Photonic Network Communications, Vol.5, No.1, pp.89-102 (2003).

A New Scheme to Realize Crosstalk-free Permutation in Vertically Stacked Optical MINs[1]

Xiaohong Jiang, Hong Shen, Md. Mamun-ur-Rashid Khandker and Susumu Horiguchi
Graduate School of Information Science,
Japan Advanced Institute of Science and Technology,
JAIST, Tatsunokuchi, ISHIKAWA 923-1292, JAPAN
Email: { jiang,shen,hori }@jaist.ac.jp

ABSTRACT
Vertical stacking is a novel alternative for constructing nonblocking multistage interconnection networks (MINs). Rearrangeably nonblocking optical MINs are attractive since they have lower complexity than their strictly nonblocking counterparts. In this paper, we study the realization of crosstalk-free permutations in rearrangeably nonblocking, self-routing banyan-type optical MINs built on vertical stacking. An available scheme for realizing crosstalk-free permutation in this type of optical MINs requires to first decompose a permutation into multiple crosstalk-free partial permutations based on the Euler-Split technique, and then to realize them crosstalk-free in different planes (stacked copies) of the MIN simultaneously. The overall time complexity of this scheme to realize a corsstalk-free permutation in an $N\times N$ optical MIN is $O(N\log N)$ which is dominated by the complexity of crosstalk-free decomposition. In this paper, we propose a new scheme for realizing permutations in this class of vertically stacked optical MINs crosstalk-free. The basic idea of the new scheme is to classify permutations into permutation classes such that all permutations in one class share the same crosstalk-free decomposition pattern. By running the Euler-Split based crosstalk-free decomposition only once for a permutation class and applying the obtained crosstalk-free decomposition pattern to all permutations in the class, crosstalk-free decomposition of permutations can be realized in a more efficient way. We show that the number of permutations in a permutation class is huge (at least $\left(\sqrt{N}!\right)^{\sqrt{N}}$ when $\log_2 N$ is even and $\left(\sqrt{2N}!\right)^{\sqrt{\frac{N}{2}}}$ when $\log_2 N$ is odd), and thus the average time complexity of crosstalk-free decomposition of a permutation becomes $O(N)$.

Keywords: banyan network, optical switch, optical crosstalk, rearrangeably nonblocking.

1. Introduction

A basic element of optical switching networks is a directional-coupler (DC) with similar function of 2×2 switching element (SE). The crossing or parallel state of DC can be created by applying a suitable control voltage to it. DC-based optical switching networks can switch signals at the very high speed, and such networks are also capable of switching signals with multiple wavelengths. Crosstalk is a major shortcoming of DC, which occurs between two signals carried in the two waveguides of the coupler [1][2]. By ensuring that only one signal passes through a switch at a time, the first-order crosstalk in SEs can be eliminated and this provides a cost-effective solution to the crosstalk problem. Due to the stringent bit-error rate requirement of optical transmission facilities, elimination of crosstalk in a DC-based switching system has been widely studied [1,7,10,11,12,13, 14, 15].

Banyan [3] or its topologically equivalent (e.g. *baseline*, *omega*) networks [5, 8] are a class of attractive switching networks because they are fast in switch setting (self-routing) and also have a small number of switches between an input-output pair. These characteristics make banyan-type network an ideal network structure for constructing DC-based optical switching networks because crosstalk and signal attenuation in a DC-based optical switching network are proportional to the number of couplers that a light signal passes through. A typical network structure for this class of banyan-type network is that each network has N inputs and outputs and $n = \log_2 N$ stages,

[1] This work is support by Telecommunications Advancement Organization of Japan.

with each stage consisting of $N/2$ 2×2 switches and any two adjacent stages connected by N inter-stage links as shown in Fig.1.

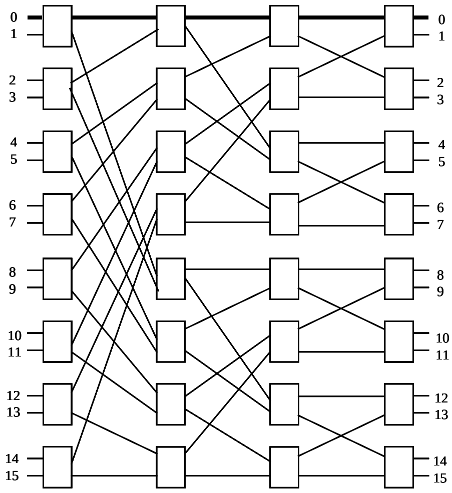

Fig.1 16×16 Banyan-type network

Banyan-type networks have a unique path between an input-output pair, and this makes them blocking networks. Vertical stacking [6] is a novel scheme for constructing nonblocking network as illustrated in Fig.2.

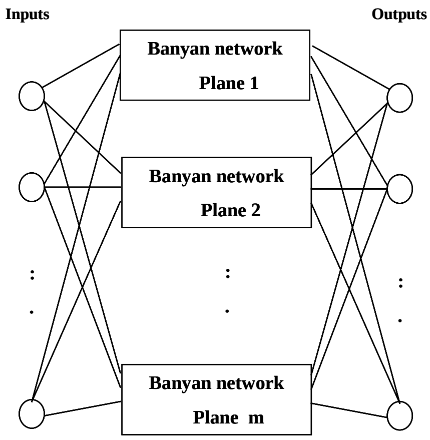

Fig.2 Creating non-blocking network based on the vertical stacking scheme

There are three types of nonblocking networks, namely strictly nonblocking, wide-sense nonblocking and rearrangeably nonblocking [16,17]. In a strictly nonblocking network, any input can be routed to any unused output regardless of the way other inputs signals are routed. This high degree of flexibility comes at the cost of a high hardware cost in term of the number of SEs required. Strictly nonblocking optical networks built on vertical stacking have been explored in recent work [10, 11, 18]. In a wide-sense nonblocking network, a rule must be followed to establish a connection. This indicates that not every free path can be used. Some results for wide-sense nonblocking optical networks with vertical stacking can be found in [19].

Rearrangeably nonblocking is an interesting choice for optical switching networks. This kind of network can always route any idle input to any unused output, but one or more existing connections may have to be rerouted to establish the path. The rearrangeably nonblocking optical networks are attractive because the cost and signal degradation of a rearrangeably nonblocking optical network are much lower than its strictly nonblocking and wide-sense nonblocking counterparts. Based on the vertical-stacking scheme, the condition for a banyan-type network to be rearrangeably nonblocking and free of crosstalk in SEs (we refer to this as crosstalk-free hereafter) has been determined in [9,12]. In this paper, we look into the crosstalk-free permutation in rearrangeably nonblocking banyan-type optical MINs built on the vertical stacking technique. A scheme was proposed in [20] to realize crosstalk-free permutation in this type of rearrangeably nonblocking optical MINs. The basic idea of this scheme is to first decomposed a permutation into multiple crosstalk-free partial permutations which requires amount of $O(N\log N)$ time, and then to realize them crosstalk-free in different planes (stacked copies) of the MIN simultaneously in amount of $O(\log N)$ time taking advantage of the self-routing property of a banyan-type network. In this paper, we present a new scheme for realizing permutations in this class of vertically stacked crosstalk-free optical MINs. Since inherent similarities among permutations in the sense of crosstalk-free decomposition are fully utilized in the new scheme, crosstalk-free decomposition can be realized in a more efficient way with an average time complexity of $O(N)$.

2. Rearrangeably noblocking networks under crosstalk-free constraint

Ideally, we are interested in designing a network without any crosstalk. For convenience, we use the notation $B(N,p)$ to refer to an $N\times N$ MIN that consists of p vertically stacked copies of banyan-type networks with $n=\log_2 N$ stages and having no crosstalk along the path of each connection. We have the following result concerning the rearrangeably nonblocking conditions for a $B(N,p)$ network [9,12].

Theorem 1: A $B(N,p)$ network is rearrangeably nonblocking if the following is true

$$p \geq 2^{\lfloor (n+1)/2 \rfloor} \tag{1}$$

It is easy to verify that under the crosstalk-free constraint, we need at least $2^{\lfloor (n+1)/2 \rfloor}$ planes to realize the identical permutation in an $N\times N$ network. Thus, $2^{\lfloor (n+1)/2 \rfloor}$ is the minimum number of planes to guarantee a $B(N,p)$ network to be nonblocking. Hereafter, we will use $RB(N)$ to refer to the rearrangeably nonblocking network $B(N,p)$ which consists of $2^{\lfloor (n+1)/2 \rfloor}$ copies (planes) of the banyan-type network. Note that under the crosstalk-free constraint, the complexity of a rearrangeably Banyan-type optical MIN is much lower than its strictly nonblocking counterpart [10] and its wide-sense nonblocking counterpart [19].

3. Permutation and Crosstalk-Free Partial Permutation (CFPP)

A permutation is a full one-to-one mapping between the network inputs and outputs. For an $N \times N$ banyan-type network, suppose input x_i is mapped to output y_i, where $x_i = i$ and $y_i \in \{0,1,\ldots,N\text{-}1\}$ for $i = 0,1,\ldots, N\text{-}1$. We denote this permutation as

$$\begin{pmatrix} x_0, x_1, \cdots, x_{N-1} \\ y_0, y_1, \cdots, y_{N-1} \end{pmatrix} \tag{2}$$

And also, we call a one-to-one mapping between N_1 inputs and N_1 outputs in the network ($N_1 < N$) a partial permutation.

Definition 1 *A partial permutation in an optical MIN is called a Crosstalk Free Partial Permutation (CFPP) if the partial permutation is crosstalk-free realizable in the optical MIN.*

Example 1 The decomposition of a permutation of an 16×16 banyan network into CFPPs.

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 8 & 2 & 0 & 12 & 4 & 13 & 3 & 11 & 9 & 1 & 6 & 7 & 5 & 10 & 15 & 14 \end{pmatrix} \Rightarrow$$
$$\begin{pmatrix} 1 & 7 & 10 & 14 \\ 2 & 11 & 6 & 15 \end{pmatrix} \circ \begin{pmatrix} 2 & 5 & 8 & 12 \\ 0 & 13 & 9 & 5 \end{pmatrix} \circ \begin{pmatrix} 0 & 6 & 11 & 15 \\ 8 & 3 & 7 & 14 \end{pmatrix} \circ \begin{pmatrix} 3 & 4 & 9 & 13 \\ 12 & 4 & 1 & 10 \end{pmatrix} \tag{3}$$

4. Crosstalk-free permutation in a RB(N) network

The above result indicates that all permutations can be realized crosstalk-free in a RB(*N*). In [20], the following Lemma 1 concerning the crosstalk-free property of an optical MIN and Theorem 2 concerning the CFPP decomposability of a permutation were presented.

Lemma 1 For an $N \times N$ banyan-type network and an integer i ($0 \le i \le (1/2)(n-2)$ when n is even or $0 \le i \le (1/2)(n-1)$ when n is odd), define the sets:

$$I_j^{[i]} = \{u_{2^i \cdot j}, u_{2^i \cdot j+1}, \cdots, u_{2^i \cdot j+2^i-1}\},\ O_j^{[i]} = \{v_{2^i \cdot j}, v_{2^i \cdot j+1}, \cdots, v_{2^i \cdot j+2^i-1}\},\ 0 \le j \le \tfrac{N}{2^{i+1}} - 1$$

where $u_0, u_1, \ldots, u_{N/2-1}$ are the $N/2$ inputs switches and $v_0, v_1, \ldots, v_{N/2-1}$ are the $N/2$ outputs switches. For the two inputs (outputs) of any two one-pair mappings in the network, the two mappings will be crosstalk-free in the first (last) i+1 stages of the network if their corresponding two input (output) switches belong to two different input (output) sets defined above.

Theorem 2: Any permutation of an *N*-element set {0,1,…, *N*-1} can be decomposed into $2^{\lfloor (n+1)/2 \rfloor}$ CFPPs, and there exist permutations which can be decomposed into at least $2^{\lfloor (n+1)/2 \rfloor}$ CFPPs.

The following algorithm was also developed in [20] to actually decompose a permutation of set {0,1,…, *N*-1} into $2^{\lfloor (n+1)/2 \rfloor}$ CFPPs.

Algorithm 1: Decomposition of a permutation into CFPPs:

Initiate: $i = 0$ and take the permutation as the 0-level partial permutation.

Step 1: If $i = \lfloor (n+1)/2 \rfloor$, exit.

Step 2: For each *i*-level partial permutation, do step 3 and step 4.

Step 3: Construct a undirected bipartite graph $G = (V_1, V_2; E)$ for the *i*-level partial permutation.

The vertex sets of *G* are defined by:

$$V_1 = \left\{ I_0, I_1, \cdots, I_{\frac{N}{2^{i+1}}-1} \right\},\ V_2 = \left\{ O_0, O_1, \cdots, O_{\frac{N}{2^{i+1}}-1} \right\}$$

Here $I_j = \{u_{2^i \cdot j}, u_{2^i \cdot j+1}, \cdots, u_{2^i \cdot j+2^i-1}\}$ and $O_j = \{v_{2^i \cdot j}, v_{2^i \cdot j+1}, \cdots, v_{2^i \cdot j+2^i-1}\}$ for $0 \le j \le \frac{N}{2^{i+1}} - 1$, and $u_0, u_1, \ldots, u_{N/2-1}$ are the $N/2$ inputs switches and $v_0, v_1, \ldots, v_{N/2-1}$ are the $N/2$ outputs switches. The edge set *E* is defined as: for any one-pair mapping $\begin{pmatrix} x_i \\ y_i \end{pmatrix}$ in the *i*-level partial permutation, it holds that if the input switch corresponding to x_i belongs to I_{j_1} and the outputs switch corresponding to y_i belongs to O_{j_2}, then there is an edge between vertex I_{j_1} and vertex O_{j_2} in *E*.

Step 4: Find the Euler tour.

Since any vertex in each connected component of G has degree 2, we know from graph theory [4] that there exists an Euler tour which traverses each edge of the component exactly once. Then for each connected component of G, start from any vertex in V_1 in it, traverse through an unvisited edge to the neighboring vertex V_2, back and forth until return to the starting vertex. During the traversing, a visited edge is will be placed into set E_1 if the traverse direction on this edge is from V_1 to V_2; and placed into set E_2 if the direction is opposite. It is easy to see that the set of all edges in E_1 is a perfect matching of the bipartite graph *G*, and so is the set of edges in E_2.

Step 5: Take all one-pair mappings corresponding to the edges in E_1, to form one (i+1)-level partial permutation corresponding to the *i*-level partial permutation; let the remaining one-pair mappings, corresponding to the edges in E_2, form another (i+1)-level partial permutation corresponding to the *i*-level partial permutation.

Step 6: $i \Leftarrow i+1$. Go to Step 1.

It is clear that after executing Steps 2-5 for the *i*-level partial permutations of a permutation, the (i+1)-level partial permutations obtained will eliminate all crosstalks in both the first (i+1) stages and the last (i+1) stages as guaranteed by Lemma 1. Thus, after running the decomposition algorithm for a permutation in an $N \times N$ Banyan-type MIN, the permutation will be decomposed into $2^{\lfloor (n+1)/2 \rfloor}$ partial permutations that eliminate the crosstalk in all stages of the network. By realizing each of these CFPPs in a single plane of a RB(*N*) network, the full permutation can be realized crosstalk-free in a single pass based on the parallel message transmission.

The Steps 2-5 take $O(N)$ steps and these steps repeat $O(\log_2 N)$ times, so that the time complexity of the decomposition algorithm is therefore $O(N \log_2 N)$. Since the RB(N) network consists of multi-copies of Banyan-type networks which are self-routing, so the routing complexity is $O(\log_2 N)$.Thus, the overall time complexity to realize a permutation crosstalk-free in a RB(N) network is $O(N \log_2 N)+ O(\log_2 N)$ based on the decomposition algorithm 1.

Example 2: The decomposition of the permutation in Example 1 into CFPPs based on Algorithm 1.

Since N=16 and (1/2)($\log_2 N$)=2, we need two levels decompositions to decompose the permutation into $\sqrt{N} = 4$ CFPPs.

First-level decomposition: Take the full permutation in Example 1 as the 0-level partial permutation. The bipartite graph and edge traverses are shown in Figure 3,

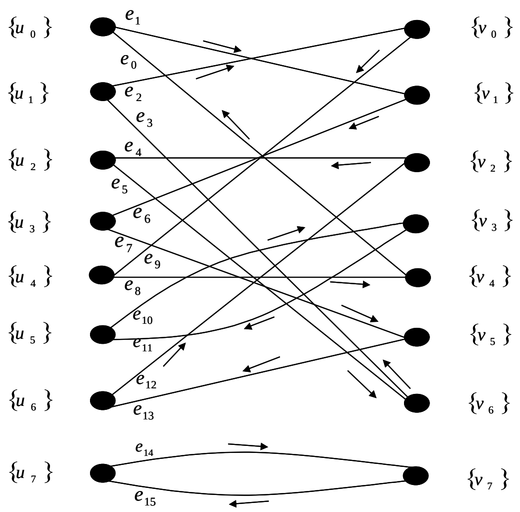

Fig.3 The bipartite graph and edge traverses of the first-level decomposition

where

$$e_0=\begin{pmatrix}0\\8\end{pmatrix}, e_1=\begin{pmatrix}1\\2\end{pmatrix}, e_2=\begin{pmatrix}2\\0\end{pmatrix}, e_3=\begin{pmatrix}3\\12\end{pmatrix}, e_4=\begin{pmatrix}4\\4\end{pmatrix}, e_5=\begin{pmatrix}5\\13\end{pmatrix}, e_6=\begin{pmatrix}6\\3\end{pmatrix}, e_7=\begin{pmatrix}7\\11\end{pmatrix}, e_8=\begin{pmatrix}8\\9\end{pmatrix}, e_9=\begin{pmatrix}9\\1\end{pmatrix}, e_{10}=\begin{pmatrix}10\\6\end{pmatrix},$$

$$e_{11}=\begin{pmatrix}11\\7\end{pmatrix}, e_{12}=\begin{pmatrix}12\\5\end{pmatrix}, e_{13}=\begin{pmatrix}13\\10\end{pmatrix}, e_{14}=\begin{pmatrix}14\\15\end{pmatrix}, e_{15}=\begin{pmatrix}15\\14\end{pmatrix}.$$

Then the pairs e_1, e_2, e_5, e_7, e_8, e_{10}, e_{12} and e_{14} corresponding to the edges in E_1 form a first-level partial permutation

$$\begin{pmatrix}1 & 2 & 5 & 7 & 8 & 10 & 12 & 14\\ 2 & 0 & 13 & 11 & 9 & 6 & 5 & 15\end{pmatrix} \quad (4)$$

and the pairs e_0, e_3, e_4, e_6, e_9, e_{11}, e_{13} and e_{15} corresponding to the edges in E_2 form another first-level partial permutation

$$\begin{pmatrix}0 & 3 & 4 & 6 & 9 & 11 & 13 & 15\\ 8 & 12 & 4 & 3 & 1 & 7 & 10 & 14\end{pmatrix} \quad (5)$$

which completes the fist-level decomposition.

Second-level decomposition: For the first-level partial permutation (4), the bipartite graph and edge traverses are shown in Figure 4.

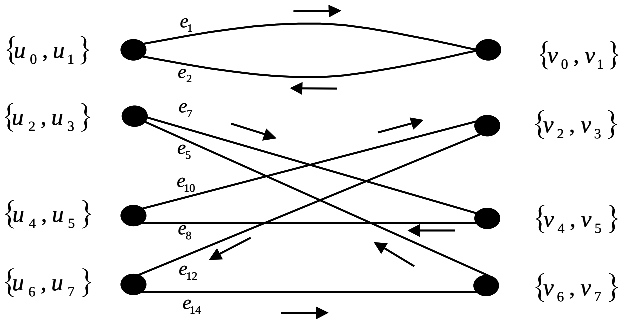

Fig.4 The first bipartite graph and edge traverses of the second-level decomposition

Then the pairs e_1, e_7, e_{10} and e_{14} corresponding to the edges in E_1 form

$$\begin{pmatrix} 1 & 7 & 10 & 14 \\ 2 & 11 & 6 & 15 \end{pmatrix} \tag{6}$$

and the pairs e_2, e_5, e_8 and e_{12} corresponding to the edges in E_2 form

$$\begin{pmatrix} 2 & 5 & 8 & 12 \\ 0 & 13 & 9 & 5 \end{pmatrix} \tag{7}$$

For the first-level partial permutation (5) the bipartite graph and edge traverses are shown in Figure 5.

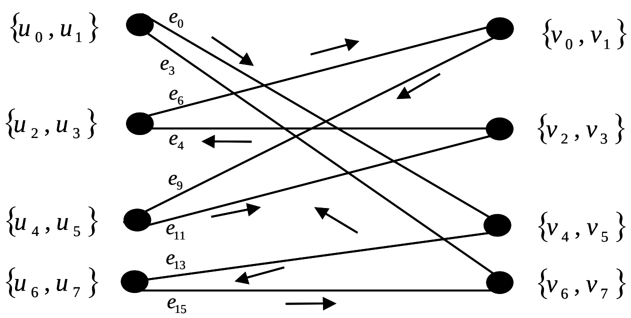

Fig.5 The second bipartite graph and edge traverses of the second-level decomposition

Then the pairs e_0, e_6, e_{11} and e_{15} corresponding to the edges in E_1 form

$$\begin{pmatrix} 0 & 6 & 11 & 15 \\ 8 & 3 & 7 & 14 \end{pmatrix} \tag{8}$$

and the pairs e_3, e_4, e_9 and e_{13} corresponding to the edges in E_2 form

$$\begin{pmatrix} 3 & 4 & 9 & 13 \\ 12 & 4 & 1 & 10 \end{pmatrix} \tag{9}$$

which completes the whole decomposition and the four CFPPs are just the partial permutations (6) to (9). By realizing each of the CFPPs (6) to (9) in a single plane of a RB(N) network consisting of four planes, the full permutation in Example 1 can be realized crosstalk-free in the network in a single pass.

In general, there are many kinds of CFPPs for a permutation. The results of Algorithm 1 and Theorem 2 show that any permutation of N-element set {0,1,…, N-1} can be decomposed into $2^{\lfloor (n+1)/2 \rfloor}$ CFPPs and each of these CFPPs consists of $N/2^{\lfloor (n+1)/2 \rfloor}$ mapping pairs. To avoid confusion, we will refer to this special kind of CFPPs as the *Specified Crosstalk-Free Partial Permutation* (SCFPPs) of a permutation. It is easy to see that the CFPPs of a permutation obtained by Algorithm 1 over the permutation are just the SCFPPs of the permutation.

5. A new scheme for crosstalk-free decomposition

As discussed in Section 4, the overall time complexity to realize a permutation crosstalk-free in a RB(N) network is dominated by the time complexity of the CFPP decomposition algorithm, which is $O(N \log_2 N)$. Thus, the performance of high speed optical MIN can be enhanced significantly if the time complexity of decomposition algorithm can be reduced. In this section, we present a new scheme for crosstalk-free decomposition. The basic idea of the scheme is to classify permutations into permutation classes such that all permutations in one class share the same crosstalk-free decomposition pattern. By running the Euler-Split based algorithm 1 only once for a permutation class and applying the obtained crosstalk-free decomposition patterns to all permutations in the class, crosstalk-free decomposition of permutations can be realized in a more efficient way.

5.1 Permutation pattern and permutation class

To explore the class of permutations that have similar crosstalk-free decompositions, we start with the following definition based on the crosstalk-free condition of an entire banyan-type network as described in Lemma 1.

Definition 2: For a permutation of form (2), we set $i = \lfloor (n+1)/2 \rfloor - 1$. Then a specified undirected bipartite graph G^* = $(V_1, V_2 ; E)$ can be constructed for the permutation as that of step 3 in Algorithm 1. We define the topology pattern (ignoring the details of mapping pairs) of the bipartite graph $G^* = (V_1, V_2; E)$ to be the *permutation pattern* of the permutation. If we let the vertices of V_1 correspond to the rows of a matrix and the vertices of V_2 correspond to the columns of the matrix, then the permutation pattern can also be expressed as a matrix $M_{PP} = (a_{jk})_{\frac{N}{2^{i+1}} \times \frac{N}{2^{i+1}}}$ with its entry a_{jk} being the number of edge(s) from j-th vertex in V_1 to k-th vertex in V_2. We define the matrix M_{PP} as the *permutation pattern matrix* (PPM) of the given permutation. Furthermore, we can also construct a matrix $M_{DPP} = (b_{jk})_{\frac{N}{2^{i+1}} \times \frac{N}{2^{i+1}}}$ with its entry b_{jk} being the set of the mapping pair(s) corresponding to the edge(s) from j-th vertex in V_1 to k-th vertex in V_2. We call the matrix $M_{DPP} = (b_{jk})_{\frac{N}{2^{i+1}} \times \frac{N}{2^{i+1}}}$ the *detailed permutation pattern matrix* (DPPM).

Example 3. The permutation pattern, PPM and DPPM of permutation (3) in Example 1.
According to the definition, the permutation pattern of permutation (3) is illustrated in Figure 6.

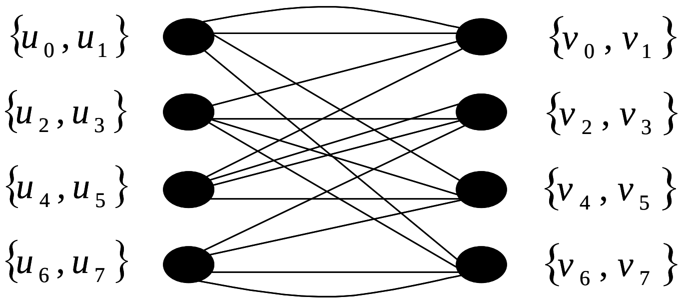

Fig.6: The permutation pattern of permutation (3)

The corresponding PPM of the permutation is:

$$M_{PP} = \begin{bmatrix} 2 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

If we number the mapping pairs as $e_j = \begin{pmatrix} x_j \\ y_j \end{pmatrix}$ for $0 \le j \le N\text{-}1$, the DPPM of the permutation is thus:

$$M_{DPP} = \begin{bmatrix} \{e_1, e_2\} & \{-\} & \{e_0\} & \{e_3\} \\ \{e_6\} & \{e_4\} & \{e_7\} & \{e_5\} \\ \{e_9\} & \{e_{10}, e_{11}\} & \{e_8\} & \{-\} \\ \{-\} & \{e_{12}\} & \{e_{13}\} & \{e_{14}, e_{15}\} \end{bmatrix}$$

Here {-} means a null set.

Since a permutation pattern is completely specified by its PPM, we have:

Corollary 1: Two permutations have the same permutation pattern if and only if their PPMs are identical.

We are now in the position to introduce the definition of permutation class.

Definition 3: We define the set of all the permutations corresponding to the same PPM (permutation pattern matrix) to be a *permutation class*.

It is should be noted that all permutations in a class share a common PPM, but each permutation has a distinct DPPM.

About the number of permutations in a permutation class, we have the following result:

Theorem 3: In a permutation class of {0,1,…, N-1}, there are at least $\left(\sqrt{N}!\right)^{\sqrt{N}}$ permutations if $\log_2 N$ is even, and at least $\left(\sqrt{2N}!\right)^{\sqrt{\frac{N}{2}}}$ permutations if $\log_2 N$ is odd.

Proof: We only prove for the case when $\log_2 N$ is even, the odd case can be proven in a similar way. Let the permutation be the form of

$$\begin{pmatrix} x_0, x_1, \cdots, x_{N-1} \\ y_0, y_1, \cdots, y_{N-1} \end{pmatrix}$$

where $x_j = j$ for $0 \le j \le N-1$ and $\{y_0, y_1, \ldots, y_{N-1}\} = \{0,1,\ldots,N-1\}$.

When $\log_2 N$ is even, we decompose the permutation into $\sqrt{N}$ $\sqrt{N}$ -element partial permutations

$$\begin{pmatrix} x_0 \cdots x_{\sqrt{N}-1} \\ y_0 \cdots y_{\sqrt{N}-1} \end{pmatrix}, \begin{pmatrix} x_{\sqrt{N}} \cdots x_{\sqrt{N}+\sqrt{N}-1} \\ y_{\sqrt{N}} \cdots y_{\sqrt{N}+\sqrt{N}-1} \end{pmatrix}, \ldots, \begin{pmatrix} x_{(\sqrt{N}-1)\sqrt{N}} \cdots x_{N-1} \\ y_{(\sqrt{N}-1)\sqrt{N}} \cdots y_{N-1} \end{pmatrix}$$

By the definition of permutation pattern, two permutations have the same permutation pattern if and only if they have the same sets $\{y_0, \cdots, y_{\sqrt{N}-1}\}$, $\{y_{\sqrt{N}}, \cdots, y_{\sqrt{N}+\sqrt{N}-1}\}, \ldots,$ $\{y_{(\sqrt{N}-1)\sqrt{N}}, \cdots, y_{N-1}\}$. If we fix the connections between the input ports and the link of permutation pattern (i.e. neglecting the permutations of input ports in each of the above partial permutations), the permutations corresponding to a permutation pattern will be only determined by both $\sqrt{N}$ sets $\{y_0, \cdots, y_{\sqrt{N}-1}\}$, $\{y_{\sqrt{N}}, \cdots, y_{\sqrt{N}+\sqrt{N}-1}\}, \ldots,$ $\{y_{(\sqrt{N}-1)\sqrt{N}}, \cdots, y_{N-1}\}$ and the permutations of the $\sqrt{N}$ elements in each of these $\sqrt{N}$ sets, and there are in total $\left(\sqrt{N}!\right)^{\sqrt{N}}$ permutations of N-element set {0,1,…, N-1} that correspond to sets $\{y_0, \cdots, y_{\sqrt{N}-1}\}$, $\{y_{\sqrt{N}}, \cdots, y_{\sqrt{N}+\sqrt{N}-1}\}, \ldots,$ $\{y_{(\sqrt{N}-1)\sqrt{N}}, \cdots, y_{N-1}\}$ (permutation pattern). Thus, there are at least $\left(\sqrt{N}!\right)^{\sqrt{N}}$ permutations on {0,1,…, N-1} in one permutation class.

QED.

5.2 CFPP matrices of a permutation

As indicated in the decomposition Algorithm 1 and Definition 2, the permutation pattern of a permutation of N-element set {0,1,…, N-1} can be decomposed into $2^{\lfloor (n+1)/2 \rfloor}$ disjoint sub-patterns, and each of these sub-patterns is a perfect match of the specified bipartite graph $G^* = (V_1, V_2; E)$ of the permutation and corresponds to a SCFPP of the permutation. Since a permutation pattern is completely specified by its permutation pattern matrix (PPM) defined in Definition 2, a sub-pattern discussed above will also be completely specified by a matrix defined in the same way as that of PPM. We introduce the following definition to characterize the matrix defined for a sub-pattern.

Definition 4: We define the matrices, which are defined by the $2^{\lfloor (n+1)/2 \rfloor}$ disjoint sub-patterns decomposed from the permutation pattern of a permutation on {0,1,…, N-1} by using Algorithm 1, to be the *CFPP matrices* of the permutation.

We have the following results regarding the CFPP matrices of a permutation:
Lemma 2: For any permutation, each of its CFPP matrices is a permutation matrix, and the sums of its CFPP matrices are just the PPM of the permutation.

Proof: Let matrix $M_{CFPP} = (c_{jk})_{\frac{N}{2^{\lfloor (n+1)/2 \rfloor}} \times \frac{N}{2^{\lfloor (n+1)/2 \rfloor}}}$ be the CFPP matrix corresponding to one of the $2^{\lfloor (n+1)/2 \rfloor}$ disjoint sub-patterns decomposed from the permutation pattern of a permutation by using Algorithm 1. Since a CFPP matrix of a permutation is defined in the same way as that of PPM of the permutation, entry c_{jk} is the number of edge(s) from the j-th vertex of V_1 to the k-th vertex of V_2 in the specified bipartite graph $G^* = (V_1,V_2 ; E)$ defined for the permutation. From Algorithm 1, we know that the set of all edges in the sub-pattern is a perfect match of the bipartite graph $G^* = (V_1,V_2 ; E)$, there will be exactly one unit entry in each row and each column of the matrix $M_{CFPP} = (c_{jk})_{\frac{N}{2^{\lfloor (n+1)/2 \rfloor}} \times \frac{N}{2^{\lfloor (n+1)/2 \rfloor}}}$, and 0 in all other entries, so the CFPP matrix $M_{CFPP} = (c_{jk})_{\frac{N}{2^{\lfloor (n+1)/2 \rfloor}} \times \frac{N}{2^{\lfloor (n+1)/2 \rfloor}}}$ is a permutation matrix.

The decomposition Algorithm 1 indicates that the permutation pattern of a permutation of N-element set {0,1,…, N-1} can be decomposed into $2^{\lfloor (n+1)/2 \rfloor}$ disjoint sub-patterns, each of which is completely specified by a permutation matrix. Let the matrix $M_{sum} = (s_{jk})_{\frac{N}{2^{\lfloor (n+1)/2 \rfloor}} \times \frac{N}{2^{\lfloor (n+1)/2 \rfloor}}}$ be the sums of these $2^{\lfloor (n+1)/2 \rfloor}$ permutation matrixes, entry s_{jk} is then the total number of edge(s) from the j-th vertex of V_1 to the k-th vertex of V_2 in $G^* = (V_1,V_2 ; E)$. By the definition of PPM of the permutation, $M_{pp} = (a_{jk})_{\frac{N}{2^{\lfloor (n+1)/2 \rfloor}} \times \frac{N}{2^{\lfloor (n+1)/2 \rfloor}}}$, we know that $s_{jk} = a_{jk}$. Thus, the PPM of a permutation is the sum of the CFPP matrices of the permutation.
QED.

Since all permutations in one class have a same permutation pattern and the CFPP matrices of a permutation are only determined by its permutation pattern, we have:

Corollary 2: All permutations in one permutation class have the same set of CFPP matrices.

5.3 Computing the CFPP matrices for a permutation class

Since all the permutations in one permutation class have a same permutation pattern and same set of CFPP matrices, we can randomly select one permutation from the class to get the CFPP matrices for the class. After we run the decomposition Algorithm 1 over the selected permutation, we can get $2^{\lfloor (n+1)/2 \rfloor}$ SCFPPs of the permutation and $2^{\lfloor (n+1)/2 \rfloor}$ disjoint sub-patterns of the specified bipartite graph $G^* = (V_1,V_2 ; E)$ defined for the permutation, and each of these sub-patterns corresponds to a SCFPP. Then the $2^{\lfloor (n+1)/2 \rfloor}$ CFPP matrices of the permutation can be constructed based on these $2^{\lfloor (n+1)/2 \rfloor}$ SCFPPs, as that of constructing PPM based on the permutation. That is, let the vertices of V_1 and V_2 in the SCFPP correspond to the rows and columns of the CFPP matrix respectively, the CFPP matrix corresponding to the SCFPP will be the matrix $M_{CFPP} = (c_{jk})_{\frac{N}{2^{\lfloor (n+1)/2 \rfloor}} \times \frac{N}{2^{\lfloor (n+1)/2 \rfloor}}}$ with its entry c_{jk} being the number of mapping pair(s) from j-th vertex in V_1 to k-th vertex in V_2.
Example 4 Getting the CFPP matrices for the permutation class containing permutation (3).
As discussed in example 2, the 4 SCFPPs of the permutation (3) are:

$$\begin{pmatrix} 1 & 7 & 10 & 14 \\ 2 & 11 & 6 & 15 \end{pmatrix}, \begin{pmatrix} 2 & 5 & 8 & 12 \\ 0 & 13 & 9 & 5 \end{pmatrix}, \begin{pmatrix} 0 & 6 & 11 & 15 \\ 8 & 3 & 7 & 14 \end{pmatrix}, \begin{pmatrix} 3 & 4 & 9 & 13 \\ 12 & 4 & 1 & 10 \end{pmatrix}$$

Based on these 4 SCFPPs, the 4 CFPP matrices of the permutation can be constructed as:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Since all the permutations in one permutation class have the same set of CFPP matrices, the four CFPP matrices obtained for permutation (3) are just the CFPP matrices of the permutation class containing permutation (3). We can see that every matrix above is just a permutation matrix and the sum of these 4 permutation matrices is the PPM of the permutation (3) as given in example 3.

It is easy to verify that the following permutation has the same PPM as that of permutation (3), thus they belong to the same permutation class.

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 2 & 12 & 8 & 0 & 10 & 13 & 3 & 6 & 1 & 9 & 5 & 4 & 14 & 11 & 15 & 7 \end{pmatrix} \tag{10}$$

Then permutation (10) has the same set of CFPP matrices as that give in example 4.

Noted that every CFPP matrix of a class is just a permutation matrix, and a permutation matrix can be completely specified by the positions of unit entries in the matrix. So we actually only need to calculate and keep the positions of unit entries in the CFPP matrices of a class.

Definition 5: For a permutation class and its CFPP matrices, we construct a matrix such that each row of the matrix contains the positions of unit entries in a CFPP matrix. We define the matrix as the *compact CFPP matrix* of the class.

We present here an algorithm for getting the compact CFPP matrix of a class.

Algorithm 2 Getting the compact CFPP matrix for a class

Step 1:For a permutation class of N-element set {0,1,…, N-1}, we set $i = \lfloor (n+1)/2 \rfloor - 1$ and take a permutation from the class. Express the permutation in form of $(e_0, \ e_1, \ \cdots, \ e_{N-1})$ with $e_j = \begin{pmatrix} x_j \\ y_j \end{pmatrix}$ being the mapping pair.

Step 2: Run the decomposition algorithm 1over the permutation, and summarize the SCFPPs obtained as a matrix:

$$M_1 = \begin{bmatrix} e_{d_0} & e_{d_1} & \cdots & e_{d_{\frac{N}{2^{i+1}}-1}} \\ e_{d_{\frac{N}{2^{i+1}}}} & e_{d_{\frac{N}{2^{i+1}}+1}} & \cdots & e_{d_{2\times\frac{N}{2^{i+1}}-1}} \\ \cdots & \cdots & \cdots & \cdots \\ e_{d_{\left(2^{i+1}-1\right)\times\frac{N}{2^{i+1}}}} & e_{d_{\left(2^{i+1}-1\right)\times\frac{N}{2^{i+1}}+1}} & \cdots & e_{d_{N-1}} \end{bmatrix}$$

Here $\{d_0,d_1,\ldots,d_{N-1}\}=\{0,1,\ldots,N-1\}$, and each row of the matrix corresponds to a SCFPP of the original permutation..

Step3: Get the following matrix based on matrix M_1.

$$M_2 = \begin{bmatrix} \left(\left\lfloor \frac{x_{d_0}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_0}}{N/2^{i+1}} \right\rfloor\right) & \left(\left\lfloor \frac{x_{d_1}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_1}}{N/2^{i+1}} \right\rfloor\right) & \cdots & \left(\left\lfloor \frac{x_{d_{N/2^{i+1}-1}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{N/2^{i+1}-1}}}{N/2^{i+1}} \right\rfloor\right) \\ \left(\left\lfloor \frac{x_{d_{N/2^{i+1}}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{N/2^{i+1}}}}{N/2^{i+1}} \right\rfloor\right) & \left(\left\lfloor \frac{x_{d_{N/2^{i+1}+1}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{N/2^{i+1}+1}}}{N/2^{i+1}} \right\rfloor\right) & \cdots & \left(\left\lfloor \frac{x_{d_{2N/2^{i+1}-1}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{2N/2^{i+1}-1}}}{N/2^{i+1}} \right\rfloor\right) \\ \cdots & \cdots & \cdots & \cdots \\ \left(\left\lfloor \frac{x_{d_{(2^{i+1}-1)\frac{N}{2^{i+1}}}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{(2^{i+1}-1)\frac{N}{2^{i+1}}}}}{N/2^{i+1}} \right\rfloor\right) & \left(\left\lfloor \frac{x_{d_{(2^{i+1}-1)\frac{N}{2^{i+1}}+1}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{(2^{i+1}-1)\frac{N}{2^{i+1}}+1}}}{N/2^{i+1}} \right\rfloor\right) & \cdots & \left(\left\lfloor \frac{x_{d_{N-1}}}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_{d_{N-1}}}{N/2^{i+1}} \right\rfloor\right) \end{bmatrix}$$

Note that M_2 is converted from M_1 by replacing each mapping pair $e_j = \begin{pmatrix} x_j \\ y_j \end{pmatrix}$ in M_1 with the number pair $\left(\left\lfloor \frac{x_j}{N/2^{i+1}} \right\rfloor, \left\lfloor \frac{y_j}{N/2^{i+1}} \right\rfloor\right)$. The number pairs in each row of M_2 are just the positions of the units in the CFPP matrix corresponding to the SCFPP in the same row of M_1. Thus, M_2 is just the compact CFPP matrix of the class.

Example 5 Computation of the compact CFPP matrix for the permutation class containing permutation (3).

From example 2 we know that after we run the decomposition algorithm 1 over permutation (3), the four SCFPPs obtained can be summarized as the following matrix:

$$M_1 = \begin{bmatrix} e_1 & e_7 & e_{10} & e_{14} \\ e_2 & e_5 & e_8 & e_{12} \\ e_0 & e_6 & e_{11} & e_{15} \\ e_3 & e_4 & e_9 & e_{13} \end{bmatrix}$$

Here we number the mapping pair as $e_j = \begin{pmatrix} j \\ - \end{pmatrix}$ for $0 \le j \le 15$. The corresponding compact CFPP matrix will be:

$$M_2 = \begin{bmatrix} \left(\left\lfloor \frac{1}{4} \right\rfloor, \left\lfloor \frac{2}{4} \right\rfloor\right) & \left(\left\lfloor \frac{7}{4} \right\rfloor, \left\lfloor \frac{11}{4} \right\rfloor\right) & \left(\left\lfloor \frac{10}{4} \right\rfloor, \left\lfloor \frac{6}{4} \right\rfloor\right) & \left(\left\lfloor \frac{14}{4} \right\rfloor, \left\lfloor \frac{15}{4} \right\rfloor\right) \\ \left(\left\lfloor \frac{2}{4} \right\rfloor, \left\lfloor \frac{0}{4} \right\rfloor\right) & \left(\left\lfloor \frac{5}{4} \right\rfloor, \left\lfloor \frac{13}{4} \right\rfloor\right) & \left(\left\lfloor \frac{8}{4} \right\rfloor, \left\lfloor \frac{9}{4} \right\rfloor\right) & \left(\left\lfloor \frac{12}{4} \right\rfloor, \left\lfloor \frac{5}{4} \right\rfloor\right) \\ \left(\left\lfloor \frac{0}{4} \right\rfloor, \left\lfloor \frac{8}{4} \right\rfloor\right) & \left(\left\lfloor \frac{6}{4} \right\rfloor, \left\lfloor \frac{3}{4} \right\rfloor\right) & \left(\left\lfloor \frac{11}{4} \right\rfloor, \left\lfloor \frac{7}{4} \right\rfloor\right) & \left(\left\lfloor \frac{15}{4} \right\rfloor, \left\lfloor \frac{14}{4} \right\rfloor\right) \\ \left(\left\lfloor \frac{3}{4} \right\rfloor, \left\lfloor \frac{12}{4} \right\rfloor\right) & \left(\left\lfloor \frac{4}{4} \right\rfloor, \left\lfloor \frac{4}{4} \right\rfloor\right) & \left(\left\lfloor \frac{9}{4} \right\rfloor, \left\lfloor \frac{1}{4} \right\rfloor\right) & \left(\left\lfloor \frac{13}{4} \right\rfloor, \left\lfloor \frac{10}{4} \right\rfloor\right) \end{bmatrix} = \begin{bmatrix} (0,0) & (1,2) & (2,1) & (3,3) \\ (0,0) & (1,3) & (2,2) & (3,1) \\ (0,2) & (1,0) & (2,1) & (3,3) \\ (0,3) & (1,1) & (2,0) & (3,2) \end{bmatrix}$$

Since all the permutations in one permutation class have the same set of CFPP matrices and thus same compact CFPP matrix, the compact CFPP matrix obtained for permutation (3) is just the compact CFPP matrix of the permutation class containing permutation (3). Note that the four rows in compact CFPP matrix M_2 correspond to the positions of unit of four CFPP matrices for the permutation (3).

The results of Theorem 3 indicate that there are a huge number of permutations in one permutation class, and all these permutations have the same permutation pattern and thus same crosstalk-free decompositions. In the following, we will present an approach for crosstalk-free decomposition of permutations in one class. The approach consists of two integrated parts: Getting the compact CFPP matrix for a class by running the decomposition algorithm 1 over one of the permutation in the class and applying the obtained compact CFPP matrix to any new permutation in the class to get the crosstalk-free decomposition of the new permutation.

5.4 Crosstalk-free decomposition based on the compact CFPP matrix of a class

By the definition of detailed permutation pattern matrix (DPPM) and permutation pattern matrix (PPM), we know that there is a one-to-one mapping between a permutation pattern and a PPM, and there is a one-to-one mapping between a permutation and a DPPM. For a permutation, its PPM has the same structure as that of its DPPM except that every entry in PPM is only the number of mapping pair(s) of the corresponding entry in DPPM. All the permutations in one class have a common PPM and permutation pattern, but every permutation in a class has its distinct DPPM.

For a permutation of N-element set {0,1,…, N-1}, we can get its $2^{\lfloor (n+1)/2 \rfloor}$ SCFPPs based on its $2^{\lfloor (n+1)/2 \rfloor}$ CFPP matrices as follows: For the first CFPP matrix, we take one mapping pair from each position in the DPPM of the permutation, where 1 occurs in the same position in this CFPP matrix, and then get a set of mapping pairs corresponding to the first CFPP matrix. For the second CFPP matrix, we take one mapping pair from each position in the DPPM, where 1 occurs in the same position in this second CFPP matrix, and then get a set of mapping pairs corresponding to the second CFPP matrix. We repeat this until the last CFPP matrix. Its easy to see that each set of mapping pairs corresponding to each CFPP matrix is just a SCFPP because of the definitions of CFPP and SCFPP. Since the DPPM of a permutation is specified by the PPM of the permutation, and the sums of all the CFPP matrices of the permutation is just the PPM of the permutation, the above process works for any permutation if we obtain its CFPP matrices.

Noted that all the permutations in one class have the same set of CFPP matrices, and once we get the CFPP matrices of a class, we can get the SCFPPs of any permutation as discussed above.

Example 6 Crosstalk-free decomposition of the permutation (10) based on its CFPP matrices.

Since the permutation (10) has the same PPM as that of permutation (3), they belong to the same class and have the same CFPP matrices. From example 4, we know that the four CFPP matrices of the permutation are:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

If we number the mapping pairs as $e_j = \begin{pmatrix} j \\ - \end{pmatrix}$ for $0 \le j \le 15$, the DPPM of the permutation is:

$$M_{DPP} = \begin{bmatrix} \{e_0, e_3\} & \{-\} & \{e_2\} & \{e_1\} \\ \{e_6\} & \{e_7\} & \{e_4\} & \{e_5\} \\ \{e_8\} & \{e_{10}, e_{11}\} & \{e_9\} & \{-\} \\ \{-\} & \{e_{15}\} & \{e_{13}\} & \{e_{12}, e_{14}\} \end{bmatrix}$$

After we apply the four CFPP matrices to the DPPM one by one, we can get:

$$\begin{bmatrix} \{e_0\} & 0 & 0 & 0 \\ 0 & 0 & \{e_4\} & 0 \\ 0 & \{e_{10}\} & 0 & 0 \\ 0 & 0 & 0 & \{e_{12}\} \end{bmatrix}, \begin{bmatrix} \{e_3\} & 0 & 0 & 0 \\ 0 & 0 & 0 & \{e_5\} \\ 0 & 0 & \{e_9\} & 0 \\ 0 & \{e_{15}\} & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & \{e_2\} & 0 \\ \{e_6\} & 0 & 0 & 0 \\ 0 & \{e_{11}\} & 0 & 0 \\ 0 & 0 & 0 & \{e_{14}\} \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & \{e_1\} \\ 0 & \{e_7\} & 0 & 0 \\ \{e_8\} & 0 & 0 & 0 \\ 0 & 0 & \{e_{13}\} & 0 \end{bmatrix}$$

Then the four SCFPPs of the permutation are:

$$\begin{pmatrix} 0 & 4 & 10 & 12 \\ 2 & 10 & 5 & 14 \end{pmatrix}, \begin{pmatrix} 3 & 5 & 9 & 15 \\ 0 & 13 & 9 & 7 \end{pmatrix}, \begin{pmatrix} 2 & 6 & 11 & 14 \\ 8 & 3 & 4 & 15 \end{pmatrix}, \begin{pmatrix} 1 & 7 & 8 & 13 \\ 12 & 6 & 1 & 11 \end{pmatrix}.$$

The Algorithm 2 indicates that all the CFPP matrices of a permutation class can be summarized as a single compact CFPP matrix, so the crosstalk-free decomposition of the permutation can also be implemented by using its compact CFPP matrix. We present here an efficient algorithm to decompose any permutation of a class into SCFPPs based on the compact CFPP matrix of the class.

Algorithm 3 Crosstalk-free Decomposition Based on Compact CFPP Matrix of a Class
Step 1: For a permutation in the permutation class of N-element set {0,1,…, N-1}, construct the DPPM of the permutation.
Step 2: For every row in the compact CFPP matrix of the class, we take the mapping pairs from the DPPM of the permutation according to the positions indicated in the row of the compact CFPP matrix, and then get a set of mapping pairs corresponding to a row of the compact CFPP matrix. Each set of mapping pairs obtained is just a SCFPP of the permutation.

The above algorithm is correct by the meanings of SCFPP and compact CFPP matrices. Since Step 1 takes $O(N)$ time to construct the DPPM of a permutation, and Step 2 takes $O(N)$ time to get the SCFPPs of the permutation based on its DPPM and compact CFPP matrices, the time complexity of the algorithm is $O(N)$.

Example 7 Crosstalk-free decomposition of permutation (10) based on its compact CFPP matrix.

Since permutation (10) has the same PPM as that of permutation (3), they belong to the same class and thus have the same compact CFPP matrix. From example 5, we know that the compact CFPP matrix of the permutation is:

$$M_2 = \begin{bmatrix} (0,0) & (1,2) & (2,1) & (3,3) \\ (0,0) & (1,3) & (2,2) & (3,1) \\ (0,2) & (1,0) & (2,1) & (3,3) \\ (0,3) & (1,1) & (2,0) & (3,2) \end{bmatrix}$$

If we number the mapping pairs as $e_j = \begin{pmatrix} j \\ - \end{pmatrix}$ for $0 \le j \le 15$, the DPPM of the permutation will be:

$$M_{DPP} = \begin{bmatrix} \{e_0, e_3\} & \{-\} & \{e_2\} & \{e_1\} \\ \{e_6\} & \{e_7\} & \{e_4\} & \{e_5\} \\ \{e_8\} & \{e_{10}, e_{11}\} & \{e_9\} & \{-\} \\ \{-\} & \{e_{15}\} & \{e_{13}\} & \{e_{12}, e_{14}\} \end{bmatrix}$$

After we apply the compact CFPP matrix to the DPPM according to algorithm 3, we get immediately the four SCFPPs of the permutation: $(e_0, e_4, e_{10}, e_{12}), (e_3, e_5, e_9, e_{15}), (e_2, e_6, e_{11}, e_{14}), (e_1, e_7, e_8, e_{13})$, where $e_0 = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$,

$e_1 = \begin{pmatrix} 1 \\ 12 \end{pmatrix}, e_2 = \begin{pmatrix} 2 \\ 8 \end{pmatrix}, e_3 = \begin{pmatrix} 3 \\ 0 \end{pmatrix}, e_4 = \begin{pmatrix} 4 \\ 10 \end{pmatrix}, e_5 = \begin{pmatrix} 5 \\ 13 \end{pmatrix}, e_6 = \begin{pmatrix} 6 \\ 3 \end{pmatrix}, e_7 = \begin{pmatrix} 7 \\ 6 \end{pmatrix}, e_8 = \begin{pmatrix} 8 \\ 1 \end{pmatrix}, e_9 = \begin{pmatrix} 9 \\ 9 \end{pmatrix}, e_{10} = \begin{pmatrix} 10 \\ 5 \end{pmatrix}, e_{11} = \begin{pmatrix} 11 \\ 4 \end{pmatrix}$,
$e_{12} = \begin{pmatrix} 12 \\ 14 \end{pmatrix}, e_{13} = \begin{pmatrix} 13 \\ 11 \end{pmatrix}, e_{14} = \begin{pmatrix} 14 \\ 15 \end{pmatrix}, e_{15} = \begin{pmatrix} 15 \\ 7 \end{pmatrix}$.

5.5 Numbering the permutation class

Since a permutation class is identified by its permutation pattern matrix (PPM), we can number each permutation class by assigning a distinct number to its distinct PPM. For a permutation class of N-element set {0,1,…, N-1}and if $\log N$ is even, its PPM will be in the form $M_{PP} = (a_{jk})_{\sqrt{N} \times \sqrt{N}}$ with $0 \le a_{jk} \le \sqrt{N}$. Then we can assign the PPM a distinct number as:

$$\sum_{j=0}^{\sqrt{N}-1} \sum_{k=0}^{\sqrt{N}-1} a_{jk} \cdot \sqrt{N}^{\, j \cdot \sqrt{N} + k} \qquad (11)$$

When log N is odd, the distinct number can be evaluated in a similar way. Noted that the numbering of a PPM can be finished in $O(N)$ time.

5.6 The overall decomposition algorithm based on permutation class

The results in Section 5.3 indicate that we only need to run the decomposition Algorithm 1 once for a permutation class to get the compact CFPP matrix of the class, then the crosstalk-free decomposition of any permutation in the class can be implemented in a simple way as discussed in Section 5.4.

Since a permutation class is identified by its permutation pattern matrix (PPM), we can determine a class for a permutation by checking its PPM. We are now in the position to give a high-level description of the overall decomposition algorithm based on the idea of permutation class.

Algorithm 4 Crosstalk-free Decomposition based on Permutation Class

Step 1: For a new permutation of N-element set {0,1,…, N-1}, construct its PPM and calculate its class number.

Step 2: Check the address corresponding to the class number. Go to Step 3 if the compact CFPP matrix is available in the address, go to Step 4 otherwise.

Step 3: Use Algorithm 3 in Section 5.4 to get the SCFPPs of the permutation.

Step 4: Use Steps 2-3 in Algorithm 2 to get the SCFPPs and compact CFPP matrix of the permutation, and save the compact CFPP matrix to the address corresponding to the class number.

Since the Steps 1-3 take $O(N)$ time and Step 4 takes $O(N\log N)$ time, then we have the following results regarding time complexity of Algorithm 4.

Theorem 5: To decompose a permutation of N-element set {0,1,…, N-1} into SCFPPs based on Algorithm 4, the average time complexity is $O(N)$.

Proof: We only prove for the case when $\log_2 N$ is even, the odd case can be proven in a similar way. If $\log_2 N$ is even, each permutation class consists of at least $\left(\sqrt{N}!\right)^{\sqrt{N}}$ permutations as shown in Theorem 3, so there are at most $N!/\left(\sqrt{N}!\right)^{\sqrt{N}}$ classes corresponding all permutations. Since we need $O(N\log N)$ time to get the compact CFPP matrix for each permutation class, and then the decomposition of any permutation can be finished in $O(N)$ time, the overall time complexity of crosstalk-free decomposition of all permutations will be upper bounded by the case when each class contains $\left(\sqrt{N}!\right)^{\sqrt{N}}$ permutations. In this case, there are $N!/\left(\sqrt{N}!\right)^{\sqrt{N}}$ classes and the decompositions of $\left(\sqrt{N}!\right)^{\sqrt{N}}-1$ permutations in each class can be obtained in $O(N)$ time. Thus, the overall time to decompose all permutations will be upper bounded by:

$$\frac{N!}{\left(\sqrt{N}!\right)^{\sqrt{N}}}\cdot O(N\log N)+\frac{N!}{\left(\sqrt{N}!\right)^{\sqrt{N}}}\cdot\left[\left(\sqrt{N}!\right)^{\sqrt{N}}-1\right]\cdot O(N)$$

Therefore, the average time for each permutation is upper bounded by:

$$\frac{\frac{N!}{\left(\sqrt{N}!\right)^{\sqrt{N}}}\cdot O(N\log N)+\frac{N!}{\left(\sqrt{N}!\right)^{\sqrt{N}}}\cdot\left[\left(\sqrt{N}!\right)^{\sqrt{N}}-1\right]\cdot O(N)}{N!}=O(N)\cdot\left[1+\frac{\log N}{\left(\sqrt{N}!\right)^{\sqrt{N}}}-\frac{1}{\left(\sqrt{N}!\right)^{\sqrt{N}}}\right]\sim O(N)$$

Since there are N elements in each permutation, then a lower bound of time complexity of decomposing a permutation is $O(N)$. Thus, the average time for each permutation is $O(N)$.

Since a RB(N) network consists of multi-copies of Banyan-type networks which are self-routing, so the routing complexity is $O(\log_2 N)$.Thus, the overall time complexity of the new scheme is $O(N)+O(\log_2 N)\sim O(N)$ to realize a permutation crosstalk-free in a RB(N) network.

6. Conclusions

In this paper, we have proposed a new scheme for crosstalk-free realization of permutations in rearrangeably and self-routing optical MINs under the vertical stacking scheme. We have introduced the idea of permutation class in the scheme to make the crosstalk-free decomposition permutations more efficient. We have shown that there is a large number of permutations in a permutation class that share the same permutation pattern and thus the same crosstalk-free decompositions. By getting the crosstalk-free decomposition pattern of a class based on the Euler-split technique and applying it to all the permutations in the class, the crosstalk-free decompositions of these permutations can be made more efficient. Our new scheme has an average time complexity of $O(N)$ instead of $O(N\log N)$ required previously for realizing any crosstalk-free permutation of $\{0,1,\ldots, N\text{-}1\}$ in this type of optical MINs.

References

[1] V.R.Chinni et al.: Crosstalk in a lossy directional coupler switch, Journal of Lightwave Technology, vol.13, no.7, (July 1995), pp.1530-1535.
[2] H.S.Hinton: An introduction to Photonic Switching Fabrics, New York: Plenum, 1993.
[3] G.R.Goke and G.J.Lipovski: Banyan networks for partitioning multiprocessor systems, Proc.1st Annu. Symp. Comp. Arch (1973), pp.21-28.
[4] B.Kolman, R.C.Busby and S.Ross: Discrete Mathematical Structures, 3nd Edition, Prentice Hall, 1996.
[5] C.Kruskal and M.Snir: The performance of multistage interconnection networks for multiprocessors, IEEE Trans. Commun., vol.COM-32, (Dec.1983), pp.1091-1098.
[6] C.-T. Lea: Muti-$\log_2 N$ networks and their applications in high speed electronic and photonic switching systems, IEEE Trans. Commun., vol.38, (Oct. 1990), pp.1740-1749.
[7] C.-T. Lea: Crossover minimization in directional coupler-based photonic switching systems, IEEE Trans. Commun., vol.36, (Mar.1988), pp.355-363.
[8] J.Patel: Performance of processor-memory interconnections for multiprocessors, IEEE Trans. Comput., vol.C-30, (Oct.1981), pp.771-780.
[9] Xiaohong Jiang, Md. Mamun-ur-Rashid Khandker and S.Horiguchi: Nonblocking Optical MINs Under Crosstalk-free Constraint, Proceedings of the 2001 IEEE Workshop on High Performance Switching and Routing (Dallas, USA, May.2001), pp.307-311.
[10]M.M.Vaez and C.-T. Lea: Strictly nonblocking directional-coupler-based switching networks under crosstalk constraint, IEEE Trans. Commun., vol.48,no.2, (Feb. 2000), pp.316-323.
[11]M.M.Vaez and C.-T. Lea: Space-wavelength tradeoff in the design of nonblocking directional coupler based network under crosstalk constraint, J. Lightwave Technol.,vol.16, (Aug.1998), pp.1373-1379.
[12]G.Maier and A.Pattavina: Design of photonic rearrangeably networks with zero first-order switching-element-crosstalk, IEEE Trans. Commun., vol.49,no.7, (July.2001), pp.1268-1279.
[13]K.Padmanabhan and A.Netravali: Dilated networks for photonic switching, IEEE Trans. Commun., vol.COM-35, (Dec.1987), pp.1357-1365.
[14]D.Li: Elimination of crosstalk in directional coupler switches, Optical Quantum Electron., vol.25, no.4, (Apr.1993), pp.255-260.
[15] T.-S. Wong and C-T. Lea: Crosstalk reduction through wavelength assignment in WDM photonic switching networks, IEEE Trans. Commun., vol.49, no.7,(July.2001),pp.1280-1287.
[16]V.Benes: Mathematical Theory of Connecting Networks and Telephone Traffic, New York: Academic,1965.
[17] A.Pattavina: Switching Theory - Architecture and Performance in Broadband ATM Networks, 1st ed. New York: Wiley,1998.
[18] M.M.Vaez and C.-T. Lea: Strictly and wide-sense nonblocking photonic switching systems under crosstalk constraint, Proc. IEEE INFOCOM'98 (San Francisco, CA,USA, March/April 1998), vol.1, pp.118-125.
[19] M.M.Vaez and C.-T. Lea: Wide-sense nonblocking Banyan-type switching systems based on directional couplers, IEEE J. Select. Areas Commun., vol.16, (Sept.1998), pp.1327-1332.
[20] Xiaohong Jiang, Md. M. Khandker, Hong Shen and S.Horiguchi: Permutation in rearrangeably noblocking optical MINs with zero first-order switching-element-crosstalk, Proceedings of the 2002 IEEE Workshop on High Performance Switching and Routing (Kobe, Japan, May.2002), pp.19-23.