Parallel Algorithms

by

Biing-Feng Wang
王 炳 豐
Department of Computer Science
National Tsing Hua University

References:

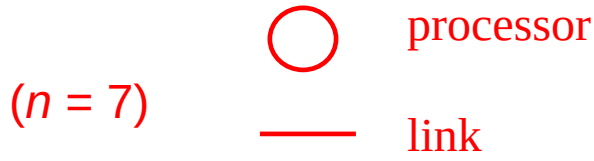
[1] J. JáJá, *An Introduction to Parallel Algorithms*, Addison Wesley, 1992.

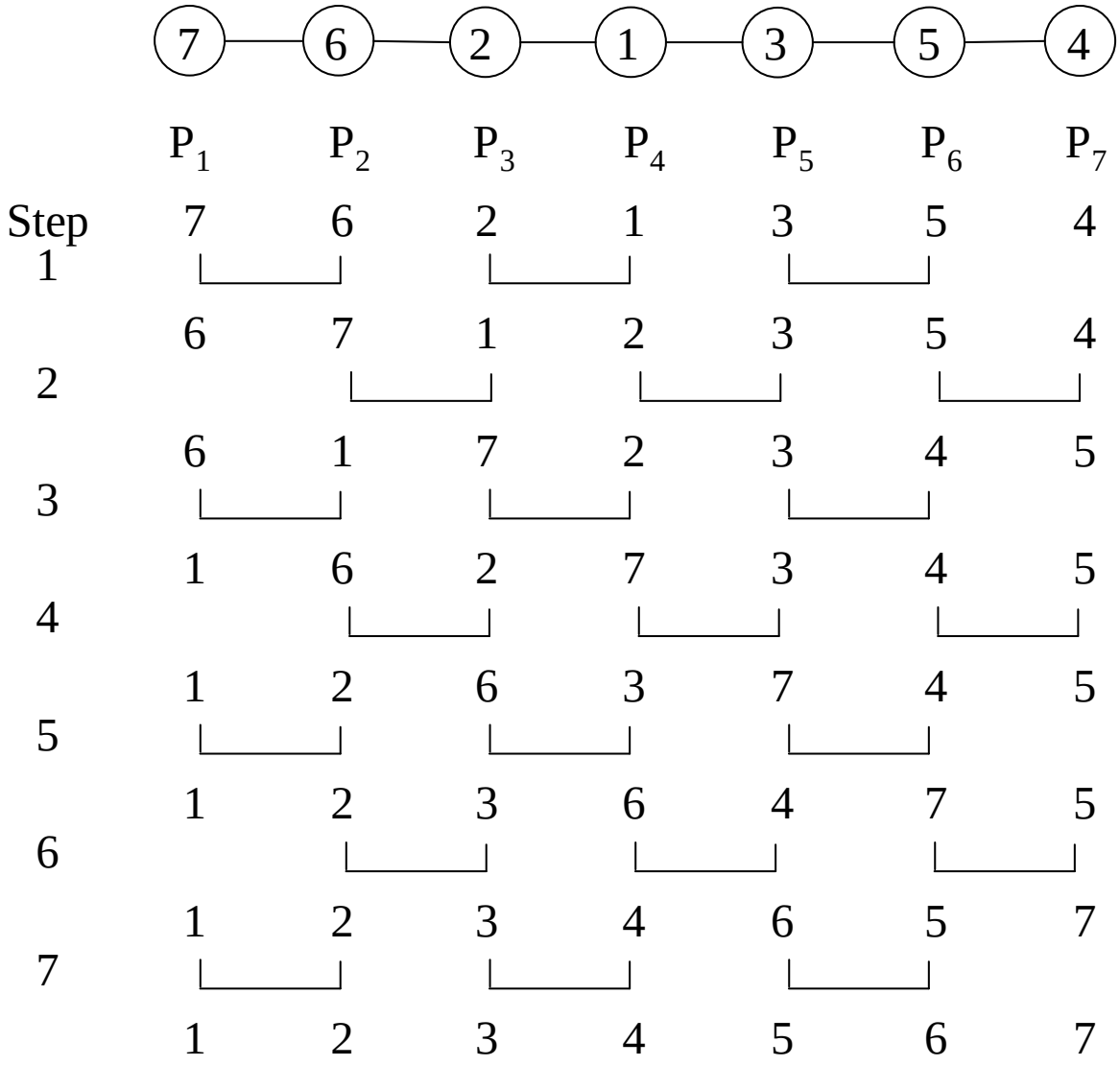
[2] S. G. Akl, *Parallel Computation: Models and Methods*, Prentice-Hall, 1997.

[3] Journals and proceedings

An Example

Problem : Sorting (Odd-Even Transposition Sort)
Input : 7, 6, 2, 1, 3, 5, 4
Output : 1, 2, 3, 4, 5, 6, 7
Model : Linear Processor Array ($n = 7$)





Theorem: Odd-Even transposition sort produces a sorted sequence of n data after n steps.

$O(N)$ time

Outline

Part I

1. Introduction
   - Classification of Parallel Computers
   - Performance of Parallel Algorithms
2. Shared-Memory Computers, Basic Techniques, and Brent's Theorem

Part II

3. Tree Machines
4. Linear Processor Arrays
5. Mesh-Connected Computers
6. Hypercubes
7. Perfect Shuffles
8. Mesh-Connected Computers with Multiple Broadcasting
9. Processor Arrays with Reconfigurable Bus Systems

Part III

10. Systolic Architectures
11. Randomized Algorithms and P-Completeness

Marking

· Midterm Exam: 60%
· Final Report: 40%
$\Rightarrow$ normalize
· extra: discussion (+), absence (–)

Office Hours

· 13:00 ~ 15:00, Monday to Friday:

1

Introduction

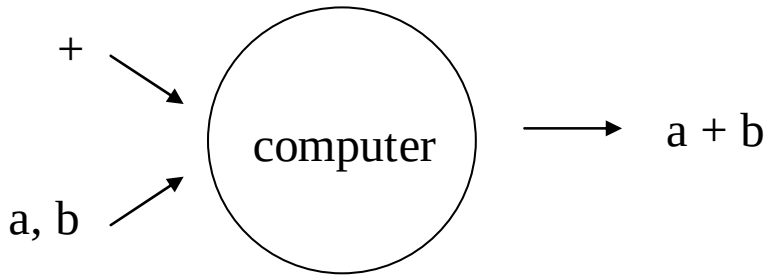
■ Classification of Parallel Computers

1. Flynn's Classification
   (M. J. Flynn, "Very high speed computing systems," *Proceedings of the IEEE*, vol. 54, 1966, pp. 1901-1909)

Instruction stream (*I*): a sequence of instructions performed by a computer

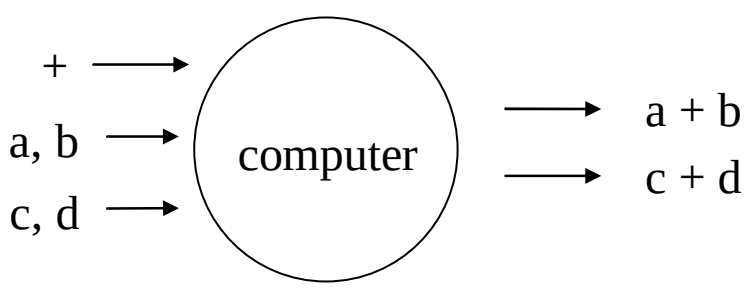
Data stream (*D*): a sequence of data used to execute an instruction stream

SISD: single instruction stream, single data stream



One instruction is performed at a time, on one set of data.

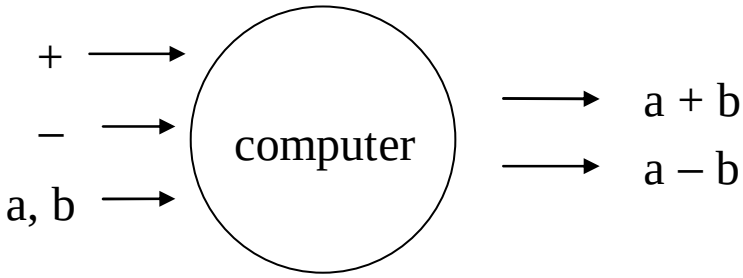
SIMD: single instruction stream, multiple data streams



vector computer
$A + B$

One type of instruction is performed at a time, possible on different sets of data.

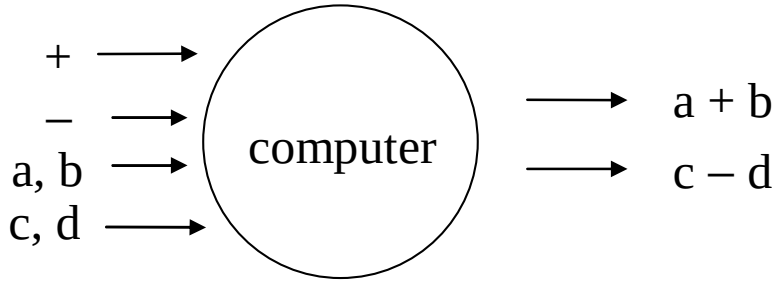
MISD: multiple instruction streams, single data stream



pipeline computer ???!!!

Different instructions on the same data can be performed at a time.

MIMD: multiple instruction streams, multiple data streams

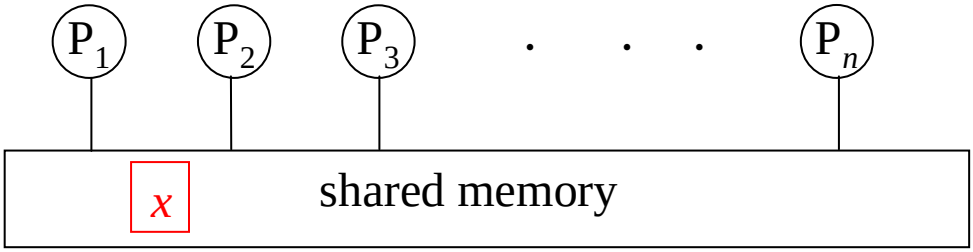


Different instructions on different data can be performed at a time.

2. Schwartz's Classification

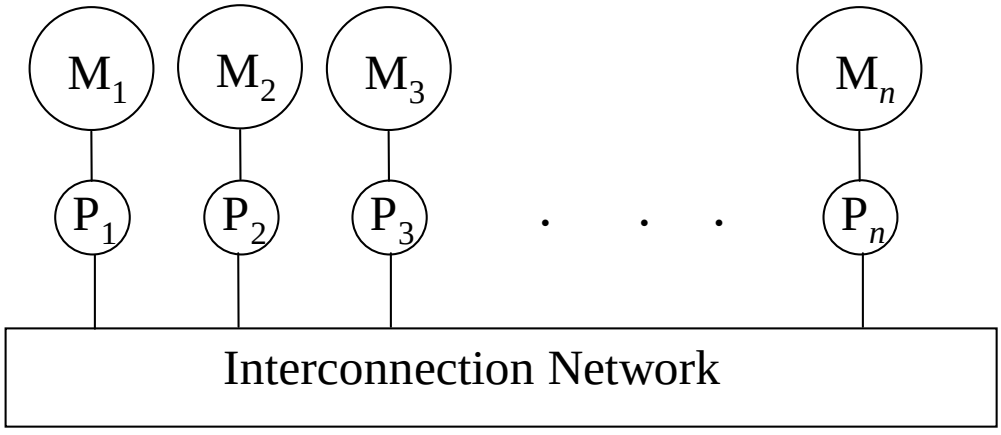
(J. T. Schwartz, "Ultra-computers", *ACM Transactions on Programming Languages and Systems*, vol. 2, no. 4, 1980, pp. 484-521.)

1. *Paracomputer* (*shared-memory computer*)



* Communication between any two processors takes $O(1)$ time through the shared memory
* The SIMD shared-memory computer is also called *Parallel Random Access Machine* (*PRAM*)

2. *Ultracomputer*

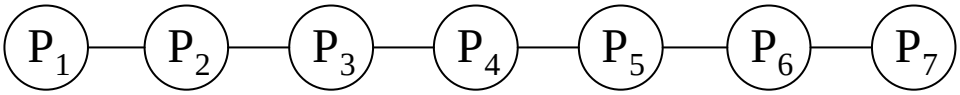


* The processors communicate with one another through an interconnection network. $\Rightarrow$ $O(1)$

$\neq$ distributed systems (computer networks)
(distributed algorithms)

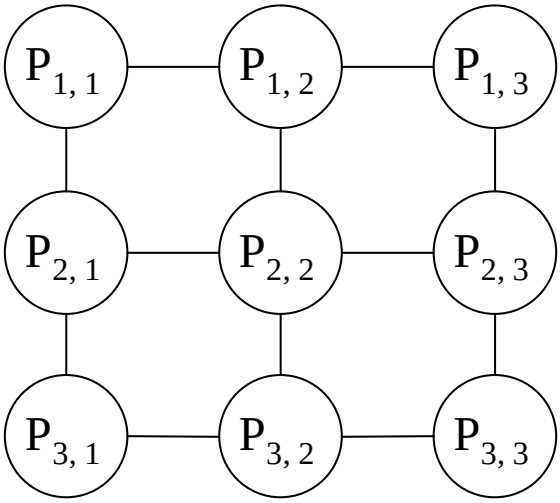
Examples of widely used interconnection network:

1. Linear Processor Array ($n = 7$)

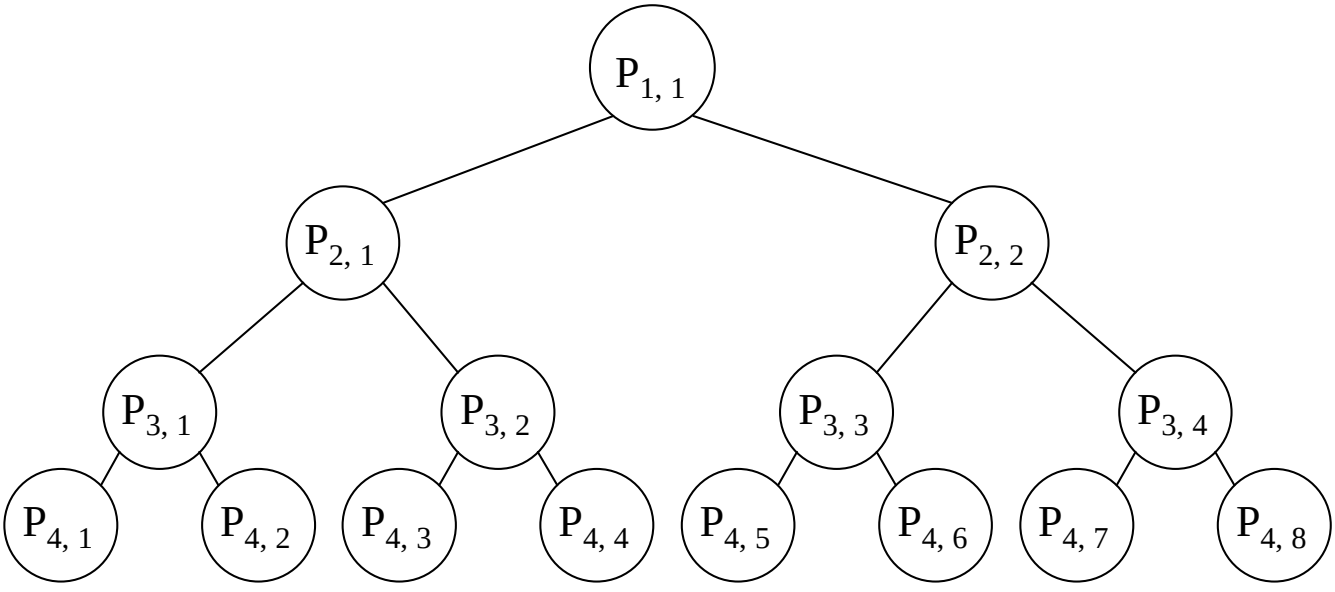


2. Mesh-Connected Computer (2d 3×3 mesh)

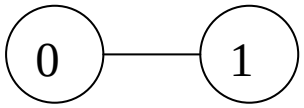
diameter
maximum degree
link cmplexity



3. Tree Machine ($n = 2^4 - 1$)

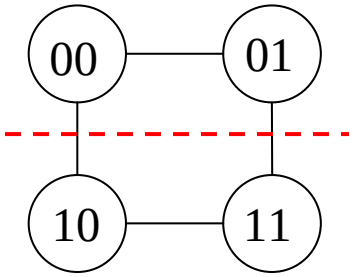


4. *Hypercube* ($n = 2^k$)



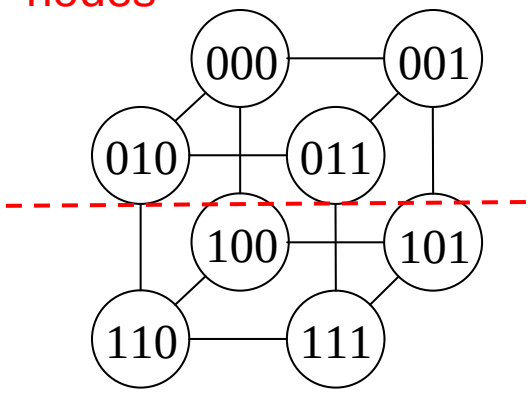
1-cube

2^1 nodes



2-cube

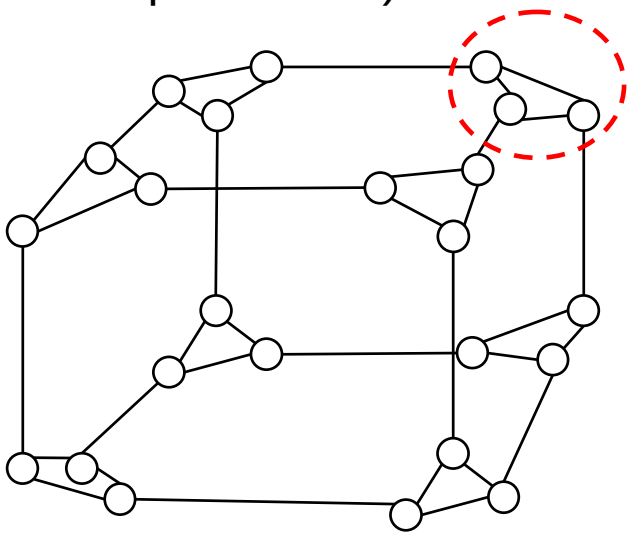
2^2 nodes



3-cube

2^3 nodes

5. *Cube-connected Cycle network* (each node of a *k*-cube is replayed by a cycle of *k* processors)

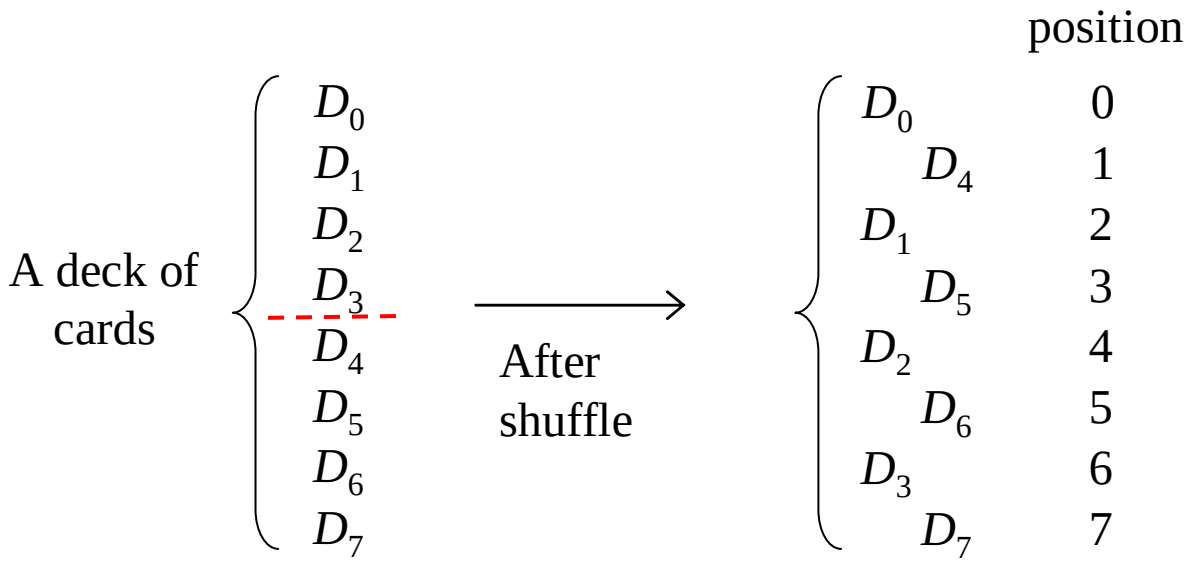


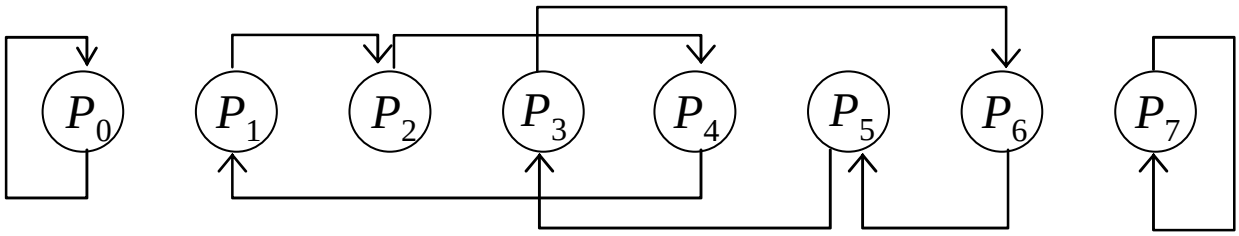
dimension = 3

$2^3 \times 3$ nodes

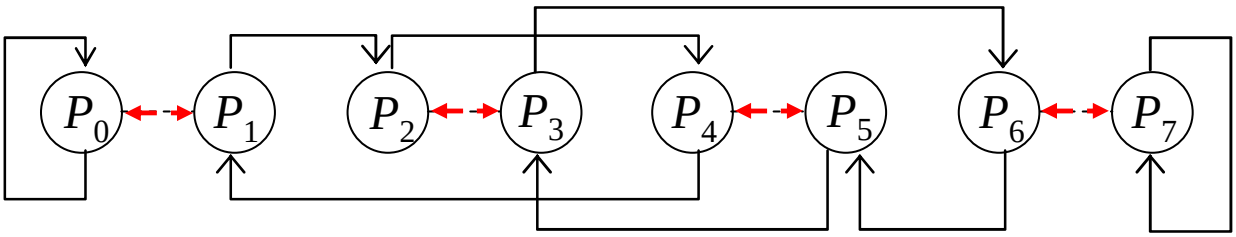
6. *Perfect Shuffle*

$k\text{-}d \Rightarrow n = 2^k$

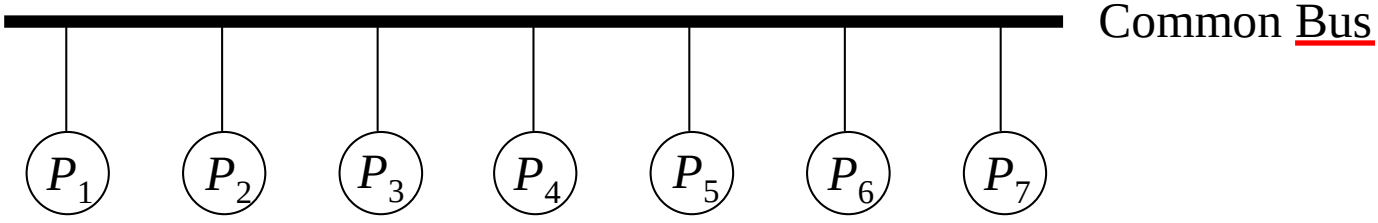




7. *Shuffle-Exchange Network*

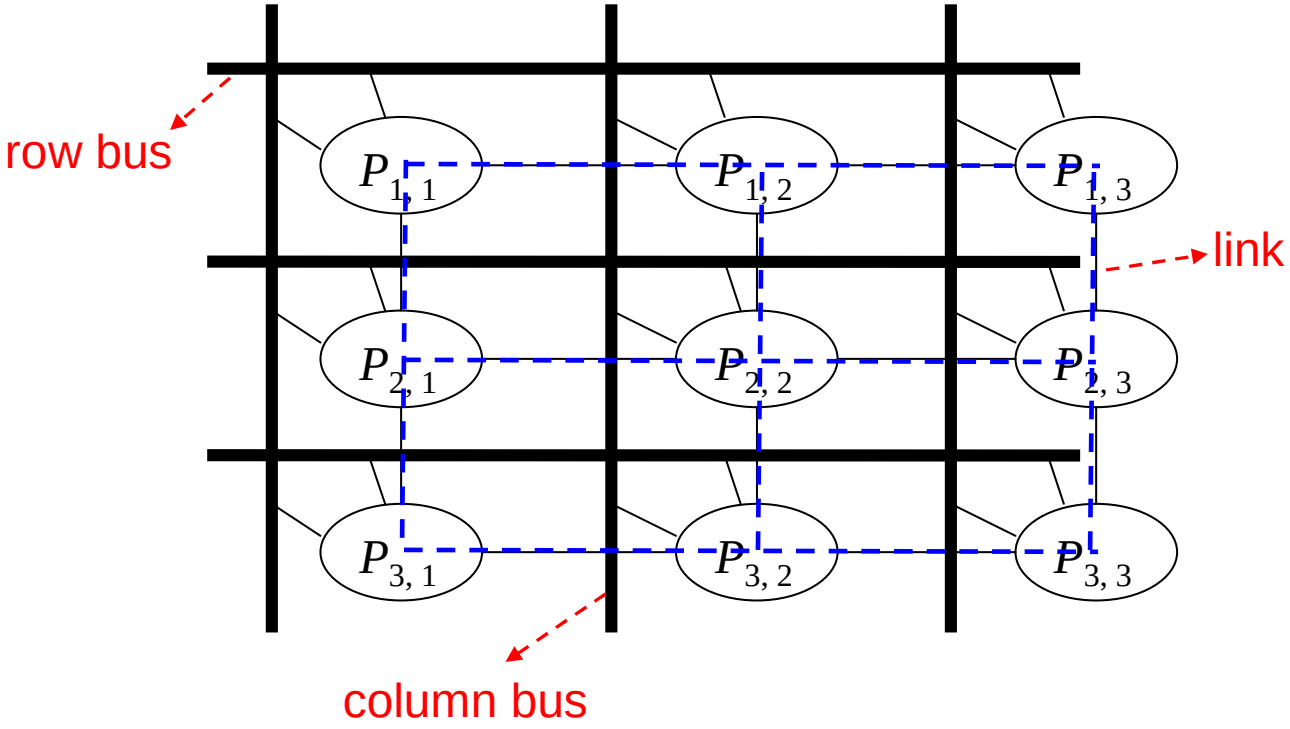


8. *Single-channel broadcast communication System*



* Broadcast a data requires $O(1)$ time.

9. *Mesh-Connected Computer with Multiple Broadcasting*



■ Performance of Parallel Algorithms

$$\textbf{\textit{Speedup}} = \frac{\text{worst-case running time of fastest known sequential algorithm}}{\text{worst-case running time of parallel algorithm}}$$

*****A parallel algorithm is said to achieve *linear speedup* if the speedup with p processor is $\theta(p)$.

* O, θ, Ω

Cost = (parallel running time)×(number of processors used)

*****A parallel algorithm is said to be *cost-optimal* if the cost matches the (sequential) lower bound to within a constant multiplicative factor.

fastest known
目前已知最快

$$\textbf{\textit{Efficiency}} = \frac{\text{worst-case running time of fastest known sequential algorithm}}{\text{cost of parallel algorithm}}$$

= (speedup) / (number of used processors)

≤ 1

Example: Consider the odd-even transposition sort on a linear array of n processors, which sort n data in $O(n)$ time.

Speedup = $(n\log n)/n = \log n$ (big-O)
cost = $n \times n = n^2$ (non-optimal!)
efficiency = $\log n/n$