SWITCHING TECHNIQUES

- **A generic router model**
- **Three layers in an interconnection network**
  - **Routing layer:**
    make routing decision at intermediate router and establish the path through the network.
  - **Switching layer:**
    use physical layer protocols to implement mechanisms for forwarding messages through the network.
  - **Physical layer:**
    transfer messages and manage the physical channels between adjacent routers.

- **Switching techniques determine**
  1. **when and how internal switches are set to connect router inputs to outputs;**
  2. **the time at which messages may be transfered along these paths.**
- **Assumptions:**
  - **Consider L-bit message in the absence of any traffic**
  - **Channel width: W bits**
  - **Message size: $L+W$ bits (message+ header)**
  - **Routing decision time: t_r sec.**
  - **Physical channel bandwidth: BW bits/sec.**
  - **Propagation delay of one channel: $t_w = \frac{1}{B}$**
  - **Switching delay (the delay inside the router): t_s**
  - **Source and destination are D links apart**

- **Basic switching techniques**
  - **Circuit switching:**

    A physical path from the source to the destination is established and the switches on the path remain in their specified states until the path is released.

    How it works:

    * **Establish the path by a routing probe**
    * **Destination sends an acknowledgement**
    * **Transmit data**
    * **Release the path by destination or last few bits of the message**

    Latency:

$$t_{circuit} = t_{setup} + t_{data}$$

$$t_{setup} = D[t_r + 2(t_s + t_w)]$$

$$t_{data} = \frac{1}{B}\left\lceil\frac{L}{W}\right\rceil$$

    Suitable for infrequent, long messages.

– **Packet switching:**

A packet (a group of bits of fixed length) moves from node to node, releasing links and switches immediately after using them. Also called store and forward switching.

How it works:

* **Message is divided into fixed-length packets**
* **Each packet contains routing information (in its header) and is routed individually.**
* **A packet is completely buffered at each intermediate node.**
* **Latency is proportional to the distance between source and destination.**

Latency:

$$t_{packet} = D\left[t_r + (t_s + t_w)\left\lceil\frac{L+W}{W}\right\rceil\right]$$

Suitable for frequent, short messages.

– **Worm-hole switching:**

Pipelined (hardware) packet switching. A compromise between packet switching and circuit switching.

How it works:

* **Divide a packet into flits.**
* **Only header flit contains the routing information and all flits in a packet follows the same path.**
* **Only buffer a few flits at each router (not the entire packet).**
* **In the case of blocking, message blocked in place.**

Latency:

$$t_{wormhole} = D(t_r + t_s + t_w) + \max(t_s, t_w) \left\lceil \frac{L}{W} \right\rceil$$

– **Virtual cut-through:**

Similar to worm-hole switching, but if the channel is blocked, the complete message is buffered at the node. At high network load, it behaves like packet switching.

Latency:

$$t_{vct} = D(t_r + t_s + t_w) + \max(t_s, t_w) \left\lceil \frac{L}{W} \right\rceil$$

INTERCONNECTION NETWORKS

- **A major component of a parallel computer, providing connections among processors and/or memory modules.**
- **Static networks (or direct networks): dedicated links between nodes (point-to-point connections).**
- **Dynamic networks (or indirect networks): network links can form different physical paths from sources to destinations (end-to-end connections).**

- **Network control:**

  Generate the necessary control setting on the switches to ensure reliable data routing from source to destination.

- **Control strategies:**

  - **Centralized control:**

    A single network controller takes requests from each input (source) and establishes paths. Easy to use global information to obtain optimal path settings.

  - **Distributed control:**

    Control circuit is associated to each switch/node. Each switch/node uses local information and a routing tag stored in packets.

- **Network design factors:**
  - **Network size:**
    The number of nodes in the network.
  - **Message latency (or network latency):**
    The time elapsed between the time a message is generated at its source node and the time is delivered at its destination node.
  - **Network throughput:**
    The maximum amount of information delivered by the network per time unit.
  - **Scalability:**
    As the network size increases, the network bandwidth should increase proportionally.
  - **Node degree:**
    The number of links incident on a node, denoted as d.

* **Network diameter:**

  The maximum of the shortest path between any two nodes, proportional to network latency.

* **Expandability:**

  The ability to add a node, depending on the number of components and connections required for adding a node.

* **Redundancy (Reliability):**

  The number of different paths between a source and a destination.

* **Bisection width:**

  Cut the network into two halves. The minimum number of links along the cut, denoted as b. It indicates the maximum communication bandwidth.

* **Routing algorithm complexity:**

  Fast or slow. Affects network latency.

- **Routing functions (or interconnection functions)**
  - **Rotation:** $+1$ **mod** N
  - **Shifting:** $+i$ **mod** N
  - **Mesh function (for an** $n \times n$ **mesh)**

$$M_{+1}(x) = (x+1) \textbf{ mod } N$$
$$M_{-1}(x) = (x-1) \textbf{ mod } N$$
$$M_{+n}(x) = (x+n) \textbf{ mod } N$$
$$M_{-n}(x) = (x-n) \textbf{ mod } N$$

  - **Shuffle-exchange:**

    Let $m = \log N$ **and represent a node in binary** $b_{m-1}b_{m-2}\ldots b_1b_0$**.**

    Shuffle function

$$S(b_{m-1}b_{m-2}\ldots b_1b_0) = b_{m-2}b_{m-3}\ldots b_0b_{m-1}$$

    Exchange function

$$E(b_{m-1}b_{m-2}\ldots b_1b_0) = b_{m-1}b_{m-2}\ldots b_1\bar{b_0}$$

$\log N$ **passes of shuffle-exchange function can implement all permutations.**

- **Cube function**

$$C_i(b_{m-1}b_{m-2}\ldots b_1b_0) = b_{m-1}b_{m-2}\ldots \bar{b_i}\ldots b_1b_0$$

for $0 \leq i < m$**.**

- **Plus minus** 2^i **(PM2I) function**

$$PM2_{+i}(x) = (x + 2^i) \textbf{ mod } N$$
$$PM2_{-i}(x) = (x - 2^i) \textbf{ mod } N$$

for $0 \leq i < m$**.**

- **Network performance measures**
  - **Data routing capability:**
    blocking, nonblocking, permutation, multicast, etc.
  - **Hardware cost:**
    the number of links, number of switches
  - **Network Latency**
  - **Bandwidth (data rate)**
  - **Scalability:**
    performance increases as the network size increases

- **Typical interconnection networks**
  - **Static networks**

    Fixed links between nodes, suitable to the applications with communication patterns match the structure of the network.

    * **Ring based networks**
      - **Linear array**

        Degree $d = 2$

        Diameter $D = N - 1$

        Bisection $b = 1$

        Different from a bus.

      - **Ring**

        Degree $d = 2$

        Diameter $D = \lfloor \frac{N}{2} \rfloor$

        Bisection $b = 2$

      - **Chordal ring**

N: number of nodes, even

W: chordal length, odd

Every odd-numbered node p ($p = 1, 3, \ldots, N-1$) is connected to $(p + W)$ mod N (an even-numbered node).

Degree $d = 3$

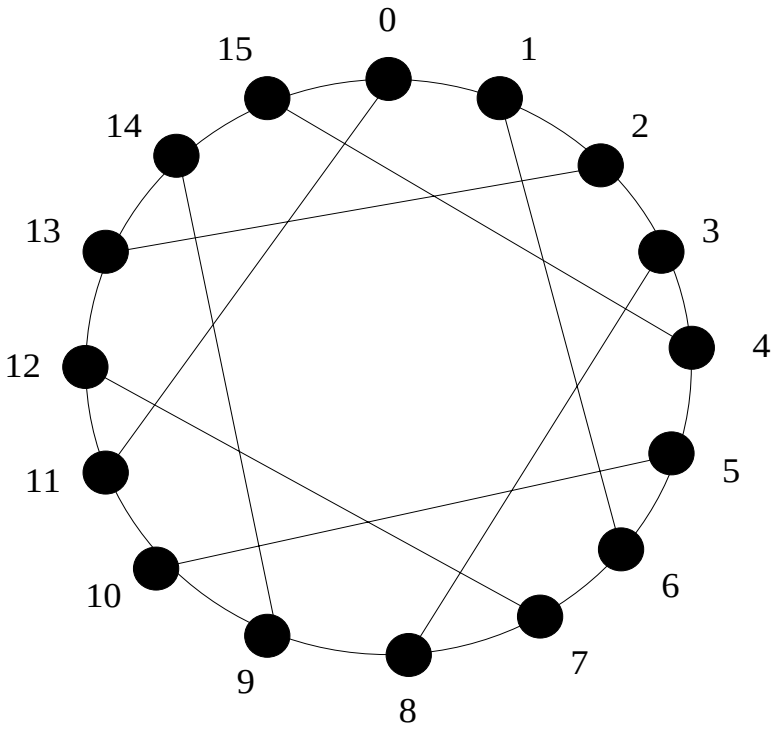
Diameter $D = O(\sqrt{N})$

Bisection $b = 6$

Basic routing algorithm:

Follow ring edge and chordal alternating path to the nearby area, then follow edges within a chordal distance.

- **Further generalization: two chordals.**



A 16 node chordal ring

- **Completely connected**

  Degree $d = N - 1$

  Diameter $D = 1$**.**

- **Barrel shifter**

  $N = 2^n$

  Node i **is connected to node** j **if** $|j - i| = 2^r$ **for** $r = 0, 1, \ldots, n - 1$.

  Degree $d = 2n - 1$

  Diameter $D = n/2$

- **Tree based networks**
  - **Star**

    Degree $d = N - 1$

    Diameter $D = 2$

  - **Tree**

    Binary tree: $N = 2^k - 1$ **nodes**

    Degree $d = 3$

    Diameter $D = 2(k-1)$, $O(\log N)$

    Constant degree, but heavy traffic at root node.

  - **Fat tree**

    Thinking machines Connection machine CM-5 uses this network.

    Basic idea: wider channels towards the root to release the bottleneck but not constant degree any more.

- **Mesh based networks**
  - **k-dimensional mesh**

    $N = n^k$ nodes, n nodes in each dimension and each node has two neighbors in each dimension

    Degree $d = 2k$

    Diameter $D = k(n-1)$
  - **Illiac IV network**

    Two dimensions, 64 nodes, $D = n - 1$
  - **Torus**

    Similar to mesh, but symmetric

    $D = 2\lfloor \frac{n}{2} \rfloor$**.**

    In general, for k-dimensional,

    $D = K\lfloor \frac{n}{2} \rfloor$**.**

  - **Systolic arrays**

    Pipelined array architecture for implementing fixed algorithm. Two dimension, but

the degree is not necessarily 4, can be larger. Matches the communication pattern of the algorithm.

- **Cube based networks**
  - **Hypercube**

    n-cube architecture with $N = 2^n$ nodes.
    - **Geometrical definition: N nodes on the corner of n "cube" in n-space**
    - **Recursive definition: Form a hypercube of dimension n by taking two hypercubes of dimension $n-1$ and directly connecting corresponding nodes.**
    - **Interconnection function:**

      Cube function, connect the nodes iff they have only one-bit difference.
  - **Degree $d = \log N$**

    Diameter $D = \log N$
  - **Easy routing: only need to look at bit i of the destination node at step i.**
  - **Drawback: variable degree, poor expandability.**

- **Cube-connected cycles (CCCs)**

  A hierarchical network.

  Replace each node in an n-cube with a small ring with n-nodes.

  $N = 2^n \times n$ **nodes**

  Constant degree for any n: $d = 3$

  Slightly shorter diameter $D = 2n - 1 + \lfloor \frac{n}{2} \rfloor = O(\log N)$**.**

- **Even poorer expandability**

- **k-ary n-cube networks**

  Radix k ($k = 2$: binary hypercube)

  Each node represented as $a_{n-1}a_{n-2}\ldots a_0$ **with** $0 \leq a_i \leq k-1$ **for** $i = 0, 1, \ldots, n-1$**.**

  n **dimensions, each dimension has** k **nodes, connected as a cycle.**

  Each dimension connected to "plus minus 1" nodes

  e.g. $k = 4$**,** $n = 3$

$N = k^n$ **nodes,** $k = N^{1/n}$**,** $n = \log_k N$**.**

Degree $d = 2n$

Diameter $D = n\lfloor \frac{k}{2} \rfloor$**.**

- **Summary of static networks**

- **Dynamic interconnection networks**
  - **Implement all communication patterns, suitable to general purpose applications.**
  - **Components: switches and sharable links**
  - **Dynamically change the path settings**
  - **Cost of dynamic network: switches and links, usually in terms of crosspoints.**
  - **Performance measures: bandwidth, latency, communication patterns supported.**

- **Types of dynamic networks (in the increasing order of cost and performance):**
  Buses – multistage interconnection networks (MINs) – Crossbars
  - **Buses**
    Time sharing, low cost, very limited bandwidth.
    One transaction at a time. Only one pair of nodes can use the bus. Not scalable, vulnerable to bus controller failures.
  - **Multistage network consists of switch modules and links**
    Switch module: $a \times b$ switch module with a input and b output.
    Crosspoints: ab
    One-to-one connection switch
    One-to-many connection switch
    Legitimate states

Group switches into stages. Connect stages by certain interconnection functions.

* **Crossbar**

  1 stage, the most powerful connecting capability, $O(N^2)$ switches

* **Generalized cube network**

  $N \times N$ network
  $N/2$ switches in each stage
  $n = \log N$ stages, numbered from $n-1$ to 0
  Interconnection function c_i (cube) function for stage i
  Setting switch to swap at stage i realizes c_i function
  Routing algorithm (distributed)

Source $S = S_{n-1}S_{n-2}\ldots S_1S_0$

Destination $D = D_{n-1}D_{n-2}\ldots D_1D_0$

The switch at stage i **in the path from** S **to** D **must be set to swap if** $D_i \neq S_i$ **and set to straight if** $D_i = S_i$**.**

Unique path from S **to** D**.**

Routing example.

* **Data manipulator network**

$N \times N$ **network**

Each stage has N **switching elements**

Each switching element accepts one from three input links and outputs one from three output links (implemented by DE-MUX and MUX)

Interconnection function of stage i**:**

  - $PM2_{+i}$
  - $PM2_{-i}$

- **Straight connection**

Control signals:

- **S - straight**
- **U - up** (-2^i)
- **D - down** $(+2^i)$

Routing:

From source S to destination D. Compute link sum $(D - S)$ **mod** N **and decompose it into the sum of power of 2**

* **Omega network**

$N/2$ 2×2 **switches at each stage**

$\log N$ **stages**

Each stage has identical interconnection function: shuffle exchange

Routing:

Controlled by the address of the destination node

At stage i, if $D_i = 0$ go to upper output of the switch, if $D_i = 1$ go to lower output of the switch.

The number of permutations an $N \times N$ network can realize: $N^{N/2}$.

* **Baseline network (general structure of blocking network)**

- **Clos network (also called $v(m, n, r)$ network)**
  - **Network structure**

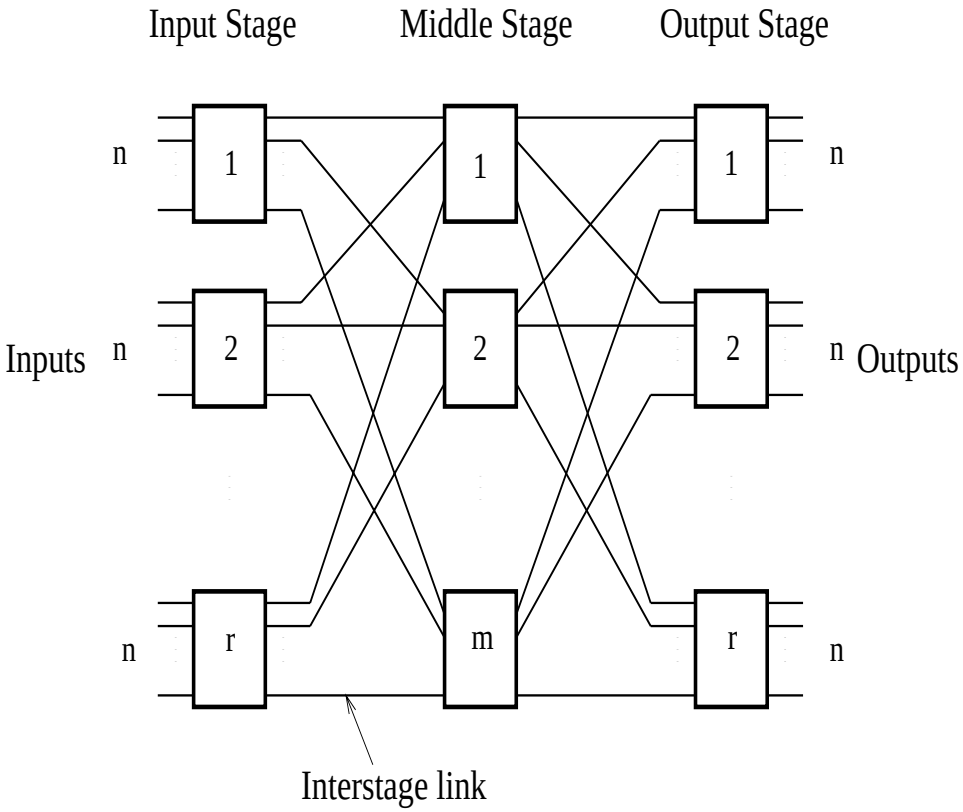
    Three stages of switches

    Input stage: r $n \times m$ switches

    Middle stage: m $r \times r$ switches

    Output stage: r $m \times n$ switches

    $N = nr$ inputs/outputs



- **Rearrangeable permutation network**

  Condition: $m \geq n$

  Rearrangeable for permutation: can satisfy any new connection request from an idle input to an idle output, but sometimes it is necessary to interrupt and rearrange the existing connections in the network.

- **Nonblocking permutation network**

  Condition: $m \geq 2n - 1$

  Nonblocking for permutation: can satisfy any new connection request from an idle input to an idle output and the rearrangement is never required.

- **Multicast network**

  Condition: $m = O\left(n\frac{\log r}{\log\log r}\right)$ **for both nonblocking and rearrangeable multicast.**

– **Proof of rearrangeable permutation condition $m \geq n$**

Basic combinatorial theorem:

Hall's Theorem: Let A be any finite set, and let $A_1, A_2, \ldots, A_r$ be any r subsets of A. A necessary and sufficient condition that there exist a set of distinct representatives $a_1, a_2, \ldots, a_r$ of $A_1, A_2, \ldots, A_r$, i.e. elements $a_1, a_2, \ldots, a_r$ of A such that

$$a_i \in A_i, \quad i = 1, 2, \ldots, r$$

$$a_i \neq a_j \quad for \quad j \neq i$$

is that for each k in the range $1 \leq k \leq r$ the union of any k of the sets $A_1, A_2, \ldots, A_r$ have at least k elements.

Proof.

Suppose

inputs are $1, 2 \ldots, N$

outputs are $1, 2, \ldots, N$

Input switches are $I_1, I_2, \ldots, I_r$

Output switches are $O_1, O_2, \ldots, O_r$

Consider a permutation

$$\{i \to \pi(i), i = 1, 2, \ldots, N\}$$

Let

$$K = \{1, 2, \ldots, r\}$$

For any $K_i \subseteq K$**,**

$$K_i = \{j : \pi(l) \in O_j, l \in I_i\}$$

Consider any k **input switches**

$$I_{i(1)}, I_{i(2)}, \ldots, I_{i(k)}$$

corresponding to

$$K_{i(1)}, K_{i(2)}, \ldots, K_{i(k)}$$

$$K_{i(j)} \subseteq K, 1 \leq j \leq k$$

Consider

$$T = \cup_{j=1}^{k} K_{i(j)}$$

Let $|T| = t$**.**

Note that

$$|\cup_{j=1}^{k} I_{i(j)}| = k \times n$$

This is because that each $I_{i(j)}$ **has** n **distinct inputs, and theses** kn **inputs are connected to** t **output switches with a total of** tn **outputs. Therefore,**

$$tn \geq kn$$

Then we have $t \geq k$**.**

That is,

$$T = \cup_{j=1}^{k} K_{i(j)}$$

has at least k **elements. Then by Hall's theorem, there exists a set of distinct represen-**

tatives

$$k(i) \in K_i, i = 1, 2, \ldots, r$$

$$k(i) \neq k(j)$$

Thus we have a mapping from each input switch to a distinct output switch:

$$I_i \rightarrow K(i)$$

Since all these connections are from different input switches to different output switches, we can direct all connections to a single middle switch.

The remaining network becomes
a $v(m-1, n-1, r)$ **network.**
Note that $v(1, 1, r)$ **is a permutation network.**
By induction on n**, we know that a network with** $m = n$ **can realize all permutations.**

– **Nonblocking permutation network**

$m \geq 2n - 1$**.**

Consider connecting an input from input switch i **to an output of output switch** j**. Note that at most** $n - 1$ **inputs on input switch** i **can be busy and at most** $n - 1$ **outputs on output switch** j **can be busy. So we need one more middle switch to make the new connection. Thus, the number of middle switches needed for nonblocking is**

$$(n - 1) + (n - 1) + 1 = 2n - 1.$$

– **Number of crosspoints**

$$\#cp = 2n \times m \times r + r^2 \times m$$

When $m = n = r$**,** $\#cp = 3N^{3/2}$

– **Generalization to** $2k+1$ **stage for any** $k \geq 1$**. Replacing each** $r \times r$ **middle switch by an** $r \times r$ **Clos network.**

Crosspoints:

$$\#cp = O(N^{1+1/k})$$

for $2k+1$ **stage network.**

- **A special type of Clos network: Benes network. Set** $m = n = 2$ **in Clos network. Recursive construction until all switches become** 2×2 **switches.**

  $2 \log N - 1$ **stages**

  $\#cp = O(N \log N)$

  A $O(N \log N)$ **permutation network.**

- **Summary of dynamic networks**
  - **Buses**

    $\#cp = O(N)$

  - **Multistage networks**
    - $\log N$ **stage networks**

      $\#cp = O(N \log N)$

      Most are blocking networks.

    - **Constant stage networks**

      $\#cp = O(N \cdot N^{1/k})$

      Rearrangeable networks

      Nonblocking networks

  - **Crossbars**

    $\#cp = N^2$