國 立 交 通 大 學

應用數學系

碩 士 論 文

廣義的 Shuffle-Exchange 網路之

最佳化全體對全體個人訊息交換演算法

Optimal All-to-All Personalized Exchange Algorithms in Generalized Shuffle-Exchange Networks

研 究 生：邱鈺傑

指導教授：陳秋媛　教授

中 華 民 國 九 十 七 年 六 月

廣義的 Shuffle-Exchange 網路之
最佳化全體對全體個人訊息交換演算法

Optimal All-to-All Personalized Exchange Algorithms in Generalized Shuffle-Exchange Networks

研 究 生：邱鈺傑　　Student：Well Y. Chou

指導教授：陳秋媛　　Advisor：Chiuyuan Chen

國 立 交 通 大 學

應 用 數 學 系

碩 士 論 文

A Thesis
Submitted to Department of Applied Mathematics
College of Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master
in
Applied Mathematics
June 2008
Hsinchu, Taiwan, Republic of China

中 華 民 國 九 十 七 年 六 月

廣義的 Shuffle-Exchange 網路之
最佳化全體對全體個人訊息交換演算法

研究生：邱鈺傑　　指導老師：陳秋媛　教授

國 立 交 通 大 學

應 用 數 學 系

摘　要

以往文獻中全體對全體個人訊息交換演算法提出的對象主要是針對 hypercube、mesh 及 torus 網路。在文獻[17]中，Yang 以及 Wang 首先提出了針對多級式連接網路的全體對全體個人訊息交換演算法。他們的演算法是最佳的，但是只能在具有唯一路徑(unique-path)與自動找路(self-routable)性質的多級式連接網路中運作(例如：baseline、omega、banyan 網路)。必須注意到的是，文獻[17]中所有被考慮到的多級式連接網路都必須具有唯一路徑性質、而且滿足 N 是 2 的 n+1 次方，其中 N 表示多級式連接網路中輸入及輸出端的個數，2 表示所有的交換器為 2×2 大小，n+1 是多級式連接網路的層級數。就我們所知，目前尚未有人針對不具有唯一路徑性質、而且不滿足 N 為 2 的整數次方的多級式連接網路做過全體對全體個人訊息交換的研究。在文獻[12]中，Padmanabhan 提出了廣義的 shuffle-exchange 網路(GSEN)，允許 N 不是 2 的次方(在此 N 可以是任一偶數)。而當 N 為 2 的次方時，GSEN 即為 omega 網路(也就是原來的 shuffle-exchange 網路)。既然 GSEN 未必具有唯一路徑性質，Yang 和 Wang 的最佳演算法就不一定適用。本篇論文的目的即在於提出兩個 GSEN 之最佳化全體對全體個人訊息交換演算法。不同於 Yang 和 Wang 的演算法的是，我們捨棄對唯一路徑性質的要求。第一個演算法利用層級控制技術，且能應用在所有偶數 N；在使用層級控制技術之下，我們將證明它是最佳的。相反地，第二個演算法並不使用層級控制技術，它只能應用在 N 是偶數但不是 4 的倍數的 GSEN；我們也會證明它是最佳的。

關鍵詞：多級式連接網路，Shuffle-Exchange 網路，Omega 網路，平行與交換式計算，全體對全體溝通，全體對全體個人訊息交換。

中 華 民 國 九 十 七 年 六 月

Optimal All-to-All Personalized Exchange Algorithms in Generalized Shuffle-Exchange Networks

Student: Well Y. Chou Advisor: Chiuyuan Chen

Department of Applied Mathematics
National Chiao Tung University

Abstract

Previous all-to-all personalized exchange algorithms are mainly for hypercube, mesh, and torus. In [17], Yang and Wang first proposed an all-to-all personalized exchange algorithm for multistage interconnection networks (MINs). Their algorithm is optimal and works for a class of unique-path, self-routable MINs (for example, baseline, omega, banyan networks). Do notice that all the MINs considered in [17] must have the unique-path property and must satisfy $N = 2^{n+1}$, in which N is the number of inputs (outputs), 2 means all the switches are of size 2×2, and $n + 1$ is the number of stages in the MINs. To our knowledge, no one has studied all-to-all personalized exchange in MINs which do not have the unique-path property and do not satisfy $N = 2^{n+1}$. In [12], Padmanabhan proposed the generalized shuffle-exchange network (GSEN), which allows $N \neq 2^{n+1}$ (thus N can be any even number). A GSEN becomes an omega network (i.e., the shuffle-exchange network) when $N = 2^{n+1}$. Since a GSEN is not necessarily a unique-path MIN, Yang and Wang's optimal algorithm may not apply. The purpose of this thesis is to propose two optimal all-to-all personalized exchange algorithms for GSENs. Unlike Yang and Wang's algorithm, we abandon the the requirement on the unique-path. The first algorithm uses the stage control technique and works for all even N. We will prove it is optimal when the stage control technique is assumed. On the contrary, the second algorithm does not use the stage control technique and works for all N such that $N \equiv 2 \pmod 4$. We will prove that it is optimal.

Keywords: Multistage interconnection network; Shuffle-exchange network; Omega network; Parallel and distributed computing; All-to-all communication; All-to-all personalized exchange.

誌　　謝

光陰似箭，歲月如梭，轉眼間兩年半已經過去了。還記得當初考上國立交通大學應用數學所，在組合界是擁有最堅強師資陣容的系所。懷著期待的心情進入。

組合組擁有優秀的師資，以及團結和睦氣氛！在這種環境之下，同學間不僅在生活上互相幫助，課業上也互相砥礪。短短的時間內，在老師們的教導後讓我的視野更加開闊。感謝陳秋媛老師的演算法等課程、傅恆霖老師的圖論課程、翁志文老師的組合編碼等課程，以及黃大原老師的設計理論等課程。不只是理論的教導，更展延相關的應用。

其中最感謝的老師，就是我的指導老師：陳秋媛教授。她不只是一位好老師，更是像一位好姐姐，連結網路研究方向上給我啓蒙，生活上幫助我更多。我期待自己將來能成爲一位像陳教授一樣偉大的老師。在待人處事上，也開導我許多。

還要感謝傅恆霖教授，在我參加應數系男排的階段，他對系隊的支持讓每個人都深深感激，可惜的是在畢業的今天，我所參與的兩次大數盃比賽，皆未能幫助交大留下冠軍獎盃。隊友們之間留下了一句話：「沒冠軍不畢業。」

另外還要感謝我同屆的同學：威雄、敏筠、兆函、皜文，在奇怪的時間組成 **94.5g**。還有同研究室的國元學長、柏澍、志文、子鴻、信菖、松育、宜君、士慶、慧棻，博班的學長姐宏賓、棨丰、喻培、元勳、惠蘭，同組差半屆的偉慈、若宇、政緯、智懷、政軒、佩純、奇璁、偉帆、雅榕，同在系隊打拼的明淇、國安、小馬、大樹、昇哥、假死、阿翔、蛤仔、吉利、超人、小太、圈圈、新手、文慶、佑憲、企鵝、亥派、季子。有你們的參與下，讓我的研究所生活多彩多姿！

最後感謝我的家人，爸爸媽媽養育我，一直支持我唸書到今天，你們是我能成功的最重要支柱！姐姐和弟弟從小就和睦相處，也讓我確定了讀書很重要。最重要的朋友怡樺，謝謝妳兩年來的陪伴，我永遠不會忘記的。感恩的心，不止於此，僅以微薄紙筆，代表我心！

Contents

List of Figures

1 Introduction

Processors in a parallel and distributed processing system often need to communicate with other processors. The communication among these processors could be *one-to-one*, *one-to-many*, or *all-to-all*. All-to-all communication can be further classified into *all-to-all broadcast* and *all-to-all personalized exchange*. In all-to-all broadcast, each processor sends the same message to all other processors; while in all-to-all personalized exchange, each processor sends a specific message to every other processor. This thesis focuses on all-to-all personalized exchange.

All-to-all personalized exchange occurs in many important applications (for example, matrix transposition and fast Fourier transform (FFT)) in parallel and distributed computing. The all-to-all personalized exchange problem has been extensively studied for hypercubes, meshes, and tori; see [11, 17] for details. As was mentioned in [17], 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. In [17], Yang and Wang had proven that a MIN (defined later) is a better choice for implementing all-to-all personalized exchange due to its shorter communication delay and better scalability.

Given N processors $P_0, P_1, \ldots, P_{N-1}$ (i is the unique identifier (UID) of P_i), an $N \times N$ multistage interconnection network (MIN) can be used for communication among these processors as shown in Figure 1, where $N \times N$ means N inputs and N outputs. Figure 2 shows an example of a 10×10 MIN. A column in a MIN is called a *stage* and the nodes in a MIN are called *switches* (or *switching elements* or *crossbars*). Throughout this thesis, N denotes the number of processors and $n + 1$ denotes the number of stages of a MIN. Also, all the switches are assumed to be of size 2×2; see also [1, 2, 3, 5, 10] for switches of other sizes. It is well known that a 2×2 switch has only two possible states: *straight*

or *cross*, as shown in Figure 3.

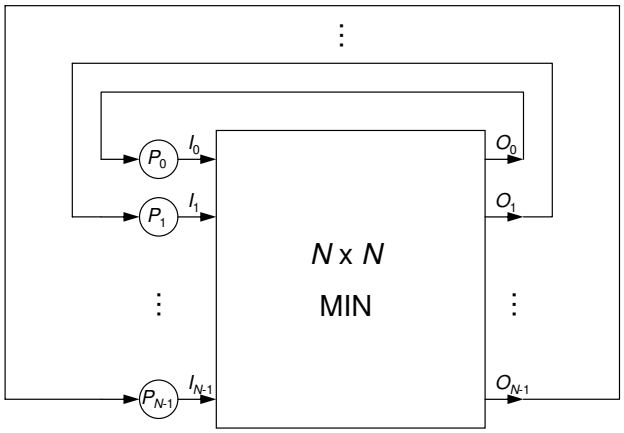

Figure 1: Communications among processors using a MIN.

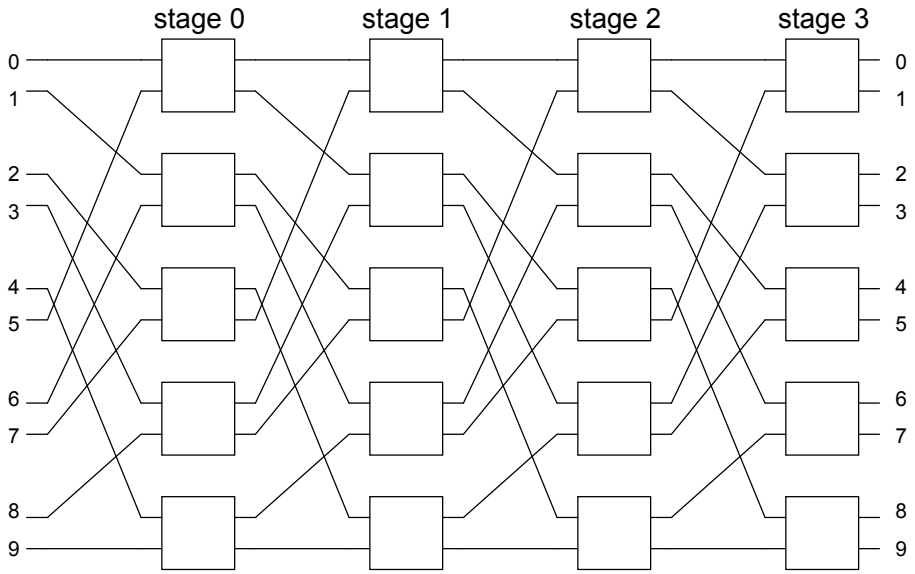

Figure 2: A 10×10 MIN which is also a 10×10 GSEN.

A MIN is *unique-path* if there is a unique path between each pair of input and output. A MIN is *self-routable* if the routing decision at a switch depends only on the addresses of the source processor and the destination processor. In [17], Yang and Wang first proposed an all-to-all personalized exchange algorithm for a class of unique-path, self-routable MINs; for example, baseline, omega, banyan networks, and the reverse networks of these networks. Yang and Wang's algorithm [17] uses stage control (see [13]), which is a commonly used technique to reduce the cost of the network setting for all-to-all

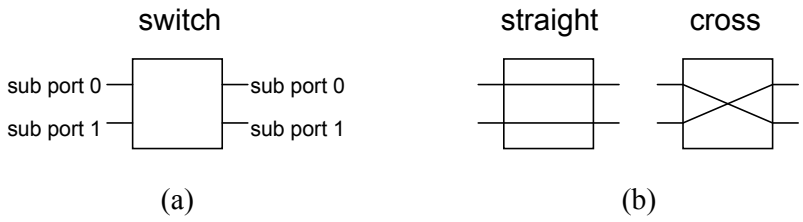

Figure 3: (a) A 2×2 switch and its sub ports. (b) The two possible states of a 2×2 switch.

personalized exchange communication. Stage control means that the states of all the switches of a stage have to be identical. With stage control, a single control bit (0 for straight and 1 for cross), or in other words, one electronic driver circuit, can be used to control all the switches of a stage. Thus the number of expensive electronic driver circuits needed is significantly lower than that of individual switch control.

Do notice that all the networks considered in [17], which include omega networks, must have the unique-path property and must satisfy $N = 2^{n+1}$. An omega network is also called a shuffle-exchange network (see [9]) and has been proposed as a popular architecture for MINs; see [4, 5, 8, 12, 14].

In [12], Padmanabhan proposed the generalized shuffle-exchange network (GSEN), which allows $N \neq 2^{n+1}$ (recall that $n+1$ is the number of stages). More precisely, assume that N is an even number and

$$2^n < N \leq 2^{n+1}.$$

Then an $N \times N$ *generalized shuffle-exchange network* is a MIN that has N inputs and N outputs and contains exactly $n+1$ stages such that each stage consists of the perfect shuffle on N terminals followed by $N/2$ switches. The N terminals in an $N \times N$ GSEN are numbered $0, 1, \ldots, N-1$ and the *perfect shuffle operation* on the N terminals is the permutation π defined by

$$\pi(i) = (2i + \left\lfloor \frac{2i}{N} \right\rfloor) \bmod N, \quad 0 \leq i \leq N-1.$$

See Figure 2 for an example.

Although Yang and Wang's algorithm in [17] is optimal, it works only for MINs that have the unique-path property and satisfy $N = 2^{n+1}$. Since a GSEN is not necessarily a unique-path MIN, Yang and Wang's optimal algorithm may not apply. To our knowledge, no one has studied all-to-all personalized exchange in MINs which do not have the unique-path property and do not satisfy $N = 2^{n+1}$. The purpose of this thesis is to propose all-to-all personalized exchange algorithms for GSENs. In particular, we propose two optimal all-to-all personalized exchange algorithms for GSENs. The first algorithm uses the stage control technique and works for all even N. We will prove it is optimal when the stage control technique is assumed. On the contrary, the second algorithm does not use the stage control technique and works for all N such that $N \equiv 2 \pmod 4$. We will prove that it is also optimal.

This thesis is organized as follows: Section 2 gives preliminaries. Section 3 is a lower bound on the maximum communication delay of all-to-all personalized exchange when the stage control technique is assumed. Section 4 is our first all-to-all personalized exchange algorithm for GSENs. Section 5 is our second all-to-all personalized exchange algorithm for GSENs. Concluding remarks are given in the final section.

2 Preliminaries

In the remaining part of this thesis, unless otherwise specified, a MIN means an $N \times N$ MIN and a GSEN means an $N \times N$ GSEN. In a GSEN, the switches are aligned in $n+1$ stages: stage 0, stage 1, ..., stage n, with each stage consists of $N/2$ switches. The *network configuration* of a GSEN is defined by the states of its switches. Since a GSEN has $(N/2) \times (n+1)$ switches, its network configuration can be represented by an $(N/2) \times (n+1)$ matrix in which each entry is defined by the state of its corresponding switch. For example, the network configuration of the GSEN in Figure 4 is shown in Figure 5.

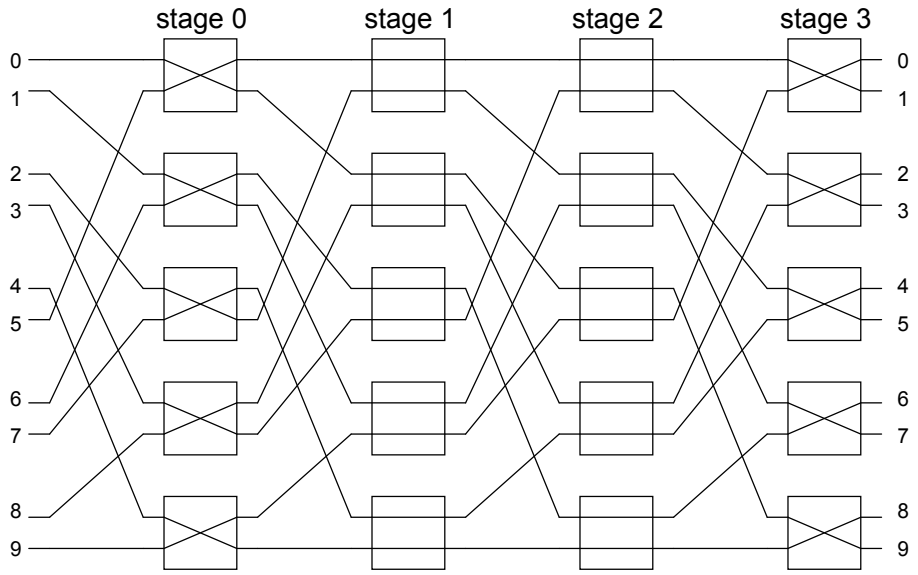

Figure 4: A 10×10 GSEN.

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

Figure 5: The network configuration of the GSEN in Figure 4.

When the stage control technique is assumed, the network configuration of a GSEN can be represented by a number as follows. Let c_ℓ denotes the state, 0 for straight and 1 for cross, of all the switches at stage $n - \ell$. Then the network configuration of the GSEN can be represented by the number

$$C = c_n 2^n + c_{n-1} 2^{n-1} + \cdots + c_1 2^1 + c_0 2^0$$

or by the binary number

$$(c_n \ c_{n-1} \ \cdots \ c_1 \ c_0)_2.$$

For example, the network configuration of the GSEN in Figure 4 can be represented by 9 or by $(1001)_2$. Clearly,

$$0 \leq C < 2^{n+1}.$$

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

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

or simply

$$p(0)\ p(1)\ \cdots\ p(N-1)$$

to denote the permutation.

Given the network configuration of a MIN, a permutation can be obtained. For example, the network configuration shown in Figure 4 maps input 0 to output 9, input 1 to output 7, input 2 to output 5, ..., and input 9 to output 0; thus this network configuration obtains the permutation

9 7 5 3 8 1 6 4 2 0.

It is obvious that a MIN has $N!$ possible permutations. However, not all of the $N!$ permutations are realizable. For example, permutation 7 3 9 5 1 6 2 8 4 0 is not realizable by the MIN shown in Figure 2. Permutations realizable by a MIN are called *admissible permutations* of that MIN.

An $N \times N$ *Latin square* is an $N \times N$ matrix such that each entry is in the set $\{0, 1, \ldots, N-1\}$ and no two entries in a row or a column are identical. In [17], Yang and Wang found that: to realize all-to-all personalized exchange for a unique-path, self-routable MIN, one only needs to arrange N network configurations so that their corresponding admissible permutations form an $N \times N$ Latin square. By using this Latin square method, Yang and Wang [17] proposed an optimal all-to-all personalized exchange algorithm for a class of unique-path, self-routable MINs; see also [10, 11, 15, 16, 18].

The following conventions are used in the remaining part of this thesis. Terminal i (j) is assumed on the left-hand (right-hand) side of the network and therefore is an input

(output) processor. An (i,j)-*request* denotes a request for sending a message from i to j. An (i,j)-*path* denotes a path between i and j. Obviously, an (i,j)-request can be fulfilled by an (i,j)-path.

In a MIN, a path from a source processor to a destination processor can be described by a sequence of labels that label the successive links on this path. Such a sequence is called a *control tag* [12] or *tag* [2] or *path descriptor* [6]. The control tag may be used as a header for routing a message: each successive switch uses the first element of the sequence to route the message, and then discards it. More precisely, suppose the control tag is

$$F = f_n 2^n + f_{n-1} 2^{n-1} + \cdots + f_1 2^1 + f_0 2^0.$$

Then bit f_ℓ controls the switch at stage $n-\ell$ in the path and if $f_\ell = 0$, then a connection is made to sub port 0; if $f_\ell = 1$, then a connection is made to sub port 1. For example, in Figure 4, $i = 2$ can get to $j = 5$ by using the control tag $F = 9 = (1001)_2$, which means that the $(2,5)$-request can be fulfilled by the path via sub port 1 at stage 0, sub port 0 at stage 1, sub port 0 at stage 2, and sub port 1 at stage 3. Note that

$$0 \le F < 2^{n+1}.$$

In this thesis, $\oplus$ denotes the XOR operation. As a reference,

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

If $U = (u_n\ u_{n-1}\ \cdots\ u_0)_2$ and $V = (v_n\ v_{n-1}\ \cdots\ v_0)_2$, then we define

$$U \oplus V = (u_n \oplus v_n\ \ u_{n-1} \oplus v_{n-1}\ \ \cdots\ \ u_0 \oplus v_0)_2.$$

Let $\mathcal{R}(N)$ denote the minimum number of network configurations required to realize all-to-all personalized exchange in an $N \times N$ GSEN. Also, let $\mathcal{R}_{sc}(N)$ denote the minimum number of network configurations required to realize all-to-all personalized exchange in an $N \times N$ GSEN when the stage control technique is assumed. We now prove a lemma.

Lemma 1.

$$N \leq \mathcal{R}(N) \leq \mathcal{R}_{sc}(N) \leq 2^{n+1}.$$

Proof. Given a network configuration, a permutation can be obtained, which means N (personalized) messages can be sent simultaneously. The inequality $N \leq \mathcal{R}(N)$ thus follows from that fact that N^2 messages have to be sent to fulfill all-to-all personalized exchange and each network configuration can send only N of them. The inequality $\mathcal{R}(N) \leq \mathcal{R}_{sc}(N)$ is obvious. The inequality $\mathcal{R}_{sc}(N) \leq 2^{n+1}$ follows from the fact that a GSEN has at most 2^{n+1} network configurations when the stage control technique is assumed. ∎

3 A lower bound when the stage control technique is assumed

The purpose of this section is to prove the following lower bound.

Theorem 2. *When the stage control technique is assumed, the maximum communication delay of all-to-all personalized exchange in an $N \times N$ GSEN of $n+1$ stages, where $2^n < N \leq 2^{n+1}$, is at least $\Omega(2^{n+1} + n)$.*

Before we prove this lower bound, we mention a lower bound obtained by Yang and Wang in [17]. Recall that the algorithm in [17] also uses the stage control technique.

Lemma 3. *[17] The maximum communication delay of all-to-all personalized exchange in an $N \times N$ MIN of $n+1$ stages, where $N = 2^{n+1}$, is at least $\Omega(N + n)$.*

This lemma holds since each of the N processors (say, processor j) has to receive N messages and it takes $n+1$ rounds (a *round* is the process of transmitting all the messages from the current stage to the next stage) for the first message to arrive j and after that, it takes $N-1$ rounds for the remaining $N-1$ messages to arrive j.

By similar arguments, we have the following lemma and its proof is omitted.

Lemma 4. *The maximum communication delay of all-to-all personalized exchange in an $N \times N$ GSEN of $n+1$ stages, where $2^n < N \leq 2^{n+1}$, is at least $\Omega(N+n)$.*

Let

$$N = 2^n + M, \text{ with } 0 < M \leq 2^{n+1} - 2^n.$$

In [12], Padmanabhan had proven the following theorem.

Theorem 5. *[12] Any i, $0 \leq i < N$, can set up a path to a j, $0 \leq j < N$, by using the control tag*

$$F_1 = (j + 2Mi) \mod N.$$

In addition, if $F_1 + N < 2N$, then a second control tag exists and is given by

$$F_2 = F_1 + N.$$

Consider an (i,j)-request and an (i,j)-path P. (See Figure 6.) When a message is sent from i to j along P, the message enters a switch at stage $n-\ell$ via sub port b_ℓ and leaves the switch via sub port f_ℓ. Recall that $F = f_n 2^n + f_{n-1}2^{n-1} + \cdots + f_1 2^1 + f_0 2^0$ is called a control tag for i to get to j. Now let

$$B = b_n 2^n + b_{n-1}2^{n-1} + \cdots + b_1 2^1 + b_0 2^0.$$

Clearly,

$$0 \leq B < 2^{n+1}.$$

In [7], Lan et al. called F the *forward control tag*. They also called B the *backward control tag* since if a message is sent from j to i along P, then the message enters a switch at stage $n-\ell$ via sub port f_ℓ and leaves the switch via sub port b_ℓ.

Note that in [7], Lan et al. considered switches of size $k \times k$. By setting $k = 2$, we have the following two lemmas.

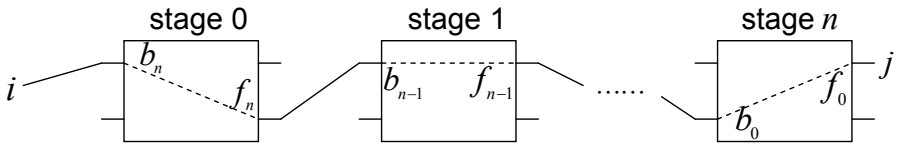

Figure 6: An (i, j)-path P and the sub ports on P.

Lemma 6. *[7] Given i and F, the destination processor j is given by*

$$j = (i \cdot 2^{n+1} + F) \mod N.$$

Lemma 7. *[7]*

$$B = \left\lfloor \frac{i \cdot 2^{n+1} + F}{N} \right\rfloor.$$

In this thesis, the purpose of introducing B is to prove the following result.

Lemma 8. *When the stage control technique is assumed, F and B together uniquely determine the network configuration C and*

$$C = B \oplus F.$$

Proof. Consider stage n–ℓ. Since the stage control technique is assumed, all switches in stage n–ℓ are of the same state. Let $C = c_n 2^n + c_{n-1} 2^{n-1} + \cdots + c_1 2^1 + c_0 2^0$ be the network configuration and see Figure 6. At stage $n-\ell$, a message enters sub port b_ℓ and leaves sub port f_ℓ. If $b_\ell = f_\ell$, then the state of the switch is straight; hence $c_\ell = 0 = b_\ell \oplus f_\ell$. If b_ℓ differs from f_ℓ (in this case, (b_ℓ, f_ℓ) is $(0, 1)$ or $(1, 0)$), then the state of the switch is cross; hence $c_\ell = 1 = b_\ell \oplus f_\ell$. From the above, $C = B \oplus F$. ∎

To prove Theorem 2, the following terminologies are introduced. Suppose F is given. Let $P_F(i)$ denote the path started from i by using the control tag F; note that the destination processor j of $P_F(i)$ can be determined by Lemma 6. Let

$$\mathcal{P}_F = \{P_F(i) \mid i = 0, 1, \ldots, N-1\}.$$

Let $B_F(i)$ denote the backward control tag of $P_F(i)$ and let

$$\mathcal{B}_F = \{B_F(i) \mid i = 0, 1, \ldots, N-1\}.$$

We now prove:

Lemma 9. *Each path in $\mathcal{P}_{2^n} \bigcup \mathcal{P}_{2^n+1}$ is a unique path between its source processor and destination processor.*

Proof. Recall that N is even and $2^n < N \leq 2^{n+1}$. So $2^n + 2 \leq N \leq 2^{n+1}$. Consider an arbitrary path $P_{2^n}(i)$ in $\mathcal{P}_{2^n}$ first. Suppose $P_{2^n}(i)$ joins i to j. If $P_{2^n}(i)$ is not a unique path, then there exists another control tag F such that i can also get to j by using F. By Theorem 5, the difference between control tag F and control tag 2^n is N; thus either $F - 2^n = N$ or $2^n - F = N$. In the former case, $F = 2^n + N > 2^{n+1}$; this is impossible since $0 \leq F < 2^{n+1}$. In the latter case, $F = 2^n - N < 0$; this is also impossible.

Next consider an arbitrary path $P_{2^n+1}(i)$ in $\mathcal{P}_{2^n+1}$. Suppose $P_{2^n+1}(i)$ joins i to j. If $P_{2^n+1}(i)$ is not a unique path, then there exists another control tag F' such that i can also get to j by using F'. By Theorem 5, either $F' - (2^n + 1) = N$ or $(2^n + 1) - F' = N$. In the former case, $F' = 2^n + 1 + N > 2^{n+1}$; this is impossible. In the latter case, $F' = 2^n + 1 - N < 0$; this is also impossible. ∎

We have proven that each path in $\mathcal{P}_{2^n} \bigcup \mathcal{P}_{2^n+1}$ is a unique path. We now prove that the sets $\mathcal{B}_{2^n}$ and $\mathcal{B}_{2^n+1}$ are equal.

Lemma 10.

$$\mathcal{B}_{2^n} = \mathcal{B}_{2^n+1}.$$

Proof. The binary representations of 2^n and $2^n + 1$ differ only at their rightmost bits. Thus for $i = 0, 1, \ldots, N-1$, paths $P_{2^n}(i)$ and $P_{2^n+1}(i)$ differ only at their destination processors; so $B_{2^n}(i) = B_{2^n+1}(i)$. Consequently, $\mathcal{B}_{2^n} = \mathcal{B}_{2^n+1}$. ∎

For convenience, if a number is in $\{0, 1, 2, \ldots, 2^{n+1} - 1\}$ but is not in $\mathcal{B}_F$, then we call it a *hole* of $\mathcal{B}_F$. The following lemma shows that the elements of $\mathcal{B}_F$ are distributed very *uniformly* on the set $\{0, 1, 2, \ldots, 2^{n+1}-1\}$.

Lemma 11. *For any $F \in \{0, 1, 2, \ldots, 2^{n+1} - 1\}$, $\mathcal{B}_F$ has no two consecutive holes.*

Proof. We will prove this lemma by showing $B_F(0) \leq 1$, $B_F(i-1) + 1 \leq B_F(i) \leq B_F(i-1) + 2$ for $i = 1, 2, \ldots, N-1$, and $B_F(N-1) \geq 2^{n+1} - 2$. Recall that $2^n < N \leq 2^{n+1}$ and $0 \leq F < 2^{n+1}$. By Lemma 7, $B_F(0) = \left\lfloor \frac{F}{N} \right\rfloor \leq 1$. Also, $B_F(N-1) = \left\lfloor \frac{(N-1)\cdot 2^{n+1}+F}{N} \right\rfloor \geq \left\lfloor \frac{(N-1)\cdot 2^{n+1}}{N} \right\rfloor \geq 2^{n+1} - 2$. Finally, consider $i = 1, 2, \ldots, N-1$. By Lemma 7, $B_F(i-1) + 1 = \left\lfloor \frac{(i-1)\cdot 2^{n+1}+F}{N} \right\rfloor + 1 = \left\lfloor \frac{i\cdot 2^{n+1}+F}{N} - \frac{2^{n+1}}{N} \right\rfloor + 1 \leq \left\lfloor \frac{i\cdot 2^{n+1}+F}{N} \right\rfloor = B_F(i) = \left\lfloor \frac{(i-1)\cdot 2^{n+1}+F}{N} + \frac{2^{n+1}}{N} \right\rfloor \leq \left\lfloor \frac{(i-1)\cdot 2^{n+1}+F}{N} \right\rfloor + 2 = B_F(i-1) + 2$. ∎

Now we are ready to prove Theorem 2.

Proof. Recall that a round is the process of transmitting all the messages from the current stage to the next stage. In an all-to-all personalized exchange, each processor has to receive N messages. It takes at least n rounds before a message can get to its destination processor. Thus to prove this theorem, it suffices to prove that when the stage control technique is assumed, 2^{n+1} network configurations are required for each processor to receive N messages; in other words, it suffices to prove that $\mathcal{R}_{sc}(N) = 2^{n+1}$. By Lemma 1, $\mathcal{R}_{sc}(N) \leq 2^{n+1}$. Thus it remains to prove that $\mathcal{R}_{sc}(N) \geq 2^{n+1}$.

When the stage control technique is assumed, the network configuration C can be determined by an arbitrary path P set up by C. In particular, if F is the control tag used by P and B is the the backward control tag of P (see Figure 6), then by Lemma 8, $C = B \oplus F$. If P is a unique path, then C must be used in all-to-all personalized exchange.

Recall that $0 \leq C < 2^{n+1}$. Our idea used in proving $\mathcal{R}_{sc}(N) \geq 2^{n+1}$ is to prove that for each C in $\{0, 1, \ldots, 2^{n+1} - 1\}$, at least one of the paths set up by C is a unique path and hence C must be used in all-to-all personalized exchange.

Suppose to the contrary there is a $\hat{C}$ in $\{0, 1, \ldots, 2^{n+1}-1\}$ such that none of the paths set up by $\hat{C}$ is a unique path. Then consider $2^n \oplus \hat{C}$ and let $\hat{B} = 2^n \oplus \hat{C}$; consider $(2^n + 1) \oplus \hat{C}$ and let $\hat{B}' = (2^n + 1) \oplus \hat{C}$. Since none of the paths set up by $\hat{C}$ is a unique path, we have

$$\hat{B} \notin \mathcal{B}_{2^n} \quad \text{and} \quad \hat{B}' \notin \mathcal{B}_{2^n+1}.$$

By Lemma 10, $\mathcal{B}_{2^n} = \mathcal{B}_{2^n+1}$. Thus

$$\hat{B} \notin \mathcal{B}_{2^n} \quad \text{and} \quad \hat{B}' \notin \mathcal{B}_{2^n}.$$

Since $\hat{B}$ and $\hat{B}'$ differ by 1, they are two consecutive holes in $\mathcal{B}_{2^n}$; this contradicts with Lemma 11. Thus for each network configuration C in $\{0, 1, \ldots, 2^{n+1}-1\}$, at least one of the paths set up by C is a unique path; hence C must be used in all-to-all personalized exchange. So $\mathcal{R}_{sc}(N) \geq 2^{n+1}$ and we have Theorem 2. ∎

4 All-to-all personalized exchange that uses stage control

In this section, we will propose our first all-to-all personalized exchange algorithm for GSENs. This algorithm assumes the stage control technique. For convenience, the row index and the column index of a matrix start from 0. Again, a round is the process of transmitting all the messages from the current stage to the next stage.

To ensure the stage control technique, the switches of a given GSEN are set according to the following rule.

Rule-SC: All the messages sent out at round $k + 1$ are equipped with the network configuration k. Suppose $k = (c_n\ c_{n-1}\ \cdots\ c_1\ c_0)_2$. Then, before all the messages sent out at round $k+1$ enter the switches at stage ℓ, all the switches at stage ℓ are set to straight if $c_{n-\ell} = 0$ and set to cross if $c_{n-\ell} = 1$.

Our algorithm has a preprocessing phase, which is used to construct a matrix $D = (d_{i,k})$ (here D denotes "destination") so that $d_{i,k} = j$ means processor i will send a personalized message to processor j (the *destination*) at round $k+1$. After D is constructed, our algorithm uses it to fulfill all-to-all personalized exchange.

Recall that the UID of input i is i. The following is the preprocessing phase of our algorithm; it constructs D. To construct D, a matrix $S = (s_{j,k})$ is constructed first (here S denotes "source") so that $s_{j,k} = i$ means processor i (the *source*) will send a personalized message to processor j at round $k + 1$. To construct S, the UID of every processor is equipped with a network configuration before it is sent; at round k, the equipped network configuration is k.

Note that an array (called mark) is used to ensure that each processor j receives only one message from each processor i. More precisely, if mark$[i] \neq 0$, then there exist k and k' such that $d_{i,k} = d_{i,k'} = j$ and $k < k'$. Then $d_{i,k'}$ will be set to -1, which means at round $k' + 1$, a null message instead of a personalized message from i will be sent to j.

Algorithm CONSTRUCT-MATRIX-D
(preprocessing phase of Algorithm GSEN-ATAPE-SC)
for each processor i $(0 \leq i < N)$ **do**
  for each k $(0 \leq k < 2^{n+1})$ **do**

- equip the UID of i (which is the message) with k (which is the network configuration for **Rule-SC**) and send the UID out;
- when an output (say, j) receives the UID, set $s_{j,k} = i$ if the network configuration equipped with the UID is k;

for each j $(0 \leq j < N)$ **do**
  for each k $(0 \leq k < 2^{n+1})$ **do**
    if $s_{j,k} = i$ **then** set $d_{i,k} = j$;
for each i $(0 \leq i < N)$ **do**
  for each j $(0 \leq j < N)$ **do**
    set mark$[j] = 0$;
  for each k $(0 \leq k < 2^{n+1})$ **do**
    if mark$[d_{i,k}] = 0$ **then** set mark$[d_{i,k}] = 1$;
    else set $d_{i,k} = -1$;

For example (in this example, -1 is represented by $-$), the matrices S and D for the

GSEN shown in Figure 2 are

$$S = \begin{bmatrix} 0 & 4 & 8 & 5 & 7 & 6 & 1 & 3 & 5 & 9 & 3 & 0 & 2 & 1 & 6 & 8 \\ 4 & 0 & 5 & 8 & 6 & 7 & 3 & 1 & 9 & 5 & 0 & 3 & 1 & 2 & 8 & 6 \\ 8 & 3 & 0 & 2 & 1 & 5 & 7 & 9 & 3 & 8 & 5 & 7 & 6 & 0 & 2 & 4 \\ 3 & 8 & 2 & 0 & 5 & 1 & 9 & 7 & 8 & 3 & 7 & 5 & 0 & 6 & 4 & 2 \\ 7 & 2 & 6 & 3 & 0 & 9 & 4 & 5 & 2 & 7 & 1 & 8 & 5 & 4 & 9 & 0 \\ 2 & 7 & 3 & 6 & 9 & 0 & 5 & 4 & 7 & 2 & 8 & 1 & 4 & 5 & 0 & 9 \\ 6 & 1 & 7 & 9 & 4 & 8 & 0 & 2 & 1 & 6 & 2 & 4 & 9 & 3 & 5 & 7 \\ 1 & 6 & 9 & 7 & 8 & 4 & 2 & 0 & 6 & 1 & 4 & 2 & 3 & 9 & 7 & 5 \\ 5 & 9 & 4 & 1 & 3 & 2 & 6 & 8 & 0 & 4 & 9 & 6 & 8 & 7 & 1 & 3 \\ 9 & 5 & 1 & 4 & 2 & 3 & 8 & 6 & 4 & 0 & 6 & 9 & 7 & 8 & 3 & 1 \end{bmatrix}$$

and

$$D = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & - & - & - & - & - & - \\ 7 & 6 & 9 & 8 & 2 & 3 & 0 & 1 & - & - & 4 & 5 & - & - & - & - \\ 5 & 4 & 3 & 2 & 9 & 8 & 7 & 6 & - & - & - & - & 0 & 1 & - & - \\ 3 & 2 & 5 & 4 & 8 & 9 & 1 & 0 & - & - & - & - & 7 & 6 & - & - \\ 1 & 0 & 8 & 9 & 6 & 7 & 4 & 5 & - & - & - & - & - & - & 3 & 2 \\ 8 & 9 & 1 & 0 & 3 & 2 & 5 & 4 & - & - & - & - & - & - & 6 & 7 \\ 6 & 7 & 4 & 5 & 1 & 0 & 8 & 9 & - & - & - & - & 2 & 3 & - & - \\ 4 & 5 & 6 & 7 & 0 & 1 & 2 & 3 & - & - & - & - & 9 & 8 & - & - \\ 2 & 3 & 0 & 1 & 7 & 6 & 9 & 8 & - & - & 5 & 4 & - & - & - & - \\ 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & - & - & - & - & - & - \end{bmatrix}.$$

The following is our first all-to-all personalized exchange algorithm for GSENs. It consists of two phases: the message preparing phase and the message sending phase. Note that the switches of the given GSEN are set according to Rule-SC.

Algorithm GSEN-ATAPE-SC
Phase 1: The message preparing phase
for each processor i $(0 \leq i < N)$ **do in parallel**
  for each k $(0 \leq k < 2^{n+1})$ **do in sequential**

- prepare a personalized message for i to sent to $d_{i,k}$ if $d_{i,k} \neq -1$ or prepare a null message if $d_{i,k} = -1$;
- equip the message with k (which is the network configuration for **Rule-SC**) and insert the message into the message queue of i;

Phase 2: The message sending phase
for each processor i $(0 \leq i < N)$ **do in parallel**
  for each k $(0 \leq k < 2^{n+1})$ **do in sequential**
    send a message in the message queue of i;

An example of Phase 2 of Algorithm GSEN-ATAPE-SC is shown in Figures 9 and 10. In these two figures, each 0-1 string is the binary representation of the number k

with which a message is equipped. We now prove the correctness and analyze the time complexity of the above two algorithms.

Theorem 12. *Algorithm CONSTRUCT-MATRIX-D is correct and it takes $O(N^2)$ time.*

Proof. During the execution of this algorithm, $s_{j,k}$ is set to i only when i is the $(k+1)$-th UID that arrives j. The correctness of this algorithm then follows from the fact that $d_{i,k}$ is set to j only when $s_{j,k} = i$. It is obvious that the algorithm takes $O(N2^{n+1})$ time. Since $2^{n+1} < 2N$, the algorithm takes $O(N^2)$ time. ∎

Theorem 13. *Algorithm GSEN-ATAPE-SC is correct and it takes $O(2^{n+1} + n)$ time.*

Proof. To prove the correctness of this algorithm, it suffices to prove that for each pair of input i and output j, i can get to j. When the stage control technique is assumed, the network configuration for i to get to j is a number among $0, 1, \ldots, 2^{n+1} - 1$. Since Algorithm GSEN-ATAPE-SC uses every number in $0, 1, \ldots, 2^{n+1}-1$ as one of the network configurations, i can get to j.

The time complexity of Phase 1 is $O(2^{n+1})$. The time complexity of Phase 2 is $O(2^{n+1}+n)$ since it takes $n + 1$ rounds for a message to arrive at its destination processor and therefore each processor receives its first personalized message at round $n + 1$; after that, each processor receives its remaining $N - 1$ personalized messages in the other $2^{n+1} - 1$ rounds. ∎

Note that matrix D needs to be constructed only once. Thus as was mentioned in Section 6 of [17], this kind of matrix can be viewed as a system parameter and therefore the time complexity for constructing it is not counted in the communication delay of all-to-all personalized exchange. We now have the following corollary.

Corollary 14. *When the stage control technique is assumed, Algorithm GSEN-ATAPE-SC is optimal.*

Proof. This corollary follows from Theorems 2 and 13. ∎

5 All-to-all personalized exchange of GSENs with $N \equiv 2 \pmod 4$

In this section, the stage control technique is *not assumed*. The purpose of this section is to propose an optimal all-to-all personalized exchange algorithm for GSENs with $N \equiv 2 \pmod 4$. For convenience, the row index and the column index of a matrix start from 0. Again, a round is the process of transmitting all the messages from the current stage to the next stage.

When the *stage control* technique is assumed, the states of all the switches of a stage have to be identical. With stage control, a single control bit (0 for straight and 1 for cross) can be used to control all the switches of a stage. In this section, we introduce *alternating stage control* (*ASC*), which means the states of the switches of a stage *alternate between straight and cross*. See Figure 7 for an illustration.

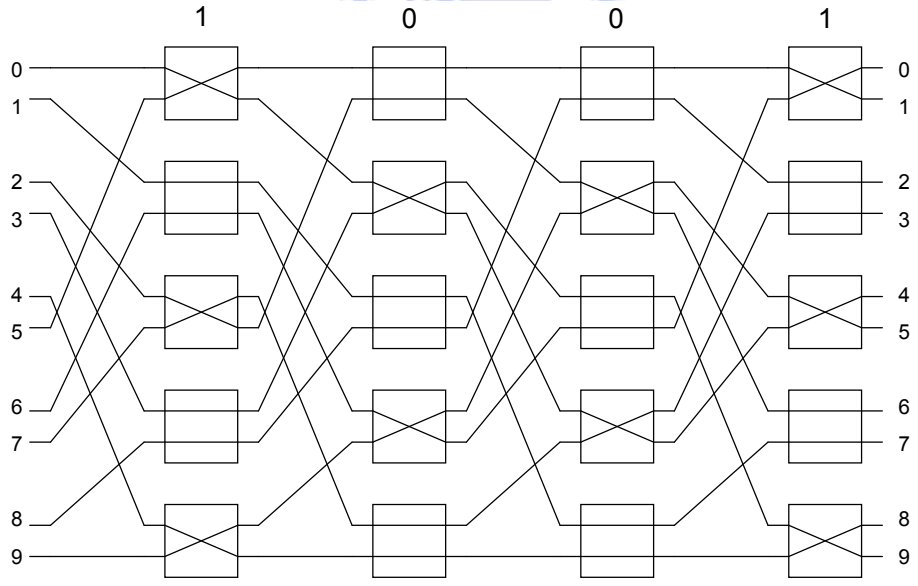

Figure 7: Applying alternating stage control on a 10×10 GSEN; the shown network configuration is $A = 9 = (1001)_2$.

It is not difficult to see that when alternating stage control is used, the network

configuration of a GSEN can be represented by a number as follows. Let a_ℓ denotes the states of the switches at stage $n-\ell$ such that

- $a_\ell = 0$ means the state of the first switch is straight, the state of the second switch is cross, the state of the third switch is straight, and so forth;
- $a_\ell = 1$ means the state of the first switch is cross, the state of the second switch is straight, the state of the third switch is cross, and so forth.

The network configuration of the GSEN can be represented by the number

$$A = a_n 2^n + a_{n-1} 2^{n-1} + \cdots + a_1 2^1 + a_0 2^0$$

or by the binary number

$$(a_n \ a_{n-1} \ \cdots \ a_1 \ a_0)_2.$$

As an example, the network configuration of the GSEN in Figure 7 can be represented by $A = 9 = (1001)_2$. Clearly,

$$0 \le A < 2^{n+1}.$$

We will call a_ℓ the *alternating control bit* of stage $n-\ell$.

We now talk about properties of alternating stage control. Each stage of a GSEN has N input terminals, namely, $\{0, 1, 2, \ldots, N-1\}$. Each stage of a GSEN also has N output terminals, namely, $\{0, 1, 2, \ldots, N-1\}$. When $N \equiv 2 \pmod 4$ and when alternating stage control is used, the N input terminals and N output terminals of stage $n-\ell$ have the following property; see Figure 8 for an illustration.

Property ($*$):

1. If $a_\ell = 0$, then

$$\{0, 2, 4, \ldots, N-2\} \overset{\text{via sub port } 0}{\longrightarrow} \{0, 2, 4, \ldots, N-2\},$$

$$\{1, 3, 5, \ldots, N-1\} \overset{\text{via sub port } 1}{\longrightarrow} \{1, 3, 5, \ldots, N-1\}.$$

That is, an even-numbered input terminal is connected to an even-numbered output terminal via sub port 0, and an odd-numbered input terminal is connected to an odd-numbered output terminal via sub port 1.

2. If $a_\ell = 1$, then

$$\{0, 2, 4, \ldots, N-2\} \overset{\text{via sub port } 1}{\longrightarrow} \{1, 3, 5, \ldots, N-1\},$$

$$\{1, 3, 5, \ldots, N-1\} \overset{\text{via sub port } 0}{\longrightarrow} \{0, 2, 4, \ldots, N-2\}.$$

That is, an even-numbered input terminal is connected to an odd-numbered output terminal via sub port 1, and an odd-numbered input terminal is connected to an even-numbered output terminal via sub port 0.

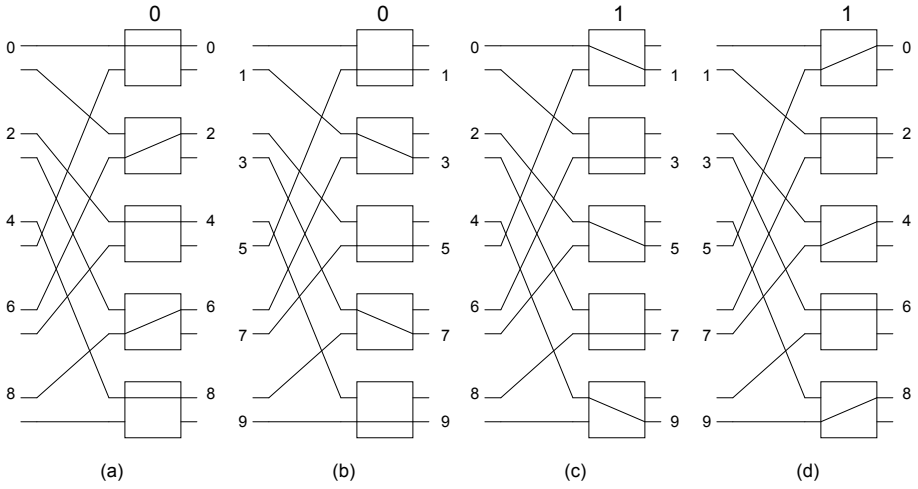

Figure 8: A stage in a 10×10 GSEN. (a) and (b) are for $a_\ell = 0$. (c) and (d) are for $a_\ell = 1$.

It should be noticed that Property $(*)$ holds only when $N \equiv 2 \pmod 4$. If $N \not\equiv 2 \pmod 4$, it does not hold. We now give other properties of alternating stage control.

Theorem 15. *Suppose $N \equiv 2 \pmod 4$, alternating stage control is used, and $A = (a_n\ a_{n-1}\ \cdots\ a_1\ a_0)_2$ is the network configuration. Then the following statements hold:*

1. *The control tags F of inputs $0, 2, 4, \ldots, N-2$ are identical.*

2. *The control tags F' of inputs $1, 3, 5, \ldots, N-1$ are identical.*

3. *The relations between F and F', and F and A are:*

$$F \oplus F' = (11 \cdots 11)_2;$$

$$A = F \oplus \left\lfloor \frac{F}{2} \right\rfloor;$$

$$F = A \oplus \left\lfloor \frac{A}{2} \right\rfloor \oplus \left\lfloor \frac{A}{2^2} \right\rfloor \oplus \cdots \oplus \left\lfloor \frac{A}{2^n} \right\rfloor.$$

Proof. By Property $(*)$, messages from inputs 0, 2, 4, $\ldots$, $N-2$ are via the same sub port at every stage $n-\ell$, $(\ell = n, n-1, \ldots, 0)$. Since the control tag of an input is the sub ports passed by a message sent out from that input, statement 1 holds. By similar arguments, statement 2 also holds.

Let $F = (f_n\ f_{n-1}\ \cdots\ f_1\ f_0)_2$. By Property $(*)$, if messages from inputs $0, 2, 4, \ldots, N-2$ are via sub port f_ℓ at stage $n-\ell$, then messages from inputs $1, 3, 5, \ldots, N-1$ are via sub port $1-f_\ell$ at stage $n-\ell$, $(\ell = n, n-1, \ldots, 0)$. Thus $F \oplus F' = (11 \cdots 11)_2$.

Clearly,

$$a_n = f_n. \tag{1}$$

For $\ell = n-1, n-2, \ldots, 0$, by Property $(*)$, we have:

1. If $a_\ell = 0$, then: $f_\ell = 0$ whenever $f_{\ell+1} = 0$; $f_\ell = 1$ whenever $f_{\ell+1} = 1$.

2. If $a_\ell = 1$, then: $f_\ell = 0$ whenever $f_{\ell+1} = 1$; $f_\ell = 1$ whenever $f_{\ell+1} = 0$.

Therefore,

$$a_\ell = f_\ell \oplus f_{\ell+1},\ (\ell = n-1, n-2, \ldots, 0). \tag{2}$$

By (1) and (2),

$$\begin{aligned} A &= (a_n\ a_{n-1}\ \cdots\ a_1\ a_0)_2 \\ &= (f_n\ \ f_{n-1}\oplus f_n\ \ f_{n-2}\oplus f_{n-1}\ \ \cdots\ \ f_0\oplus f_1)_2 \\ &= (f_n\oplus 0\ \ f_{n-1}\oplus f_n\ \ f_{n-2}\oplus f_{n-1}\ \ \cdots\ \ f_0\oplus f_1)_2 \\ &= (f_n\ f_{n-1}\ f_{n-2}\ \cdots\ f_0)_2 \oplus (0\ f_n\ f_{n-1}\ \cdots\ f_1)_2 \\ &= F \oplus \left\lfloor \frac{F}{2} \right\rfloor. \end{aligned}$$

By (2),

$$f_\ell = a_\ell \oplus a_{\ell+1} \oplus \cdots \oplus a_n,\ (\ell = n-1, n-2, \ldots, 0). \tag{3}$$

Thus by (1) and (3), $F = A \oplus \left\lfloor \frac{A}{2} \right\rfloor \oplus \left\lfloor \frac{A}{2^2} \right\rfloor \oplus \cdots \oplus \left\lfloor \frac{A}{2^n} \right\rfloor$. ∎

Corollary 16. *F and F' are the complement of each other; that is, $F' = \overline{F}$.*

Proof. This follows from $F \oplus F' = (11\cdots11)_2$. ∎

For $k = 0, 1, \ldots, N-1$, define

$$A_k = k \oplus \left\lfloor \frac{k}{2} \right\rfloor$$

and let F_k and F'_k be the control tags of even i and odd i, respectively. We first prove a lemma.

Lemma 17. *$F_k = k$ and $F'_k = 2^{n+1} - 1 - k$.*

Proof. By Theorem 15 and by the definition of A_k,

$$F_k = A_k \oplus \left\lfloor \frac{A_k}{2} \right\rfloor \oplus \left\lfloor \frac{A_k}{2^2} \right\rfloor \oplus \cdots \oplus \left\lfloor \frac{A_k}{2^n} \right\rfloor = k.$$

By Corollary 16, $F'_k = 2^{n+1} - 1 - k$. ∎

We now prove a theorem, which is the foundation of our optimal all-to-all personalized exchange algorithm.

Theorem 18. *When $N \equiv 2 \pmod 4$, the N network configurations $A_0, A_1, \ldots, A_{N-1}$ fulfill an all-to-all communication.*

Proof. Let i be an arbitrary input. To prove this theorem, it suffices to prove that when $A_0, A_1, \ldots, A_{N-1}$ are used, i can get to every output. Let j_k be the destination processor when the network configuration is A_k. First consider the case that $i \in \{0, 2, 4, \ldots, N-2\}$. By Lemma 17, $F_k = k$. Thus $\{F_0, F_1, \ldots, F_{N-1}\} = \{0, 1, \ldots, N-1\}$. By Lemma 6, $j_k = (i \cdot 2^{n+1} + F_k) \mod N$. Therefore $\{j_0, j_1, \ldots, j_{N-1}\} = \{0, 1, \ldots, N-1\}$. This proves that i can get to every output. Now consider the case that $i \in \{1, 3, 5, \ldots, N-1\}$. By Lemma 17, $F'_k = 2^{n+1}-1-k$. Thus $\{F'_0, F'_1, \ldots, F'_{N-1}\} = \{2^{n+1}-1, 2^{n+1}-2, \ldots, 2^{n+1}-N\}$. By Lemma 6, $j_k = (i \cdot 2^{n+1} + F'_k) \mod N$. Therefore $\{j_0, j_1, \ldots, j_{N-1}\} = \{0, 1, \ldots, N-1\}$. This again proves that i can get to every output. ∎

Now we are ready to propose our optimal all-to-all personalized exchange algorithm for GSENs with $N \equiv 2 \pmod 4$. This algorithm consists of two phases: the message preparing phase and the message sending phase. In the message preparing phase, we calculate the destination processor m_i of every input i when the network configuration is A_0; we then use m_i to prepare N personalized messages in the message queue of i.

Let $s_0, s_1, \ldots, s_{\frac{N}{2}-1}$ denote the $\frac{N}{2}$ switches of stage ℓ. To use alternating stage control, the switches of a given GSEN are set according to the following rule.

Rule-AlternatingSC: All the messages sent out at round $k+1$ are equipped with the network configuration A_k. Suppose $A_k = (a_n\ a_{n-1}\ \cdots\ a_1\ a_0)_2$. Then, before all the messages sent out at round $k+1$ enter the switches at stage ℓ, switch s_t at stage ℓ is set to straight if $t + a_{n-\ell}$ is even and set to cross if $t + a_{n-\ell}$ is odd, $(t = 0, 1, \ldots, \frac{N}{2} - 1)$.

An example of Phase 2 of Algorithm GSEN-ATAPE-AlternatingSC is shown in Figures 11 and 12. In these two figures,each 0-1 string is the binary representation of the number A_k with which a message is equipped.

Algorithm GSEN-ATAPE-AlternatingSC
Phase 1: The message preparing phase.
for each processor i $(0 \leq i < N)$ **do in parallel**
calculates m_i by the formula:

$$m_i = \begin{cases} (i \cdot 2^{n+1}) \mod N, & \text{if } i \text{ is even;} \\ ((i+1) \cdot 2^{n+1} - 1) \mod N, & \text{if } i \text{ is odd;} \end{cases}$$

for each k $(0 \leq k < N)$ **do in sequential**

- prepare a personalized message for destination processor

$$\begin{cases} (m_i + k) \mod N, & \text{if } i \text{ is even;} \\ (m_i - k) \mod N, & \text{if } i \text{ is odd;} \end{cases}$$

- equip the personalized message with A_k (which is the network configuration for **Rule-AlternatingSC**) and insert the message into the message queue of i;

Phase 2: The message sending phase
for each processor i $(0 \leq i < N)$ **do in parallel**
for each k $(0 \leq k < N)$ **do in sequential**
send a message in the message queue of i;

We now prove the correctness and analyze the time complexity of the algorithm.

Theorem 19. *Algorithm GSEN-ATAPE-AlternatingSC is correct and it takes $O(N+n)$ time.*

Proof. By Theorem 18, Algorithm GSEN-ATAPE-AlternatingSC fulfills an all-to-all communication. This algorithm is a personalize exchange algorithm if we can show that at round $k+1$, the message sent by processor i will reach the processor $(m_i + k) \mod N$ if i is even and reach $(m_i - k) \mod N$ if i is odd.

First consider the case that $i \in \{0, 2, 4, \ldots, N-2\}$. By Lemma 17, at round $k+1$, the messages sent by processor i will use control tag $F_k = k$. By Lemma 6, the destination processor will be

$$j = (i \cdot 2^{n+1} + k) \mod N = (m_i + k) \mod N.$$

Now consider the case that $i \in \{1, 3, 5, \ldots, N-1\}$. By Lemma 17, at round $k+1$, the messages sent by processor i will use control tag $F'_k = 2^{n+1} - 1 - k$. By Lemma 6, the

destination processor will be

$$\begin{aligned} j &= (i \cdot 2^{n+1} + 2^{n+1} - 1 - k) \mod N \\ &= ((i+1) \cdot 2^{n+1} - 1 - k) \mod N \\ &= (m_i - k) \mod N. \end{aligned}$$

From the above, Algorithm GSEN-ATAPE-AlternatingSC is correct. It is obvious that Phases 1 and 2 of this algorithm take $O(N)$ and $O(N+n)$ time, respectively. Thus the algorithm takes $O(N+n)$ time. ∎

By Lemma 4 and Theorem 19, we have the corollary.

Corollary 20. *Algorithm GSEN-ATAPE-AlternatingSC is optimal.*

We now obtain $\mathcal{R}(N)$ for $N \equiv 2 \pmod 4$.

Theorem 21. *For an $N \times N$ GSEN with $N \equiv 2 \pmod 4$,*

$$\mathcal{R}(N) = N.$$

Proof. By Lemma 1, $\mathcal{R}(N) \geq N$. Since Algorithm GSEN-ATAPE-AlternatingSC can fulfill all-to-all personalized exchange by using N network configurations, namely, $A_0, A_1, \ldots, A_{N-1}$, we also have $\mathcal{R}(N) \leq N$. Hence we have this theorem. ∎

Before ending this section, we show how to modify Algorithm GSEN-ATAPE-AlternatingSC so that a matrix D' like the matrix D used in Algorithm GSEN-ATAPE-SC can be constructed. For convenience, call the modified version Algorithm GSEN-ATAPE-ASC. Algorithm GSEN-ATAPE-ASC has a preprocessing phase, which is used to construct $D' = (d'_{i,k})$ so that $d'_{i,k} = j$ means processor i will send a personalized message to processor j at round $k+1$. Note that D' needs to be constructed only once. Thus, as

Algorithm CONSTRUCT-MATRIX-D'
(preprocessing phase of Algorithm GSEN-ATAPE-ASC)
for each processor i $(0 \leq i < N)$ **do in parallel**
calculates m_i by the formula:

$$m_i = \begin{cases} (i \cdot 2^{n+1}) \mod N, & \text{if } i \text{ is even;} \\ ((i+1) \cdot 2^{n+1} - 1) \mod N, & \text{if } i \text{ is odd;} \end{cases}$$

for each k $(0 \leq k < N)$ **do in sequential**
calculates $d'_{i,k}$ by the formula:

$$d'_{i,k} = \begin{cases} (m_i + k) \mod N, & \text{if } i \text{ is even;} \\ (m_i - k) \mod N, & \text{if } i \text{ is odd;} \end{cases}$$

---

was mentioned in Section 6 of [17], D' can be viewed as a system parameter; it can be pre-computed and can be used again and again.

For example the matrices D' for the GSEN shown in Figure 2 is below.

$$D' = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 0 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 0 & 1 \\ 3 & 2 & 1 & 0 & 9 & 8 & 7 & 6 & 5 & 4 \\ 4 & 5 & 6 & 7 & 8 & 9 & 0 & 1 & 2 & 3 \\ 5 & 4 & 3 & 2 & 1 & 0 & 9 & 8 & 7 & 6 \\ 6 & 7 & 8 & 9 & 0 & 1 & 2 & 3 & 4 & 5 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 9 & 8 \\ 8 & 9 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix}.$$

To illustrate the result of using D', let use define matrix $S' = (s'_{j,k})$ so that $s'_{j,k} = i$ if $d'_{i,k} = j$, which means processor i will send a personalized message to processor j at round $k+1$. S' is similar to the matrix S used in Algorithm GSEN-ATAPE-SC. For example, S' for the above D' is below.

$$S' = \begin{bmatrix} 0 & 1 & 8 & 3 & 6 & 5 & 4 & 7 & 2 & 9 \\ 1 & 0 & 3 & 8 & 5 & 6 & 7 & 4 & 9 & 2 \\ 2 & 3 & 0 & 5 & 8 & 7 & 6 & 9 & 4 & 1 \\ 3 & 2 & 5 & 0 & 7 & 8 & 9 & 6 & 1 & 4 \\ 4 & 5 & 2 & 7 & 0 & 9 & 8 & 1 & 6 & 3 \\ 5 & 4 & 7 & 2 & 9 & 0 & 1 & 8 & 3 & 6 \\ 6 & 7 & 4 & 9 & 2 & 1 & 0 & 3 & 8 & 5 \\ 7 & 6 & 9 & 4 & 1 & 2 & 3 & 0 & 5 & 8 \\ 8 & 9 & 6 & 1 & 4 & 3 & 2 & 5 & 0 & 7 \\ 9 & 8 & 1 & 6 & 3 & 4 & 5 & 2 & 7 & 0 \end{bmatrix}.$$

The following is Algorithm GSEN-ATAPE-ASC. We will only list the algorithm and will not give other details of it.

Algorithm GSEN-ATAPE-ASC
Phase 1: The message preparing phase
for each processor i $(0 \leq i < N)$ **do in parallel**
  for each k $(0 \leq k < N)$ **do in sequential**

- prepare a personalized message for i to sent to $d'_{i,k}$;
- equip the message with A_k (which is the network configuration for **Rule-AlternatingSC**) and insert the message into the message queue of i;

Phase 2: The message sending phase
for each processor i $(0 \leq i < N)$ **do in parallel**
  send a message in the message queue of i;

---

6 Concluding remarks

In [17], Yang and Wang proposed an optimal all-to-all personalized exchange algorithm, called ATAPE, for a class of unique-path, self-routable MINs. To their knowledge, no one has studied all-to-all personalized exchange in this type of MINs. The MINs considered in [17] include the omega network. In this thesis, we consider the generalized shuffle-exchange network (GSEN), which is a generalization of the omega network. Since a GSEN is not necessarily a unique-path MIN, the algorithm ATAPE may not apply. To our knowledge, no one has studied all-to-all personalized exchange in MINs which do not have the unique-path property and do not satisfy $N = 2^{n+1}$.

We have proposed two optimal all-to-all personalized exchange algorithms for GSENs. Unlike algorithm ATAPE, we abandon the requirement on the unique-path property. Our first algorithm uses the stage control technique and works for every even number N. We have proven that it is optimal when the stage control technique is assumed. Our second algorithm does not use the stage control technique and it works for $N \equiv 2 \pmod 4$. We have also proven that it is optimal. Note that our second algorithm does not require

constructing a Latin square in advance and does not require allocating memory for storing the Latin square.

Recall that $\mathcal{R}(N)$ is the minimum number of network configurations required to realize all-to-all personalized exchange in an $N \times N$ GSEN and $\mathcal{R}_{sc}(N)$ is the minimum number of network configurations required to realize all-to-all personalized exchange in an $N \times N$ GSEN when the stage control technique is assumed. In Lemma 1, we proved

$$N \leq \mathcal{R}(N) \leq \mathcal{R}_{sc}(N) \leq 2^{n+1}.$$

Therefore,

$$\mathcal{R}(2^{n+1}) = \mathcal{R}_{sc}(2^{n+1}) = 2^{n+1}.$$

In the proof of Theorem 2, we obtained

$$\mathcal{R}_{sc}(N) = 2^{n+1}.$$

Thus

$$N \leq \mathcal{R}(N) \leq \mathcal{R}_{sc}(N) = 2^{n+1}.$$

By Theorem 21, we have

$$N = \mathcal{R}(N) < \mathcal{R}_{sc}(N) = 2^{n+1}, \text{ if } N \equiv 2 \pmod 4.$$

It remains open to determine $\mathcal{R}(N)$ for $N \equiv 0 \pmod 4$.

References

[1] G. J. Chang, F. K. Hwang, and L. D. Tong, "Characterizing bit permutation networks," *Networks*, vol. 33, no. 4, pp. 261-267, 1999.

[2] Z. Chen, Z. Liu, and Z. Qiu, "Bidirectional shuffle-exchange network and tag-based routing algorithm," *IEEE Commun. Lett.*, vol. 7, no. 3, pp. 121-123, 2003.

[3] C. Chen and J. K. Lou, "An efficient tag-based routing algorithm for the backward network of a bidirectional general shuffle-exchange network," *IEEE Commun. Lett.*, vol. 10, no. 4, pp. 296-298, 2006.

[4] M. Gerla, E. Leonardi, F. Neri, and P. Palnati, "Routing in the bidirectional shuffle net," *IEEE/ACM Trans. Netw.*, vol. 9, no. 1, pp. 91-103, Feb. 2001.

[5] F. K. Hwang, "The Mathematical Theory of Nonblocking Switching Networks," *Series on Applied Mathematics*, vol. 15, ch. 1, pp. 12-22, 2004.

[6] C. P. Kuruskal, "A unified theory of interconnection network structure," *Theor. Comput. Sci.*, vol. 48, pp. 75-94, 1986.

[7] J. K. Lan, W. Y. Chou, and C. Chen, "Efficient routing algorithms for the bidirectional general shuffle-exchange network," submitted for possible publication, 2008.

[8] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Trans. Comput.*, vol. C-24, no. 12, pp. 1145-1155, Dec. 1975.

[9] S. C. Liew, "On the stability if shuffle-exchange and bidirectional shuffle-exchange deflection networkA," *IEEE/ACM Trans. Netw.*, vol. 5, no. 1, pp. 87-94, Feb. 1997.

[10] V. W. Liu, C. Chen, and R. B. Chen, "Optimal all-to-all personalized exchange in d-nary banyan multistage interconnection networks," *J. Comb. Optim.*, vol. 14, pp. 131-142, 2007.

[11] A. Massini, "All-to-all personalized communication on multistage interconnection networks," *Discrete Appl. Math.*, vol. 128, no. 2, pp. 435-446, 2003.

[12] K. Padmanabham, "Design and analysis of even-sized binary shuffle-exchange networks for multiprocessors," *IEEE Trans. Parallel Distrib. Syst.*, vol. 2, no. 4, pp. 385-397, Oct. 1991.

[13] C. Qiao and L. Zhou, "Scheduling switch disjoint connections in stage-controlled photonic banyans," *IEEE Trans. Commun.*, vol. 47, no. 1, pp. 139-148, 1999.

[14] R. Ramaswami, "Multi-wavelength lightwave networks for computer communication," *IEEE Commun. Mag.*, vol. 31, no. 2, pp. 78-88, Feb. 1993.

[15] Y. Yang, J. Wang, "All-to-all personalized exchange in banyan networks," *Proc. Parallel and Distributed Computing and Sysetems (PDCS'99)*, Cambridge, MA, pp. 78-86, 1999.

[16] Y. Yang, J. Wang, "Optimal all-to-all personalized exchange in multistage networks," *Proc. Seventh International Conference on Parallel and Distributed Systems (ICPADS'00)*, Iwale, Japan, 2000.

[17] Y. Yang, J. Wang, "Optimal all-to-all personalized exchange in self-routable multistage networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 3, pp. 261-274, 2000.

[18] Y. Yang, J. Wang, "Optimal all-to-all personalized exchange in a class of optical multistage networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 9, pp. 567-582, Jun. 2001.

[19] Y. Yang, J. Wang, "Routing permutations with link-disjoint and node-disjoint paths in a class of self-routable interconnects," *IEEE Trans. Parallel Distrib. Syst.*, vol. 14, no. 4, pp. 383-393, 2003.

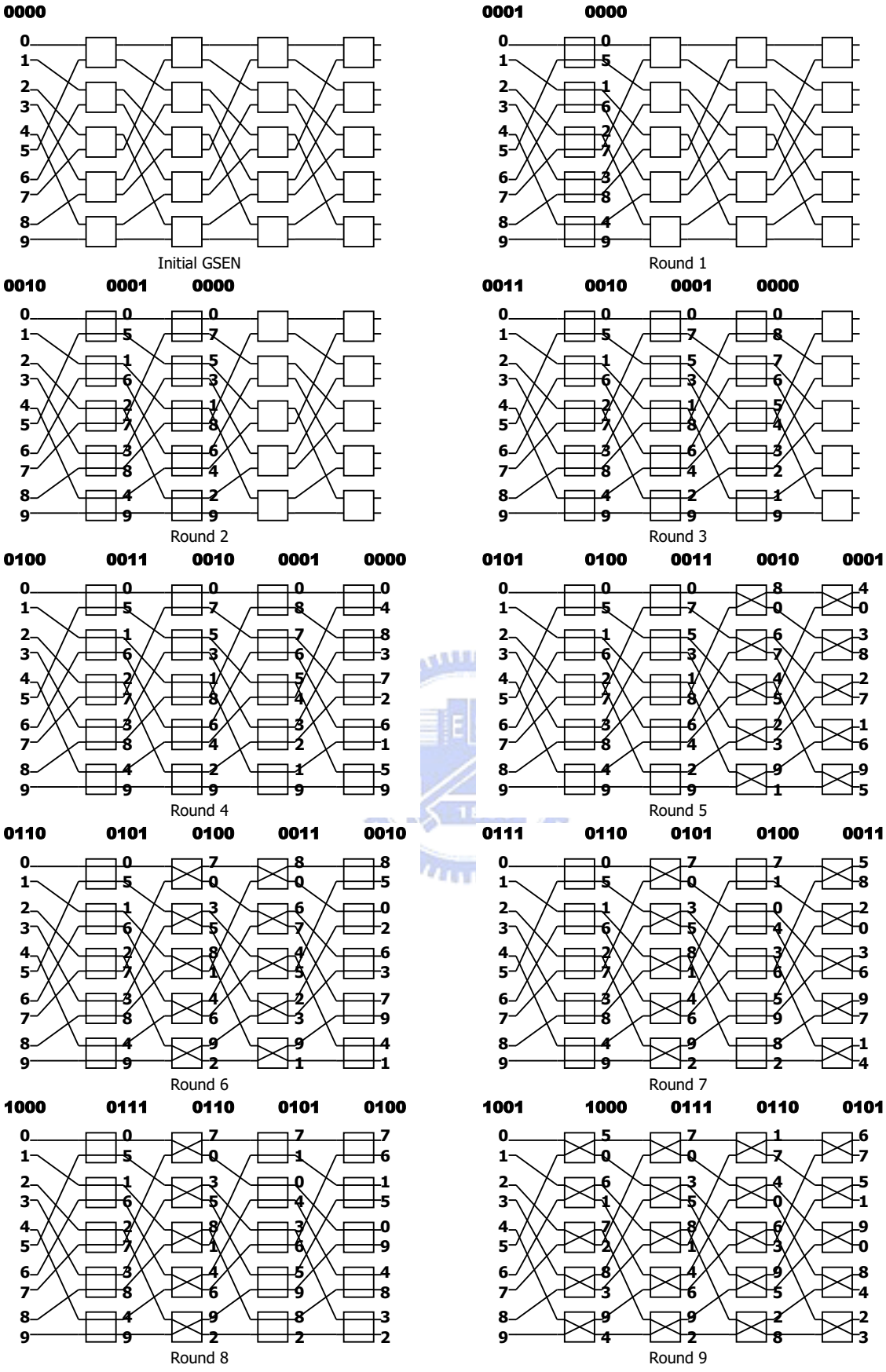

Figure 9: An example of Phase 2 of Algorithm GSEN-ATAPE-SC.

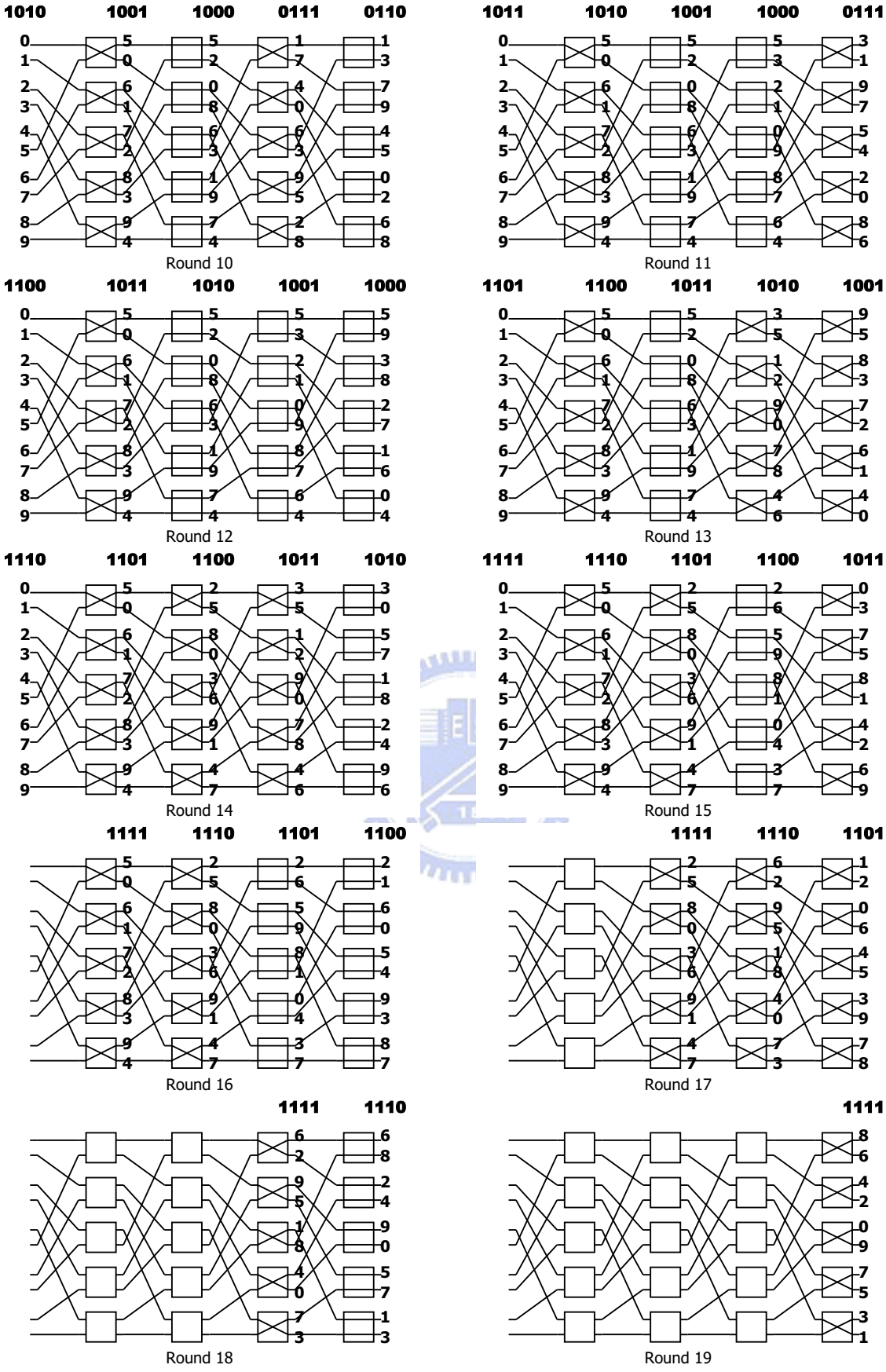

Figure 10: An example of Phase 2 of Algorithm GSEN-ATAPE-SC (continued).

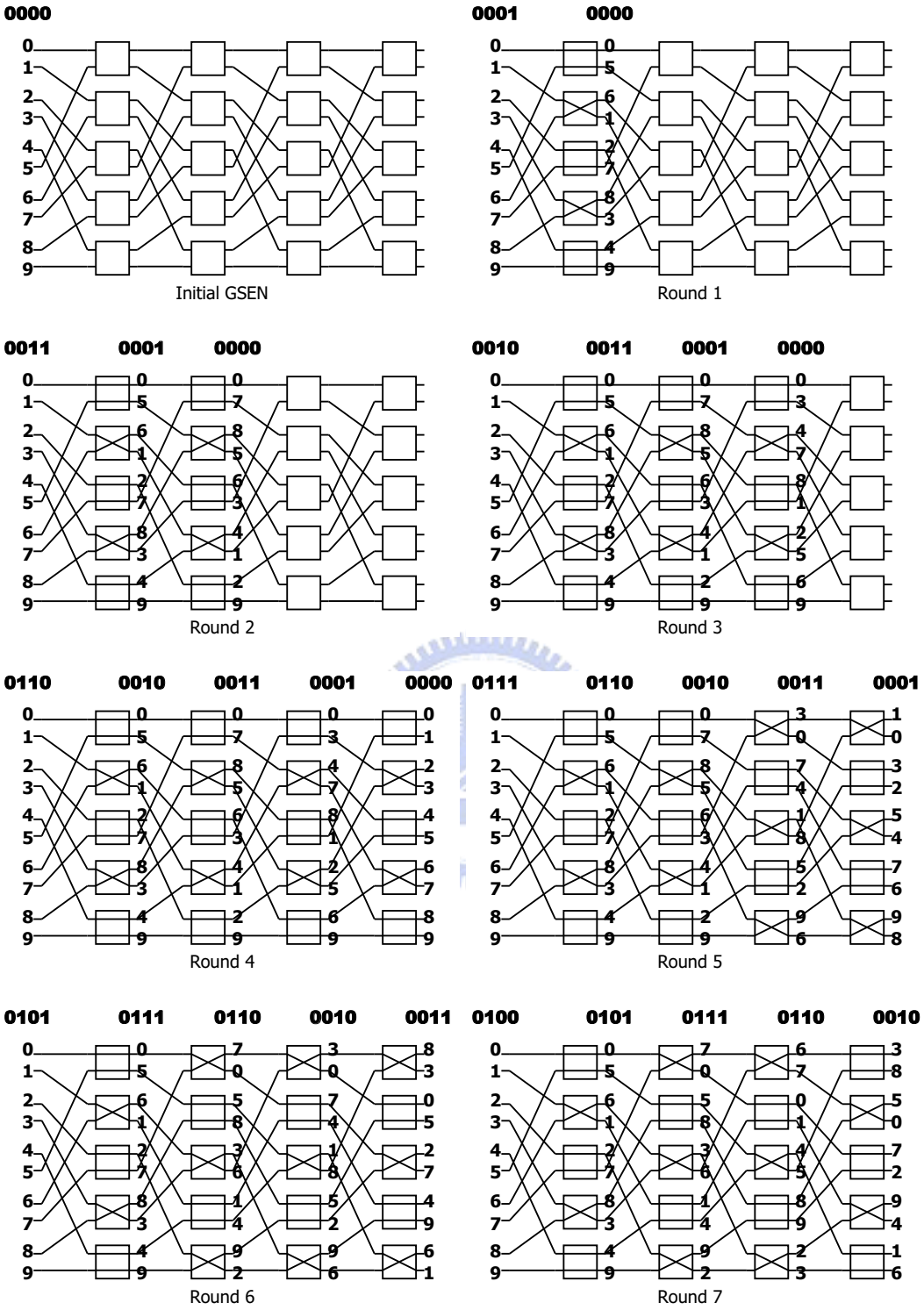

Figure 11: An example of phase 2 of Algorithm GSEN-ATAPE-ASC.

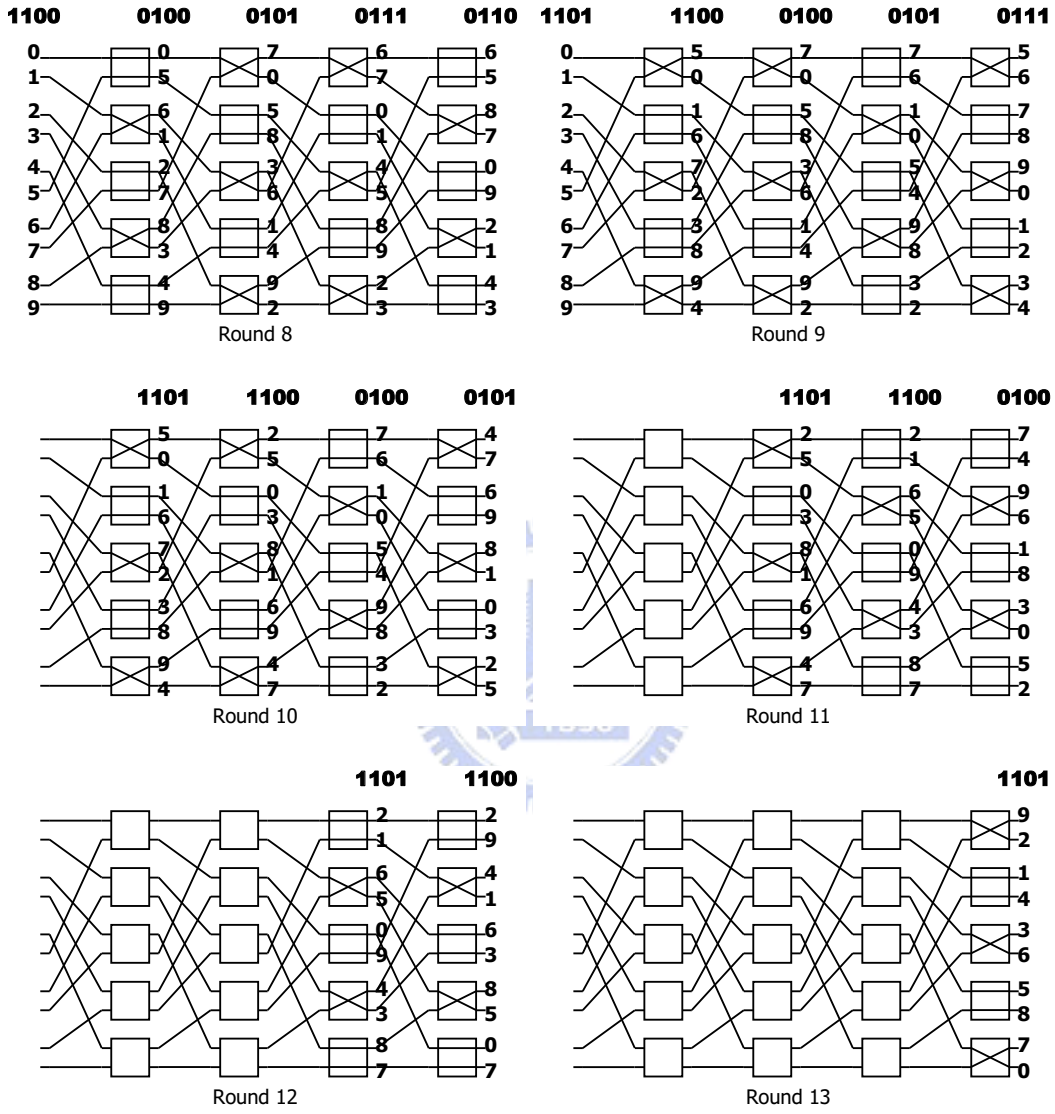

Figure 12: An example of phase 2 of Algorithm GSEN-ATAPE-ASC (continued).