ANALYSIS OF PERMUTATIONS FOR ROUTING IN MULTISTAGE INTERCONNECTION NETWORKS

INDEX TERMS :

Multistage interconnection network (MIN), Base-line network, Conflict resolution, Seed Permutation.

BY

M. DEVI PRASAD

UNDER THE SUPERVISION OF

MS. NABANITA DAS

ACKNOWLEDGEMENTS

I take this oppurtunity to thank my guide Ms.Nabanita Das for offering me this project and for the excellent guidance she has given me.

It is a pleasure to thank my frinds Bhaskar, Krishna mohan, Manisha and Prasenjit for allowing me to use their user accounts on the VAX system at ISI, Cal. for the execution of the programs to generate the seeds for a 16 x 16 base-line network.

ABSTRACT

The isomorphism of conflict graphs in multistage interconnection networks (MIN) is analyzed here. We introduce a concept called group transformation which allows us to partition the set of all permutations into several equivalence classes. All members belonging to the same class have isomorphic conflict graphs. Finally we describe two algorithms, one for the generation of the seed permutation for any given input permutation and the second one to generate all possible seeds for a given N x N base-line network.

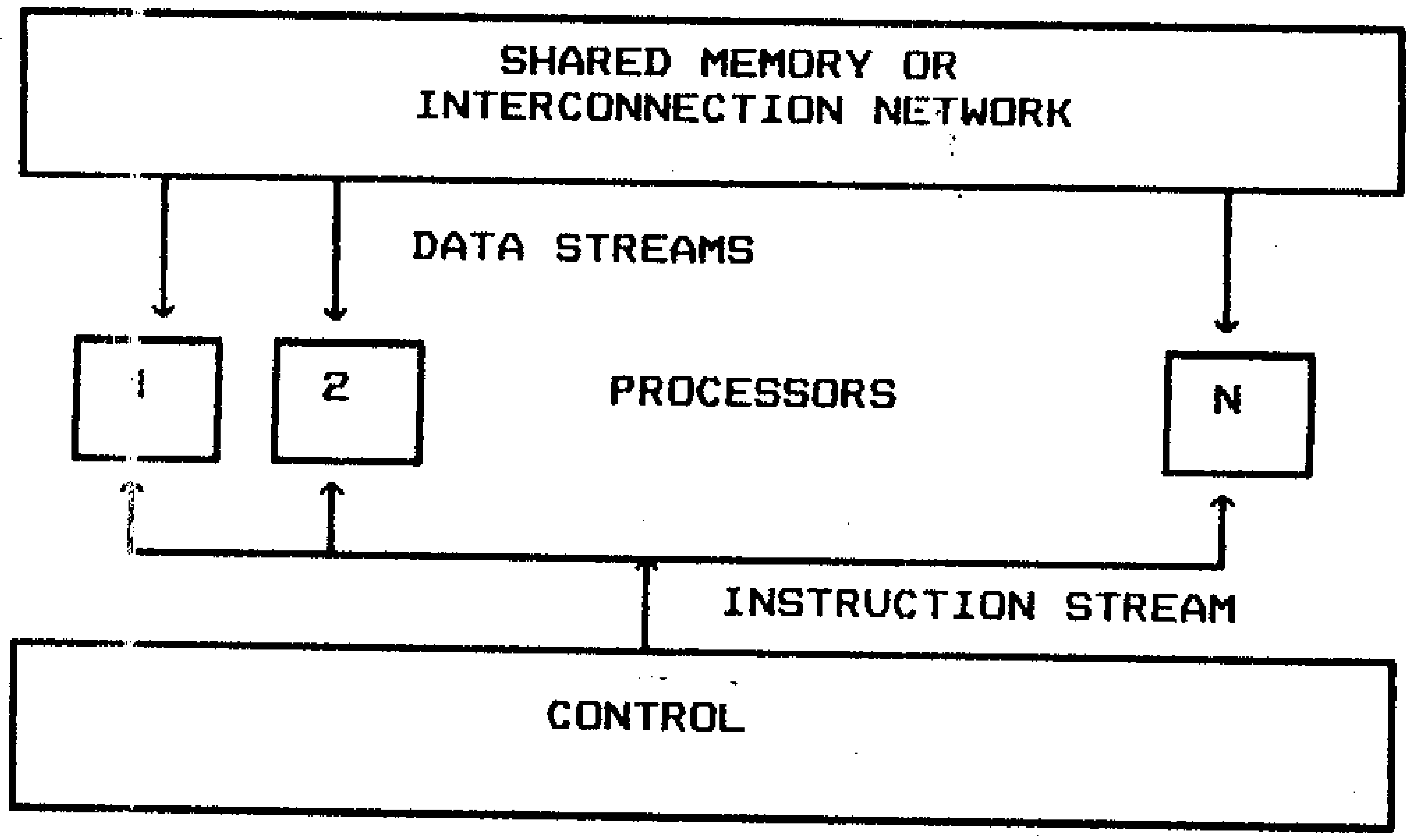
CONTENTS

CHAPTER 1 : INTRODUCTION.

Any computer whether sequential or parallel, operates by executing instructions on data. Depending on whether there is one or several of these streams, we can distinguish between four classes of computers.

1. Single instruction stream, single data stream *(SISD)*.
2. Multiple instruction stream, single data stream *(MISD)*.
3. Single instruction stream, multiple data stream *(SIMD)*.
4. Multiple instruction stream, Multiple data stream *(MIMD)*.

An SIMD computer consists of N identical processors as shown in figure 1. Each processor has its own local memory where it can store programs and data. All processors operate under the control of a single instruction stream issued by a central control unit. It is desirable for the processors to be able to communicate among themselves during the computations in order to exchange data or intermediate results. This can be achieved by two ways, giving rise to two subclasses : SIMD where the communication is through a shared memory (SM SIMD), and SIMD where the communication is via an interconnection network.



SIMD COMPUTER.
FIGURE 1.

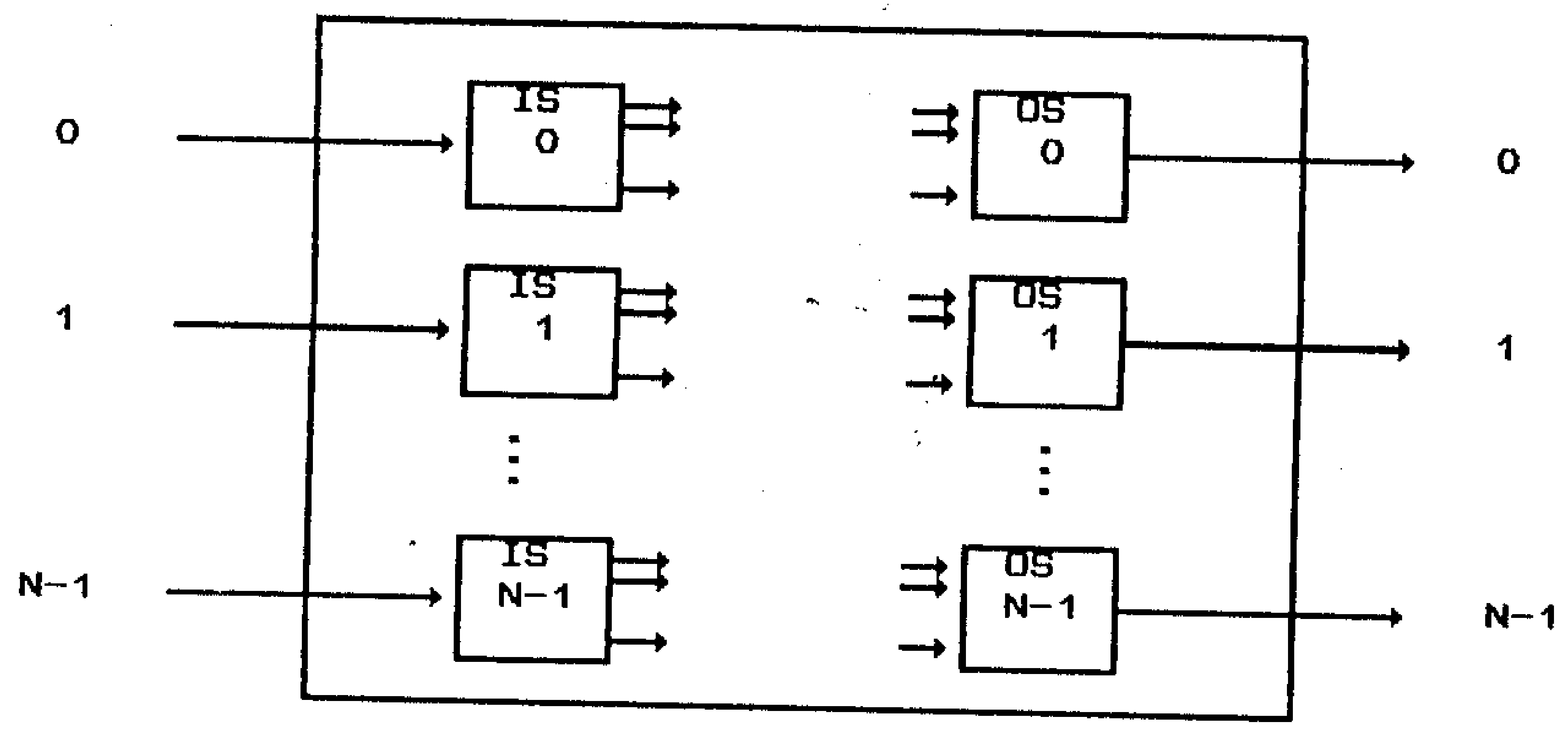
A typical interconnection is a circuit switching network which consists of a number of switching elements and interconnecting links. Interconnection functions are realized by properly setting control of the switching elements. The SM SIMD computer is a fairly powerful model of computation. This can furthur be made more efficient by dividing the shared memory into modules so that each module can be accesed by any processor independently. The Communications between processors and the memory modules can be implemented effectively in lesser cost by an interconnection network. Thus interconnection networks are extensively used in parallel processing systems. Consider a system with N processors and N memeory modules, where the processors and the memory modules are addressed by the elements of the set S = {0,...,N-1}. The interconnections at any instant of time in the interconnection network can be represented as a permutation i.e. a bijection of S onto itself.

Circuit switching networks are of three types : Non-Blocking, Rearrangeable and Blocking networks. In a blocking interconnection network simultaneous connection of more than one input-output pair might require the same link.The link is usually said to be in conflict, and these interconnections cannot be performed simultaneously. As N becomes larger, segmenting a network into more than one stage is more economical. This leads to Multistage interconnection network (MIN). A log N stage MIN requires minimum hardware cost retaining the properity of full access. In full access MIN, any output terminal can be connected from any input terminal by some path. If this path is unique then the MIN is also called a full access unique path MIN. Examples of unique path full access networks are base-line network, omega network etc. There are N! possible interconnection permutations for an N x N input-output network. For a given network structure not all permutations are conflict free. A conflict free permutation is one which does not lead to any conflict in any interconnection link. Given an arbitrary permutation dividing it into minimum number of subsets such that each of these subsets can be routed in a single pass without any conflict is called as conflict resolution problem. Routing these subsets is called as optimal routing of the permutation. The optimal routing in the blocling MIN is an NP-HARD problem.

In this dissertation work, we have simplified the optimal routing problem for a base-line network by dividing the N! permutations into several equivalence classes of permutations such that all permutations which belong to a particular class can be routed in a similar fashion. We define a equivalence relation on the permutations called as group transformations, which defines the above mentioned partition. Though the concept of group transformations have been defined for a base-line network it can be extended to the other type of networks. For each of these classes of permutations, the permutation which is the lexicographical minimum is defined as the *SEED PERMUTATION* of the class. The seed permutations or simply seeds characterize the permutation classes i.e. a seed is unique for a class. Hence if the optimal routing, for the seed of a class, is known it can be applied to the entire class of permutations to which the seed belongs. The optimal routing problem has been solved for classes of permutations such as the BPC class (See [RA-86]), but BPC includes a very small fraction of the whole set of possible permutations and with the increase in N this fraction asymptotically becomes vanishingly small. BPCL is the union of all equivalence classes generated by the BP's. (See [NBJ-90]) All BP's are seed permutations. Therefore, as the optimal routing for BP's is known, all permutations in a BPCL can be routed optimally. Finally two algorithms are given, the first one generates the seed permutation of the class of permutations into which the input permutation belongs. The second algorithm which calls the first one generates all possible seeds for a given N x N base-line network.

CHAPTER TWO : INTERCONNECTION NETWORKS.

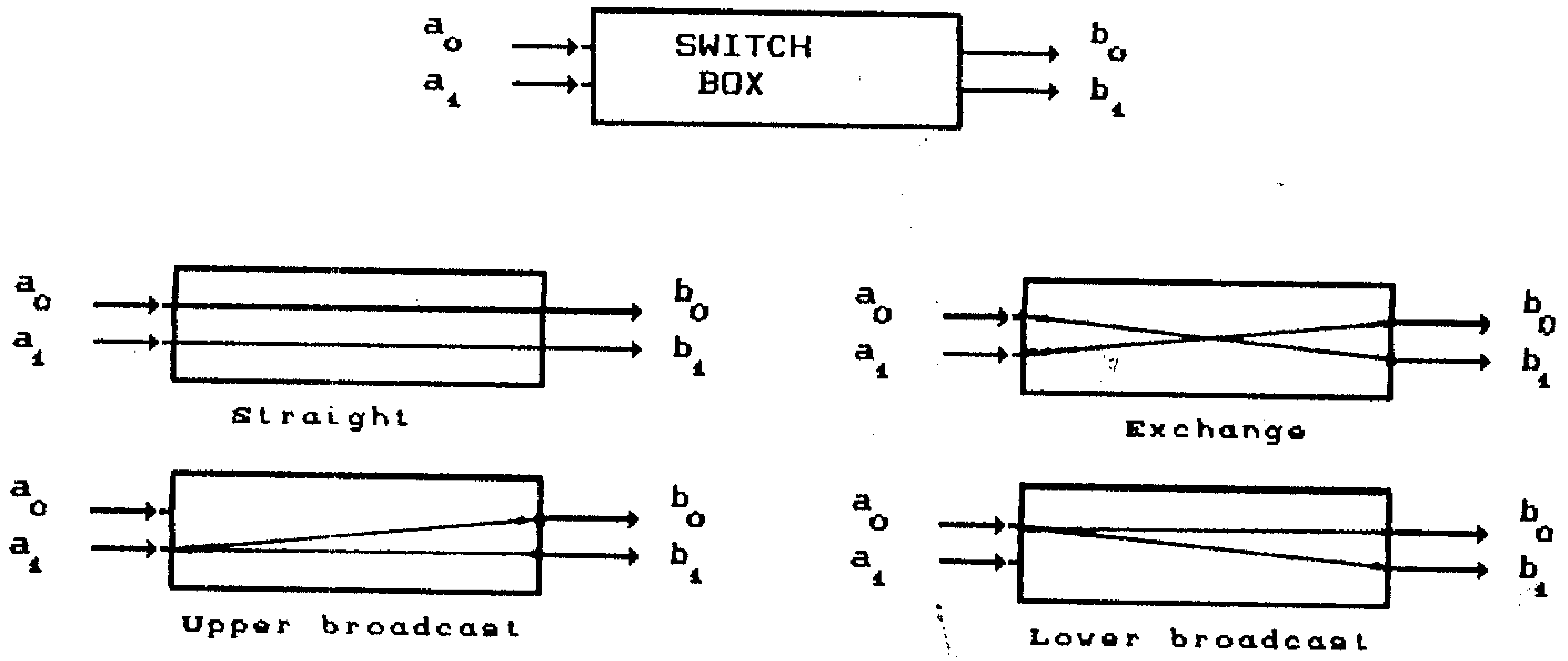
Typically, an interconnection network consists of a number of switching elements and interconnecting links. Interconnections are realized by properly setting the control of the switching elements. Interconnection networks are classified into two categories based on network topologies : STATIC AND DYNAMIC NETWORKS. In static networks, the communication paths are fixed, where as in the dynamic networks the interconnections can be altered by the control signals. Dynamic networks can furthur be classified into two classes : SINGLE-STAGE AND MULTI-STAGE NETWORKS. A single stage network is a switching network with N input selectors and N output selectors, as in figure 2. Each input selector is a 1 to D demultiplexer and each ouput selector is a M to 1 multiplexer, where $1 <= D <= N$ and $1 <= M <= N$. For instance a crossbar-switching network is a single-stage network with $D = M = N$. To establish a desired connecting path, different control signals are applied to all input and output selectors. The single-stage network is also called a recirculating network.



CONCEPTUAL VIEW OF A SINGLE STAGE INTERCONNECTION NETWORK

FIGURE 2.

Many stages of interconnected switches form a multi-stage interconnection network *(MIN)*. MINs are described by three characterizing features : the switchbox, the network topology, and the control structure. Many switch boxes are used in a MIN. Each box is essentially an interchange device with two inputs and two outputs, as depicted in the figure 3. Four stages of the switch box are illustrated : straight, exchange, upperbroadcast and lower broadcast. A two function switch box can assume either stratght of exchange states only. A MIN which is capable of connecting an arbitrary input terminal to an arbitrary output terminal is called a full-access MIN. Furthur it is called a unique path MIN if the path between each input-output pair is unique. MINs can be one sided or two sided. The two sided networks, which usually have an input side and an output side, can be divided into three classes : BLOCKING, REARRANGEABLE AND NONBLOCKING NETWORKS.



A 2 x 2 SWITCHING BOX AND ITS FOUR INTERCONNECTION STATES..

FIGURE 3.

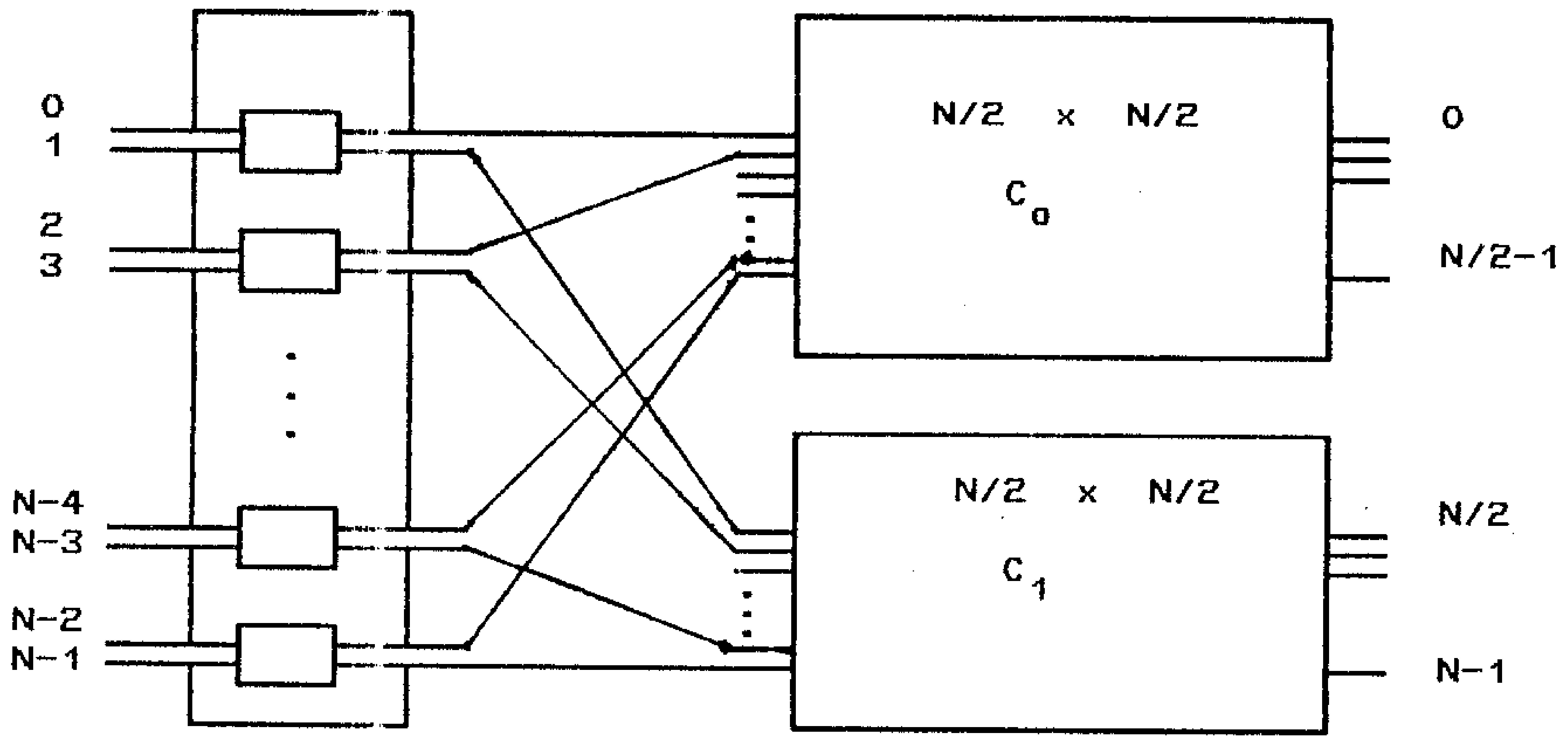
In blocking networks, simultaneous connections of more than one terminal pairmay may result in conflicts in the use of network communication links. Examples of blocking networks are Omega network, Flip network and the Base-line network. A network is called as rearrangeable network if it can perform all possible connections between inputs and outputs by rearranging its existing connections so that a connection path for a new input-output pair can always be established.Benes network is an example of this class of networks. A network which can handle all possible connections without blocing is called a nonblocking network.Clos network is an example of this class of networks.

Generally, a MIN consists of n stages where $N = 2^n$, is the number of input and output lines.Therefore each stage may use N/2 switch boxes. The interconnection pattern between stage to stage determine the network topology. Each stage is connected to the next stage by atleast N paths. The network delay is proportional to the number n stages in a network. The cost of a size N, MIN is proportional to N.Log(N). The control structure of a network determines how the stages of the switch boxes will be set. Two types of control structures are used in a network construction. The INDIVIDUAL STAGE CONTROL uses the same control signal to set all switch boxes in the same stage. Another control philosophy is to apply INDIVIDUAL BOX CONTROL. Here a seperate control signal is used to set the state of each switch box. This scheme offers higher flexibility in setting up the connecting paths at the cost of increased control circuitry cost.

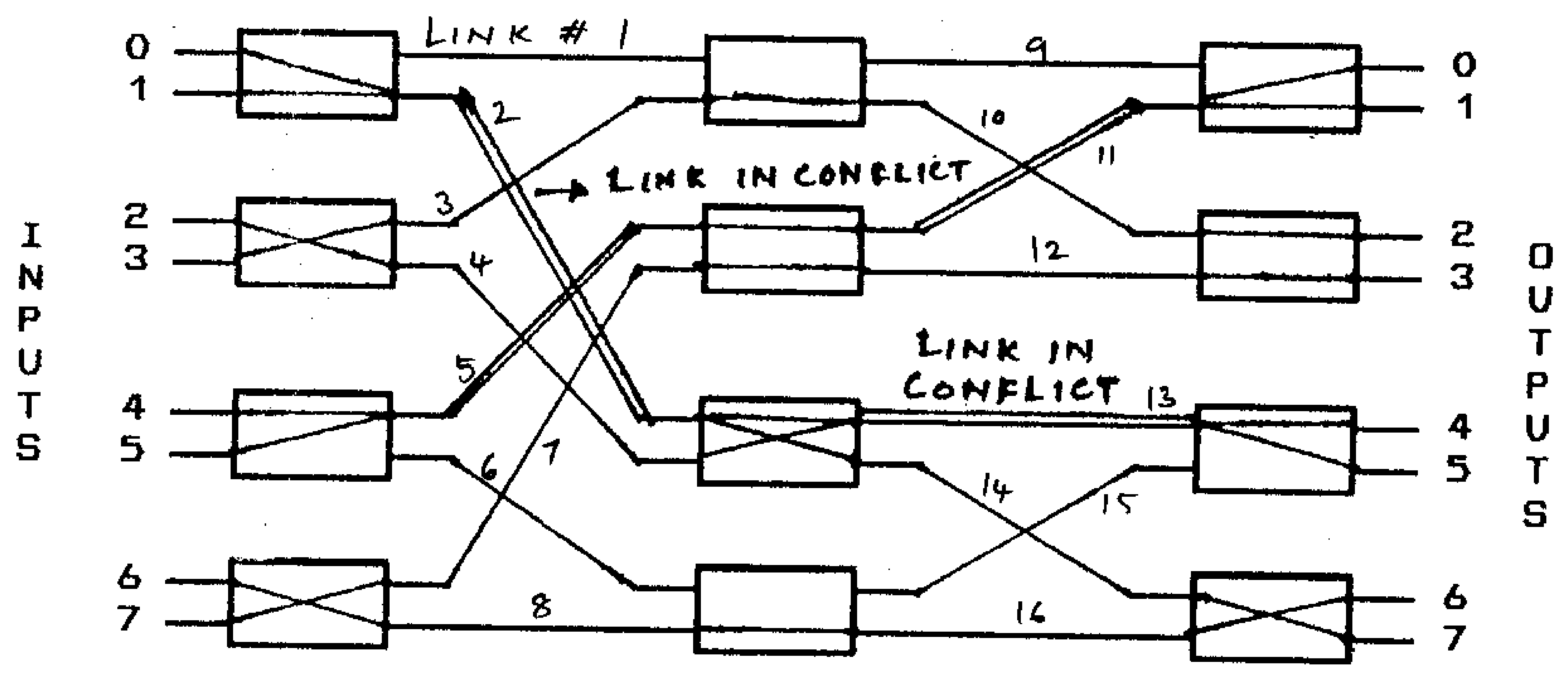
The performance of an interconnection network is determined largely by its configuration. By configuration of an interconnection network we mean the topology and the label of the components in the interconnection network. Three parameters may be used to design an interconnection network. These three parameters are the number of communication paths of each switching element, the number of columns (or stages), and the interconnection paths (or links) between switching elements. Here we consider a $\log_2 N$ stages of 2x2 switching elements, i.e. switching elements, each with two input and two output terminals. A $\log_2 N$ stage network is important because it incorporates minimum hardware required for a network to have the full access properity, but a $\log_2 N$ stage network is a blocing type of network. In this dissertation work we mainly consider the base-line network, which is a blocking,unique-path and full-access MIN

BASE-LINE NETWORK :

The topology of a base-line network can be generated in a recursive way. Figure 4 shows the first interation of the recursive process in which the first stage contains two (N/2)x(N/2) subblocks, C_0 and C_1. The process can recursively be applied to the subblocks in each iteration until the N/2 subblocks of size 2x2 are reached. To complete the process, $\log_2(N)-1$ iterations are needed. A Base-line network is a typical full-access unique path MIN. A 8x8 base-line network is shown in the figure 5.



RECURSIVE PROCESS TO CONSTRUCT THE BASE-LINE NETWORK.
FIGURE 4.



8 X 8 BASE-LINE NETWORK.
FIGURE 5.

Base-line network is a self-addressing network, i.e. a path from input i to output j is obtained by representing i XOR j in bit form, say $(q_{n-1}q_{n-2} \ldots q_0)$. Now the path is obtained by taking a diognal link or parallel link in the current switching element based on whether q_i is 1 or 0, for i = n-1 downto 0.

A permutation is a one-one mapping form the set s = {0, ... ,N-1} onto itself. It is represented as $\begin{matrix} 0 & 1 & \ldots N-1 \\ x_1 & x_2 & x_{n-1} \end{matrix}$ where x_1, x_2 .. x_{N-1} are the outputs.

We represent the above permutation simply as $(x_1, x_2, \quad x_{N-1})$, assuming the inputs to be ordered from 0, ..., N-1.

Example 1 : Consider a permutation P : (7 5 4 2 1 0 6 3). The interconnections made in a base-line network to realize P are shown in figure 5. For realizing this permutation we set up the paths from 0 to 7 and now if we try to set up the path from 1 to 5 we notice that the link number 2 is required for this, but link 2 is already used in the path from 0 to 7. This is a conflict.

If the realization of a permutation leads to a conflict in some link, it cannot be routed in a single pass. One way to route such permutations is to partition it into subsets such that each subset can be routed in a single pass. If the number of subsets is minimum then the routing is called as optimal routing. Finding a optimal routing for given permutation and a blocking MIN, is called as the optimal routing problem.

CHAPTER THREE : ANALYSIS OF PERMUTATIONS FOR A BASE-LINE NETWORK.

Given an arbitrary permutation P in an blocking MIN, the problem of partitioning p into minimum number of subsets such that each subset can be routed in a single pass without any conflict is commonly known as the conflict resolution problem. In general optimal routing is an NP-Hard problem,but it had been solved for the subclasses of permutations such as for BPC class. In this dissertation work our objective is to find classes of permutations for which the routing can be done in a similar fashion. The conflict information is usually modelled by agraph called as *Conflict graph*.

Here we propose certain transformation rules which are called group transformations. These group transformations partition the N! permutations of an NxN base-line network into partitions such that the conflict graphs of permutations in each partition are isomorphic.

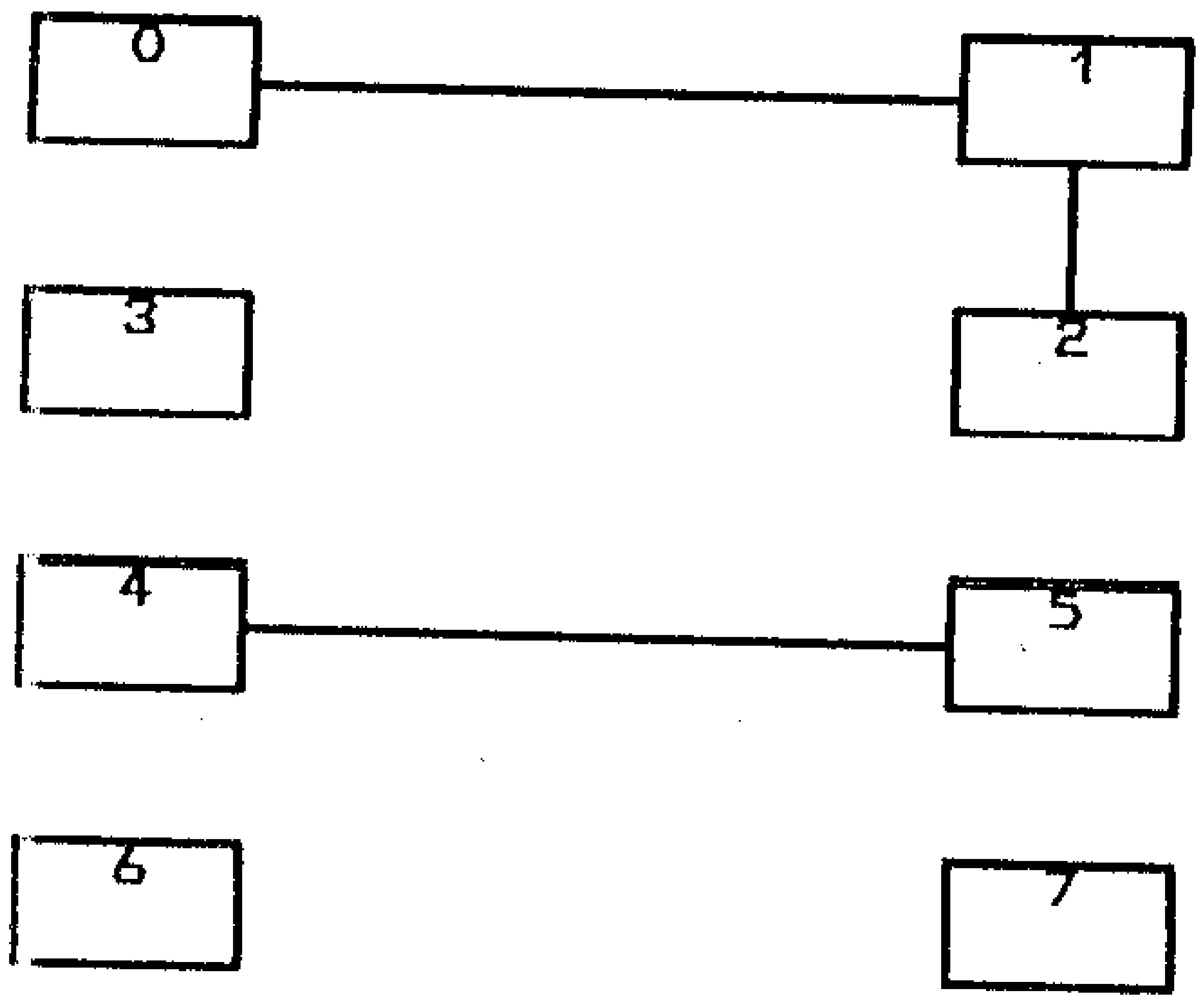
<u>Definition : 1</u>

A conflict graph G(V,E) - where V is the set of vertices and E is the set of edges, of a permutation P on a base-line network of size NxN is defined as follows :

The vertex set V consists of N vertices, each representing an input terminal of the network. The vertices are represented from 0 through (N-1). An edge e exists between vertices V_1 and V_2 iff the paths emerging from V_1 and V_2 are in conflict with each other in some link.

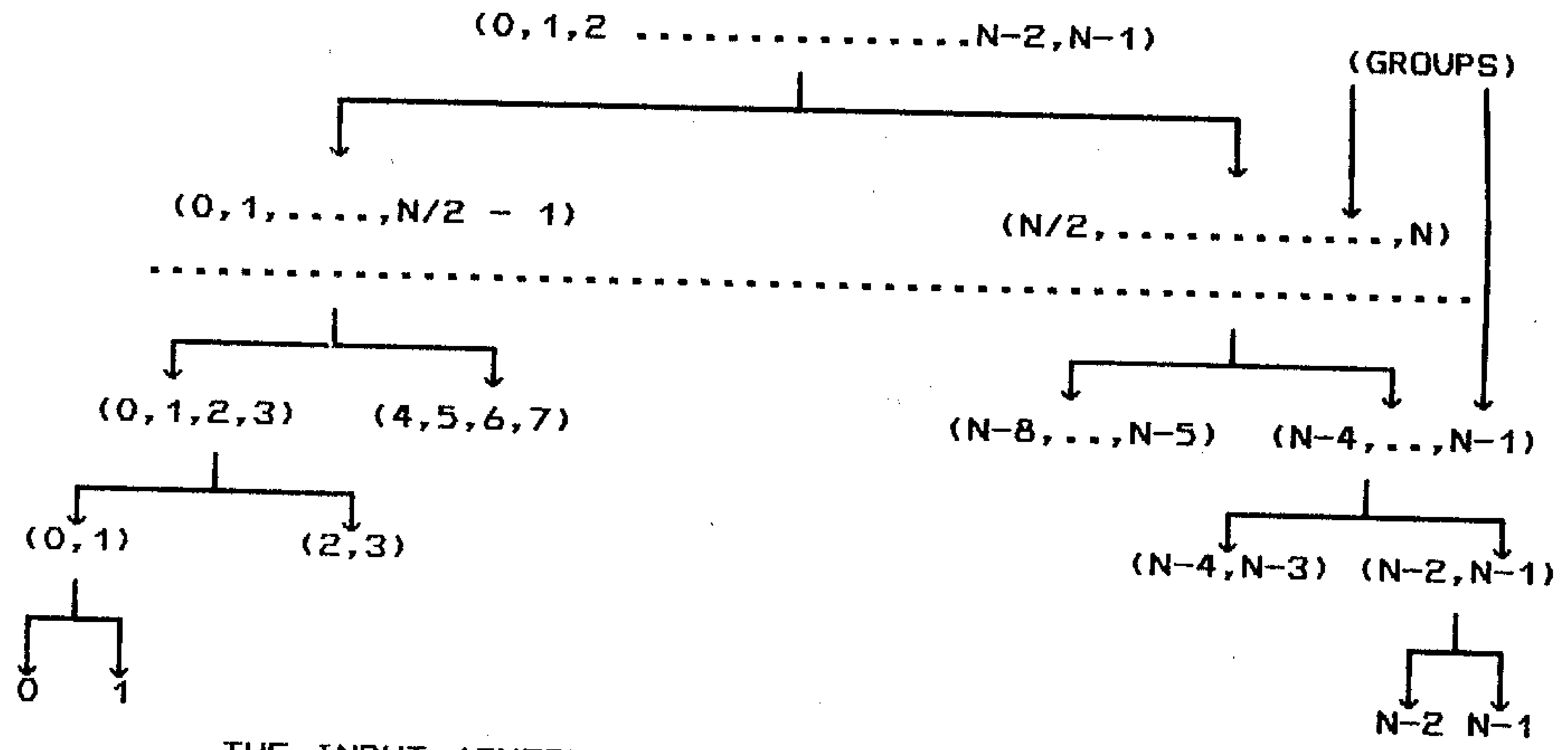
<u>Example 1 :</u>

The conflict graph of the permutation P : (7 5 4 2 1 0 6 3) is shown in the figure 6.



CONFLICT GRAPH FOR THE PERMUTATION IN EXAMPLE 1.
FIGURE 6.

For an NxN base-line network, the inputs (outputs) are grouped in different levels as shown in the figure 7. The size of any group at level-i are said to be adjacent if they have the same parent at the (i+1)th level.



THE INPUT (OUTPUT) GROUPS AT DIFFERENT LEVELS
FIGURE 7.

Definition 2 : A group-interchange $t_X(j:x)$, (where the subscript 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 the jth level, whose least element is x, by the following rule $k \longleftrightarrow (k+2)^j$, where $x <= k < (x+2)^j$. This process generates another permutation P_1 and is denoted by $t_X(j:x)[P] \longrightarrow P_1$.

Example 2 : Consider a permutation P : (7 2 6 4 0 3 1 5) and the group-interchange $t_I(1:4)$ on the input and let $t_I(1:4)[P] \longrightarrow P_1$, then P_1 is obtained from P by interchanging the input pairs $4 \longleftrightarrow 6$ and $5 \longleftrightarrow 7$. Thus P_1 : (7 2 6 4 1 5 0 3).

Similarly, a group-interchange on output $t_O(2:0)$ applied on P generates a permutation P_2, which is obtained from P by interchanging the output pairs $0 \longleftrightarrow 4$, $1 \longleftrightarrow 5$, $2 \longleftrightarrow 6$ and $3 \longleftrightarrow 7$. Thus P_2 : (3 6 2 0 4 7 5 1).

The effect of group-interchanges on inputs (outputs) may or may not be duplicated by group-interchanges on outputs (inputs). A group interchange at the jth level interchanges 2^j elements (inputs or outputs). Furthurmore for every such interchanging pair (p,q), p and q lie at unit hamming distance, differing in the jth bit (from the right end). Let P_1 be a permutation which is to be transformed to another permutation P_2, such that the input (output) X_1 of P_1 is replaced by input (output) X_2 in P_2. Then for each j, where X_1 and X_2 differ inthe jth bit, a group interchange only on inputs (outputs) at level j will accomplish the desired transformation.

<u>Example 3 :</u>

To replace the output 7 by 4 in P of example 2, we have to apply group-interchange at levels 0 and 1 since 4(100) and 7(111) differ in 0-th and 1st bit positions, i.e.

$$P \xrightarrow{t_o(0:6)} (6\ 2\ 7\ 4\quad 0\ 3\ 1\ 5) \xrightarrow{t_o(1:4)} (4\ 2\ 5\ 6\quad 0\ 3\ 1\ 7).$$

<u>Definition 3 :</u>

Given a permutation P, an input (output) cluster $C_i(o)(\overset{(j,x)}{\cancel{x,j}})$ defines the set of inputs (outputs) corresponding to the outputs (inputs) of the group at levelj whose least output (input) is x.

For the permutation $\hat{P}$: (2 7 4 6 1 5 0 3), $C_o(\overset{2,6}{\cancel{0,2}})$={2,7,4,6} and $Ci(\overset{2,4}{\cancel{4,2}})$={1,2,3,5}. The set of input (output) clusters at a level j denoted by $S_{Ci(o)}(j)$ is the set of all input (output) clusters at level j.

For $\hat{P}$, $S_{Co}(1) = \{(2,7),(4,6),(1,5),(0,3)\}$.

Definition 4 :

A group-transformation T is a sequence of group interchanges on inputs followed by sequence of group interchanges on outputs.

Group-transformation defines an equivalence partition an set of all permutations. Note that if a permutation p' is derivable from another permutation P by applying some group transformation T,i.e., if $T[P] \longrightarrow P'$, we will say that P'~ P (P' is related to P) and it is easy to see that '~' is an equivalence relationship.

Definition 5 :

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

Definition 6 :

Given a closure set C of any permutation P, we define the seed permutation or simply the seed of set C as the lexicographical minimum permutation of C.

Theorem 1 : The conflict graphs of all permutations in any closure set are isomophic.

Proof : See [NBJ-90].

Theorem 2 : In an N x N base-line network, given a permutation and its closure set C. Then, #C $\geq 2^{N-1}$.

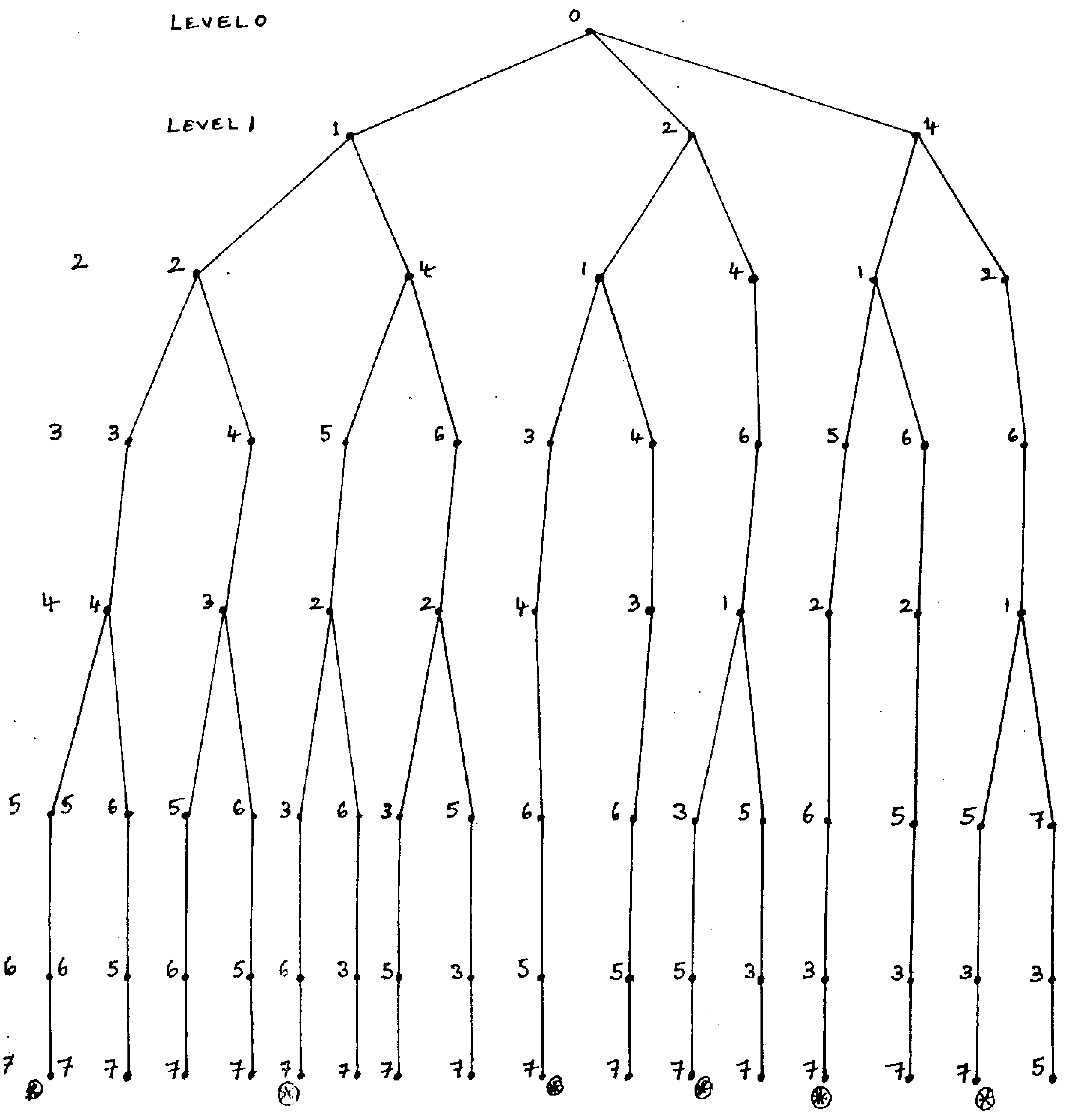
Proof : See [NBJ-90].

Theorem 3 : There is a single, unique seed in a closure set C of any permutation P.

Proof : Let C be the closure set of any permutation P. Let if possible, S_1 and S_2 be two seeds in C. Since S_1 and S_2 belong to the same closure set there are group transformations from S_1 to S_2 and S_2 to S_1. Since lexicographical ordering is a complete ordering if $S_1 \neq S_2$ either $S_1 < S_2$ or $S_2 < S_1$. In case $S_1 < S_2$, S_2 is not a seed and similarly if $S_2 < S_1$ then, S_1 is not a seed.

Thus there cannot be more than one seed in a closure set.

From Theorem 2, we have an estimate of the cardinality of the closure set of a permutation P. This is a large number. Thus by means of group transformations we are able to map the entire closure set onto a single seed. To have an idea the seeds for N = 8 are given in the figure 8. There are 16 seeds for N = 8. The total number of permutations is 8! = 40,320. Thus we are able to map the 40,320 permutations into just 16 permutations. If we are able to route the seed of a closure set then we can also optimally route the permutations which belong to that class. The optimal routing of the BP's is known. All BP's are also seeds. Hence the optimal routing for the BPCL is known. For an N x N base-line network the total number of permutations in BPCL is $\geq n!.2^{N-1}$.



SEEDS FOR N=8. REPRESENTED AS A TREE

FIGURE 8.

SEEDS MARKED BY ⊛ ARE BP'S. THERE ARE 6 OF THEM.

CHAPTER FOUR : ALGORITHMS & CONCLUSIONS.

In the previous chapter we have seen the importance of the seeds. In this chapter we present two algorithms. Given a permutation P the first algorithm generates the seed of the closure set of P. The second algorithm generates all possible seeds for an N x N base-line network.

We use the following definitions and notations in the algorithms to follow.

In a permutation the position corresponding to an input i is referred to as inp(i) position.

The output corresponding to an input i is represented as O(i) and similarly the input corresponding to an output j is represented as I(j).

<u>Definition 1 :</u>

Let p, q be two integers whose bit representations are $(p_{n-1} \ldots p_0)$ and $(q_{n-1} \ldots q_0)$ respectively. The distance between p and q is defined as the maximum bit position at which p and q differ, when they are expressed in bit form.

<u>Example 1 :</u>

Distance between 4 (100) and 7 (111) is 1, since they differ at the zeroth and the first bit positions only.

Definition 2 :

The distance vector of an output cluster not containing 0, $C_o(i,x)$ is a vector of length (2^i). It is the ordered set of the distances of the (2^i) elements of the cluster with respect to the zero.

Example 2 : Consider a permutation P = (0 3 2 5 7 6 4 1), the distance vector of the cluster Co(0,2) is {2,1,1,2}

Definition 3 :

A distance vector $V = \{v_1, v_2, \ldots, v_n\}$ is said to be greater than another distance vector $U = \{u_1, u_2, \ldots, u_n\}$ iff $v_i = u_i$ for i = 1 ... k-1 and $v_k > u_k$, where $k <= n$.

A distance vector $V = \{v_1, v_2, \ldots, v_n\}$ is said to be equal to another distance vector $U = \{u_1, u_2, \ldots, u_n\}$ iff $v_i = u_i$ for i = 1 ... n.

A distance vector V is said to be less than another distance vector U if V is not greater than U and V is not equal to U.

The following observations are inportant in view of the algorithm.

1. Any group interchange on inputs does not alter the output clusters.

2. Any group interchange on the outputs does not alter the distance vector of the output clusters.

3. The distances of all elements in an output group will be the same with respect to zero.

In order to minimize any output cluster C we do the following :

Perform group interchanges on inputs in C so as to miminize the distance vector of C.

Perform the group interchanges on the outputs so as to minimize the elements of C.

<u>Example 3</u> : Consider the permutation P : (0 3 1 6 2 7 4 5). P is not a seed. The seed can be generated as follows :

Since among the four pairs (0,3), (1,6), (2,7), (4,5), the distance between 4 and 5 is the least, placing it at inp(0) and inp(1) positions leads to a smaller permutation as shown :

$$P : (0\ 3\ 1\ 6\quad 2\ 7\ 4\ 5) \xrightarrow{t_i(1,4),\, t_i(2,0)} P_1 : (4\ 5\ 2\ 7\quad 0\ 3\ 1\ 6)$$

Making O(0) as 0 by $t_o(2,0)$ we get P_2 : (0 1 6 3 4 7 5 2). Now in P_2 the distance vector of $C_o(1,2)$ is found to be V = (2,1). Since in the elements of V, 1 is the least we perform the transformation $t_i(0,2)$ on P_2 giving P_3 : (0 1 3 6 4 7 5 2). Now in P_3 the cluster $C_o(1,2)$ can be minimized furthur only by the output group interchanges. Hence the distance vector of $C_o(1,2)$ i.e.(1,2) cannot be changed, hence we can replace 3 by 2 to make P_3 lesser. By applying $t_o(0,2)$ on P_3 we get P_4 : (0 1 2 6 4 7 5 3). Similarly 6 can be replaced by 4 by the transformations $t_o(1,4)$ giving P_5 : (0 1 2 4 6 5 7 3). Continuing similarly we get the seed S : (0 1 2 4 3 6 5 7) by applying the transformations $t_i(0,6)$, $t_i(1,4)$, $t_o(0,6)$ on P_5.

We now describe the algorithm to generate the seed of a closure set of a given permutation P.

ALGORITHM 1 : GET-SEED

INPUT : Any permutation P.

OUTPUT : Seed permutation of the closure set of P.

DATA-STRUCTURES UTILIZED :

Available list : This contains a list of ordered sets, where each set contains the maximal subgroup of input elements which are not yet utilized in the process of generating the seed. Initially the list contains a set {0 ... N-1}. Now, if 0 is utilized in the process of finding the seed then the available list will break up the remaining output elements into the following sets :{1}, {2,3}, {4,5,6,7}, ... , {N/2, ... ,N-1}. Thus the avaiable list always contains the maximal output subgroups togeather. At this instant of time we say, in the above list 1, 2, 4, ... ,N/2 are the minimum of subgroup elements in the available list.

The algorithm uses a stack for storing the intermediate permutations, S. For each permutation there is an associated available list and a pointer called decided_till which indicates the position till where this permutataion has been developed. The stack S is special in the sense that it always hold the permutations all of which are equal till the position marked by decided_till. If a new permutation which is greater than the permutations in the stack, it is discarded and if it is lesser than the permutations of the stack, it is retained while the other permutations in the stack are discarded.

The procedure getseed presented is a recursive one. It takes i, the starting position of the output cluster to be developed.Before the algorithm is called the stack has to be initialized with the candidate permutations which have been decided till the inp(1) position. The initialization is done by the procedure initialize given below. It takes the input permutation P as a parameter. The getseed procedure is called with i = 2.

```
Procedure initialize(P)
Begin
  Find the list L, distances of all pairs of elements in $SC_o(1)$.
  For all pairs (x,y) in $SC_o(1)$, whose distance is minimum in L, do
  Begin
    Perform group interchanges on inputs to get (x,y) to inp(0) and
          inp(1) positions respectively.
    With (x,y) and (y,x)  at the inp(0) and inp(1) positions
          respectively do
    Begin
      Make O(o) ~~the output at the inp(0) position~~ as 0.
      Make O(1) ~~the output at the inp(1) position~~ as a minimum of the
          subgroup element.
      Set the decided_till pointer as 1 and push the permutation
          into the stack.
    End
  End
End.


Procedure Getseed(i)
Begin
```

```
if i is equal to N-1 then terminate.
While a permutation P exists in the stack with decided_till
        pointer d less than (i - 1) do
Begin
  Find the maximal output cluster $C_o(i,d+1)$, starting with d+1.
  Find the distance vector V, of $C_o(i,d+1)$.
  For all elements of V, whose value is minimum dm, and the
        corresponding output is x do
  Begin
    Get x to the inp(d+1) position.
    Make the element at the inp(d+2) position as the minimum
        element of the subgroup containing it in the avaiable
        list.
    Push this permutation in the stack.
  end
  ~~end~~
end
  ~~Getseed(2*i)~~
 GetSeed (2*i)
 ~~end~~
end.
```

ALGORITHM 2 : GENERATE-SEEDS.

INPUT : N.

OUTPUT : All possible seed permutations for the N x N base-line network.

DATA-STRUCTURES UTILIZED :

Available-list : This is the list as described in the algorithm 1.

Here we generate the possible candidates for the seeds and then run the first algorithm on these candidates to detect the actual seeds.

The following observation is important in view of the algorithm.

1. In a seed the elements starting from O(0),O(1), ,O(N-1) will always be minimum elements of a subgroup in the available list at that instant of time.

Example 4 : Consider the generation of all possible seeds for a 4x4 base-line network. Initially the available list consists of {0,1,2,3}. Now 0 is the only minimum element of the available list. Hence the position inp(0) of all seeds will have 0. The avaialble list breaks up into {1},{2,3}. Hence the position inp(1) can be taken up by 1 or 2, giving the partial candidate permutations P_1 : (0 1 ..) and P_2 : (0 2 ..) .

Starting with P_1, we have the available list as {2,3}. Hence position inp(2) can be taken up by 2 alone, leading to the partial candidate permutation P_3 : (0 1 2 .). Now the available list has a single element {3}. Hence the position inp(3) in P_3 can be taken up by 3 giving the first candidate C_1 : (0 1 2 3).

Starting with P_2, we have the available list as {1},{3}. Proceeding as above we get two more candidate seeds C_2 : (0 2 1 3) and C_3 : (0 2 3 1). By running the algorithm getseed on the candidates we find that only C_1 and C_2 are the seeds.

In the actual implementation we have used several filtering techniques so as to minimize the candidate permutations.

Results :

The seeds for a 8 x 8 base-line network by the second algorithm, 17 candidates were generated by the second algorithm the first algorithm gave 16 of these as seeds. The candidates and the seeds are shown in figure 9. We have made an attempt to generate all possible seeds for N = 16. Although the complete set of seeds could not be generated we can have a fairly good estimate of the total number of seeds for N = 16. The partial results are shown in the figure 10 in a tabular form. In figure 10, the seeds are grouped according to their starting sequence.

Conclusions :

Furthur sudies in this area may be done for mapping a non-BP seed into a BP such that the conflict graph of the BP is a minimal super-graph of that of the non-BP seed. Then the routing for the non-BP seed can be done similar to the BP seed to which it is mapped. Conjecture is that this gives an optimal routing for the non-BP seed as well. All these ideas can be extended to the other blocking, full-access, unique path MIN's.

S.NO.	CANDIDATES GENERATED BY ALGORITHM 2								
01	0	1	2	3	4	5	6	7	
02	0	1	2	3	4	6	5	7	
03	0	1	2	4	3	5	6	7	
04	0	1	2	4	3	6	5	7	
05	0	1	4	5	2	3	6	7	
06	0	1	4	5	2	6	3	7	
07	0	1	4	6	2	3	5	7	
08	0	1	4	6	2	5	3	7	
09	0	2	1	3	4	6	5	7	
10	0	2	1	4	3	6	5	7	
11	0	2	4	6	1	3	5	7	
12	0	2	4	6	1	5	3	7	
13	0	2	4	6	1	7	3	5	*
14	0	4	1	5	2	6	3	7	
15	0	4	1	6	2	5	3	7	
16	0	4	2	6	1	5	3	7	
17	0	4	2	6	1	7	3	5	

CANDIDATES AND SEEDS FOR N = 8.
CANDIDATE 13 IS FOUND TO BE A NON-SEED
BY THE ALGORITHM 1.

FIGURE 9.

S.no	Starting sequence				Candidates	Seeds
01.	0	1	2	3	2796	1186
02.	0	1	2	4	4961	
03..	0	1	2	8	9102	
04.	0	1	4	5	3624	
05.	0	1	4	6	5956	
06.	0	1	4	8	15552	
07.	0	1	8	9	5064	
08.	0	1	8	10	9092	
09.	0	1	8	12	16326	
10.	0	2	1	3	0952	0418
11.	0	2	1	4	2064	1287
12.	0	2	1	8	3855	2154
13.	0	2	4	6	4027	
14.	0	2	4	8	11257	
15.	0	2	8	10	6471	
16.	0	2	8	12	15543	
17.	0	4	1	5	0492	0184
18.	0	4	1	6	0898	0249
19.	0	4	1	8	2639	1600
20.	0	4	2	6	1156	0147
21.	0	4	2	8	4027	1139
22.	0	4	8	12	6094	
23.	0	8	1	9	0282	0099
24.	0	8	1	10	0540	0134
25.	0	8	1	12	1014	0184
26.	0	8	2	10	0642	0074
27.	0	8	2	12	1284	0121
28.	0	8	4	12	1080	0068

CANDIDATES AND SEEDS FOR N = 16.

FIGURE 10.

REFERENCES :

1. Nanbanita Das, Bhargab B Bhattacharya and Jayasree Dattagupta "Isomorphism of conflict graphs in MIN and its applications in optimal routing" IEEE Region 10 Conference on Computer and Communication systems (TENCON-90) Hong Kong-90. [NBJ-90]

2. Chang-Lin Wu and Tse-Yun Feng "On a class of multistage interconnection networks" IEEE Trans. Comput., Vol C-29, pp 694-702, Aug 1980. [CT-80]

3. C S Raghavendra and Anujan varma "Fault tolerant multiprocessors with redundant path interconnection networks" IEEE Trans. Comput., Vol C-35, No. 4, pp 307-316, April 1986. [RA-86]

4. M A Abidi and D P Agrawal "On conflict free permutations in multistage interconnection network" J. Digital system, Vol. V, pp 115-136, 1980. [AA-80]