Available online at www.sciencedirect.com
SCIENCE DIRECT®

Theoretical Computer Science 352 (2006) 197–214

Theoretical Computer Science

www.elsevier.com/locate/tcs

Rearrangeability of bit permutation networks

Xuewen Bao[a,*,1], Frank K. Hwang[b], Qiao Li[a]

[a] Department of Mathematics, Shanghai Jiaotong University, 1954 Huashan Road, Shanghai 200030, China
[b] Department of Applied Mathematics, National Chiaotung University, Hsinchu, Taiwan

Received 9 September 2004; received in revised form 16 September 2005; accepted 28 September 2005

Communicated by D.-Z. Du

Abstract

In this paper, we introduce the concept of routing grid as a tool for analyzing realizability of permutations on bit permutation networks (BPNs). We extend a result by Linial and Tarsi which characterizes permutations realizable on shuffle-exchange networks to any BPNs. A necessary condition for a BPN to be rearrangeable is given, and the rearrangeability of two families of BPNs are explored. Finally, we present a method which may help to tackle one kind of balanced matrix problems whose solution implies an answer to Benes conjecture. Hopefully, our treatment brings some new insight into the problem of permutation routing.

Keywords: Multistage interconnection network; Rearrangeability; Layered cross product; Permutation; Balanced matrix

1. Introduction

Multistage interconnection networks (MINs) have been an important interconnection scheme to realize permutations. A MIN is a staged network connecting a number of inputs and outputs, where each stage consists of a column of crossbars and consecutive stages are connected by left-to-right directed edges (or links). Unless otherwise stated, all networks considered here are MINs using $d \times d$ crossbars and having s stages and $N = d^n$ inputs (outputs). Crossbars will be treated as nodes, and input (output) lines do not appear in any drawing of networks. Two networks are topologically equivalent if there is an isomorphism between them which maps all nodes in a stage of the first network to those in the corresponding stage of the second network.

The first (last) node stage is called the input (output) stage. A request is a pair of input–output nodes. A permutation is a set of N requests in which each node in the input and the output stage appears exactly d times. A network can realize a permutation if there are N edge-disjoint paths connecting the N input–output pairs, and is rearrangeable if it can realize all permutations.

By an edge stage we mean the set of edges between two consecutive node stages. Sometimes, an edge stage may be regarded as a 2-stage network, and when more than one consecutive edge stage is considered together, it is regarded as a MIN consisting of all the involving edges and nodes. An s-stage network will be denoted by

* Corresponding author. Tel.: +86 591 22283288.
E-mail address: baoxw@163.com (X. Bao).
1 Partially supported by NNSFC Grant No. 10371075.

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2005.09.077

$G = (V_1, V_2, \ldots, V_s; E_1, E_2, \ldots, E_{s-1})$, where $V_1, V_2, \ldots, V_s$ and $E_1, E_2, \ldots, E_{s-1}$ are its node and its edge stages, respectively.

The interest of this paper is only in the ability of networks to realize permutations. Therefore, we do not differentiate between topologically equivalent networks. Our attention is restricted to bit permutation networks (BPNs) which were formally defined and studied by Chang et al. [11]. BPN is a very interesting and important class of networks. It is well-structured, and includes most of the intensively studied MINs such as the Omega equivalent networks and their extra-stage versions, the shuffle-exchange networks and Benes networks. Also, many other networks of interest use BPNs as basic building blocks, such as those networks derived by vertical stacking and by horizontal concatenating.

We consider the permutation routing problem in the whole class of BPNs. It is shown that any permutation routing problem on a BPN is equivalent to finding a matrix in which a number of column sets form balanced matrices. This result generalizes a theorem by Linial and Tarsi [18] which characterizes permutations realizable on shuffle-exchange networks.

An important tool of this paper is the concept of routing grid which is constructed easily from a layered cross product (LCP) decomposition of BPNs. It gives rise to the above unified formulation of all permutation routing theories on BPNs. In addition, a necessary condition is also derived from it, which states that any rearrangeable BPN with minimum number of stages is a concatenation of two Omega equivalent networks. For this reason, we introduce two families of $(2n-1)$-stage BPNs which are of special interest. Each family contains the shuffle-exchange network and Benes network as special cases, and their rearrangeability relies on the solution of one kind of balanced matrix problems. We present a method which may be helpful in tackling this problem. We also compare the method with Cam's in his attempt to prove Benes conjecture [9], and identify the incompleteness in his proof.

The rest of this paper is organized as follows. In Section 2, some basic definitions and preliminary results about BPNs, LCP and balanced matrices are given. Section 3 includes the definition of routing grid and the formulation of routing problems as balanced matrix problems. Section 4 gives a necessary condition for a BPN to be rearrangeable, and an analysis of the rearrangeability of two families of $(2n-1)$-stage BPNs. A method is presented in Section 5 which may help to tackle one kind of balanced matrix problems whose solution implies an answer to Benes conjecture. And Section 6 concludes the paper.

2. Basic definitions and preliminary results

2.1. Bit permutation networks

BPNs were first introduced as a class of MINs by Chang et al. [11], and further studied by a number of researchers [14,16,23]. In the following, we give a definition with minor modifications and list some results related to them.

Definition 1. Let G be an s-stage network. G is called a BPN if the nodes in every stage can be labelled by the d-nary numbers of length $n-1$ which admits the existence of $s-1$ permutations $p_1, p_2, \ldots, p_{s-1}$ on set $\{1, 2, \ldots, n-1\}$ and $s-1$ integers $a_1, a_2, \ldots, a_{s-1} \in \{1, 2, \ldots, n-1\}$, such that, for any i $(1 \leqslant i \leqslant s-1)$, node $(x_1, \ldots, x_{n-1})$ in stage i is adjacent to node $(y_1, \ldots, y_{n-1})$ in stage $i+1$ if and only if $y_j = x_{p_i(j)}$ for all j except for possibly $j = a_i$.

We can see from the definition that, if the bits in the labels of all nodes in V_i are permuted by p_i: $(x_1, \ldots, x_{n-1}) \to (x_{p_i(1)}, \ldots, x_{p_i(n-1)})$, then E_i connects a node in V_i to a node in V_{i+1} if and only if their labels are different at most at the a_ith bit. Obviously the Omega equivalent networks, the shuffle-exchange networks and Benes network all fit into this definition, therefore they are all BPNs. In [11], Chang et al. also proved that the permutations of bits, $p_1, p_2, \ldots, p_{s-1}$, can always be set to identity by wisely labelling the nodes. That is

Theorem 2.1. *G is a BPN of s-stage if and only if the nodes in every stage can be labelled by the d-nary numbers of length* $n-1$ *which admits the existence of* $s-1$ *integers* $a_1, a_2, \ldots, a_{s-1} \in \{1, 2, \ldots, n-1\}$ *such that, for any* i $(1 \leqslant i \leqslant s-1)$, *node* $(x_1, \ldots, x_{n-1})$ *in stage i is adjacent to node* $(y_1, \ldots, y_{n-1})$ *in stage* $i+1$ *if and only if* $y_j = x_j$ *for all j except for possibly* $j = a_i$.

The characterization of BPNs in Theorem 2.1 is equivalent to the definition of generalized butterfly networks in [14]. We use the sequence $(a_1, a_2, \ldots, a_{s-1})$ to represent the BPN described in Theorem 2.1. Such a sequence induces a partition of the set of edge stages $\{E_1, E_2, \ldots, E_{s-1}\}$ in which E_i and E_j belong to the same subset if and only if $a_i = a_j$. It was proved in [11] that two BPNs are topologically equivalent if and only if their sequences induce the same partition. Theorem 2.3 gives a characterization of BPNs which does not depend on the existence of a labelling system of nodes [23]. It is also a generalization of a theorem by Bermond et al. [6], a well-known result which first characterizes the Omega equivalent networks graph-theoretically. Another characterization of BPNs will be presented in Section 2.2 and more characterizations can be found in [14,23].

We use the symbol $G_{i,j}$ to denote the subnetwork of G induced by the nodes in V_i through V_j. Let π be a partition of the set $\{E_1, E_2, \ldots, E_{s-1}\}$, and let $t_{i,j} = |\{[E_i], [E_{i+1}], \ldots, [E_{j-1}]\}|$, where $[E_i]$ is the subset in π containing E_i and $|A|$ stands for the number of different elements in A. The following definitions are derived from [1,6,23].

Definition 2. Let $i \leqslant j$.

(i) G is said to satisfy the Banyan property if each pair of input and output nodes is connected by a unique path.
(ii) G is said to satisfy the $P(i, j)$ property if $G_{i,j}$ contains exactly 2^{n-j+i} connected components, and is said to satisfy the $P(*, *)$ property if it satisfies $P(i, j)$ for all possible ordered pairs (i, j).
(iii) G is said to satisfy the $P_\pi(i, j)$ property if $G_{i,j}$ contains exactly $d^{\,n-t_{i,j}}$ connected components, and is said to satisfy the $P_\pi(*, *)$ property if it satisfies $P_\pi(i, j)$ for all possible ordered pairs (i, j).
(iv) G is said to satisfy the buddy property if any edge stage consists of disjoint 4-cycles.
(v) G is said to satisfy the extended buddy property $Q(i, j)$ if for any $u, v \in V_i$, the sets of nodes in V_j reachable from u and v are either the same or disjoint. G satisfies $Q(*, *)$ if it satisfies $Q(i, j)$ for all possible ordered pairs (i, j).

Theorem 2.2 (*Bermond et al. [6]*). *G is topologically equivalent to the binary Omega network if and only if G satisfies the Banyan property and property $P(*, *)$.*

Theorem 2.3 (*Wu et al. [23]*). *G is a BPN if and only if there exists a partition π of $\{E_1, E_2, \ldots, E_{s-1}\}$ such that G satisfies properties $P_\pi(*, *)$ and $Q(*, *)$.*

When $d = 2$, $s = n$, and each equivalent class of π contains exactly one element, then the $P_\pi(i, j)$ and $P_\pi(*, *)$ property are reduced to the $P(i, j)$ and $P(*, *)$ property of Theorem 2.2, respectively.

In [23], it was also proved that two BPNs are topologically equivalent if and only if they have the same partition of edge stages. Simple analysis reveals that the partition in Theorem 2.3 and the partition induced by its sequences are identical.

By Theorem 2.3, the number of subsets in π is exactly $n - 1$ in an $N \times N$ connected BPN. When a BPN is considered, we always assume a proper labelling is already given such that the short notation of sequences in Theorem 2.1 is applicable. For example, we will use $(1, 2, \ldots, n-1)$ and $(1, \ldots, n-1, n-1, \ldots, 1)$ to denote Omega network and Benes network, respectively. In drawing a figure of any BPN, we always assume the direction of edges are from left to right and the nodes are implicitly labelled by the d-nary numbers of length $n - 1$ in the natural order. So the nodes in the first row have label $00\cdots0$ and the nodes in the last row have label $11\cdots1$.

2.2. Layered cross product

Definition 3 (*Even and Litman [12]*). Let $G' = (V'_1, V'_2, \ldots, V'_s; E'_1, E'_2, \ldots, E'_{s-1})$ and $G'' = (V''_1, V''_2, \ldots, V''_s; E''_1, E''_2, \ldots, E''_{s-1})$ be two s-stage networks. Their LCP, $G' \times G''$, is an s-stage network $G = (V_1, V_2, \ldots, V_s; E_1, E_2, \ldots, E_{s-1})$, where $V_i = V'_i \times V''_i$ and $((u', u''), (v', v'')) \in E_j$ if and only if $(u', v') \in E'_j$ and $(u'', v'') \in E''_j$, $1 \leqslant i \leqslant s$, $1 \leqslant j \leqslant s-1$.

The definition above can be extended for the LCP of any number of factors recursively by letting $G_1 \times G_2 \times G_3 = (G_1 \times G_2) \times G_3$.

Lemma 2.4. *Up to network isomorphism, LCP is associative and commutative.*

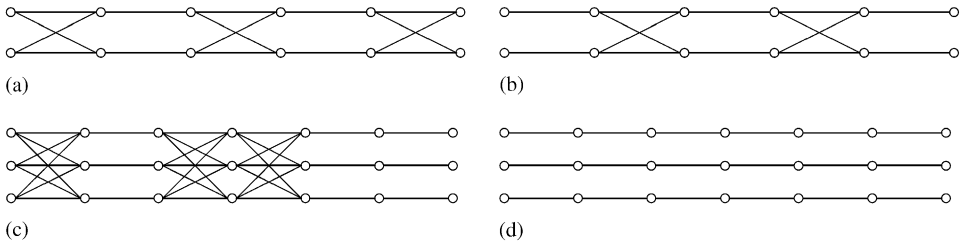


Fig. 1. (a) a 2-track with 3 crosses; (b) a 2-track with 2 crosses; (c) a 3-track with 3 crosses; (d) a 3-track without cross.

Definition 4. A d-track is a multistage interconnection network consisting of d parallel paths of equal length in which some edge stages are identical to the complete bipartite graph $K_{d,d}$.

Sometimes a d-track is referred to as a track for short. We also use the term *cross* to refer to an edge stage identical to $K_{d,d}$ in a track. Fig. 1 shows two 2-tracks and two 3-tracks.

The following result was proved for the binary case in [14].

Theorem 2.5. *A MIN is a BPN with $N = d^n$ inputs/outputs if and only if it is the LCP of $n - 1$ d-tracks subject to the condition that, in each edge stage, there is one and only one track which has a cross.*

Proof. Let G be the LCP of $n - 1$ d-tracks. For each track, label all nodes on the ith path by number i, $0 \leqslant i \leqslant d - 1$. Then any stage of G contains d^{n-1} nodes which are labelled by the d-nary numbers of length $n - 1$. Consider an edge stage E_j of G. If the cross is on the ith track in this edge stage, then there is an edge in E_j between two nodes if and only if the labels of these two nodes differ at most at bit i. Thus G is a BPN.

Conversely, if G is an s-stage BPN with sequence $(b_1, b_2, \ldots, b_{s-1})$, let the ith track has a cross in the jth edge stage whenever $b_j = i$, $1 \leqslant j \leqslant s - 1$, $1 \leqslant i \leqslant n - 1$. Then, in each edge stage, there is one and only one track which has a cross. It can be easily checked that G is the LCP of these $n - 1$ tracks. □

If a track has no cross, then it has no more edges than those on the d parallel paths. It is obvious that, if G is decomposed into the LCP of $n - 1$ tracks and t of them have no cross, G has exactly d^t connected components. From the proof above, one can also see that the partition of edge stages of a BPN can be determined by letting the cross stage indices on each track comprise a subset. Fig. 2 shows a binary BPN and its LCP decomposition as tracks.

By the definition of LCP, every path in G is the product of $n - 1$ paths, respectively, in the $n - 1$ tracks of the decomposition, and vice versa. The work of this paper is primarily based on Theorem 2.5 and the following lemma, which can be easily verified by the definition of LCP.

Lemma 2.6. *Let G be a BPN with N inputs/outputs. A set of N paths in G, all going from the input stage to the output stage, are edge-disjoint if and only if the following condition holds*:

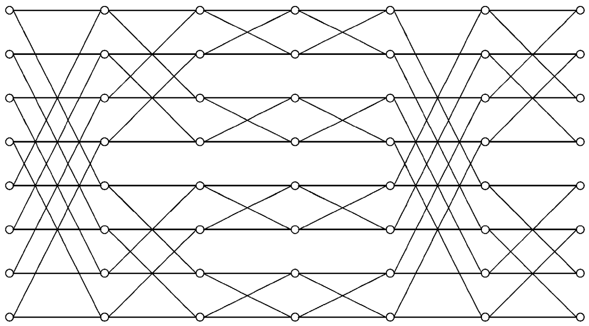
No two paths, when viewed as the product of $n - 1$ paths in the $n - 1$ tracks, share the same combination of track edges in any edge stage.

2.3. Balanced matrices

Let P be an $N \times n$ matrix, whose entries are numbers from the set $\{0, 1, \ldots, d - 1\}$. We call P a *p-matrix* if all the rows of P are different. A p-matrix can be considered as a permutation p on the set $\{0, 1, \ldots, N - 1\}$ if the ith row $(0 \leqslant i \leqslant N - 1)$ is interpreted as the d-nary number of $p(i)$. We denote by I the *identity p-matrix* which corresponds to the identity permutation.

Definition 5. Let M be an $N \times k$ matrix, whose entries are numbers from the set $\{0, 1, \ldots, d - 1\}$. M is balanced if

(1) for $k \leqslant n - 1$, M is obtained from a p-matrix by deleting $n - k$ columns;
(2) for $k \geqslant n$, every consecutive n columns form a p-matrix.



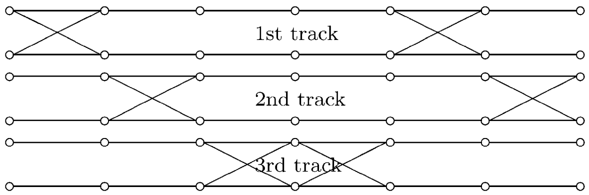


Fig. 2. (a) BPN (1,2,3,3,1,2); (b) the LCP decomposition.

Obviously, a balanced matrix of $k(\leqslant n)$ columns has exactly d^k different rows and each different row repeats exactly d^{n-k} times. When requests of a network are considered as from input lines to output lines, a permutation of the network can be represented by a pair of p-matrices, $\{P, Q\}$, where the ith row of P and Q represents the ith request, $0 \leqslant i \leqslant N-1$. Note that P can be assumed to be the identity p-matrix I, for otherwise we could change the row order of P and Q until P becomes I.

Linial and Tarsi [18] first established the relationship between balanced matrices and the permutation routing problem on shuffle-exchange networks by the following theorem.

Theorem 2.7. *A permutation $\{P, Q\}$ can be realized by the s-stage $(s \geqslant 1)$ shuffle-exchange network if and only if there exists an $N \times (s+n)$ balanced matrix M such that P and Q are, respectively, the first and the last n columns of M.*

For $k \geqslant 0$, Theorem 2.7 is equivalent to saying that an $(n+k)$-stage d-nary shuffle-exchange network can realize permutation $\{P, Q\}$ if and only if there exists an $N \times (2n+k)$ balanced matrix (P, M, Q), where M is $N \times k$. They also made the following conjecture.

Conjecture 1. *For any two d-nary p-matrices P and Q, there exists an $N \times (n-1)$ matrix M such that (P, M, Q) is balanced.*

We would like to make the following conjecture which extends the above one.

Conjecture 2. *For any two d-nary p-matrices P and Q, there exists a matrix M of k columns such that both (P, M) and (M, Q) are balanced, where $1 \leqslant k \leqslant n-1$.*

If Conjecture 2 holds, then it holds for any balanced matrices P, Q and any integer $k \geqslant 1$. If the number of columns of P (Q) is less than n, we first extend P (Q) to a p-matrix by adding some columns to the left (right). If the number of columns is greater than n, we just delete some columns of P (Q) on the left (right) to get a p-matrix. After M is computed, we delete (add) the columns formerly added (deleted). Further, if $k \geqslant n$, we first find a matrix M_1 of $k-n+1$ columns such that (P, M_1) is balanced (always possible), then find a matrix M_2 of $n-1$ columns such that (P, M_1, M_2) and (M_2, Q) are balanced. Then (M_1, M_2) is the desired k-column matrix.

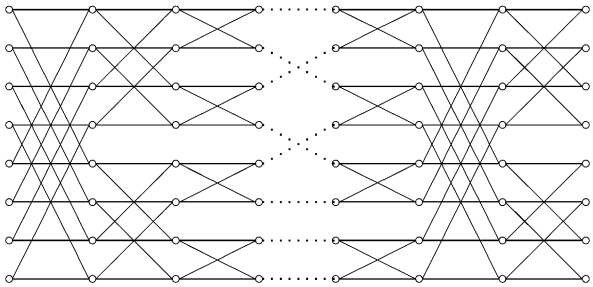


Fig. 3. A network in $\Omega \oplus \Omega$ but not in $\Omega \otimes \Omega$.

The binary case of Conjecture 1 is equivalent to the long standing conjecture of Benes [4], which states that the binary $(2n-1)$-stage shuffle-exchange network is rearrangeable. Benes conjecture has attracted the attention from many researchers. There has been a very slow progress toward solving this problem. Several unsuccessful proofs appeared in the literature [13,17,21]. The most recent proof was by Cam [9], but it was pointed out [3] that the proof is also incomplete. Conjecture 1 is a special case of Conjecture 2 for $k = n - 1$. In Section 5 we will have a partial discussion of Conjecture 2 by presenting a method which may help to solve it. The results in Section 3 show that balancedness is an essential issue in the permutation routing problem on any BPNs.

The following lemma is a slight extension of a result in [18] and will be used repeatedly in Section 4.

Lemma 2.8. *Let P and Q be two balanced matrices. Then there exists a column vector v such that both (P, v) and (v, Q) are balanced.*

Proof. First we assume both P and Q have $n - 1$ columns. Define a bipartite graph $H = (V_1, V_2; E)$, where V_1 and V_2 consist of d^{n-1} nodes labelled by the d-nary numbers of length $n-1$, and for every i, $0 \leqslant i \leqslant N-1$, $(x_1, \ldots, x_{n-1}) \in V_1$ is adjacent to $(y_1, \ldots, y_{n-1}) \in V_2$ if and only if $(x_1, \ldots, x_{n-1})$ is the ith row of P and $(y_1, \ldots, y_{n-1})$ is the ith row of Q. Then E contains exactly N edges and each edge corresponds to a row number i, $0 \leqslant i \leqslant N - 1$. Since P, Q are balanced, H is a d-regular multigraph. By König's Theorem [5], the edges of H can be colored using $0, 1, \ldots, d - 1$. Let the ith row of v be the color number of the ith edge, $0 \leqslant i \leqslant N - 1$. Then (P, v) and (v, Q) are balanced.

If P (Q) has more than $n - 1$ columns, delete some columns on the left (right) of P (Q) so that the resulting matrix has exactly $n - 1$ columns. If P (Q) has fewer than $n - 1$ columns, add some columns to it so that the resulting matrix is balanced and has exactly $n - 1$ columns. Use the new matrices to determine v, and then add (delete) the columns which were deleted (added) earlier. □

Lemma 2.8 suggests that Conjecture 2 holds for $k = 1$. We denote the procedure in the proof of Lemma 2.8 by a function LOOP$(P, Q; v)$. After reading Sections 3 and 4, one will find out that LOOP$(P, Q; v)$ in the binary case works quite similarly to one step of the looping algorithm [2,20]. So we call it a *looping operation*.

2.4. $\Omega \oplus \Omega$ and $\Omega \otimes \Omega$ networks

Concatenation is a common method to get larger network from smaller ones. In concatenating two networks, we identify the nodes in the last stage of the first network with those in the first stage of the second network. In many cases this is done with respect to a specific drawing of the networks, so the order of nodes to be identified in both stages is not explicitly specified. In the general setting, however, concatenation is virtually a 1–1 correspondence between the two node stages. This correspondence matches the pairs of nodes to be identified. It should be noted that concatenation of two BPNs does not always result in BPNs [23].

Let $\Omega \oplus \Omega$ be the set of all concatenated networks of two Omega equivalent networks. And let $\Omega \otimes \Omega$ be the networks in $\Omega \oplus \Omega$ which are also BPNs. We use different notations because they are indeed not the same. As an example, let us see the network in Fig. 3, where each dotted line connects two nodes to be identified. It is in $\Omega \oplus \Omega$ but not in $\Omega \otimes \Omega$. This can be checked by Theorem 2.3 because it does not satisfy the extended buddy property.

It is not difficult to see that there are exactly $(n-1)!$ networks in $\Omega \otimes \Omega$, since there are $(n-1)!$ different ways to arrange $n-1$ crosses in the second half among all the track factors. The number of networks in $\Omega \oplus \Omega$ is still not known.

One nice feature of the Omega equivalent network is that it can be decomposed as the LCP of two complete binary trees, one with the root in the first stage and the other with the root in the last stage [12]. But, unfortunately, LCP may not preserve under concatenations. That means networks in the $\Omega \oplus \Omega$ class may not have a LCP decomposition into two factors which are concatenations of the binary trees. It is still not shown even for the smaller class $\Omega \otimes \Omega$.

One work that focused on the whole class $\Omega \oplus \Omega$ was done by Hu and Shen [15], in which an algorithm of complexity $N^4 \log N$ was presented for checking equivalence between networks in the class. Though a better performance algorithm was announced in [7] for checking equivalence in $\Omega \oplus \Omega$, it actually works in a smaller class, for the networks in the class was counted to $(n-1)!$ by the authors, which is the size of $\Omega \otimes \Omega$.

$\Omega \otimes \Omega$ is an important class of networks and has received intensive study in the literature. Many rearrangeability results and problems are related to this class. In Section 4, we will show that $\Omega \otimes \Omega$ contains all rearrangeable BPNs with minimum number of stages, a fact that narrows the range of candidates if one seeks rearrangeable BPNs for one purpose or another.

3. Permutation routing and balanced matrices

Theorem 2.7 states that the realizability of a permutation on an s-stage shuffle-exchange network is equivalent to the existence of a specific $(s+n)$-column balanced matrix M. M defines $s+1$ p-matrices, including P and Q. Since P and Q represent, respectively, the input and the output matrix, they are automatically p-matrices by the definition of a permutation. So Theorem 2.7 actually requires a total of $s-1$ new p-matrices. Note that the order in which the columns of M are arranged is not essential. We could arbitrarily arrange them but keep track of the $s-1$ sets of columns which form p-matrices. Of course, M contains the information about the paths on which each input–output request should take. This information would not be lost as long as we know the $s-1$ sets and the rule by which the paths are established.

Recall that we do not include input (output) lines in the networks of this paper. The nodes in the first (last) stage are the inputs (outputs), and a permutation of the network is a set of N requests in which each input (output) node appears exactly d times. Such a permutation can be represented by a pair of $N \times (n-1)$ balanced matrices, $\{P, Q\}$, where the ith row of P and Q represents the ith request, $0 \leqslant i \leqslant N-1$. This extra assumption is not a restriction, for we can always delete the last column from P and Q if $\{P, Q\}$ is the pair of p-matrices representing a permutation from input lines to output lines.

In this section, we show what is revealed by Theorem 2.7 on shuffle-exchange networks is actually a common characteristic of all BPNs. That is, we show that the realizability of a permutation on any s-stage BPN is equivalent to the existence of a specific $(s+n-2)$-column matrix in which $s-1$ sets of columns form p-matrices. These sets are determined by the topology of the BPN, and they contain the information of the paths on which all N requests are routed.

To see how permutation routing on a BPN is related to balanced matrices, let us take a small binary BPN as an example. Fig. 4(a) shows BPN $G = (1, 2, 1)$, Fig. 4(b) is a permutation to be realized on it. The permutation is represented as a 2-regular bipartite digraph, where the nodes in the left column are the inputs, and each edge represents a request. So the permutation is a pair of balanced matrices $\{P, Q\}$, where

$$P = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad Q = \begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

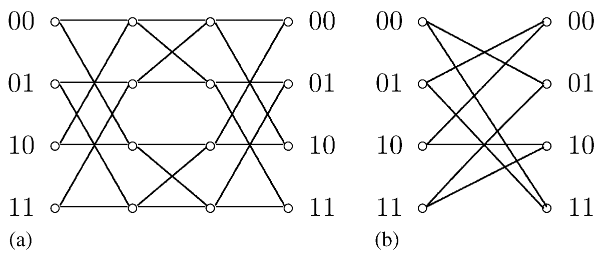


Fig. 4. (a) $G = (1, 2, 1)$; (b) a permutation.

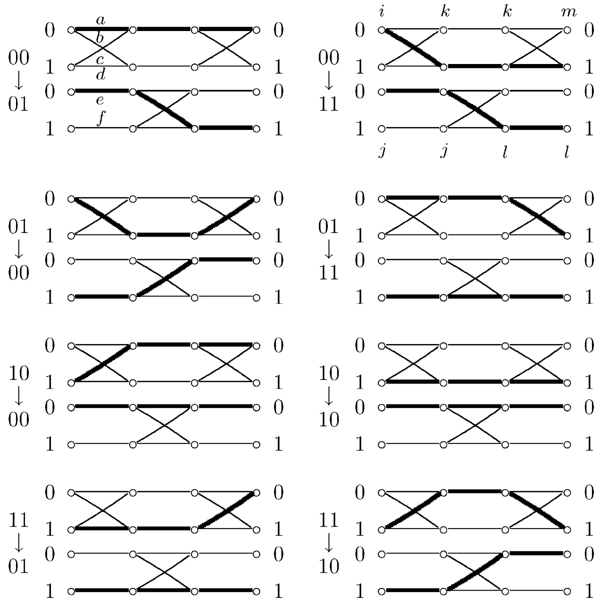


Fig. 5. Realizing the permutation on tracks.

To realize $\{P, Q\}$, we have to find 8 edge-disjoint paths in the network, each connecting one request of the permutation. As the network is the LCP of two 2-tracks, we draw 8 copies of its LCP representation in Fig. 5, each corresponding to one request (indicated on the left of each copy). Consider the first copy in Fig. 5. We want to realize the request $00 \to 01$ on it. This request can be further decomposed into two sub-requests. The first is the first-bit request of $00 \to 01$, that is, $0 \to 0$, which is to be realized on the upper track. The second is the second-bit request, $0 \to 1$, which is to be realized on the lower track. Once the two sub-requests are realized on respective tracks, the LCP of this two paths is the path in the original network, which connects $00 \to 01$. The LCP can be easily accomplished by combining the two paths bit-by-bit. For example, the thick lines on the two tracks show a possible realization of the two sub-requests, which read: $0 \to 0 \to 0 \to 0$ and $0 \to 0 \to 1 \to 1$. So $00 \to 01$ is realized by $00 \to 00 \to 01 \to 01$ in G.

To be edge-disjoint, no edge can be used more than once in any edge stage. Now consider only the first edge stage. There are 8 edges. They should all appear on the paths in the final realization. Note that these 8 edges are the "product edges" of the 4 edges on the upper track and the 2 edges on the lower track. This implies all possible combinations of the 4 upper-track edges and the 2 lower-track edges should appear in the 8 copies of Fig. 5. Let us give a symbol to each track edge as indicated in the first copy. The 8 possible combinations are $\{ae, af, be, bf, ce, cf, de, df\}$. The request $00 \to 01$ have used combination ae. One can check that the 8 copies have used each of the 8 combinations exactly once in the first edge stage (thick lines). Therefore, the realization in Fig. 5 contains no conflict in the first edge stage. Similarly, we can check that nor is there conflict in other edge stages. So Fig. 5 contains a legitimate routing for the permutation of Fig. 4(b).

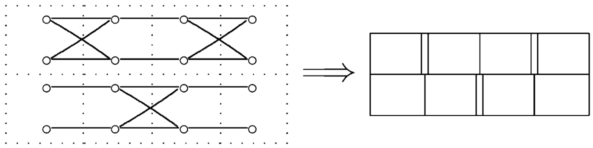


Fig. 6. Grid representation of (1,2,1).

Now let us analyze Fig. 5 further. Every request in the permutation of Fig. 4(b) is realized by a path decomposed as the LCP of two track paths in one copy of Fig. 5. A track path consists of four nodes, and each node is labelled by one bit. We still restrict to the first edge stage. In the first copy, the used track edges are $0 \to 0$ and $0 \to 0$, respectively, belonging to the upper and the lower track. In the second (right topmost) copy, the used edges are $0 \to 1$ and $0 \to 0$. And in the third copy, the used edges are $0 \to 1$ and $1 \to 1$. One may have noticed that the two bits of the used edge $i \to k$ on the upper track may or may not be identical, while the two bits of the used edge $j \to l$ on the lower track always remain the same. This is because the first edge stage of the upper track is a cross and that of the lower track consists of two parallel horizontal lines. Therefore, if we ignore the second bit l of the edges on the lower track, and put i, j, k together as a 3-bit number, (i, j, k), we get a characterization of the combination of used track edges in the first edge stage. Then, two copies of Fig. 5 use different combinations of track edges if and only if their 3-bit numbers are different. And the whole realization contains no conflict in the first edge stage if and only if the eight 3-bit numbers, when each is put in a row of an 8×3 matrix, form a p-matrix. Similar analysis applies to the second and the third edge stage.

Let the path on the upper track be $i \to k \to k \to m$, and the path on the lower track be $j \to j \to l \to l$. The symbols are indicated in the second copy of Fig. 5, where a different symbol is assigned only if a cross is passed. Now we create an 8×5 matrix M by putting the path information together as follows: the rows of M are labelled from 1 to 8, every copy of Fig. 5 corresponds to a row of M; the columns of M are labelled by i, j, k, l, m, and the bits on each path are placed in corresponding columns. Then we have

$$M = \begin{array}{c} \begin{matrix} i & j & k & l & m \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array}.$$

If the realization of Fig. 5 is edge-disjoint in the first, second and third edge stage, then the column sets $\{i, j, k\}$, $\{j, k, l\}$ and $\{k, l, m\}$ form p-matrices, respectively. One can check that M meets these requirements (the general case will be proved in Theorem 3.2). Note that $\{i, j\}$ are the input columns and $\{m, l\}$ are the output columns. On the contrary, if M is an 8×5 matrix satisfying the balance requirements above, we can take the reverse procedure and realize a permutation on the network of Fig. 4(a).

Let G be an s-stage $N \times N$ BPN. Then there are $n - 1$ tracks in the LCP decomposition of G. Let $\mathcal{G}$ be a grid of size $(n - 1) \times s$, that is, $\mathcal{G}$ consists of $(n - 1) \times s$ cells arranged in $n - 1$ rows and s columns. In $\mathcal{G}$, rows and columns are separated by line segments. Let the ith row correspond to the ith track of G, and the jth column correspond to the jth stage of the track, $1 \leqslant i \leqslant n - 1$, $1 \leqslant j \leqslant s$. If the jth edge stage of the ith track is a cross, we put a double vertical line segment to separate the jth and the $(j + 1)$th cells in the ith row, instead of a single vertical line segment. We call $\mathcal{G}$ the *grid representation* of G. It is not difficult to see that the grid representation of a BPN is unique, except that the rows of the grid can be interchanged. Fig. 6 shows the grid representation of the BPN in Fig. 4(a).

If a row of $\mathcal{G}$ is filled with one-bit numbers from $\{0, 1, \ldots, d - 1\}$, under the constraint that each cell contains one and only one bit and that different numbers are allowed to be adjacent in a row only if they are separated by a double vertical line, then the s bits in this row represent a path on the corresponding track. If all the rows are filled in this way,

(a)

c_1	c_3	c_3	c_5
c_2	c_2	c_4	c_4

(b)

c_1	c_4	c_4	c_4	c_4	c_8	c_8
c_2	c_2	c_5	c_5	c_5	c_5	c_9
c_3	c_3	c_3	c_6	c_7	c_7	c_7

Fig. 7. Routing grids of (1, 2, 1) and (1, 2, 3, 3, 1, 2).

$n-1$ paths are designated for the $n-1$ tracks, and thus a path in G is obtained by reading out the groups of $n-1$ bits in every column top-to-bottom and treating them as the node labels of G in respective stages.

In the above, if $\mathcal{G}$ is filled in the order from the cells in the first column to the cells in the last column, then each of the cells in the first column and the cells after a double vertical line can receive an arbitrary number. Any other cell gets the same number as in its preceding cell. So a total of $s+n-2$ arbitrary numbers are used. Generally, let c_j, $1 \leqslant j \leqslant s+n-2$, be $s+n-2$ variables each of which can take any value in $\{0, 1, \ldots, d-1\}$ independently. And let $S = \{c_1, c_2, \ldots, c_{s+n-2}\}$. We can fill $\mathcal{G}$ with these variables to get the general path in G. Let us formally define the procedure in the following:

1. The order to fill the cells is from the first column to the last column. And in each column, cells are filled from the first row to the last row.
2. When a new variable is needed, always choose the one with least index from the remaining set of S. Once a variable is chosen, it is deleted from S.
3. All cells in the first column use new variables. In other columns, new variables are needed for each cell after a double vertical line. Any other cell is filled with the same variable as in its preceding cell in the row.

We call $\mathcal{G}$ the *routing grid* of G when it is filled in the above way. The routing grid can be considered as a description of the general path in G, for it defines a concrete path if every variable is fixed to a value. Fig. 7(a) shows the routing grid of the BPN in Fig. 4(a), which gives the general path of the BPN $c_1c_2 \to c_3c_2 \to c_3c_4 \to c_5c_4$. Fig. 7(b) shows the routing grid of the BPN in Fig. 2.

In the routing grid $\mathcal{G}$, let A_i be the set of different variables in columns i and $i+1$, $1 \leqslant i \leqslant s-1$. Obviously, the sets A_i are determined by the topological structure of G. They are important in the analysis of permutation routing.

Lemma 3.1. $|A_i| = n$, $|A_j \cap A_{j+1}| = n-1$, *for* $1 \leqslant i \leqslant s-1$, $1 \leqslant j \leqslant s-2$.

Proof. In the first column, $n-1$ different variables are placed. In the second column, a new variable is introduced after the unique double vertical line, which stands for the unique cross in the first edge stage of all tracks. So the first two columns contain n different variables. When the two columns are shifted to the right by one column, there is always a new variable entering and an old variable leaving. Therefore, the number of different variables in any two consecutive columns remains constant, and every two sets with consecutive subscripts have exactly $n-1$ variables in common. □

As explained before, giving a value to every c_j results in a path in G through the routing grid. Conversely, any path in G (from input node to output node) can also be expressed by a sequence of $s+n-2$ variable values. Furthermore, two paths do not conflict in edge stage i if and only if the variable values in A_i differ in at least one c_j. This observation is important in formulating the permutation routing on a BPN into a balanced matrix problem.

Suppose there are N paths in G, each is from an input node to an output node. Let M be the $N \times (s+n-2)$ matrix in which entry (i, j) is the value of c_j on the ith path, $0 \leqslant i \leqslant N-1$, $1 \leqslant j \leqslant s+n-2$. Label the jth column of M by c_j. We call M the *routing matrix* of G.

Denote the variables in cell $(i, 1)$ and (i, s) of $\mathcal{G}$ by $c_{(i,1)}$ and $c_{(i,s)}$, respectively (these are the variables in the first and the last column of the routing grid). If a request is realized on G, the value of $c_{(i,1)}$ ($c_{(i,s)}$) is the ith bit of the input (output). If a permutation $\{P, Q\}$ is realized on G, $c_{(i,1)}$ ($c_{(i,s)}$) sequentially takes all the values in the ith column of P (Q). So in the routing matrix M, the column labelled by $c_{(i,1)}$ ($c_{(i,s)}$) is the ith column of P (Q). Note that for the ith path, the n variable values of A_k which are spread into two columns in the routing grid are in the same row i in M.

One can refer to the beginning part of this section for an example of routing matrix. The BPN and permutation is given in Fig. 4, and the routing grid is given in Fig. 7(a) (where variables i, j, k, l, m are used instead of c_1, c_2, c_3, c_4, c_5). Each copy in Fig. 5 gives a path in the BPN and corresponds to a row of the routing matrix M (given just before Fig. 5).

Theorem 3.2. *N paths in G are edge-disjoint if and only if the n columns (of M) in A_k form a p-matrix for any k, $1 \leqslant k \leqslant s-1$.*

Proof. If the N paths are edge-disjoint, by Lemma 2.6, no two paths share the same combination of track edges in any stage. So, for any k, the n columns of A_k form a submatrix in which no two rows are identical. This means the columns of A_k form a p-matrix. The other part of the proof is equally simple. □

From the definition of routing grid, we see that $c_{(i,1)} = c_i$ for $1 \leqslant i \leqslant n-1$. But the subscripts of $c_{(i,s)}$ are dependent on the structure of G. This implies P always occupies the first $n-1$ columns in M, but the columns of Q can be in different positions when G changes. As pointed out at the beginning of the section, the order in which the columns of M are arranged is not an essential problem. Theorem 3.2 holds even if the columns of M are interchanged, as long as the column labels are interchanged accordingly.

Theorem 3.3. *A permutation $\{P, Q\}$ is realizable on G if and only if there is an $N \times (s+n-2)$ matrix M in which the columns of P and Q appear in prescribed positions and the columns in A_k form p-matrix for any k, $1 \leqslant k \leqslant s-1$.*

One may find that there is a difference between Theorems 2.7 and 3.3 when applied to the shuffle-exchange network. It results partly from our assumption about permutation (it is a pair of $N \times (n-1)$ balanced matrices) and partly from the labelling scheme (Theorem 2.1) for the shuffle-exchange network. But this difference does not obscure the fact that the latter is a generalization of the former from the shuffle-exchange network to the BPN. To do so, the notion of the balanced matrix is further extended to the more general routing matrix.

4. Rearrangeability of BPNs

We have seen that the results in Section 3 are based on the LCP decomposition of Theorem 2.5. This characterization of BPNs casts new light not only on structural aspects of BPNs, but also on permutation routing on them. For example, we have shown that realizing a request on a BPN can be decomposed into realizing $n-1$ bits separately, with each bit to be routed on a separate track. Since adding an extra stage to the network is actually adding a cross to some track, the new stage only increases the flexibility to route the corresponding bit. If a track has no cross, the track is not connected. If it has t ($\geqslant 1$) crosses, then there are d^{t-1} ways to route any one-bit request, for there are d ways to pass each of the first $t-1$ crosses and only one way for the last cross. If there are $t_1, t_2, \ldots, t_{n-1}$ ($\geqslant 1$) crosses on the $n-1$ tracks, respectively, then there are exactly $d^{t_1+t_2+\cdots+t_{n-1}-n+1}$ ways to route any single request of the network.

Another main contribution of the LCP characterization is that it gives rise to the definition of routing grid of a BPN. The routing grid seems an ideal tool for analyzing realizability of permutations. In this section, we will use it to derive a necessary condition for a BPN to be rearrangeable, and to explore the rearrangeability of two families of BPNs, including the shuffle-exchange networks and Benes networks.

It is well-known that the $(2n-1)$-stage Benes network is rearrangeable. It is also known that $2n-1$ is a lower bound on the number of stages for a shuffle-exchange network to be rearrangeable. The rearrangeability of the $(2n-1)$-stage shuffle-exchange network is also known as Benes conjecture, whose validness is still open. Both networks share the similarities that they are BPNs, have $2n-1$ stages, and are obtained by concatenating two Omega equivalent networks. That is, both belong to the class $\Omega \otimes \Omega$. The next theorem shows that $\Omega \otimes \Omega$ contains all rearrangeable BPNs with minimum number of stages.

Theorem 4.1. *The minimum number of stages for a BPN to be rearrangeable is* $2n - 1$. *Furthermore, if G is a rearrangeable BPN with* $2n - 1$ *stages, then both of the two subnetworks,* $G_{1,n}$ *and* $G_{n,2n-1}$, *are topologically equivalent to the Omega network.*

Proof. Suppose the *i*th track of *G* has no cross. Then $c_{(i,1)}$ and $c_{(i,s)}$ are the same variable, which represents both a column of *P* and a column of *Q*. This means *P* and *Q* must have an identical column to be realizable on *G*. Suppose the *i*th track has only one cross. In the routing grid, there is only one double vertical line in the *i*th row. So the cells on the left (right) of the double vertical line all have variable $c_{(i,1)}$ ($c_{(i,s)}$). This implies $c_{(i,1)}$ and $c_{(i,s)}$ belong to A_k for some *k*. Therefore, the *i*th column of *P* and the *i*th column of *Q* belong to a *p*-matrix, contradicting the rearrangeability requirement that *P* and *Q* be arbitrary. The discussion shows that every track of a rearrangeable BPN has at least two crosses. So *G* has at least $2n - 2$ edge stages and at least $2n - 1$ node stages.

Since the $(2n - 1)$-stage Benes network is rearrangeable, any rearrangeable BPN with minimum number of stages has also $2n - 1$ stages, and each track has exactly two crosses. If in some track both of the two crosses appear before the *n*th node stage, in the routing grid, the output variable $c_{(i,s)}$ of this row will appear in column *n*. As the Pigeonhole Principle suggests, the two crosses of some other row will appear later than the *n*th node stage. So the input variable $c_{(j,1)}$ of that row must also appear in column *n*. This implies both $c_{(i,s)}$ and $c_{(j,1)}$ belong to A_n, contradicting the rearrangeability requirement. Hence, if *G* is a rearrangeable BPN with $2n - 1$ stages, each track of $G_{1,n}$ and $G_{n,2n-1}$ has exactly one cross. That is, both $G_{1,n}$ and $G_{n,2n-1}$ are Omega equivalent. □

The proof above virtually shows that every track of a rearrangeable BPN has at least two crosses, and the first cross of one track cannot appear later than the last cross of any other track. Noting that the Omega network is the connected BPN with minimum number of stages, we can make Theorem 4.1 even more general: if *G* is an *s*-stage rearrangeable BPN, then there exists a number *k*, $1 < k < s$, such that both $G_{1,k}$ and $G_{k,s}$ are connected.

Note that the routing grid of a BPN can also be constructed easily from its sequence notation described in Theorem 2.1. For example, the $(2n - 1)$-stage shuffle-exchange network has sequence $(1, 2, \ldots, n - 1, 1, 2, \ldots, n - 1)$, then we know the first track in its LCP decomposition has a cross in the first and the *n*th edge stage, respectively, the second track has a cross in the second and the $(n + 1)$th edge stage, respectively, and so on. The routing grid is shown in Fig. 8(a).

Let *B* be $(2n-1)$-stage Benes network. For $k = 1, 2, \ldots, n-1$, let B_k^1 be the network obtained from *B* by replacing the subnetwork $B_{2n-k-1,2n-1}$ (or the last *k* edge stages) by its mirror image, and let B_k^2 be the network obtained from *B* by replacing the subnetwork $B_{n,n+k}$ (or the first *k* edge stages of the second Omega equivalent network) by its mirror image. It is obvious both B_k^1 and B_k^2 belong to $\Omega \otimes \Omega$. The sequences of B, B_k^1 and B_k^2 are $(1, 2, \ldots, n-1, n-1, \ldots, 2, 1)$, $(1, \ldots, n - 1, n - 1, \ldots, k + 1, 1, \ldots, k)$ and $(1, \ldots, n - 1, n - k, \ldots, n - 1, n - k - 1, \ldots, 1)$, respectively. It is easy to see that $B_1^1 = B_1^2 = B$ and that $B_{n-1}^1 = B_{n-1}^2$ is the $(2n - 1)$-stage shuffle exchange network. Their routing grids are shown in Fig. 8(b)–(d). In the following, the $N \times (n - 1)$ matrices *P* and *Q* are, respectively, the input and the output matrix of a permutation $\{P, Q\}$. Now we discuss the rearrangeability of the mentioned networks separately.

(1) *Shuffle-exchange network*: The $(2n - 1)$-stage shuffle-exchange network has the simplest structure in its routing grid. In the routing matrix *M*, the input matrix *P* and the output matrix *Q* appear as the first and the last $n - 1$ columns, respectively. $\{P, Q\}$ is routable on the network if and only if there is an $N \times (n - 1)$ matrix *T* such that $M = (P, T, Q)$ is balanced.

Proposition 4.1. *The* $(2n - 1)$*-stage shuffle-exchange network is rearrangeable if and only if for any two* $N \times (n - 1)$ *balanced matrices P and Q, there is an* $N \times (n - 1)$ *matrix T such that* $M = (P, T, Q)$ *is balanced.*

It is not difficult to see that Proposition 4.1 also holds if *P* and *Q* are replaced with two *p*-matrices. This implies the binary case of Conjecture 1 is equivalent to Benes conjecture.

For shuffle-exchange networks with *s* stages, realizability conditions can also be derived by applying routing grid. In such cases, the columns of *Q* also appear in the last part of *M*, but probably in a different order. We can deduce that realizability of a permutation is always equivalent to the existence of an $N \times (s + n - 2)$ balanced matrix. When *s* is small, some columns of *P* and *Q* may appear in a *p*-matrices concurrently or even overlap, meaning *P* and *Q* cannot be arbitrary.

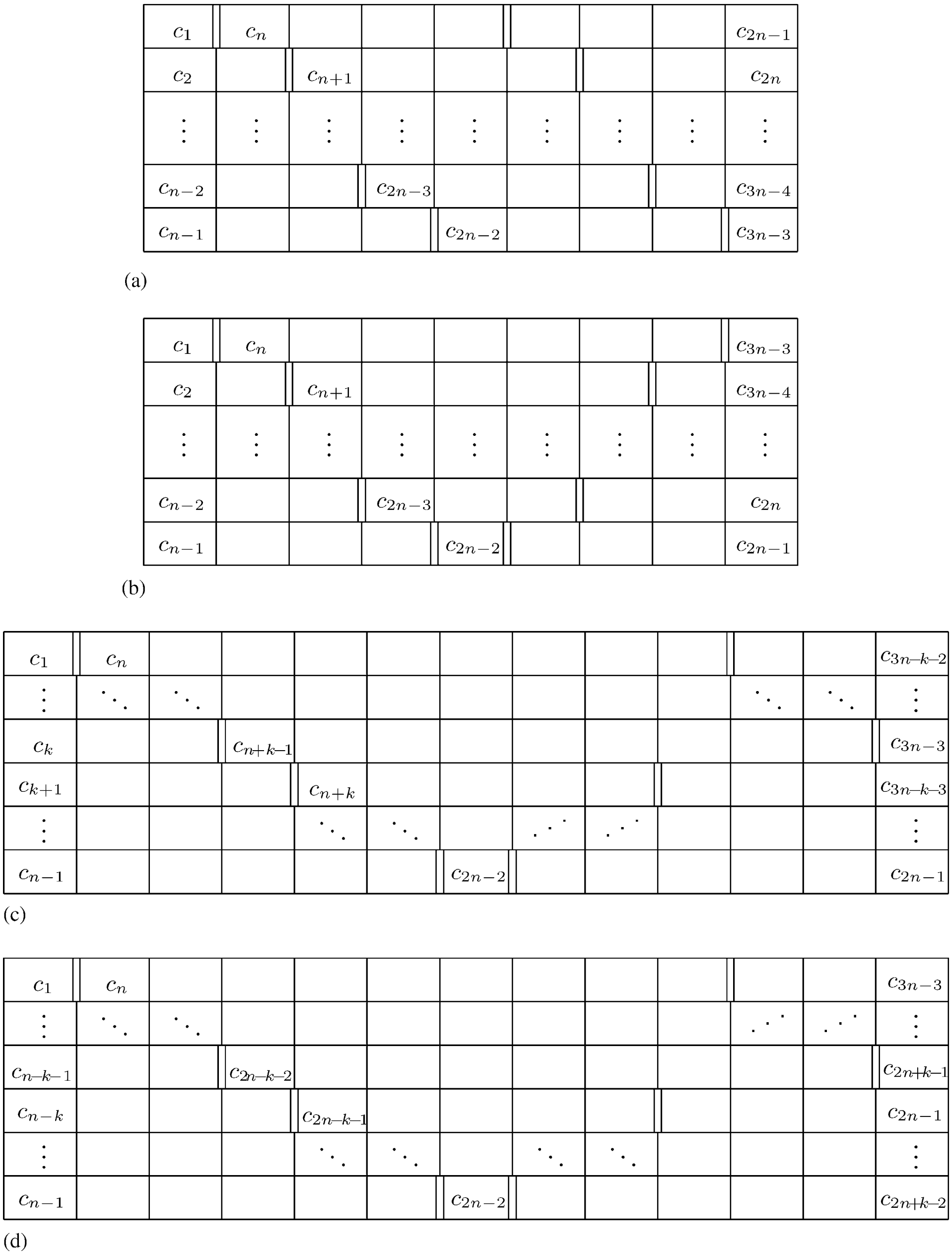


Fig. 8. Routing grids of four networks: (a) $(2n-1)$-stage shuffle-exchange network; (b) $(2n-1)$-stage Benes network; (c) $(2n-1)$-stage network B_k^1; and (d) $(2n-1)$-stage network B_k^2.

(2) *Benes network*: The $(2n-1)$-stage Benes network is among the few rearrangeable BPNs known so far. Let Q^r be the matrix obtained from Q by reversing the order of its columns. From the routing grid, we see that routing a permutation on it is equivalent to finding an $N \times (3n-3)$ matrix M in which $2n-2$ p-matrices reside. For example, the nth column v_n should be balanced with both P and Q ((P, v_n) and (v_n, Q^r) being p-matrices), and the $(n+1)$th column v_{n+1} should be such that (P, v_n, v_{n+1}) and (v_{n+1}, v_n, Q^r) are balanced, and so on. This can be easily implemented by applying function LOOP$(P, Q; v)$ defined in Section 2.3.

Algorithm–Benes

Let $\{P, Q\}$ be a permutation.
$Q_0 \leftarrow Q^r$; $Q \leftarrow Q^r$;
FOR $i = 1$ TO $n - 1$
  LOOP$(P, Q; v)$;
  $P \leftarrow (P, v)$; $Q \leftarrow (v, Q)$
  NEXT i
$M \leftarrow (P, Q_0)$;
OUTPUT M
END

We remark that Algorithm–Benes is actually a matrix version of the looping algorithm for any $d \geqslant 2$. Though not a brand-new algorithm, we include it for its neat form based on the operation LOOP$(P, Q; v)$.

(3) *Networks* B_k^1, $k = 1, 2, \ldots, n-1$: For the networks B_k^1, the nth to $(n+k-1)$th variables correspond to an $N \times k$ matrix T which must satisfy the balancedness of (P, T) and (T, Q'), where Q' has the same columns as Q but the order is different. If Q is an arbitrary balanced matrix, so is Q'. If T is found, the remaining computation needs only the looping operation. So we have

Proposition 4.2. *The rearrangeability of* B_k^1, $k = 1, 2, \ldots, n-1$, *is equivalent to the conclusion of Conjecture* 2.

(4) *Networks* B_k^2, $k = 1, 2, \ldots, n-1$: The networks B_k^2 have a recursive structure in its outer-stages, with the central $2k+1$ stages identical to d^{n-k-1} copies of the $d^{k+1} \times d^{k+1}$ $(2k+1)$-stage shuffle-exchange network.

Proposition 4.3. B_k^2 *is rearrangeable if the* $d^{k+1} \times d^{k+1}$ $(2k+1)$-*stage shuffle-exchange network is rearrangeable.*

Proof. Given a permutation $\{P, Q\}$, we get $n-k-1$ column vectors corresponding to variables $n, \ldots, 2n-k-2$ in the routing grid, by applying the looping operation $n-k-1$ times. These $n-k-1$ vectors constitute an $N \times (n-k-1)$ balanced matrix T. There are d^{n-k-1} different rows in T, each different row appears d^{k+1} times. Let the last k columns of P form matrix P', and the last k columns of Q form matrix Q'. From the routing grid we see that $\{P, Q\}$ is routable on B_k^2 if we can find an $N \times k$ matrix T' (corresponding to variables $2n-k-1, \ldots, 2n-2$) such that both (P', T, T') and (T', T, Q') are balanced. The set $\{0, 1, \ldots, N-1\}$ of row labels can be divided into d^{n-k-1} groups. In each group, the rows of T are identical. Let P_i' (Q_i') be the submatrix of P' (Q') containing the rows of P' (Q') in group i. Since (P', T) and (T, Q') are balanced, both P_i' and Q_i' are $d^{k+1} \times k$ balanced matrices, so $\{P_i', Q_i'\}$ can be considered as a permutation on a $d^{k+1} \times d^{k+1}$ network. If the $d^{k+1} \times d^{k+1}$ $(2k+1)$-stage shuffle-exchange network is rearrangeable, there is a $d^{k+1} \times k$ matrix T_i' such that (P_i', T_i', Q_i') is balanced. Let T' be the $N \times k$ matrix containing the rows of all T_i', in which the rows of T_i' appear in the same positions as those of P_i' appear in P'. Then it is easy to check both (P', T, T') and (T', T, Q') are balanced. So $\{P, Q\}$ is routable on B_k^2. □

The proof above shows that routing a permutation on B_k^2 can be reduced to route a number of permutations on a smaller shuffle-exchange network. An algorithm may be obtained by coordinating the looping operation with an algorithm for the $d^{k+1} \times d^{k+1}$ $(2k+1)$-stage shuffle-exchange network.

Corollary 4.1. B_2^2 *and* B_3^2 *are rearrangeable.*

The rearrangeability of 8×8 5-stage and 16×16 7-stage shuffle-exchange network was established in [22,19], respectively. A simple method was presented in [18] to route the 8×8 5-stage shuffle-exchange network. The rearrangeability of B_2^2 was also shown in [8] by algorithmic proof.

5. A method for balanced matrices

By Propositions 4.2 and 4.3, a solution of Conjecture 2 would solve the rearrangeability problem of B_k^1 and B_k^2, and therefore, would put an end to the validity question of Benes conjecture. In this section, we give an analysis of

$$
\text{(a)}\quad
\begin{array}{c}
\\ 0\\1\\2\\3\\4\\5\\6\\7
\end{array}
\begin{array}{c}
\begin{array}{ccccc}1&2&3&4&5\end{array}\\
\begin{pmatrix}
0&0&0&0&0\\
0&0&1&1&1\\
0&1&0&\textcircled{1}&0\\
0&1&1&0&\boxed{1}\\
1&0&0&1&1\\
1&0&1&0&\boxed{0}\\
1&1&0&\textcircled{0}&1\\
1&1&1&1&0
\end{pmatrix}
\end{array}
\qquad
\text{(b)}\quad
\begin{array}{c}
\begin{array}{ccccc}1&2&3&4&5\end{array}\\
\begin{pmatrix}
0&0&0&0&0\\
0&0&1&1&1\\
0&1&0&\boxed{0\ \ 1}\\
0&1&1&0&1\\
1&0&0&1&1\\
1&0&1&0&0\\
1&1&0&\boxed{1\ \ 0}\\
1&1&1&1&0
\end{pmatrix}
\end{array}
$$

(a) (b)

Fig. 9. Illustration of (a) groups and (b) a 4-shift.

Conjecture 2 and present some results which may help to attack it. The method here is based on some matrix operations which preserve balancedness. We also compare the method with Cam's in his attempt to prove Benes conjecture [9]. It is seen that our method is an abstraction and a generalization of the latter. Meanwhile, the simplification in our method can help readers a great deal to understand Cam's paper. At the end of this section, we give a brief analysis of Cam's proof, and point out its incompleteness as a support to another work of the authors [3].

Let $\mathcal{R} = \{0, 1, \ldots, N-1\}$ be the set of row labels. Let P be a p-matrix. Deleting the last column of P results in an $N \times (n-1)$ balanced matrix P' in which there are N/d different rows and each different row appears d times. This gives a partition of $\mathcal{R}$ in which i and j belong to the same subset if and only if the ith and the jth row of P' are identical. We call the subsets of this partition the *groups in column n* (n-groups, for short). An *n-group* of P is a set of d row labels which is defined by deleting the nth column. If P is a balanced matrix with $s \geqslant n$ columns, for each i ($i \geqslant n$), we define the *i-groups* of P to be the n-groups of the p-matrix formed by columns $i-n+1$ through i in P. Obviously, in column i, the d entries in every i-group form the set $\{0, 1, \ldots, d-1\}$. We also say these *d numbers belong to an i-group*.

Fig. 9(a) shows an 8×5 binary balanced matrix. Its 3-groups, 4-groups and 5-groups are $\{\{0,1\},\{2,3\},\{4,5\},\{6,7\}\}$, $\{\{0,4\},\{1,5\},\{2,6\},\{3,7\}\}$ and $\{\{0,6\},\{1,7\},\{2,4\},\{3,5\}\}$, respectively. So the two numbers in circles belong to a 4-group, and the two in squares belong to a 5-group.

Let r and s be two row labels in an i-group of P. For all $j \geqslant i$, exchanging the positions of entry (r, j) and (s, j) in P results in a new matrix. We call this operation an *i-shift* on P. So an i-shift exchanges the positions of two semi-rows in an i-group beginning at column i. Fig. 9(b) shows a 4-shift between the two semi-rows beginning at the circled numbers in Fig. 9(a). The exchanged semi-rows are put in rectangles in Fig. 9(b).

Proposition 5.1. *If P is balanced, then for any i, a number of i-shifts in P results in a new balanced matrix. Furthermore, if a series of shifts are performed on P in the order that all j-shifts are performed before i-shifts whenever* $j > i$, *the resulting matrix is also balanced.*

Proof. Since an i-shift on P changes two semi-rows beginning at column i, it only affects the balancedness of submatrices beginning after column $i-n$. So it suffices to check the balancedness of the submatrix from column $i-n+1$ to the last column of P. Before the i-shift, this submatrix of P is balanced. By the definition of i-shift, the two rows to be changed are in an i-group, this implies these two rows are identical in the first $n-1$ bits. So the i-shift can be considered as actually exchanging two whole rows in the submatrix. Since exchanging the positions of two rows in a balanced matrix always results in a balanced matrix, the balancedness of the submatrix is preserved after the i-shift. So a number of i-shifts on P do not break balancedness.

If all j-shifts ($j > i$) have been performed and the new matrix remains balanced, all columns of P before column i are kept intact so far. So the proof in the last paragraph still applies if an i-shift is to be performed at this stage. This means the second part of the proposition also holds. □

Let $P_0 = (P, M_0)$ be balanced, where P is a p-matrix and M_0 has $k \geqslant 1$ columns. Now define the i-groups ($1 \leqslant i \leqslant k$) of M_0 to be the $(n+i)$-groups of P_0. Using these groups, we can directly perform any sequence of shifts in M_0 (in the order that j-shifts always go before i-shifts whenever $1 \leqslant i < j \leqslant k$). Denote by $\mathcal{M}$ the set of all resulting matrices.

Proposition 5.1 states that (P, M) is balanced for all $M \in \mathcal{M}$. The next proposition shows $\mathcal{M}$ contains exactly all the matrices M of k columns satisfying the balancedness of (P, M).

Proposition 5.2. *(P, M) is balanced if and only if $M \in \mathcal{M}$.*

Proof. We only need to prove the necessity. Let $P_1 = (P, M)$ be balanced, and $M_0 = (u_1, u_2, \ldots, u_k)$, $M = (v_1, v_2, \ldots, v_k)$. It is not difficult to see that the $(n+1)$-groups of P_0 and P_1 are the same. So by a number of $(n+1)$-shifts on P_1, the first column of M, v_1, can be converted to u_1. We still use $v_2, \ldots, v_k$ to denote the last $k-1$ columns of P_1 after the $(n+1)$-shifts. That is, after the $(n+1)$-shifts, P_1 becomes the balanced matrix $P_2 = (P, u_1, v_2, \ldots, v_k)$. Now, the $(n+2)$-groups of P_0 and P_2 are also the same. So a number of $(n+2)$-shifts on P_2 convert it to a balanced matrix $P_3 = (P, u_1, u_2, v_3, \ldots, v_k)$. Repeat this procedure until P_0 is obtained. That is, we have found a series of i-shifts $(n+1 \leqslant i \leqslant n+k)$ on P_1, performed in the order that j-shifts follow i-shifts for $j > i$, which convert P_1 to P_0 and thus M to M_0. Therefore, applying these shifts to P_0 in the reversed order will convert P_0 to P_1. So $M \in \mathcal{M}$. □

From the proposition above, it is seen that the selection of M_0 is not important in defining $\mathcal{M}$, for, as long as (P, M_0) is balanced, $\mathcal{M}$ always consists of all matrices M satisfying the balancedness of (P, M). It is also clear that Conjecture 2 is equivalent to the statement: for a fixed P and any p-matrix Q, there exists $M \in \mathcal{M}$ such that (M, Q) is balanced.

Let $P_0 = (P, M_0)$ be as above and $\mathcal{S}$ be the set of all $N!$ p-matrices. Let $\mathcal{U} = \{(M_0, Q) \mid Q \in \mathcal{S}, (M_0, Q) \text{ is balanced}\}$ and $\mathcal{V} = \{Q \mid (M_0, Q) \in \mathcal{U}\}$. Obviously, $\mathcal{U}$ and $\mathcal{V}$ are nonempty. For each $(M_0, Q) \in \mathcal{U}$, we define its i-groups $(1 \leqslant i \leqslant k)$ to be the $(n+i)$-groups of P_0. Beginning with $\mathcal{U}$, we gradually enlarge it (and thus enlarge $\mathcal{V}$) by making all possible series of i-shifts $(1 \leqslant i \leqslant k)$ to all its members. At the point when $\mathcal{U}$ can no longer be extended, the submatrices consisting of the first k columns of every member in $\mathcal{U}$ form exactly set $\mathcal{M}$. By analysis similar to the proof of Proposition 5.1, we see that each member of $\mathcal{U}$ is balanced.

Proposition 5.3. *Conjecture 2 holds if and only if $\mathcal{V} = \mathcal{S}$ (after it has been enlarged as above).*

There is another way of looking at the problem. If M is a k-column matrix and (P, M) is balanced, we say M is a *k-extension of P on the right*. Similarly, we can define a k-extension of P on the left. We can also present a characterization of k-extensions on the left parallel to Proposition 5.2. This can be done if only we modify the definition of i-groups and i-shifts. So Conjecture 2 holds if and only if there is a k-extension of P on the right which is also a k-extension of Q on the left.

In the following, we will have a comparison of our method with Cam's in his proof of Benes conjecture, and give a brief analysis of his central part. The comparison and analysis can help readers in comprehending Cam's paper.

In [9], only $k = n - 1$ is considered for the binary case, and P is implicitly assumed to be I^r, the matrix obtained from the identity p-matrix by reversing the order of columns. Cam also chooses the matrix (denoted by I_1^r) obtained from I^r by deleting the last column as M_0. This is legitimate since $P_0 = (I^r, I_1^r)$ is balanced. His idea is essentially to show $\mathcal{V} = \mathcal{S}$ for this selection of P_0. Three facts should be noted to have a fair comparison:

1. Cam's method depends heavily on the concept of "frames" [10]. We point out that balancedness is a more general concept than frame is. For example, one can see that a matrix M fits the $N \times n$ frame if and only if (I^r, M) is balanced. So balancedness can fit into a wider setting of studies.
2. Ref. [9] contains a characterization of permutations realizable on a concatenated network of the reverse baseline network and the Omega network, and a proof of functional equivalence between the concatenated network and the $(2n-1)$-stage shuffle-exchange network. This part of work is unnecessary in our method as Theorem 2.7 or Proposition 4.1 guarantees the correctness if an $N \times (n-1)$ matrix M such that (P, M, Q) is balanced can always be found.
3. Cam's method also introduces the "inverse shuffle-exchange network". Its function corresponds to performing shifts in our method.

Except for those parts that are not necessary in our method, both methods are parallel in principle. For example, the special case of Proposition 5.2 is presented in Lemma 4.4 of [9]. A major difference appears in the last step. That is the definition of $\mathcal{U}$ and $\mathcal{V}$ before Proposition 5.3. In [9], it is done by defining a number of sets noindent as follows:

$$
\begin{aligned}
M_0 &= I_1^r = (v_1, v_2, \ldots, v_{n-1}),\ P_0 = (I^r, I_1^r),\ \mathcal{U}_0 = \mathcal{S}.\\
\mathcal{U}_1{}' &= \{(v_{n-1}, Q) \mid Q \in \mathcal{U}_0 \text{ and } (v_{n-1}, Q) \text{ is balanced}\}\\
\mathcal{V}_1{}' &= \{Q \mid Q \text{ is the last } n \text{ columns of some } R \in \mathcal{U}_1{}'\}\\
\mathcal{U}_1 &= \{Q \mid Q \text{ is obtained from some } R \in \mathcal{U}_1{}' \text{ by shifts}\}\\
\mathcal{V}_1 &= \{Q \mid Q \text{ is the last } n \text{ columns of some } R \in \mathcal{U}_1\}\\
&\quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots\\
\mathcal{U}_k{}' &= \{(v_{n-k}, Q) \mid Q \in \mathcal{U}_{k-1} \text{ and } (v_{n-k}, Q) \text{ is balanced}\}\\
\mathcal{V}_k{}' &= \{Q \mid Q \text{ is the last } n \text{ columns of some } R \in \mathcal{U}_k{}'\}\\
\mathcal{U}_k &= \{Q \mid Q \text{ is obtained from some } R \in \mathcal{U}_k{}' \text{ by shifts}\}\\
\mathcal{V}_k &= \{Q \mid Q \text{ is the last } n \text{ columns of some } R \in \mathcal{U}_k\}\\
&\quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots
\end{aligned}
$$

where $1 \leqslant k \leqslant n-1$, and the shifts for computing $\mathcal{U}_k$ are 1-shifts using the $(2n-k)$-groups of P_0. Actually, the sets $\mathcal{U}_k{}'$ and $\mathcal{V}_k{}'$ do not appear in [9], we use them only for ease of reference.

Note that $\mathcal{V}_{n-1} = \mathcal{V}$. Cam's idea is to prove $\mathcal{V}_k = \mathcal{S}$ for all $k = 1, 2, \ldots, n-1$. This statement is included in his Theorem 4.6. The basis for his proof is the following lemma.

Lemma 5.1 (*Corollary 4.2 of [9]*). *Consider a* 0–1 *vector whose rows are partitioned into* $N/2$ *groups and each group contains* $\{0, 1\}$. *Suppose* $1 \leqslant k \leqslant n-1$. *Let* $\mathcal{T}$ *denote the set of all* $N \times k$ *matrices* M *satisfying the balancedness of* (v, M). *Then any* $N \times k$ *balanced matrix can be obtained in the last k columns if all* $2^{N/2}$ *possible series of* 1-*shifts are performed to each* (v, M), $M \in \mathcal{T}$.

When v_{n-k} is appended to the left of every element in $\mathcal{U}_{k-1}$, some unbalanced matrices are discarded because the first $n-k$ columns of every element in $\mathcal{V}_k{}'$ must be balanced with v_{n-k}. But it is not a sufficient condition for a p-matrix to remain in $\mathcal{V}_k{}'$. This is testified by an example in [3]. The fact that "the first $n-k$ columns of a p-matrix being balanced with v_{n-k} does not guarantee its membership in $\mathcal{V}_k{}'$" may be a hint to the complexity of $\mathcal{V}_k{}'$. Cam shows that the set of submatrices in the first $n-k$ columns of members in $\mathcal{V}_k{}'$ contains all $(n-k)$-column matrices that are balanced with v_{n-k}. The invalidity of this part of proof is pointed out in [3]. But whether or not it contains all $(n-k)$-column matrices balanced with v_{n-k} is, in fact, not very important, for it involves only a fraction of columns in $\mathcal{V}_k{}'$ and Lemma 5.1 can only generate conclusions about the set of submatrices corresponding to these columns. To prove $\mathcal{V}_k = \mathcal{S}$, we need a characterization of $\mathcal{V}_k{}'$, not of its submatrices. So we can safely say that Lemma 5.1 itself is not sufficient to prove Theorem 4.6 of [9]. One possible solution is to relax the conditions of Lemma 5.1 by removing the upper bound of k (it holds!). But then the example in [3] suggests that $\mathcal{V}_k{}'$ does not meet the condition of this new version of Lemma 5.1.

6. Conclusions

In this work, by representing BPNs as the LCP of a set of uniform factors, we introduced the concept of routing grid as a tool for analyzing realizability of permutations on them. We extended a result by Linial and Tarsi which characterizes permutations realizable on shuffle-exchange networks to any BPNs. It was shown that any minimum rearrangeable BPN is a concatenation of two Omega equivalent networks. As an application of routing grid, we explored the rearrangeability of two interesting families of networks, and reduced it to one kind of balanced matrix problems. A method which might help to attack it was presented and a comparison was made with Cam's method in his attempt to prove Benes conjecture. Hopefully, our treatment brings some new insight into the problem of permutation routing.

References

[1] D.P. Agrawal, Graph theoretical analysis and design of multistage interconnection networks, IEEE Trans. Comput. 32 (1983) 637–648.

[2] S. Andresen, The looping algorithm extended to base 2^t rearrangeable switching networks, IEEE Trans. Comm. 25 (1977) 1057–1063.

[3] X. Bao, F.K. Hwang, On a proof of the shuffle-exchange conjecture, submitted for publication.

[4] V.E. Benes, Proving the rearrangeability of connecting networks by group calculations, Bell System Tech. J. 45 (1975) 421–434.
[5] C. Berge, Graphs and Hypergraphs, North Holland, Amsterdam, 1976.
[6] J.C. Bermond, J.M. Fourneau, A. Jean-Marie, Equivalence of multistage interconnection networks, Inform. Process Lett. 26 (1987) 45–50.
[7] T. Calamoneri, A. Massini, Efficiently checking the equivalence of multistage interconnection networks, Proc. Parallel and Distributed Computing and Systems (PDCS'99), Cambridge, MA, 1999, pp. 23–30.
[8] T. Calamoneri, A. Massini, A new approach to the rearrangeability of $(2 \log N{-}1)$ stage MINs, Proc. IASTED Internat. Symp. Applied Informatics (AI 2001), 2001, pp. 365–370.
[9] H. Cam, Rearrangeability of $(2n{-}1)$-stage shuffle-exchange networks, SIAM J. Comput. 32 (2003) 557–585.
[10] H. Cam, J.A.B. Fortes, Frames: a simple characterization of permutations realized by frequently used networks, IEEE Trans. Comput. 44 (1995) 695–697.
[11] G.J. Chang, F.K. Hwang, L.-D. Tong, Characterizing bit permutation networks, Networks 33 (1999) 261–267.
[12] S. Even, A. Litman, Layered cross product—a technique to construct interconnection networks, Networks 29 (1997) 219–223.
[13] T.-Y. Feng, S.-W. Seo, A new routing algorithm for a class of rearrangeable networks, IEEE Trans. Comput. 43 (1994) 1270–1280.
[14] N. Golbandi, A. Litman, Characterizations of generalized butterfly networks, preprint, 2002.
[15] Q. Hu, X. Shen, J. Yang, Topologies of combined $(2 \log N{-}1)$-stage interconnection networks, IEEE Trans. Comput. 46 (1997) 118–124.
[16] F.K. Hwang, C.-H. Yen, Characterizing the bit permutation networks obtained from the line digraphs of bit permutation networks, Networks 38 (2001) 1–5.
[17] M.K. Kim, H. Yoon, S.R. Maeng, On the correctness of inside-out routing algorithm, IEEE Trans. Comput. 46 (1997) 820–823.
[18] N. Linial, M. Tarsi, Interpolation between bases and the shuffle-exchange network, European J. Combin. 10 (1989) 29–39.
[19] H.Q. Ngo, D. Du, On the rearrangeability of shuffle-exchange networks, Proc. Fourth Internat. Conf. on Algorithms and Architectures for Parallel Processing, 2000.
[20] D.C. Opferman, N.T. Tsao-Wu, On a class of rearrangeable switching networks—Part I: Control algorithm, Bell System Tech. J. 50 (1971) 1579–1600.
[21] C.S. Raghavendra, On the rearrangeability conjecture of $(2 \log_2 N - 1)$-stage shuffle/exchange network, position paper, Computer Architecture Technical Committee Newsletter, Winter 1994–1995, pp. 10–12.
[22] C.S. Raghavendra, A. Varma, Rearrangeability of the five stage shuffle-exchange network for $N = 8$, IEEE Trans. Comm. 35 (1987) 808–812.
[23] Y. Wu, X. Bao, X. Jia, Q. Li, Graph theoretical characterizations of bit permutation networks, manuscript, 2001.