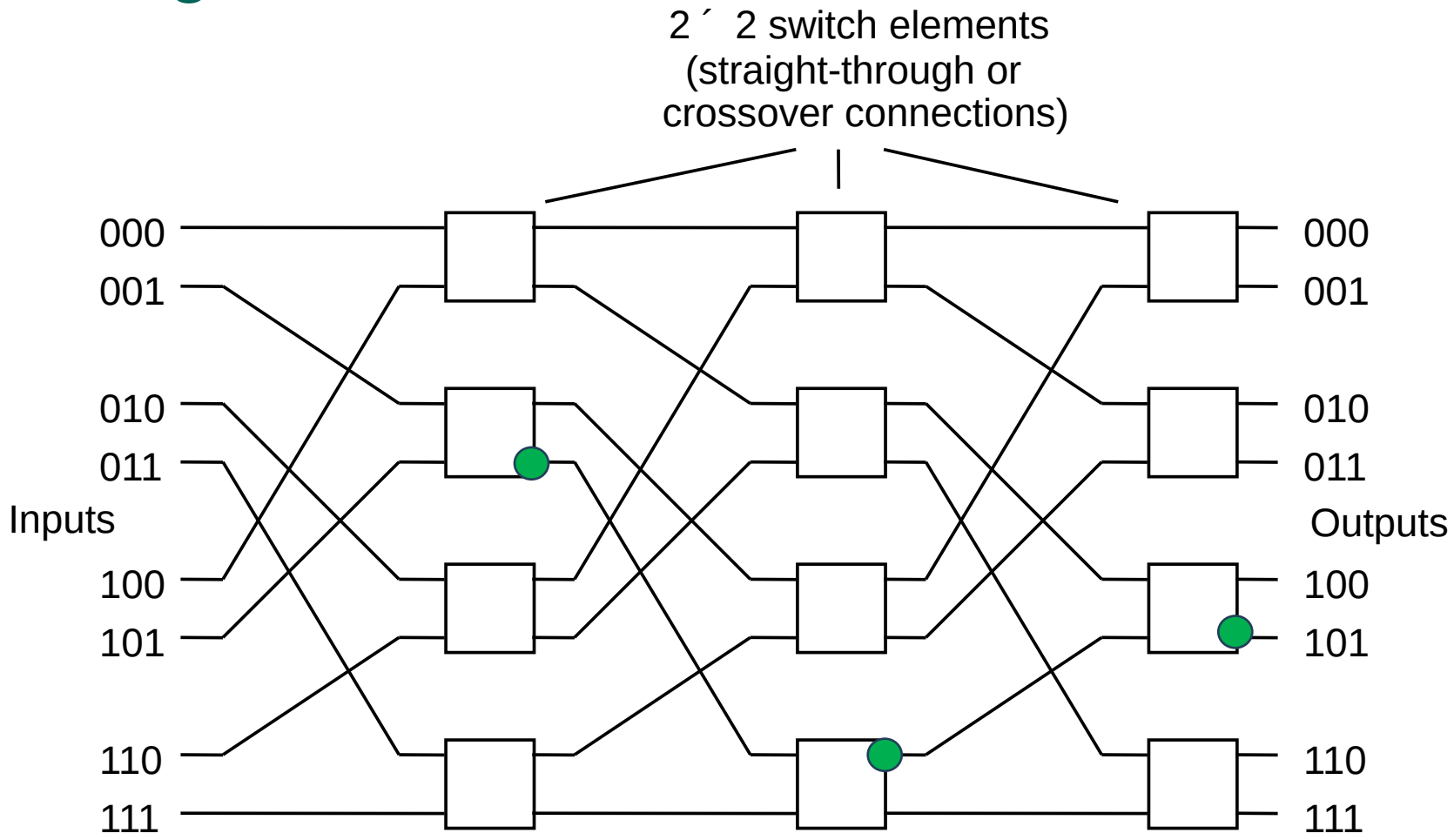


Tree



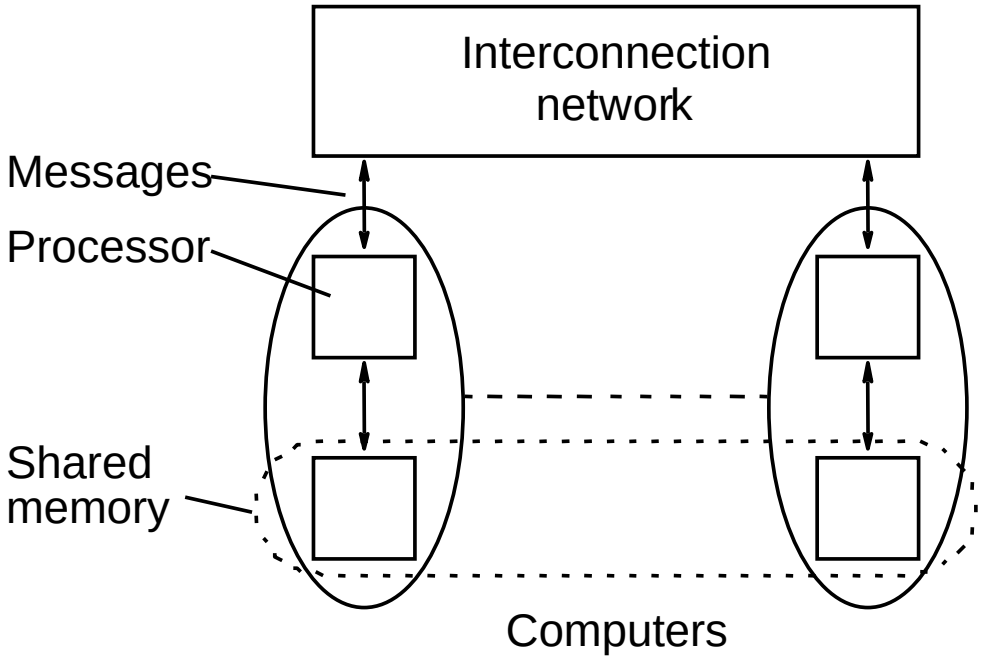
Multistage Interconnection Network
Example: Omega network

Example:
Source 001
Destination 101



Distributed Shared Memory

Making main memory of group of interconnected computers look as though a single memory with single address space. Then can use shared memory programming techniques.



LAU
الجامعة اللبنانية الأميركية
Lebanese American University

LAU
School of Engineering

Flynn's Classifications

SISD: Single instruction stream, single data stream

SIMD: Single instruction stream, multiple data streams

MIMD: Multiple instruction streams, multiple data streams

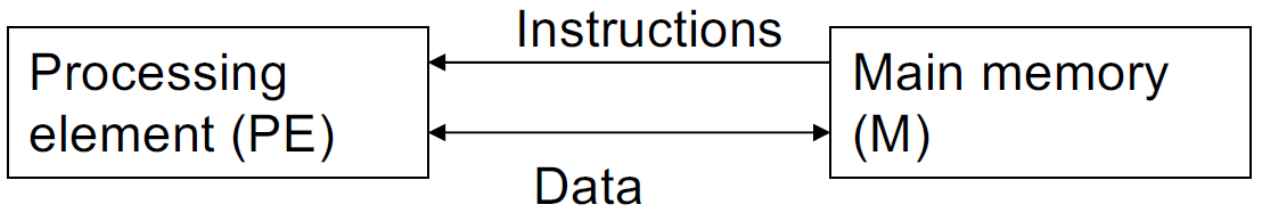
MISD: Multiple instruction streams, single data stream (not common)

Flynn's Classifications

Flynn (1966) created a classification for computers based upon instruction streams and data streams:

- *Single instruction stream-single data stream (SISD) computer*

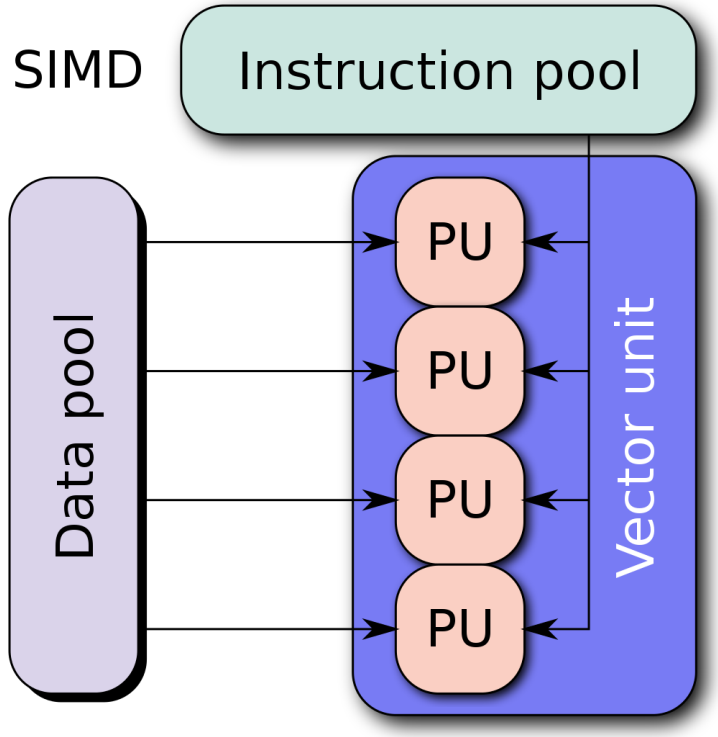
Single processor computer - single stream of instructions generated from program. Instructions operate upon a single stream of data items.



Single Instruction Stream-Multiple Data Stream (SIMD) Computer

- A specially designed computer - a single instruction stream from a single program, but multiple data streams exist. Instructions from program broadcast to more than one processor. Each processor executes same instruction in synchronism, but using different data.

- Developed because a number of important applications that mostly operate upon arrays of data.



Single Program Multiple Data (SPMD) Structure

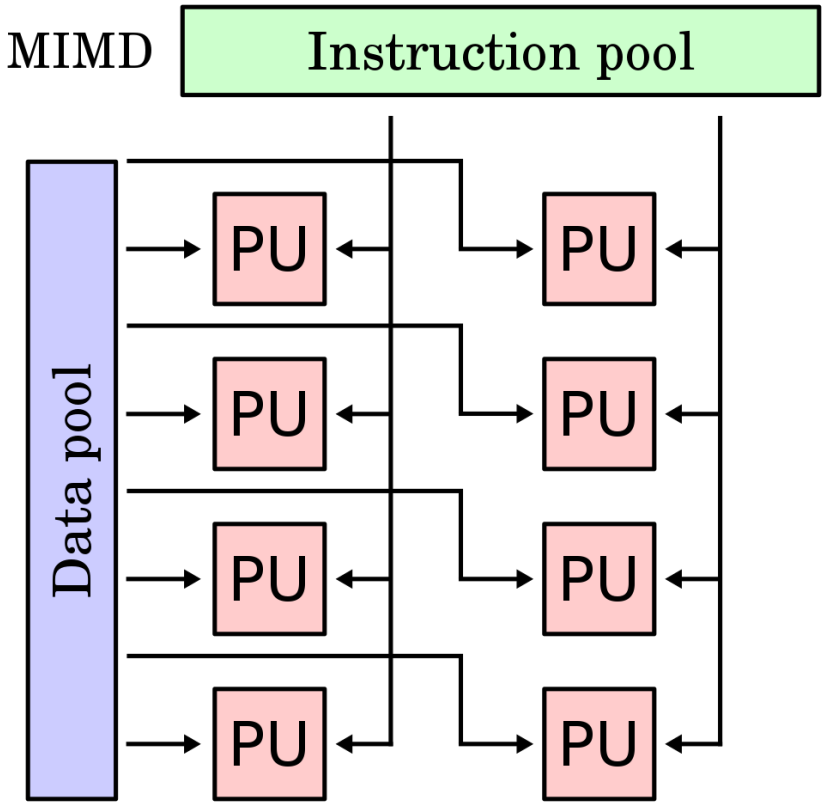
Single source program written and each processor executes its personal copy of this program, although independently and not in synchronism.

Source program can be constructed so that parts of the program are executed by certain computers and not others depending upon the identity of the computer.

Multiple Instruction Stream-Multiple Data Stream (MIMD) Computer

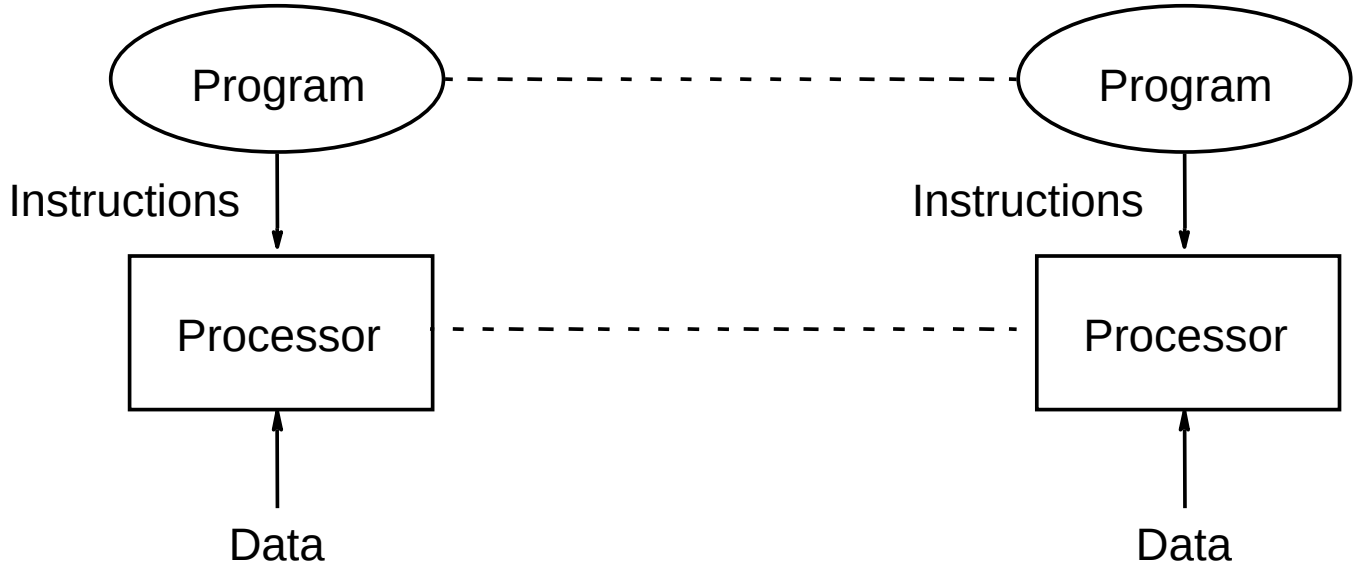
General-purpose multiprocessor system - each processor has a separate program and one instruction stream is generated from each program for each processor. Each instruction operates upon different data.

Both the shared memory and the message-passing multiprocessors so far described are in the MIMD classification.



Multiple Program Multiple Data (MPMD) Structure

Within the MIMD classification, each processor will have its own program to execute:



LAU
الجامعة اللبنانية الأميركية
Lebanese American University

LAU
School of Engineering

Thank You