Introduction to High Performance Computing

PDC Summer School

Interconnection Networks

2011-08-24

Michael Schliephake

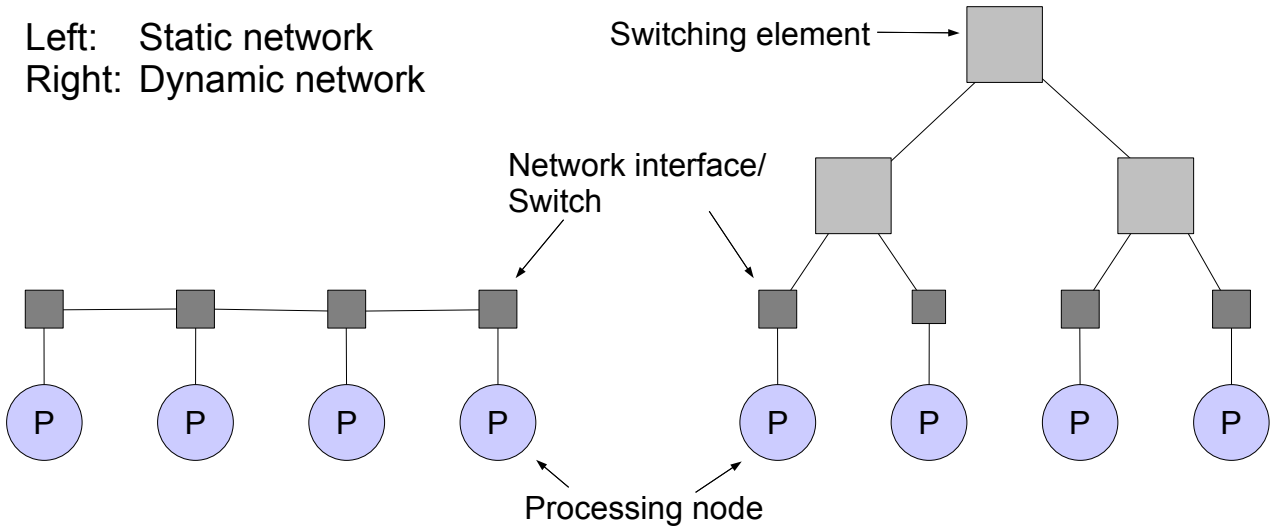
KTH PDC - michs@kth.se

Agenda

1. Introduction
2. Static Networks
3. Dynamic Networks
4. Routing and Switching
5. Communication Operations
6. Practical Aspects

1. Introduction
Interconnection Networks I

- Data transfer between
  - Processing nodes
  - Processors and memory
- Abstract view: n inputs, m outputs
- Typical components: links, switches, interfaces



Interconnection Networks II

- Topology – describes the geometric structure
  - Graph – switches, processors, memory as nodes; connection links as edges
  - Static networks = Direct or Point-to-point networks
  - Dynamic networks = Indirect networks
- Routing technology – defines how and along which path messages are transported
  - Routing = routing algorithm selects a path
  - Switching strategy = defines segmentation of messages, mapping to a path, handling by switches and processors

Topology and Routing technology determine decisive the performance of the communication

Criteria for Networks I

Diameter

Maximum distance between any two processing nodes

- Distance: Shortest Path (number of links) between two nodes
- Measure for the time to transfer messages between arbitrary nodes

Criteria for Networks II

Degree

Maximum degree of all processing nodes

- Degree of a node is the number of in- and outgoing links of a node
- Measure for the number of simultaneously active communication connections
- Measure for hardware efforts

Criteria for Networks III

Connectivity

Measure of the multiplicity of pathes between arbitrary processing nodes

- High connectivity lowers contention, increases reliability
- Arc (node) connectivity = number of links (nodes) to remove to separate the network in two disconnected networks

Criteria for Networks IV

Bisection Width, Bisection Bandwidth

Bisection width = minimum number of links to remove to get two equal halves,

Bisection bandwidth = minimum volume of communication between the halves

- Bisection bandwidth
  = bisection width x channel bandwidth
- Measure of loading capacity: simultaneously transmission of „bisection width + 1“ messages may saturate the network
- Multicore – Bisection bandwidth per core

Criteria for Networks V

Cost

- Many criteria possible
  - Number of communication links, number of wires
  - Bisection bandwidth
- Technology
- Additional equipment

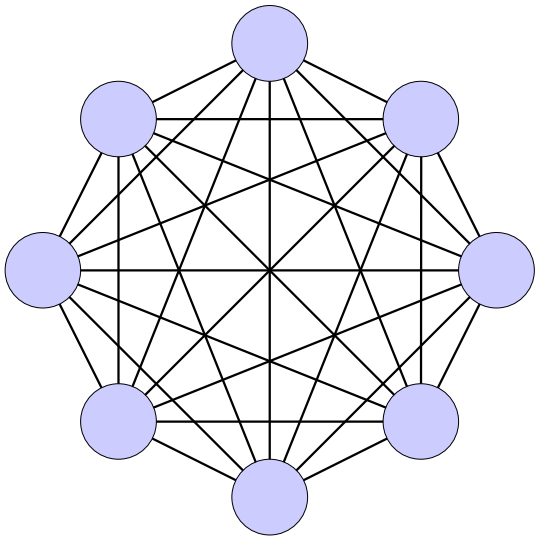
Criteria for Networks VI

General Requirements

- Small diameter
  short distances for transmissions
- Small node degree
  reduction of the hardware efforts
- High bisection bandwidth
  high throughput
- High connectivity
  high reliability
- Good extensibility

Topologies I

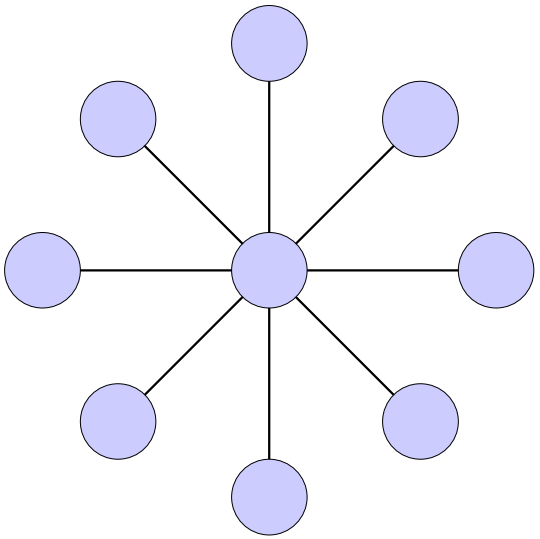
Completely connected



Diameter	1
Degree	$p-1$
Arc connectivity	$p-1$
Bisection bandwidth	$(\frac{p}{2})^2$
Cost (# links)	$\frac{p \cdot (p-1)}{2}$

Topologies II

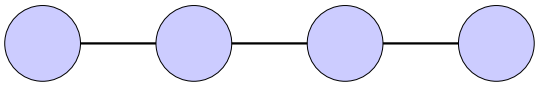
Star



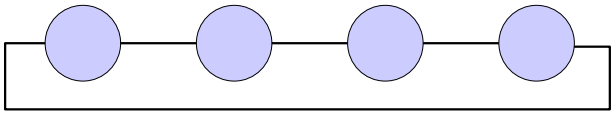
Diameter	2
Degree	$p-1$
Arc connectivity	1
Bisection bandwidth	1
Cost (# links)	$p-1$

Topologies III

Linear Array



Linear Array (no wraparound)



Ring

	Lin. Array	Ring
Diameter	$p-1$	$\llbracket \frac{p}{2} \rrbracket$
Degree	2	2
Arc connectivity	1	2
Bisection bandwidth	1	2
Cost (# links)	$p-1$	p

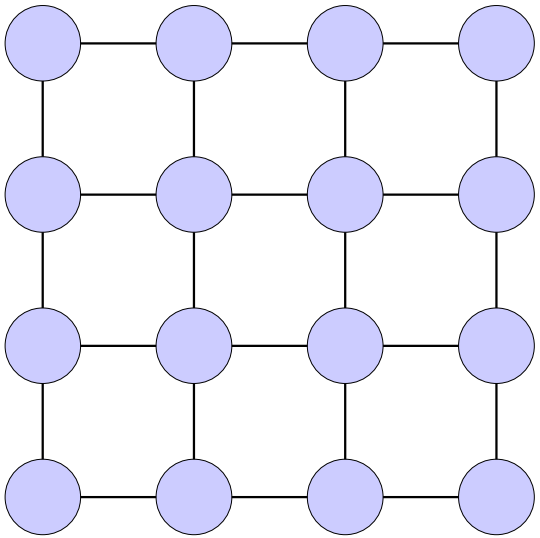
Topologies IV

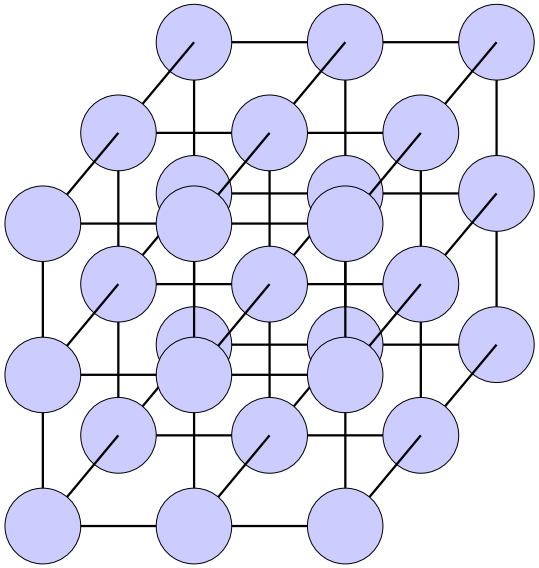
Mesh

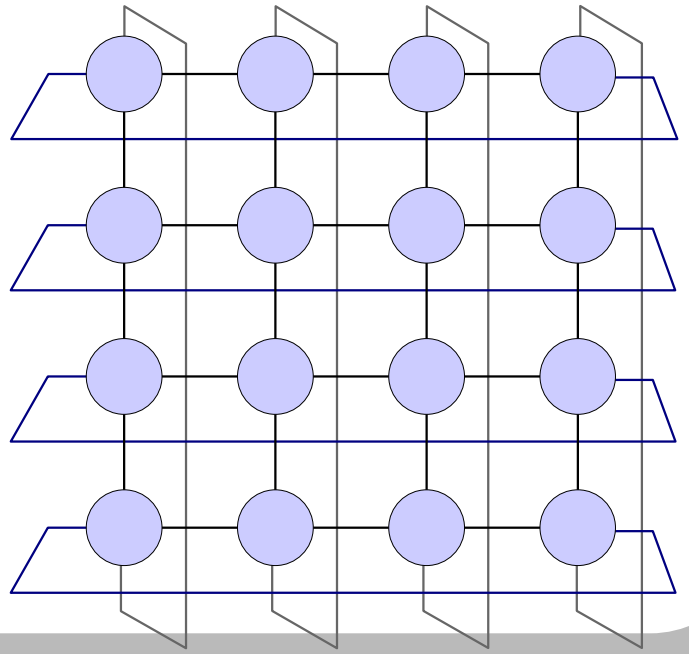
- Mesh (d dimensions)
- Torus (d dimensions)

$$p = r^d$$

	Mesh	Torus
Diameter	$d \cdot (\sqrt[d]{p} - ١)$	$d \cdot \lfloor \frac{\sqrt[d]{p}}{٢} \rfloor$
Degree	$٢ \cdot d$	$2 \cdot d$
Arc connectivity	d	$2 \cdot d$
Bisection bandwidth	$p^{\frac{d-1}{d}}$	$2 \cdot p^{\frac{d-١}{d}}$
Cost (# links)	$d \cdot (p - \sqrt[d]{p})$	$d \cdot p$

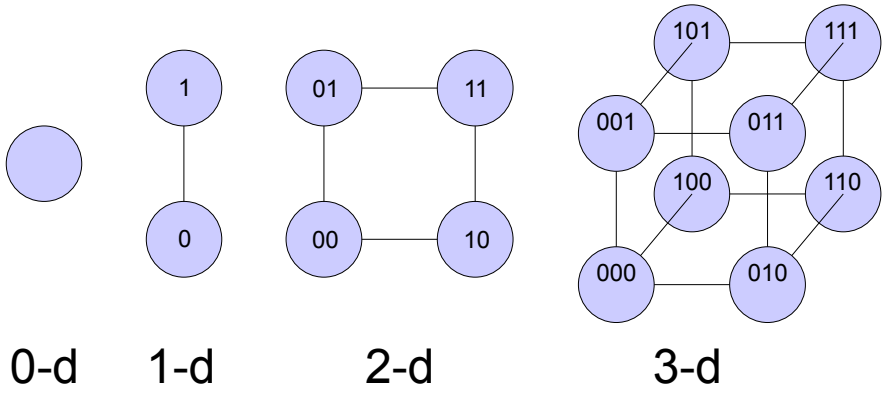


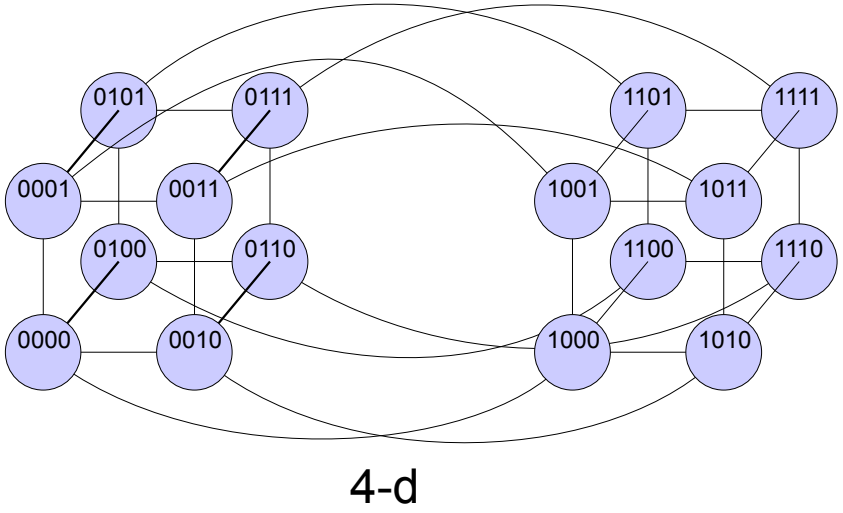




Topologies V

Hypercube





$$p = 2^d$$

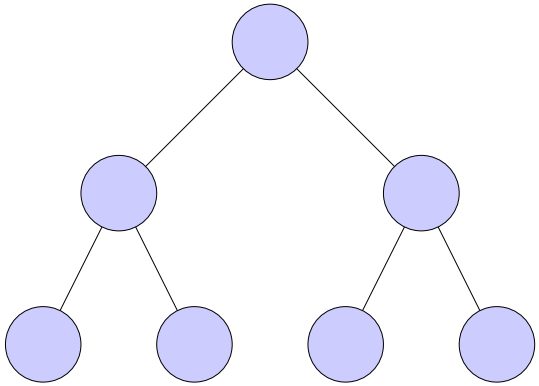
Diameter	$\log(p)$
Degree	$\log(p)$
Arc connectivity	$\log(p)$
Bisection bandwidth	$\frac{p}{2}$
Cost (# links)	$\frac{p \cdot \log(p)}{2}$

Topologies VI

(Static) Tree

$$p = 2^k - 1$$

Compl. binary tree



Complete binary tree

Diameter	$2 \cdot \log(\frac{n+1}{2})$
Degree	3
Arc connectivity	1
Bisection bandwidth	1
Cost (# links)	$p-1$

Topologies VII

Summary table

	Compl. Connect.	Star	Lin. Array	Ring	Mesh	Torus	Hyper-cube	Binary Tree
Diameter	1	2	$p-1$	$\lfloor\frac{p}{2}\rfloor$	$d\cdot(\sqrt[d]{p}-1)$	$d\cdot\lfloor\frac{\sqrt[d]{p}}{2}\rfloor$	$\log(p)$	$2\cdot\log(\frac{n+1}{2})$
Degree	$p-1$	$p-1$	2	2	$2\cdot d$	$2\cdot d$	$\log(p)$	3
Arc connectivity	$p-1$	1	1	2	d	$2\cdot d$	$\log(p)$	1
Bisection bandwidth	$(\frac{p}{2})^2$	1	1	2	$p^{\frac{d-1}{d}}$	$2\cdot p^{\frac{d-1}{d}}$	$\frac{p}{2}$	1
Cost (# links)	$\frac{p\cdot(p-1)}{2}$	$p-1$	$p-1$	p	$d\cdot(p-\sqrt[d]{p})$	$d\cdot p$	$\frac{p\cdot\log(p)}{2}$	$p-1$

4. Dynamic Networks

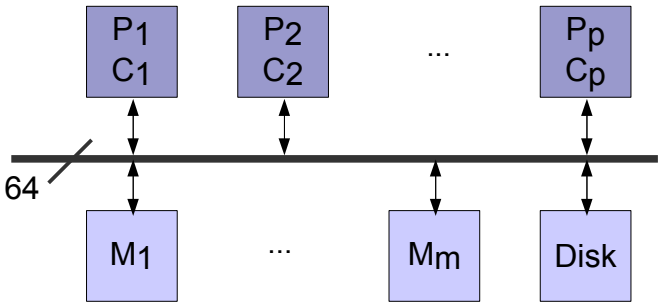
Criteria for Networks I

- Similar to static networks
  - Processing in switches costs time seen as nodes
- Diameter = maximum distance between any node (in practice processing nodes)
- Node and edge connectivity = number of nodes or edges to remove to get two networks
- Bisection bandwidth = any possible partitioning of processing nodes into two equal parts to consider induces partitioning of switching nodes with minimized number of crossed edges

Bus, Crossbar

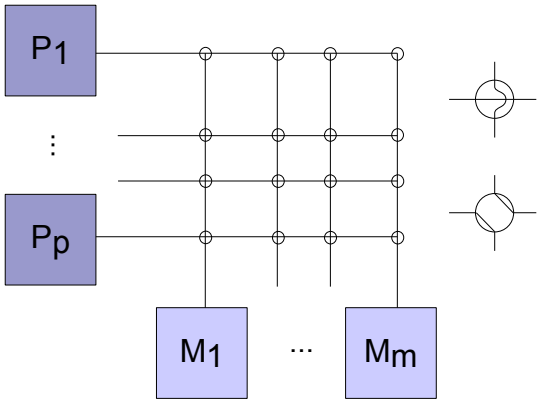
Bus

- Simple, cheap
- Constant distance
- Good for Broadcasts
- Scaling limited by performance



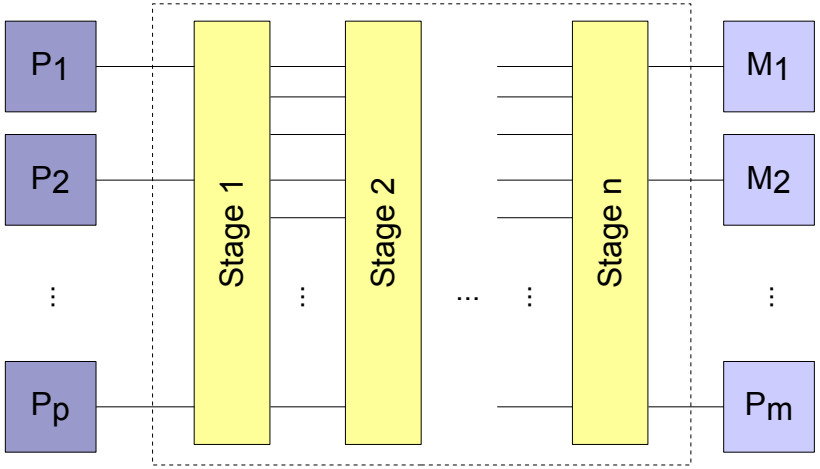
Crossbar

- Complex, expensive
- Non-Blocking
- Realization hard for large p and high speed
- Scaling limited by cost

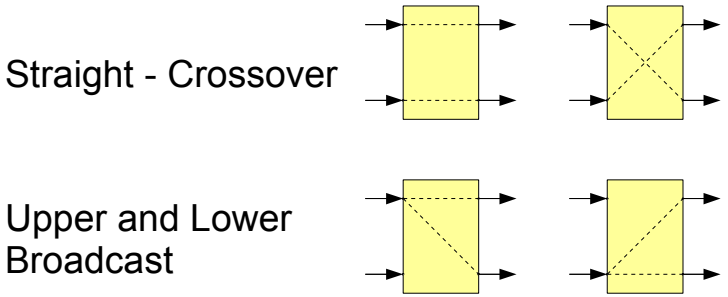


4. Dynamic Networks
Multistage Interconnection Networks I

- Intermediate features between bus and crossbar
- Characteristics
  - constuction rule
  - degree of switching nodes
- Examples
  - omega network
  - baseline network
  - butterfly network
  - benes network



Schematic view of a multistage interconnect network



Switch positions for a 2x2 switch

Multistage Interconnection Networks II

Omega network

- Example of a blocking network
- log(p) stages
- Construction rule:

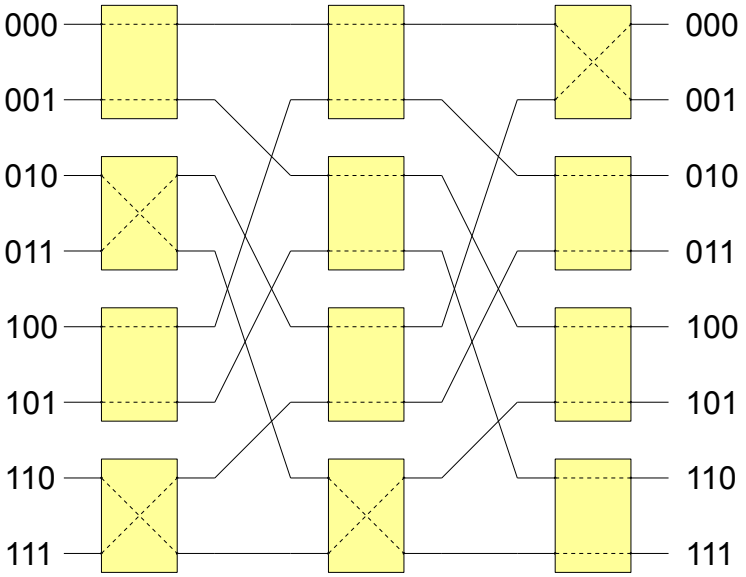
$$j = \begin{cases} ٢\mathrm{i}, & ٠ \le i \le \frac{p}{٢} \\ 2\mathrm{i}+1-p, & \frac{p}{2} \le i \le p-1 \end{cases}$$

(perfect shuffle)

- Number of switching nodes

$$\frac{p}{2} \cdot \log(p)$$

$$\pi^8 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 4 & 7 & 0 & 2 & 6 & 5 & 3 \end{pmatrix}$$

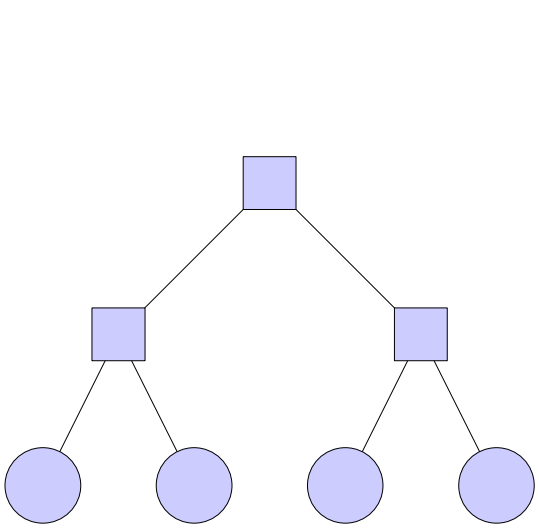


Realisation of an omega network with 2 x 2 switches

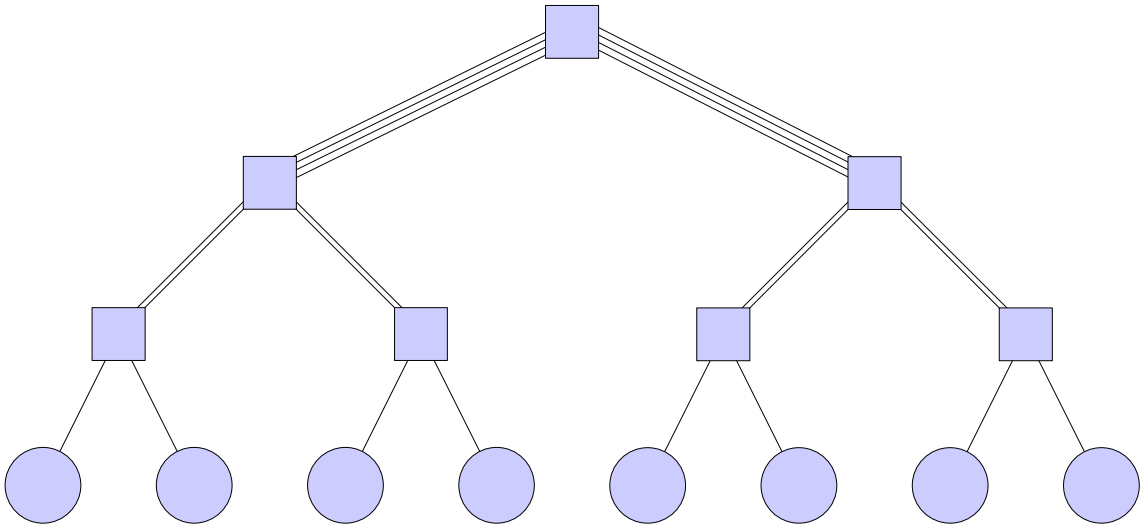
Tree-based networks

(Dynamic) Tree

- Nodes at intermediate levels are switches, processing nodes are leafs
- Communication bottleneck at higher levels



Tree with dynamic network



Fat tree

Properties

Summary table

	Crossbar	Omega Network	Dynamic Tree
Diameter	1	$\log(p)$	$2 \cdot \log(p)$
Arc connectivity	1	2	2
Bisection bandwidth	p	$\frac{p}{2}$	1
Cost (# links)	p^2	$\frac{p}{2}$	$p-1$

Routing

- Routing algorithm defines a path to send messages between nodes
- Requirements
  - Deadlock-free
  - Consideration of the topology
  - Avoid Contention
  - Avoid Congestion
- Types of algorithms
  - minimal, non-minimal
  - deterministic, adaptive

Switching

- Switching strategy defines how a message travels along a routing path
  - Segmentation
  - Allocation type of the path
  - How messages are processed in switching nodes

  Strong influence on the transfer time of messages between nodes

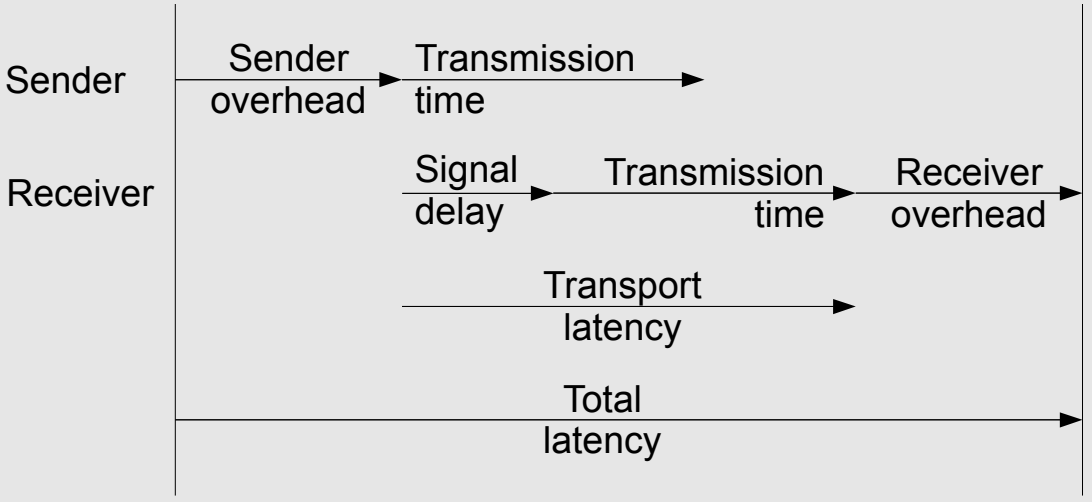
Message Passing Costs

- Message size $m[B]$
- Bandwidth $[B \cdot s^{-1}]$
- Byte transfer time $t_B = \frac{1}{bandwidth}$
- Transfer time $\frac{m}{bandwidth} = t_B m$
- Hop time t_h
- Signal Delay
- Transport Latency
- Sender overhead
- Receiver overhead

- Latency

Latency = Overhead + Transfer time

$$t(m) = t_s + t_B m$$



[after Hennessy, Patterson]

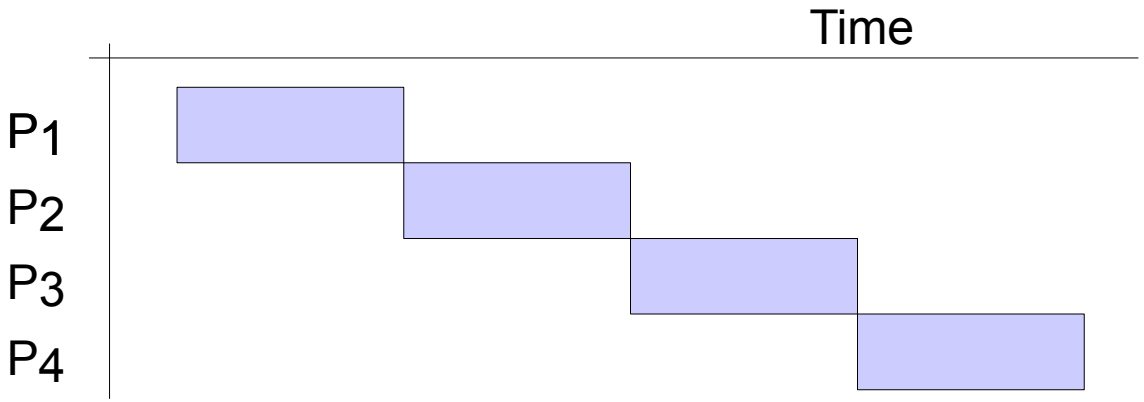
Store-and-Forward Routing

- Message is transferred completely between nodes on the path
- Communication time for path of length l:

$$t_{comm} = t_s + (m\,t_B + t_h)\,l$$

simplified for modern equipment:

$$t_{comm} = t_s + m\,l\,t_B$$



Store-and-Forward Routing with 4 nodes

Packet Switching

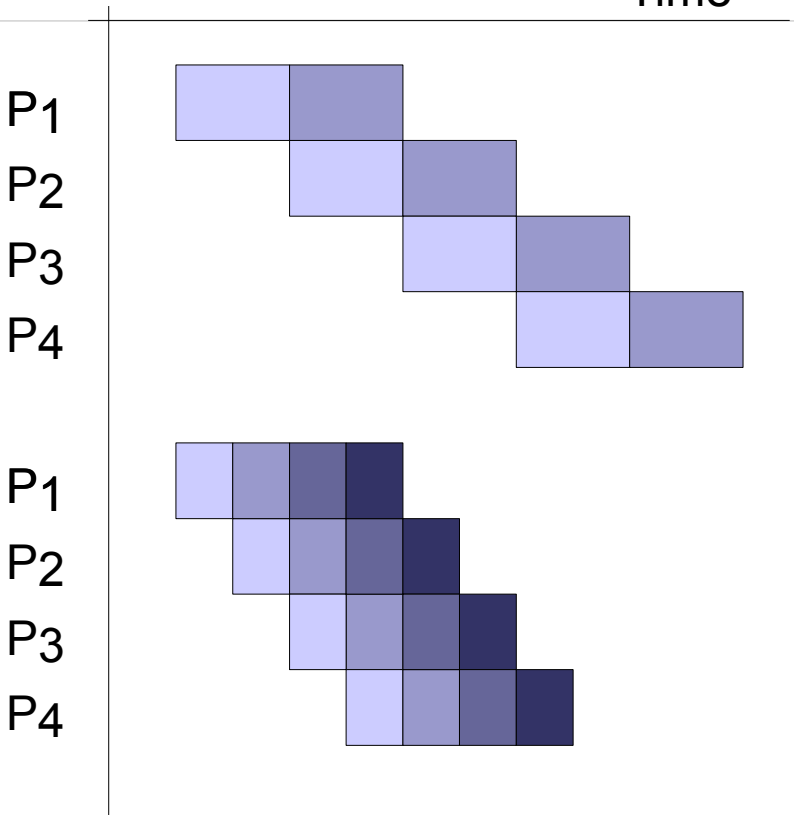
- Divide mesage in packages to reduces transfer time
- Other advantages
  - Packet losses cheaper
  - Packages can use different pathes
  - Better error correction possibilities
- Communication time (static route)

$$t_{comm} = t_s + t_h l + t_B' m$$

$$t_B' = t_p + t_B\left(1 + \frac{s}{r}\right)$$

$t_p \ldots$ effort to packetize message,

$r \ldots$ message length in packet, $s \ldots$ header size of packet



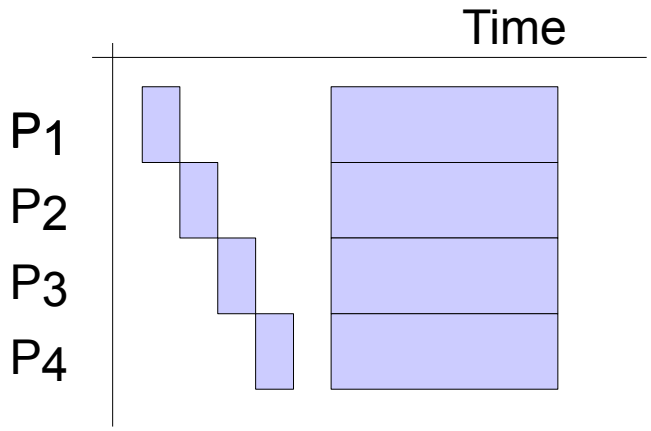
Packet Routing with 4 nodes
for a message divided in packages

Circuit Switching

- Path established through the sending of a control message, exists until the the communi-cation ends
- All nodes active at the same time to transfer the message
- Communication time for path of length l:

$$t_{comm} = t_s + t_B(m_c l + m)$$

$m_c \ldots$ length of control message



Circuit switching with 4 nodes

Cut-Through Routing

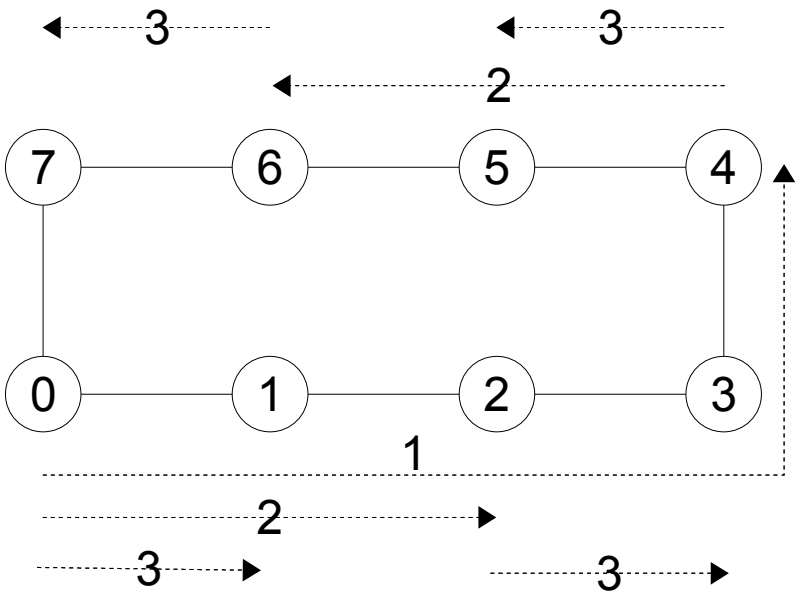
- Virtual Cut-Through Routing
  - Packages are subdivided and transported in a pipelined manner after evaluation of the header
    - Message is divided into „flow control units“ (flits) – smaller than packets
  - Collection at nodes where route is blocked
- Variant Wormhole Routing
  - Flits are blocked at their current position
- Advantages:
  - Safe of intermediate stores and sends
  - Reduced buffer size

Communication Time: $t_{comm} = t_s + l\,t_h + t_B\,m$

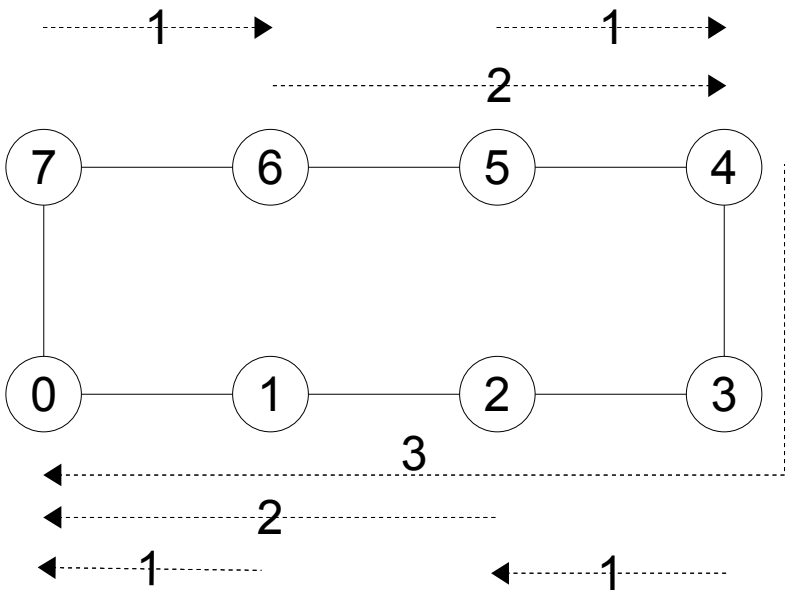
5. Communication Operations

- Communication influences the efficiency of a prallel program essentially
- Examples presented here should give an impression how important good implementations are
  (what will be done for most of us by the developers of libraries like MPI)
- Assumptions for the following
  - Cut-through routing
  - Bidirectional links
  - Single-port communication model

5. Communication Operations
Linear Array, Ring I



One-to-All Broadcast on a ring with eight nodes (MPI_Bcast)

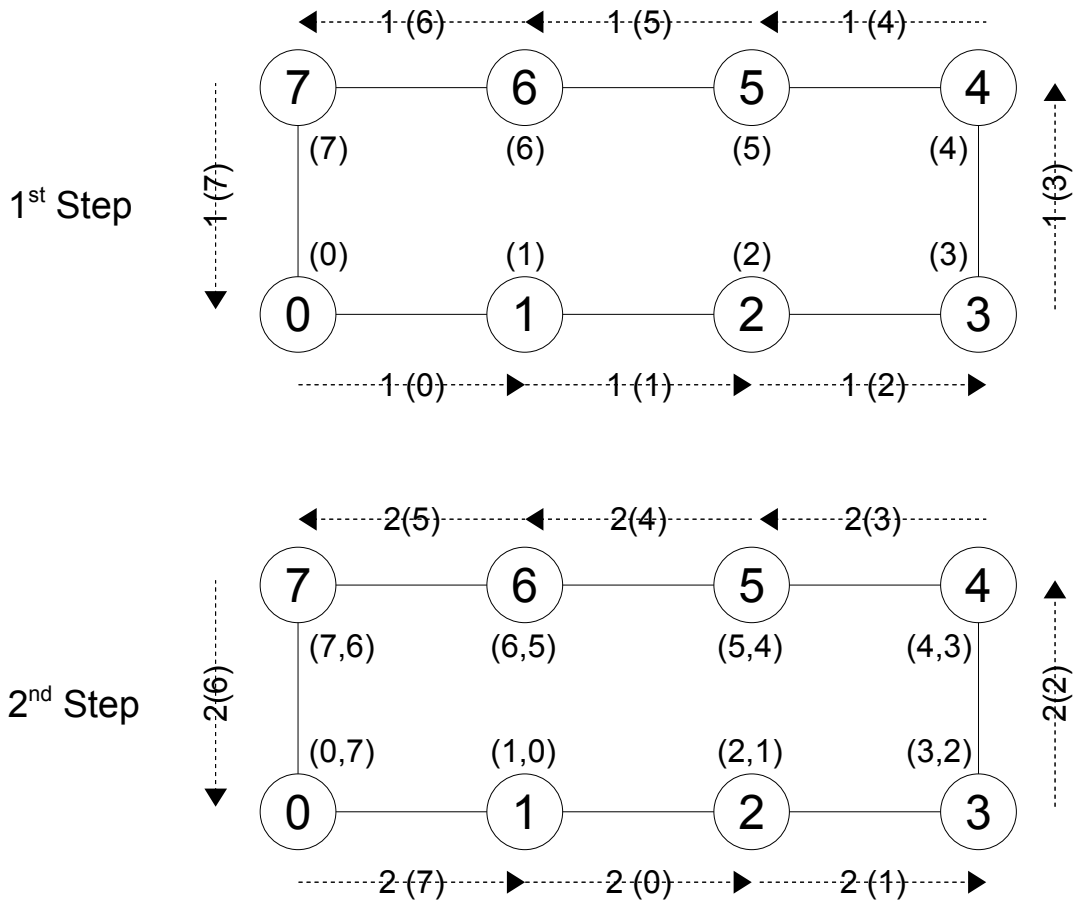


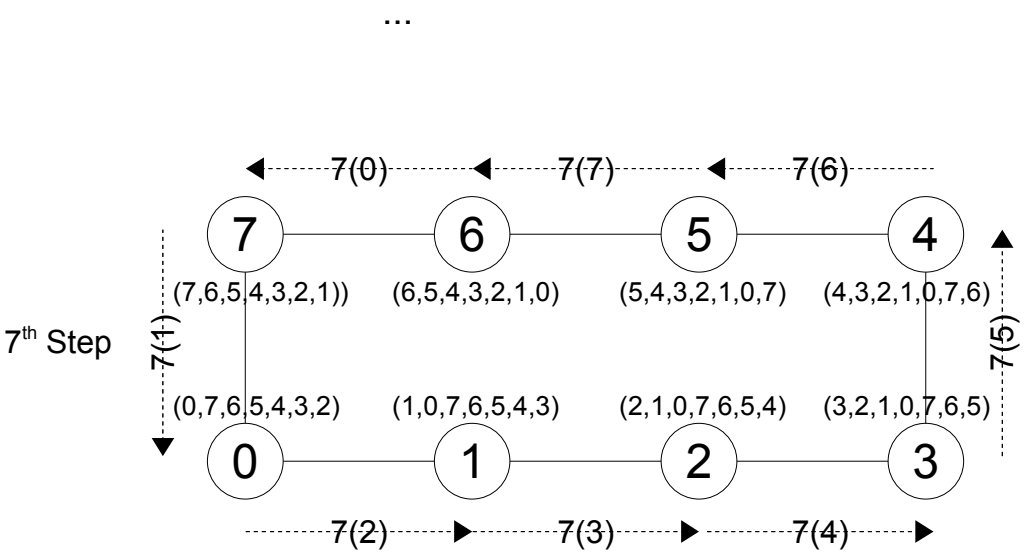
All-to-one Reduction on a ring with eight nodes (MPI_Reduce)

Dual operations

[after Grama et al.]

5. Communication Operations
Linear Array, Ring II

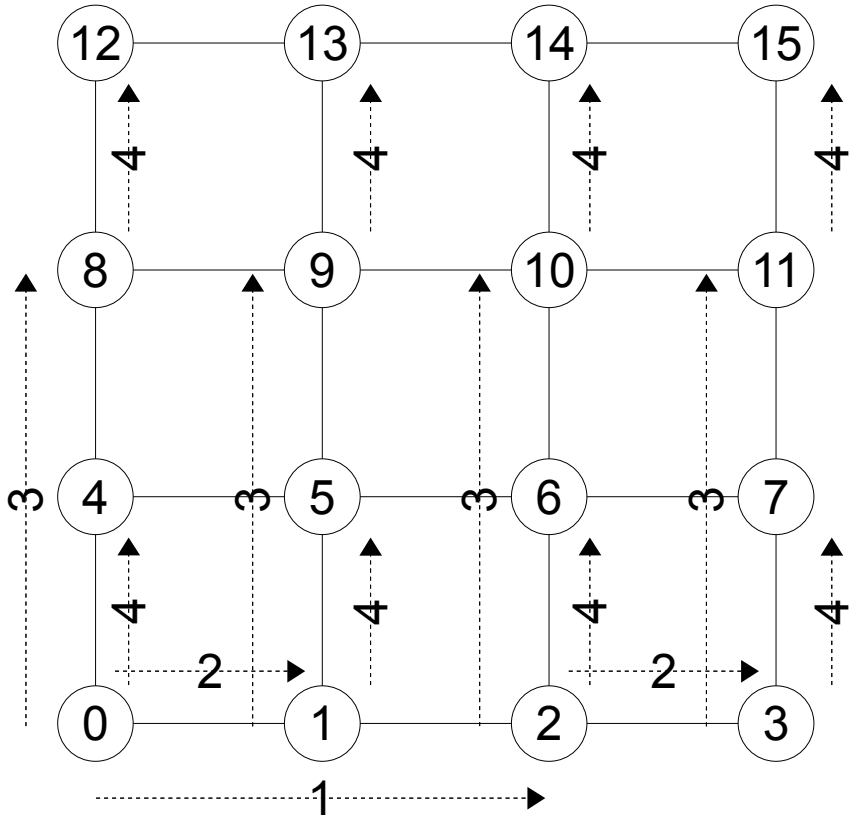




All-to-All Broadcast on a ring with eight nodes (MPI_Allgather)

[after Grama et al.]

5. Communication Operations
Mesh

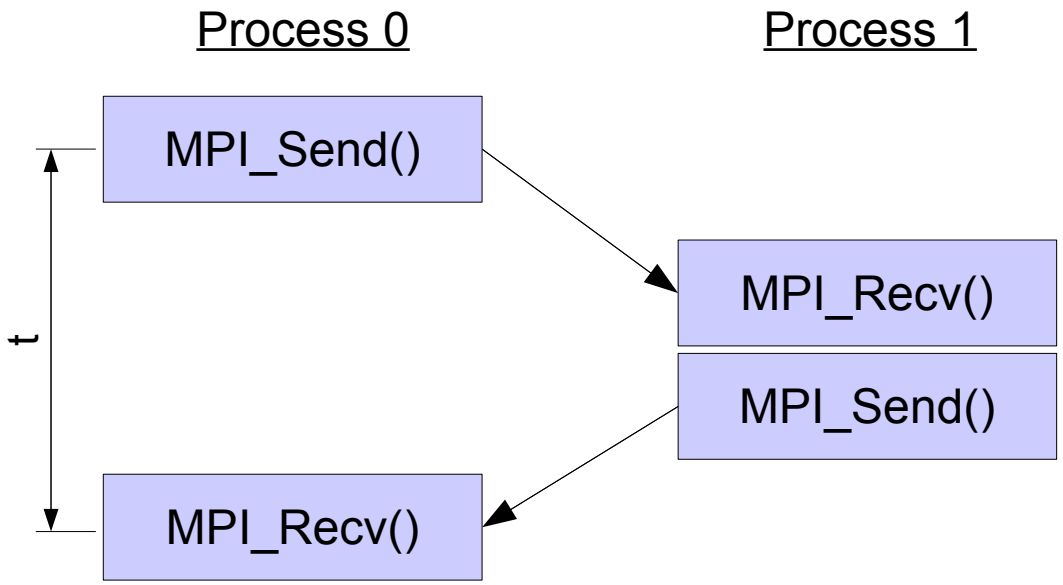


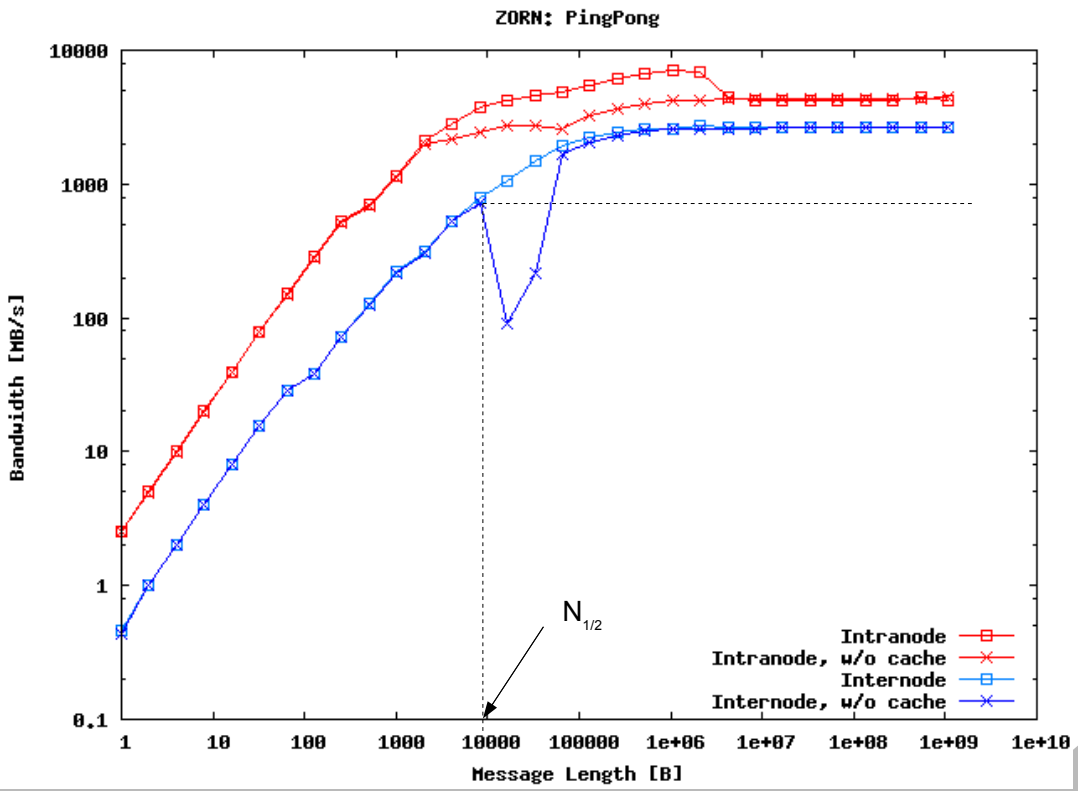
One-to-All Broadcast on a mesh (MPI_Bcast)

[after Grama et al.]

MPI Benchmarks

- Quick evaluation of a system
  - Self-written or some general available tests
  - IMB: http://software.intel.com/en-us/articles/intel-mpi-benchmarks/
- Example: Ping-Pong





Simplified Cost Model

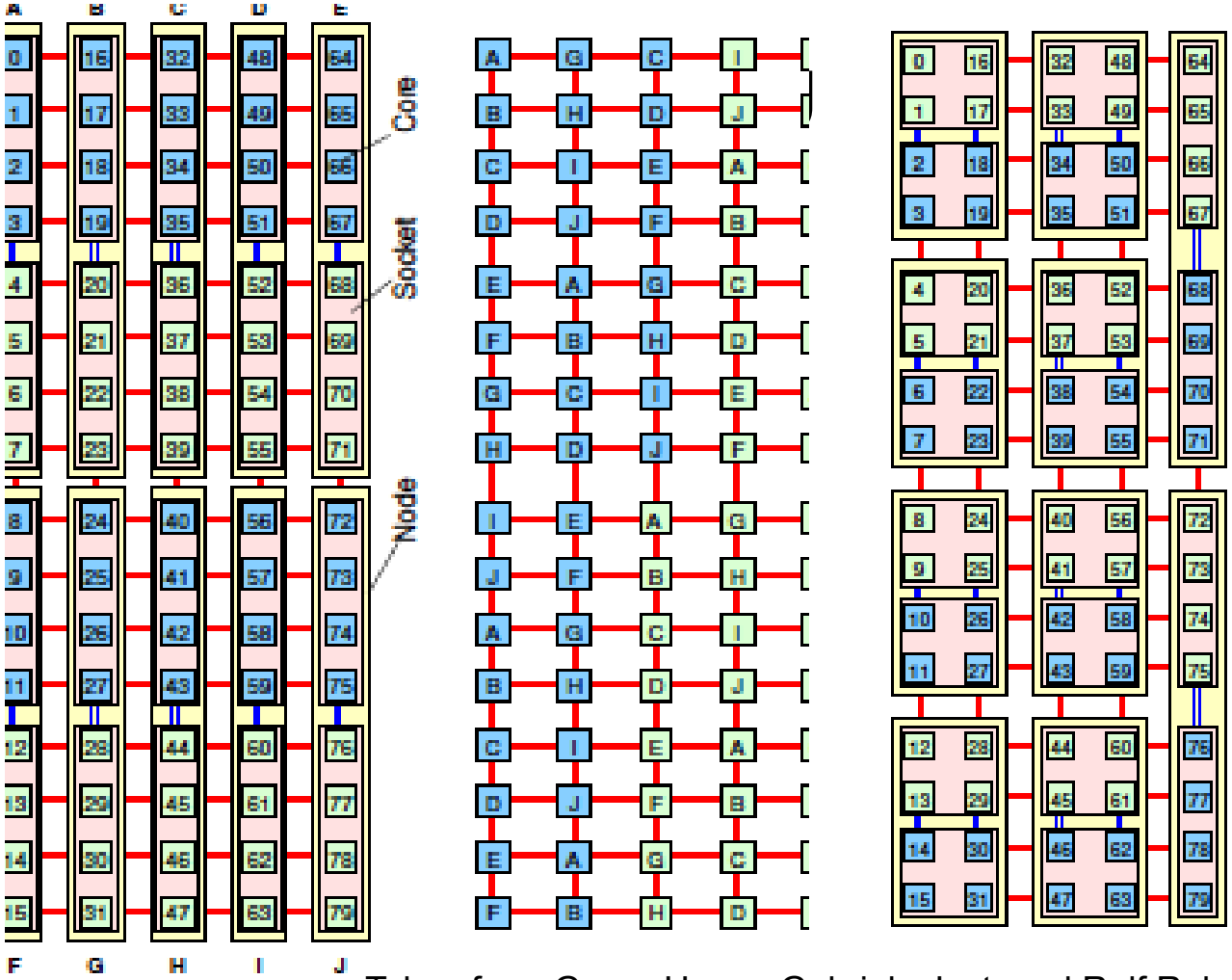
- Cut-Through-Routing $t_{comm} = t_s + l\,t_h + t_B\,m$
  - Prefer communication in bulk
  - Minimization of the transfer distance
  - Minimization of the data volume
- Reality allows simplification $t_{comm} = t_s + t_B\,m$
  - Limited influence on process mapping
  - Often randomized routing
  - Per-Hop time can be ignored often
- Consequences for programmer
  - Assumes the same time between arbitrary nodes (= assume completely connected network)
  - Accuracy loss: Only valid in networks without congestion
    - Topologies are sensible to congestion in different grade
    - Communication patterns produce different congestion

6. Practical Aspects

Hybrid parallelization

- Hybrid parallelization – f.ex. combined use of MPI and OpenMP or MPI and Pthreads or...
- Promising approach due to increasing number of cores sharing memory and decreasing available network bandwidth per core
- Potential advantages
  - Better scaling (avoiding data decomposition)
  - Reducing load imbalances
  - Reducing resource requirements (communication volume and bandwidth, memory)
- Problems
  - Increase of the complexity in the development
  - Success not guaranteed, time effort

6. Practical Aspects
Process mapping



Example for the Influence of the process mapping on the communication in an application with cartesian neighbourhood MPI communication.

Blue double lines - Inter-socket
Red single lines - Inter-node

Left: Ranks sequentially placed onto nodes

Middle: Ranks placed in round-robin-order onto nodes

Right: Two-level decomposition

Taken from Georg Hager, Gabriele Jost, and Rolf Rabenseifner: Communication Characteristics and Hybrid MPI/OpenMP Parallel Programming on Clusters of Multi-core SMP Nodes. Proceedings of the Cray Users Group Conference 2009 (CUG 2009), Atlanta, GA, USA, May 4-7, 2009.

6. Practical Aspects
Nonblocking and asynchronous communication

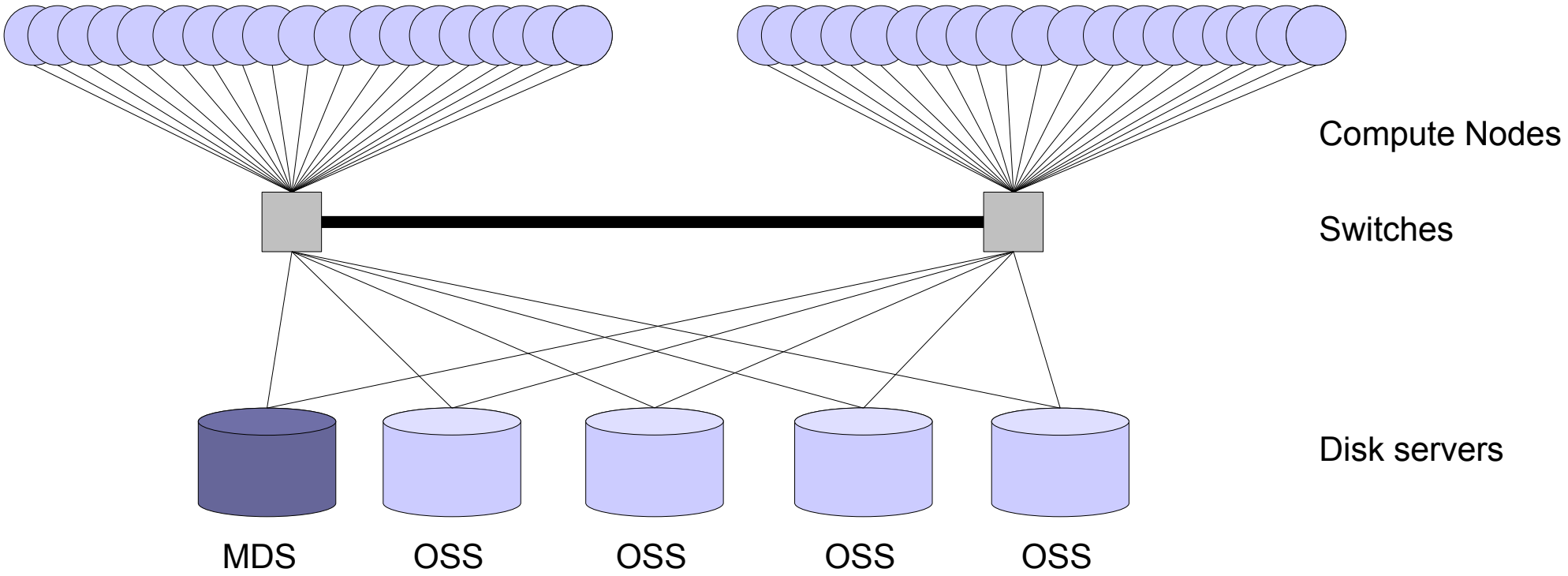
- Do not confuse nonblocking and asynchronous communication
- Further advantages of nonblocking communication
  - Avoid deadlocks
  - Reduced effort for process synchronisation (f.ex. eager protocol)
  - Handling of multiple requests (f.ex. duplex communication)

```
1 double precision :: delay
2 integer :: count, req
3 count = 80000000
4 delay = 0.d0
5
6 do
7   call MPI_Barrier(MPI_COMM_WORLD, ierr)
8   if(rank.eq.0) then
9     t = MPI_Wtime()
10    call MPI_Irecv(buf, count, MPI_BYTE, 1, 0, &
11           MPI_COMM_WORLD, req, ierr)
12    call do_work(delay)
13    call MPI_Wait(req, status, ierr)
14    t = MPI_Wtime() - t
15  else
16    call MPI_Send(buf, count, MPI_BYTE, 0, 0, &
17    MPI_COMM_WORLD, ierr)
18  endif
19  write(*,*) 'Overall: ',t,' Delay: ',delay
20  delay = delay + 1.d-2
21  if(delay.ge.2.d0) exit
22 enddo
```

Simple Benchmark to test asynchronous MPI communication (taken from [4] Hager, Wellein: Introduction to...)

6. Practical Aspects
Optimization of I/O

- Example of a cluster file system (Lustre)



6. Practical Aspects

Optimization of I/O (contd.)

- Program initialization, results, checkpoints
- Every user's activities influence all other users!
- How to do inefficient I/O?
  - Read all data on one processor and distribute it with broadcast, vice versa collect all results on one node and save it there to a file. (produces a lot of idle time)
  - Distribute all data in a huge number of small files and let every processor read one file (makes MDS the bottleneck and slows down all users)
- How to improve the I/O?
  - Define an I/O processor for a group of processors
    - Reduces load in the filesystem as well as the size of communication collectives
  - Example: 20.000 processes where each of 200 I/O processes distribute the data to 99 other processes

Literature I

1) Introduction to parallel computing / Ananth Grama ... [et al.].
Harlow, England ; New York : Addison-Wesley, 2003.
ISBN: 0201648652

2) Parallel programming : for multicore and cluster Systems / Thomas Rauber and Gudula Rünger
Berlin, Heidelberg: Springer, 2010.
ISBN: 978-3-642-04817-3

3) (Predecessor of 2) in German language and used for the lecture)
Parallele Programmierung / Thomas Rauber ; Gudula Rünger
Berlin ; Heidelberg ; New York : Springer, 2007.
ISBN: 978-3-540-46549-2

Literature II

4) Introduction to High Performance Computing for Scientists and Engineers / Georg Hager; Gerhard Wellein
Boca Baton, London, New York : CRC Press, 2011.
ISBN: 978-1-4398-1192-4

5) High Performance Computing : Programming and Applications / John Levesque; Gene Wagenbreth
Boca Baton, London, New York : CRC Press, 2011.
ISBN: 978-1-4200-7705-6