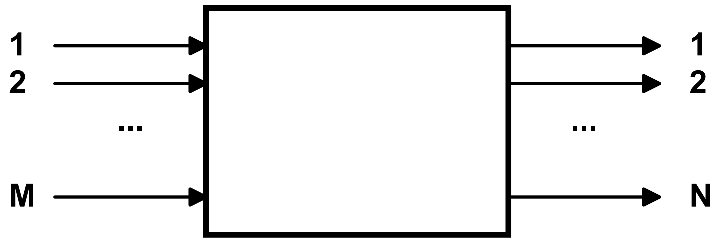
Switching



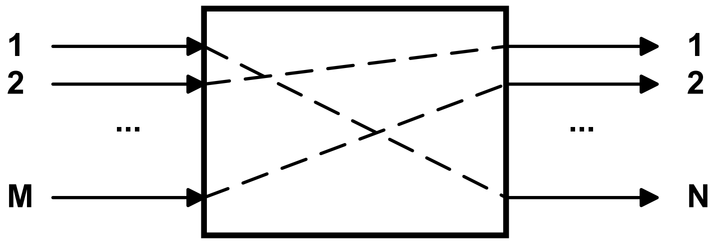
Function: connect inputs to outputs, so that bits or packets that arrive on one link, leave on another designated link.

Performance measures:

- Delay: per packet (ps), per bit (cs)
- Set-up time: per circuit (cs or virtual cs)
- Throuhgput: # lines, rates
- Complexity: # crosspoints (cs), buffer size (ps)

Examples:

a) N$\times$N point - to - point switch:
We want to be able to connect any input to any output, as long as two different inputs do not have the same output



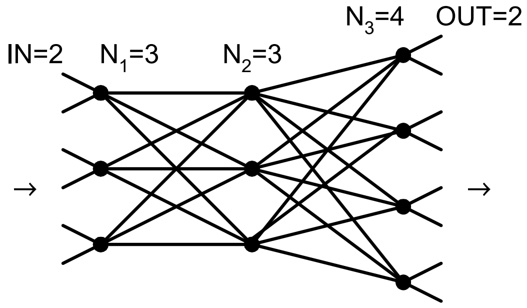
\# different configurations = N!
\# crosspoints $\geq \log_2(N!) = \Theta(N\log N)$

b) N$\times$M multipoint switch
We want to be able to connect any input to any set of outputs, but no two inputs should be connected to the same output.

Clos Networks

Specified by (IN, N_1, N_2, N_3, OUT)

6 x 8 switch



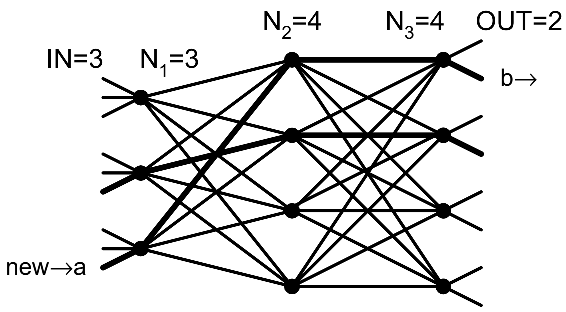
● = elementary cross-bar switch

A switch is **non-blocking** if all the one-to-one connections are compatible.

A switch is **strictly non-blocking** (SNB) if any new connection from a free input to free output is possible without modifying other connections.

Otherwise, it is **rearrangeably non-blocking** (RNB).

Theorem: if $N_2 \geq$ IN+OUT-1, the Clos switch is SNB.



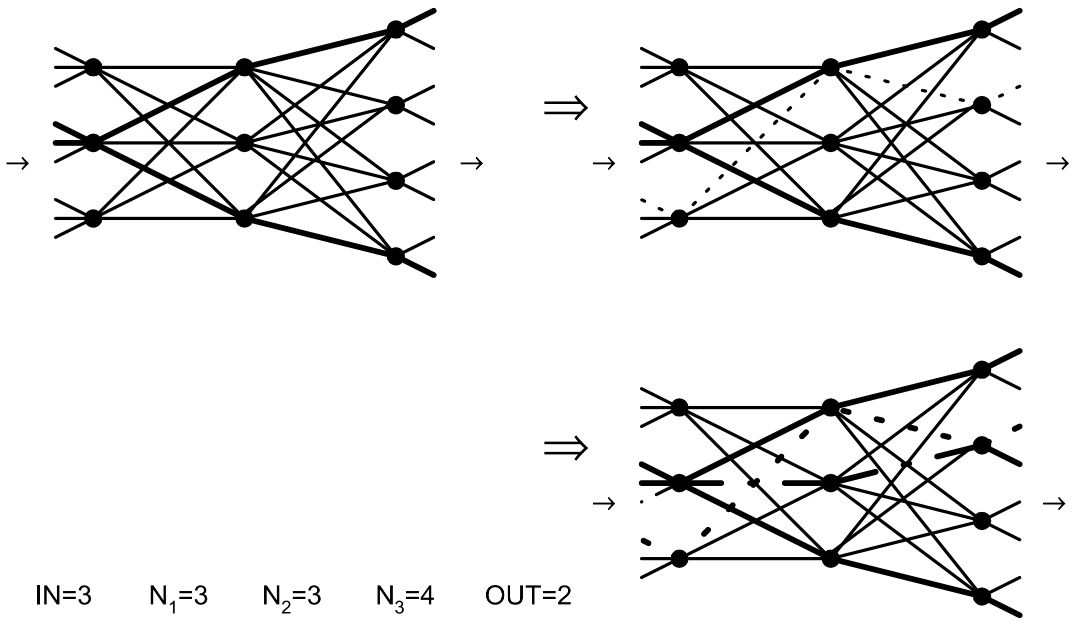
Proof:
Assume we want to set-up a connection from a to b.

a connected to ≤ IN-1 middle switches

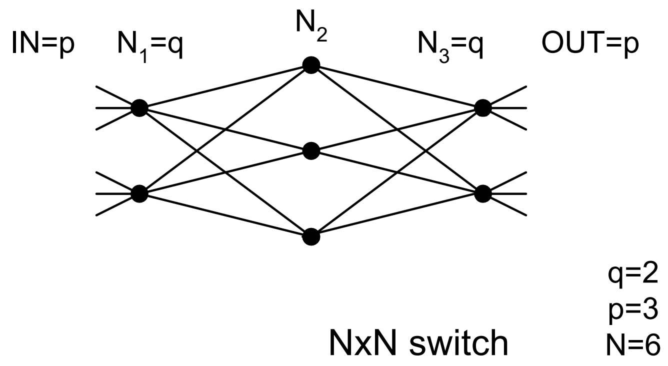
b connected to ≤ OUT-1 middle switches

⇒ there is a middle switch available.

Theorem: if $N_2 \geq$ max{IN, OUT}, the Clos switch is RNB.



Complexity of a Clos switch (example)



$N = pq \Rightarrow$ crossbar requires p^2q^2 crosspoints

RNB Clos: $N_2 = p$

Assume each • is implemented as a crossbar.

crosspoints$=qp^2 + pq^2 + qp^2 = p^2q^2(1/p + 2/q) \ll p^2q^2$

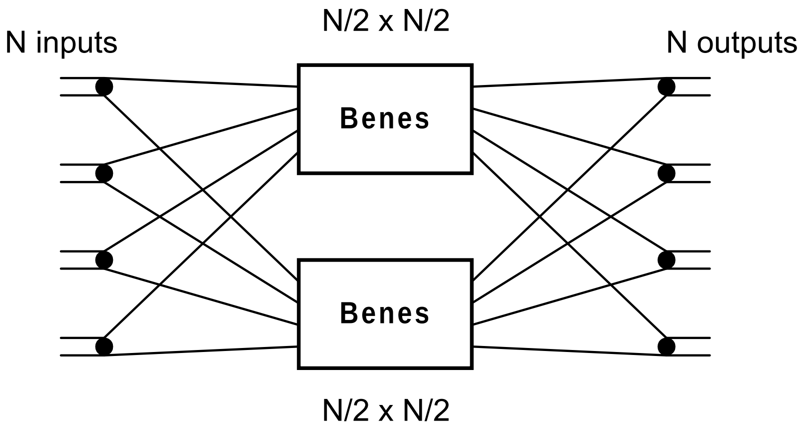
SNB Clos: $N_2 = 2p - 1$

crosspoints $= 2pq(2p - 1) + (2p - 1)q^2 =$
$= 4p^2q + 2pq^2 - 2pq - q^2 =$
$= p^2q^2(4/q + 2/p - 2/pq - 1/q^2)$

e.g. if p = q = 100, SNB has only 6% of the crosspoints of a crossbar.

Benes Network (Rearrangeably nonblocking)

Let $N = 2^n$ and consider the recursive construction:



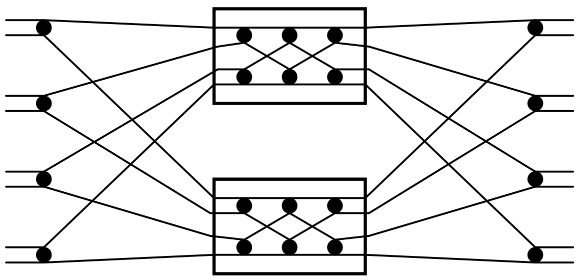
Looks like a (2, N/2, 2, N/2, 2) Clos network

$N_2 = 2 \geq \max\{IN, OUT\} = \max\{2, 2\} \Rightarrow$ RNB
(provided that the
$N/2 \times N/2$ boxes
are RNB)

Each $N/2 \times N/2$ is itself constructed recursively as a Benes network

Each N/2 $\times$ N/2 switch can again be decomposed …

(2,2,2,2,2) Benes



T(N): # of 2 $\times$ 2 switches in N $\times$ N Benes

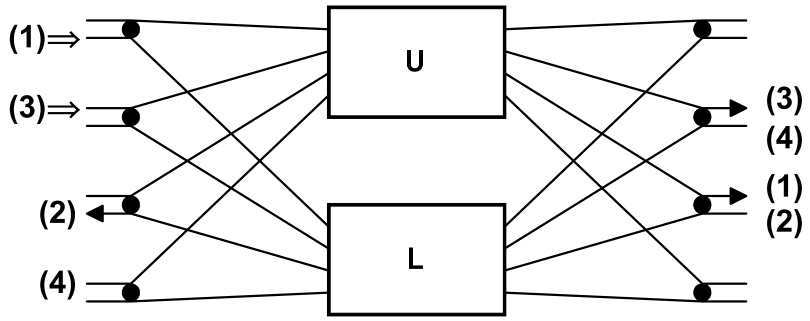
$$\left.\begin{array}{r} T(N) = N + 2T(N/2) \\ T(2) = 1 \end{array}\right\} \Rightarrow T(N) = (2\log_2 N - 1)\frac{N}{2}$$

A 2 $\times$ 2 switch requires 4 cross points: • $\Rightarrow$
$\Rightarrow$ $4N\log_2 N - 2N$ crosspoints = optimal order

Example: N=10^4 $\Rightarrow$ $5 \cdot 10^5$ as opposed to 10^8 required by a crossbar

Routing in a Benes Network

Start idle. Let C be a desired set of connections.



Step 1: Select input not yet selected.
(Terminate if none exists)

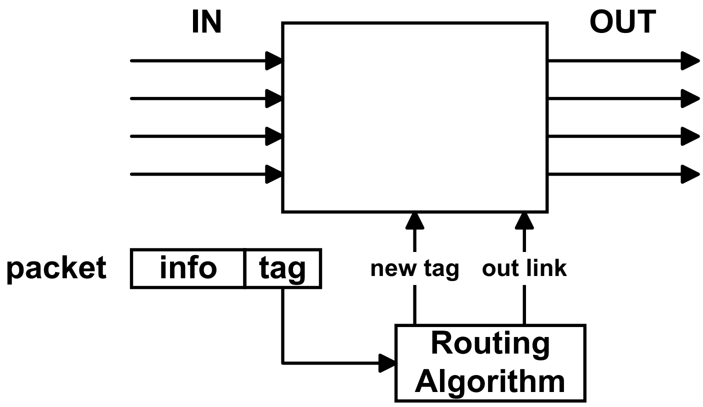
Step 2: Connect input to desired output (of S, say) via U

Step 3: If other output of S desires connections to input of T(say), connect via L
Go to step 1.

Apply the algorithm recursively to U and L

Fast Packet Switching (FPS)

The frequency of changes in the switch configuration is orders of magnitude greater in the case of packet switching than in circuit switching (μs, or even ns, as opposed to seconds)

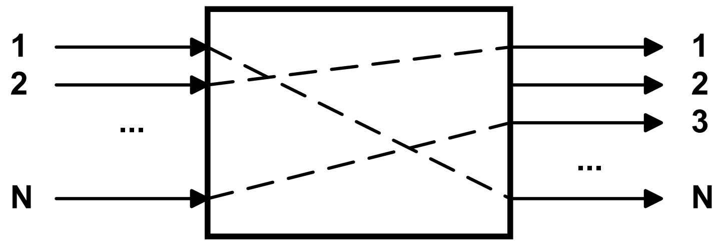


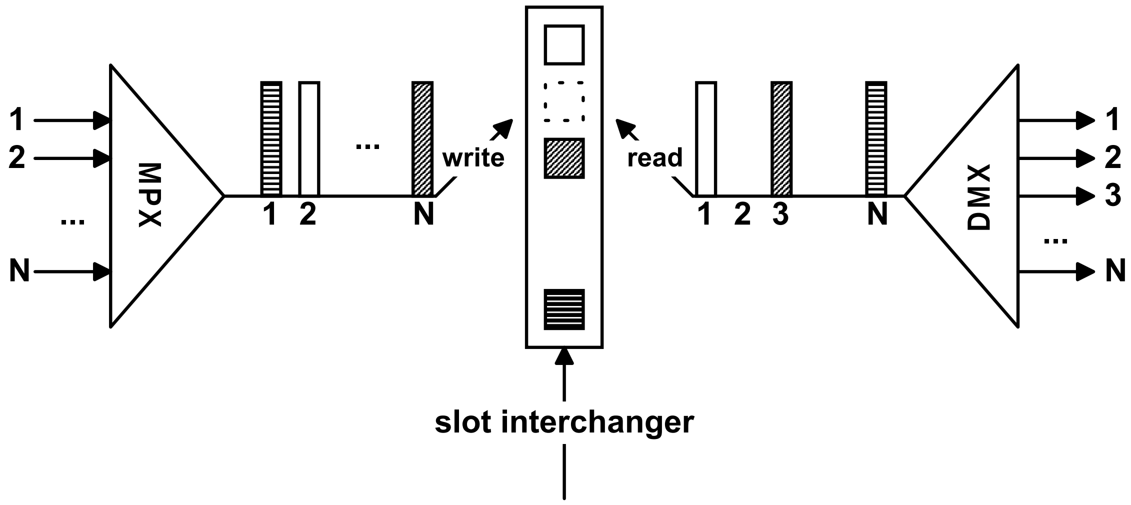
Tag = { destination address (datagram)
virtual circuit identifier (VC switching)

Advantage of FPS: flexible bandwidth allocation

Disadvantage of FPS: buffering is needed

Time Division Switching (e.g. telephone network)





Delay ≤ 2N slots

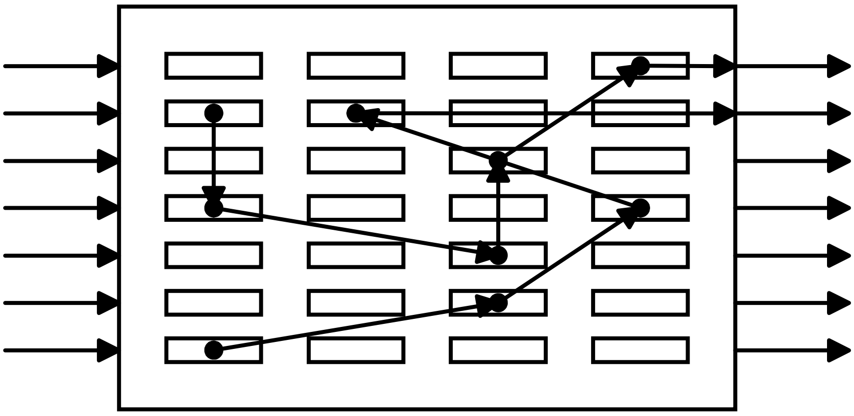
Disadvantage: internal rate is N times larger

Basic Designs

Shared buffer   Distributed buffer   Output buffer

Shared Buffer

One linked list per output. One linked list of free buffer.

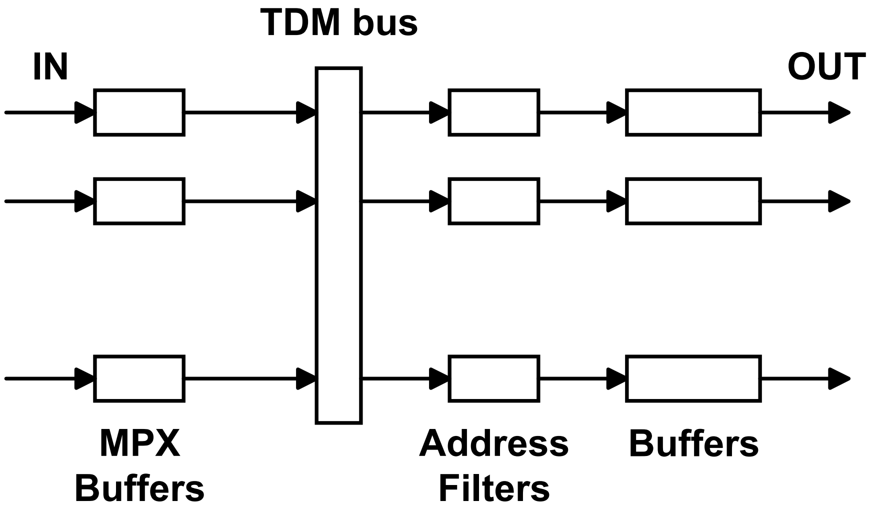


Buffer

Advantage: maximum utilization of buffers.

Disadvantage: fast pointer manipulation needed.

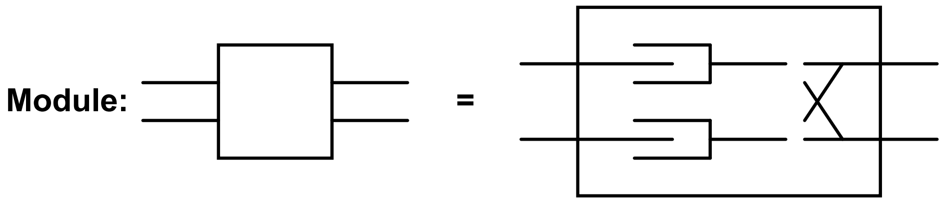
Output Buffers

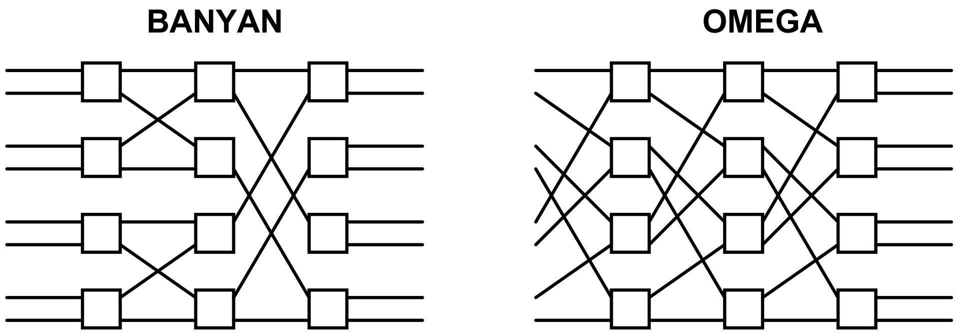


Advantage: Conceptually simple

Disadvantage: Requires fast bus

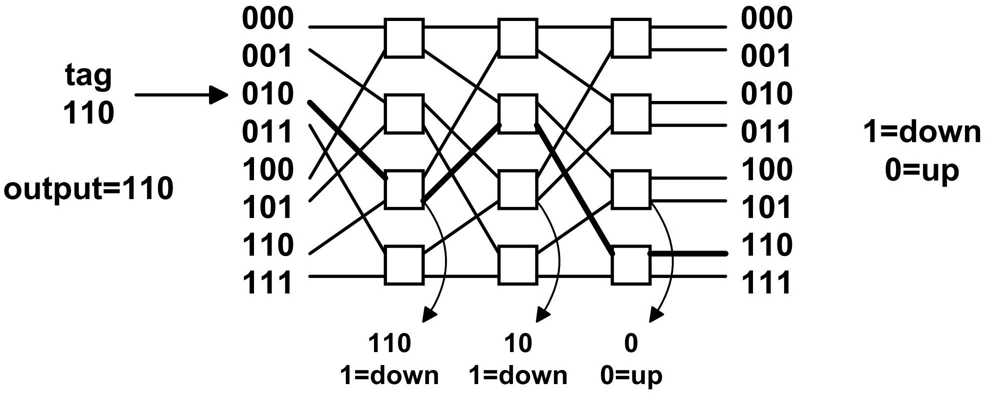
Distributed Buffer





(other examples are Baseline, Flip, Delta, etc)

Self-routing in Omega (others are similar)



Here the tag is the output port number.

In the i^{th} stage, if corresponding (i^{th}) bit of tag is 1 it goes down and if it is 0 it goes up.