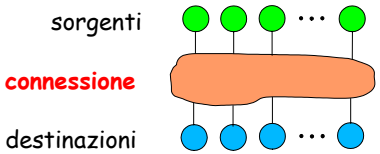
Data switches

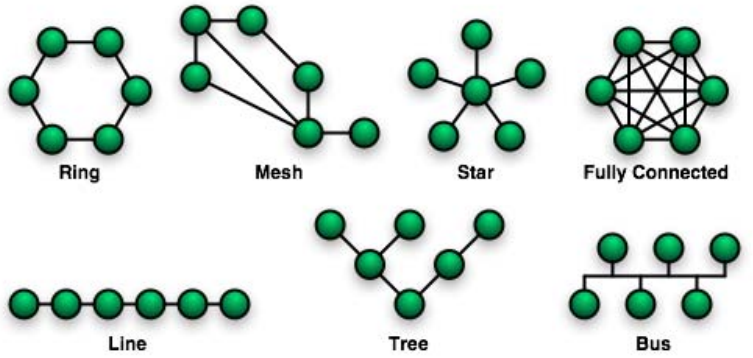
In un esperimento è necessario **connettere** in modo dinamico i rivelatori (**sorgenti** di dati) con processori (**destinazioni**) che eseguono calcoli e provvedono a registrare i dati stessi.



Sorgenti e destinazioni sono chiamati **nodi** o *hosts*

I dati si muovono dalle sorgenti verso le destinazioni, ma anche il percorso inverso in generale è richiesto per funzioni di controllo.

Qual'è il modo più conveniente per realizzare la connessione tra sorgenti e destinazioni?



Terminologia per una rete (*fabric*) di N nodi (*hosts*)

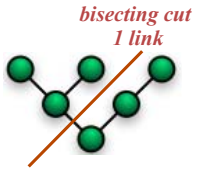
partition:
dividere la rete in due (o più) sottoinsiemi che non comunicano tra di loro

connectivity:
numero di vie possibili tra due nodi, oppure minimo numero di connessioni che si devono togliere per dividere la rete in due sottoinsiemi indipendenti (*partition*)

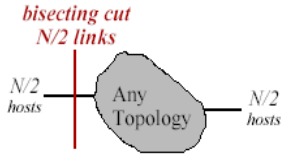
bisettrice *(bisecting cut, bisection width)*:
divide la rete in due gruppi, ciascuno di N/2 nodi

Se i due sottoinsiemi contengono lo stesso numero di nodi N/2:
connectivity = bisecting cut

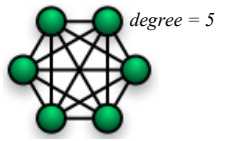
bisettrice minima:
minimo numero di connessioni tra due gruppi, ciascuno di $N/2$ nodi



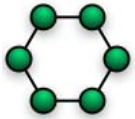
bisettrice completa (*full bisection*):
quando il numero minimo di connessioni tra due gruppi qualsiasi, ciascuno di $N/2$ nodi, è uguale (o superiore) a $N/2$



degree:
numero di *links* uscenti/entranti in un nodo



Se per connettere due nodi si deve transitare attraverso nodi intermedi:



diametro massimo

numero massimo di *links (hops)* percorsi dalla connessione più breve tra due nodi: è una misura della latenza massima.

diametro medio:

numero di *links (hops)* percorsi dalla connessione più breve tra due nodi scelti a caso

distanza media *(average distance):*

numero di *links (hops)* successivi verso una destinazione aleatoria

Secondo la destinazione del messaggio:

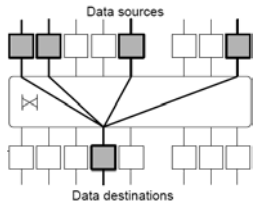
unicast:

connessione (messaggio, *call*) da un nodo verso **un solo** nodo
esempio: una connessione (*call*) telefonica ordinaria

multicast:

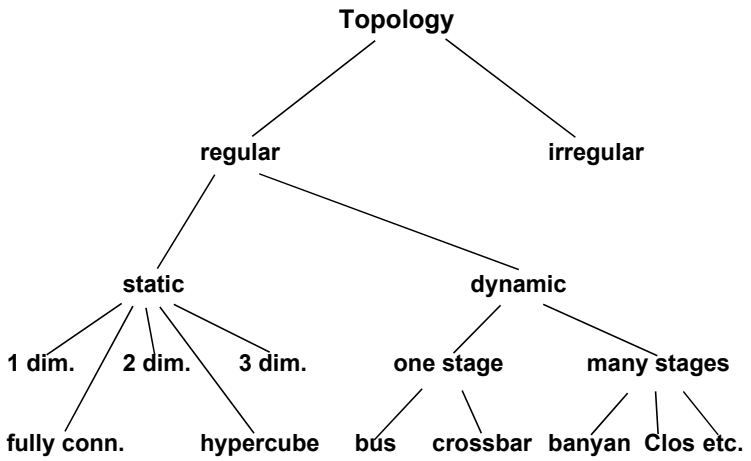
connessione (messaggio) da un nodo verso **più di un** nodo

Nota: In un esperimento i dati si muovono da molte sorgenti verso una sola destinazione, situazione poco frequente nelle reti commerciali!



broadcast:

connessione (messaggio) da un nodo verso **tutti gli altri** nodi
esempio: allarme



Topologie commerciali disponibili scelte secondo criteri specifici per:

reti di calcolatori

LAN (es. Ethernet), MAN, WAN

I nodi sono processori diversi ma abbastanza omogenei

telecomunicazioni

I nodi sono svariatissimi

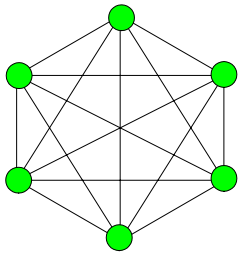
supercomputers

MPP (Massively Parallel processors), (symmetric) multiprocessors

Tutti i nodi sono rigorosamente identici

Topologie (1)

Interconnessione completa di tutti gli N nodi (*hosts*)



degree, numero di connessioni ad ogni nodo: $N-1$

diametro massimo, numero massimo di *links* nella connessione piu breve tra due nodi: **1**

distanza media: 1

bisection width: $(N/2)^2$

Due modi di funzionamento possibili:

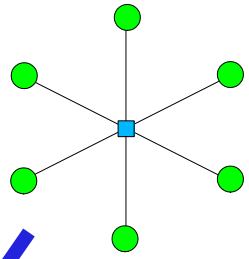
- ogni nodo, quando invia dati, li invia simultaneamente a tutti gli altri nodi: connessione statica e *broadcast*
- ogni nodo sceglie il destinatario dei dati

Topologie (2)

	Degree	Max_Diam	Ave Dist	Bisection
1D mesh	≤ 2	N-1	N/3	1
2D mesh	≤ 4	$2(N^{1/2} - 1)$	$2N^{1/2} / 3$	$N^{1/2}$
Ring	2	N / 2	N/4	2
2D Torus	4	$N^{1/2}$	$N^{1/2} / 2$	$2N^{1/2}$
n-Hypercube	n	n = LogN	n/2	N/2

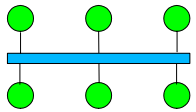
Topologie (3)

Connessione di ogni nodo a un punto centrale (*hub*)

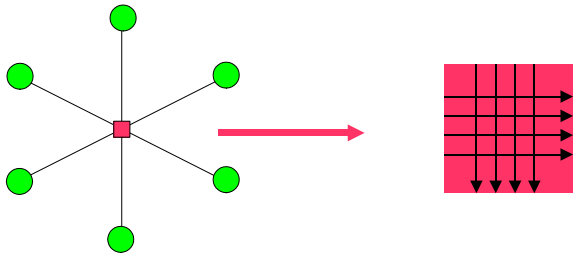


Hub = Bus

banda passante $\propto N$



Topologie (4): *Hub = Switch*

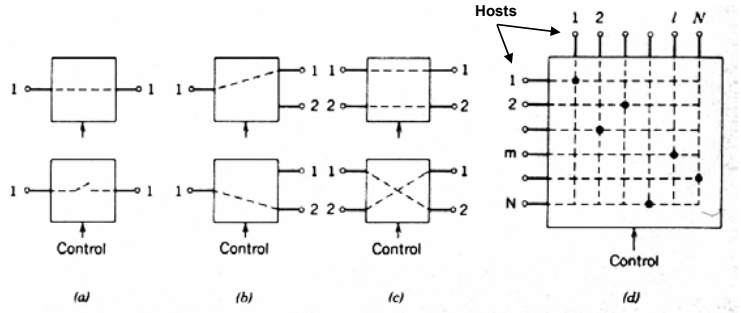


È concettualente semplice aumentare le dimensioni $N \times N$ di un *crossbar switch,* ma il numero di elementi e la potenza salgono come $O(N^2)$

È possibile costruire l'equivalente di un grande *crossbar switch* con elementi connessi in stadi successivi? Rete di piccoli *crossbar switches* o altra topologia?

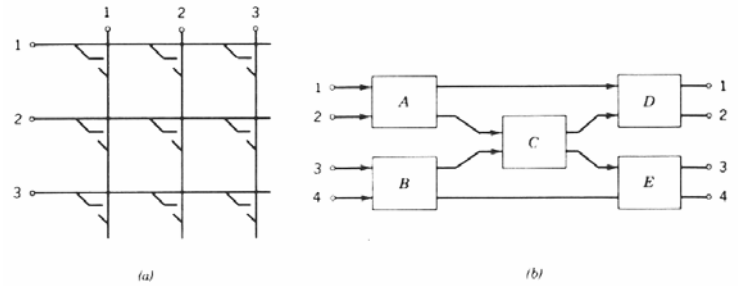
Data Switches

Servono a interrompere (interruttore) o a commutare (commutatore) un flusso di dati tra due nodi (*hosts*)



(a) - 1 x 1 *(on-off)* serve a connettere o a disconnettere due linee
(b) - 1 x 2 connette una linea con una delle altre due
(c) - 2 x 2 *(crossbar)* connette due linee a due linee
(d) - N xN (*crossbar*) connette N linee a N linee

Nel caso *crossbar (a)* ogni linea d'ingresso può in ogni situazione essere connessa ad **una** linea d'uscita (unicast) senza bloccaggio: ***non-blocking***.
Ciò non è vero in generale per altre topologie, per esempio *(b)*.



(a) - *switch* 3 x 3 costruito con 9 *switches* 1 x 1
(b) - *switch* 4 x 4 costruito con 5 *switches* 2 x 2

rearrangeable

se la rete (*switch*) è capace di connettere ogni permutazione di connessioni tra ingressi e uscite.

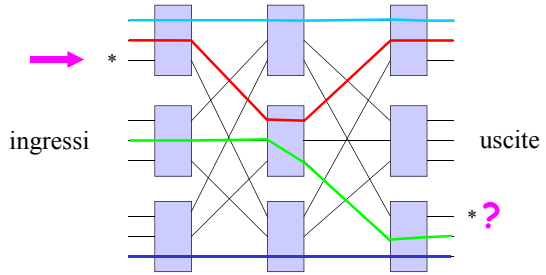
Per esempio, uno switch 8x8 deve essere capace di stabilire le connessioni

dagli ingressi	0	1	2	3	4	5	6	7
	|	|	|	|	|	|	|	|
alle uscite	3	5	1	0	7	6	2	4

e tutte le rimanenti 8! - 1 = 40319 permutazioni possibili.

Una rete *rearrangeable* possiede bisettrice completa, ma una rete con bisettrice completa non è necessariamente *rearrangeable*.

***blocking* interno**:
quando non si può stabilire una connessione (***call***) tra un ingresso e una uscita entrambi liberi



***nonblocking* interno**:
strict-sense nonblocking se c'è sempre almeno una via disponibile tra un ingresso e un'uscita liberi
rearrangeably nonblocking se per una nuova connessione tra un ingresso e un'uscita liberi può essere necessario modificare connessioni pre-esistenti

blocking esterno, **alla periferia dello *switch***

Head Of the Line (HOL) *blocking:* code in ingresso o perdita di dati

output blocking: code in uscita

self-routing

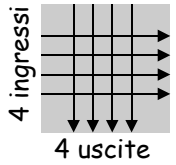
se lo *switch* elementare interpreta al passaggio l'informazione di destinazione contenuta nel pacchetto. Altrimenti è necessario un controllore esterno che determina il percorso più appropriato

speedup

rapporto tra la velocità di trasferimento dati dall'ingresso all'uscita dello *switch* e la velocità di linea in ingresso

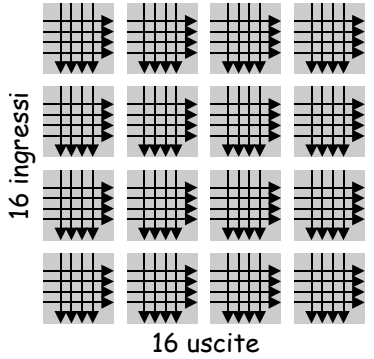
Crossbar switch costruito con circuiti elementari

circuito elementare:

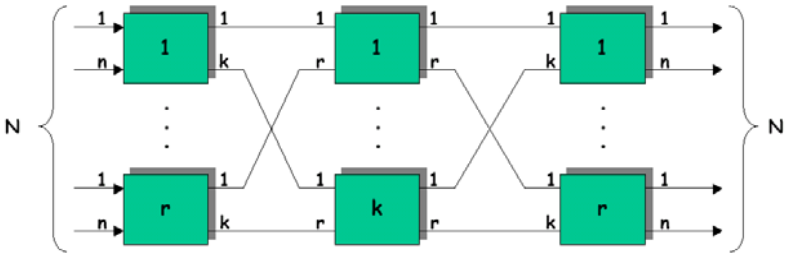


Si devono trasmettere ingressi e uscite all'elemento successivo!

16x16 *crossbar switch*:



Clos Network (1)



$N = nr$ nodi in ingresso $N = nr$ nodi in uscita

gli **stadi di ingresso e uscita** (***leaves***) sono uguali

hanno r elementi asimmetrici $k \times n$

lo **stadio intermedio** (***spine***)

ha k elementi simmetrici $r \times r$

Clos network (2)

La dimensione dello stadio intermedio determina le proprietà di bloccaggio:

strict-sense nonblocking

quando $k \geq 2n-1$ per connessioni ***unicast***

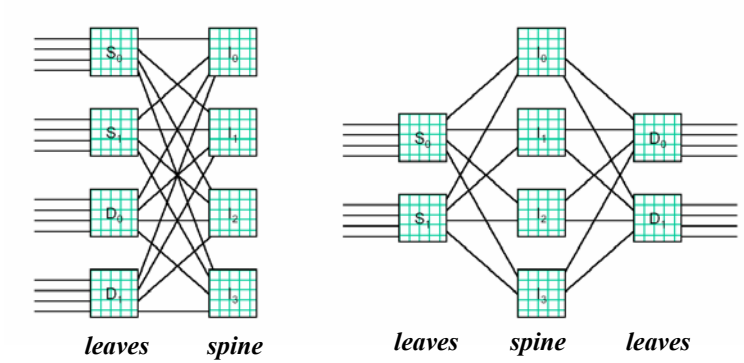
rearrangeably nonblocking

quando $k \geq n$ per connessioni ***unicast***

blocking **in generale** per connessioni ***multicast***

Una rete di Clos con N ingressi e N uscite ha $\in 6N^{3/2}$ circuiti, mentre il *crossbar* corrispondente ne ha N^2 .

Clos Network (3)

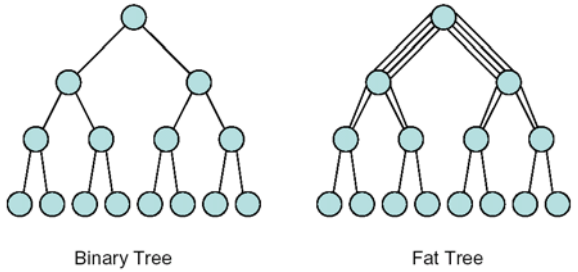


Quando i *links* sono bidirezionali non c'è distinzione tra ingressi e uscite

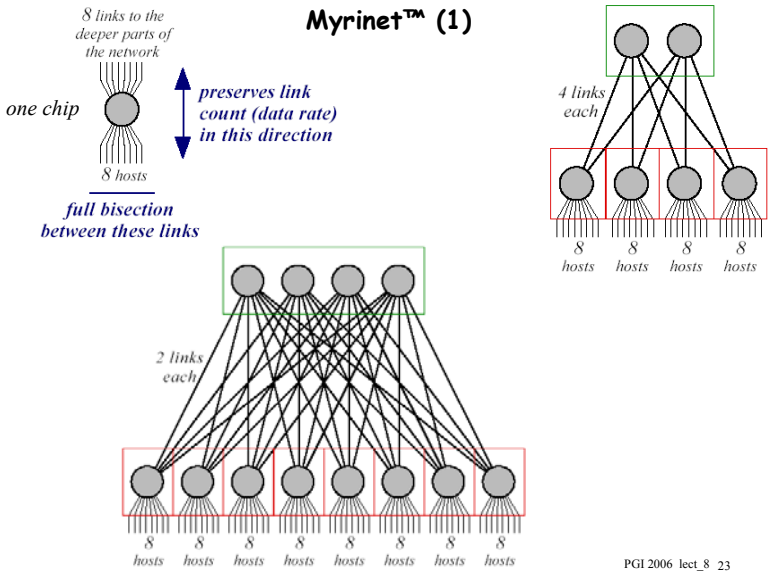
Le reti di Clos sono anche chiamate:

Constant Bisectional Bandwidth (**CBB**)

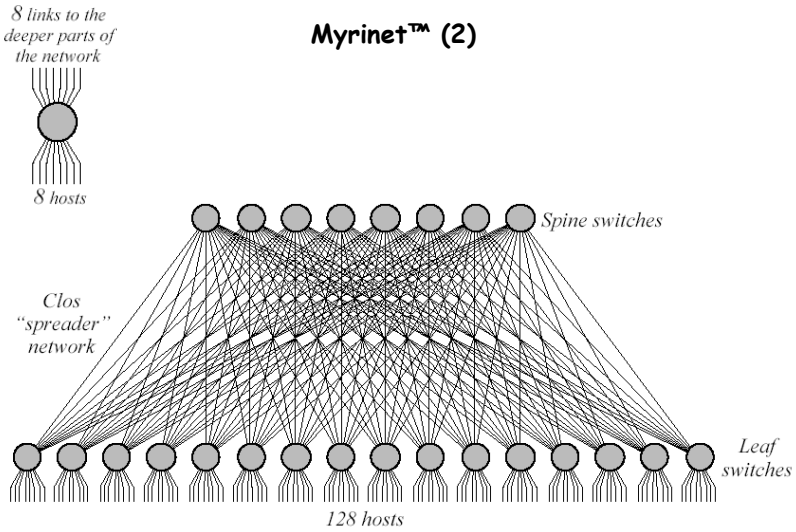
Sono un caso particolare di ***Fat Tree***



Myrinet™ (1)

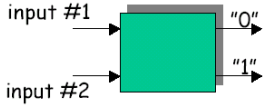


Myrinet™ (2)



Banyan Network (1)

Una rete di Banyan è costituita da molti stadi di *switches* elementari.

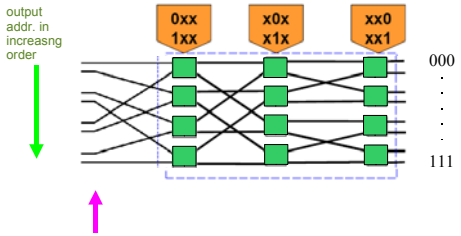


Monodirezionale (sorgenti ==> destinazioni)
Self-routing

Costruita in modo recursivo

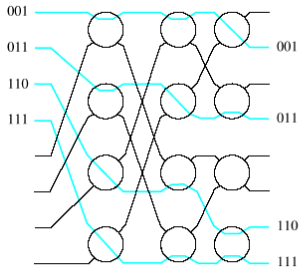
Scalabile, la complessità cresce come $N \log_2 N$

Strict-sense nonblocking
se i pacchetti sono presentati all'ingresso con indirizzi di destinazione in ordine crescente



Notare come sono ordinati gli ingrassi

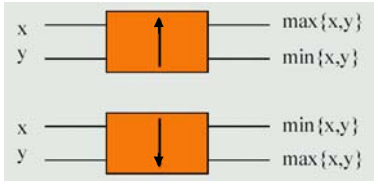
Beispiel für Banyan Network



keine Kollisionen wenn Ziel-Adressen aufsteigend vorliegen

Banyan Network (2)

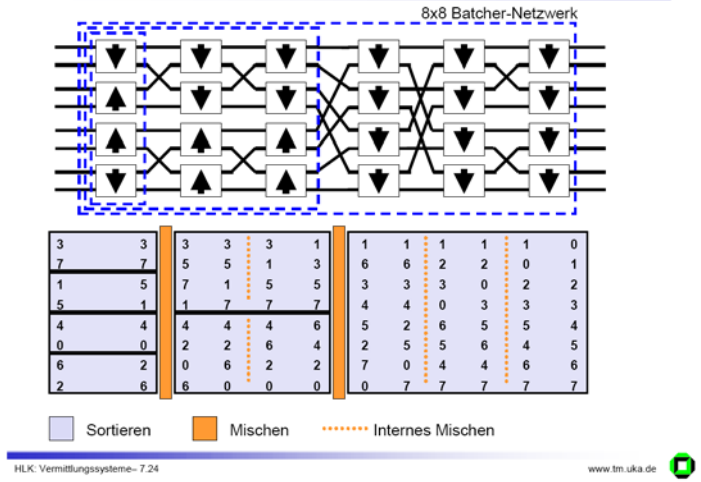
Per ottenere gli insirizzi di uscita in ordine crescente si installa a monte un circuito che modifica l'ordine degli ingressi (*sorter*).
Un *sorter* elementare è un comparatore che seleziona il più grande o il più piccolo di due numeri.



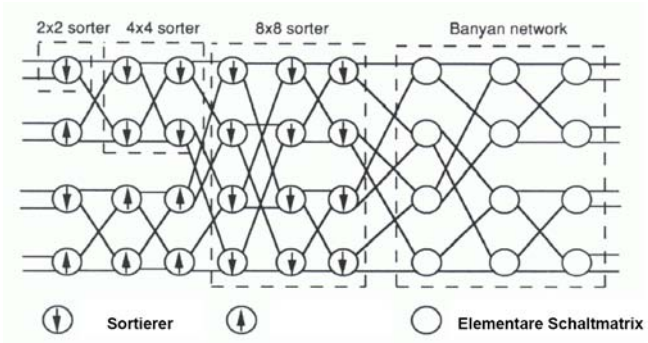
Un *sorter* di **Batcher** è una rete di *sorters* elementari

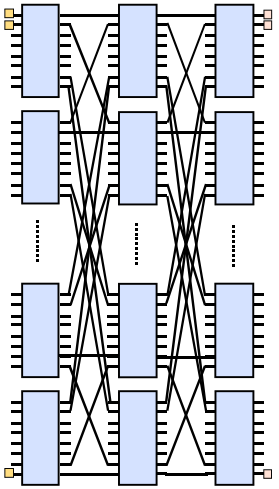
La configurazione di un *sorter* di Batcher in tandem con una rete Banyan è *strict-sense nonblocking* per *unicast*, ma *blocking* per *multicast*

Batcher-Netzwerk (2)



Batcher-Banyan





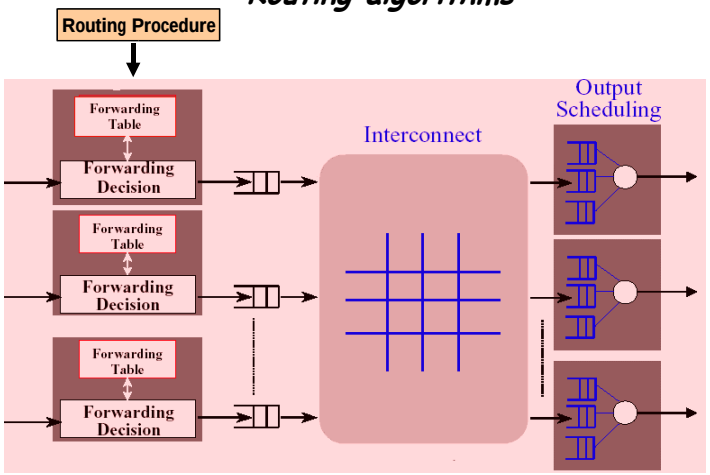
Switch costruito nello stile banyan ma con con elementi *crossbar* n x n (e non 2 x 2)

self-routing dipende dalle proprietà dei *crossbars* componenti

blocking da vedersi: quando gli stadi sono 3 si applicano le regole di uno *switch* di Clos

In figura , 3 stadi uguali di 8 switches, ognuno 8 x 8

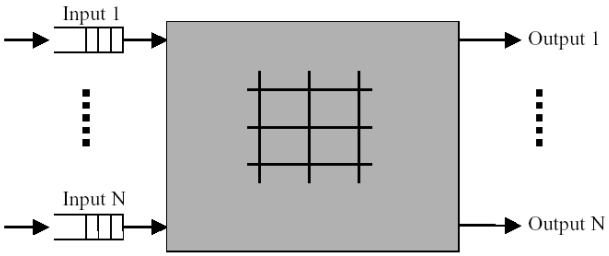
Routing algorithms



Routing Algorithms

Sistema	a commutazione di circuiti (*circuit switching*), es. telefono a commutazione di pacchetti (*packet switching*), es. rete locale
Protocollo	*connection oriented*, es.telnet *connectionless*, es TCP/IP
Percorso (*routing*)	precalcolato *offline*, *look-up-table* precalcolato al momento contenuto nel pacchetto *store-and-forward* ovvero *wormhole*
Controllo	centrale, usa un percorso precalcolato locale, legge l'indirizzo del passo seguente contenuto nel pacchetto

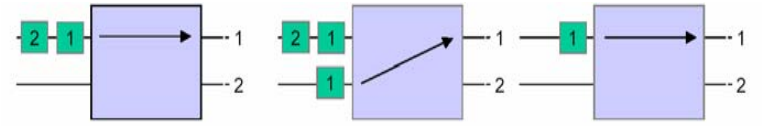
Input Queuing (IQ)



- Una coda per ingresso
- Di solito coda FIFO
- La coda accumula i pacchetti che non possono partire subito
- Appena un pacchetto arriva alla porta di uscita parte immediatamente (non ci sono code in uscita)
- Funziona con *switches* lenti, *speedup* = 1
- *Head-Of-Line (HOL) blocking*

Head of Line (HOL) Blocking

Problema intrinseco alla natura della FIFO in IQ



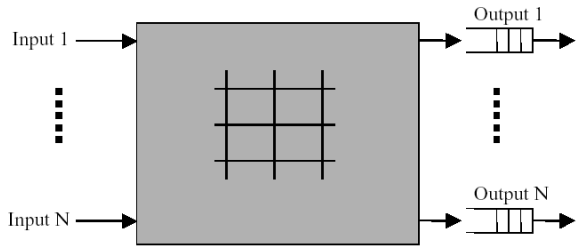
il pacchetto per la porta 2 è bloccato dal pacchetto in transito verso la porta 1

il paccketto per la porta 2 è bloccato dal pacchetto in attesa della porta 1

non ci sono paccketti per la porta 2

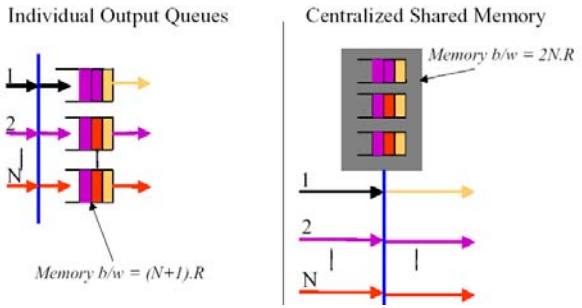
In ciascuno dei tre casi la porta 2 in uscita non è utilizzata.
Con distribuzione aleatoria dei pacchetti in arrivo,
l'utilizzazione dello *switch* è inferiore al 58.6%

Output Queuing (OQ)



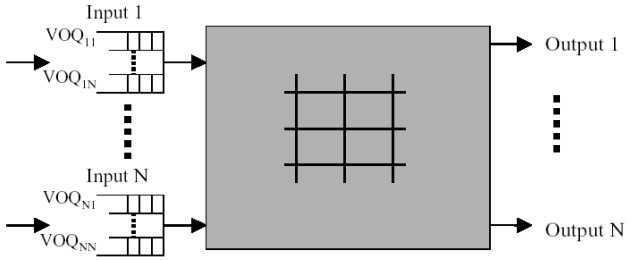
- Una coda per ogni uscita.
- Appena un pacchetto entra nello *switch* passa subito alla coda di uscita.
- La coda accumula i pacchetti che non possono partire immediatamente.
- Massimo *throughput*.
- Minima latenza
- Necessario uno *speedup* di *N* (numero di uscite).
- Non realizzabile per switches veloci con un numero elevato di porte

Output Queuing (OQ)



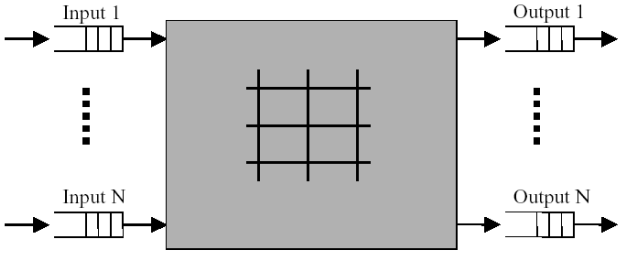
La banda passante delle code o della memoria deve essere sufficiente per ricevere tutti i messaggi diretti alla stessa destinazione;
altrimenti *Block on Output Port*

Virtual Output Queuing (VOQ)



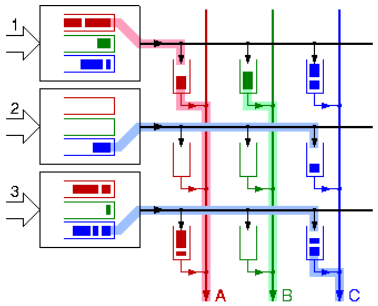
- I pacchetti diretti ad un ingresso si dividono in *N* code, una coda per ogni uscita (*N*x*N* code in totale).
- *Speedup* = 1.
- Evita *HOL blocking*.
- Algoritmi di priorità tra le code sullo stesso ingresso (vedi anche *barrel shifter e multicast*).

Combined Input Output Queuing (CIOQ)



- *Buffers* tanto in ingresso quanto in uscita.
- Lo scopo e di ottenere i vantaggi di IQ e OQ.
- Algoritmi di priorità per le code
- Per uno *speedup* 1<S<N sono necessari *buffers* tanto in ingresso quanto in uscita per usare il sistema in modo efficiente
- Uno *switch* CIOQ con uno *speedup* = 2 si comporta come uno *switch* OQ con code FIFO, per ogni condizione (ragionevole) di traffico.

Combined Input-Crosspoint Queuing (CICQ)



- I pacchetti ad ogni ingresso si dividono in *N* code, come in VOQ.
- *Buffers* a ogni *crosspoint*
- *Distributed routing*
- Permette il passaggio di pacchetti di dimensioni diverse

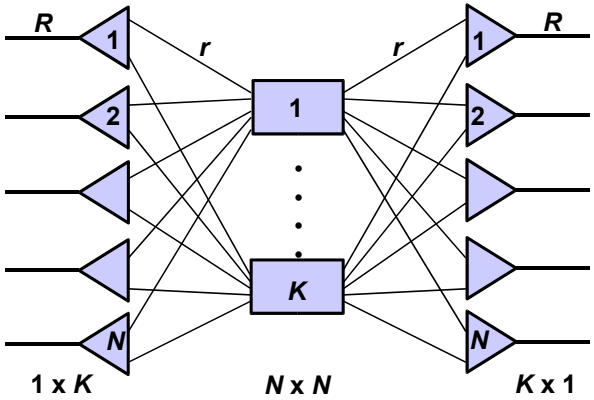
Parallel Packet Switching (PPS)

Costruire uno *switch* più veloce usando *K switches* lenti

Topologia di una rete di Clos

N ingressi:	*N demultiplexers* da 1 a *K*, velocità d'ingresso *R*, d'uscita *r = R/K*,
Stadio intermedio: (*spine*)	*K switches OQ* *N* x *N* ,
N uscite:	*N multiplexers* da *K* a 1, velocità d'ingresso *r*, d'uscita *R*

Parallel Packet Switching (PPS)



Tecnologie di *switching*

Elettronica

Tanto i dati quanto il controllo sono segnali elettrici
Tradizionale in telecomunicazioni e reti di dati

Elettro-ottica

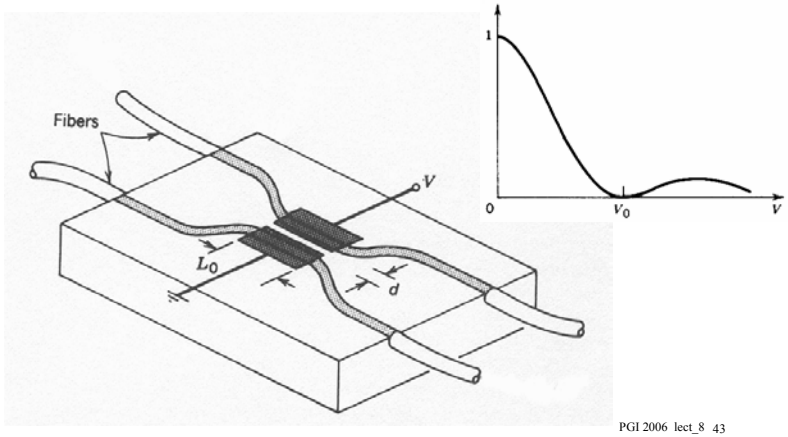
Il controllo è elettronico mentre i dati sono ottici
Directional couplers
MicroElectroMechanical mirror arrayS (**MEMS**)
In rapida espansione tanto in telecomunicazioni quanto in reti di dati

Ottica-ottica

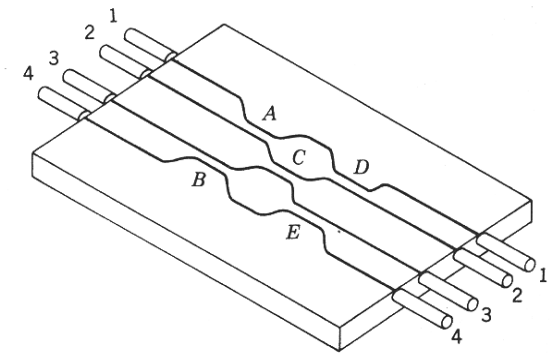
Tanto i dati quanto il controllo sono segnali ottici
Richiede tecniche elaborate, per esempio polarizzazione e ottica non lineare

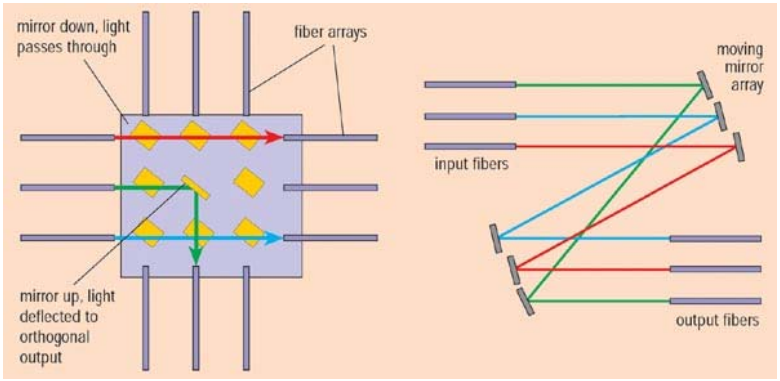
Switch* realizzato con *directional couplers

La tensione applicata ***V*** cambia l'accopiamento tra le due guide di luce

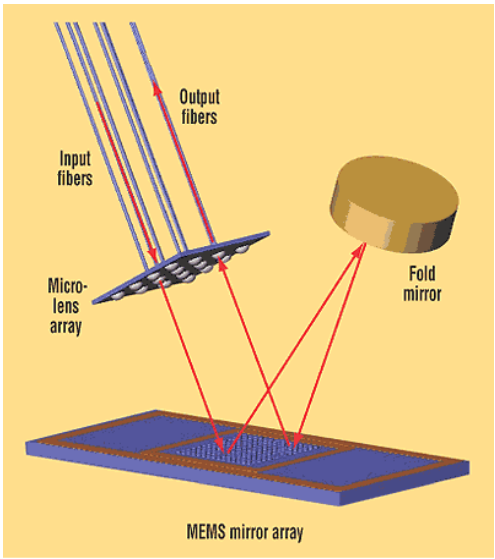


Con 5 *directional couplers A, B, C, D* ed *E* si realizza uno *switch* 4 x 4





2D MEMS **3D MEMS**



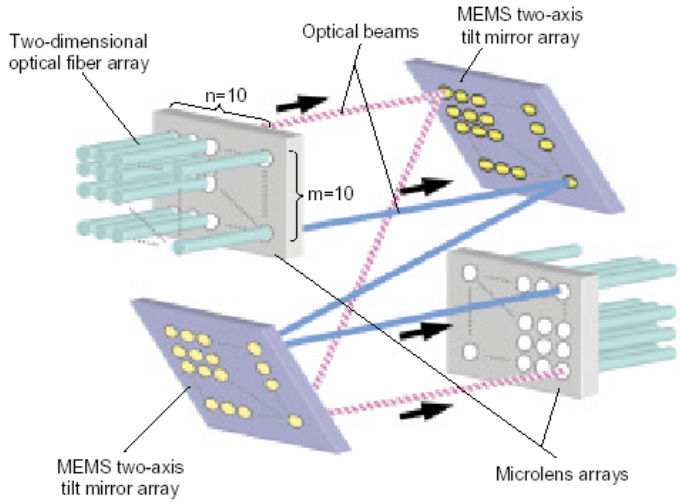


Fig. 1. Basic structure of 3D MEMS optical switch.

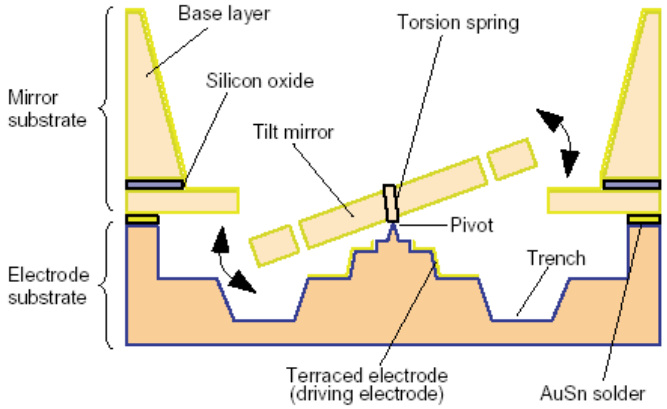


Fig. 2. Cross-sectional schematic of MEMS two-axis tilt mirror.

Caratteristiche di uno *switch*

- Dimensioni: numero di linee in ingresso e in uscita ovvero numero di connessioni, bidirezionali o no
- Tempo di commutazione, da 10 ps a 10 ms (MEMS)
- Banda passante per connessione e attraverso una bisettrice
- Tempo di propagazione
- Flusso, portata, *throughput*
- *Buffers* in ingresso e in uscita?
- Energia di commutazione 10 fJ – 20 fJ
- Dissipazione 1 μW per connessione
- *Insertion loss* *2±1*dB *(MEMS)*
- *Crosstalk*
- Dimensioni 1 cm^2 per 16 x 16 porte
- Riconosce gli indirizzi dei pacchetti? *self-routing* o no
- Tipo di protocollo: connectionless o connection oriented
  elettronico: può essere *self routing* e *connectionless*
  elettro-ottico: *connection oriented*

Referenze

Per una descrizione delle tecnologie elettroniche attuali di *switching*:

Guide to Myrinet-2000 Switches and Switch Networks, Myricom,
http://www.myri.com/myrinet/m3switch/guide/

InfiniBand, Mellonanox Technologies,
http://www.mellanox.com/pdf/whitepapers/scaling10gbsclusters.pdf
http://www.mellanox.com/pdf/whitepapers/IB_vs_Ethernet_Clustering_WP_100.pdf