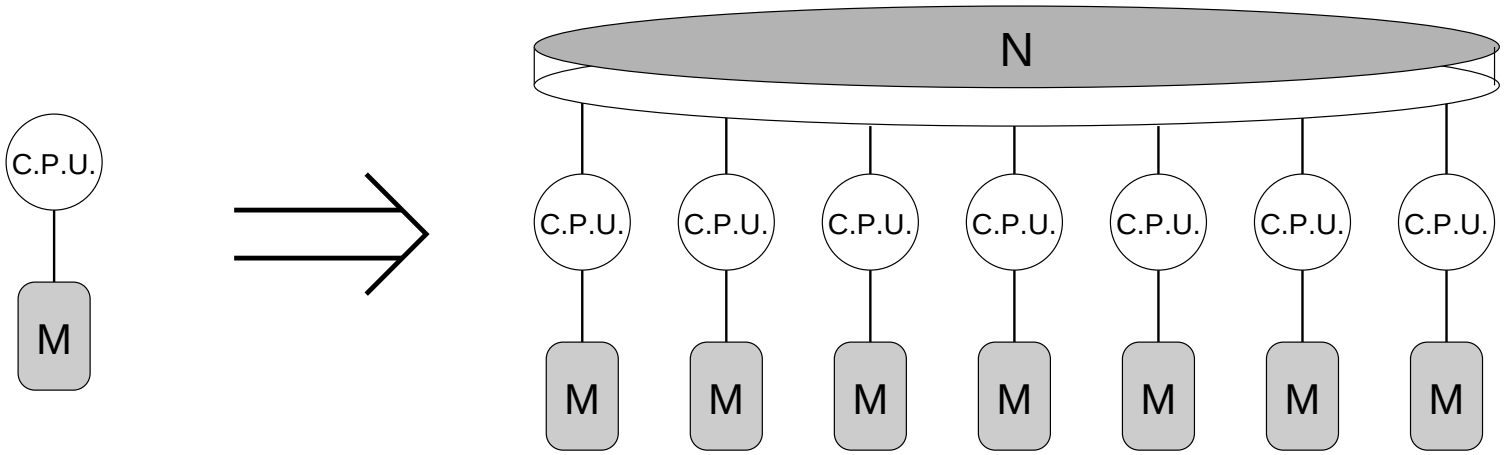
MIMD Distributed Memory Multiprocessor Systems



- Replicated Processing Elements
- Each PE is a full CPU with its own memory
- Communications Network - Processor to Processor

MIMD Distributed Memory Multiprocessor Systems

We have seen two models for parallel processing:

- Data Parallel Model
  - This model assumes that we have a *single process* which performs *calculations over whole data structures in parallel*.
  - Can use existing code following manual or automatic parallelization. Sometimes better to write new code.

- Shared Memory Model
  - Assumes that we have *multiple processes* all of which *share the same memory map*.
  - Can use existing code developed for multi-tasking on single processors.

Although it is possible to simulate each of these models on a distributed memory machine, we need a new model in order to exploit such a machine more efficiently.

MIMD Distributed Memory Multiprocessor Systems

- Communicating Processes Model for Distributed Memory Machines

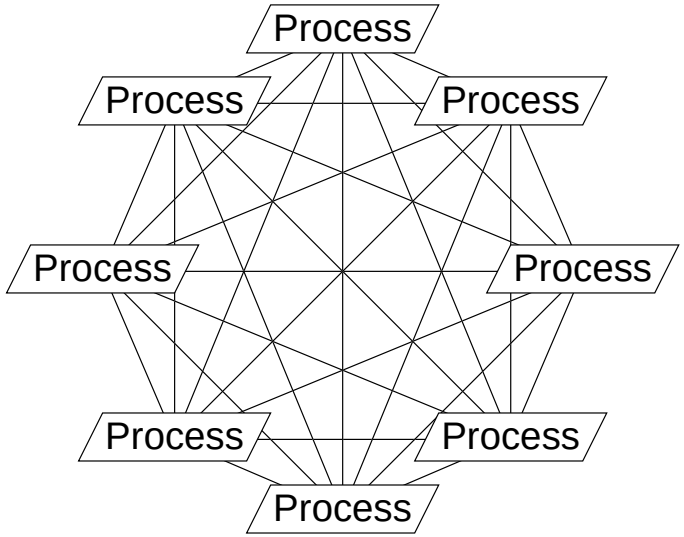
  - Assumes that we have *multiple independent processes* which *communicate and co-ordinate via the sending of messages*.

  - Each process can communicate with any other process, but can only access its own local memory.

  - In general we must write new code for this model.

MIMD Distributed Memory Multiprocessor Systems

Communications Network



We have suggested that our new model supports communication from any process to any other process.
It appears to follow that we will connect all processors to all other processors.

Connectivity - Some Options

- Full connection

  Each of *P* processors has *P-1* bi-directional communication links. One for each of the other processors.

- Bus Connection

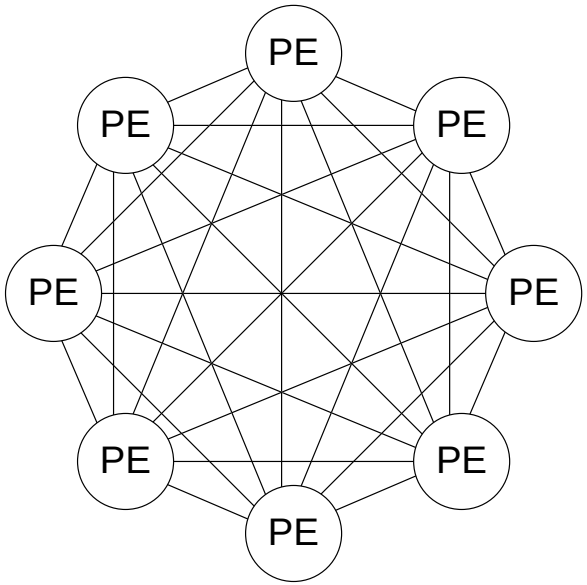
  All Processors share a single bus and compete for its use.

- Partial Connection

  Each of *P* processors has a number of connections to a number of different processors supporting a fraction of the required interconnection.

  The remainder of the connectivity is supported via message forwarding. Messages to distant nodes are forwarded from one node to the next until they reach their destination.

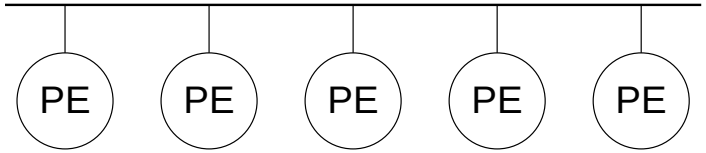
Fully Connected Network



- Expensive on link hardware ($(P-1)$ bi-directional links per PE).
- Complex wiring ($(P(P-1)/2)$ bi-directional links must be routed).
- Difficult to expand (each PE must be changed in order to add just one extra PE).

MIMD Distributed Memory Multiprocessor Systems

Bus Connection[1]

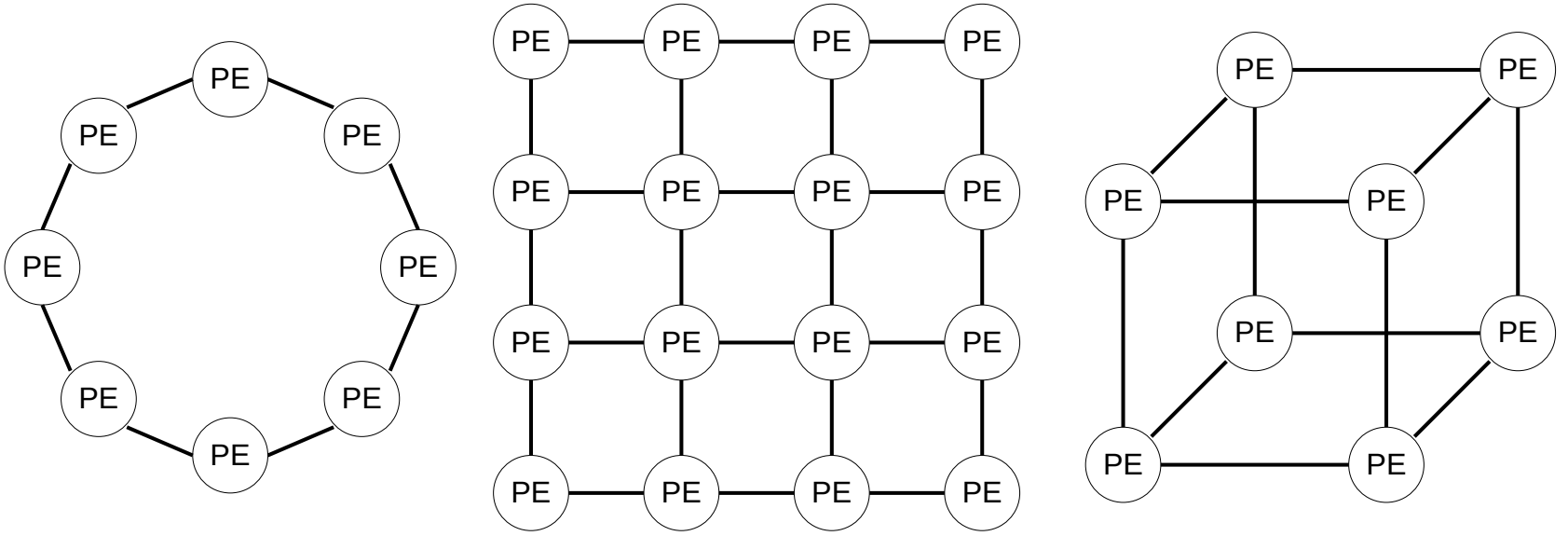


- Low Bandwidth.
- Bandwidth doesn't increase as more processors are added.
- Arbitration Problems.

[1] e.g. Ethernet

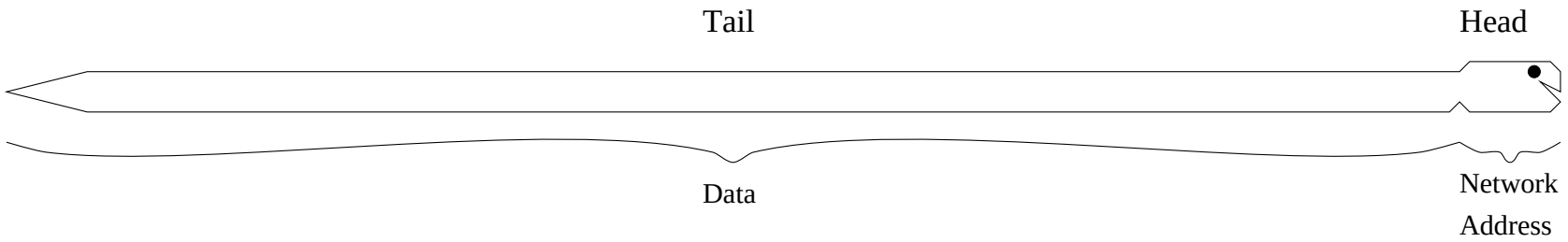
Partial Connection

Direct Networks:



Software Message Routing

Messages



- The message consists of a header containing a unique network address and a tail consisting of the message data.
- With software packet routing we run a routing process on each PE.
- The routing process is timeslice multi-tasked with the other processes running on the PE.
- Messages arriving at a node, carrying the local node address are consumed.
- All other messages are forwarded to other nodes in accordance with the prevailing routing strategy.

Network Performance

Network performance is gauged in terms of *Message Latency* and *Network Bandwidth*.

- Message Latency

  transit time for a single message

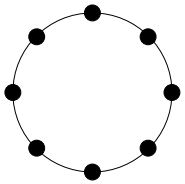
  depends on:

  - single hop latency
  - number of inter-node hops

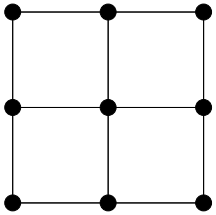
    the average distance will be affected by the *network connectivity*

    worst case is the *network diameter*

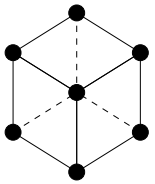
Network Diameter & Connectivity



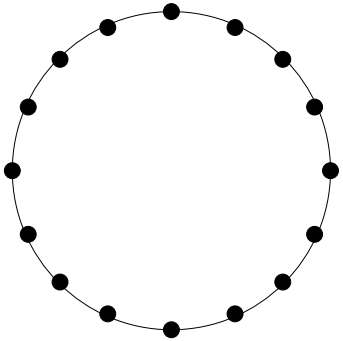
P = 8
d = 4
v = 2



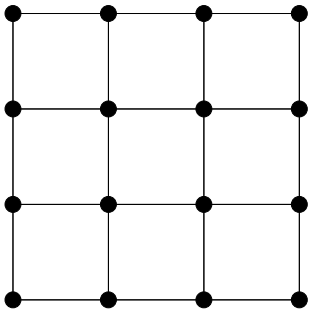
P = 9
d = 4
v = 4



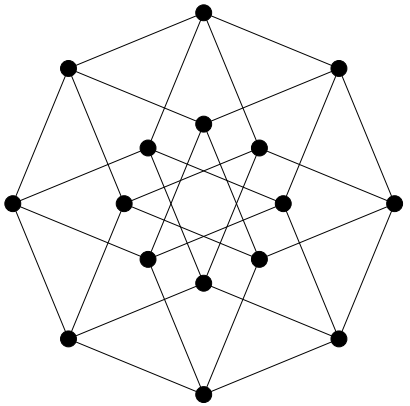
P = 8
d = 3
v = 3



P =16
d = 8
v = 2



P =16
d = 6
v = 4



P =16
d = 4
v = 4

Network Diameter & Node Valency

Topology	Diameter (d)	Valency (v)	$d \times v$
Ring ($\rightleftharpoons$)	$P/2$	2	P
2D Grid	$2(\sqrt{P}-1)$	4	$8(\sqrt{P}-1)$
ND Binary Hypercube	$log_2 P \ \{= N\}$	$log_2 P \ \{= N\}$	$(log_2 P)^2 \ \{= N^2\}$
Fully Connected Network	1	$P-1$	$P-1$

Since the *valency* (number of links per node) gives a measure of node cost, $d \times v$ gives a measure of *value for money*.

- The ND Hypercube is a popular topology for large networks.
  It offers excellent value for money with a low $d \times v$.
- The 2D Grid is a popular fixed valency topology.
  The same nodes may be used in large and small networks.
  Wiring is considerably simpler than the hypercube.
  Many finite element problems only require 2D nearest neighbour comms.

Message Latency

- The time taken for the delivery of a message is calculated as follows;

$$T = n * \left(\frac{l + h}{B} + o \right)$$

where

  - n is the number of hops.
  - l is the number of data bits.
  - h is the number of header bits.
  - B is the link bandwidth in bits per second.
  - o is the overhead for software routing.

it is assumed that the message is not delayed due to interaction with other messages.

Network Performance

- Network Bandwidth

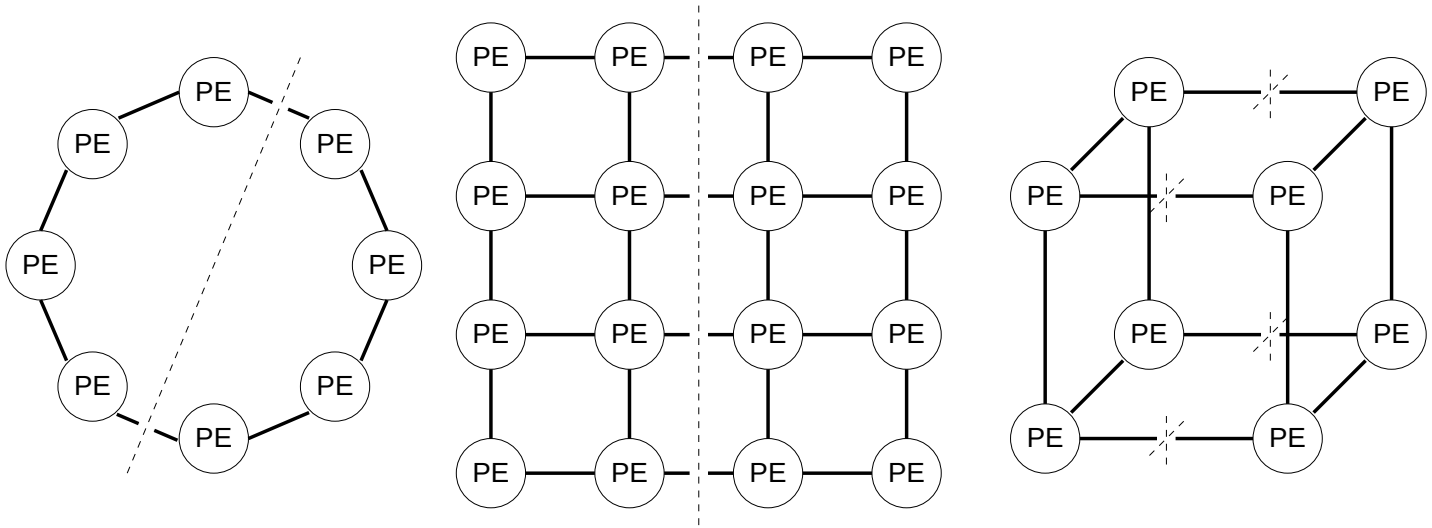
  how much message traffic the network can handle

  depends on:
  - single link bandwidth
  - number and arrangement of links

  The *total bandwidth* measure is a best case ignoring the arrangement of links.

  The *bisection bandwidth* is rather closer to the worst case.

Network Bisection Bandwidth



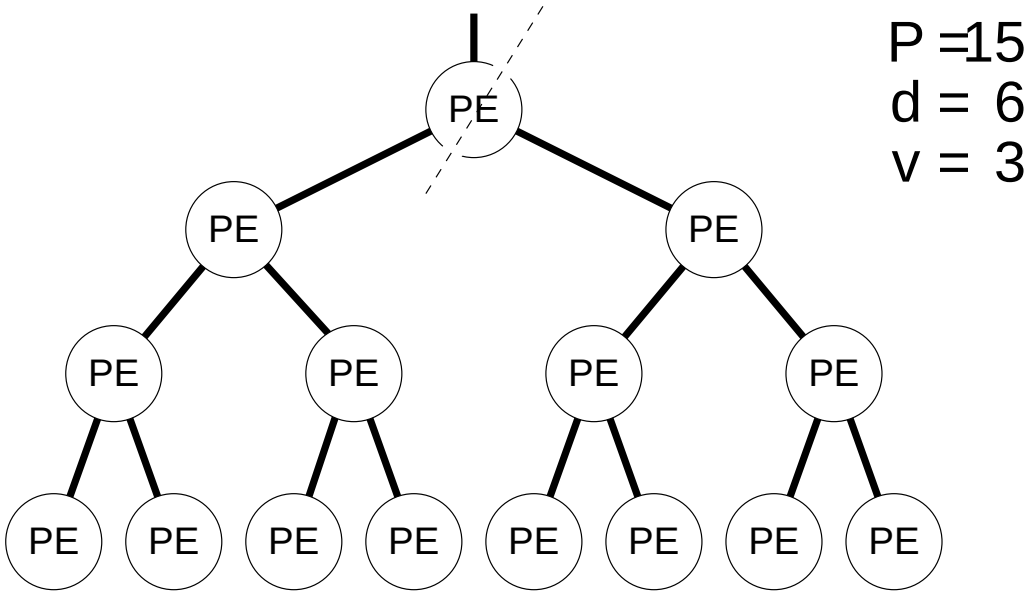
The network is divided in two in such a way as to give the worst possible bandwidth.

Scalability

Ideally this bisection bandwidth should increase in proportion to the number of processors as it does with the hypercube.

$$\textit{Bisection Bandwidth}_{Hypercube} = B \times P/2$$

Network Bisection Bandwidth



The bisection bandwidth highlights the *root hot-spot* of a binary tree.