# Scalable and Practical Nonblocking Switching Networks

S.Q. Zheng
Department of Computer Science
Erik Jonsson School of Engineering and Computer Science
Box 830688, MS EC 31
University of Texas at Dallas
Richardson, TX 75083-0688, USA
Email: sizheng@utdallas.edu

# Acknowledgement

Part of the presented work was jointly carried out with Dr. Ashwin Gumate and Dr. Hong Shen.

# Switching Networks

- **Permutation Switching Networks.**
- **Multiple-multicasting Switching Networks.**

# Applications of Switching Networks

- **Communication subsystem (interconnection network) in a parallel computing system.**
- **Switching fabric in network switches and routers.**

# Permutation Switching Networks

**An $N \times N$ switching network has $N$ inputs and $N$ outputs. It usually comprises a number of** *switching elements* **(SEs) interconnected by a set of links.**

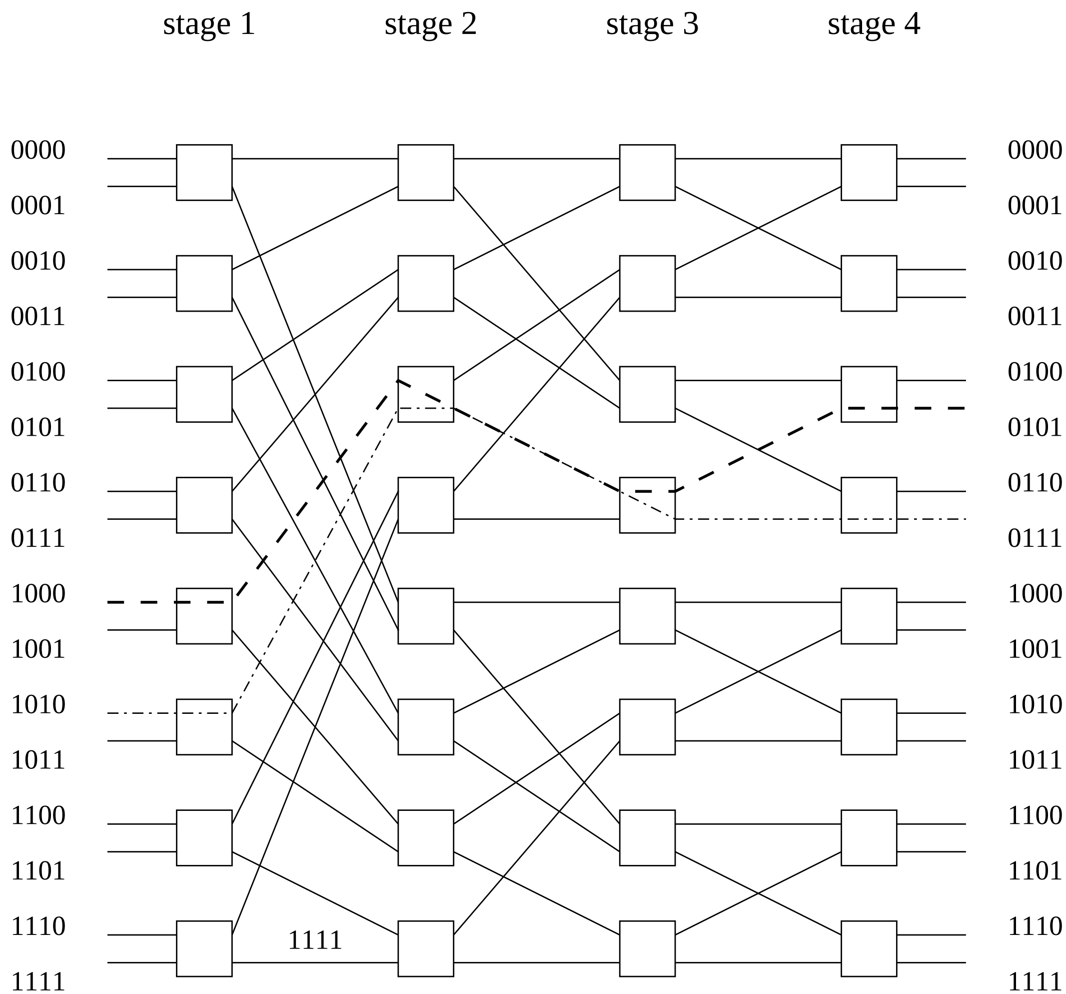
**Without loss of generality, we assume that an SE is of size $2 \times 2$; i.e. it has $2$ inputs and $2$ outputs.**

**The two inputs of an SE intending to be connected with the same output causes** *link conflict***.**

**If an I/O connection path does not have any link conflict with other connection paths, it is called a** *conflict-free* **path.**

**Nonblocking switching networks have been favored in switching systems because they can be used to set up conflict-free I/O connection paths for any (partial) I/O permutation.**

# Example




**A baseline network.**

# Three Basic Types of Nonblocking Switching Networks

- **Strictly Nonblocking (SNB) networks.**
  - **A connection from any idle input to any idle output can always be established regardless of how existing connections were established.**
  - **Useful for both circuit switching and packet switching.**
- **Wide-Sense Nonblocking (WSNB) networks**
  - **A connection from any idle input to any idle output can always be established by following a rule (algorithm).**
  - **Useful for both circuit switching and packet switching.**

- **Rearrangeable Nonblocking (RNB) networks.**
  - **A connection from any idle input to any idle output can be established if rearrangement of existing connections is allowed.**
  - **Useful for packet (cell) switching but not suitable for circuit switching.**

## Lower Bound for the Cost of Nonblocking Switching Networks

**Cost:** $\Omega(N \log N)$ **SEs.**

## Optimal SNB and WSNB Networks

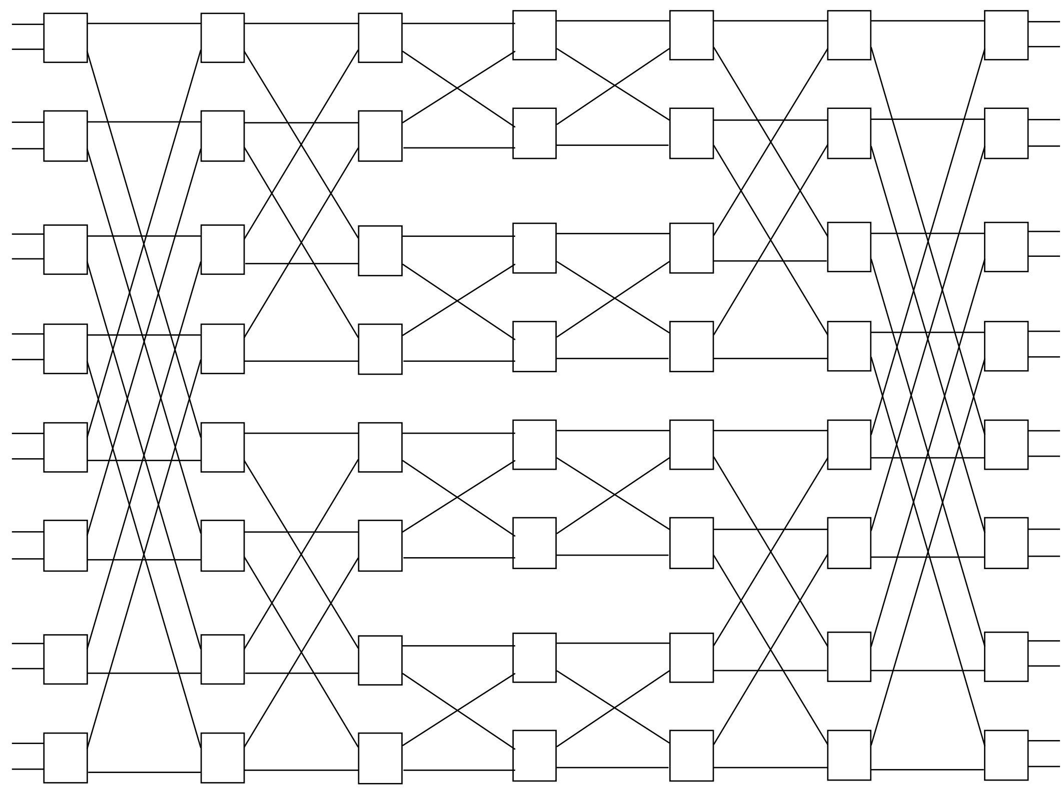
$O(N \log N)$**-cost theoretically possible, but not practical.**

**Suboptimal: Clos network of** $O(N^{1.5})$ **cost, and Cantor network of** $O(N \log^2 N)$ **cost. Routing I/O connections is time-consuming.**

## Optimal RNB Network

$O(N \log N)$ **cost practically achievable. Example: Benes network.**

# Benes Network



# An Outstanding Open Problem

**Find practical $O(N \log N)$-cost SNB and WSNB Networks for circuit-switching applications.**

## Question: Do we really need SNB and WSNB networks?

**Practically, maybe not.**

# Virtual Nonblocking Networks

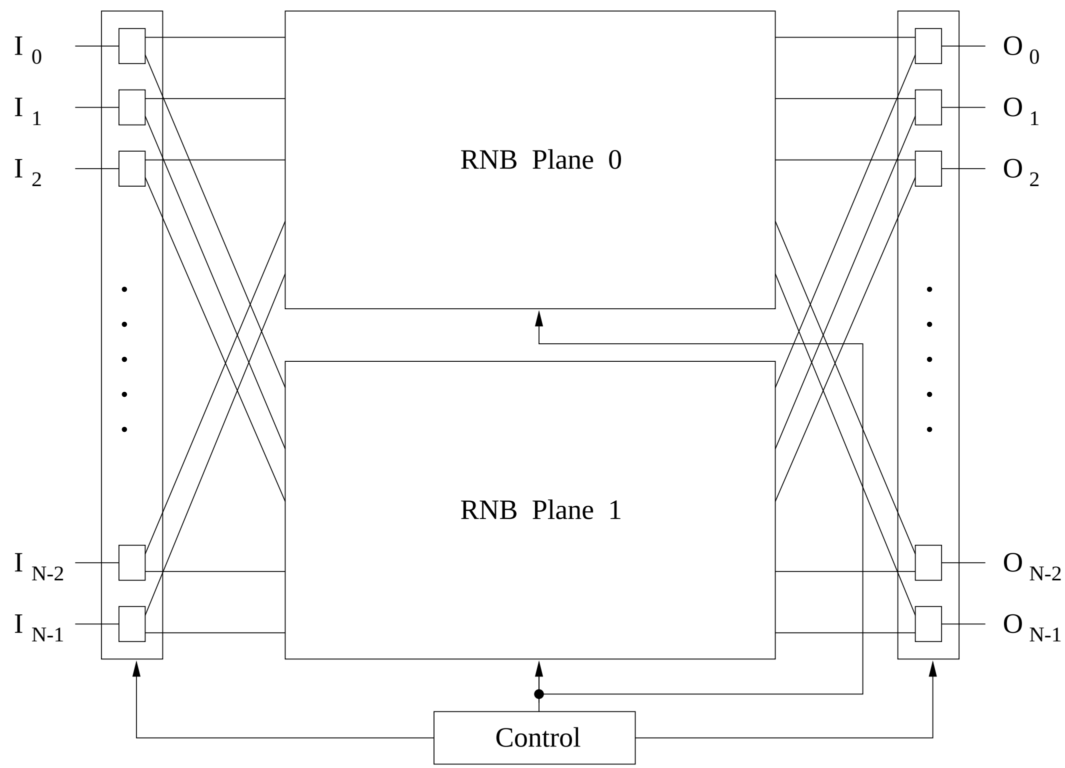
**A switching network is a virtual nonblocking (*VNB*) network if it satisfies the following conditions:**

**(i) it provides more than one path from any input (i.e. source) to any output (i.e. destination) such that different paths that connect the same source/destination pair can be momentarily set up simultaneously but any two paths connecting different input/output pairs are link-disjoint at any time; and**

**(ii) when properly controlled, the network can perfectly emulate an *WSNB* network.**

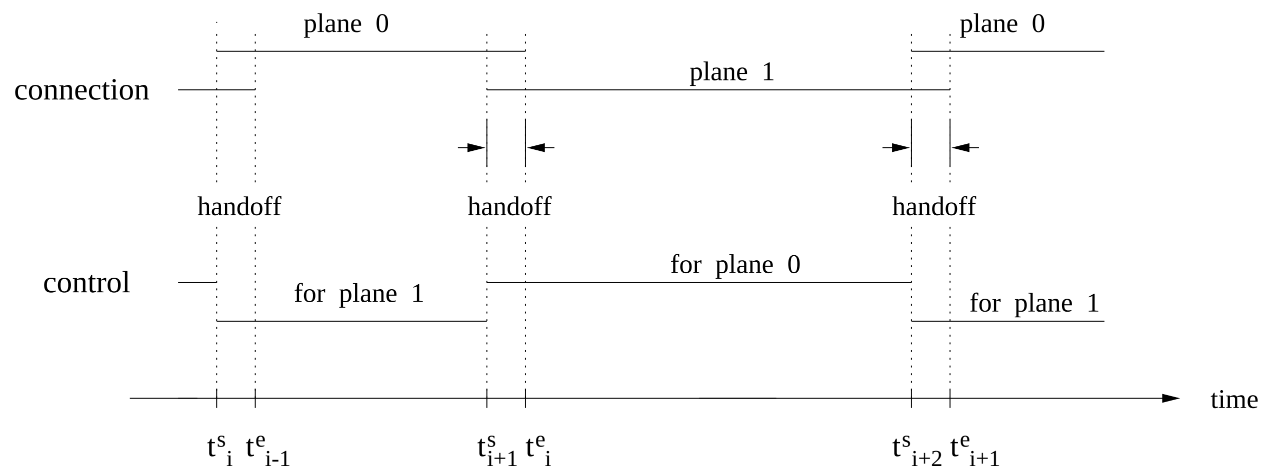
**This definition is broad enough to cover many embodiments of *VNB* network architectures.**

# Ping-Pang VNB Network




**Ping-pang *VNB* network architecture.**

# Handoff Process of Ping-Pang VNB Network




**Timing diagram of the operations of a ping-pang *VNB* network.**

# Handoff Schemes

*Lazy handoff* **scheme: handoff is only needed when finding conflict-free paths fails.**

*Eager handoff* **scheme: handoff is invoked when new connection requests arrive, regardless of whether conflict-free paths for the new connection requests exist or not.**

*VNB* **networks are useful in circuit switched networks. They are used as switching networks within network routers and switches to establish and maintain nondisruptive circuits.**

# Optimal VNB Networks

**The ping-pang VNB network constructed using two copies of Benes network has cost $O(N \log N)$, whitch is optimal.**

# Routing Performance Consideration

**Parallel Benes routing algorithm using an $N$-processor hypercube takes $O(\log^4)$ time, which is too slow.**

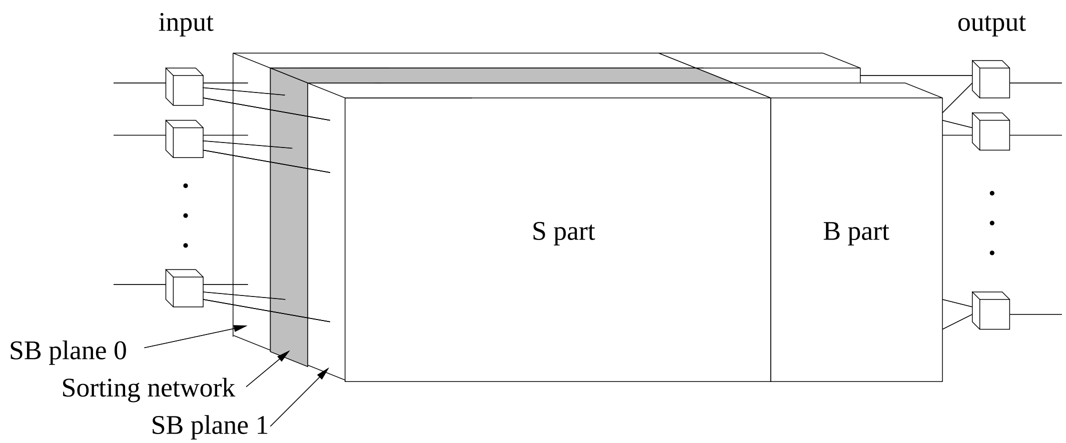
**Self-routing: the ability that any connection in a network can be established by inspecting addresses of its source and destination regardless of other connections.**

**Generalization: This property can be generalized to include the case that the setup of each SE only depends on the $O(\log N)$ information, which may not be the addresses of source and destination, available at the SE during routing.**

**Self-routing VNB networks are desirable.**

# SB VNB Networks

**Each plane is a *concatenation* of a Sorting network and a Banyan (butterfly, Omega, baseline) network.**

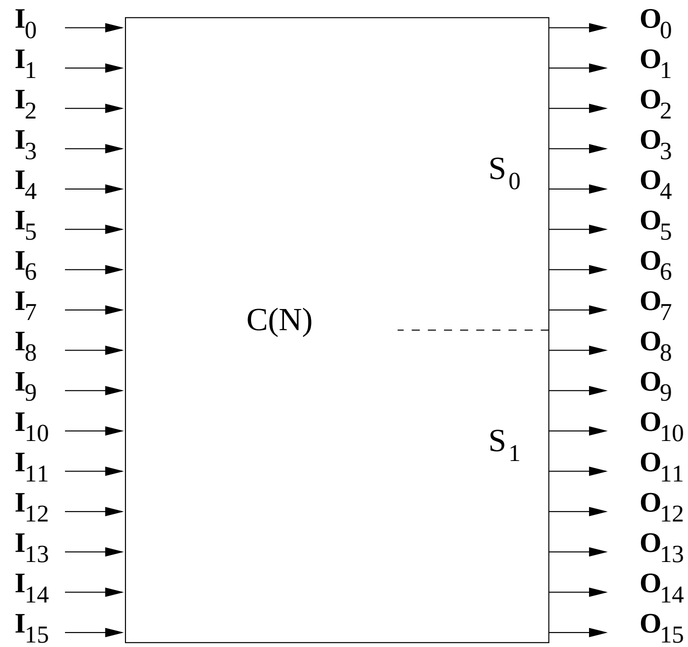



**An *SB* ping-pang *VNB* network.**

**Using AKS sorting network, this VNB network has cost $O(N \log N)$. But this is not practical because the coefficient hidden by big-$O$ is too large.**

# Q-sort: A New Sorting Network

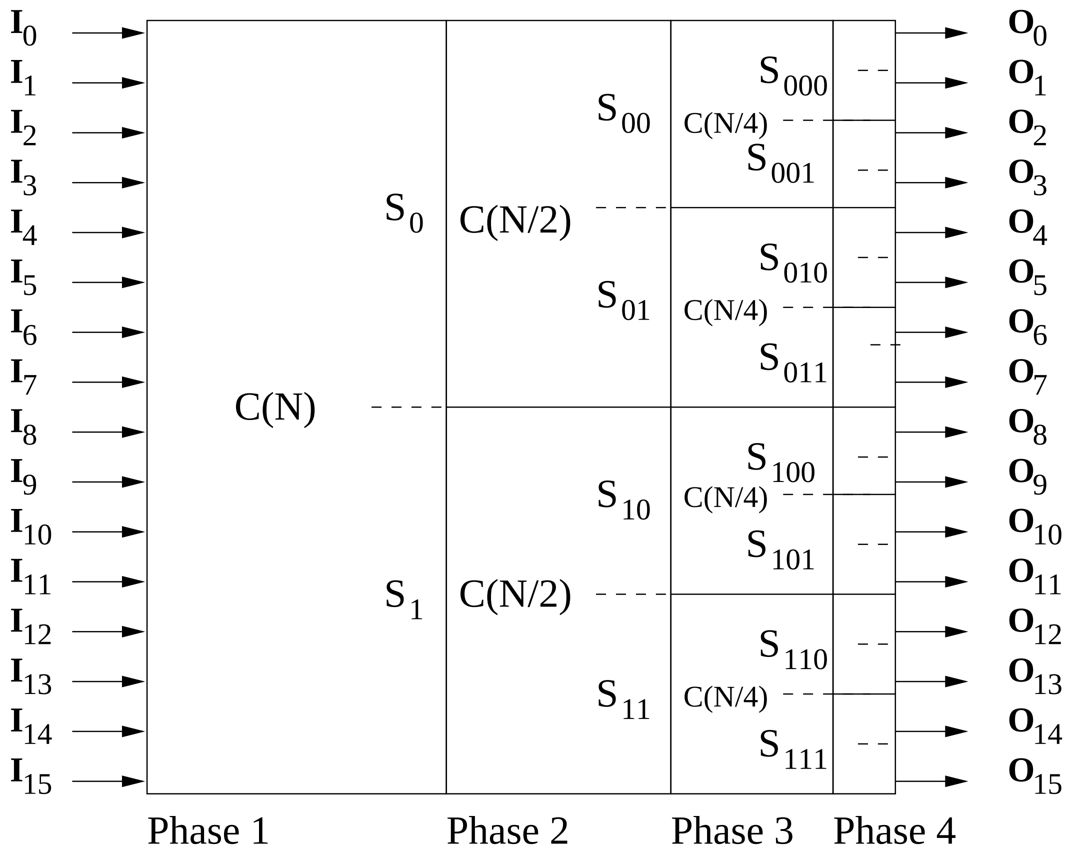
**Building block: $N$-classifier $C(N)$.**




**Classifier $C(16)$.**

**$C(N)$ is an $N \times N$ network that separates a set $S$ of $N$ input numbers into two subsets $S_0$ and $S_1$ such that $|S_0| = |S_1| = \frac{N}{2}$, no number in $S_0$ is larger than any number in $S_1$, and numbers in $S_0$ and $S_1$ appear at the first and second half of the outputs, respectively.**

# $Q$-Sort Network $Q(N)$: Constructed Using Classifiers Recursively

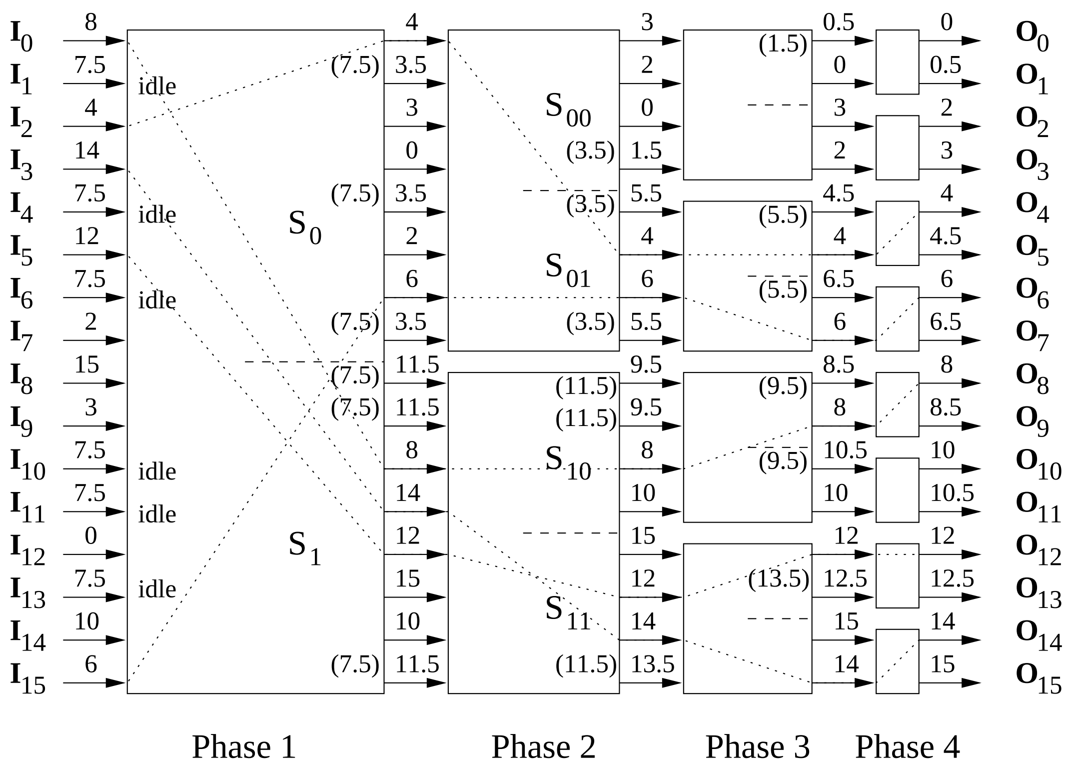



A $Q$-sort network $Q(N)$, $N = 16$. Cost: $O(N \log^2 N)$.

# $Q$-Sort Network $Q(N)$ as an RNB Network

## Routing a partial permutation

$$\pi = (h_0, h_1, \cdots, h_{N-1}) = (8, \Lambda, 4, 14, \Lambda, 12, \Lambda, 2, 15, 3, \Lambda, \Lambda, 0, \Lambda, 10, 6)$$

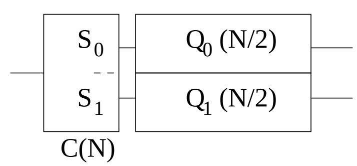



**Self-routing process in $Q(16)$ for partial permutation $\pi$, where $h_i = \Lambda$ indicates that input $i$ is idle (not to be connected to output).**

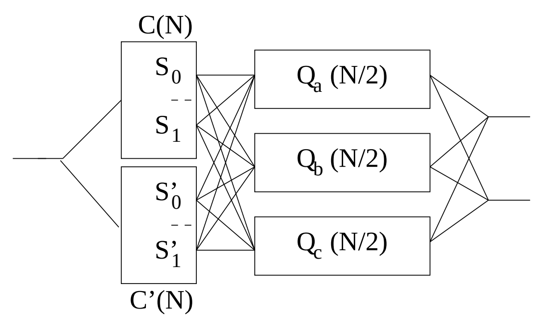
## Self-Routing Ping-Pang VNB Network Based in $Q(N)$

**Using two copies of $Q$-sort network, we can construct a self-routing ping-pang *VNB* network.**

## $Q^*(N)$ Network: Cost Saving VNB Network



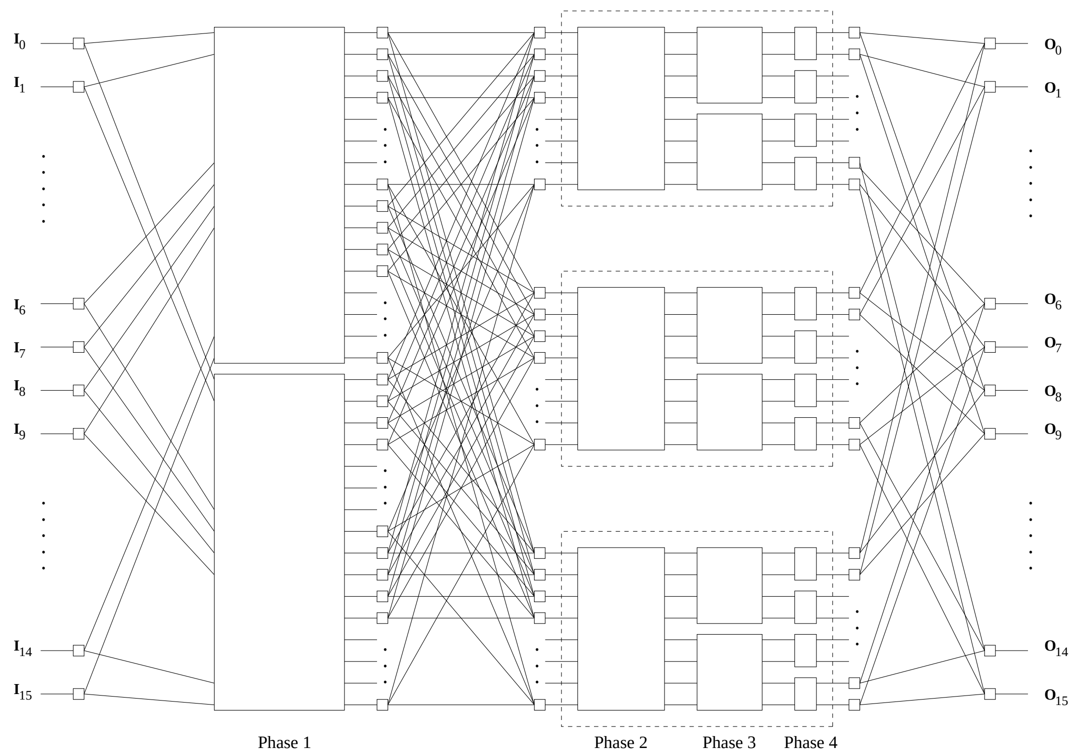

(a)




(b)

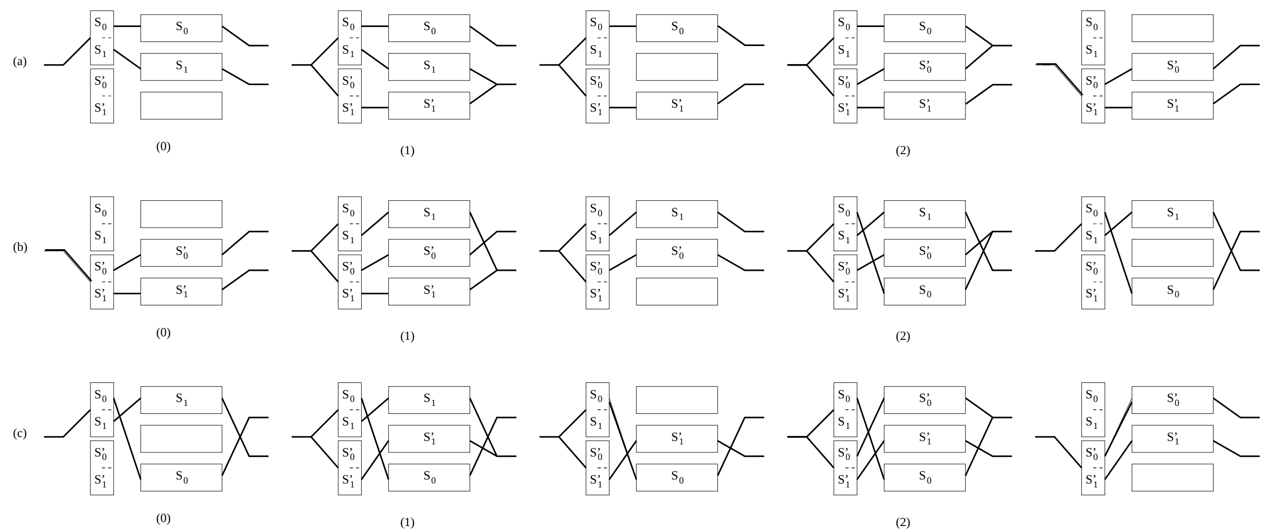
**(a) The structures of a $Q$-sort network $Q(N)$. (b) The cost-saving *VNB* network $Q^*(N)$ based on $Q(N)$.**

# More Details of $Q^*(N)$ Network




The interconnection details of $Q^*(16)$ The part in a dashed block is a $Q(8)$.

# Cyclic Handoff Scheme for $Q^*(N)$ Network



Handoff operations of $Q^*(N)$.

# Summary for Permutation VNB Switching Networks

- **The concept of virtual nonblocking (*VNB*) networks, which are operationally equivalent to SNB or WSNB networks but as simple as *RNB* networks.**

- **The the ping-pang structure, for designing a class of *VNB* networks.**

- **It is simple to design an optimal *VNB* network that contains $O(N \log N)$ SEs using two Benes networks. In contrast, no explicitly constructed *SNB* network of cost in the order of $O(N \log N)$ cost with a small coefficient has been reported in the literature.**

- **Theoretically it is possible to construct self-routing *VNB* networks with $O(N \log N)$ SEs. Practically it is easy to construct self-routing *VNB* networks with $O(N \log^2 N)$ SEs by adopting sorting networks. To the best of our knowledge, no self-routing *SNB* network of $O(N \log^2 N)$ SEs has been reported in the literature.**

- **A new sorting network, called the $Q$-sort network, based on classifiers, and show how to construct self-routing *RNB* networks with $O(N \log^2 N)$ cost.**

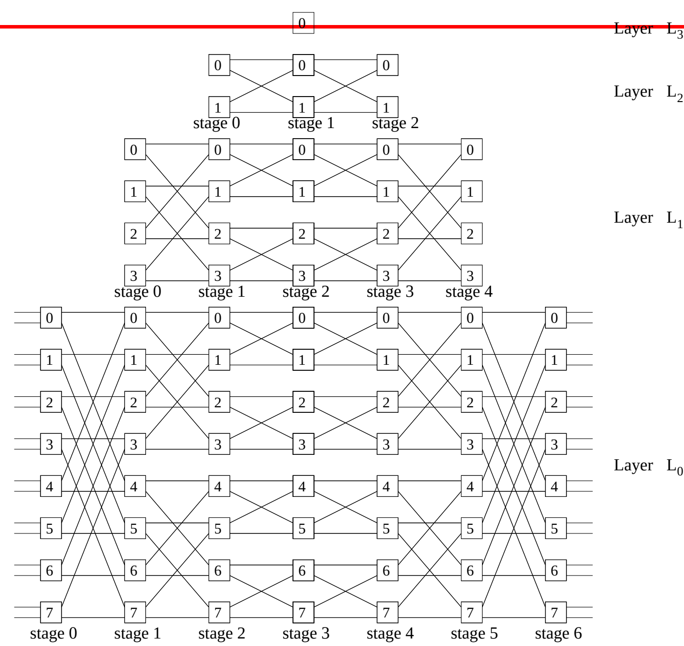
- **A cyclic handoff scheme for constructing *VNB* network $Q^*(N)$ with reduced cost.**

- ***VNB* networks can achieve lower cost and faster routing, which are not easy to achieve simultaneously for *SNB* networks. Thus, for all circuit switching applications, all we need are *VNB* networks.**

# Almost-Nonblocking Permutation Switching Networks

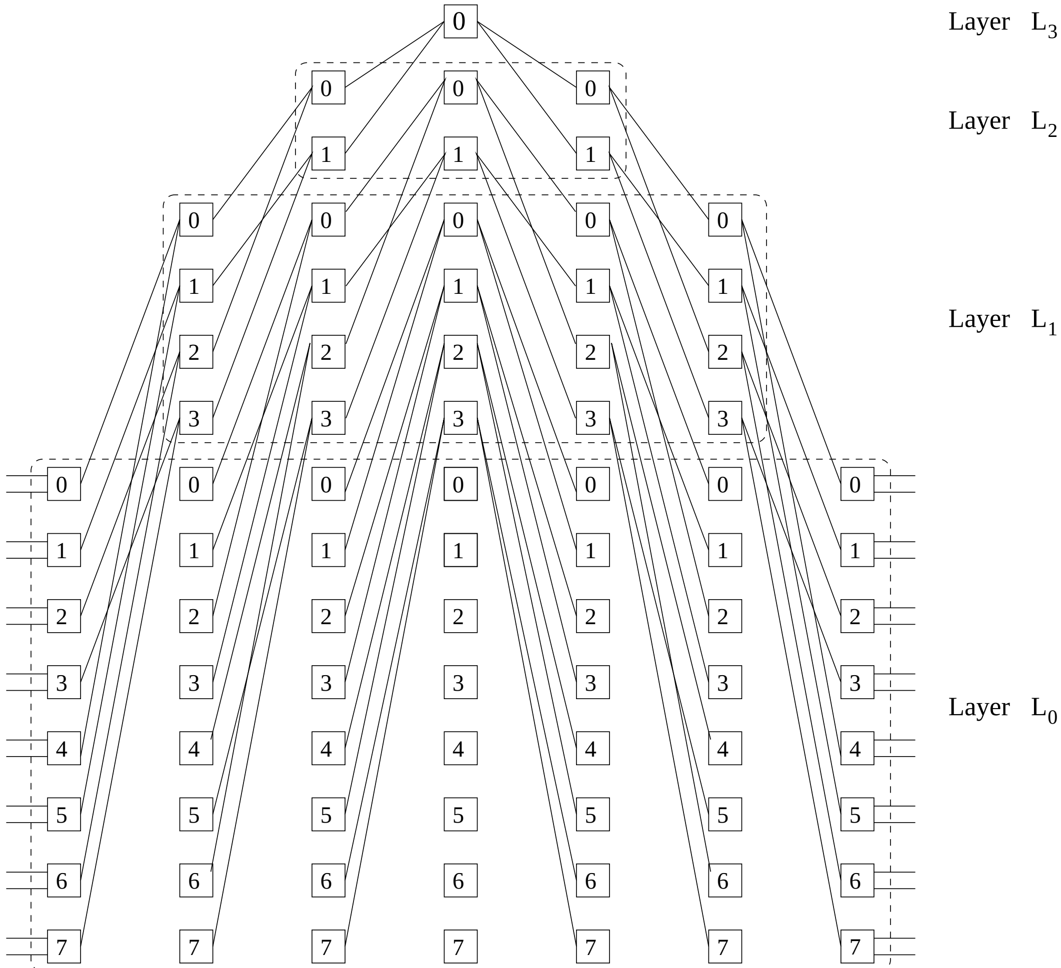
*Pyramid Benes network* $PB(2^n)$ **has** $n-1$ **layers** $L_i$**,** $0 \leq i \leq n-2$**, such that** $L_i$ **is a** $2^{n-i} \times 2^{n-i}$ **Benes network.**

## Intra-layer Connections




**Topology of** $PB(16)$**: Intra-layer connections.**

# Inter-layer Connections




**Topology of $PB(16)$: Inter-layer connections.**

# Nonblokingness of $PB(N)$ Networks

**$PB(N)$ has $O(N \log N)$ cost.**

**By extensive simulations, we have not found any blocking case for $PB(N)$.**

**We cannot claim that $PB(N)$ is strictly nonblocking.**

**However, we can claim that $PB(N)$ is** *almost strictly nonblocking*.

# Multiple-Multicasting Switching Networks

**Broadcasting is a communication pattern which sends a message from a given input port $i$ to all output ports.**

**Multicast is a communication pattern that sends a message from a given input port $i$ to a subset $O_i$ of output ports.**

**A multicast I/O connection pattern is represented by a pair $(i, O_i)$.**

**A** *legal multiple-multicast* **is a communication pattern that is represented by a set of pairs $\{(i_1, O_{i_1}), (i_1, O_{i_1}), \cdots, (i_m, O_{i_m})\}$ such that $i_j \neq i_k$ and $O_{i_j} \cap O_{i_k} = \emptyset$ if $j \neq k$.**

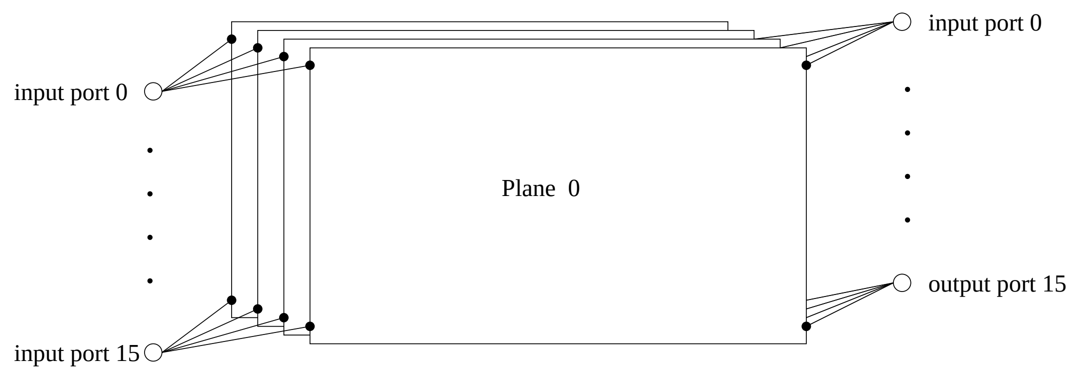
**Permutation (i.e one-to-one) and broadcast (i.e. one-to-all) are special cases of legal multiple-multicast.**

# VNB Networks for Multiple-Multicast

**In order to obtain a VNB network with multiple-multicast capability, we have to design a RNB network with multiple-multicast capability.**

# Multi-log$_2N$ Networks

**An $N \times N$ network of $p$ identical planes, each being a multistage network of $O(\log_2 N)$ stages.**

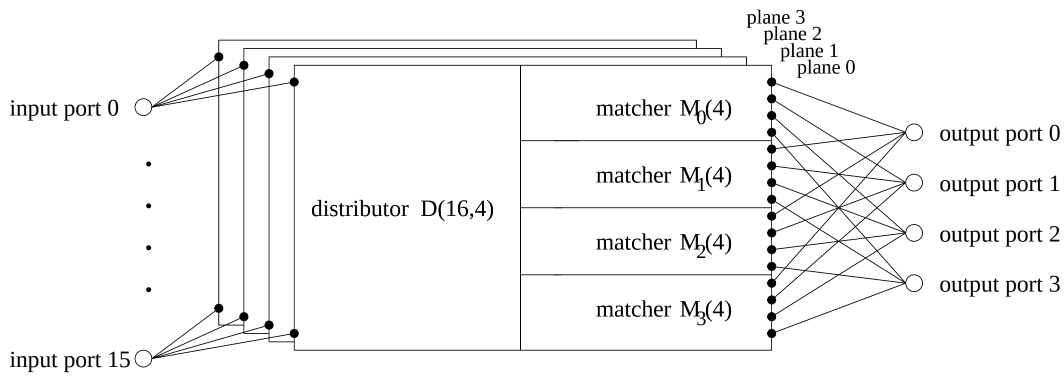



**Multi-$\log_2 N$ networks.**

**If each plane is a Banyan network, then $p \geq 2^{\lfloor \frac{n}{2} \rfloor}$ planes are required for the multi-log$_2N$ network to be an RNB permutation network. Cost: $O(N^{1.5} \log N)$.**

**No known multi-log$_2N$ of cost $O(N^{1.5} \log N)$ can realize conflict-free multiple-multicast.**

# $DM(N)$: A Self-Routing Multi-$\log_2 N$ Network for Multiple-Multicast

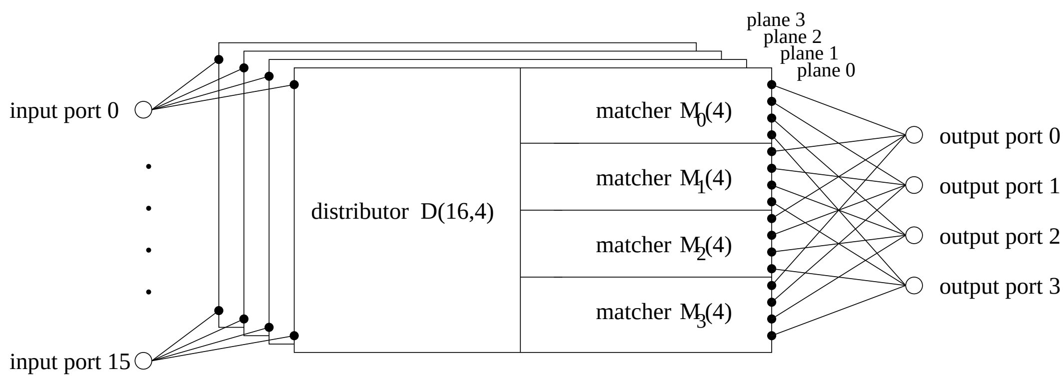



**Structure of $DM(N)$ networks.**

# Structure of $DM(N)$ Network

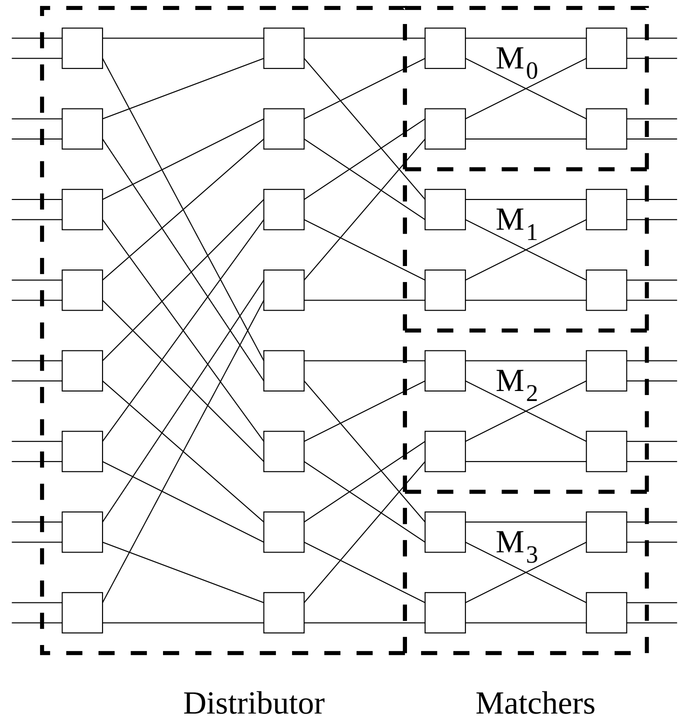
**Assume that $N = 2^n$, where $n$ is even.**

- **$\sqrt{N}$ planes, $P_0, P_1, \cdots, P_{\sqrt{N}-1}$, each being a baseline network.**
- **Input port $i$ is connected to the $i$-th inputs of all planes through a tree $T_i$.**
- **Plane $P_j$ is dedicated to connecting any input port to any output ports $j \cdot \sqrt{N} + k$, $0 \leq k < \sqrt{N}$.**
- **Plane $P_j$ has a distributor, and $\sqrt{N}$ matchers , each having $\sqrt{N}$ outputs, with the $k$th outputs of the matchers connected to output port $j \cdot \sqrt{N} + k$.**




**Structure of $DM(N)$ networks.**

# Structure of a Plane




**A plane of $DM(16)$ is a Baseline network.**

# 3-Phase Routing Algorithm

**An input is *active* if it is to be connected to an output.**

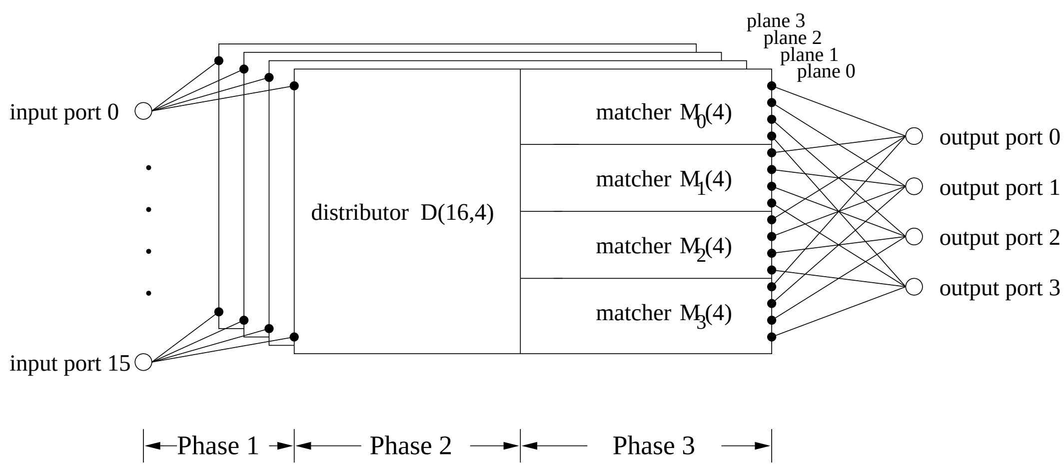
**Multiple-multicast communication pattern:** $\{(i_1, O_{i_1}), (i_1, O_{i_1}), \cdots, (i_m, O_{i_m})$ **such that** $i_j \neq i_k$ **and** $O_{i_j} \cap O_{i_k} = \emptyset$ **if** $j \neq k$**.**

**Let** $O_{i_j,p} = \{q|$ **output** $q$ **belongs to the** $k$**-th plane** $\}$**.** $O_{i_j,p}$ **is the set of output ports to be connected to input port** $i_j$ **in plane** $P_p$**.**

# Phase 1: Route inputs to the inputs of planes.

**For all $(i_j, O_{i_j})$ do the following in parallel:**

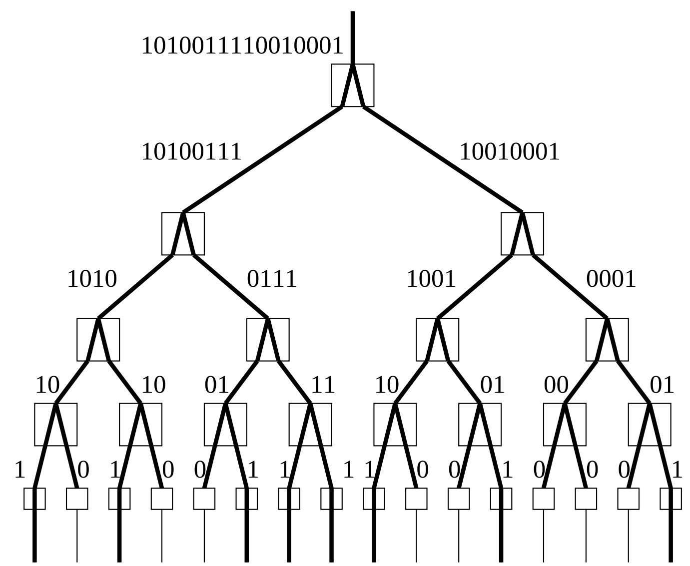
**Connect input port $i_j$ to the $k$-th plane if $O(i_j, k) \neq \emptyset$ using input tree $T_{i_j}$.**




**3-phase routing in $DM(N)$ networks.**

# Implementatin of Phase 1

Each input port $i_j$ computes a bit string for planes it has to be connected. Then, this bit string is used to route through the tree $T_{i,j}$ using broadcast-and-select technique.

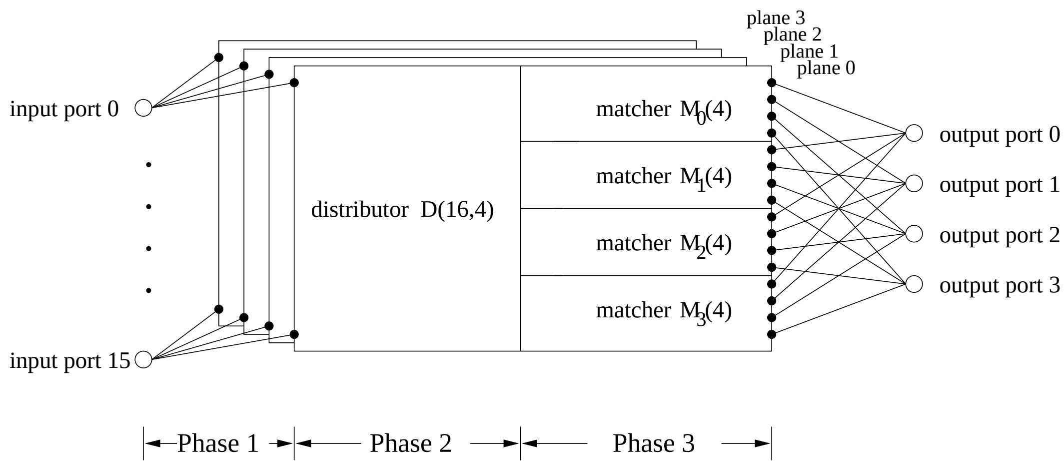



Broadcast-and-select: with a bit-string 101001111001000, the input of the tree is connected to the outputs specified by the bit-string.

# Phase 2: Routing through distributors.

**For all planes do the following in parallel:**

**Connect active inputs of each plane to the inputs of the plane's matchers such that each matcher has at most 2 active inputs.**




**3-phase routing in $DM(N)$ networks.**

# Implementation of Phase 2

**Each plane is equipped with a circuit for generating prefix sums of $N$ Boolean values, which are used for self-routing in the plane. We call this circuit *intraplane routing preprocessing* (IRP) circuit.**

**It is sufficient to consider routing operations in a single plane.**

**If input $i$ is active, set $R_i = 1$; else set $R_i = 0$.**

**The IRP circuit is used to compute binary prefix sums of $R_i$s, where the binary prefix sum $S_i$ is defined as**

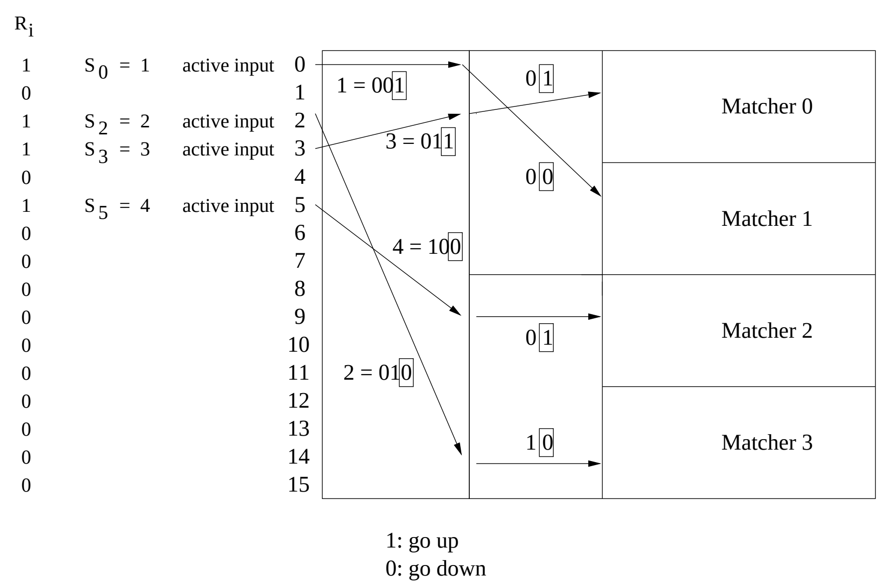
$$S_i = R_0 + R_1 + \cdots + R_i \tag{1}$$

**$S_i$s can be computed in $O(\log N)$ time by an IRP circuit of tree structure.**

**Use the bits of $S_i$ to route active inputs up or down recursively.**

# Example:

**For** $R_0 = 1, R_1 = 0, R_2 = 1, R_3 = 1, R_4 = 0, R_5 = 1, R_6 = R_7 = R_8 = R_9 = R_{10} = R_{11} = R_{12} = R_{13} = R_{14} = R_15 = 0$, **we have** $S_0 = 1, S_1 = 1, S_2 = 2, S_3 = 3, S_4 = 3, S_5 = 4, S_6 = S_7 = S_8 = S_9 = S_{10} = S_{11} = S_{12} = S_{13} = S_{14} = S_{15} = 4$.

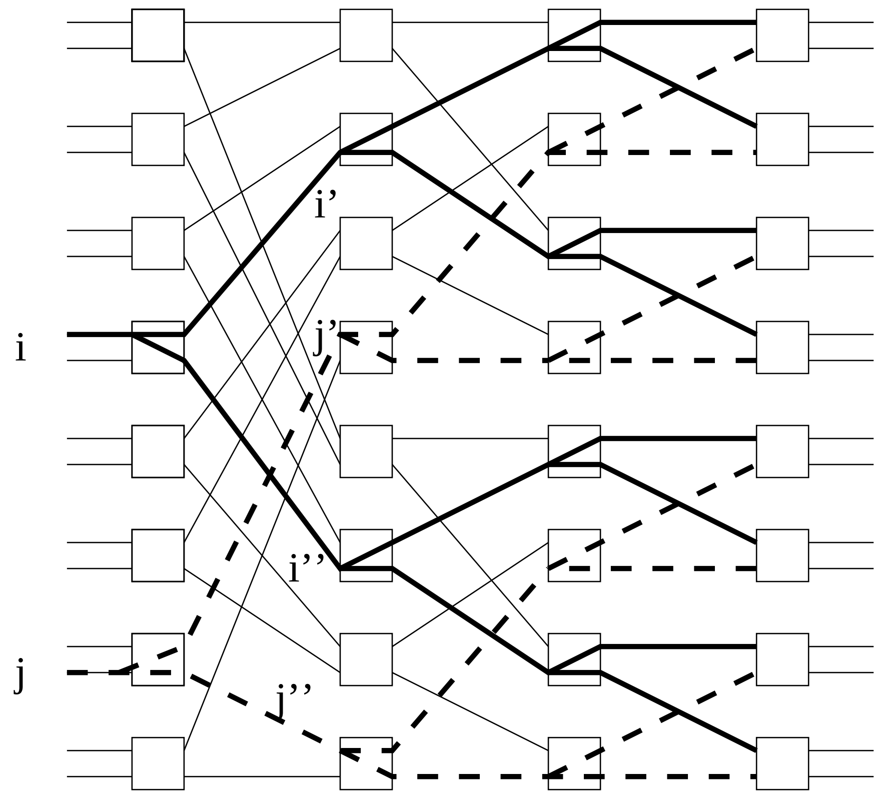



**Self-routing in Phase 2.**

# Active Inputs of Matchers

**Phase 2 guarantees that each matcher has at most 2 active inputs.**

**Phase 2 guarantees the two active inputs have link-disjoint broadcast trees in the matcher, which is a Baseline subnetwork.**

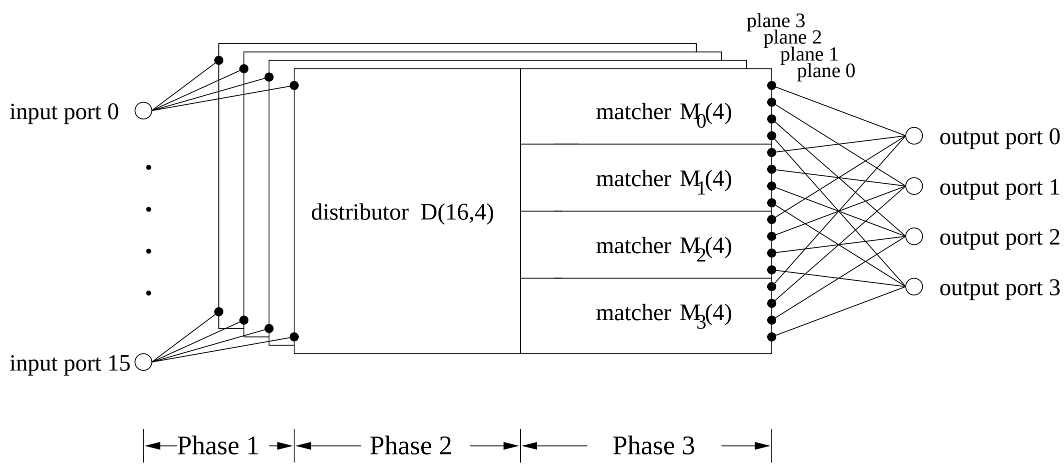



**Two broadcasting trees in $M(16)$.**

# Phase 3: Routing through matchers.

For all matchers in all planes $P_k$, $0 \le k < \sqrt{N}$, do the following in parallel:

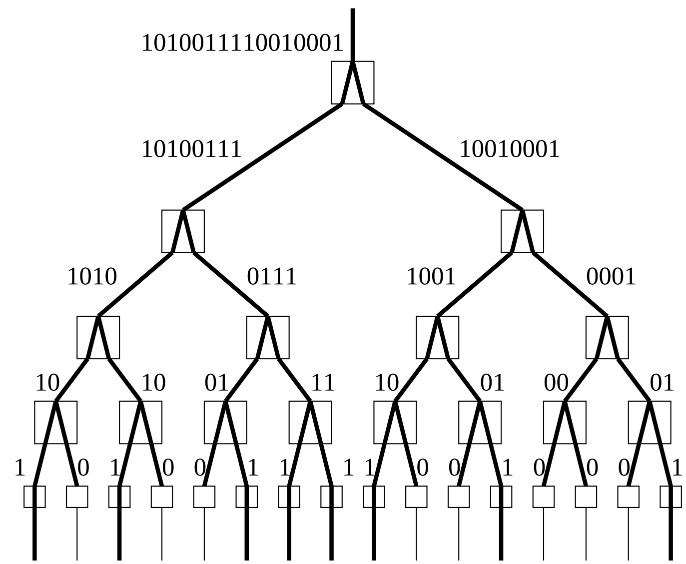
If an active input is originated from input port $i_j$, then connect the active input to all outputs of $O_{i_j,k}$ within the matcher.




3-phase routing in $DM(N)$ networks.

# Implementation of Phase 3

## Using broadcast-and-select technique based on bit strings..




## Broadcast-and-select.

# Summary on Routing $DM(N)$ Network

**Routing in $DM(N)$ is performed in 3 phases.**

**Bit strings for Phase 1 and Phase 3 are computed by input ports.**

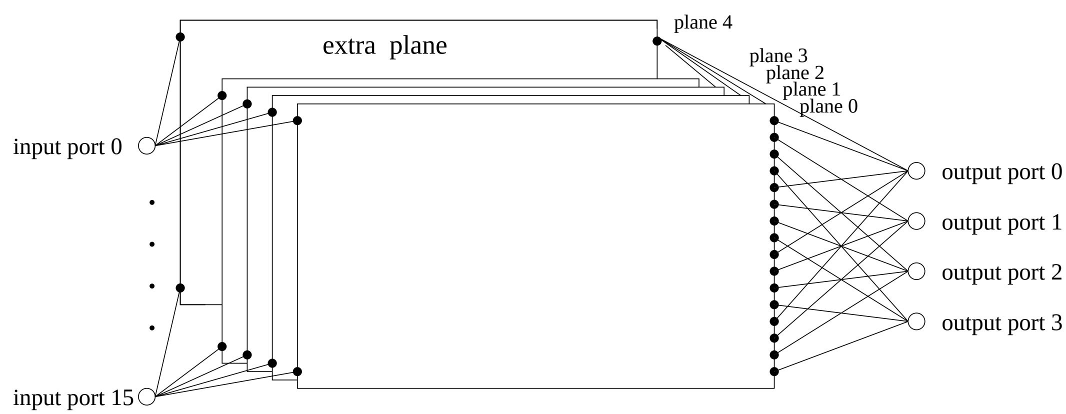
**Binary profix sums for Phase 2 are performed by a circuit at the input of each plane.**

**Then, self-routing is conducted in all phases.**

# Making $DM(N)$ Virtually Nonblocking

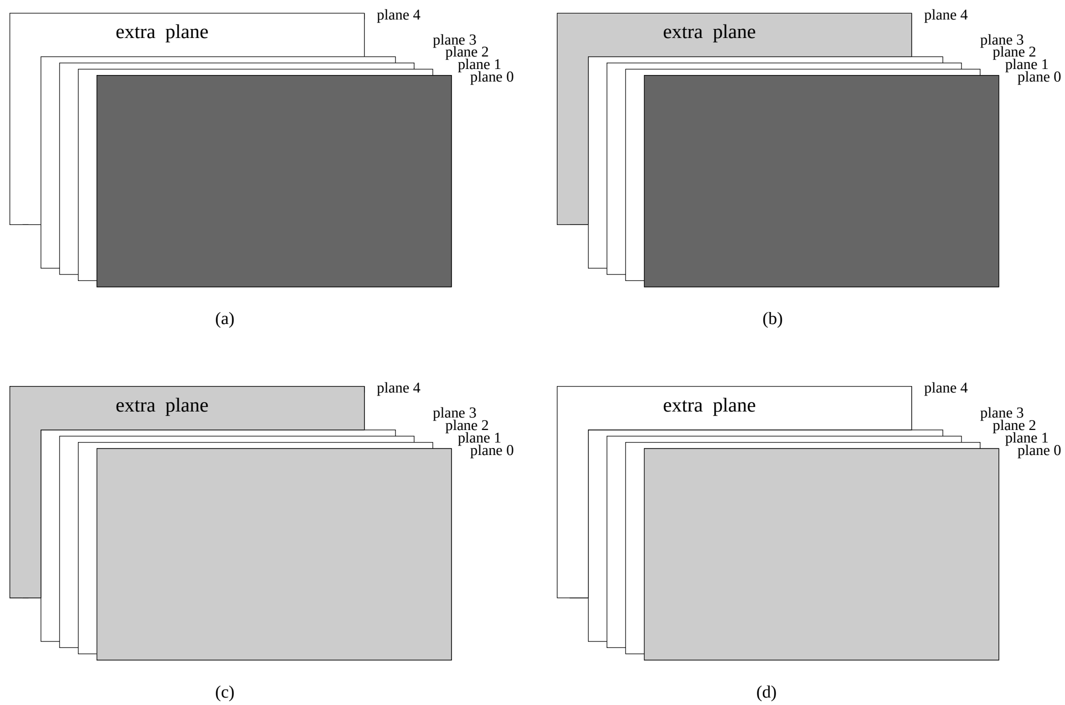
**Using ping-pang scheme, we need two copies of $DM(N)$.**

**But we can reduce the cost by only introduce 1 extra plane.**




**Extended $DM(16)$ network.**

# Hand-off in Virtually Nonblocking Multiple-Multicast $DM(N)$ Network




**Using 1 extra plane to do hand-off.**

# Concluding Remarks

**The high degree of connection capability in** *SNB* **and** *WSNB* **networks is achieved at much higher cost, which implies their lower scalability.**

**We introduce the concept of virtual nonblockingness. We show that a virtual nonblocking network functions like a strictly or wide-sense nonblocking network, but it is constructed with the cost of a rearrangeable nonblocking network.**

**While finding low cost** *SNB* **networks remains to be a challenging theoretical and scientific pursuit, VNB networks can satisfy all the requirements of an** *SNB* **or** *WSNB* **in practice.**

**Our results indicate that for large-scale circuit switching applications, we only need to build virtual nonblocking networks.**