JOURNAL OF SYSTEMS ARCHITECTURE

Journal of Systems Architecture 46 (2000) 529–542

O(n) routing in rearrangeable networks

Nabanita Das, Krishnendu Mukhopadhyaya, Jayasree Dattagupta *

HACM Unit, Indian Statistical Institute, 203, B.T. Road, Calcutta 700 035, India

Received 21 November 1997; received in revised form 21 March 1999; accepted 31 May 1999

Abstract

In $(2n-1)$-stage rearrangeable networks, the routing time for any arbitrary permutation is $\Omega(n^2)$ compared to its propagation delay O(n) only. Here, we attempt to identify the sets of permutations, which are routable in O(n) time in these networks. We define four classes of self-routable permutations for Benes network. An O(n) algorithm is presented here, that identifies if any permutation P belongs to one of the proposed self-routable classes, and if yes, it also generates the necessary control vectors for routing P. Therefore, the identification, as well as the switch setting, both problems are resolved in O(n) time by this algorithm. It covers all the permutations that are self-routable by anyone of the proposed techniques. Some interesting relationships are also explored among these four classes of permutations, by applying the concept of '*group-transformations*' [N. Das, B.B. Bhattacharya, J. Dattagupta, Hierarchical classification of permutation classes in multistage interconnection networks, IEEE Trans. Comput. (1993) 665–677] on these permutations. The concepts developed here for Benes network, can easily be extended to a class of $(2n-1)$-stage networks, which are topologically equivalent to Benes network. As a result, the set of permutations routable in a $(2n-1)$-stage rearrangeable network, in a time comparable to its propagation delay has been extended to a large extent.

Keywords: Rearrangeable networks; Benes network; Self-routing; Full-access unique-path multistage interconnection networks (MIN)

1. Introduction

An $N \times N$ connection network, often used for ATM switches or multiprocessor systems, is a switching device that connects its N input ports to its N output ports dynamically. A self-routing network is one in which a path is set up from an input to an output, considering only the local information available at each switch. Now, with the advent of Gigabit and Terabit optical networks, the problem of designing a self-routing switching network, with the capability of realizing any of the $N!$ permutations of its inputs onto its outputs, is becoming significantly important [1].

An $N \times N$ Benes network is a well-known rearrangeable multistage interconnection network (MIN), with $(2n-1)$ stages ($n = \log_2 N$), that can connect its N inputs to its N outputs in all possible ways [2–4]. The recursive structure of an $N \times N$ Benes network $B(n)$ is shown in Fig. 1. In [5], Yeh and Feng have introduced a class of rearrangeable MIN's, topologically equivalent to Benes network. Baseline–baseline, omega–omega^{-1}, baseline–omega^{-1}, banyan^{-1}–banyan are some examples of this class of rearrangeable networks. The idea of topological equivalence of $(2n-1)$-stage MIN's have been generalized further in [6].

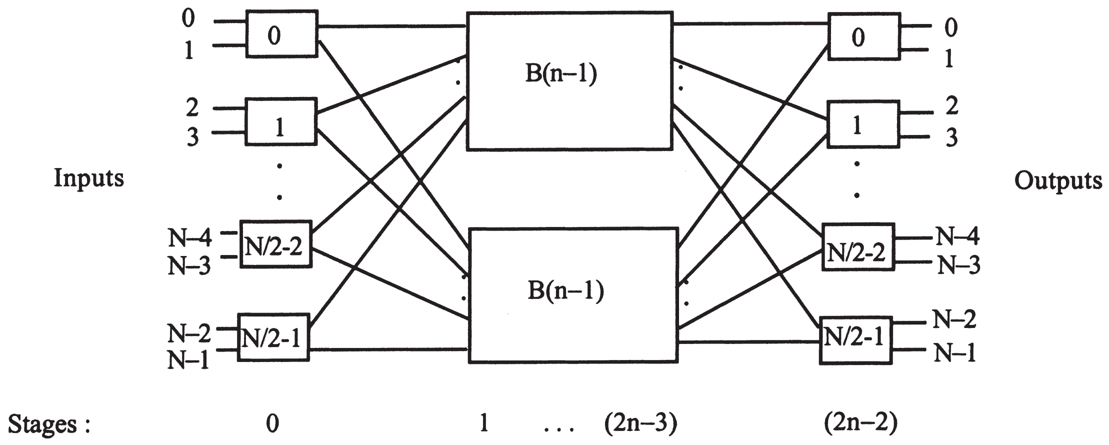

Fig. 1. An $N \times N$ Benes network $B(n)$, $n = \log_2 N$.

Now the best known routing algorithm for an arbitrary permutation in a $(2n-1)$-stage MIN, is found to be of time-complexity O(Nn) on a uniprocessor system [2–4], compared to its propagation delay O(n) only. Even with parallel set-up algorithms, the routing time is reported to be $\Omega(n^2)$ [7]. However, in [8], a new routing algorithm has been reported for $2n$-stage rearrangeable MIN's with O(N) steps. Therefore, in all these cases, the time needed to realize an arbitrary permutation in rearrangeable networks is dominated by the set-up time.

However, in a Benes network, many useful permutations, often required in parallel processing environments are found to be *self-routable* [9–12]. Lenfant proposed efficient set-up algorithms for some *frequently used bijections* (FUB) [9], namely the FUB family. Nassimi and Sahni [10] proposed a simpler algorithm for routing the F class of permutations that includes the *bit-permute complement* (BPC) and *inverse omega* (IΩ) classes of permutations. Boppana and Raghavendra [11] developed another self-routing technique for *linear-complement* (LC) class and IΩ class of permutations.

In all these earlier works, the authors started from some known classes of permutations and developed suitable self-routing strategies for each. In effect, we just know about some classes of permutations, self-routable by some known techniques. Now, in general, the required communications are not restricted to any particular class of permutations. Therefore, given any arbitrary permutation P, before we can apply these self-routing algorithms, we must have to identify that P belongs to a particular class of self-routable permutations. The identification step may increase the effective routing complexity. Moreover, the applicability of the different self-routing techniques developed so far is yet to be fully explored. In fact, there exists many more permutations that are routable by the existing self-routing algorithms; but their characterizations are not yet complete.

In this paper, we investigate the problem in a more realistic way. For Benes network, we classify the self-routable (SR) permutations into four categories, namely:

(i) *Top-Control Routable set of permutations* (TCR),

(ii) *Bottom-Control Routable set of permutations* (BCR),

(iii) *Least-Control Routable set of permutations* (LCR)

(iv) *Highest-Control Routable set of permutations* (HCR).

We will show that each of these classes contain at least $2^n(2^{n(N/2)-n} + n! - 1)$ permutations. Each of the above classes actually contains many more permutations. In fact, it is a lower bound to the size of the intersection of all the classes (i.e., TCR, BCR, LCR and HCR) considered here. We also develop an algorithm that will detect whether any $N \times N$ permutation P belongs to any of the four classes, and if yes, it also generates the appropriate control vectors for routing P. This algorithm can be implemented on a multi-processor system with a time complexity O(n). Therefore, this algorithm enables us to route all the permutations in the union of TCR, BCR, LCR and HCR in O(n) time. By this algorithm a much larger class of permutations will be identified to be self-routable in Benes network than it has been reported earlier. The exact number of permutations covered by each of these classes is found for $N = 4$ and 8. These experimental results help us to have an idea about the total number of permutations self-routable in Benes network by any of these self-routing strategies. In [13], for n-stage full-access unique-path MIN's, a new equivalence relation on permutations has been introduced, namely the *group-transformations*, that explores a new technique for optimal routing of permutations in these networks. Here, we apply the idea of group-transformations to ($2n-1$)-stage networks. It helps us to find a one-to-one correspondence between the permutations in the set TCR (LCR) with those in BCR (HCR) for Benes network. It also enables us to cover a much larger number of permutations routable in O(n) time in Benes network.

The ideas presented here, for routing permutations in Benes network, may be extended to any ($2n-1$)-stage MIN, topologically equivalent to Benes network [6]. Hence it enables us to identify and route a larger set of permutations, in a time comparable to its propagation delay, on a broader class of ($2n-1$)-stage networks.

2. Classification of self-routable permutations

In self-routing algorithms, the setting of a switch is done locally using the destination tags of its two inputs. For any ($2n-1$)-stage rearrangeable MIN, which is a concatenation of two full-access unique-path MIN's, with the central stage being common (as it is in the Benes network), the routing through the last n-stages, must follow the normal destination tag routing scheme. We are to determine the routing strategy for the first ($n-1$) stages only, so that ultimately it becomes conflict-free through the rest of the network. We shall concentrate on algorithms in which one of the two destination tags is selected using a global property (say, that of the top input or the lesser of the two destination values etc.). This input will be called the *R-input*. One particular bit of the R-input is chosen, depending on the stage in which the switch lies and that bit is used for setting the switch. This particular bit will be referred to as the *R-bit*. We will assume that a switch at stage i is denoted by $S_{i,j}$, where $0 \leqslant i \leqslant 2n-2$ and $0 \leqslant j < N/2$ and the destination tags attached to the two inputs of a particular switch $S_{i,j}$ are identified as $T_{i,j}$ (the top one) and $B_{i,j}$ (the bottom one), respectively. For an 8×8 Benes network, the labelling of the switches are shown in Fig. 2.

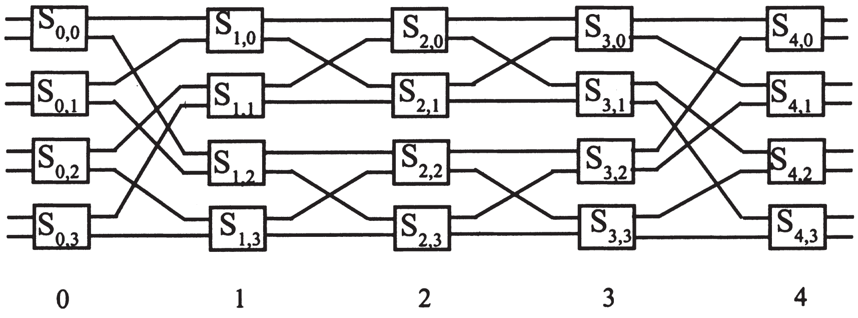

Fig. 2. An 8×8 Benes network and the labeling of the switches.

For the classes of self-routable permutations in Benes network, discussed here, whatever be the R-input at any stage i, $0 \leqslant i \leqslant n-2$, the R-bit is the bit x_i of the destination tag $x_{n-1}\dots,x_i\dots x_1x_0$ of the R-input.

Definition 1. In an $N\times N$ Benes network, we consider four classes of self-routable permutations:
(i) Top Control Routable (TCR), where the R-input is $T_{i,j}$,
(ii) Bottom Control Routable (BCR), where the R-input is $B_{i,j}$,
(iii) Least Control Routable (LCR), where the R-input is $\min\{T_{i,j}, B_{i,j}\}$ and
(iv) Highest Control Routable (HCR), where the R-input is $\max\{T_{i,j}, B_{i,j}\}$.

Fig. 3(a)–(d) show examples of these routing schemes for different permutations on a Benes network. Here, we will represent an $N\times N$ permutation by the sequence of outputs corresponding to the sequence of inputs $(0,\ 1,\ 2,\dots,\ N-1)$.

Remark 1. *For each of the permutations P shown in Fig. 3(a)–(d), it can be verified that no self-routing strategy, other than that chosen, from the above four types, can successfully route P. In other words, none of the classes TCR, BCR, LCR or HCR is contained in any of the other ones.*

Definition 2. A permutation will be called *Top/Bottom/Least/Highest/Control/Routable* if it can be routed on an $N\times N$ Benes network using the Top/Bottom/Least/Highest Control Routing technique.

Among the four classes of permutations mentioned above, the classes TCR and LCR have already been introduced earlier [10,11]. Nassimi and Sahni [10] have shown that a rich class of permutations, namely the F class, that includes the BPC class, IΩ class and also the five classes of permutations considered by Lenfant [9], is top control routable, i.e., TCR$\supseteq$F. Boppana and Raghavendra showed that the LC and IΩ classes of permutations are least control routable [11]. However, the four simplest self-routing strategies proposed above, extend the set of self-routable permutations beyond these known classes. Here follow some interesting properties of the classes which finally lead us to develop the general algorithm for routing a given permutation P, if P belongs to any of the self-routable classes, mentioned above.

Definition 3. In an $N\times N$ Benes network, the set of *free-choice self-routable* (FSR) permutations is defined as the intersection of TCR, BCR, LCR and HCR.

Example 1. The permutation $P=(0\ 1\ 4\ 5\ 3\ 2\ 6\ 7)$ is routable in the 8×8 Benes network by any one of the proposed self-routing techniques, namely the top-control, bottom-control, least-control and highest-control. Therefore, $P\in$ FSR.

Definition 4. In an $N\times N$ Benes network, the set of *R-invariant self-routable* (RSR) permutations is defined as the subset of FSR, such that for any $P\in$ RSR, the switch settings are independent of the choice of R-input. In other words, the R-bits of the two inputs of each switch are complement to each other for any permutation $P\in$ RSR.

Example 2. The permutation $P=(2\ 5\ 7\ 4\ 6\ 1\ 0\ 3)$ is R-invariant self-routable on a Benes network, i.e., $P\in$ RSR.

Remark 2. *In an $N\times N$ Benes network, RSR is actually the IΩ set of permutations. Hence* $|\mathrm{RSR}| = 2^{Nn/2}$.

Lemma 1. *In Benes network, BPC$\subseteq$FSR.*

Proof. It has already been proved that BPC$\subseteq$TCR [10]. Now let us prove that BPC$\subseteq$BCR. We shall prove the result by induction on n. The result can easily be verified for $n=1$. Let the result be true for $(n-1)$.

Let us consider a permutation $P\in$ BPC, described by the bit-permute complement rule

$$P:(x_{n-1},\dots,x_0)\rightarrow(y_{n-1},\dots,y_0),$$

where $(y_{n-1},\dots,y_0)$ is a permutation of $(x_{n-1},\dots,x_0)$, with complementation of some bits. Since $(y_{n-1},\dots,y_0)$ without complementations is a

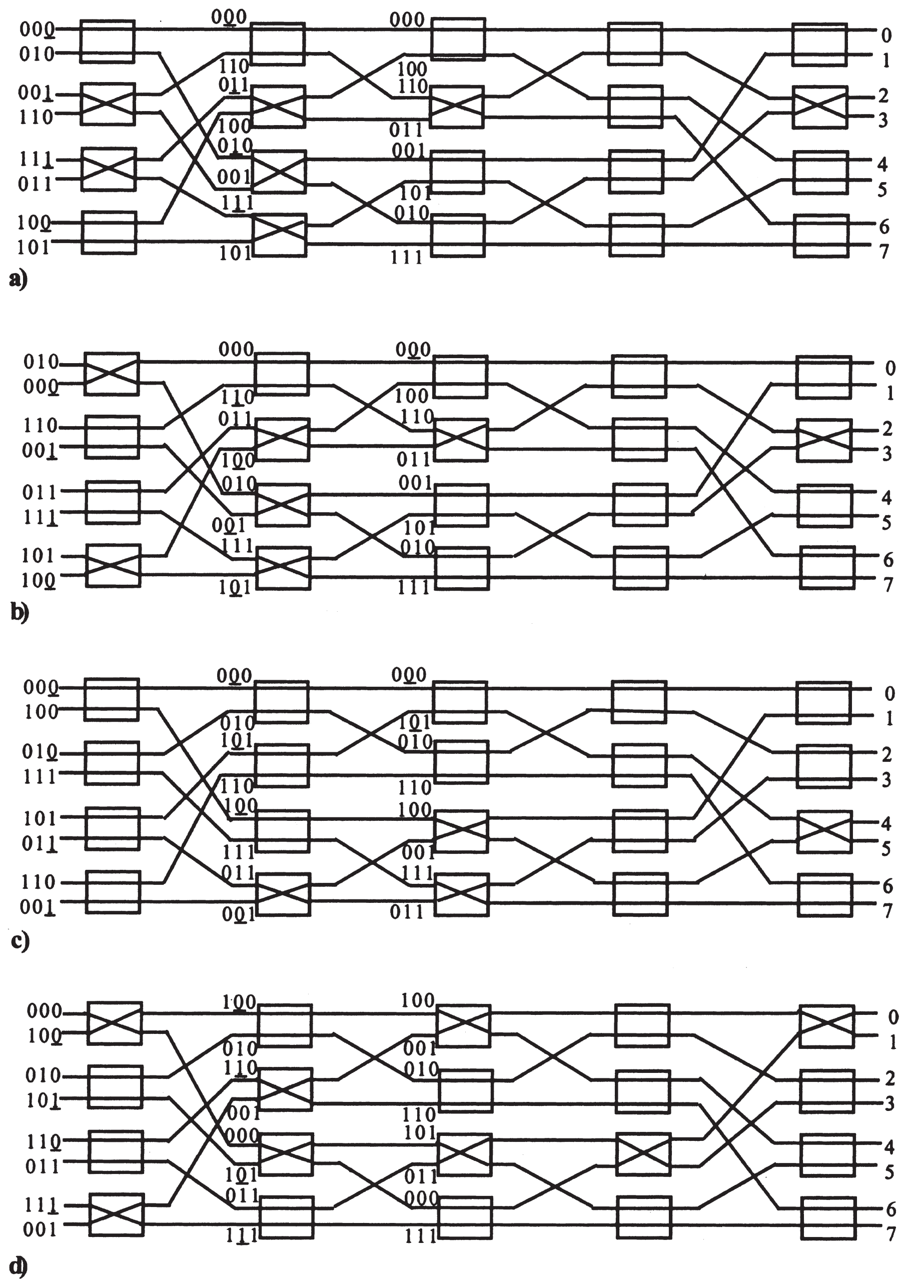

Fig. 3. Routing of permutations: (a) (0 2 1 6 7 3 4 5) by top-control-routing, (b) (2 0 6 1 3 7 5 4) by bottom-control-routing, (c) (0 4 2 7 5 3 6 1) by least-control-routing, and (d) (0 4 2 5 6 3 7 1) by highest-control-routing. (The underlined input bit determines the switchsetting).

permutation of $(x_{n-1},\dots,x_0)$, there exists $k, 0 \leqslant k \leqslant n-1$, such that $y_k = x_0$ or $\bar{x}_0$. Construct P' from P such that

$$P' : (x_{n-1},\dots,x_0) \to (y_{n-1},\dots,y_k,\dots,y_0).$$

Since $P' \in$ BPC, we have $P' \in$ TCR. From the mapping rule of P and P' it can be shown that if

$P = (p_0, p_1, p_2, p_3, \dots, p_{N-2}, p_{N-1})$ then

$$P' = (p_1, p_0, p_3, p_2, \dots, p_{N-1}, p_{N-2})$$

In other words, at the input of switches at stage 0, each switch will have the same pair of inputs (destination tags) for P and P', but their positions (top or bottom) will be reversed. Hence, if we route P according to BCR and P' according to TCR, we will have the same destination tags at the input of stage 1. It has been shown in [10] that, when P' is routed by the top inputs, the two halves of inputs at stage 1 will again be two BPC's ignoring the LSB. Therefore, when P is routed by bottom inputs, stage 1 will similarly have two BPC's at the two halves of the inputs. Therefore, by induction, P is bottom control routable.

It has already been shown that LCR $\supseteq$ BPC [11]. In order to show that BPC is also in HCR, we observe that for a BPC permutation, the larger of the two destination values at the inputs of every switch at stage 0 will consistently be always at the top or always at the bottom input. Hence, so far as stage 0 is concerned, routing a BPC by HCR amounts to routing it by either TCR or BCR. Again, routing a BPC by top or bottom input in stage 0 generates two BPC's at stage 1. Hence, it can be shown by induction that, HCR $\supseteq$ BPC.

Remark 3. *A BPC permutation P, generated from a BP permutation P^*, belongs to the IΩ set of permutations, if and only if $P^* \in I\Omega$. It has already been shown that $|BPC \cap I\Omega| = 1$ (the identity permutation)* [10].

Corollary 1. $|FSR| \geqslant 2^n(2^{n(N/2)-n} + n! - 1)$.

Proof. From definition, FSR contains RSR. Now RSR $\equiv$ IΩ. So, from Lemma 1, it follows that FSR contains BPC $\cup$ IΩ.

Now, $|\mathrm{BP} \cap \mathrm{I\Omega}| = 1$ (the identity permutation). Therefore, $|\mathrm{BPC} \cap \mathrm{I\Omega}| = 2^n$. Since $|\mathrm{BPC}| = 2^n n!$ and $|\mathrm{I\Omega}| = 2^{n.(N/2)}$, hence $|\mathrm{BPC} \cup \mathrm{I\Omega}| = 2^{n(N/2)-n} + n! - 1$.

Remark 4. *By definition, FSR is the intersection of four classes of self-routable permutations. Therefore, Corollary* 1 *gives a lower bound on the size of each class.*

3. Algorithm for class identification and routing

Given any $N \times N$ permutation P, our algorithm will check if P is routable by any of the self-routing techniques TCR, BCR, LCR or HCR as defined in Section 2. If there is a success, i.e., P is routable by one of the techniques, the algorithm will also generate the corresponding controls for switch setting.

We assume a multi-processor system with $N/2$ processing elements (PE) numbered as $0, 1, \dots, N/2 - 1$. Each PE-j will have two registers I_{2j} and I_{2j+1}, containing the destination tags for the inputs $\mathrm{T}_{j,j}$ and $\mathrm{B}_{i,j}$, of the switch $\mathrm{S}_{i,j}$, at any stage i, $0 \leqslant i \leqslant 2n-2$. To store the control bits for each $\mathrm{S}_{i,j}$, $0 \leqslant i \leqslant n-1$, each PE-$j$ will have an array of registers $C_0, C_1, \dots, C_{n-2}$. Initially, PE-j will store the outputs corresponding to inputs $2j$ and $(2j+1)$, as are in P, in registers I_{2j} and I_{2j+1}.

At any intermediate step i, $0 \leqslant i \leqslant 2n-2$, PE-j will store the destination tags of switching element $\mathrm{S}_{i,j}$, in stage i, and routes them according to a particular routing strategy M. We use two operations '*' (circular right shift on the least significant (n–i) bits) and '^' (circular left shift on the least significant $(i\text{–}n+3)$ bits), to simulate the interconnections between stages i and $(i+1)$, for $0 \leqslant i \leqslant n-1$ and $n-1 \leqslant i \leqslant 2n-2$, respectively. Finally, if the 0th bit of I_{2j} and I_{2j+1} are complements of each other, PE-j sets $success = 1$. If all the processing elements result $success = 1$, the control vectors $C_0, C_1, \dots, C_{n-2}$ from each PE-j are directly fed to set up the switching elements-j at stages i, $0 \leqslant i < n-1$. Thus, given any permutation P, if $P \in S_i$, where S_i is a self-routable class of permutations, the algorithm will output the control vectors necessary for routing.

In case of failure, some alternative values of M may be tried. Moreover, the algorithm may be modified a little to accommodate the trials for all classes of self-routable permutations sequentially one after another until it achieves a success or fails in all the cases when we are to apply the general looping algorithm [2] for routing.

3.1. Algorithm self-routing

Assume that M is the specified self-routing strategy (TCR/BCR/LCR/HCR) and $x(i)$ denotes the ith bit of the variable x.

```
begin
for any processor $j$, $0< j< N/2-1$ do
begin
Input (M); success := 0;
for $i := 0$ to $n-2$ do
begin
if M = TCR then $C_i := I_{2j}$ (i)
else if M = BCR then $C_i := I_{2j+1}$ (i)
    else begin
    if M = LCR then $k :=$ least $\{I_{2j}, I_{2j+1}\}$
    else $k :=$ highest $\{I_{2j}, I_{2j+1}\}$;
    if $k = I_{2j}$ then $C_i := I_{2j}$ (i)
    else $C_i := I_{2j+1}$(i);
    end;
if $C_i = 1$ then exchange($I_{2j}$, $I_{2j+1}$);
$j' := {*}2j$ ; $j'' := {*}(2j+1)$ ; $I'_j := I_{2j}$ ; $I''_j := I_{2j+1}$ ;
end;
for $i := (n-1)$ to $(2n-3)$ do
begin
if $I_{2j}$ $(2n-2-i) = I_{2j+1}$ $(2n-2-i)$
then terminate
    else begin
    if $I_{2j}$ $(2n-2-i) = 1$
    then exchange($I_{2j}$, $I_{2j+1}$);
    $j' := {}^{\wedge}2j$ ; $j'' := {}^{\wedge}(2j+1)$ ; $I_j := I_{2j}$; $I''_j := I_{2j+1}$;
    end
If $I_{2j}$ $(0) \neq I_{2j+1}$ $(0)$ then success := 1 and terminate
end;
end;
end.
```

It is easy to see that at each step-i, $0 \leqslant i < 2n-1$, each PE performs at most one comparison, one exchange and two data transfers (which can be done in parallel). Considering all these steps as a unit computation, the complexity of the algorithm comes out to be O(n) only.

4. Concepts of equivalence and closure sets

In [13–14], the concept of group-transformations was developed as a tool for optimal routing in n-stage unique-path full-access MIN's, i.e., baseline, omega etc. In this paper, we extend the idea to ($2n-1$)-stage networks, which are the concatenations of two n-stage unique-path full-access MIN's, with the central stage being common. It finds some excellent applications in the analysis of the relations between TCR (HCR) and BCR (LCR). For better understanding, the idea of group-transformations and some observations relevant to the following section, are described in brief.

Definition 5. For an $N \times N$ Benes network, the inputs (outputs) are grouped in different levels, as shown in Fig. 4. The size of a group at level i is 2^i. Two groups at level i are said to be adjacent, if both have the same parent at level $(i+1)$.

Definition 6. Let $a \leftrightarrow b$ denote: interchange a and b. A group-interchange tX(j:x), (where X = I stands for input and X = O refers to output) applied on a permutation P, interchanges elements of two adjacent groups of inputs (outputs) at level j, $0 \leqslant j < n$, following the rule $k \leftrightarrow k+2^j$, $x \leqslant k < x+2^j$, where x is the least element of the two groups. This process generates another permutation P' and is denoted by: tX(j:x) $[P] \rightarrow P'$.

Example 3. Consider a permutation $P = (7\ 6\ 2\ 4\ 0\ 3\ 1\ 5)$, and the group-interchange tI(1:4) such that tI(1:4)$[P] \rightarrow P'$; the interchanging input-pairs are: $4 \leftrightarrow 6$ and $5 \leftrightarrow 7$. Hence $P' = (7\ 6\ 2\ 4\ 1\ 5\ 0\ 3)$.

Similarly, a group-interchange on output tO(2:0) applied on P generates a permutation P'' given by $P'' = (3\ 2\ 6\ 0\ 4\ 7\ 5\ 1)$. (The output-pairs interchanged are: $0 \leftrightarrow 4$, $1 \leftrightarrow 5$, $2 \leftrightarrow 6$, $3 \leftrightarrow 7$).

Definition 7. Given a permutation P, an *input* (*output*) *cluster* CX(j,x), (X = I for input cluster and O for output cluster), is defined as the

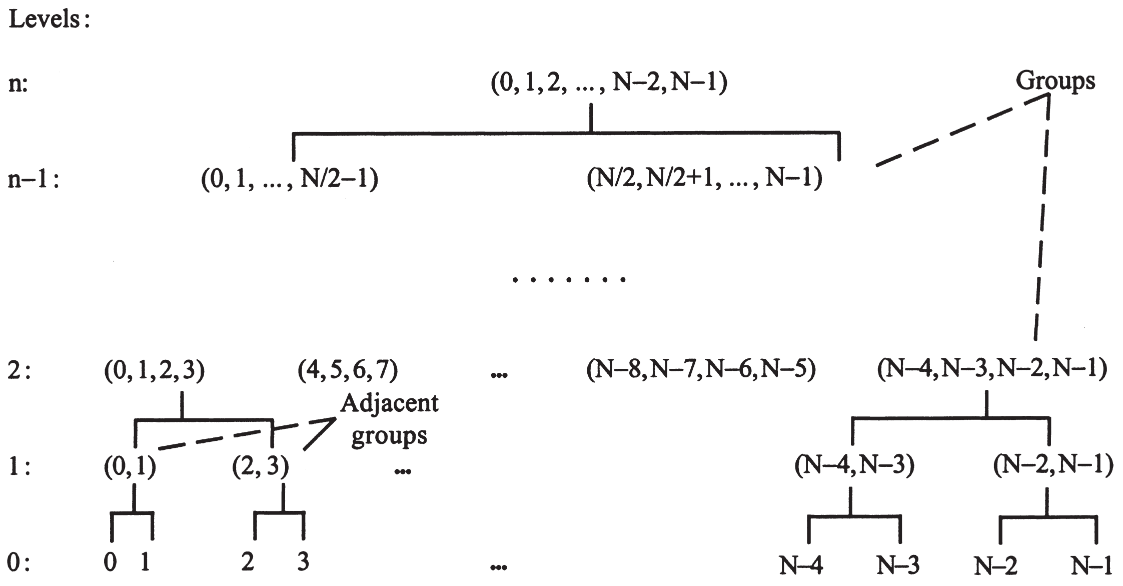

Fig. 4. The input (output) groups at different levels.

sequence of inputs (outputs) corresponding to the outputs (inputs) of the group at level j, whose least output (input) is x.

The set of input (*output*) *clusters* at level j, denoted by SCX(j) is the collection of all input (output) clusters at level j.

Example 4. For the permutation $P = (6\,7\,4\,2\,1\,5\,0\,3)$, CO(2,0) = (6,7,4,2), CI(2,4) = (2,5,0,1); SCO(1) = {(6,7), (4,2), (1,5), (0,3)}.

Observation 1. Any group-interchange on inputs (outputs) tX(j:x), keeps the sets of output (input) clusters SCX(i) of a permutation unaltered, for $0 \leqslant i \leqslant j$; for $i > j$, the elements of any output (input) cluster are preserved, but the sequence may change.

Definition 8. A *sequence of input* (*output*) *group-interchanges* $\{t\mathrm{X}(l_1 : x_1);\ t\mathrm{X}(l_2 : x_2);\ \ldots;\ t\mathrm{X}(l_k : x_k)\}$ is said to be ordered, if $i < j \Rightarrow l_i \leqslant l_j$ and if $l_i = l_j \Rightarrow x_i < x_j$.

Definition 9. Two sequences of input (output) group-interchanges RX_1 and RX_2 are said to be equivalent if for every permutation P, $\mathrm{RX}_1\ [P] = \mathrm{RX}_2\ [P]$.

Observation 2. For any sequence of input (output) group-interchanges, there exists an equivalent ordered sequence of input (output) group-interchanges.

Definition 10. An input (output) group-transformation GX, X = I or O is an ordered sequence of input (output) group-interchanges.

Observation 3. Input group-transformation defines an equivalence relation that partitions the set of all permutations into some equivalence classes. If a permutation P' is derivable from another permutation P by the application of some GI, i.e., $\mathrm{GI}[P] \rightarrow P'$ we will say that P and P' belong to the same partition defined by input group-transformation.

Obviously, the same is also true for output group-transformations but the partitions in the two cases may be different i.e., if $\mathrm{GO}[P] \rightarrow P'$, we will say that P and P' belong to the same equivalence class defined by output group-transformations.

Definition 11. Given any permutation P, let SX(P), X = I or O, denote the set of all permutations derivable from P by the application of all possible input (output) group-transformations. Then SX(P) is said to be the input (output) equivalence set of P.

Observation 4. Given any permutation P, the cardinality of its input (output) equivalence set SX(P) is $2N+1$, X = I or O.

Definition 12. A *group-transformation T* is defined as a sequence of an output group-transformation followed by an input group-transformation (any one may be a null sequence also).

Example 5. A group-transformation $T = \{t\mathrm{O}(0{:}0), t\mathrm{I}(1{:}4)\}$ applied on P will transform P in the following way:

$$P = (7\ 2\ 6\ 4\ 0\ 3\ 1\ 5) \overset{t\mathrm{O}(0:0)}{\rightarrow} P'$$
$$= (7\ 2\ 6\ 4\ 1\ 3\ 0\ 5) \overset{t\mathrm{I}(1:4)}{\rightarrow} P'' = (7\ 2\ 6\ 4\ 0\ 5\ 1\ 3).$$

It is easy to show that for any random sequence of input and output group-interchanges, there exists a group-transformation which imparts the same effect on any permutation.

Observation 5. Group-transformation induces an *equivalence partition* on the set of all permutations. If a permutation P' is derivable from another permutation P by applying some group-transformation T, i.e., if T $[P] \rightarrow P'$, we say that P' ^ P (P' is related to P) and it is easy to see that '^' is an equivalence relationship.

Definition 13. Given a permutation P, let C(P) denote the set of permutations derivable from P by the application of all possible group-transformations. Then C(P) is said to be the *closure set* of P.

Note that C(P) is a superset of SI(P) as well as of SO(P).

Observation 6. Given any permutation P, $|\mathrm{C}(P)| \geqslant 2^{N-1}$.

Observation 7. In a unique-path full-access MIN, all permutations in the same closure set have isomorphic conflict graphs; hence they all are routable by the same optimal routing algorithm.

Now, it is interesting to note that these concepts of equivalence classes and closure sets find an application in self-routing of permutations in Benes network. The following section gives the details.

5. Group-transformations and self-routable permutations

The following lemmata state some results on the cardinalities of the four self-routable classes of permutations.

Lemma 2. *The cardinality of the set of TCR permutations is exactly equal to the cardinality of the set of BCR permutations.*

Proof. Let us consider a permutation $P \in$ TCR and route it by top control routing technique. Now, let us apply input group-interchanges tI(0:x), for all possible values of x, on P that transforms it into P'. It will essentially exchange the two inputs, i.e., the destination tags $\mathrm{T}_{0,j}$ and $\mathrm{B}_{0,j}$ of all the switches $\mathrm{S}_{0,j}$, for $0 \leqslant j < N/2$. Since P was routable by top control, if we route P' by bottom control technique, the switches $\mathrm{S}_{1,j}$, for $0 \leqslant j < N/2$, will all have the same destination tags as they have for P under top control routing.

Similarly, it is easy to see that if we apply all the input group-interchanges $t\mathrm{I}(j : x)$, for $0 \leqslant j < n$ and for each j, with all possible values of x, on P to transform it into P^*, then if P is top control routable, P^* will be bottom control routable.

Therefore, for each $P \in$ TCR, there exists a unique P^* in the set BCR. It proves the lemma.

Example 6. Let $P = (0\ 2\ 1\ 6\ 7\ 3\ 4\ 5)$, $P \in$ TCR, its routing is shown in Fig. 3. Now let us generate P^* from P by the input group transformation, $\{t\mathrm{I}(2{:}0),\ t\mathrm{I}(1{:}0),\ t\mathrm{I}(1{:}4),\ t\mathrm{I}(0{:}0),\ t\mathrm{I}(0{:}2),\ t\mathrm{I}(0{:}4),\ t\mathrm{I}(0{:}6)\}[P] \rightarrow P^*$, i.e., $P^* = (5\ 4\ 3\ 7\ 6\ 1\ 2\ 0)$. Note that $P{*} \in$ BCR. The routing of P^* is shown in Fig. 5.

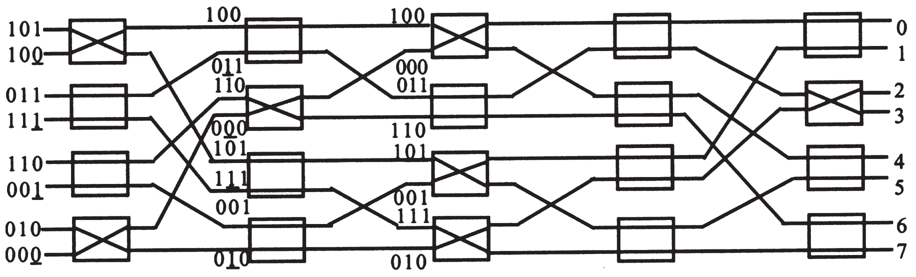

Fig. 5. Routing of permutation P^*:(5 4 3 7 6 1 2 0) by bottom-control-routing. (The underlined input bit determines the switchsetting.)

Lemma 3. *The cardinality of the set of LCR permutations is exactly equal to the cardinality of the set of HCR permutations.*

Proof. Let us consider a permutation $P \in$ LCR and route it through the Benes network by least control routing technique. Say it generates the permutation P_i, at the input of stage i, $0 \leqslant i \leqslant n-2$.

Now let us complement all the bits of each output corresponding to each input in P. It is actually obtained by applying all possible output group-interchanges at each level i, $0 \leqslant i \leqslant n-1$, on P, that generates a new permutation P'. At stage $i=0$, for any switch, if the top (bottom) input was the least one for P, now it is the highest one for P'.

Now let us route P', by HCR through the stage $i=0$. Note that the position of the R-input remains the same, but the R-bits are complements to each other, for P and P', respectively. Let the permutation at the input of stage $i=1$ be P'_1. If we complement all the bits of each destination of P'_1, and compare it with P_1, it is found to be just $\{t\text{I}(\text{n}-1{:}0)\}[P_1]$.

Let us route P'_1 by HCR at stage $i=1$. Say it generates the permutation P'_2 at the input of stage $i=2$. If we complement all the bits of each destination tag of P'_2, and compare it with P_2, we find that it is nothing but $\{t\text{I}(n{-}1{:}0),\ t\text{I}(n{-}2{:}0),\ t\text{I}(n{-}2{:}N/2)\}$ $[P_2]$.

If we continue in this way at the input of stage $i=n{-}1$, we will find that by complementing all the bits of each destination tag of P'_{n-2}, it turns out to be

$$\{t\text{I}(n-1:z_1), t\text{I}(n-2:z_2), \ldots, t\text{I}(1:z_{n-1})\}[P_{n-2}]$$

Now since P_{n-2} was routable in the remaining n stages of the network, so will be P'_{n-2}. It proves that P' is routable in the Benes network by HCR.

Hence, it is evident that in a Benes network, for any permutation $P \in$ LCR, there exists a corresponding permutation $P' \in$ HCR. Since the mapping from P to P', is a one-to-one and onto mapping (complementation of all bits of each destination tag), it proves that the cardinality of the set of LCR permutations is exactly equal to that of the set of HCR permutations.

Example 7. Let us consider a permutation $P = (0\ 4\ 2\ 7\ 5\ 3\ 6\ 1)$, where $P \in$ LCR, the routing is shown in Fig. 3(c).

Now let us transform P to P' in the following way:$\{t\text{O}(2{:}0),\ t\text{O}(1{:}0),\ t\text{O}(1{:}4),\ t\text{O}(0{:}0),\ t\text{I}(0{:}2),\ t\text{O}(0{:}4),\ t\text{O}(0{:}6)\}[P] \rightarrow P'$, where $P' = (7\ 3\ 5\ 0\ 2\ 4\ 1\ 6)$. Note that $P' \in$ HCR. The routing of P' is shown in Fig. 6.

In [11], it has been mentioned that the least-control self-routing technique is applicable to LC (linear-complement) class of permutations as well as IΩ permutations. The following theorem states an interesting property of LCR and HCR classes, that extends the applicability of LCR (HCR) class further.

Theorem 1. *If a permutation $P \in LCR(HCR)$, then any permutation ($P' \in SI(P)$ also belongs to LCR (HCR), where $SI(P)$ is the input equivalence set of P.*

Proof. Let us route both P and P' by least control routing technique in the first n stages. After that P is obviously routable by destination tag. We

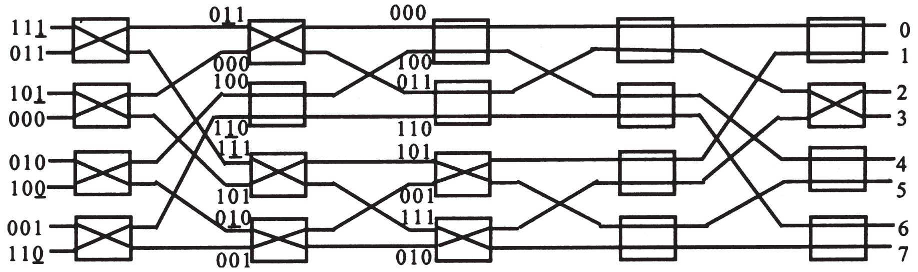

Fig. 6. Routing of permutation P': (7 3 5 0 2 4 1 6) by highest-control-routing. (The underlined input bit determines the switchsetting.)

are to show that P' is also routable by destination tag.

We note certain similarities in the sets of inputs to the different stages of the network for P and P'. From Observation 1, the output clusters of P and P' are the same. Specifically, in both the cases, the set of output clusters at level 1 are the same. The routing algorithms are the same and so also are the pairings at all the switches. So an input which is routed through the upper link of a switch in P, will also be routed through the upper link of a switch in P'. The set of inputs which are routed through the upper links of the switches in stage 0 forms the input set for the top half of stage 1. So, both P and P' will have the same set of inputs for the top half (and also the bottom half) of stage 1.

Let us think of the inputs to stage 1, also as a permutation. Let the permutations corresponding to P and P' be $P_{(1)}$ and $P'_{(1)}$, respectively. We observe that the sets of output clusters of $P_{(1)}$ and $P'_{(1)}$ are the same. Take the output clusters of size 2^k in $P_{(1)}$ (say, the top half). This is formed by the upper links of an output cluster of size 2^{k+1} of P. This output cluster of P is also present somewhere in P' (Observation 1). Now consider the inputs, routed through the upper links of that output cluster of P'. It forms an output cluster of size 2^k in $P'_{(1)}$. This output cluster of $P'_{(1)}$ is the same as the output cluster of $P_{(1)}$ we started with.

If we repeat the above arguments, it is clear that at stage i, the output clusters at level i, are not only the same, but also in same position with respect to the inputs. Also, in general the set of output clusters of $P_{(i)}$ and $P'_{(i)}$ are the same. From this, we see that, the input pairs to all the switches of stage $(n-1)$ are the same for $P_{(n-1)}$ and $P'_{(n-1)}$. Since $P_{(n-1)}$ is passable, so is $P'_{(n-1)}$.

It is evident that the same will be true for any permutation routable by highest control routing technique as well.

Definition 14. Let Q be any set of permutations, then QI is defined as the union of all input equivalence classes generated by the permutations $P \in Q$, i.e., $QI = \bigcup_{\forall P \in Q} \mathrm{SI}(P)$.

Corollary 2. *Any permutation* $P \in QI$, *where* $Q = L \bigcup I\Omega$ *is routable by least-control routing technique.*

Proof. Follows directly from Theorem 1, since LC and IΩ classes are routable by least-control routing technique.

Definition 15. The BPIE (*bit-permute input-equivalence*) class of permutations is defined as: $\mathrm{BPIE} = \bigcup_{\forall\, P \in \mathrm{BP}} \mathrm{SI}(P)$.

Remark 5. *Since*

$BP \subseteq LCR(HCR), BPIE \subseteq LCR(HCR)$.

Remark 6. $|BPIE| = n!2^{N-1}$, *as* $|BP| = n!$.

Corollary 3. $|LCR \cap HCR| \geqslant n!2^{N-1}$.

Proof. Clear from the remark above, since $|\mathrm{BPIE}| = n!2^N - 1$. The exact number of permutations covered by each of these classes are much

Table 1

N	$N!$	LCR	HCR	TCR	BCR	SR	BPC	BPIE	BPC ∪ IΩ
4	24	24	24	20	20	24	8	16	20
8	40 320	21 888	21 888	11 632	11 632	30 208	48	768	4136

larger than BPIE class or IΩ class . For $N=4$ and 8 , the exact number of permutations in each class has been found by simulation. The results are given in Table 1.

Fig. 7 shows the relationship among the different self-routable sets of permutations for $N=8$. Here total number of self-routable permutations is denoted by SR, i.e.,

$$\mathrm{SR} = \mathrm{TCR} \cup \mathrm{BCR} \cup \mathrm{HCR} \cup \mathrm{LCR}.$$

From Table 1, it can be seen that for $N=4$, all permutations are self-routable, and for $N=8$, approximately 75% of the toal number of permutations are self-routable. Moreover, in both cases, the total number of self-routable permutations are much much larger than BPC or BPIE or BPC ∪ IΩ class of permutations. Some additional observations have been made from the experimental results.

Observation 8. For $N=$ 8, $|\mathrm{FSR}|=8034$, where $\mathrm{FSR}=\ \mathrm{TCR} \cap \mathrm{BCR} \cap \mathrm{HCR} \cap \mathrm{LCR}$, i.e., about 20% permutations are self-routable by any one of the self-routing techniques, mentioned here, where for $N=4$, $|\mathrm{FSR}|=20$, compared to $N!=24$ only.

Observation 9. For $N=8$, $|\mathrm{SR}|=30\,208$, whereas $|\mathrm{LCR} \cup \mathrm{HCR}|=28\,132$. It indicates that most of the self-routable permutations are routable by LCR and or HCR only. For $N=4$, $|\mathrm{SR}|=\mathrm{N}!=|\mathrm{LCR}|=|\mathrm{HCR}|$.

6. Conclusion

In this paper, we propose four simple self-routing strategies for $N\times N$ Benes network B(n), $n=\log_2 N$. It is shown that the union of the BPC and IΩ classes of permutations is a subset of the intersection of all the four classes of permutations

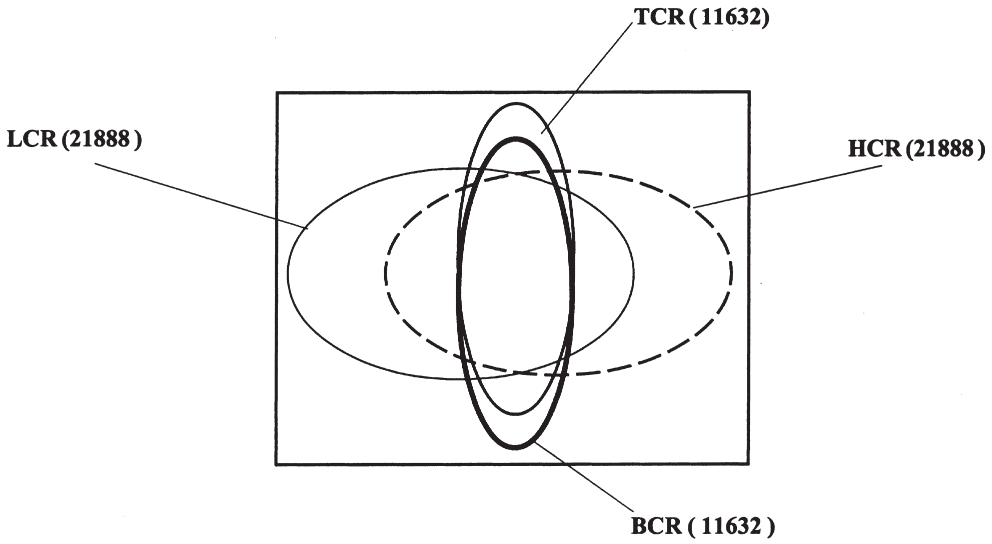

Fig. 7. Different self-routable sets for $N=8$.

routable by the proposed self-routing strategies. It implies that $2^n(2^{(nN/2)-n}+n!-1)$ is the lower bound on the cardinality of any class of self-routable permutations, considered here. The enumeration of the exact cardinality of each class is still an open problem. But by the application of the theory of equivalence classes as presented in [13], we establish that $|\mathrm{TCR}|=|\mathrm{BCR}|$, $|\mathrm{LCR}|=|\mathrm{HCR}|$ and that $|\mathrm{LCR}\cap\mathrm{HCR}|\geqslant n!2^{N-1}$. We develop an algorithm, with time complexity O(n) that will identify if any given permutation $P\in \mathrm{S}_i$, where S_i is a self-routable class of permutation mentioned here and also generates the necessary controls for self-routing of P. It has been established that if a permutation $P\in$ LCR (HCR), any permutation P' in the input-equivalence set of P, is also least (highest)-control routable. Hence, the routing algorithms presented here, effectively enhances the set of permutations routable in Benes network in O(n) time, to a large extent. Moreover, it is evident that the same self-routing techniques can be applied to any ($2n-1$)-stage MIN, which is topologically equivalent to a Benes network.

References

[1] T.H. Szymanski, Design principle for practical self-routing nonblocking switching networks with O ($N$log$N$) bit complexity, IEEE Trans. Comput. (1997) 1057–1069.

[2] A. Waksman, A permutation network, J. Assoc. Comput. Mach. 15 (1) (1968) 159–163.

[3] D.C. Opferman, N.T. Tsao-Wu, On a class of rearrangeable switching networks-Part I: Control algorithm, Bell. Syst. Tech. J. 50 (5) (1971) 1601–1618.

[4] K.Y. Lee, A new benes network control algorithm, IEEE Trans. Comput. (1987) 768–772.

[5] Y.M. Yeh, T. Feng, On a class of rearrangeable networks, IEEE Trans. Comput. (1992) 1361–1379.

[6] Q. Hu, X. Shen, J. Yang, Topologies of combined $2\log N-1$)-stage interconnection networks, IEEE Trans. Comput. (1997) 118–124.

[7] D. Nassimi, S. Sahni, Parallel algorithm to set up the Benes permutation network, IEEE Trans. Comput. (1982) 148–154.

[8] T.Y. Feng, S.W. Seo, A new routing algorithm for a class of rearrangeable networks, IEEE Trans. Comput. (1994) 1270–1280.

[9] J. Lenfant, Parallel permutations of data: A Benes network control algorithm for frequently used permutations, IEEE Trans. Comput. (1978) 637–647.

[10] D. Nassimi, S. Sahni, A self-routing Benes network and parallel permutation algorithms, IEEE Trans.Comput. (1981) 332–340.

[11] R. Boppana, C.S. Raghavendra, On self-routing in Benes and shuffle exchange networks, in: Proceedings of the International Conference on Parallel Processing, 1988, pp. 196–200.

[12] D. Nassimi, A fault-tolerant routing algorithm for BPC permutations on multistage interconnection networks, in: Proceedings of the 1989 International Conference on Parallel Processing, 1989, pp. I278–I287.

[13] N. Das, B.B. Bhattacharya, J. Dattagupta, Analysis of conflict graphs in multistage interconnection networks, IEEE Trans. Comput. (1993) 665–677.

[14] N. Das, B.B. Bhattacharya, J. Dattagupta, "Hierarchical classification of permutation classes in multistage interconnection networks, IEEE Trans. Comput. (1994) 1439–1444.

Nabanita Das received the B.Sc. (Hons.) degree in Physics, B. Tech. degree in Radio Physics and Electronics from the University of Calcutta, the M. E. degree in Electronics and Telecomm. Engineering and Ph. D. degree in Computer Science, both from the Jadavpur University, Calcutta. Since 1986, she has been on the faculty of the Advanced Computing and Microelectronics unit, Indian Statistical Institute, Calcutta. In 1997, she visited the Department of Mathematik and Informatik, University of Paderborn, Germany, as a visiting scientist. Her research interests include parallel processing, interconnection networks, testing, distributed architectures and mobile computing.

Krishnendu Mukhopadhyaya received the B. Stat (Hons.), M. Stat, M. Tech. in Computer Science and Ph. D. in Computer Science degrees in the years 1983, 1985, 1987 and 1994, respectively, all from the Indian Statistical Institute, Calcutta. Since 1993 he is working in the Department of Mathematics, Jadavpur University, Calcutta as a Lecturer in Computer Science. Currently, he is an Associate Professor in the Advanced Computing & Microelectronics unit of the Indian Statistical Institute, Calcutta. During 1998–1999 he visited the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, USA, for one year, under the BOYSCAST Fellowship of the Government of India. He was the recipient of the Young Scientist Award of the Indian Science Congress Association in the year 1994. His research interests are in the areas of Interconnection Networks, Parallel and Distributed Processing etc.

Jayasree Dattagupta received the B. E. degree in Electronics and Telecomm. Engineering and a Post-Graduate Diploma in Computer Science, both from Jadavpur University, Calcutta in 1968 and 1969, respectively. She received the M. Phil. degree from the Brunel University, UK in 1976 and Ph. D. degree in Computer Science from Calcutta University in 1980. In 1970 she joined the faculty of Indian Statistical Institute, Calcutta, India where she is currently a Professor. She had been with the Informatik Kolleg, GMD, Bonn during 1982–1983 as a visiting scientist. She served as the chairman of the Indian sub-committee of the International Test Conferences in 1989. She was also the vice-chairman of the Indian sub-committee of the International test conferences during 1987–1989 and a member of the programme committee of the International test conferences during 1987–1990. Her research interests are in the areas of computer networks, fault-tolerant computing and parallel processing.