J Comb Optim (2007) 14: 131–142
DOI 10.1007/s10878-007-9065-5

Optimal all-to-all personalized exchange in d-nary banyan multistage interconnection networks

Victor W. Liu · Chiuyuan Chen · Richard B. Chen

Published online: 4 April 2007

Abstract All-to-all personalized exchange occurs in many important applications in parallel processing. In the past two decades, algorithms for all-to-all personalized exchange were mainly proposed for hypercubes, meshes, and tori. Recently, Yang and Wang (*IEEE Trans Parallel Distrib Syst* 11:261–274, 2000) proposed an optimal all-to-all personalized exchange algorithm for binary (each switch is of size 2×2) banyan multistage interconnection networks. It was pointed out in Massini (*Discret Appl Math* 128:435–446, 2003) that the algorithm in Yang, Wang (*IEEE Trans Parallel Distrib Syst* 11:261–274, 2000) depends on the network topologies and requires pre-computation and memory allocation for a Latin square. Thus in (*Discret Appl Math* 128:435–446, 2003), Massini proposed a new optimal algorithm, which is independent of the network topologies and does not require pre-computation or memory allocation for a Latin square. Unfortunately, Massini's algorithm has a flaw and does not realize all-to-all personalized exchange. In this paper, we will correct the flaw and generalize Massini's algorithm to be applicable to d-nary (each switch is of size $d \times d$) banyan multistage interconnection networks.

Keywords Multistage interconnection network · Banyan network · All-to-all communication · All-to-all personalized exchange · Latin square

1 Introduction

In a parallel (distributed) computing system, processors often need to communicate with each other. The communication among these processors can be *one-to-one* (uni-

Dedicated to Professor Frank K. Hwang on the occasion of his 65th birthday.

This research was partially supported by the National Science Council of the Republic of China under the grant NSC94-2115-M-009-006.

V.W. Liu · C. Chen (✉) · R.B. Chen
Department of Applied Mathematics, National Chiao Tung University, Hsinchu 300, Taiwan
e-mail: cychen@mail.nctu.edu.tw

cast), *one-to-many* (multicast), *one-to-all* (broadcast), and *all-to-all*. The all-to-all communication can be further classified as *all-to-all broadcast* and *all-to-all personalized exchange*. In the all-to-all broadcast, every processor sends the same message to every other processor; while in the all-to-all personalized exchange, every processor sends a distinct message to every other processor.

All-to-all personalized exchange occurs in many important applications (for example, matrix transposition and fast Fourier transform (FFT)) in parallel processing. Since a processor can send only one message in each time unit, the time needed to complete all-to-all personalized exchange is $\Omega(N)$, where N is the number of processors. The all-to-all personalized exchange problem has been extensively studied for hypercubes, meshes, and tori; see (Massini 2003; Yang and Wang 2000) for details. As was mentioned in (Yang and Wang 2000), although the algorithm for a hypercube achieves optimal time complexity, a hypercube suffers from unbounded node degrees and therefore has poor scalability. On the other hand, although a mesh or torus has a constant node degree and better scalability, its algorithm has a higher time complexity (Yang and Wang 2000).

Given N processors $P_0, P_1, \ldots, P_{N-1}$, a multistage interconnection network (MIN) can be used for communication among processors as shown in Fig. 1. An MIN is $N \times N$ if it has N inputs and N outputs. The nodes in an MIN are called *switches* (or *crossbars* or *switching elements*). Recently, switches of size other than 2×2 are used; see (Chang et al. 1999; Chen et al. 2003; Hwang 2004). An MIN is *d-nary* if each switch is of size $d \times d$; a 2-nary MIN is also called a *binary* MIN. A column in an MIN is called a *stage*. An MIN is *banyan* if there is a unique path between any input and any output. Let d be a positive integer and assume that N is a power of d. It is well-known that an $N \times N$ d-nary banyan MIN has exactly $\log_d N$ stages and an $N \times N$ d-nary $(\log_d N)$-stage MIN is banyan. For convenience, throughout this paper, an $N \times N$ d-MIN means an $N \times N$ d-nary $(\log_d N)$-stage MIN; see Fig. 2 for examples.

An $N \times N$ *Latin square* is an $N \times N$ matrix $\mathcal{A} = (a_{i,j})$, $i, j = 0, 1, \ldots, N-1$, such that entries $a_{i,j}$ are in the set $\{0, 1, \ldots, N-1\}$ and no two entries in a row or a column are identical. Yang and Wang (2000) found that: to realize all-to-all personalized exchange on an $N \times N$ 2-MIN, one only needs to arrange N network configurations so that their corresponding permutations (defined in Sect. 2)

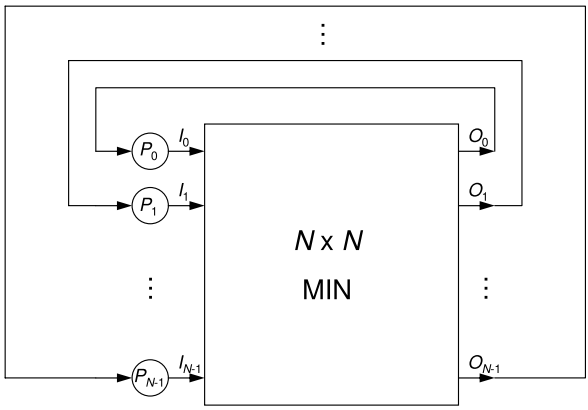

Fig. 1 Communications among processors using an MIN

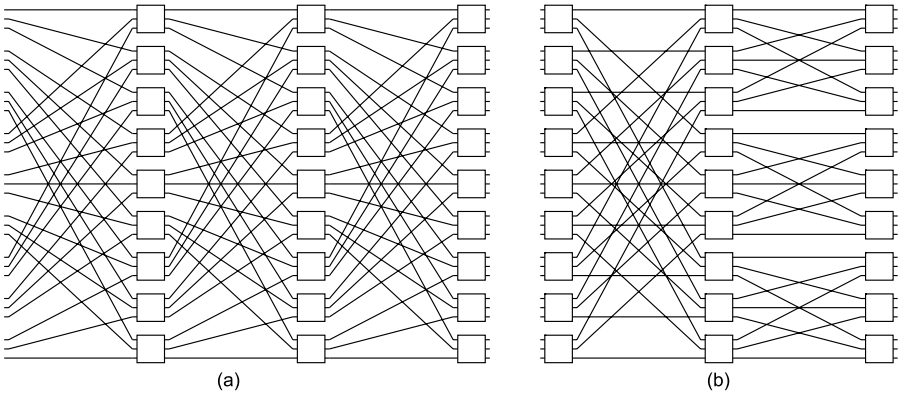

Fig. 2 **a** A 27×27 3-MIN which is also a shuffle-exchange network. **b** A 27×27 3-MIN which is also a baseline network

form an $N \times N$ Latin square. By using this Latin square method, Yang and Wang (2000) proposed an optimal all-to-all personalized exchange algorithm for an $N \times N$ 2-MIN; see also Yang and Wang (1999, 2000b, 2001). In Yang and Wang (2000), the time for constructing the Latin square is not included in the communication delay since the Latin square is constructed only once at the time the network is built. In (Massini 2003), Massini mentioned that Yang and Wang's algorithm depends on the network topologies and requires pre-computation and memory allocation for a Latin square. Massini (2003) then proposed a new algorithm and claimed that the proposed algorithm is optimal, is independent of the network topologies (hence is applicable to any $N \times N$ 2-MIN), and does not require pre-computation or memory allocation for a Latin square.

Unfortunately, Massini's algorithm (Massini 2003) has a flaw and does not realize all-to-all personalized exchange. The reason is: In all-to-all personalized exchange, a processor needs to send a personalized message to every other processor. Hence before a message is sent out from a processor, the processor needs to know which processor will be the destined processor in order to prepare a personalized message. Yang and Wang's algorithm (Yang and Wang 2000) uses a Latin square to store these information. Massini's algorithm (Yang and Wang 2000) only guarantees that every processor sends a message to every other processor, but when a message is sent out from a processor, Massini's algorithm does not know which processor will be the destined processor and therefore does not prepare a personalized message.

In this paper, we will correct the flaw of Massini's algorithm (Massini 2003) and propose an optimal all-to-all personalized exchange algorithm for an $N \times N$ d-MIN. Our algorithm is a generalization of Massini's algorithm (Massini 2003) and the algorithm in (Yang and Wang 2000) since these two algorithms work only for an $N \times N$ 2-MIN. We now summarize current results of all-to-all personalized exchange in Table 1; see also (Massini 2003; Yang and Wang 2000).

This paper is organized as follows: Sect. 2 gives some preliminary definitions. Section 3 is our optimal all-to-all personalized exchange algorithm. Section 4 is the concluding remarks.

Table 1 Comparisons of various networks used for all-to-all personalized exchange

Network model	Node degree	Diameter/No. of stages	Communication delay
hypercube one-port[a]	$\log_2 N$	$\log_2 N$	$O(N \log N)$
hypercube all-port[a]	$\log_2 N$	$\log_2 N$	$O(N)$
2D mesh/torus[b]	4	$O(N^{\frac{1}{2}})$	$O(N^{\frac{3}{2}})$
3D mesh/torus[c]	6	$O(N^{\frac{1}{3}})$	$O(N^{\frac{4}{3}})$
binary MIN[d]	1	$\log_2 N$	$O(N)$
d-MIN[e]	1	$\log_d N$	$O(N)$

[a]Johnsson and Ho (1989)

[b]Scott (1991); Suh and Shin (1998); Suh and Yalamanchili (1998); Thakur and Choudhary (1994); Tseng and Gupta (1996); Tseng et al. (1997)

[c]Suh and Shin (1998); Suh and Yalamanchili (1998); Tseng et al. (1997)

[d]Massini (2003); Yang and Wang (2000)

[e]This paper

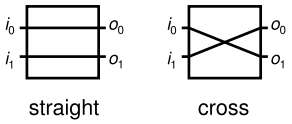

Fig. 3 The states of a 2×2 switch

2 Network configurations, permutations, and XOR

It is well known that a 2×2 switch has only two possible states: *straight* or *cross*, as shown in Fig. 3. Clearly, a $d \times d$ switch has $d!$ possible states. The *network configuration* of an MIN is defined by the states of its switches. Since an $N \times N$ d-MIN has $\frac{N}{d} \times \log_d N$ switches and each switch has $d!$ possible states, an $N \times N$ d-MIN has $(d!)^{\frac{N}{d} \cdot \log_d N}$ possible network configurations.

A *permutation* of an MIN is a one-to-one mapping between the inputs and outputs. For an $N \times N$ network, if there is a permutation that maps input $p(i)$ to output i, where $p(i) \in \{0, 1, \ldots, N-1\}$ for $i = 0, 1, \ldots, N-1$, then we will use

$$\begin{pmatrix} 0 & 1 & \ldots & N-1 \\ p(0) & p(1) & \ldots & p(N-1) \end{pmatrix}$$

or simply use

$$p(0) \quad p(1) \quad \ldots \quad p(N-1)$$

to denote the permutation. Given an MIN and a network configuration of it, a permutation (i.e., a communication between processors) can be obtained. For example, the network configuration shown in Fig. 5(b) maps input 3 to output 0, input 0 to output 1, input 1 to output 2, ..., and input 14 to output 15; this configuration obtains the permutation

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

i.e., the permutation

$$3 \quad 0 \quad 1 \quad 2 \quad 7 \quad 4 \quad 5 \quad 6 \quad 11 \quad 8 \quad 9 \quad 10 \quad 15 \quad 12 \quad 13 \quad 14.$$

Permutations realizable by an MIN are called *admissible permutations*. Note that not all of the $N!$ permutations are realizable by an $N \times N$ MIN. For example, the identity permutation is not realizable by the MIN in Fig. 2(b). Since an $N \times N$ d-MIN has $(d!)^{\frac{N}{d} \cdot \log_d N}$ possible network configurations, an $N \times N$ d-MIN can realize $(d!)^{\frac{N}{d} \cdot \log_d N}$ permutations, each corresponding to one of the $(d!)^{\frac{N}{d} \cdot \log_d N}$ network configurations.

Let $\oplus$ denote the XOR operation. It plays an important role in Massini's algorithm (Massini 2003). It is well known that

$$0 \oplus 0 = 0, \qquad 0 \oplus 1 = 1, \qquad 1 \oplus 0 = 1, \qquad 1 \oplus 1 = 0.$$

3 All-to-all personalized exchange in an $N \times N$ d-MIN

In this section, we will propose an optimal all-to-all personalized exchange algorithm for an $N \times N$ d-MIN. We first define the XOR operation for two d-nary digits. Let x, y be two d-nary digits, i.e., $x, y \in \{0, 1, \ldots, d-1\}$. The d-XOR operation (denoted as $\oplus_d$) is defined by

$$x \oplus_d y = (x + y) \bmod d.$$

The d-XOR operation is a generalization of the ordinary XOR operation since

$$0 \oplus_2 0 = 0, \qquad 0 \oplus_2 1 = 1, \qquad 1 \oplus_2 0 = 1, \qquad 1 \oplus_2 1 = 0.$$

Let $i_0, i_1, \ldots, i_{d-1}$ and $o_0, o_1, \ldots, o_{d-1}$ be the d input ports and d output ports of a $d \times d$ switch, respectively. A $d \times d$ switch is said to have the *k-shift state* if i_u is connected to $o_{(u+k) \bmod d}$ for $u = 0, 1, \ldots, d-1$; see Fig. 4 for an illustration. When $d = 2$, the 0-shift state is the straight state and the 1-shift state is the cross state. We will show that among the $d!$ possible states of a $d \times d$ switch, the d states in the set

$$\mathcal{S} = \{0\text{-shift}, 1\text{-shift}, \ldots, (d-1)\text{-shift}\}$$

are sufficient to realize all-to-all personalized exchange on an $N \times N$ d-MIN.

A network configuration of an $N \times N$ d-MIN can be represented by an $\frac{N}{d} \times \log_d N$ matrix $\mathcal{M} = (m_{i,j})$ such that entry $m_{i,j}$ represents the state of the ith switch of stage j (in this paper, we assume that stage 0 is the leftmost stage). Note that $m_{i,j} \in \{0, 1, \ldots, d-1\}$ since only the d states in $\mathcal{S}$ will be used by our algorithm; moreover, $m_{i,j} = k$ if the ith switch of stage j has the k-shift state.

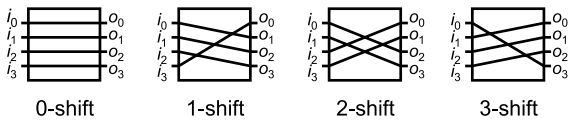

Fig. 4 The 0-shift, 1-shift, 2-shift, and 3-shift states of a 4×4 switch

Define N matrices $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{N-1}$ each of size $\frac{N}{d} \times \log_d N$ as follows (each $\mathcal{M}^i$ represents a network configuration of the given $N \times N$ d-MIN). Matrix $\mathcal{M}^0 = (m^0_{i,j})$ is defined by setting

$$m^0_{i,j} = \begin{cases} 0 & \text{if } d = 2 \text{ and the initial state of the } i\text{th switch of stage } j \text{ is straight,} \\ 1 & \text{if } d = 2 \text{ and the initial state of the } i\text{th switch of stage } j \text{ is cross,} \\ 0 & \text{if } d > 2, \end{cases}$$

for each i, j. When $d = 2$, the set $\mathcal{S}$ contains all the possible states of a switch; thus in this case, $m^0_{i,j}$ is set to 0 or 1 according to the initial state of the ith switch of stage j. When $d > 2$, the set $\mathcal{S}$ does not contain all the possible states of a switch; thus in this case, we will simply set $m^0_{i,j}$ to 0. Let x be an integer in $\{1, 2, \ldots, N-1\}$ and let

$$x_{(\log_d N)-1} x_{(\log_d N)-2} \cdots x_0$$

be the $(\log_d N)$-digit d-nary representation of x. For $x = 1, 2, \ldots, N-1$, matrix $\mathcal{M}^x = (m^x_{i,j})$ is obtained from matrix $\mathcal{M}^0$ by setting

$$m^x_{i,j} = m^0_{i,j} \bigoplus_d x_{(\log_d N)-1-j}$$

for each i, j. For example, if $N = 16$ and $d = 4$, then

$$\mathcal{M}^0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \mathcal{M}^1 = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \qquad \mathcal{M}^2 = \begin{bmatrix} 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 0 & 2 \end{bmatrix}, \qquad \ldots,$$

$$\mathcal{M}^{15} = \begin{bmatrix} 3 & 3 \\ 3 & 3 \\ 3 & 3 \\ 3 & 3 \end{bmatrix}.$$

The following lemma is obvious and its proof is omitted.

Lemma 1 *If $x \neq y$, $0 \leq x, y \leq N-1$, then corresponding rows of $\mathcal{M}^x$ and $\mathcal{M}^y$ are different, i.e., row r of $\mathcal{M}^x$ is different from row r of $\mathcal{M}^y$ for $r = 0, 1, \ldots, N-1$.*

Recall that given an MIN and a network configuration of it, a permutation can be obtained. The 16 permutations obtained from the network configurations represented by $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{15}$ and the $(\log_4 16)$-digit 4-nary representation for each x in $\{0, 1, 2, \ldots, 15\}$ are shown in Fig. 5. The following matrix $\mathcal{A}$ is formed by putting together (column by column) all the 16 permutations shown in Fig. 5. In matrix $\mathcal{A}$, row j represents the sequence of inputs arrived at output j. It is not difficult to see that $\mathcal{A}$ is a 16×16 Latin square and the Latin square property ensures that every processor receives a message from every other processor. Thus the 16 network configurations represented by $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{15}$ realize all-to-all-personalized exchange on the

given 16×16 4-MIN

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

We now prove the following theorem.

Theorem 2 *The matrix $\mathcal{A} = (a_{i,j})$ formed by putting together (column by column) all the permutations obtained by $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{N-1}$ is an $N \times N$ Latin square.*

Proof To prove that $\mathcal{A}$ is an $N \times N$ Latin square, it suffices to prove that every row and every column of $\mathcal{A}$ is a permutation of $\{0, 1, \ldots, N-1\}$. By the definition of $\mathcal{A}$, every column of $\mathcal{A}$ is a permutation of $\{0, 1, \ldots, N-1\}$. Thus it remains to prove that every row of $\mathcal{A}$ is also a permutation of $\{0, 1, \ldots, N-1\}$. Row j of $\mathcal{A}$ represents the sequence of inputs arrived at output j. An $N \times N$ d-MIN is banyan. Thus there is a unique path between any input and the output j. By Lemma 1, if $x \neq y$, $0 \leq x, y \leq N-1$, then row r of $\mathcal{M}^x$ is different from row r of $\mathcal{M}^y$ for $r = 0, 1, \ldots, N-1$. Thus $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{N-1}$ obtain N different paths arriving to the output j. By the banyan property of the MIN, N different inputs arrive to the output j. Thus every element in $\{0, 1, \ldots, N-1\}$ appears exactly once in row j of $\mathcal{A}$. Thus row j of $\mathcal{A}$ is a permutation of $\{0, 1, \ldots, N-1\}$. Hence every row of $\mathcal{A}$ is a permutation of $\{0, 1, \ldots, N-1\}$ and $\mathcal{A}$ is an $N \times N$ Latin square. □

We now have the following corollary.

Corollary 3 *Permutations obtained by the network configurations represented by $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{N-1}$ realize all-to-all personalized exchange on an $N \times N$ d-MIN.*

Recall that $\mathcal{M}^1, \mathcal{M}^2, \ldots, \mathcal{M}^{N-1}$ are obtained from $\mathcal{M}^0$ and when $d = 2$, $m^0_{i,j}$ is set to 0 or 1 according to the initial state of the ith switch of stage j. We therefore have the following corollary.

Corollary 4 *Given any initial network configuration of an $N \times N$ 2-MIN, permutations obtained by the network configurations represented by $\mathcal{M}^0, \mathcal{M}^1, \ldots, \mathcal{M}^{N-1}$ realize all-to-all personalized exchange.*

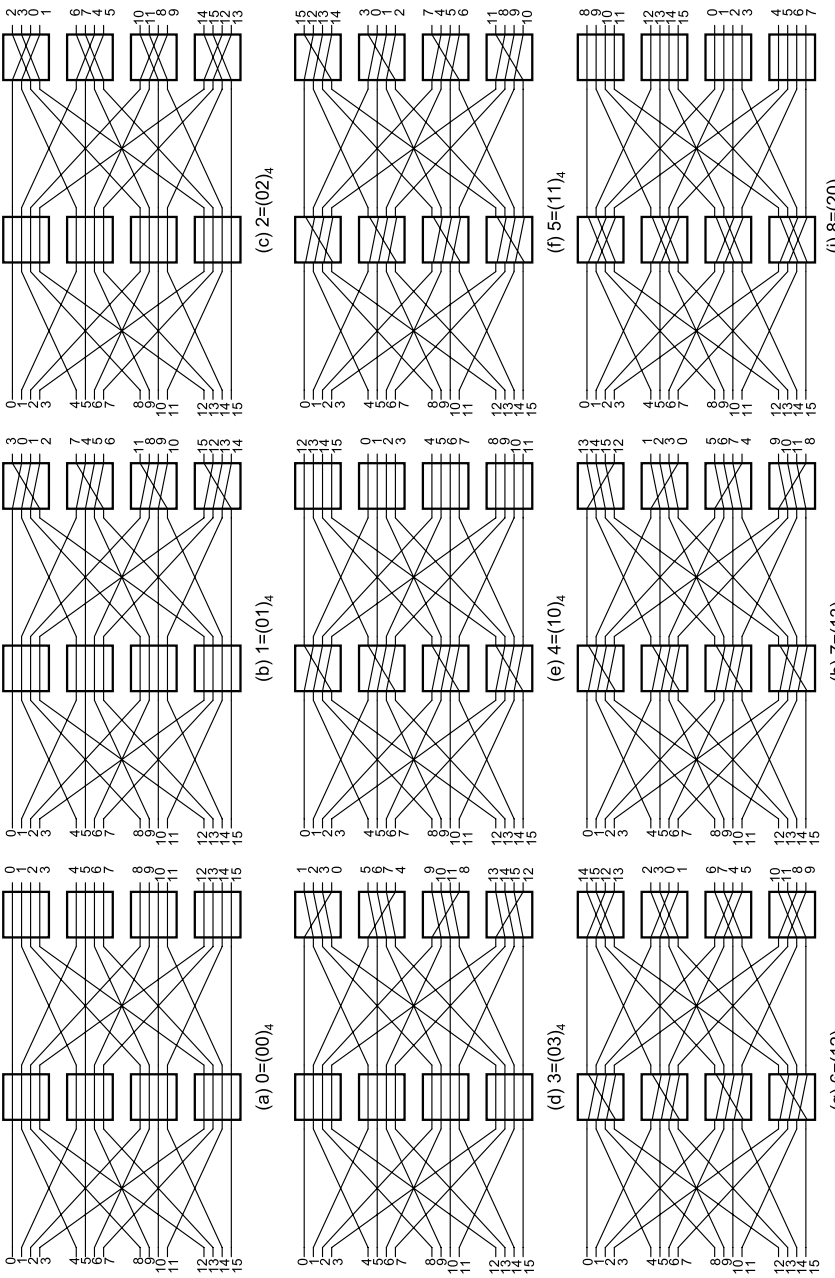

Fig. 5 A 16×16 4-MIN and the 16 network configurations and their corresponding permutations (listed at the right side of the MIN) that realize all-to-all personalized exchange

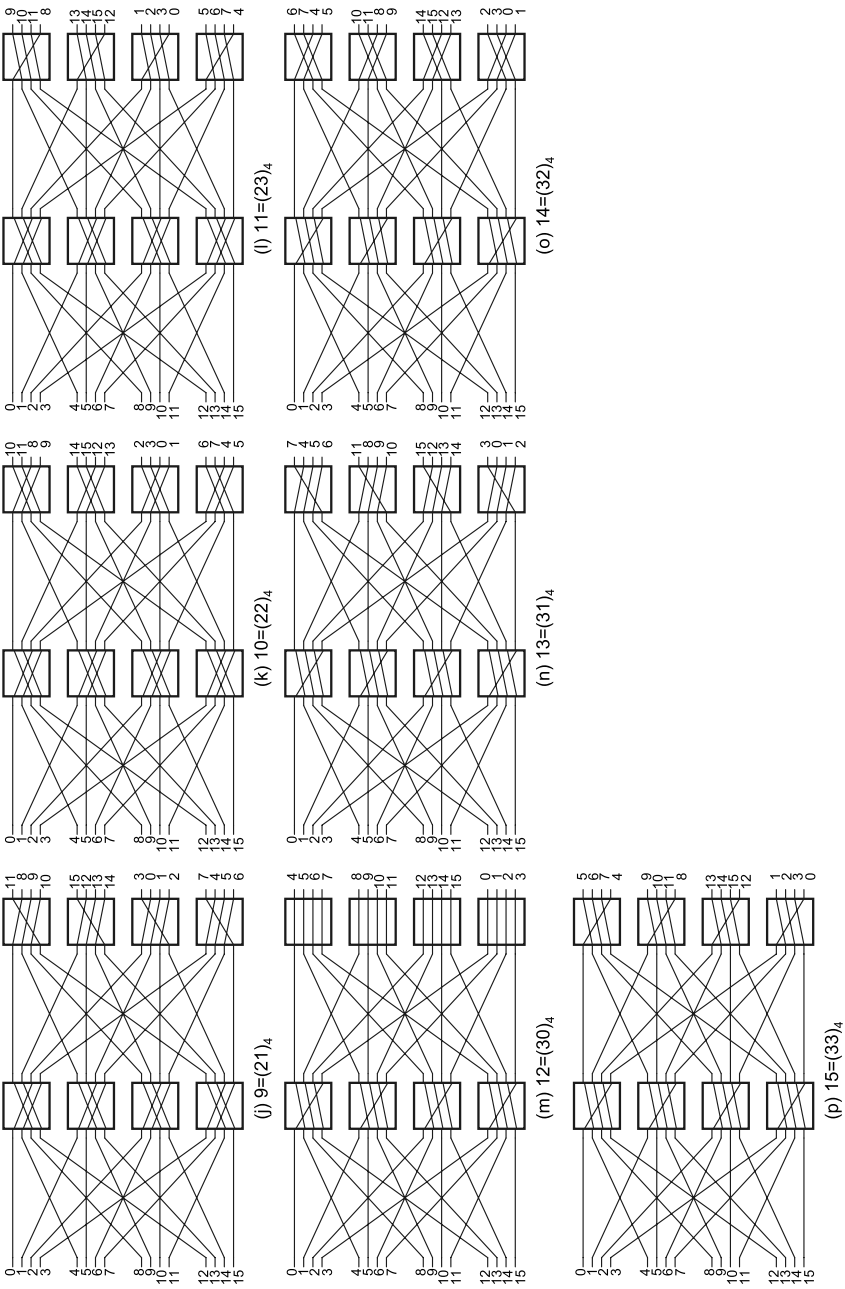

Fig. 5 Continued

The following is our all-to-all personalized exchange algorithm for an $N \times N$ d-MIN. Our algorithm has two phases. In the first phase, personalized messages that need to be sent out from each processor are prepared. In the second phase, personalized messages are sent out from the message queue of each processor. The first phase is used to correct the flaw of Massini's algorithm (Massini 2003); in this phase, personalized messages that need to be sent out from each processor are inserted into the message queue of that processor. The second phase is similar to Massini's algorithm except that the message sent out from each processor is a personalized message from the message queue of that processor.

All-to-all personalized exchange algorithm for an $N \times N$ d-MIN.
Phase 1: The message preparing phase.

- The $(\log_d N)$-digit d-nary representations $x_{(\log_d N)-1}x_{(\log_d N)-2}\ldots x_0$ of numbers $0, 1, \ldots, N-1$ are sequentially generated and the labels of every input of the MIN (the label of input 0 is 0, the label of input 1 is 1, etc.) are equipped with the current d-nary representation $x_{(\log_d N)-1}x_{(\log_d N)-2}\ldots x_0$.
- Before a label enters the ith switch of stage j, the switch is set to the k-shift state if $m^0_{i,j} \bigoplus_d x_{(\log_d N)-1-j} = k$.
- After a label leaves the ith switch of stage j, the switch is set to the $m^0_{i,j}$-shift state.
- When a label reaches an output, a personalized message is prepared; in particular, if label s reaches output t, then a personalized message that processor s wants to send to processor t is prepared and is inserted into the message queue of processor s.

Phase 2: The message sending phase.

- The $(\log_d N)$-digit d-nary representations $x_{(\log_d N)-1}x_{(\log_d N)-2}\ldots x_0$ of numbers $0, 1, \ldots, N-1$ are sequentially generated and the personalized messages in the message queue of every input of the MIN are equipped with the current d-nary representation $x_{(\log_d N)-1}x_{(\log_d N)-2}\ldots x_0$.
- Before a message enters the ith switch of stage j, the switch is set to the k-shift state if $m^0_{i,j} \bigoplus_d x_{(\log_d N)-1-j} = k$.
- After a message leaves the ith switch of stage j, the switch is set to the $m^0_{i,j}$-shift state.
- When a message reaches an output, that output receives a personalized message for it.

End of the algorithm.

Yang and Wang (2000) proved the following lemma (their proof for an $N \times N$ 2-MIN is easily extended to the general case).

Lemma 5 (Yang and Wang 2000) *The maximum communication delay of all-to-all personalized exchange on an $N \times N$ d-MIN is $\Omega(N + \log N)$, i.e., $\Omega(N)$.*

We now have the following theorem.

Theorem 6 *Our algorithm is an optimal all-to-all personalized exchange algorithm for an $N \times N$ d-MIN.*

Proof The correctness of our algorithm follows from Corollary 3. In the first phase of our algorithm, the labels pass through the stages of the MIN in a pipelined fashion; that is, when N labels leave a stage, other N new labels can enter the switches of the stage. In the second phase of our algorithm, the messages pass through the stages of the MIN also in a pipelined fashion; that is, when N messages leave a stage, other N new messages can enter the switches of the stage. Thus each phase of our algorithm takes $O(N + \log_d N)$ time, i.e. $O(N)$ time. By Lemma 5, our algorithm is optimal. □

4 Concluding remarks

In (Yang and Wang 2000), Yang and Wang proposed an optimal algorithm for realizing all-to-all personalized exchange on an $N \times N$ 2-MIN. It was pointed out in (Massini 2003) that Yang and Wang's algorithm depends on the network topologies and requires pre-computation and memory allocation for a Latin square. Thus in (Massini 2003), Massini proposed a new algorithm; this algorithm is applicable to any $N \times N$ 2-MIN (since it is independent of the network topologies) and it does not require pre-computation or memory allocation for a Latin square. Unfortunately, Massini's algorithm (Massini 2003) has a flaw and does not realize all-to-all personalized exchange.

In this paper, we have corrected the flaw of Massini's algorithm (Massini 2003) and have generalized it to be applicable to any $N \times N$ d-MIN. As was mentioned in (Yang and Wang 2000), the Latin square for realizing all-to-all personalized exchange on a network can be constructed only once at the time the network is built, and the Latin square associated with the network can be viewed as one of the system parameters. It is not difficult to see that the first phase of our algorithm can be easily modified to construct such a Latin square.

Acknowledgement The main results of this paper was done while the first author was a student in Professor Frank K. Hwang's Interconnection Networks class. Professor Hwang provided many helpful comments that greatly improves the presentation of this paper.

References

Chang GJ, Hwang FK, Tong LD (1999) Characterizing bit permutation networks. Networks 33:261–267

Chen Z, Liu Z, Qiu Z (2003) Bidirectional shuffle-exchange network and tag-based routing algorithm. IEEE Commun Lett 7:121–123

Hwang FK (2004) The mathematical theory of nonblocking switching networks. Series on applied mathematics, vol 15

Johnsson SL, Ho CT (1989) Optimum broadcasting and personalized communication in hypercubes. IEEE Trans Comput 38:1249–1268

Massini A (2003) All-to-all personalized communication on multistage interconnection networks. Discret Appl Math 128:435–446

Scott DS (1991) Efficient all-to-all communication patterns in hypercube and mesh topologies. In: Proceedings of the sixth distributed memory computing conference, pp 398–403

Suh YJ, Shin KG (1998) Efficient all-to-all personalized exchange in multidimensional torus networks. In: Proceedings of the international conference on parallel processing, pp 468–475

Suh YJ, Yalamanchili S (1998) All-to-all communication with minimum start-up costs in 2D/3D tori and meshes. IEEE Trans Parallel Distrib Syst 9:442–458

Thakur R, Choudhary A (1994) All-to-all communication on meshes with wormhole routing. In: Proceedings of the eighth IEEE international parallel processing symposium, pp 561–565

Tseng Y-C, Gupta S (1996) All-to-all personalized communication in a wormhole-routed torus. IEEE Trans Parallel Distrib Syst 7:498–505

Tseng Y-C, Lin T-H, Gupta S, Panda DK (1997) Bandwidth-optimal complete exchange on wormhole routed 2D/3D torus networks: a diagonal-propagation approach. IEEE Trans Parallel Distrib Syst 8:380–396

Yang Y, Wang J (1999) All-to-all personalized exchange in banyan networks. In: Proceedings of the parallel and distributed computing and systems (PDCS'99), Cambridge, MA, pp 78–86

Yang Y, Wang J (2000a) Optimal all-to-all personalized exchange in self-routable multistage networks. IEEE Trans Parallel Distrib Sys 11:261–274

Yang Y, Wang J (2000b) Optimal all-to-all personalized exchange in multistage networks. In: Proceedings of the seventh international conference on parallel and distributed systems (ICPADS'00), Iwale, Japan

Yang Y, Wang J (2001) Optimal all-to-all personalized exchange in a class of optical multistage networks. IEEE Trans Parallel Distrib Syst 12:567–582