Course Outline

- Introduction in software and applications
- Parallel machines and architectures
  - Overview of parallel machines
  - Cluster computers (Myrinet)
- Programming methods, languages, and environments
  - Message passing (SR, MPI, Java)
  - Higher-level languages: Linda, Orca, HPF
- Applications
  - N-body problems, graphics, game tree search
- World-wide parallel computing (Globus)

Parallel Machines

Parallel Computing - Theory and Practice (2/e)

Chapter 3

Michael J. Quinn

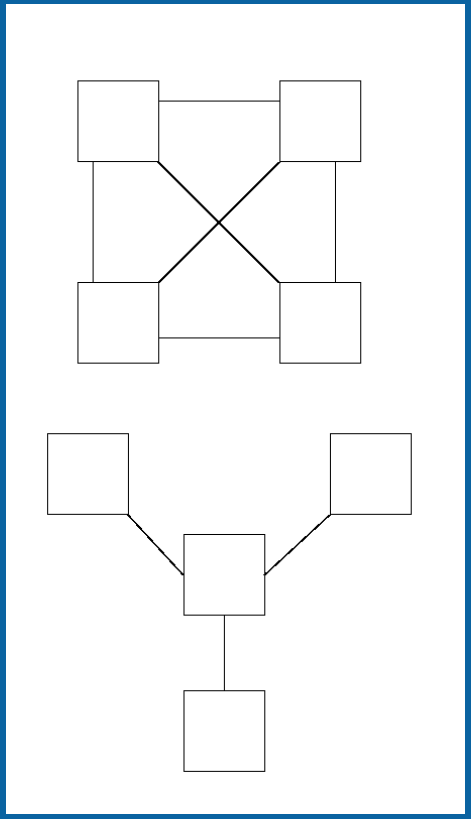
mcGraw-Hill, Inc., 1994

Overview

- Processor organizations
- Types of parallel machines
  - Processor arrays
  - Shared-memory multiprocessors
  - Distributed-memory multicomputers
- Taxonomy parallel machines
- Network of workstations & processor pools
- Myrinet
- Performance metrics

Processor Organization

- Network topology is a graph
  - A node is a processor
  - An edge is a communication path
- Evaluation criteria
  - Diameter (maximum distance)
  - Bisection width
  - Number of edges per node
  - Maximum edge length

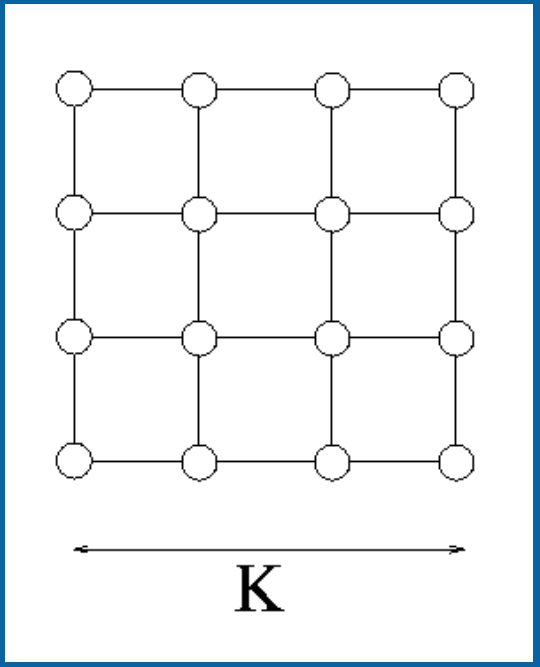


Mesh

q-dimensional lattice

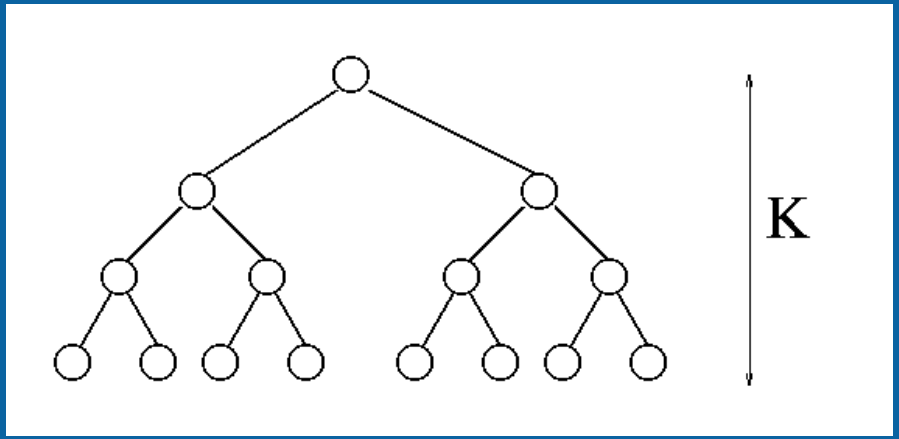
q=2 $\Rightarrow$ 2-D grid

Number of nodes	k^2
Diameter	2(k - 1)
Bisection width	k
Edges per node	4



Binary Tree

Number of nodes	$2^k - 1$
Diameter	$2(k-1)$
Bisection width	1
Edges per node	3



Hypertree

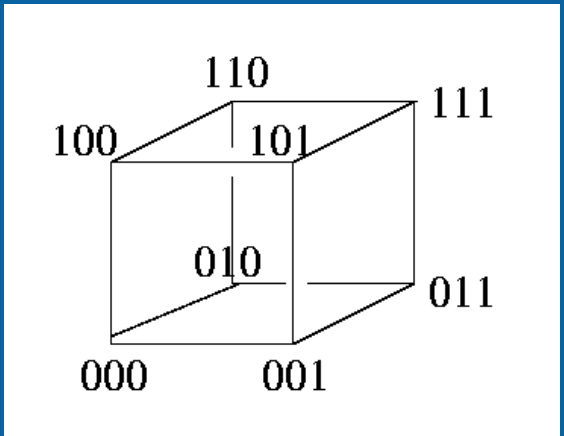
Tree with multiple roots (see Figure 3-3), gives better bisection width

4-ary tree:

Number of nodes	$2^k(2^{k+1}-1)$
Diameter	$2k$
Bisection width	2^{k+1}
Edges per node	6

Hypercube

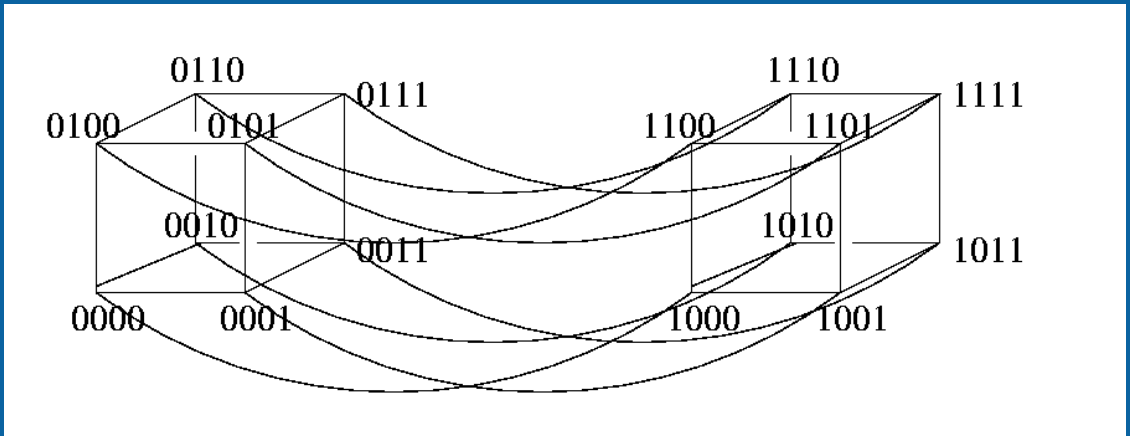
k-dimensional cube, each node has binary value, nodes that differ in 1 bit are connected



Number of nodes	2^k
Diameter	k
Bisection width	2^{k-1}
Edges per node	k

Hypercube

Label nodes with binary value, connect nodes that differ in 1 coordinate



Number of nodes	2^k
Diameter	k
Bisection width	2^{k-1}
Edges per node	k

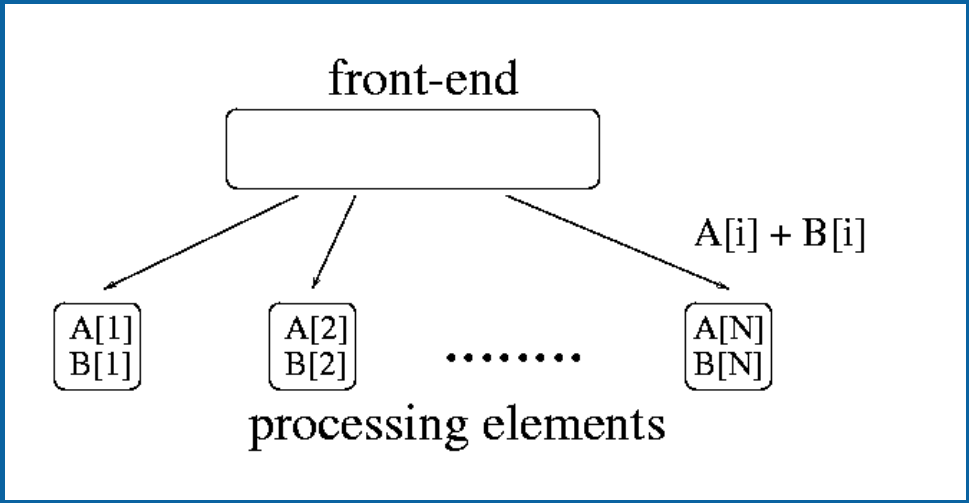
3 types of parallel machines

- Processor arrays
- Shared-memory multiprocessors
- Distributed-memory multicomputers

Processor Arrays (1)

Instructions operate on scalars or *vectors*

Processor array = front-end + synchronized processing elements



Processor Arrays (2)

Front-end

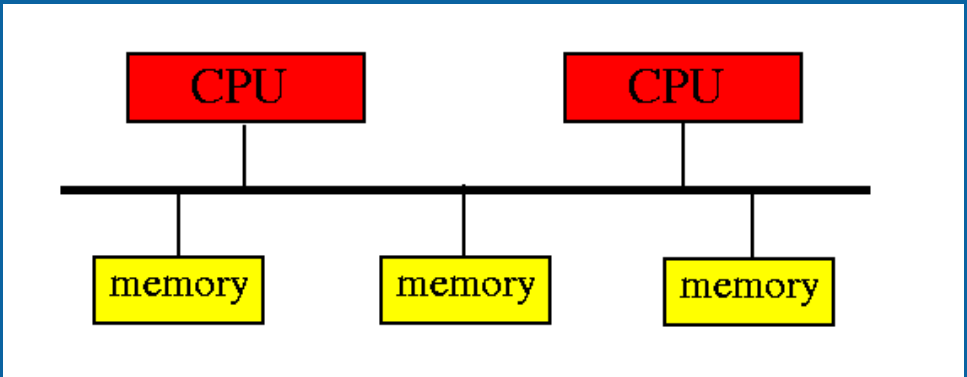
- Sequential machine that executes program
- Vector operations are broadcast to PEs

Processing element

- Performs operation on its part of the vector
- Communicates with other PEs through a network

Examples: CM-200, Maspar MP-1, MP-2, ICL DAP

Shared-Memory Multiprocessors



Bus easily gets saturated $\Rightarrow$ add caches to CPUs

Central problem: cache coherency

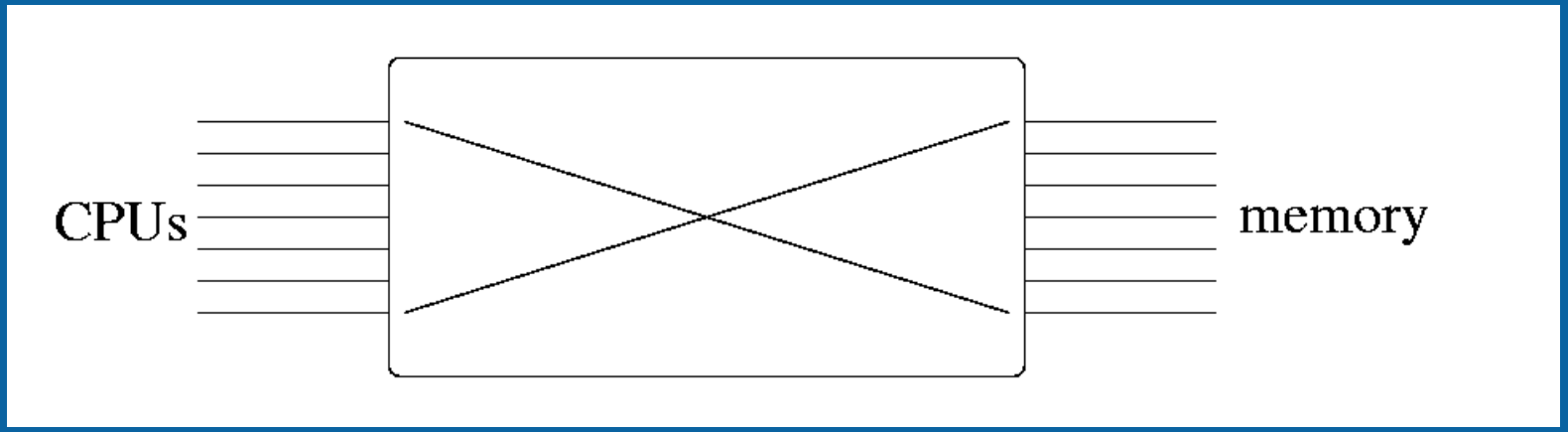
- Snooping cache: monitor bus, invalidate copy on write
  Write-through or copy-back

Bus-based multiprocessors do not scale

Other Multiprocessor Designs

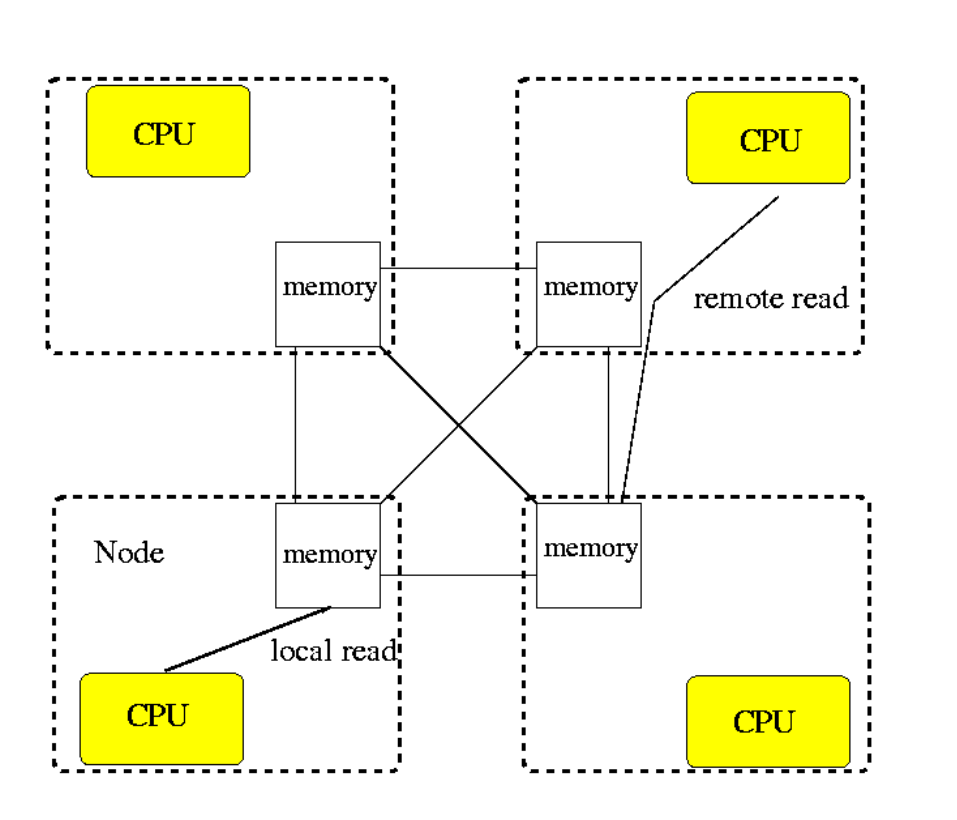
- Switch-based multiprocessors (e.g., crossbar)
- Non-Uniform Memory Access (NUMA) multiprocessors

Switch-based multiprocessors



- Expensive (requires many very fast components)

NUMA



- Memory is distributed
- Some memory is faster to access than other memory

Distributed-Memory Multicomputers

Each processor only has a local memory

Processors communicate by sending messages over a network

Routing of messages:

- Store-and-forward (1st generation)
- Circuit-switched message routing (2nd generation)

Store-and-forward Routing

Messages are forwarded one node at a time

Forwarding is done in software

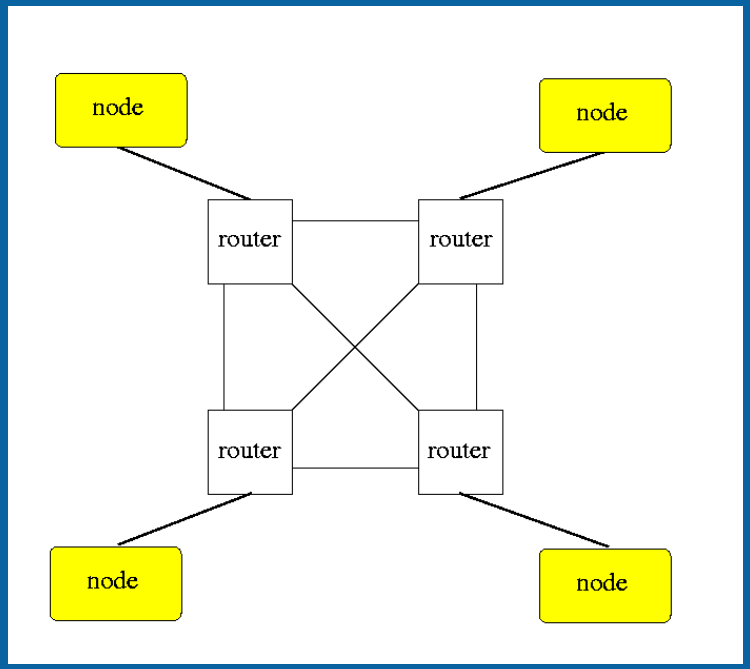
Every processor on path from source to destination is involved

Latency linear to $distance \times message\_length$

Examples: Parsytec GCel (T800 transputers), Intel iPSC

Circuit-switched Message Routing

- Each node has a routing module
- Circuit set up between source and destination
- Latency linear to $distance + message\_length$
- Example: Intel iPSC/2



Flynn's Taxonomy (1)

Instruction stream: sequence of instructions

Data stream: sequence of data manipulated by instructions

	Single Data	Multiple Data
Single Instruction	SISD	SIMD
Multiple Instruction	MISD	MIMD

Flynn's Taxonomy (2)

SISD: Single Instruction Single Data

Traditional uniprocessors

SIMD: Single Instruction Multiple Data

Processor arrays

MISD: Multiple Instruction Single Data

Nonexistent?

MIMD: Multiple Instruction Multiple Data

Multiprocessors and multicomputers