Networks

Networks are a vital part of **all** computer systems.

Distributed computing implies processing units which are geographically separated.

Chip
System Box
Computer Room
Sites

But networks are needed where ever information is communicated between two or more places.

processor – processor
processor – memory [cache] (banks)
cache – cache
I/O

Network topologies are vital to an understanding of computation at every scale.

Themes and techniques recur at every scale.

Brief overview

A piece of wire between two points is a network – albeit of a simple kind

Implications

Networks determine the scalability

- The ultimate size
- The ease of growth

Networks are a determinate of performance

- Speed
- Energy consumption

Networks (partially) control

- communication speed
- communication latency
- communication contention
- communication cost
- energy requirements for communication

Topology

How are the nodes (switches) connected.

Affects: growth
throughput
contention
distance

Routing

Message path source to destination

May be *Static* or *adaptive*

Buffering and Flow Control

Data storage in transit

Response to congestion (*throttling*)

Although presented as separate they are all closely coupled

Node
A processing or switching element in the network. May do both.

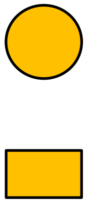
Link
Connections between two nodes.

Protocol
Agreed meaning to communication between nodes.

Each communication path ends in a **node**.
The node may by a network node:
    switch which only routes traffic.
It may be a computational node:
    routes traffic
    consumes messages
    initiates messages

Represented by some suitable shape

Normal trade-off is *performance v cost*



Performance
    Can have various meanings.
In practice often find mixed topologies

Interconnection topology

Wide range of application domains.
On chip or on board. Share features with memory
Across systems. Share features with storage & I/O

Concepts: *latency, bandwidth, queues.*

On chip networks

Interconnection of functional units: register files; cache; processor cores;
A few tens of devices and distances ~ cm.
Custom design: for instance propagation delay.
Peak speed in the Gbps

Storage area networks (SANs)

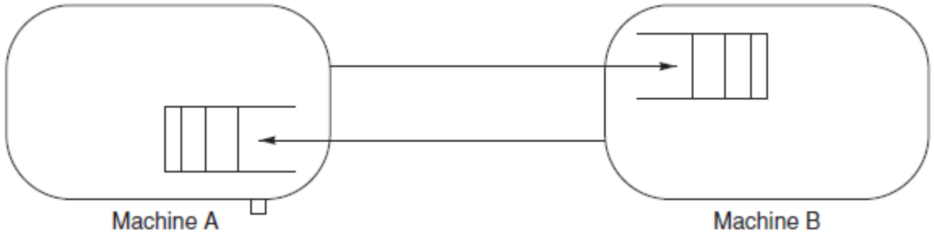
Interprocessor comms; storage; I/O inside data centres. Few thousand devices, 10s to 100s of metres. eg *inifiband* – 120Gbps

Local area networks (LANS)
Autonomous systems in a data centre and across campus. A few kilometres, thousands of devices and upto 100Gbps

Wide area networks (WANS)
Millions of computers; global; max at 10Gbps

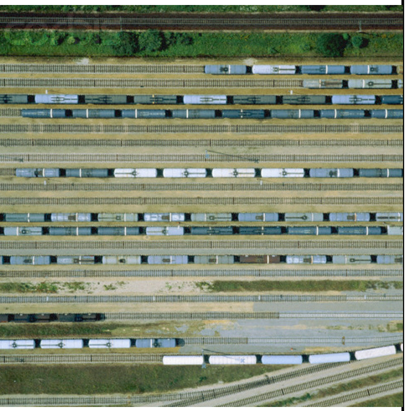
Look at the general properties of networks and the things which limit their performance

Pairwise



Request A to B: address of data in B
Reply B to A: of requested

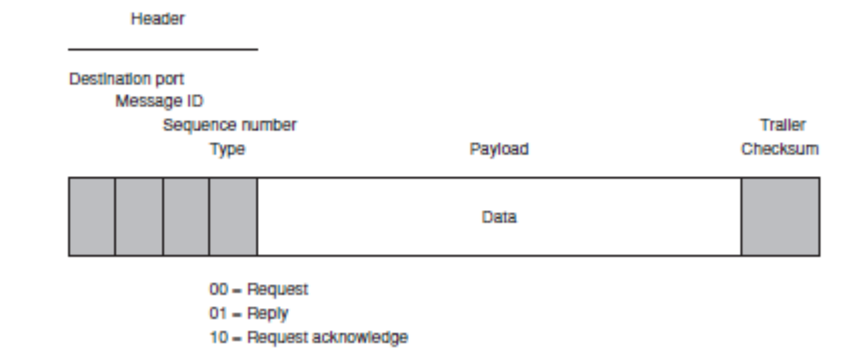
The contents are known as the payload.
Other bits are control bits which define the message type and destination.
The creation and interpretation of the message (marshalling and un-marshalling. Is performed by the *network interface*
May be a dedicated card/board or built into the processor chip.



Hardware as far as possible, but some software required. Split long messages up – requires *sequence ID* for re-assembly in the correct order.

Pairwise (i)

The definition of the steps for communication is the *communication protocol.* Includes description of packet.



To distinguish between different processes on the machine a port is included.
Checksum will ensure the message has not been corrupted in transfer.
Speed – bypass OS and give network interface direct access to memory (*message buffers).*
Protocols known as zero-copy protocols.

Normally an acknowledge (ACK) is sent. Then sending computer will discard message. Automatic resend after timeout

Auto resend – loss in transit.

Flow control

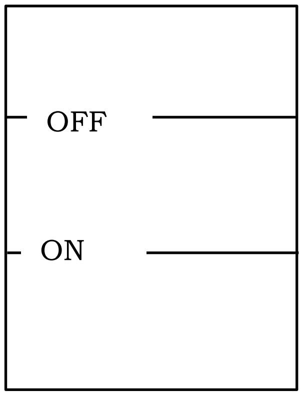
Two common protocols
Xon/Xoff – and credit based

Xon/Xoff: receiver sends transmit as buffer empties, pause as buffer fills. Either special packets or control wires.

Credit: sender starts with tokens – decrements on send. Receiver returns credits as messages taken out of buffer.

Credit needs smaller buffers – Xon/Xoff must be larger enough to take messages in flight.
Xon/Xoff generates fewer control messages.
To saturate bandwidth the buffer needs to be large enough – depends on trip time and hence distance.

Also Ack/Nack upstream optimistically sends downstream – buffer not dealloc until ack/nack received. Inefficient use of buffer space



Buffer

Losses

Networks with flow control are “lossless”
Lossy ones need to retransmit packets, wasting bandwidth.

Use … time protocol doesn’t mind if packets are dropped.
Answers to FEA must be returned.

Worry about garbled packets checksum.

Handle duplicate packet transmission

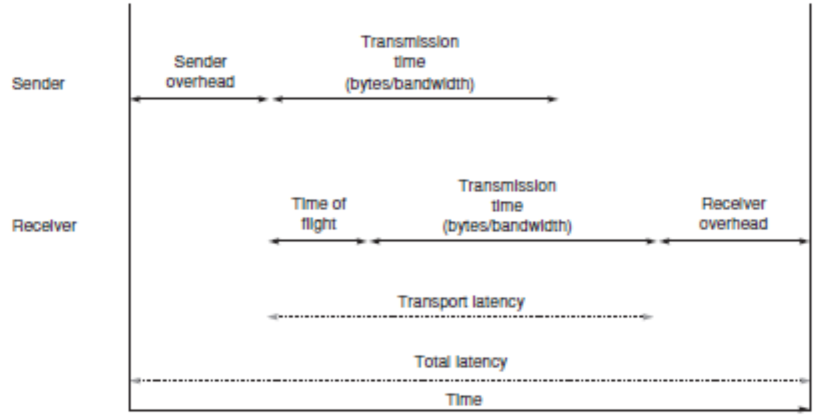
In general
Local – lossless
Wide - lossy

example

Speed 1Gbps
Distance 200m
Packet size 1kb
What is the credit number and hence buffer size to saturate the line.
Speed of light 300,000 km/sec. Transmission speed 2/3 c.
Propagation delay 100/200,000,000s = 1 microsec.
Round trip is 2 microsec.
2 microsec transmit 2kb.
So that is 2 packets – so we need 2/3credits and a buffer size of 2/3kb.
Clearly scales with distance
Xon/Xoff – needs same amount of space from Xoff to top of buffer.
But when it sends the switch on there must be at least enough in the buffer to last until the next package arrives, so same amount Xon to empty.
Hysteresis to stop the link flapping. So gap between Xon/Xoff levels

Ignore other sources of delay. Switches/routers

Terminology



Bandwidth in network terms is bits/second sustained flow.
(More normally range of frequencies for which the attenuation per unit length is less than some fixed value)

Time of flight – how long it takes a bit to go from sender to receiver.
Transmission time – now long it takes to launch a packet. ie size/bandwidth.
Transport latency – time of flight plus transmission time
Sending overhead/receiving overhead

Includes passage through routers

Congestion delays

Independent of data size

Proportional to data size

> two devices

Joining greater than two devices involves more complex arrangements.

Especially for LAN and WAN the connections may not be permanent. Pair wise connection of every possible pair of devices would be prohibitively expensive and wasteful.

Mechanism to get packet to its correct destination is **routing**.
May calculate route at start – do partial routes at intermediate points, or even deliver everywhere and let the receivers decided on relevance.

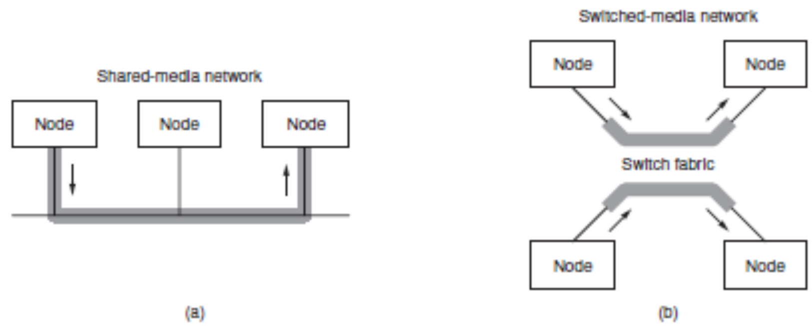
Routing
Arbitration
Switching
Required for all non trivial networks

Two packets want to use the same path
Arbitration – switching,

Independent of data size

Proportional to data size

Shared or switched



Ethernet was originally shared. Now more normally switched.
To improve throughput for shared need to upgrade all the fabric, for switched only need high speed switching.

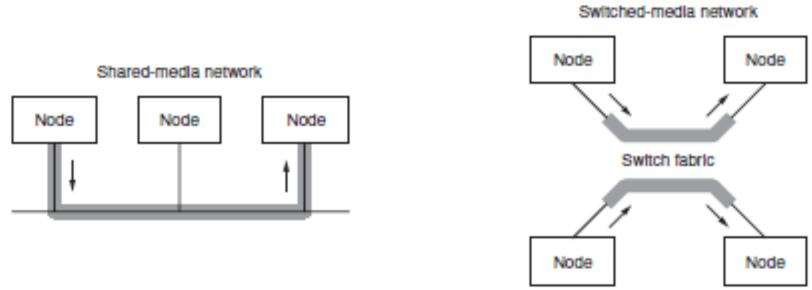
Shared

Carrier **S**ense-**C**ollision **D**etect

random backoff

Once arbitration is performed the rest is easy. No switching and broadcasting is just sending a message with something in the header saying it is for everyone. (multi-casting – same but to a subset)

Shared can lead to network failures if a single node does something illegal.



(a) (b)

Switched

Point to point links between switch components.
Passive and active components.
Extra layer of control – who gets the path in the case of contention.

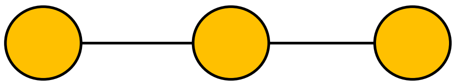
Comparison:

Shared do not scale with number of endpoints.
Adding nodes can be difficult.
Adding nodes adds to the parasitic capacitance and may slow down the network.
Contention means that the network degrades long before 100% occupancy is reached. Collisions waste bandwidth.

Concepts

Network concepts

Degree: Number of links per node
Link: Communication channel between two nodes
Hop: communication between a pair of nodes is a hop



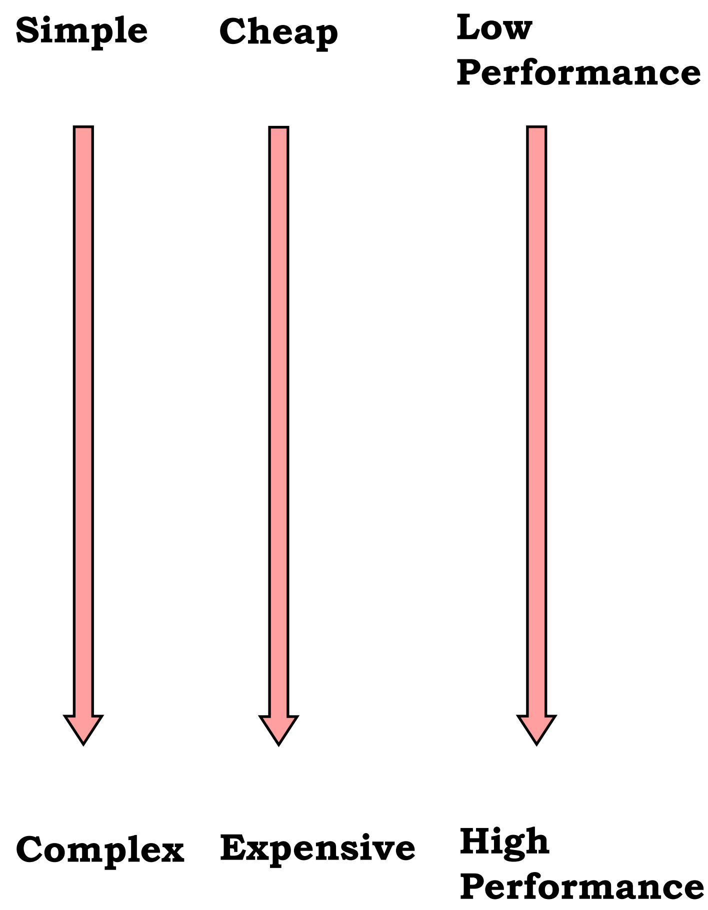
Two hops. Communication time is often counted in hops. Assumes that links take equal time and/or time is dominated by routing at node. Analysis often assumes that bandwidths are equal on all hops

Source: produces messages
Sink: consumes messages
Diameter: longest distance between two nodes in the network
Cost: Number of links and switches
Bisection width: minimum number of wire cuts to divide the network in two halves. (resilience). Use with care
Non blocking: each pair of independent source and sinks has a separate (disjoint path).
Blocking: some paths between pairs of nodes may conflict.
Topology: How the switches are wired
Routing: how the message gets from source to destination – static or adaptive
Channel: a single logical connection between nodes

Topology

How are the nodes (switches) linked together

Bus
Ring
Crossbar
Hypercube
Torus
Omega
Mesh
Butterfly
Point to Point



Routing how the switches are set

Performance

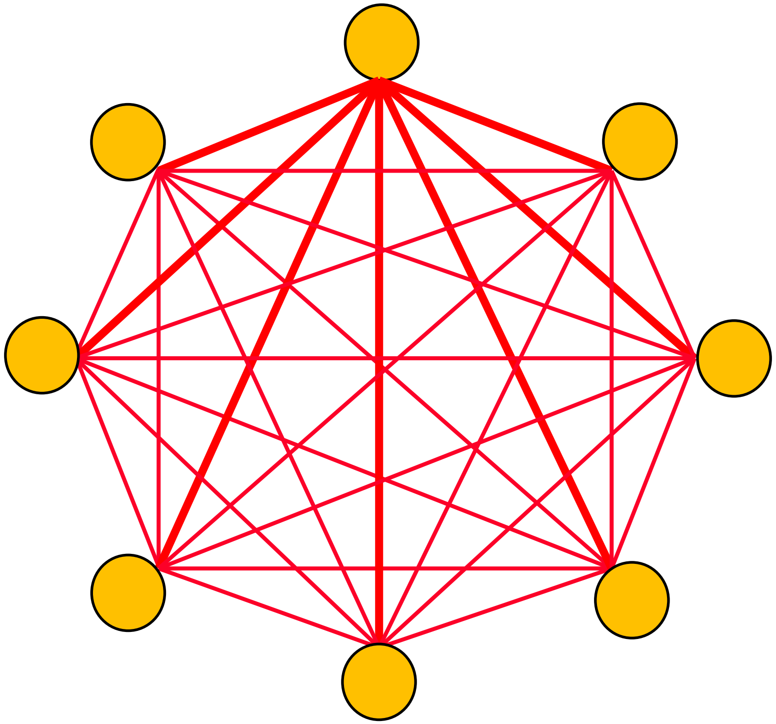
Latency – in hops or nanoseconds.
Often assumed time proportional to number of hops
Assumes that links take equal time and/or time is dominated by routing at node. Analysis often assumes that bandwidths are equal on all hops

Latency = sending overhead + Propagation time + switching time + arbitration time + routing time + transmission time + receiving overhead.
(Lower bound in the absence of contention)

Contention:
how many messages can be simultaneously transmitted

Fully connected

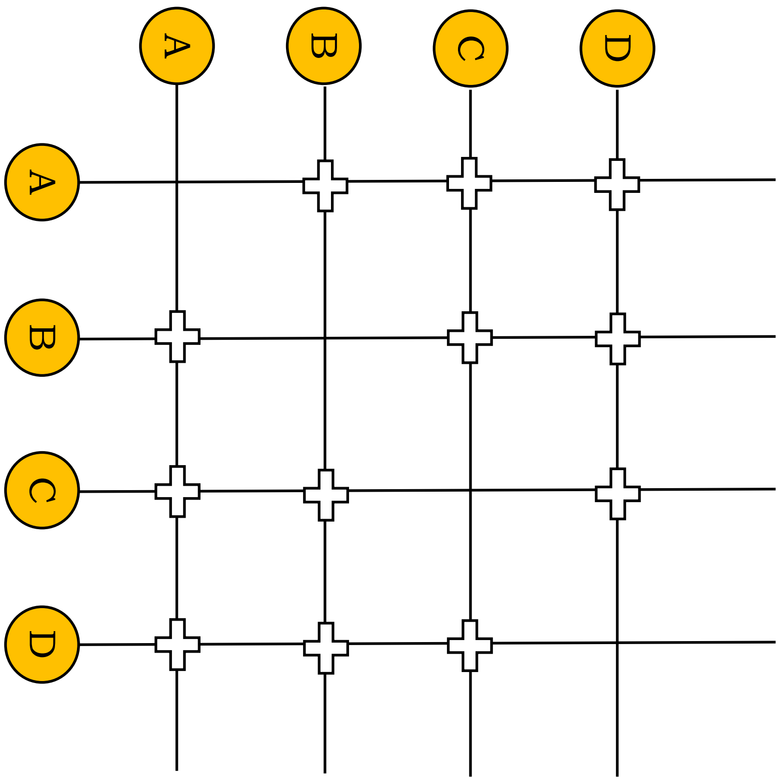
Point to Point (Complete):
Lowest contention: Multi-port nodes means that single node could communicate with more than one partner.
Latency: lowest (in principle)
Cost: highest
Connections/node: O(N)
Links: $O(N^2)$
Doesn't scale:
Non Planar: hard to fit on a chip



Crossbar

Earth simulator: high performance system (Top500) had a 640 node crossbar.

Crossbar: non blocking, concurrent transfers to many pairs, one hop connection between any pair. Excellent performance – very expensive – goes as the square of the number of machines. Low latency – high bandwidth. $O(N^2)$ links and thus cost.
Bus arbitration becomes hard as N increases.
Used in core to cache-banks



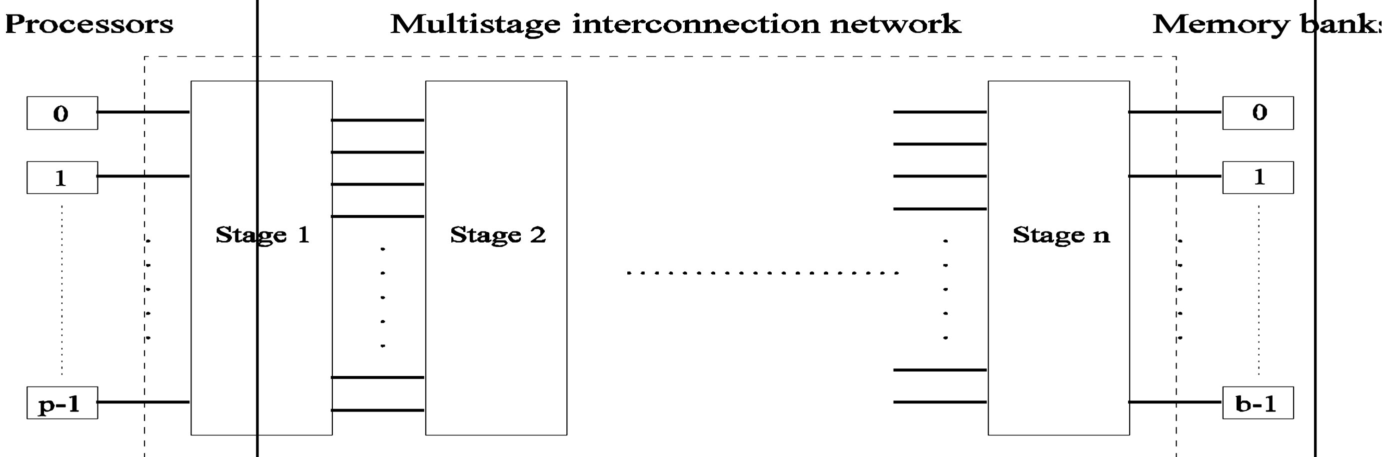
Compromise between bus and full.

Multistage

Multi-stage: A compromise between bus and crossbar

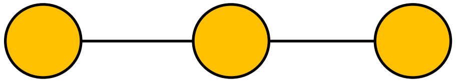
Partially blocking - better scaling than bus – less costly than a cross-bar

Cost: O(NlogN)
Latency: O(logN)



Dynamic v static

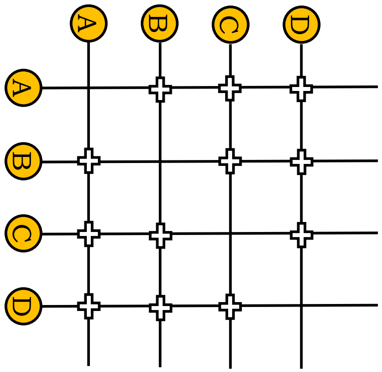
Static: where routing connections are frixed and permanent



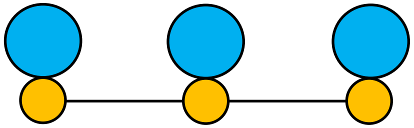
direct

Dynamic: where the connections vary in response to requirements.

indirect



Direct: endpoints (computational units) sit inside the network
Indirect: computational units sit at the end points and do no routing.



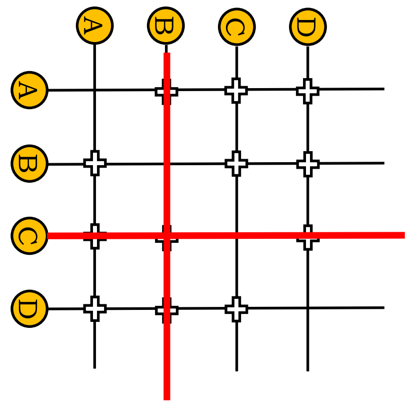
May choose to show switch and computational element separately.

Circuit v Packet

Packet Switching: routes each packet
think exchanging letters with a friend.

Route each packet individually (internet)
Only blocking during packet transit

Slower – most dynamically swtich
But no setup and tear down time
Flexible – full use of links
Circuit Switching: sets up full path
Used to be phone system
Establish route
send data
Exclusive use … route blocked.
Faster arbitration
Link set up and tear down takes time



topology

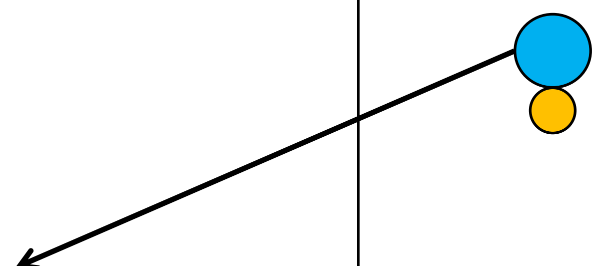
Bus: cost effective for small number of nodes. Easy to implement coherence (snooping) high contention. Large number of nodes leads to electrical loading, reduced frequency and bandwidth

Crossbar: goods for small numbers, low latency and high throughput. Expensive O(n^2). Difficult to arbitrate – core to cache bank networks

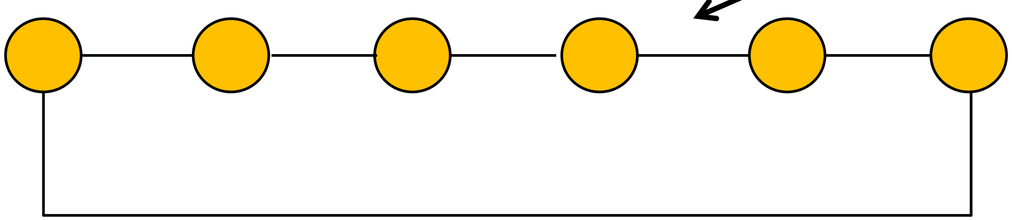
Linear Array: Cheap O(N) – high latency O(N). Easy to add to – doesn't scale well. Bisection 1. N/2 average hops.

Ring: Cheap O(N) – high latency O(N). Not easy to scale. Bisection 2. A linear array that loops back. N/4 average hops if bidirectional

Separate routing possible

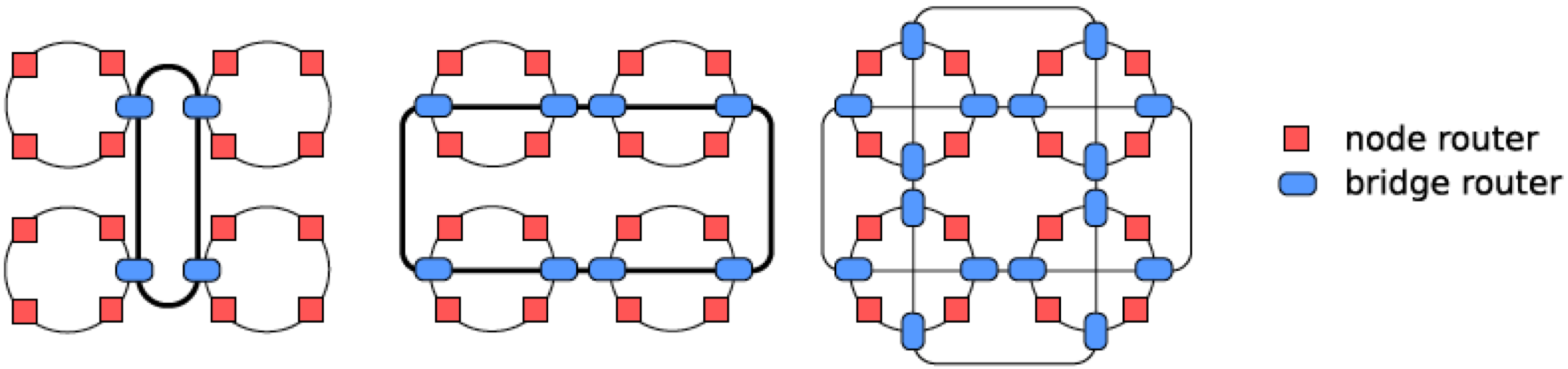


Linear Array



Ring

Hierarchical rings



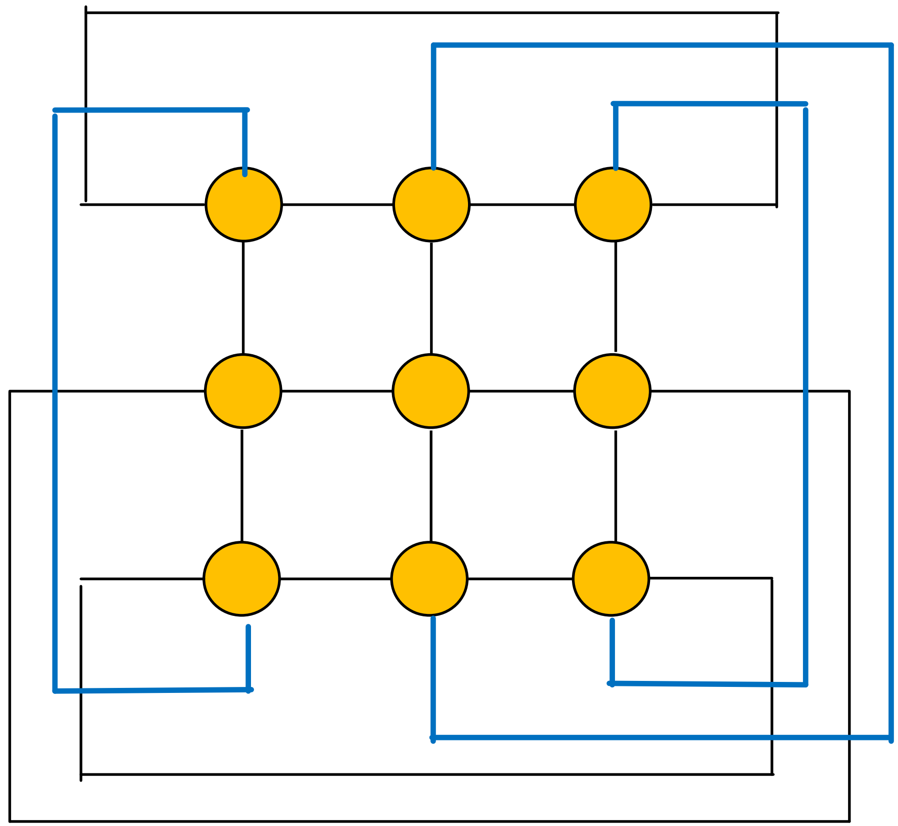
(a) 4-, 8-, and 16-bridge hierarchical ring topologies.

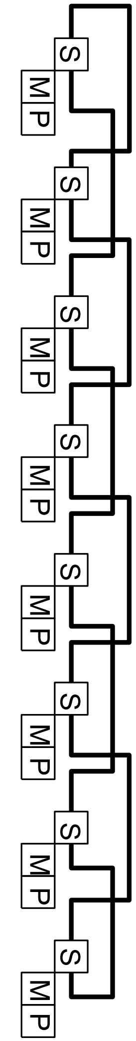
More complex.
Lower latency.
Efficiency – depends on problem. Good local connectivity. Slower long range

topology

Mesh (2D): O(N). <Latency> = O(sqrt(N)). Redundant pathways. Easy to layout on chip. Multi-path:reduced blocking

Torus (2D): Mesh not symmetric – performance sensitive to placement – edge v middle. Torus is better, harder to bisect, more path diversity. Higher cost, hard to layout. Unequal link lengths. % extra links drops with size. Hard to grow.

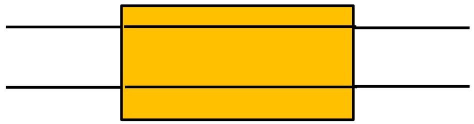




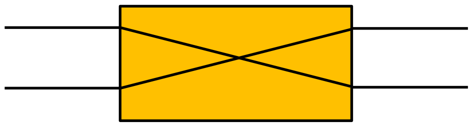
Path weaving to equalise link lengths

Multistage: comes in a number of types
Omega network
Grows as p*log(p)
log(p) stages – p = number inputs = number outputs
Individual switches

Straight through



Cross over



For all switches cost grows as the product of the number of inputs and the number of outputs

CLOS

C Clos 1953 Bell Labs

Telephony – 3 stage switched network N to N

1st stage N inpus are broken into n groups. Best if N/n is some integer k

2nd stage – switches k to k – so k connections

3rd stage – reverses stage 1 to k. k x n switches

Can extend by replacing the second stage by a Clos network

Clos showed if k <= 2n-1 … this is non-blocking

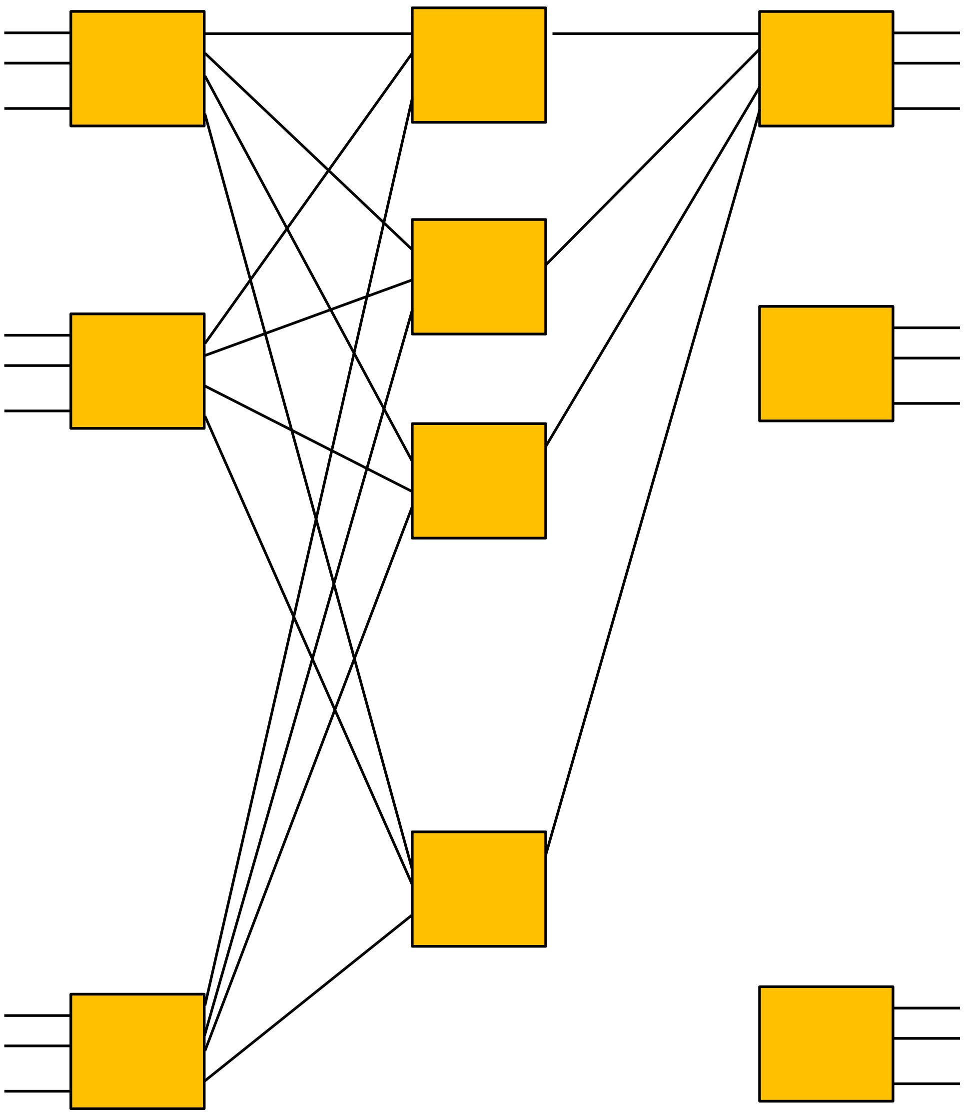
Also that the minimum number of switches required for n is approx sqrt(N/2)

Blocking
When a connection is made it can exclude the possibility of certain other connections being made

Non-blocking
A new connection can always be made without disturbing the existing connections

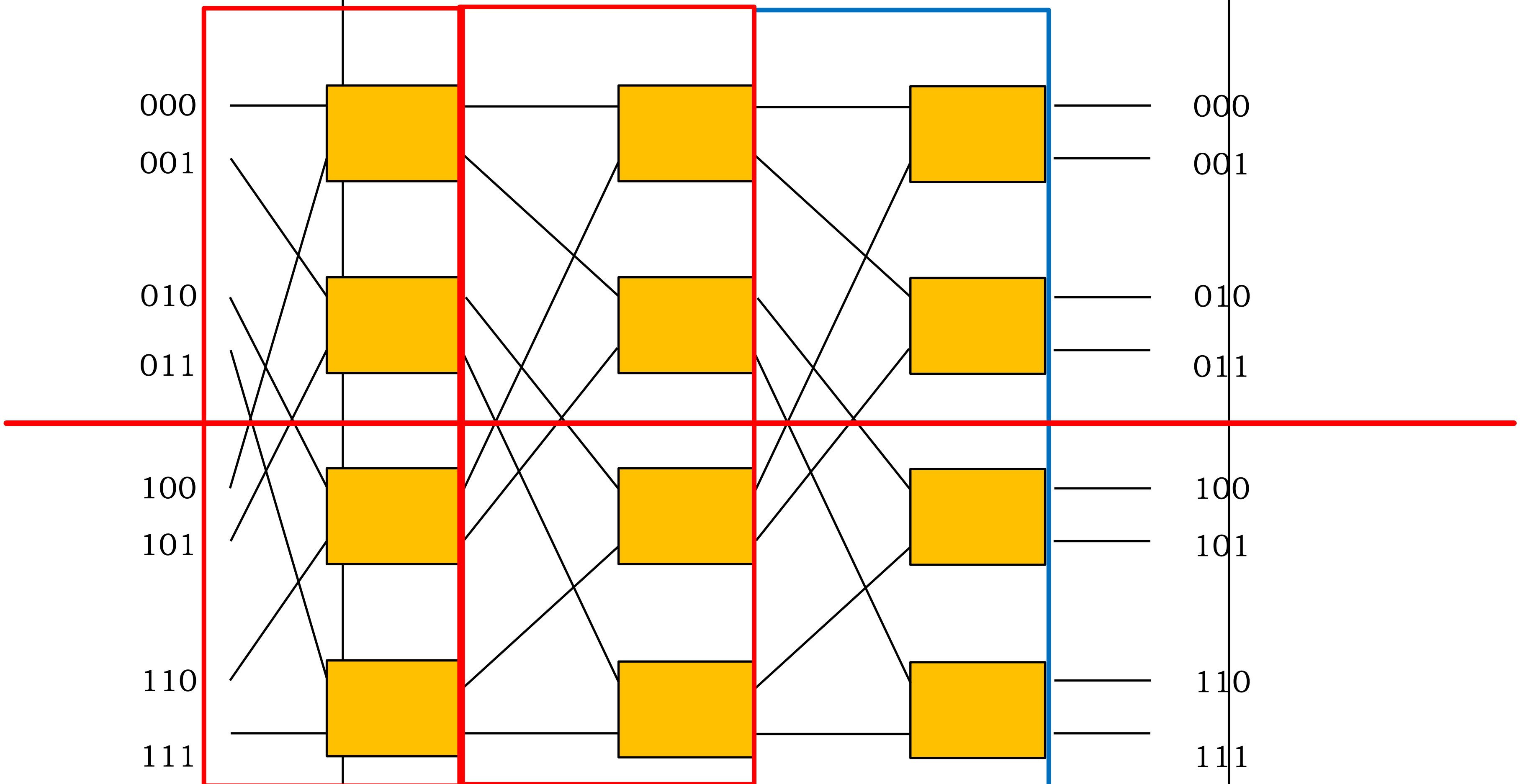
Rearrangeably non-blocking
A new connection can be made but it might be necessary to reconfigure some other connections on the switch

CLOS iii



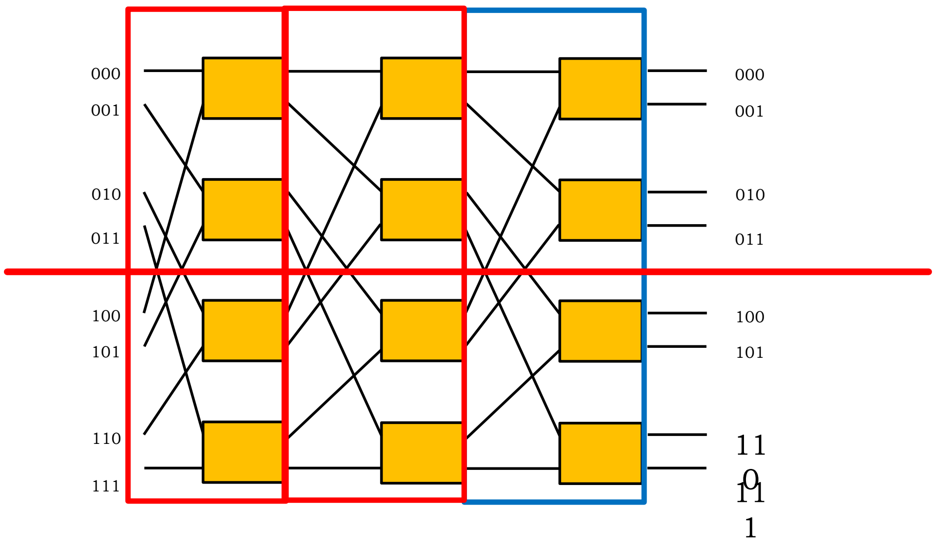
Omega Switch

Omega



Each coloured box is exactly the same
Connecting inputs to output in a perfect shuffle (left rotation
001 → 010 → 100 → 001 → 010
011 → 110 → 101 → 011 → 110 etc

Omega

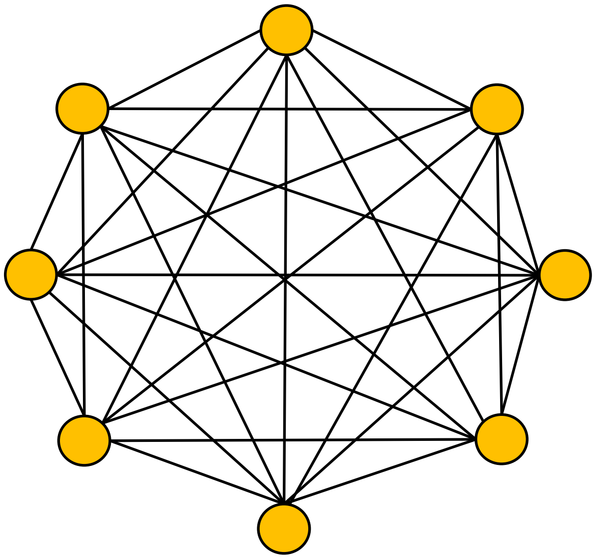


Routing is easy
Write the destination address in binary and prepend.
pop first digit – if 0 take top exit – if 1 take bottom
route message
pop first digit – if 0 take top exit – if 1 take bottom
route message
Repeat until destination reached.

But is that a straight or cross-through– by comparing
Bits of the source and destination.
If first bit of s = first bit of d – straight through
If different cross-over.
Remove first bit and forward

Completely connected:

Latency: 1 Cost $O(N^2)$



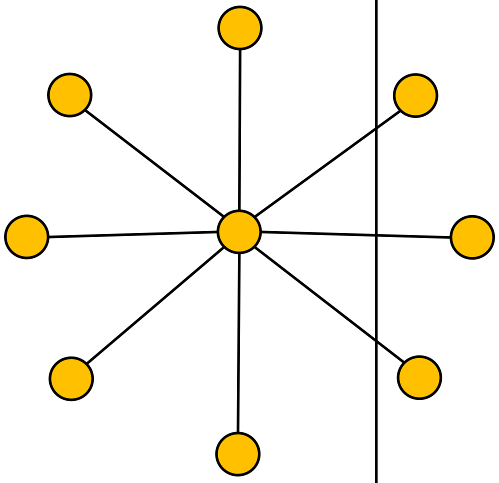
Star

Static equivalent of a bus

Single point of failure.

Blocking

Short path length

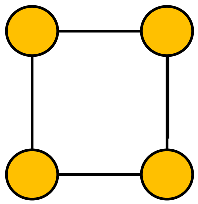


topology

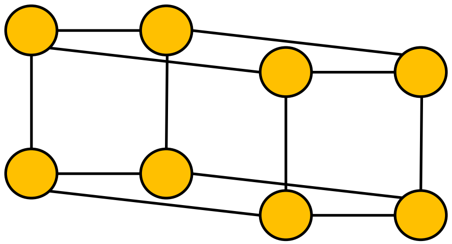
Latency: O(logN)
Links: O(NLogN)
Hard to layout in 2D.

But 6D cubes have been used in practical machines

Cube:
2D cube

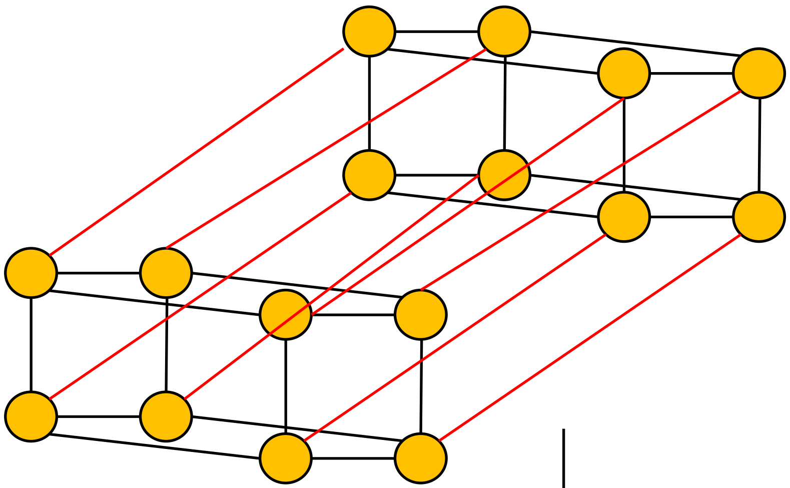


3D cube
Replicate and join
Corresponding corners



4D cube (hypercube)
Replicate and join
Corresponding corners

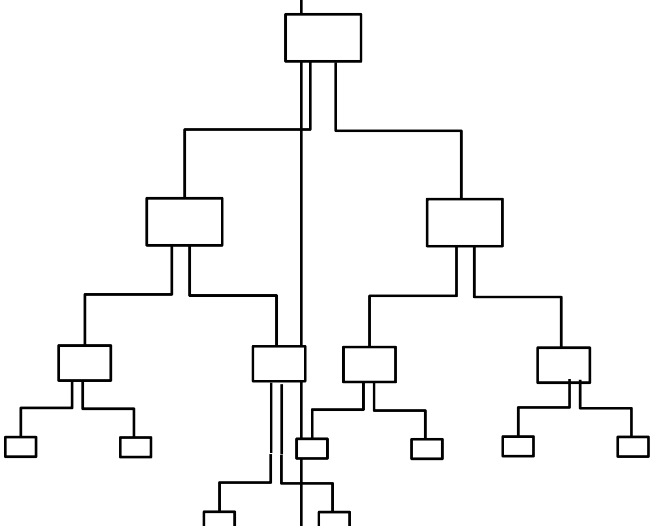
5Detc.

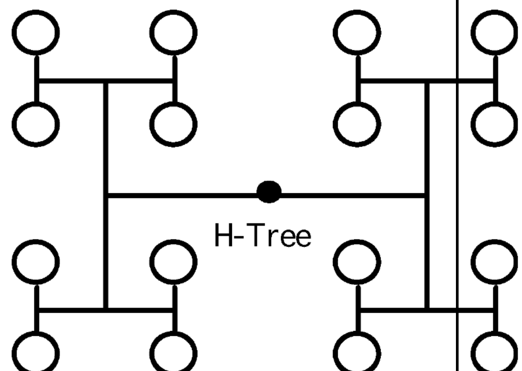


Diameter increases slowly
Bi-section increases – alternate paths increase and so blocking decreases.
Number of connections per node = dimension
Fixed number of nodes for a given dimension

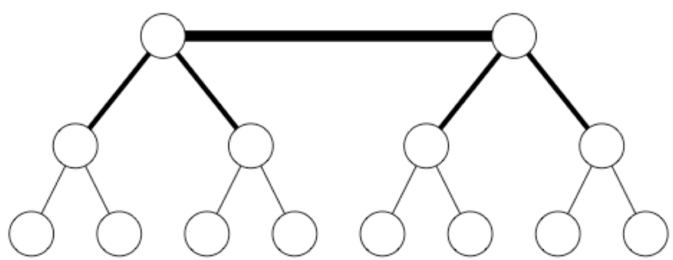
Tree

Trees: planar, hierarchical. Good for local traffic.
Latency O(logN) – cost the same. Easy to layout.
Root can be a bottleneck – and failure point.
Fat Tree to solve.





Fat tree



Binary Tree

Diameter = $2^{n-1}-1$

Bisection is 1 – almost no resilience.

Some simultaneous transfers possible.

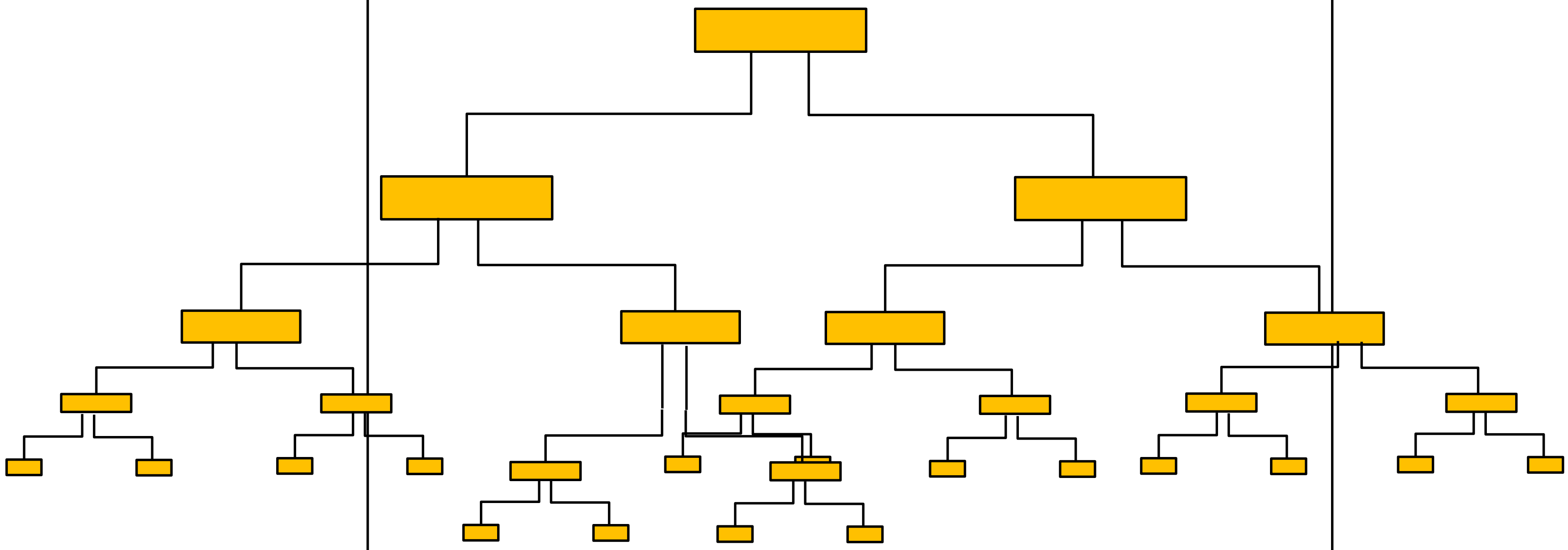
But clearly nothing crossing the root node

Easy to grow (at least in principle)

Mean number of hops

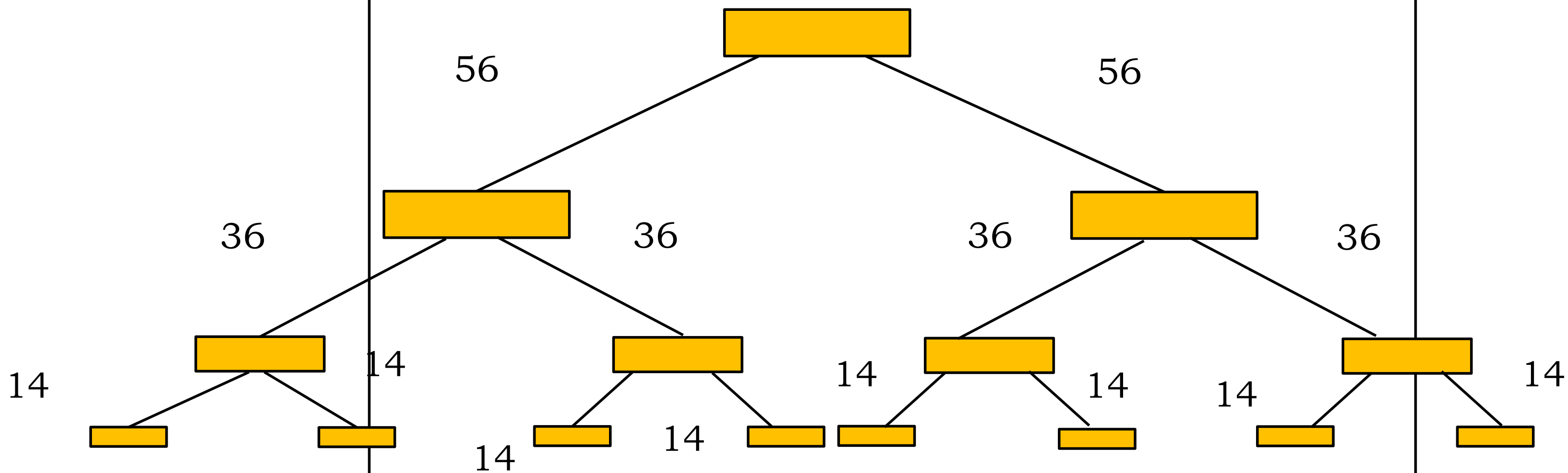
n=1 hops 0. n=2 hops 1.33; n=3 hops 2.29

Clearly if only nearest neighbour communications are important this architecture may be effective



Connections

How many machine-machine paths use each connection?
We see that the lower levels are used for less paths.
Improvement would be to increase the number of connections

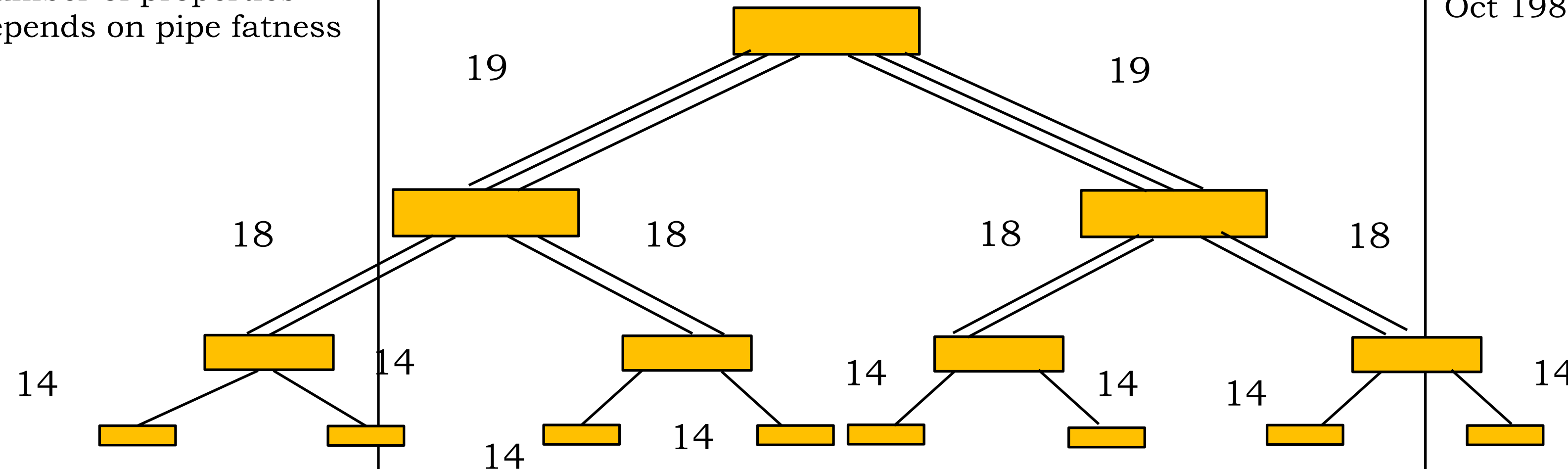


Fat Tree

Prop	Tree	Fat tree
Diam	6	6
Links	14	22
Bisect	1	3

Number of properties depends on pipe fatness

How many machine-machine paths use each connection?
We see that the lower levels are used for less paths.
Improvement would be to increase the number of connections.
Increase the number of simultaneous connections possible. Number of paths per link shows less variation.
Reduces contention – any leave can talk to any other leave on the same half of the tree.
Bisection number is greater – need to lose at least 3 links.



It has been shown as long ago as 1985 that "for any given amount of communication hardware a fat tree can simulate every other network built from the same amount of hardware, using only slightly more time."

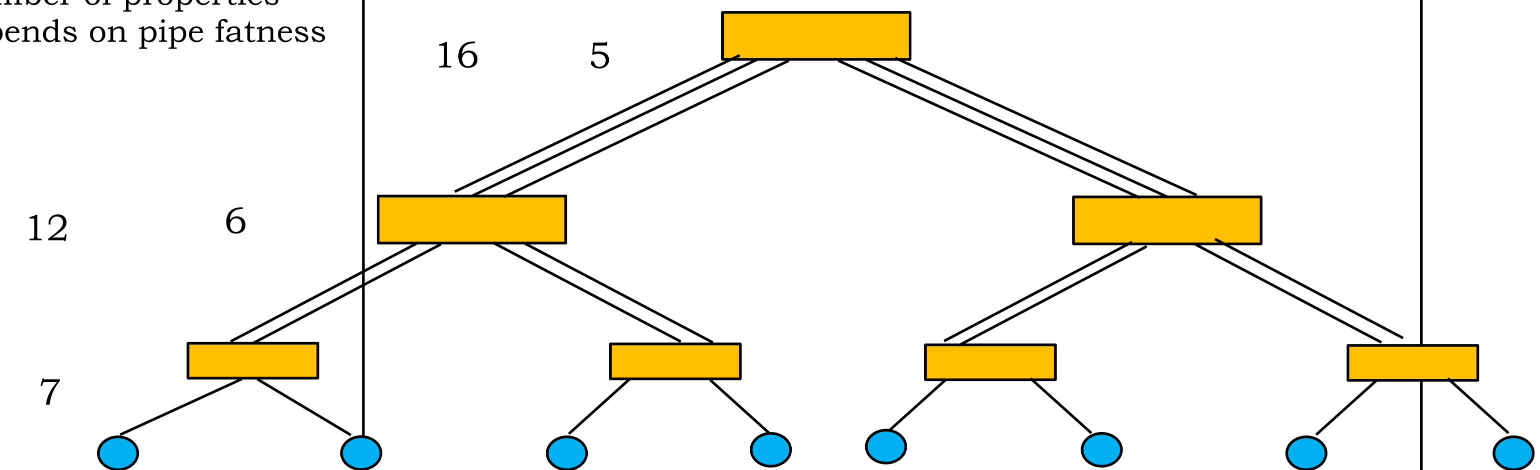
IEEE Transactions on Computers C-34,10, Oct 1985

Non-computing nodes

There is no definition of the increase in the numbers on every stage.
Here we have gone 1, 2, 3 and are actually reducing contention in the higher links – although with the same topology the higher links actually become more crowded if every node computes.
Higher links can be higher bandwidth rather than just more connections. Factor two increase means two lower links can share a higher one

Prop	Tree	Fat tree
Diam	6	6
Links	14	22
Bisect	1	3

Number of properties depends on pipe fatness

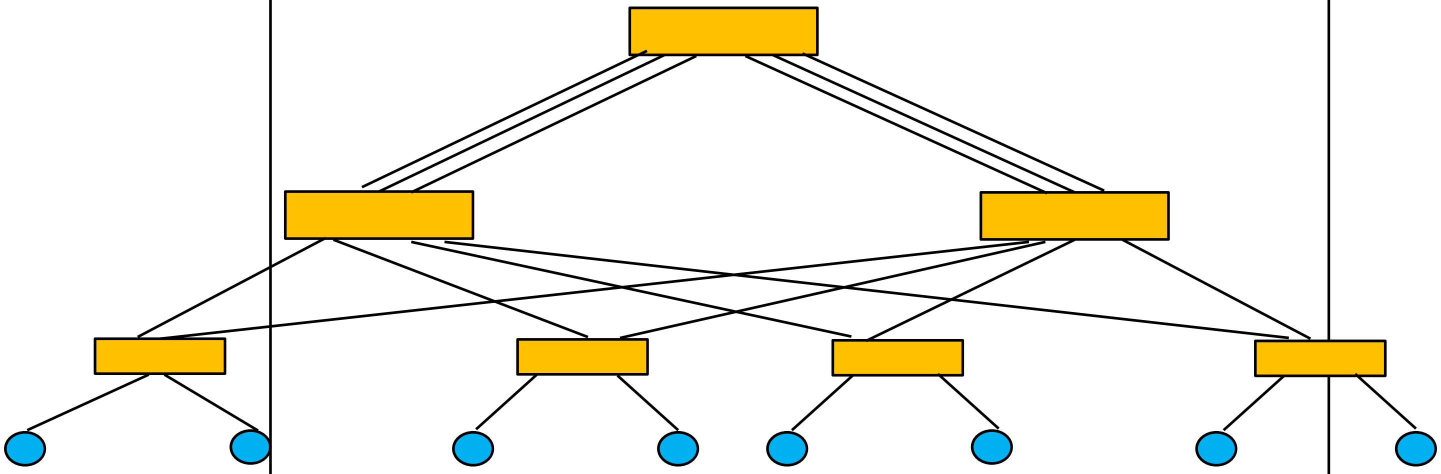


Re-wire

Prop	Tree	Rootless
Diam	6	4
Links	14	16
Bisect	1	2

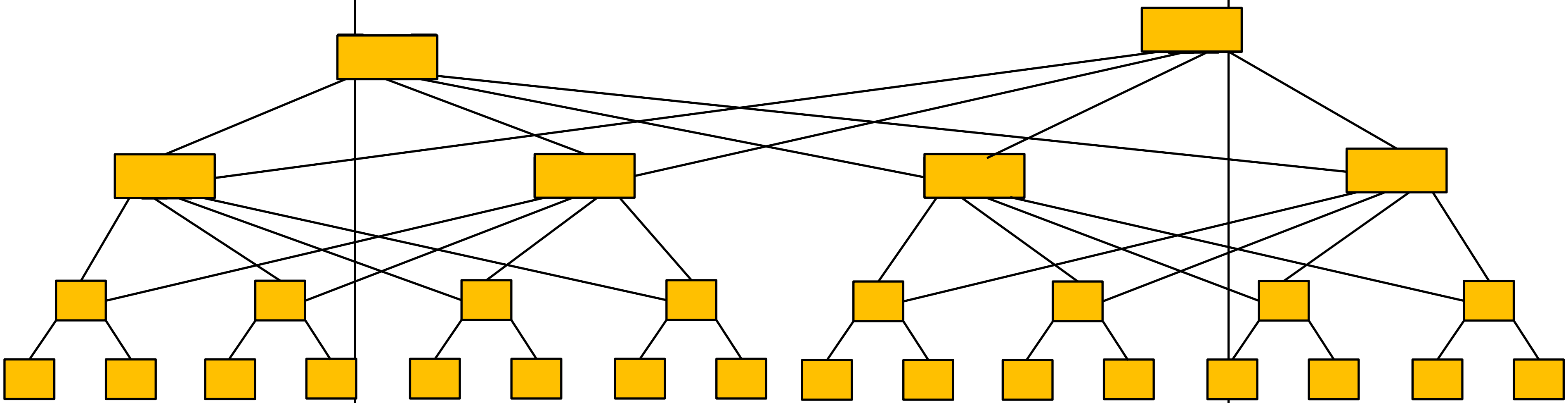
Can re-wire the fat tree.
But then the root node looks superfluous.

By removing the root node we
- reduce the diameter
- reduce the number of links
- but also reduce the bisection number



Re-wire

If the top layer here is allowed to have 4 connections per node, we can double the number of leave nodes, without increasing the tree depth.



Terminology

Terminology comes from graph theory. Network is a graph G
Processors are vertices (V) and channels are edges (E).

Number of nodes is the Cardinality.

Set of edges E linking processors e = (u,v) where u and v are two processors.
(u,v) can be bi-directional or uni-directional – so (u,v) can be different from (v,u)

Degree d(u) of node u is the number of connections.
in-degree can be different from out-degree.

Diameter: if the minimum distance between two nodes is d(u,v), then the diameter of the network is the maximum value of d(u,v) over all pairs.

Latency is the total time to send a message (including overheads).
Bandwidth is bits per unit time

Bisection width is the number of breaks to split the graph in half (approximately).

topology

Network	Diameter	Bisection width	Cost
Completely connected	1	p*p/4	p(p-1)/2
Star	2	1	p-1
Complete Binary	2log((p+1)/2)	1	p-1
Linear array	p-1	1	p-1
2D mesh	2(√(p) -1)	root(|p)	2(p- √ p)
2DTorus	√ p	2* √ p	2p
HyperCube	log(p)	p/2	(p log p)/2

Routing

Routing algorithms ensure that packets get from source to destination in the shortest time possible, while spreading traffic to minimise contention.

Two problems

Livelock

Onward path is calculated by node. Route round used link – to ensure speed. If only the state of the local links is known this can lead to packets bouncing between nodes.

Can insist on only shortest path – contention problems.

Can give a maximum number of hops. Internet uses that method.

Deadlock

Resource conflicts – handled by

deadlock avoidance: stop a deadlock forming

deadlock recovery: detect and recover

Routing is normally a combination of information in the header, defined at source.

Actions at intermediate nodes.

Arbitration

Who gets the resources

Need to avoid starvation – ensure that all messages are delivered.

Distributed arbitration avoids bottlenecks which can form with centralised arbitration.

Buffering

Do you provide storage space in the intermediate nodes to hold packets which can't be immediately be delivered.
More expensive – but no need to drop packets.

For no buffer and no link can use **deflection routing**. Where the nodes can switch more than one message at a time.
Incoming message wants to use an outgoing link which is already used. Route it the wrong way and let the downstream node sort it out. Uses the network itself to buffer traffic.

Switching

circuit switching:
Route is set up before the first packet is sent. Very efficient in terms of setup. Good for a continuous flow of data. Takes some bandwidth out of the pool even when not in use. Guaranteed connection.

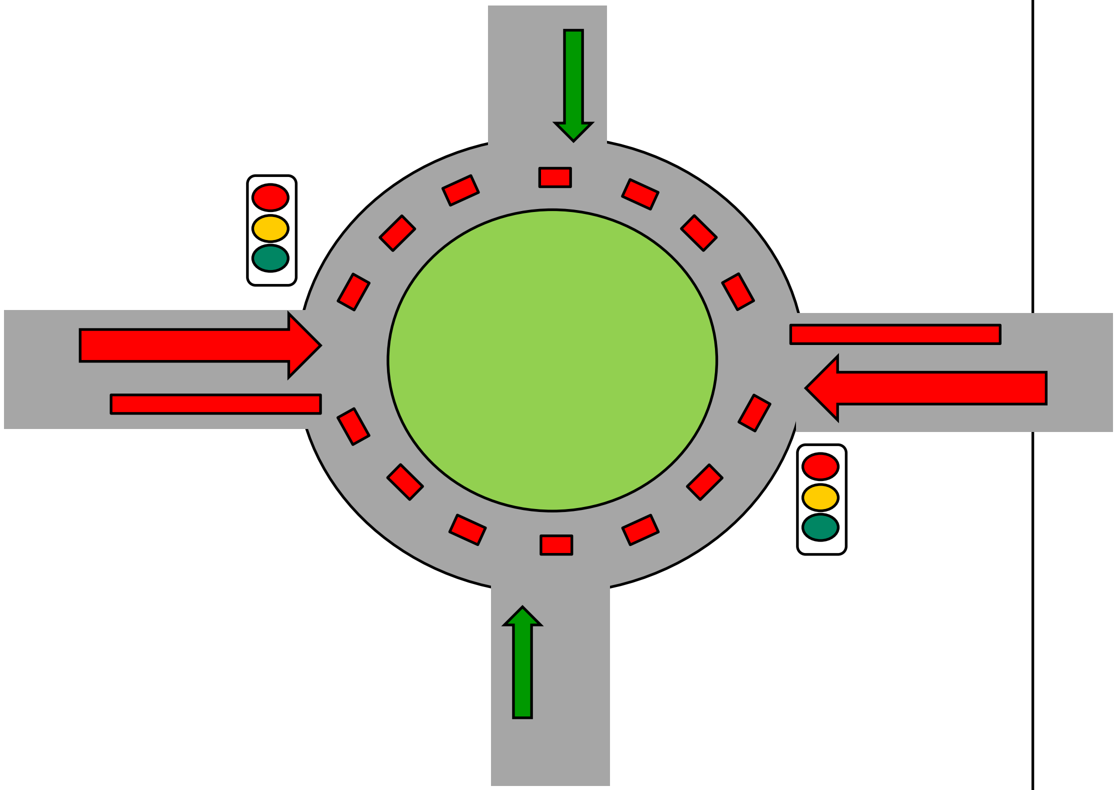
Store and forward packet switching:
Packet is read into memory. Once whole packet has arrived, transmission is started on the next hop. Needs memory – also write and read. Good bandwidth utilisation, but longer latency

Good for “bursty” traffic

cut through packet switching:
header is read to identify the destination, circuit for next hop is set up and the rest of the packet is passed straight through.

Congestion*

Incoming data exceeds data throughput.
Bottleneck.
Balanced traffic nothing to be done.
Unbalanced ….



Limit east-west and throughput increases

East-West route so congested it blocks round-about
Causes delays North-South traffic which is less than maximum

Faults

Fault free network not possible

Lossless networks . SANs

Fault tolerance

Hardware and software
Permanent and transient.
Faults will occur

Resilience is required

Transient: eg EM interference. Solved by retransmit

Permanent: aging. Component failure. Overheating.
Retransmit does not solve.
resource sparing
fault-tolerant routing
network reconfiguration

Tolerance

Fault free network not possible

resource sparing

faulty components switched off, spare ones bought into play.

fault-tolerant routing

multi-path system have built in resilience. Use alternative path(s). Possible deadlock problems.

network reconfiguration

non faulty network paths must be identified. Working paths identified and new routing tables distributed. Needs programmable switches/network interfaces.

Hot-swapping remove faulty & install working components while the network is working. *Dynamic-network reconfiguration.*

Buy good components and "maintain" them. Don't exceed environmental limits. Replace in a timely fashion.

networks

On chip

Multi-chip architecture is increasing relevance Sony/IBM/Toshiba Cell processor the "Cell Broadband Engine" has a proprietary Element Interconnect Bus (EIB). Four separate alternating uni-directional rings.
16-128 bytes. No packet headers – routes established before transmission. No error detection. Centralised arbitration. Best case sustainable rate is 204.g GB/sec.

Wastes bandwidth

Machine room

IBM Blue Gene/L – 32 x 32 x 62 3D torus for the 65,536 dual cores. Has a header and a 1 byte CRC to protect header information. Failure rate is reduced by limiting clock frequency to 700MHz. Speed is improved by using cut-through routing.

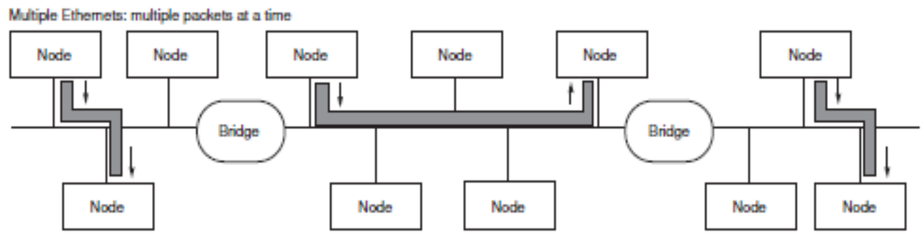
SAN

Infin-Band. Switch based interconnect technology. 2-120 Gbp/link. Complex packet structure and features to improve reliability and resilience

networks (i)

Ethernet

XeroxParc 3 Mbit/sec.
Now 10/100 Gbit/sec.
Improve performance with multiple network segments separated by bridges.



Wastes bandwidth

OSI/ISO 7 layer model.
Breaks communication into seven layers.

Internet

A protocol can be implemented at any level, as long as it has the correct interface to talk to the layers above and below.
Certain implementations are produce to provide all the layers *a stack.*

7 Layer

Ethernet

Layers increase flexibility, but increase CPU requirements at the ends and reduce bandwidth for useful data.

Probability of error v. cost of error.
Size of packet.
Traffic type. Continuous v. bursty
Lossy v lossless

No single optimisation

7	Application	Data
6	Presentation	Data
5	Session	Data
4	Transport	segments
3	Network	packet
2	Data Link	frame
1	Physical	bits

Routing algorithms

Deterministic: always chooses the same path for a communicating source-destination pair

Oblivious: chooses different paths, without considering network state

Adaptive: can choose different paths, adapting to the state of the network

Deterministic:
Simple
Deadlock Free
No use of other paths
Can get high contention

Oblivious:
Simple
Random – so mitigates contention from pairs of nodes which repeatedly clash in deterministic

Adaptive:
More complex
Avoids local congestion
Load balancing means good utilisation
Local decisions can lead to livelock (hop counter)