Types of parallelism in Software

1

- Bit-level parallelism
- Instruction-level parallelism
- Task parallelism
- Data Parallelism

1/13/2022

Bit-level parallelism

2

- From the advent of very-large-scale integration (VLSI) computer-chip fabrication technology in the 1970s until
- about 1986, speed-up in computer architecture was driven by doubling computer word size—the amount of
- information the processor can manipulate per cycle.

1/13/2022

Bit-level parallelism

3

- Increasing the word size reduces the number of instructions the processor must execute to perform an operation on variables.
- For example, where an 8-bit processor must add two 16-bit integers, it requires two instructions to
- complete a single operation, where a 16-bit processor would be able to complete the operation with a single instruction.

1/13/2022

Bit-level parallelism

4

- Historically, 4-bit microprocessors were replaced with 8-bit, then 16-bit, then 32-bit microprocessors. The 32-bit processors, have been a standard in general-purpose computing.
- Not until recently, with the advent of x86-64 architectures, have 64-bit processors become commonplace.

1/13/2022

Instruction-level parallelism

5

- A computer program, is a stream of instructions executed by a processor.
- These instructions can be re-ordered and combined into groups which are then executed in parallel.
- This is known as instruction-level parallelism.
- Advances in instruction-level parallelism dominated computer architecture from the mid-1980s until the mid-1990s.

1/13/2022

Instruction-level parallelism

6

- Modern processors have multi-stage instruction pipelines.
- Each stage in the pipeline corresponds to a different action the processor performs on that instruction in that stage;
  - a processor with an N-stage pipeline can have up to N different instructions at different stages of completion.
- The Pentium 4 processor has a 35-stage pipeline.

1/13/2022

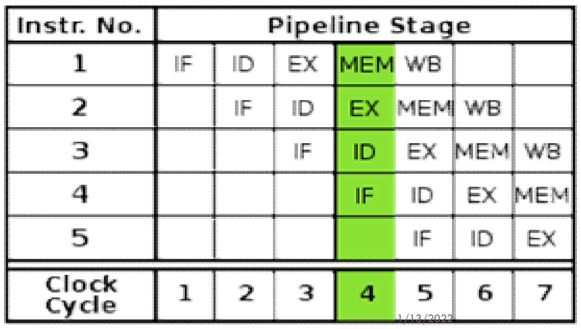
Pipeling

- Pipelining is an implementation technique where multiple instructions are overlapped in execution
- The computer pipeline is divided in stages
- Each stage completes a part of an instruction in parallel
- The stages are connected one to the next to form a pipe - instructions enter at one end, progress through the stages, and exit at the other end

Pipelining

- The canonical example of a pipelined processor is a RISC processor, with five stages:
  - instruction fetch, decode, execute, memory access, and write back

RISC processor

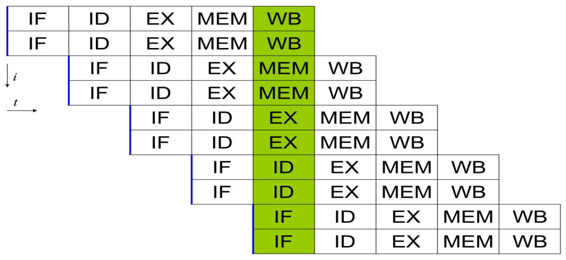


Instr. No.	Pipeline Stage						
1	IF	ID	EX	MEM	WB		
2		IF	ID	EX	MEM	WB	
3			IF	ID	EX	MEM	WB
4				IF	ID	EX	MEM
5					IF	ID	EX
Clock Cycle	1	2	3	4	5	6	7

Pipelining

- In addition to instruction-level parallelism, some processors can issue more than one instruction at a time.
- These are known as superscalar processors.
- Instructions can be grouped together only if there is no data dependency between them

Simple superscalar pipeline.



IF	ID	EX	MEM	WB				
IF	ID	EX	MEM	WB				
	IF	ID	EX	MEM	WB			
	IF	ID	EX	MEM	WB			
		IF	ID	EX	MEM	WB		
		IF	ID	EX	MEM	WB		
			IF	ID	EX	MEM	WB	
			IF	ID	EX	MEM	WB	
				IF	ID	EX	MEM	WB
				IF	ID	EX	MEM	WB

Simple superscalar pipeline

- A five-stage pipelined superscalar processor is capable of issuing two instructions per cycle.
- It can have two instructions in each stage of the pipeline, for a total of up to 10 instructions (shown in green) being simultaneously executed. (IF = Instruction Fetch, ID = Instruction Decode, EX = Execute, MEM = Memory access, WB = Register write back, *i* = Instruction number, *t* = Clock cycle [i.e., time])

Data Parallelism

13

- Data Parallelism means concurrent execution of the same task on each multiple computing core.
- Let's take an example, summing the contents of an array of size N.
- For a single-core system, one thread would simply sum the elements [0] . . . [N − 1].

1/13/2022

Data Parallelism

14

- For a dual-core system, however, thread A, running on core 0, could sum the elements [0] . . . [N/2 − 1] and while thread B, running on core 1, could sum the elements [N/2] . . . [N − 1].
- So the Two threads would be running in parallel on separate computing cores.

1/13/2022

Properties of Data Parallelism

15

- Same task are performed on different subsets of same data
- Synchronous computation is performed.
- As there is only one execution thread operating on all sets of data, so the speedup is more.
- Amount of parallelization is proportional to the input size.
- It is designed for optimum load balance on multiprocessor system

1/13/2022

Task parallelism

16

- Task parallelism is the characteristic of a parallel program that "entirely different calculations can be performed on either the same or different sets of data".
- This contrasts with data parallelism, where the same calculation is performed on the same or different sets of data.
- The word *task* in *task parallelism* is used in the general sense of an activity or job.

1/13/2022

Task parallelism

17

- Task Parallelism means concurrent execution of the different task on multiple computing cores.
- Consider the example above, an example of task parallelism might involve two threads, each performing a unique statistical operation on the array of elements.
- Again The threads are operating in parallel on separate computing cores, but each is performing a unique operation.

1/13/2022

Task parallelism Properties

18

- Different task are performed on the same or different data.
- Asynchronous computation is performed.
- As each processor will execute a different thread or process on the same or different set of data, so speedup is less.
- Amount of parallelization is proportional to the number of independent tasks is performed.
- Here, load balancing depends upon on the e availability of the hardware and scheduling algorithms.

1/13/2022

SHARED MEMORY and SHARED VARIABLES

19

- Depending on whether 2 or more processors can gain access to the same memory location simultaneously,
- we have 4 subclasses of shared memory computers

1/13/2022

SHARED MEMORY and SHARED VARIABLES

20

- **Exclusive Read, Exclusive Write (EREW) SM Computers**
- Access to memory locations is exclusive i.e. no 2 processors are allowed to simultaneously read from or write into the same location.
- **Concurrent Read, Exclusive Write (CREW) SM Computers**
- Multiple processors are allowed to read from the same location but write is still exclusive. .i.e. no 2 processors are allowed to write into the same location simultaneously

1/13/2022

SHARED MEMORY and SHARED VARIABLES

21

- **Exclusive Read, Concurrent Write (ERCW) SM Computers**
- Multiple processors are allowed to write into the same memory location but read access remains exclusive.
- **Concurrent Read, Concurrent Write (CRCW) SM Computers**
- Both multiple read and multiple write privileges are allowed.

1/13/2022

SHARED MEMORY and SHARED VARIABLES

22

- Allowing concurrent read access to the same address should pose no problems (except perhaps to the result of a calculation)
- Conceptually, each of the several processors reading from that location makes a copy of its contents and stores it in its own register (RAM)

1/13/2022

SHARED MEMORY and SHARED VARIABLES

23

- Problems arise however, with concurrent write access.
- If several processors are trying to simultaneously store (potentially different) data at the same address, which of them should succeed ?
- i.e. we need a deterministic way of specifying the contents of a memory location after a concurrent write operation.

1/13/2022

SHARED MEMORY and SHARED VARIABLES

24

- Some ways of resolving **write conflicts** include :-
  - Assign priorities to the processors and accept value from highest priority processor
  - All the processors are allowed to write, provided that the quantities they are attempting to store are equal, otherwise access is denied to ALL processors.

1/13/2022

SHARED MEMORY and SHARED VARIABLES

25

- It is only feasible to allow P processors to access P memory locations simultaneously for relatively small P (< 30)
- Usually because of the cost of the communication.

1/13/2022

Interconnection Networks

26

- We have seen that one way for processors to communicate data is to use a shared memory and shared variables.
- However this is unrealistic for large numbers of processors.
- A more realistic assumption is that each processor has its own private memory and data communication takes place using message passing via an INTERCONNECTION NETWORK.

1/13/2022

Interconnection Networks

27

- The interconnection network plays a central role in determining the overall performance of a multicomputer system.
- If the network cannot provide adequate performance, for a particular application, nodes will frequently be forced to wait for data to arrive.
- Some of the more important networks include

1/13/2022

Interconnection Networks

28

- Fully connected or all-to-all
- Mesh
- Rings
- Hypercube
- X - Tree
- Shuffle Exchange
- Butterfly
- Cube Connected Cycles

1/13/2022

Interconnection Networks -dynamic

29

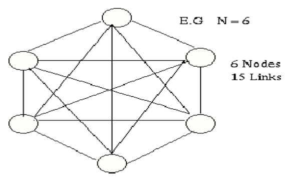
- Multi – Stage Interconnection network
- Cross – Bar Interconnection Network

1/13/2022

Fully connected or all-to-all

30

- This is the most powerful interconnection network (topology): each node is directly connected to ALL other nodes.



1/13/2022

Fully connected or all-to-all

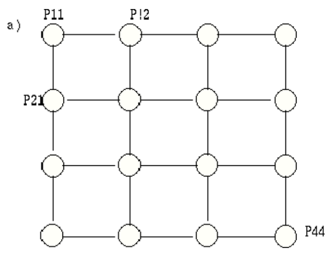
- Each node has N-1 connections (N-1 nearest neighbours)
- giving a total of N(N-1) / 2 connections for the network.
- Even though this is the best network to have,
- the high number of connections per node mean this network can only be implemented for small values of N.

Mesh (Torus)

- In a mesh network, the nodes are arranged in a **k dimensional lattice** of width w, giving a total of w^k nodes.
- Usually k=1 (linear array) or k=2 (2D array) e.g. ICL DAP.
- Communication is allowed only between neighbouring nodes.
- All interior nodes are connected to 2k other nodes.

Mesh (Torus)

EXAMPLE



2D mesh of width 4 with

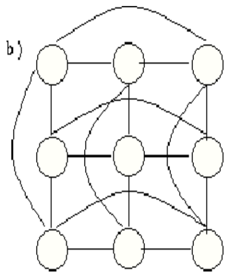
no wraparound connections

on edge or corner nodes

corner nodes have degree 2

edge nodes have degree 3

Mesh (Torus)



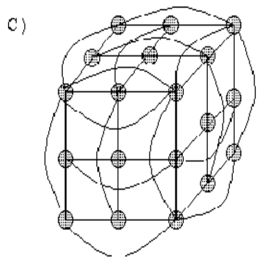
2D mesh of width 3 with wraparound

connections between nodes on same

row or column

Edge and corner nodes have degree 4

Mesh (Torus)



k = 3 w = 3

i.e. 3 ^ 3 = 27 nodes

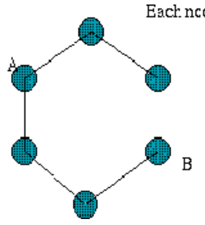
with wraparound connections

all nodes have degree 6 (2k)

Rings

- A **simple ring** is just a linear array with the end nodes linked.

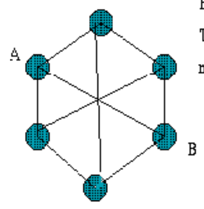
Each node has degree 2



Rings

- It is equivalent to a 1D mesh with wraparound connections.
- One drawback to this network is that some data transfers may require N/2 links to be traversed e.g. A and B above (3).
- This can be reduced by using a **chordal ring**
- This is a simple ring with cross or chordal links between nodes on opposite sides

Rings

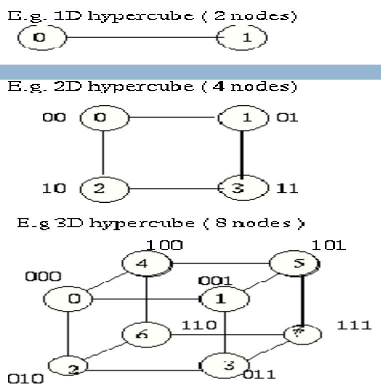


Each node has degree 3

Transfering data between A and B now requires 1 link

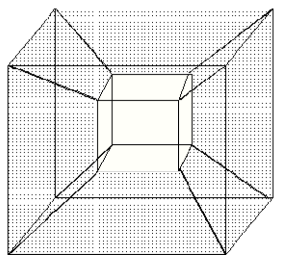
Hypercube Connection (Binary n-Cube)

- Hypercube networks consist of N = 2^k nodes
- arranged in a k dimensional hypercube.
- The nodes are numbered 0 , 1, ...2^k -1
- and two nodes are connected if their binary labels differ by exactly one bit


E.g. 1D hypercube (2 nodes)
0
1
E.g. 2D hypercube (4 nodes)
00
0
1
01
10
2
3
11
E.g 3D hypercube (8 nodes)
100
101
000
4
5
001
0
1
110
6
7
111
010
2
3
011

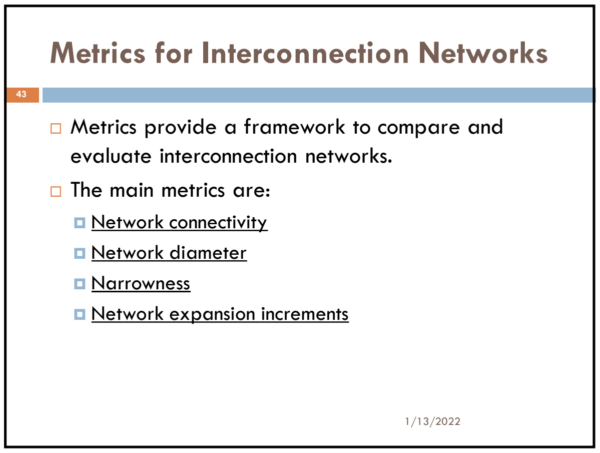
Hypercube Connection (Binary n-Cube)

4D Hypercube or Binary 4-Cube



Hypercube Connection (Binary n-Cube)

- K dimensional hypercube is formed by combining two k-1 dimensional hypercubes and connecting corresponding nodes i.e. hypercubes are recursive.
- each node is connected to k other nodes i.e. each is of degree k

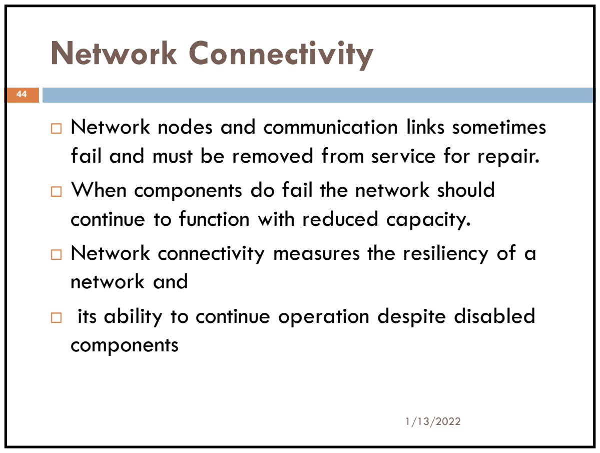


Metrics for Interconnection Networks

43

- Metrics provide a framework to compare and evaluate interconnection networks.
- The main metrics are:
  - Network connectivity
  - Network diameter
  - Narrowness
  - Network expansion increments

1/13/2022

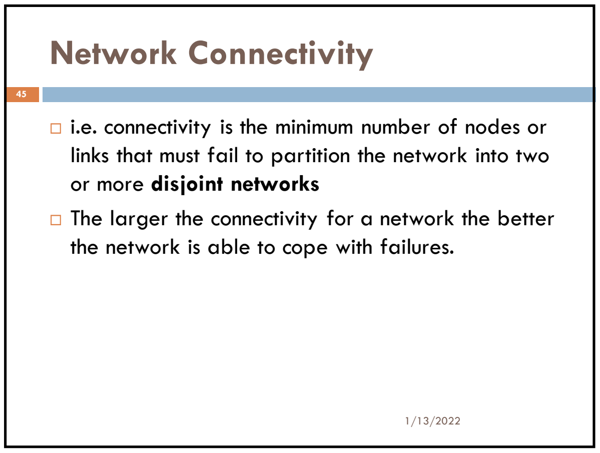


Network Connectivity

44

- Network nodes and communication links sometimes fail and must be removed from service for repair.
- When components do fail the network should continue to function with reduced capacity.
- Network connectivity measures the resiliency of a network and
- its ability to continue operation despite disabled components

1/13/2022

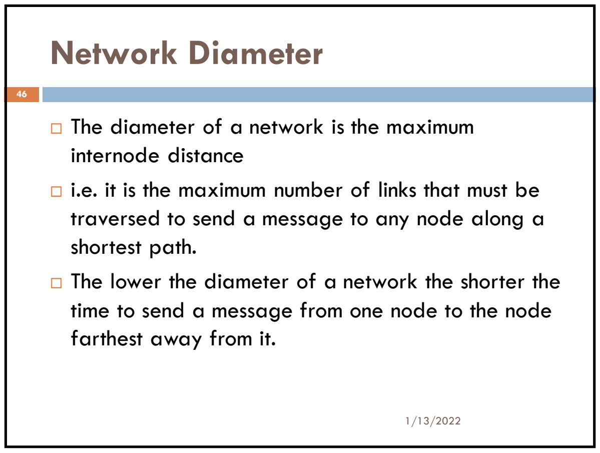


Network Connectivity

45

- i.e. connectivity is the minimum number of nodes or links that must fail to partition the network into two or more **disjoint networks**
- The larger the connectivity for a network the better the network is able to cope with failures.

1/13/2022

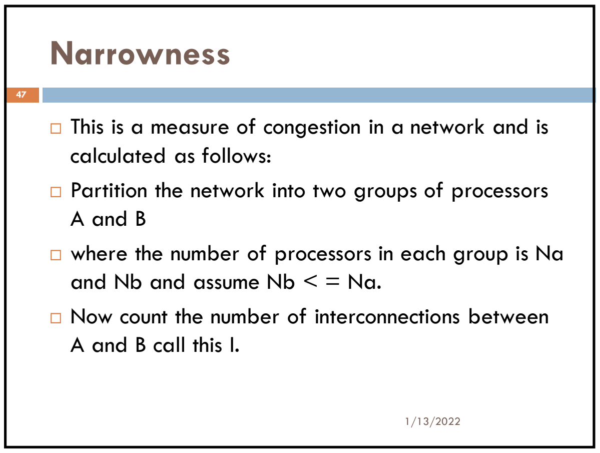


Network Diameter

46

- The diameter of a network is the maximum internode distance
- i.e. it is the maximum number of links that must be traversed to send a message to any node along a shortest path.
- The lower the diameter of a network the shorter the time to send a message from one node to the node farthest away from it.

1/13/2022

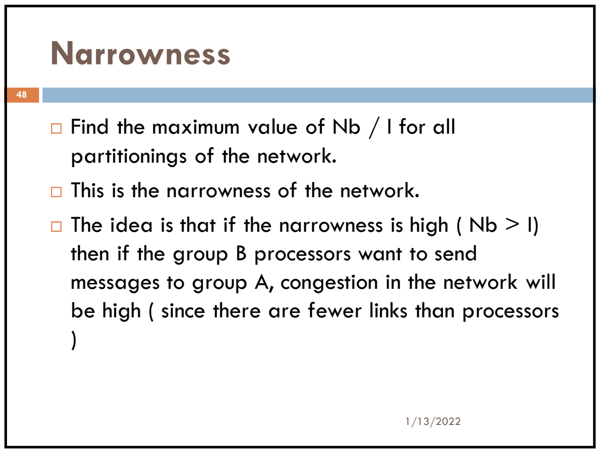


Narrowness

47

- This is a measure of congestion in a network and is calculated as follows:
- Partition the network into two groups of processors A and B
- where the number of processors in each group is Na and Nb and assume Nb < = Na.
- Now count the number of interconnections between A and B call this I.

1/13/2022



Narrowness

48

- Find the maximum value of Nb / I for all partitionings of the network.
- This is the narrowness of the network.
- The idea is that if the narrowness is high (Nb > I) then if the group B processors want to send messages to group A, congestion in the network will be high (since there are fewer links than processors)

1/13/2022

Network Expansion Increments

49

- A network should be expandable i.e.
- it should be possible to create larger and more powerful multicomputer systems by simply adding more nodes to the network.
- For reasons of cost, it is better to have the option of small increments since this allows you to upgrade your network to the size you require (i.e. flexibility) within a particular budget.

1/13/2022

Network Expansion Increments

50

- E.g. an 8 node linear array can be expanded in increments of 1 node but a 3 dimensional hypercube can be expanded only by adding another 3D hypercube. (i.e. 8 nodes)

1/13/2022

Other metrics

51

- Bisection bandwidth
  - the speed with which data from two halves of the network can be transposed across an arbitrary cut
- Cost
  - Proportional to the number of communication links

1/13/2022

52 1/13/2022