國立交通大學

應用數學系

碩士論文

雙向的廣義 Shuffle-Exchange 網路之逆向網路之快速標記式路線安排演算法

Efficient Tag-Based Routing Algorithms for the Backward Network of a Bidirectional General Shuffle-Exchange Network

研究生: 羅經凱

指導老師: 陳秋媛教授

中華民國九十四年六月

雙向的廣義 Shuffle-Exchange 網路之逆向網路之快速標記式路線安排演算法

Efficient Tag-Based Routing Algorithms for the Backward Network of a Bidirectional General Shuffle-Exchange Network

研究生: 羅經凱　　Student: Jing-Kai Lou
指導老師: 陳秋媛教授　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 Requirement
For the Degree of Master
In
Applied Mathematics
June 2005
Hsinchu, Taiwan, Republic of China

中華民國九十四年六月

雙向的廣義Shuffle-Exchange 網路之逆向網路之快速標記式路線安排演算法

研究生: 羅經凱　　　　指導老師: 陳秋媛 教授

國立交通大學
應用數學系

摘要

在記憶體管理、通訊技術中, shuffle-exchange network 是廣泛被拿來運用以及討論的網路。在1991年的時候, Padmanbhan 定義並討論了廣義的 shuffle-exchange network(簡記為GSEN), 此網路不再限制輸入與輸出個數必為k的次方 (假設 switch element 的 size 均為$k \times k$)。Padmanbhan也提出了快速標記式路線安排演算法。到了2003年, Chen、Liu 以及 Qiu 又將GSEN推廣成所有的連線均為雙向, 並稱之為 bidirectional GSEN。一個 bidirectional GSEN包含了兩個網路: 一個正向網路與一個逆向網路。關於正向網路的路線安排, 可以用 Padmanbhan 所提出的演算法解決; 關於逆向網路的路線安排, Chen、Liu 以及 Qiu 等人也提出了利用 Padmanbhan 的演算法, 先求出正向網路的標記, 然後利用此標記得出逆向網路的路線安排。在這篇論文中, 我們證出了逆向網路具有很好的性質: 對每個終點i而言, 有兩個標記伴隨著它, 任意一個起點j均可利用這兩個標記中的一個, 來安排訊息傳送路線。我們利用此性質做出快速的一對一路線安排演算法, 此演算法在建構 routing table 時, 速度比使用 Chen、Liu 以及 Qiu 所提出的演算法來得快。

關鍵字: 連接網路, 多級式網路, shuffle-exchange 網路, Omega 網路, 標記式路線安排演算法。

中華民國九十四年六月

Efficient Tag-Based Routing Algorithms for the Backward Network of a Bidirectional General Shuffle-Exchange Network

Student: Jing-Kai Lou   Advisor: Chiuyuan Chen

Department of Applied Mathematics
National Chiao Tung University

Abstract

In [7], Padmanbhan proposed the general shuffle-exchange network (GSEN) and an efficient tag-based routing algorithm for it. In [1], Chen, Liu and Qiu further enhanced the GSEN with bidirectional links. The bidirectional GSEN can be divided into two dependent networks, the forward network and the backward network. Since the forward network is a GSEN, Padmanbhan's tag-based routing algorithm can be applied on it. As for the backward network, Chen et al. [1] proposed a routing algorithm which is based on the idea of inversely using the forward control tag. In this thesis, we will show that the backward network has a wonderful property: for each destination i, there are two backward control tags associated with it such that every source j can get to i by using one of the two control tags. We will use this property to derive efficient algorithms for one-to-one routing and for constructing a routing table.

Keywords: Interconnection network, multistage network, shuffle-exchange network, Omega network, tag-based routing algorithm.

Acknowledgement

自認爲原本在大學生活中打混的我, 在上交通研究所前。手足無措、坐立難安倒是很貼切的形容我當時的處境。我自小就不認爲自己是個用功的小孩, 也不是個聰明的學生。而卻能夠很幸運的進入掛著人人羨慕光環下的學校內。我不禁思考著, 今後的研究生涯將會是如何的艱澀呢? 同儕之間的競爭下, 我的能力是否落後人家許多呢? 老師們又是會如何看待我呢? 這些問題像極了該死的蒼蠅環繞著我, 怎樣揮也揮不去。

在正式進入交通應數所這個家族後, 我才著時發覺當初的杞人憂天。陳秋媛老師與翁志文老師所主持的暑期讀書會, 是我接觸這個家族的開端。兩位老師的親切指導以及熱忱的態度, 剎時間化解了我的擔憂。爾後更加有榮幸的能夠進入陳老師門下, 現在回想起來眞是擔心我這輩子的幸運就這樣子通通花在這上面了呢。

陳秋媛老師對於學生該作的研究進度有著審密的計畫, 也會適度的關心學生的近況。讓我這個平常不怎麼用功的學生, 可也得慢慢改掉以往吊兒啷當的缺點。原本對於“老師”都懷著敬畏之心的我在長期與陳老師的meeting 後, 也有著生平第一次跟老師談心的機會。在與老師的談話中也可以感受老師的豐富人生閱歷, 使我理解不少待人處事的道理。在這裡眞的非常感謝陳秋媛老師這些日子以來的照顧以及在學術上以及心靈上的指導。

而在網路上的研究中, 陳老師是啓蒙老師的話。那麼我這篇論文的催生者, 就是黃光明老師了。黃光明老師在我眼中是位很有威嚴卻很和藹的老師。我有幸在老師的課程中, 除了對於網路有了更深一層的理解外。更加幸運的是藉著這個課程讓這篇論文得以誕生。在這裡特別感謝黃光明老師的指導。

另外同儕之間的勉勵也是促使我能夠繼續念下書的原動力, 在這裡特別感謝我的朋友們國元, 維展, 世謙, 亮詮, 明欣以及其他的同學。感謝他們平日對我的照顧, 也特別感謝維展時常不厭其煩的當個好聽衆, 聽我講述論文相關的問題。另外在交通, 我結識了不少學長: 郭俊逸學長, 是一位很好的生活老師, 他指導我很多很多領域上的知識, 是一位很聰明的學長, 也是一位我期許自己能成爲像他的人物。

最後, 一切的一切還是要感謝家人在我背後的支持。讓我可以無有後顧之憂的在校園內生活, 享受如此豐富的資源。如果沒有家人的支持, 一切皆爲空談, 自然也沒有現在的成就。老爸, 老媽以及我那憨厚的老弟感謝你們給了我個這麼棒的家。

Contents

List of Figures

1 Introduction

The purpose of this thesis is to derive tag-based routing algorithms for the backward network of a bidirectional general shuffle-exchange network. Throughout this thesis, N' denotes the number of inputs and the number of outputs of a network. We assume that all the switch elements in a network are identical and of size $k \times k$.

Shuffle-exchange networks have been proposed as a popular architecture for interconnection networks [2, 3, 6, 5, 7, 8]. The *perfect shuffle operation* on N' terminals $(k \mid N')$ is the permutation π defined by

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

In particular, when $k = 2$, the perfect shuffle operation separates the top $N'/2$ terminals from the bottom $N'/2$ terminals and precisely interleaves them, with the bottom terminal still remaining at the bottom. A *shuffle-exchange network* is a network with $N' = k^d$ inputs and outputs and each stage consists of the perfect shuffle on N' terminals followed by N'/k switch elements.

In a multistage interconnection network, a path from an input to an output can be described by a sequence of labels that label the successive edges on this path. Such a sequence is called a *control tag* [7] (or *tag* [1] or *path descriptor* [4]). The control tag may be used as a header for routing a message: each successive node uses the first element of the sequence to route the message, and then discards it. For example, in Figure 1(a), input 2 can get to output 9 by using the control tag 11 (01011), which means input 2 can get to output 9 via sub port 0 at stage 0, sub port 1 at stage 1, sub port 0 at stage 2 and sub port 1 at stage 3 and sub port 1 at stage 4; see Figure 1(b) for an illustration of sub ports.

In a shuffle-exchange network, the number of stages may be equal to or be greater than $\log_k N'$. When the number of stages is exactly $\log_k N'$, a shuffle-exchange network is identical to the Omega network defined in [5] and its control tags depend only on the

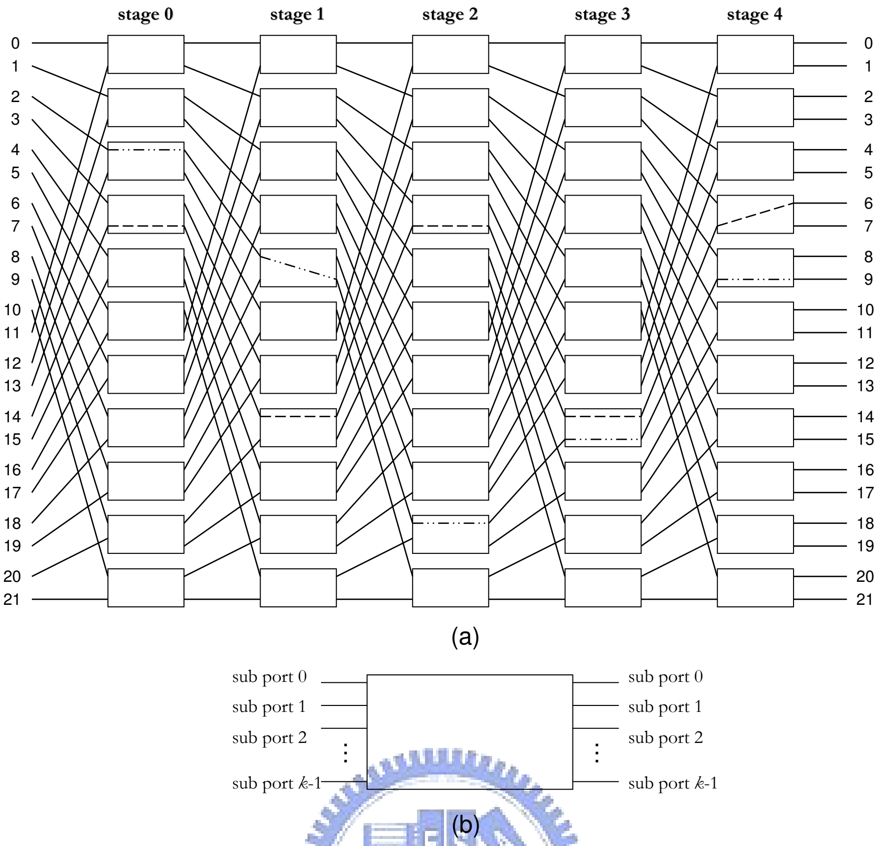


Figure 1: (a) The GSEN with $N' = 22$ and $k = 2$; this figure also shows GSEN(2,11,5). (b) A $k \times k$ switch element and its sub ports.

destination.

In [7], Padmanbhan proposed the *general shuffle-exchange network* (GSEN), which allows $N' \neq k^d$ and contains exactly $\lceil \log_k N' \rceil$ stages. Padmanbhan showed that the control tags of a GSEN depend on both the source and the destination when N' is not a power of k. Padmanbhan also proposed an elegant tag-based routing algorithm for the GSEN.

In [1], Chen, Liu and Qiu enhanced the GSEN with bidirectional links. Their reason for the enhancement is that although unidirectional links are widely used, bidirectional links also have many applications as suggested in [2]. A bidirectional GSEN can be divided into two dependent networks: the *forward network* and the *backward network*. The forward network is from the left-hand side of the network to the right-hand side of the network;

thus a request in it is sent from left to right. On the other hand, the backward network is from the right-hand side of the network to the left-hand side of the network; thus a request in it is sent from right to left. The control tags used in the forward (backward) network are called the *forward* (*backward*) *control tags*.

Since a forward network is a GSEN, Padmanbhan's tag-based routing algorithm can be used in it. As for the backward network, Chen et al. [1] implemented a tag-based routing algorithm by using the forward tag inversely. More precisely, their algorithm first runs Padmanbhan's tag-based routing algorithm to derive the forward control tag; then, their algorithm runs another procedure to convert the forward control tag to the backward control tag. If the number of stages is $n+1$, then the algorithm in [1] takes $O(n)$ time to derive the tag for a source j to get to a destination i and it takes $O(N'^2 n)$ to construct the routing table (a table that contains the backward control tags for routing the $N' \times N'$ pairs of nodes in the backward network).

In this thesis, we show that the backward network has a wonderful property: for each destination i, there are two backward control tags associated with it such that every source j can get to i by using one of the two tags. We show that the two tags can be derived in $O(n)$ time. Therefore, it is possible to derive in $O(n)$ time not only a tag for a j to get to i but also the tags for every j to get to i. So, constructing the routing table can be done in $O(N'n)$ time. We now summarize results of the backward network of a bidirectional GSEN below.

time required to	**use the algorithm in [1]**	**use our algorithm**
find a tag for a j to get to i	$O(n)$	$O(n)$
find the tags for every j to get to i	$O(N'n)$	$O(n)$
construct the routing table	$O(N'^2 n)$	$O(N'n)$

This thesis is organized as follows. In Section 2, we formally define the bidirectional GSEN and give conventions used in this thesis. In Section 3, we describe the tag-based routing algorithms in [7] and [1]. In Section 4, we describe our algorithm.

2 The bidirectional GSEN and conventions used in this thesis

The following definition was given in [1].

Definition. A *bidirectional general shuffle-exchange network* GSEN$(k, r, n+1)$ is a GSEN with bidirectional links. The switch elements are aligned in $n+1$ stages, labelled $0, 1, 2, \ldots, n$. Each stage consists of r switch elements, labelled $0, 1, 2, \ldots, r-1$. And each switch element is a $k \times k$ bidirectional crossbar.

For example, if each link is a bidirectional link, then the network in Figure 1(a) is GSEN(2,11,5). Note that in GSEN$(k, r, n+1)$, there are a total of

$$N' = k \times r$$

ports on each side of a stage, labelled $0, 1, 2, \ldots, N'-1$. The parameters k, r and n satisfy the following equation:

$$\lceil \log_k (k \cdot r) \rceil = \lceil \log_k N' \rceil = n + 1.$$

Throughout this thesis, let

$$N' = N + M, \text{ with } N = k^n \text{ and } k \le M \le (k-1)N. \tag{2.1}$$

The switch elements in the same stage are considered cyclic; that is, switch element labelled 0 is the next switch element of the switch element labelled $r-1$. Also, throughout this thesis, node i is assumed on the left-hand side of the network and node j, the right-hand side. Thus when we say a request is from i to j (j to i), we mean the request is sent through the forward (backward) network.

3 Previous tag-based routing algorithms

A tag-based control routing algorithm is one that sets up a path from an input to an output by using a control tag T. Each digit t_ℓ of the k-ary representation $(t_0 t_1 \ldots t_n)$ of

T controls the switch element at stage ℓ in the path. We now briefly describe previous tag-based routing algorithms of GSEN$(k, r, n+1)$. Recall that GSEN$(k, r, n+1)$ can be divided into the forward network and the backward network. Also recall that the forward network is a GSEN and Padmanbhan's tag-based routing algorithm can be applied on it. The following two theorems were given in [1].

Theorem 1. **[1]** *In the forward network of GSEN*$(k, r, n+1)$, *a path from* i *to* j *can be set up by using the forward control tag* T *given by*

$$T_1 = (j + kMi) \pmod{N'}. \tag{3.2}$$

In addition, other forward control tags (and paths) may be available, specified by

$$T_p = T_1 + (p-1)N' \quad \text{if } T_p < kN,\ 1 < p \le k. \tag{3.3}$$

The backward network is not a GSEN. Thus Padmanbhan's algorithm can not be applied on it. In [1], Chen et al. proposed a tag-based routing algorithm for it by using the forward control tag inversely.

Theorem 2. **[1]** *In the backward network of GSEN*$(k, r, n+1)$, *a path from* j *to* i *can be set up by using the backward control tag* $(s_0 s_1 \ldots s_n)$ *computed by the following procedure:*

Procedure GetBackwardControlTag.

1. Use (3.2) and (3.3) to get the forward control tag T. Derive the k-ary representation $(t_0 t_1 \ldots t_n)$ of T.

2. Get the port sequence $R_0, R_1, \ldots, R_n$ based on $(t_0 t_1 \ldots t_n)$ as follows:

$$R_\ell = \begin{cases} k \cdot i \pmod{N'} + t_0 & \text{if } \ell = 0, \\ k \cdot R_{\ell-1} \pmod{N'} + t_\ell & \text{if } 1 \le \ell \le n. \end{cases}$$

3. Use $R_0, R_1, \ldots, R_n$ to get the backward control tag $(s_0 s_1 \ldots s_n)$ as follows:

$$s_\ell = \begin{cases} \left\lfloor \frac{k \cdot i}{N'} \right\rfloor & \text{if } \ell = 0, \\ \left\lfloor \frac{k \cdot R_{\ell-1}}{N'} \right\rfloor & \text{if } 1 \le \ell \le n. \end{cases}$$

Consider Figure 1(a) as an example. Suppose $j = 9$ wants to get to $i = 2$. In Step 1, we derive $T = 11 = (01011)$. In Step 2, we derive $R_0 = 4, R_1 = 9, R_2 = 18, R_3 = 15$ and $R_4 = 9$. In Step 3, we have $(s_0 s_1 s_2 s_3 s_4) = (00011)$, which means $j = 9$ can get to $i = 2$ via sub port 1 at stage 4, sub port 1 at stage 3, sub port 0 at stage 2, sub port 0 at stage 1 and sub port 0 at stage 0.

Procedure GetBackwardControlTag takes $O(n)$ time to derive the backward control tag for j to get to i. It takes $O(n)$ time to route a one-to-one request and $O(N'^2 \cdot n)$ time to construct the routing table.

4 The one-to-one routing

Recall that i is on the left-hand side of a bidirectional GSEN. Also recall that the switch elements in each stage are labelled $0, 1, 2, \ldots, r-1$ and the next switch element of the switch element labelled $r-1$ is the switch element labelled 0.

The following observations are crucial to our algorithm: At stage 0, only one switch element can get to i. At stage 1, exactly k switch elements can get to i and these switch elements are consecutive. At stage 2, exactly k^2 switch elements can get to i and these switch elements are consecutive. In general, at stage ℓ, $0 \leq \ell \leq n-1$, exactly k^ℓ switch elements can get to i and these switch elements are consecutive. Clearly, at stage n, all the r switch elements can get to i.

Since the switch elements at stage ℓ that can get to i are consecutive, we only need to remember the label of the first one of them. Let C_ℓ denote this label. Clearly, we have

$$C_\ell = i \times k^\ell \pmod{r}.$$

A critical value $v(i)$ associated with i is defined to be

$$v(i) = C_n \times k.$$

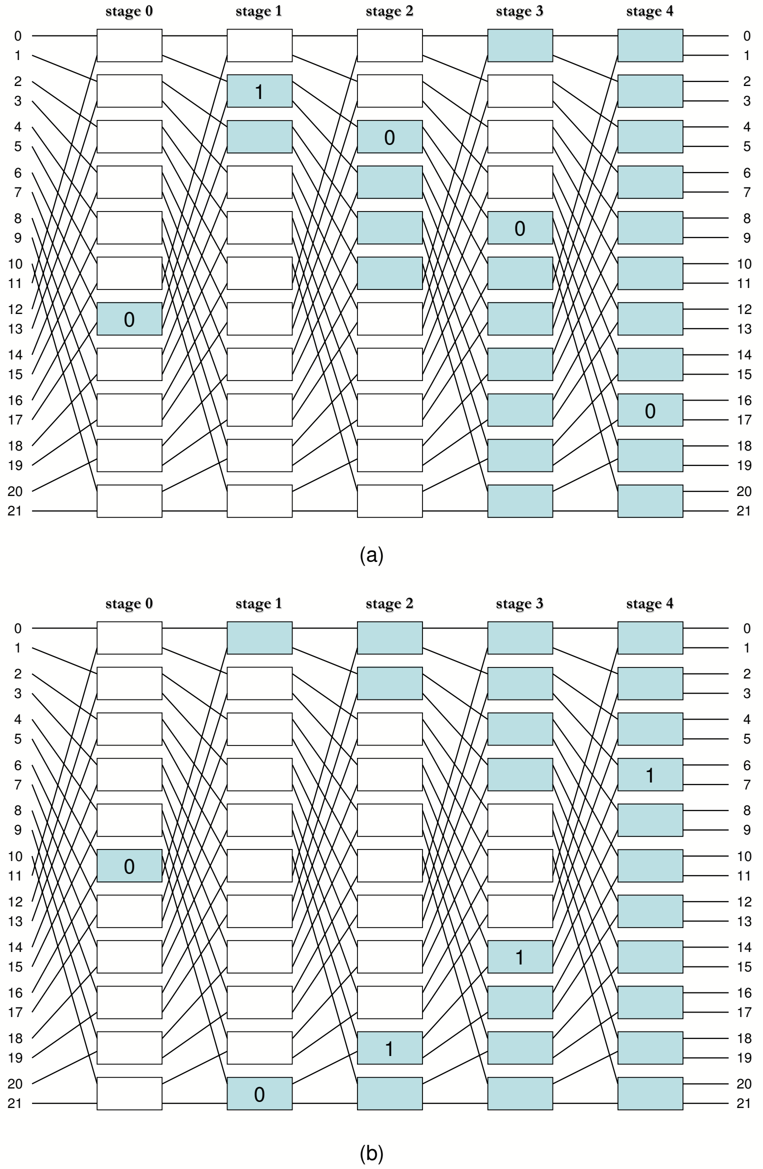
For example, in Figure 2(a), the switch elements that can get to $i = 6$ are highlighted; moreover, $C_0 = 6$, $C_1 = 1$, $C_2 = 2$, $C_3 = 4$, $C_4 = 8$ and $v(i) = 16$. In Figure 2(b), the switch elements that can get to $i = 5$ are highlighted; moreover, $C_0 = 5$, $C_1 = 10$, $C_2 = 9$, $C_3 = 7$, $C_4 = 3$ and $v(i) = 6$. We now propose an algorithm to compute the backward control tags.

BACKWARD-CONTROL-TAGS.

Input: i on the left-hand side of a bidirectional GSEN$(k, r, n+1)$.

Output: The critical value $v(i)$ and two control tags $(s_0 s_1 \dots s_n)$ and $(s'_0 s'_1 \dots s'_n)$.

```
1. /* Compute $C_0, C_1, \dots, C_n$. */

    for $\ell = 0$ to $n$ do

        $C_\ell \leftarrow i \times k^\ell \pmod{r}$;

2. /* Compute the critical value $v(i)$. */
    $v(i) \leftarrow C_n \times k$;

3. /* Compute $s'_0, s'_1, \dots, s'_n$. */
    $s'_0 \leftarrow \left\lfloor \frac{i}{r} \right\rfloor$;
    for $\ell = 1$ to $n$ do
        $s'_\ell \leftarrow \left\lfloor \frac{k \times C_{\ell-1}}{r} \right\rfloor$;

4. /* Compute $F_0, F_1, \dots, F_n$. */

    if $(r - C_{n-1}) \times k \geq r$

    then

        begin
          for $\ell = 0$ to $n-1$ do $F_\ell \leftarrow 0$;
          $F_n \leftarrow 1$;
        end

    else

        for $\ell = 0$ to $n$ do
            if $C_\ell + k^\ell > r$ then $F_\ell \leftarrow 1$ else $F_\ell \leftarrow 0$;

5. /* Compute $s_0, s_1, \dots, s_n$. */

    for $\ell = 0$ to $n$ do

        $s_\ell \leftarrow s'_\ell + F_\ell \pmod{k}$;
```


(a)

(b)

Figure 2: GSEN(2,11,5) with the switch elements that can get to (a) $i = 6$ and (b) $i = 5$ being highlighted.

Again, consider Figure 2 (a) as an example. Then $k = 2$, $r = 11$ and $n = 4$. Suppose $i = 6$. Then after Step 1, $C_0 = 6$, $C_1 = 1$, $C_2 = 2$, $C_3 = 4$ and $C_4 = 8$. After Step 2, $v(i) = 16$. After Step 3, $(s'_0 s'_1 s'_2 s'_3 s'_4) = (01000)$. After Step 4, $F_0 = 0$, $F_1 = 0$, $F_2 = 0$, $F_3 = 0$ and $F_4 = 1$. After Step 5, $(s_0 s_1 s_2 s_3 s_4) = (01001)$. It is easy to verify that: if $j < 16$, then j can get to 6 by using the tag (01001); if $j \geq 16$, then j can get to 6 by using the tag (01000). We summarize the above results in the following table.

destination i	$(s_0 s_1 s_2 s_3 s_4)$	$(s'_0 s'_1 s'_2 s'_3 s'_4)$	$v(i)$
$i = 6$	01001	01000	16

Recall that there are a total of N' ports on each side of a stage, labelled $0, 1, 2, \ldots, N'-1$. A port R consists of two parts: the number y of the switch element where R is located, and the sub port number z in the switch element where R is located; see [1]. R and y and z satisfy $R = ky + z$. The following result was proved in [1].

Lemma 3. [1] *Suppose port u of stage $\ell - 1$ and port v of stage ℓ are connected by a link, where $u = ky_1 + z_1$ and $v = ky_2 + z_2$. Then $z_2 = \left\lfloor \frac{k \cdot u}{N'} \right\rfloor$.*

Thus we have

Lemma 4. *Let u, v, y_1, z_1, y_2, z_2 be defined as in Lemma 3 and consider the switch elements labelled y_1 and y_2. Then the backward control tag for y_2 to get to y_1 (or to get to u) is z_2; moreover, $z_2 = \left\lfloor \frac{u}{r} \right\rfloor$.*

Proof. Clearly, the tag is z_2. Since $N' = k \times r$, by Lemma 3, $z_2 = \left\lfloor \frac{u}{r} \right\rfloor$. ∎

We now prove that

Lemma 5. *If $j = v(i)$, then j can get to i by using the tag $(s'_0 s'_1 \ldots s'_n)$.*

Proof. Suppose $j = v(i)$. Then j can get to i via switch elements labelled $C_n, C_{n-1}, \ldots, C_0$. For each ℓ, $1 \leq \ell \leq n$, C_ℓ is linked to $C_{\ell-1}$ via sub port 0 of $C_{\ell-1}$. Sub port 0 of $C_{\ell-1}$ is port u of $C_{\ell-1}$, where $u = k \times C_{\ell-1}$. Thus by Lemma 4, the tag for C_ℓ to get to $C_{\ell-1}$ is $\left\lfloor \frac{k \times C_{\ell-1}}{r} \right\rfloor$. Also by Lemma 4, the tag for C_0 to get to i is $\left\lfloor \frac{i}{r} \right\rfloor$. In Step 3 of BACKWARD-CONTROL-TAGS, we set $s'_0 = \left\lfloor \frac{i}{r} \right\rfloor$ and $s'_\ell = \left\lfloor \frac{k \times C_{\ell-1}}{r} \right\rfloor$, for $\ell = 1, 2, \ldots, n$. Thus we have this lemma. ∎

Lemma 6. *If $j > v(i)$, then j can get to i by using the tag $(s'_0 s'_1 \ldots s'_n)$.*

Proof. By (2.1), $k^n < N' \le k^{n+1}$. Set $d = j - v(i)$ for easy writing. Then $0 < d \le N' - 1$. Thus $0 < \frac{d}{k^{n-\ell+1}} \le \frac{N'-1}{\frac{k^{n+1}}{k^{\ell}}} \le \frac{N'-1}{N'}k^{\ell} < k^{\ell}$ and therefore $0 \le \lfloor \frac{d}{k^{n-\ell+1}} \rfloor < k^{\ell}$. Recall that at stage n, all of the r switch elements can get to i; at stage ℓ, $0 \le \ell \le n-1$, there are exactly k^{ℓ} consecutive switch elements that can get to i and the first one is labelled C_ℓ. Thus j can get to i via switch elements labelled $C_n + \lfloor \frac{d}{k} \rfloor$, $C_{n-1} + \lfloor \frac{d}{k^2} \rfloor$, $C_{n-2} + \lfloor \frac{d}{k^3} \rfloor$, $\cdots$, $C_\ell + \lfloor \frac{d}{k^{n-\ell+1}} \rfloor$, $\cdots$, $C_1 + \lfloor \frac{d}{k^n} \rfloor$, $C_0 + \lfloor \frac{d}{k^{n+1}} \rfloor$. The connection of a GSEN ensures that if C_ℓ, $1 \le \ell \le n$, is connected to $C_{\ell-1}$ via sub port z_2, then $C_\ell + \lfloor \frac{d}{k^{n-\ell+1}} \rfloor$ is connected to $C_{\ell-1} + \lfloor \frac{d}{k^{n-\ell+2}} \rfloor$ via sub port z_2. By Lemma 4, the tag for $C_\ell + \lfloor \frac{d}{k^{n-\ell+1}} \rfloor$ to get to $C_{\ell-1} + \lfloor \frac{d}{k^{n-\ell+2}} \rfloor$ is z_2; by Lemma 5, $z_2 = s'_\ell$. Note that $0 < \frac{d}{k^{n+1}} \le \frac{N'-1}{N'} < 1$. Thus $C_0 + \lfloor \frac{d}{k^{n+1}} \rfloor = C_0$. By Lemma 5, the tag for C_0 to get to i is s'_0. From the above, if $j > v(i)$, then j can get to i by using the tag $(s'_0 s'_1 \ldots s'_n)$. ∎

Lemma 7. *If $j < v(i)$ and $(r - C_{n-1}) \times k \ge r$, then j can get to i by using the tag $(s_0 s_1 \ldots s_n)$.*

Proof. Set $d = j - v(i) + N'$ for easy writing. Then j can get to i via switch elements labelled $C_n + \lfloor \frac{d}{k} \rfloor - r$, $C_{n-1} + \lfloor \frac{d}{k^2} \rfloor$, $C_{n-2} + \lfloor \frac{d}{k^3} \rfloor$, $\cdots$, $C_\ell + \lfloor \frac{d}{k^{n-\ell+1}} \rfloor$, $\cdots$, $C_1 + \lfloor \frac{d}{k^n} \rfloor$, $C_0 + \lfloor \frac{d}{k^{n+1}} \rfloor$. The connection of a GSEN ensures that if C_n is connected to C_{n-1} via sub port z_2, then $C_n + \lfloor \frac{d}{k} \rfloor - r$ is connected to $C_{n-1} + \lfloor \frac{d}{k^2} \rfloor$ via sub port $z_2 + 1 \pmod{k}$. By Lemma 4, the tag for $C_n + \lfloor \frac{d}{k} \rfloor - r$ to get to $C_{n-1} + \lfloor \frac{d}{k^2} \rfloor$ is $z_2 + 1 \pmod{k}$. By Lemma 5, $z_2 = s'_n$. In our algorithm, we set $F_n = 1$ and set $s_n = s'_n + F_n \pmod{k}$. Thus $s_n = z_2 + 1 \pmod{k}$. Again, the connection of a GSEN ensures that if C_ℓ, $1 \le \ell \le n-1$, is connected to $C_{\ell-1}$ via sub port z_2, then $C_\ell + \lfloor \frac{d}{k^{n-\ell+1}} \rfloor$ is connected to $C_{\ell-1} + \lfloor \frac{d}{k^{n-\ell+2}} \rfloor$ via sub port z_2. By Lemma 4, the tag for $C_\ell + \lfloor \frac{d}{k^{n-\ell+1}} \rfloor$ to get to $C_{\ell-1} + \lfloor \frac{d}{k^{n-\ell+2}} \rfloor$ is z_2. By Lemma 5, $z_2 = s'_\ell$. In our algorithm, we set $F_\ell = 0$ and set $s_\ell = s'_\ell + F_\ell \pmod{k}$. Thus $s_\ell = z_2$. Note that $0 < \frac{d}{k^{n+1}} \le \frac{N'-1}{N'} < 1$. Thus $C_0 + \lfloor \frac{d}{k^{n+1}} \rfloor = C_0$. By Lemma 5, the tag for C_0 to get to i is s'_0. In our algorithm, we set $F_\ell = 0$ and set $s_0 = s'_0 + F_0 \pmod{k}$. Thus $s_0 = s'_0$. We now have this lemma. ∎

Lemma 8. *If $j < v(i)$ and $(r - C_{n-1}) \times k < r$, then j can get to i by using the tag $(s_0 s_1 \ldots s_n)$.*

Proof. Set $d = j - v(i) + N'$ for easy writing. Then j can get to i via switch elements labelled L_n, L_{n-1}, $\cdots$, L_ℓ, $\cdots$, L_1, L_0, where

$$L_n = C_n + \left\lfloor \frac{d}{k} \right\rfloor - r$$

and for $\ell = n-1, n-2, \ldots, 0$,

$$L_\ell = \begin{cases} C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor & \text{if } C_\ell + k^\ell \leq r, \\ C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor - r & \text{if } C_\ell + k^\ell > r. \end{cases}$$

The connection of a GSEN ensures that if C_n is connected to C_{n-1} via sub port z_2, then L_n is connected to L_{n-1} via sub port $z_2{+}1 \pmod{k}$. By Lemma 4, the tag for L_n to get to L_{n-1} is $z_2{+}1 \pmod{k}$. By Lemma 5, $z_2 = s'_n$. Note that $C_n{+}k^n > r$. Thus our algorithm sets $F_n = 1$. Since our algorithm sets $s_n = s'_n + F_n \pmod{k}$, clearly $s_n = z_2{+}1 \pmod{k}$. Again, the connection of a GSEN ensures that if C_ℓ, $1 \leq \ell \leq n-1$, is connected to $C_{\ell-1}$ via sub port z_2, then L_ℓ is connected to $L_{\ell-1}$ via sub port z_2 if $L_\ell = C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor$ and via sub port $z_2{+}1 \pmod{k}$ if $L_\ell = C_\ell{+}\left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor{-}r$. Thus by Lemma 4, the tag for L_ℓ to get to $L_{\ell-1}$ is z_2 if $L_\ell = C_\ell{+}\left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor$ and is $z_2{+}1 \pmod{k}$ if $L_\ell = C_\ell{+}\left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor{-}r$. By Lemma 5, $z_2 = s'_n$. In our algorithm, we set $F_\ell = 0$ if $C_\ell + k^\ell \leq r$ (i.e., if $L_\ell = C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor$), set $F_\ell = 1$ if $C_\ell + k^\ell > r$ (i.e., if $L_\ell = C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor - r$) and set $s_\ell = s'_\ell + F_\ell \pmod{k}$. Thus $s_\ell = z_2$ if $L_\ell = C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor$ and $s_\ell = z_2 + 1 \pmod{k}$ if $L_\ell = C_\ell + \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor - r$. Note that $0 < \frac{d}{k^{n+1}} \leq \frac{N'-1}{N'} < 1$. Thus $L_0 = C_0$. By Lemma 5, the tag for L_0 to get to i is s'_0. Note that $C_0 + k^0 \leq r$. Thus our algorithm sets $F_0 = 0$ and set $s_0 = s'_0 + F_0 \pmod{k}$. Thus $s_0 = s'_0$. We now have this lemma. ∎

Theorem 9. *If $j < v(i)$, then j can get to i by using the backward control tag $(s_0 s_1 \ldots s_n)$; if $j \geq v(i)$, then j can get to i by using the backward control tag $(s'_0 s'_1 \ldots s'_n)$. Moreover, it takes $O(n)$ time to compute $v(i)$, $(s_0 s_1 \ldots s_n)$ and $(s'_0 s'_1 \ldots s'_n)$.*

Proof. It is obvious that it takes $O(n)$ time to compute $v(i)$, $(s_0 s_1 \ldots s_n)$ and $(s'_0 s'_1 \ldots s'_n)$. This theorem now follows from Lemma 5, Lemma 6, Lemma 7 and Lemma 8. ∎

The following is a one-to-one routing algorithm for the backward network of a bidirectional GSEN.

ONE-TO-ONE.

Input: i on the left-hand side and j on the right-hand side of a bidirectional GSEN $(k, r, n+1)$.

Output: The backward control tag for j to get to i.

```
1. Use BACKWARD-CONTROL-TAGS to derive $v(i)$, $(s_0 s_1 \ldots s_n)$ and $(s'_0 s'_1 \ldots s'_n)$;
2. if $j < v(i)$ then return $(s_0 s_1 \ldots s_n)$ else return $(s'_0 s'_1 \ldots s'_n)$;
```

It is obvious that algorithm ONE-TO-ONE takes $O(n)$ time.

5 The routing table and the all-to-all routing

In this section, we will propose an algorithm to construct the routing table of the backward network of a bidirectional GSEN. This algorithm is based on the one-to-one routing algorithm proposed in the previous section and can be used for the all-to-all routing.

ROUTING-TABLE.

Input: A bidirectional GSEN$(k, r, n+1)$.

Output: Its routing table.

```
1. /* Recall the function all to one */
     for $i = 0$ to $N' - 1$ do
        run algorithm BACKWARD-CONTROL-TAGS for $i$ and GSEN$(k, r, n+1)$;
     endfor;
```

It is obvious that algorithm ROUTING-TABLE takes $O(N'n)$ time. In the appendix, we list the computer output of the routing tables derived by algorithm ROUTING-TABLE for $N' = 18, 20, 22, \ldots, 32$. Note that in the table of $N' = 32$, each $v(i)$ is zero, which means we can get to every i by using only one tag. This result reflects the known result that when the number of stages is exactly $\log_k N'$, a shuffle-exchange network is identical to the Omega network defined in [5] and its control tags depend only on the destination.

A Backward control tags for $N' = 18, 20, \ldots, 32$

GSEN(2, 9, 5)			
i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	14
2	0 0 1 0 0	0 0 0 1 1	10
3	0 0 1 1 0	0 0 1 0 1	6
4	0 1 0 0 0	0 0 1 1 1	2
5	0 1 0 0 1	0 1 0 0 0	16
6	0 1 0 1 1	0 1 0 1 0	12
7	0 1 1 0 1	0 1 1 0 0	8
8	0 1 1 1 1	0 1 1 1 0	4
9	1 0 0 0 1	1 0 0 0 0	0
10	1 0 0 1 0	1 0 0 0 1	14
11	1 0 1 0 0	1 0 0 1 1	10
12	1 0 1 1 0	1 0 1 0 1	6
13	1 1 0 0 0	1 0 1 1 1	2
14	1 1 0 0 1	1 1 0 0 0	16
15	1 1 0 1 1	1 1 0 1 0	12
16	1 1 1 0 1	1 1 1 0 0	8
17	1 1 1 1 1	1 1 1 1 0	4

GSEN(2, 10, 5)			
i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	12
2	0 0 1 0 0	0 0 0 1 1	4
3	0 0 1 0 1	0 0 1 0 0	16
4	0 0 1 1 1	0 0 1 1 0	8
5	0 1 0 0 1	0 1 0 0 0	0
6	0 1 0 1 0	0 1 0 0 1	12
7	0 1 1 0 0	0 1 0 1 1	4
8	0 1 1 0 1	0 1 1 0 0	16
9	0 1 1 1 1	0 1 1 1 0	8
10	1 0 0 0 1	1 0 0 0 0	0
11	1 0 0 1 0	1 0 0 0 1	12
12	1 0 1 0 0	1 0 0 1 1	4
13	1 0 1 0 1	1 0 1 0 0	16
14	1 0 1 1 1	1 0 1 1 0	8
15	1 1 0 0 1	1 1 0 0 0	0
16	1 1 0 1 0	1 1 0 0 1	12
17	1 1 1 0 0	1 1 0 1 1	4
18	1 1 1 0 1	1 1 1 0 0	16
19	1 1 1 1 1	1 1 1 1 0	8

GSEN(2, 11, 5)

i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	10
2	0 0 0 1 1	0 0 0 1 0	20
3	0 0 1 0 1	0 0 1 0 0	8
4	0 0 1 1 0	0 0 1 0 1	18
5	0 1 0 0 0	0 0 1 1 1	6
6	0 1 0 0 1	0 1 0 0 0	16
7	0 1 0 1 1	0 1 0 1 0	4
8	0 1 1 0 0	0 1 0 1 1	14
9	0 1 1 1 0	0 1 1 0 1	2
10	0 1 1 1 1	0 1 1 1 0	12
11	1 0 0 0 1	1 0 0 0 0	0
12	1 0 0 1 0	1 0 0 0 1	10
13	1 0 0 1 1	1 0 0 1 0	20
14	1 0 1 0 1	1 0 1 0 0	8
15	1 0 1 1 0	1 0 1 0 1	18
16	1 1 0 0 0	1 0 1 1 1	6
17	1 1 0 0 1	1 1 0 0 0	16
18	1 1 0 1 1	1 1 0 1 0	4
19	1 1 1 0 0	1 1 0 1 1	14
20	1 1 1 1 0	1 1 1 0 1	2
21	1 1 1 1 1	1 1 1 1 0	12

GSEN(2, 12, 5)

i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	8
2	0 0 0 1 1	0 0 0 1 0	16
3	0 0 1 0 1	0 0 1 0 0	0
4	0 0 1 1 0	0 0 1 0 1	8
5	0 0 1 1 1	0 0 1 1 0	16
6	0 1 0 0 1	0 1 0 0 0	0
7	0 1 0 1 0	0 1 0 0 1	8
8	0 1 0 1 1	0 1 0 1 0	16
9	0 1 1 0 1	0 1 1 0 0	0
10	0 1 1 1 0	0 1 1 0 1	8
11	0 1 1 1 1	0 1 1 1 0	16
12	1 0 0 0 1	1 0 0 0 0	0
13	1 0 0 1 0	1 0 0 0 1	8
14	1 0 0 1 1	1 0 0 1 0	16
15	1 0 1 0 1	1 0 1 0 0	0
16	1 0 1 1 0	1 0 1 0 1	8
17	1 0 1 1 1	1 0 1 1 0	16
18	1 1 0 0 1	1 1 0 0 0	0
19	1 1 0 1 0	1 1 0 0 1	8
20	1 1 0 1 1	1 1 0 1 0	16
21	1 1 1 0 1	1 1 1 0 0	0
22	1 1 1 1 0	1 1 1 0 1	8
23	1 1 1 1 1	1 1 1 1 0	16

GSEN(2, 13, 5)

i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	6
2	0 0 0 1 1	0 0 0 1 0	12
3	0 0 1 0 0	0 0 0 1 1	18
4	0 0 1 0 1	0 0 1 0 0	24
5	0 0 1 1 1	0 0 1 1 0	4
6	0 1 0 0 0	0 0 1 1 1	10
7	0 1 0 0 1	0 1 0 0 0	16
8	0 1 0 1 0	0 1 0 0 1	22
9	0 1 1 0 0	0 1 0 1 1	2
10	0 1 1 0 1	0 1 1 0 0	8
11	0 1 1 1 0	0 1 1 0 1	14
12	0 1 1 1 1	0 1 1 1 0	20
13	1 0 0 0 1	1 0 0 0 0	0
14	1 0 0 1 0	1 0 0 0 1	6
15	1 0 0 1 1	1 0 0 1 0	12
16	1 0 1 0 0	1 0 0 1 1	18
17	1 0 1 0 1	1 0 1 0 0	24
18	1 0 1 1 1	1 0 1 1 0	4
19	1 1 0 0 0	1 0 1 1 1	10
20	1 1 0 0 1	1 1 0 0 0	16
21	1 1 0 1 0	1 1 0 0 1	22
22	1 1 1 0 0	1 1 0 1 1	2
23	1 1 1 0 1	1 1 1 0 0	8
24	1 1 1 1 0	1 1 1 0 1	14
25	1 1 1 1 1	1 1 1 1 0	20

GSEN(2, 14, 5)

i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	4
2	0 0 0 1 1	0 0 0 1 0	8
3	0 0 1 0 0	0 0 0 1 1	12
4	0 0 1 0 1	0 0 1 0 0	16
5	0 0 1 1 0	0 0 1 0 1	20
6	0 0 1 1 1	0 0 1 1 0	24
7	0 1 0 0 1	0 1 0 0 0	0
8	0 1 0 1 0	0 1 0 0 1	4
9	0 1 0 1 1	0 1 0 1 0	8
10	0 1 1 0 0	0 1 0 1 1	12
11	0 1 1 0 1	0 1 1 0 0	16
12	0 1 1 1 0	0 1 1 0 1	20
13	0 1 1 1 1	0 1 1 1 0	24
14	1 0 0 0 1	1 0 0 0 0	0
15	1 0 0 1 0	1 0 0 0 1	4
16	1 0 0 1 1	1 0 0 1 0	8
17	1 0 1 0 0	1 0 0 1 1	12
18	1 0 1 0 1	1 0 1 0 0	16
19	1 0 1 1 0	1 0 1 0 1	20
20	1 0 1 1 1	1 0 1 1 0	24
21	1 1 0 0 1	1 1 0 0 0	0
22	1 1 0 1 0	1 1 0 0 1	4
23	1 1 0 1 1	1 1 0 1 0	8
24	1 1 1 0 0	1 1 0 1 1	12
25	1 1 1 0 1	1 1 1 0 0	16
26	1 1 1 1 0	1 1 1 0 1	20
27	1 1 1 1 1	1 1 1 1 0	24

GSEN(2, 15, 5)			
i	$s_0s_1s_2s_3s_4s_5$	$s'_0s'_1s'_2s'_3s'_4s'_5$	$v(i)$
0	0 0 0 0 1	0 0 0 0 0	0
1	0 0 0 1 0	0 0 0 0 1	2
2	0 0 0 1 1	0 0 0 1 0	4
3	0 0 1 0 0	0 0 0 1 1	6
4	0 0 1 0 1	0 0 1 0 0	8
5	0 0 1 1 0	0 0 1 0 1	10
6	0 0 1 1 1	0 0 1 1 0	12
7	0 1 0 0 0	0 0 1 1 1	14
8	0 1 0 0 1	0 1 0 0 0	16
9	0 1 0 1 0	0 1 0 0 1	18
10	0 1 0 1 1	0 1 0 1 0	20
11	0 1 1 0 0	0 1 0 1 1	22
12	0 1 1 0 1	0 1 1 0 0	24
13	0 1 1 1 0	0 1 1 0 1	26
14	0 1 1 1 1	0 1 1 1 0	28
15	1 0 0 0 1	1 0 0 0 0	0
16	1 0 0 1 0	1 0 0 0 1	2
17	1 0 0 1 1	1 0 0 1 0	4
18	1 0 1 0 0	1 0 0 1 1	6
19	1 0 1 0 1	1 0 1 0 0	8
20	1 0 1 1 0	1 0 1 0 1	10
21	1 0 1 1 1	1 0 1 1 0	12
22	1 1 0 0 0	1 0 1 1 1	14
23	1 1 0 0 1	1 1 0 0 0	16
24	1 1 0 1 0	1 1 0 0 1	18
25	1 1 0 1 1	1 1 0 1 0	20
26	1 1 1 0 0	1 1 0 1 1	22
27	1 1 1 0 1	1 1 1 0 0	24
28	1 1 1 1 0	1 1 1 0 1	26
29	1 1 1 1 1	1 1 1 1 0	28

GSEN(2, 16, 5)			
i	$s_0s_1s_2s_3s_4s_5s_6$	$s'_0s'_1s'_2s'_3s'_4s'_5s'_6$	$v(i)$
0	0 0 0 0 0	0 0 0 0 0	0
1	0 0 0 0 1	0 0 0 0 1	0
2	0 0 0 1 0	0 0 0 1 0	0
3	0 0 0 1 1	0 0 0 1 1	0
4	0 0 1 0 0	0 0 1 0 0	0
5	0 0 1 0 1	0 0 1 0 1	0
6	0 0 1 1 0	0 0 1 1 0	0
7	0 0 1 1 1	0 0 1 1 1	0
8	0 1 0 0 0	0 1 0 0 0	0
9	0 1 0 0 1	0 1 0 0 1	0
10	0 1 0 1 0	0 1 0 1 0	0
11	0 1 0 1 1	0 1 0 1 1	0
12	0 1 1 0 0	0 1 1 0 0	0
13	0 1 1 0 1	0 1 1 0 1	0
14	0 1 1 1 0	0 1 1 1 0	0
15	0 1 1 1 1	0 1 1 1 1	0
16	1 0 0 0 0	1 0 0 0 0	0
17	1 0 0 0 1	1 0 0 0 1	0
18	1 0 0 1 0	1 0 0 1 0	0
19	1 0 0 1 1	1 0 0 1 1	0
20	1 0 1 0 0	1 0 1 0 0	0
21	1 0 1 0 1	1 0 1 0 1	0
22	1 0 1 1 0	1 0 1 1 0	0
23	1 0 1 1 1	1 0 1 1 1	0
24	1 1 0 0 0	1 1 0 0 0	0
25	1 1 0 0 1	1 1 0 0 1	0
26	1 1 0 1 0	1 1 0 1 0	0
27	1 1 0 1 1	1 1 0 1 1	0
28	1 1 1 0 0	1 1 1 0 0	0
29	1 1 1 0 1	1 1 1 0 1	0
30	1 1 1 1 0	1 1 1 1 0	0
31	1 1 1 1 1	1 1 1 1 1	0

References

[1] Z. Chen, Z. Liu, and Z. Qiu, "Bidirectional shuffle-exchange network and tag-based routing algorithm," *IEEE Communication Letters*, vol. 7, no. 3, Mar. 2003.

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

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

[4] C. P. Kuruskal. "A unified theory of interconnection network structure," *Theoretical Computer Science*, vol. 48, pp. 75-94 , Jun. 1986.

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

[6] S. C. Liew, "On the stability of shuffle-exchange and bidirectional shuffle-exchange deflection network," *IEEE/ACM Trans. Networking*, vol. 5, no. 1, pp. 87-94, Feb. 1997.

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

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