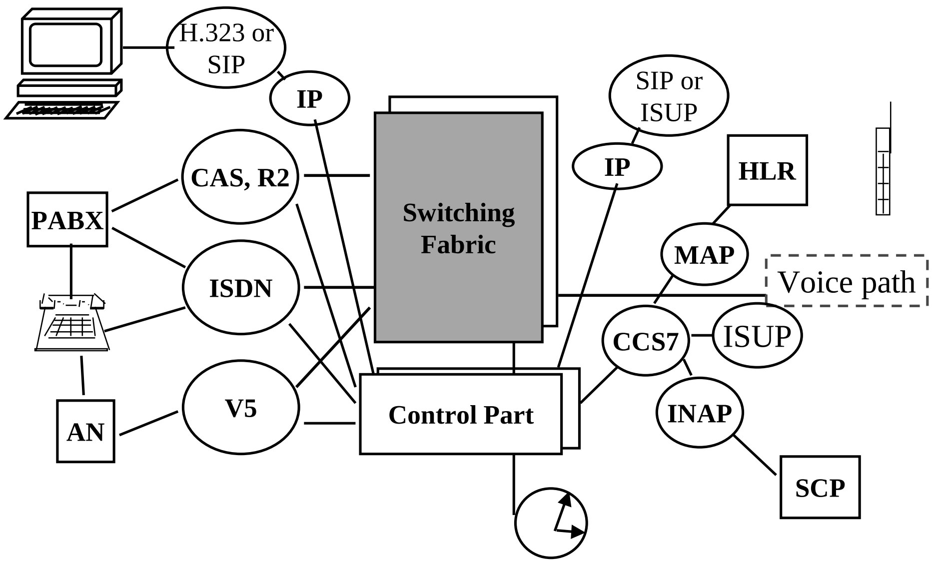
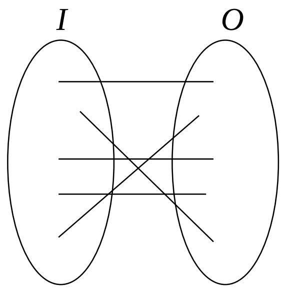
Lower bound of Complexity using Connection functions

Copy -function, Self-routing fabrics

Summary of course scope



Total nrof thruconnections in a Fabric

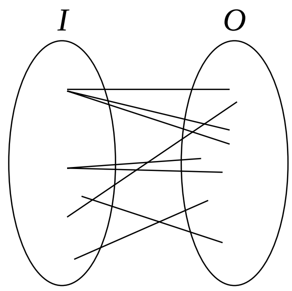


Point-to-point connection function

$C = \{(i,o) | \; i \in I, \; o \in O\}$

$(i,o) \in C$ ja $(i,o') \in C \Rightarrow o = o'$

$(i,o) \in C$ ja $(i',o) \in C \Rightarrow i = i'$



One to many

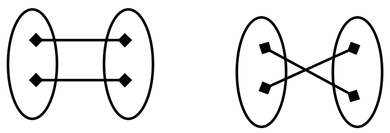
$C = \{(i,n_i) | \; i \in I, \; n_i \subset O\}$

C - is a logical mapping from inputs to outputs.

C - a description of a connection of state of the Fabric

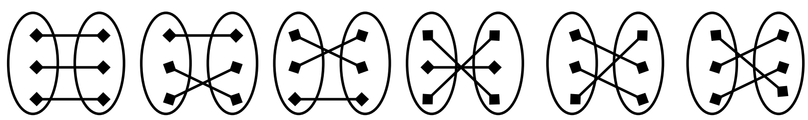
Nrof point-to-point thruconnections is N! Let us visualize mappings C

N = 2



2! = 2

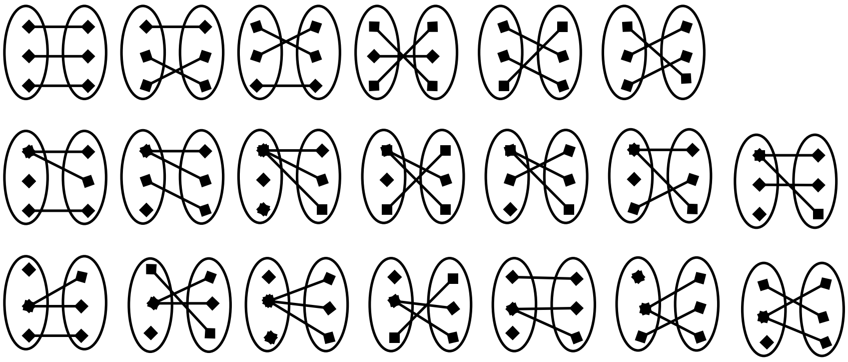
N = 3
3! = 6



Construction of C: Enumerate inputs, mix them in an arbitrary order.

The impact of multi-point thruconnections?

Mapping C, when N = 3



etc…

In mapping C each input can freely choose its output --> N^N elements in C!
Such C is significantly larger than in case of pt-to-pt connections!

Combinatorial complexity of a Fabric

ζ(G) - Log_2 of nrof distinct and legitimate C realized by graph G.

R **- Nrof cross-points in the Fabric.**

2^R - Nrof states in a Fabric with *R* cross-points.

Crude upper bound:

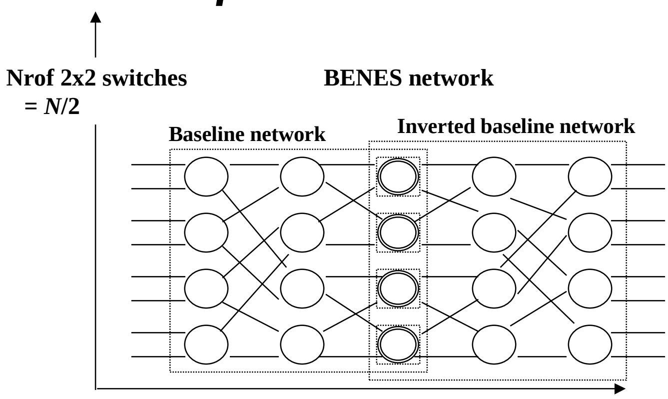
$\zeta \leq R$

More accurate upper bounds:

- remove all non-legitimate states, e.g. the ones in which two cross-points are feeding one output

- remove one of states for which another state produces the same *C*.

Growth of the Benes network



Nrof stages = 2 $\log_2 N$ - 1

Nrof cross-points is nrof-stages × nrof-switches-in-a-stage =

$4 \times N/2 \times (2 \log_2 N - 1) \approx 4N \log_2 N$

Lower bound of complexity of a fabric can be assessed using connection functions

- ✔ **Assume that the fabric is NxN and that it provides full connectivity.**
- ✔ **Nrof C = N!**
- ✔ $\zeta = \log_2 (N!) \sim N\log_2(N) - 1{,}44N + ½\log_2(N)$
- ✔ **Nrof 2x2 switches in a Benes network is**

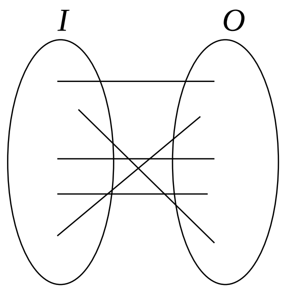
  $(N/2)(2\log_2 N-1) = N\log_2(N) - ½N \sim \zeta$

  = approximately the minimum nrof switches to implement N! states.

Connection functions characterize the goal of a fabric

C - a connection pattern

Pt-to-pt connection function



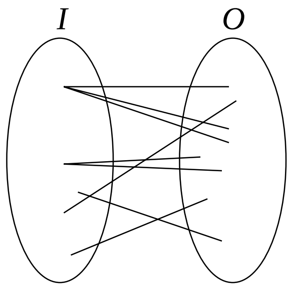
Point-to-point

$C = \{(i,o) | \ i \in I, o \in O\}$

$(i,o) \in C$ ja $(i,o') \in C \Rightarrow o = o'$

$(i,o) \in C$ ja $(i',o) \in C \Rightarrow i = i'$

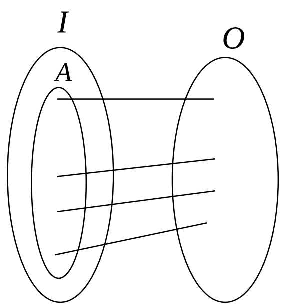
Multicast -function



One to many

Goal: Count the total nrof thruconnections in a Fabric and consequently the lower bound of combinatorial complexity of the fabric.

Concentrator function

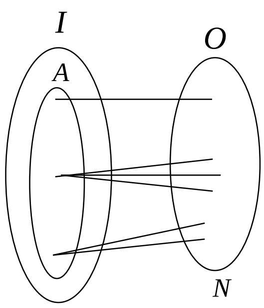


$C = \{(i,o) | \ i \in A \subset I, o \in O\}$

$(i,o) \in C$ ja $(i,o') \in C \Rightarrow o = o'$

$(i,o) \in C$ ja $(i',o) \in C \Rightarrow i = i'$

Copy function



$C = \{(i,n_i) | \ i \in I, \Sigma n_i = N\}$

The order and identity of outputs n_i are ignored (output unspecific).
NB: Copy function =/= multicast!!!

Let us observe the nrof different C measured as ζ realized by different connection functions

$\zeta_{\text{pt-pt-conn-function}}$

$\zeta_{\text{multicast-function}}$

$\zeta_{\text{concentrator-function}}$

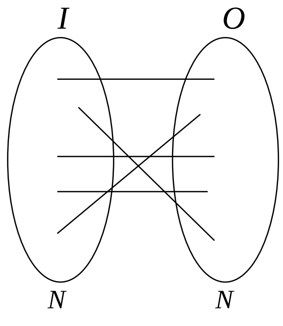
By definition we say that graph G is rearrangeably non-blocking, if it realizes all connection patterns *C*.

=> $\zeta_f \leq \zeta(G)$

It follows that by observing the nrof distinct C realized by different connection functions, we can find the lower bound of complexity for any rearrangeably non-blocking fabric.

Lower bound of complexity of the Point-to-pt-connection function

Pt-pt-connection function



One to one mapping

Each distinct *i* is mapped to exactly one *o*.

We need to implement the nrof distinct *C* equal to *N*!

Using Sterling's approximation:

$N! \approx \sqrt{2\pi}\, N^{N+1/2} e^{-N}$

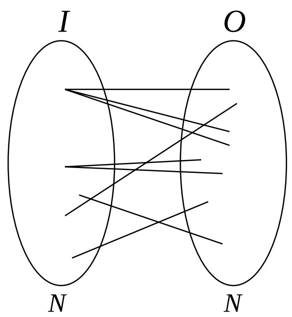
$= \sqrt{2\pi}\ \exp_2(N \log_2 N - N \log_2 e + 1/2 \log_2 N)$

$\zeta_{\text{pt-pt}} = \log_2 N! \approx N \log_2 N - 1.44N + 1/2 \log_2 N$

NB: Nrof 2x2 switches in a Benes network is $N \log_2 N - N/2$, i.e. very close to this lower bound of $\zeta_{\text{pt-pt.}}$

Lower bound of complexity of the Multicast connection function

Multicast function



$C = \{(o, i) | i \in I, o \in O\}$

Each $o \in O$ is connected to some $i \in I$.

Each *o* can choose any *i*.

=> we need to implement

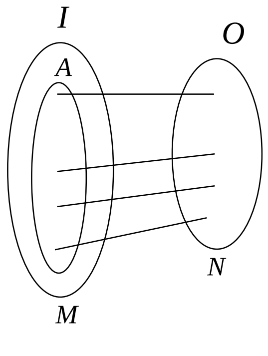
$N^N = \exp_2(N \log_2 N)$ connection patterns *C*.

$$\zeta_{\mathbf{mtcast}} - \zeta_{\mathbf{pt\text{-}pt}} \approx \mathbf{1.44}\ N$$

A Fabric that would implement multicast and would be close to the lower bound of complexity is not known. It is known that Benes network implements multicast if the nrof 2x2 switches in it is doubled compared to the pt-to-pt case.

Lower bound of complexity of the Concentrator connection function

Concentrator function



$C = \{i | i \in A, nrof\ i = N\ (< M)\}$

There are $\binom{M}{N}$ connection patterns C

$$\zeta_{\mathbf{connentrator}} = \log_2 \frac{M!}{N!\ (M-N)!}$$

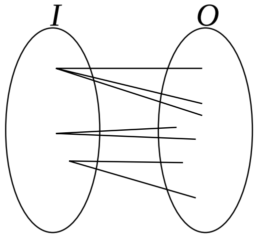
$$\zeta_{\mathbf{connentrator}} = M\,H(c) \qquad c = N/M$$

$$H(c) = -c \log_2 c - (1 - c) \log_2 (1 - c)$$

=>Theoretically, the complexity of a concentrator is a linear function of the nrof inputs. Practical solutions of this level of complexity are, however, unknown. Moreover, strict sense non-blocking concentrators are needed => M logM fabrics are used for concentration.

Nrof of connection patterns C for a Copy function

I *O*



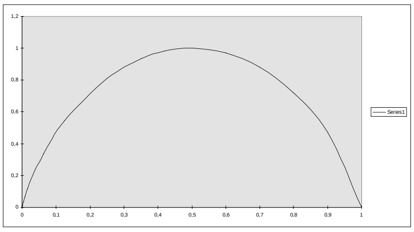
M inputs *N* outputs

$C = \{(i,n_i) | \ i \in I, \Sigma n_i = N\}$

Nrof of different *C* is

$$\binom{M-1+N}{M-1}$$

This is = in how many ways *N* objects can be put in *M* bins.



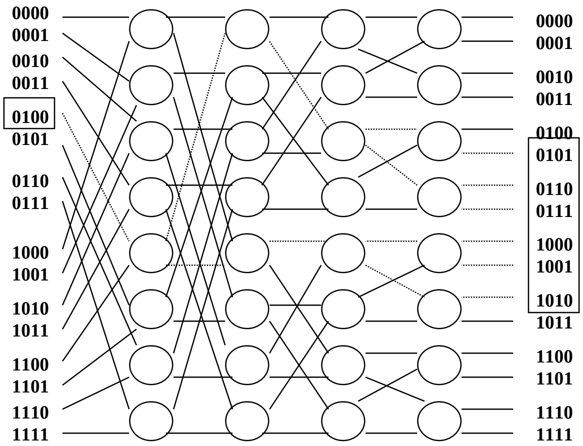
$$\zeta \geq (M-1+N)\, H\left(\frac{M-1}{M-1+N}\right)$$

$$H(c) = -c \log_2 c - (1-c)\log_2(1-c)$$

Creating copies using a binary network

- ✔ **Copy function can be implemented using a distribution tree prior to network that rearranges the signals to the right outputs.**
- ✔ **Copy function needs the identity of the input and the nrof copies.**
- ✔ **Routing/copy decision in the network is based on the binary identities of outputs in the copy interval.**

A distribution tree based on Banyan network

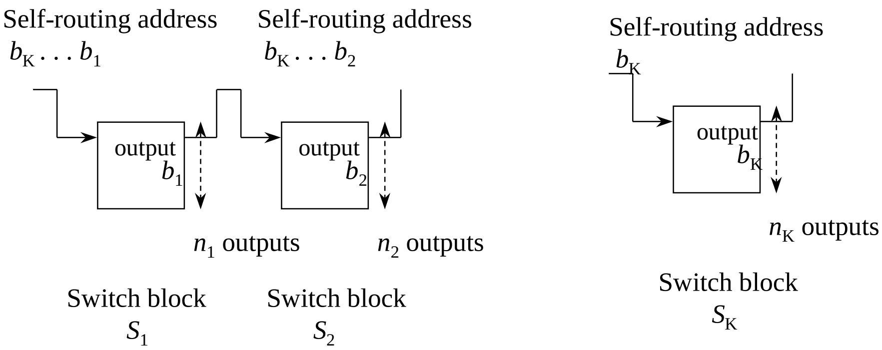


A Multicast fabric can be constructed recursively e.g. =Copy-swich + pt-to-pt switch

- ✓ **Leads to an increase in the nrof stages, which may be undesirable.**
- ✓ **It is difficult to calculate a path thru a multi-stage fabric, controlling the fabric is also complicated.**
- ✓ **A way to solve the problem is *self-routing*.**

Self-routing is based on Input/output addresses that tell also the path

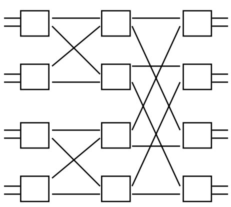
Self-routing is a popular principle in fast packet switching.
Each packet has a header, that is used to find the path thru the fabric.
There are one or more paths from an input to an output.



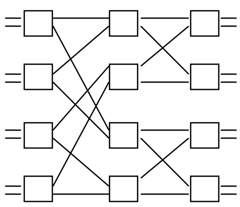
We call the switch blocks *nodes*.

NlogN complexity unique route networks are variations of the Baseline network

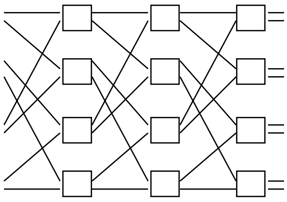
Banyan network



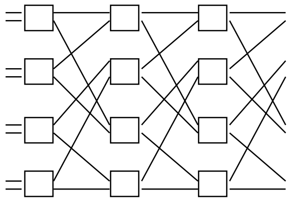
Baseline network



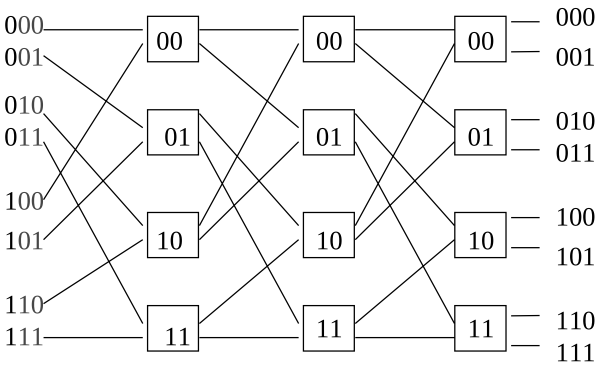
Shuffle exchange (omega)



Flip network (inverse shuffle)

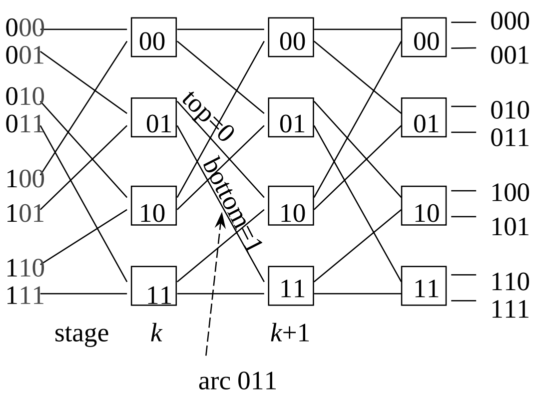


Self-routing shuffle network arcs are enumerated in a regular manner



$N = 2^n$ inputs and 2^n outputs. Benes structure => Nrof stages = n and
Nrof 2x2 switches in each stage = $N/2 = 2^{n-1}$.
2x2 switches in a stage are enumerated top to bottom 0 . . . 2^{n-1}- 1 .
Enumeration requires n - 1 bits.
The arcs between stages are enumerated top to bottom with n bits.

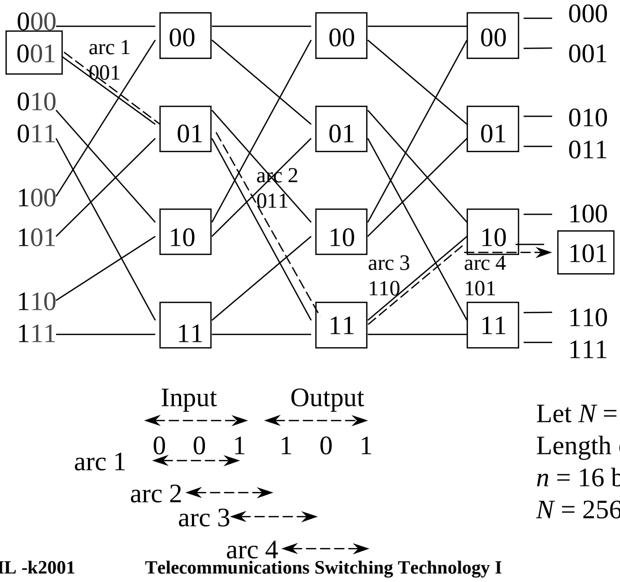
In the self-routing shuffle network, the number of target output is the self-routing address



Number of a node in stage k = number of the arc - remove right most bit

Number of the node in stage k +1 = number of the arc - remove left most bit

Self routing example



Let N = 64 000
Length of the address is
n = 16 bits.
N = 256 , Length = 8 bits.

Ratkaisun rajoitukset

✓**Banyan network is not non-blocking**.

✓It implements $\exp_2(1/2N \log_2 N) = (N^N)^{1/2}$ connection pattern.

✓This is less than the required $N! \approx \exp_2(N \log_2 N)$.

=>

✓The nrof arcs between nodes can be increased, or

✓Shuffle can be replicated like in the Cantor network, or

✓Two shuffle network can be concatenated (cmp BENES network).

✓Also buffering in intermediate nodes can be used.

✓ Simpler to use a stright switch matrix whenever possible.

Broadband (Terabit-) IP routers need a self-routing switch fabric

- ✔ **Wire speeds can reach tens…hundreds of Gbit/s. Processing electronically requires splitting the input into many. Total nrof input can become large (thousands).**
- ✔ **Simply connectivity from all inputs to all outputs requires a switch fabric.**
  - **A Switch fabric is the only feasible solution when the total switching speed grows so high that no bus structure is fast enough to carry that load or when a scalable solution is needed.**
- ✔ **A Self routing fabric is the only well functioning solution, because a different thruconnection is needed for each IP-packet. No centralized control would be fast enough and scalable enough!**