行政院國家科學委員會專題研究計畫成果報告

國科會專題研究計畫成果報告撰寫格式說明

Preparation of NSC Project Reports

計畫編號：NSC 93-2115-M-009-011
執行期限：93 年 8 月 1 日至 94 年 7 月 31 日

主持人：陳秋媛　國立交通大學應用數學系
cychen@mail.nctu.edu.tw
計畫參與人員：劉維展、藍國元、羅經凱、胡世謙
以上均為國立交通大學應用數學所研究生

一、中文摘要

本計劃之目的在於研究「雙環式網路」、「三環式網路」、及「連接網路」。本計劃為兩年期計劃，現為第一年，在這一年中，我們的研究重點放在「退化的雙環式網路」、「混合的弦環式網路」、以及「多級式連接網路」上。

本計劃截至目為止，已經完成三篇論文（請參看後面所附論文）。其中一篇是關於「雙環式網路」、兩篇是關於「多級式連接網路」。

在「雙環式網路」方面，我們探討當一個雙環式網路的 L-shape 是 degenerate 時，參數(l, h, p, n)的給法。在「多級式連接網路」方面，我們研究 buddy networks with an arbitrary number of stages 的等價關係、以及提出一個很有效率的 tag-based routing algorithm for the backward network of a bidirectional general shuffle-exchange network。

關鍵詞：環式網路、雙環式網路、三環式網路、連接網路、直徑

Abstract

The purpose of this project is to study loop networks and interconnection networks. This project is a two-year project and now it is in its first year. In this year, we focus on the study of degenerate double-loop networks, the equivalence relation of buddy networks, and the routing algorithm for the backward network of a bidirectional shuffle-exchange network.

Keywords: loop network, double-loop network, triple-loop network, interconnection network, diameter

二、緣由與目的

本人因旁聽黃光明教授之課程而對「環式網路」產生很大之興趣，在過去幾年中，所做的研究也以「環式網路」為主。然而，由於「連接網路」在現今的研究及實際應用中，都佔有極重要的地位，故想藉由此計劃，對「環式網路」中的一些問題再做研究，同時，也開始「連接網路」方面之研究。

三、結果與討論

本計劃在第一年內，截至目為止，已經完成了三篇論文：其中一篇論文已經投稿、另外兩篇論文已經全部寫完了，最近就會投稿；除此之外，還有幾篇論文正在撰寫中。總之，今年是豐收的一年。以下略述已完成的那三篇論文的結果。

論文一：Equivalence of buddy networks with arbitrary number of stages. [5]

多級式連接網路的等價關係是一個重要的研究問題，因為它能減少我們所需考慮的

網路個數。

在過去，學者們曾提出 buddy networks 及 strict buddy networks，也研究過 banyan 多級式連接網路的等價關係（網路中的 stage 數都有所限制）。在這篇論文裡，我們推廣 strict buddy networks 成為 universal buddy networks、也推廣 *P*(*,*) networks 成為 power-of-*d* networks。

由於 banyan 多級式連接網路是 universal buddy networks 的 special case，在這篇論文裡，我們研究 universal buddy networks with an arbitrary number of stages 的等價關係。

<u>論文二</u>：On degenerate double-loop L-shapes. [12]

論文[4]及論文[9]均曾提出一個「雙環式網路」的L-shape是degenerate case時，參數(*l*, *h*, *p*, *n*)的給法。然而兩篇論文所給之參數未必一致，<u>論文二</u>之目的即在討論兩者之關係。

我們首先得出了一個雙環式網路的 L-shape 是 degenerate 的充分必要條件；接著，我們證明了 L-shape 是 degenerate 時，只會有 7 種可能的 shapes： (S1), (S2), …, (S7)。

為方便，稱論文[9]的參數(*l*, *h*, *p*, *n*) 的給法為 CH-ALGO，稱論文[4]的給法為 CH-RULE。我們證明了：CH-ALGO 只會得出(S1), (S2), (S3), (S5)；CH-RULE 只會得出(S2), (S3), (S5), (S6)。我們也推導出：何時 CH-ALGO 和 CH-RULE 會得出一致的(*l*, *h*, *p*, *n*)，何時它們會得出不一致的(*l*, *h*, *p*, *n*)，以及當它們會得出不一致的(*l*, *h*, *p*, *n*)時、它們得出的(*l*, *h*, *p*, *n*)之間的關係。

<u>論文三</u>：An efficient tag-based routing algorithm for the backward network of a bidirectional general shuffle-exchange network. [8]

Shuffle-exchange network 是很常被用到的多級式連接網路。在論文[13]裡，Padmanbhan 提出了 general shuffle-exchange network (GSEN)，這是 shuffle-exchange network 的推廣，使得網路中的節點數不在受限為 *k* 的次方（假設 switch elements 都是 *k* × *k*），Padmanbhan 同時也提出了一個很有效率的 tag-based routing algorithm。在論文[7]裡，Chen、Liu 和 Qiu 又推廣 GSEN 成為所有的邊都是雙向的，並稱之為 bidirectional GSEN。

一個 bidirectional GSEN 裡包含了兩個網路：the forward network 及 the backward network。The forward network 的 routing 可以利用 Padmanbhan 所提出的 tag-based routing algorithm 來完成。至於 the backward network，Chen、Liu 和 Qiu 提出了一個 tag-based routing algorithm；這個 algorithm 必須先執行 Padmanbhan 的 tag-based routing algorithm，並「逆向」使用 Padmanbhan 的 algorithm 所產生的 tag。

在論文[8]裡，我們證明了 the backward network of a bidirectional GSEN 有一個非常好的性質是：對每個 destination *i* 而言，會有兩個 backward control tags 伴隨著它，任何 source *j* 都可利用這兩個 tags 中的某一個走到 *i*。我們利用這個性質得出 efficient routing algorithms。

四、計劃成果自評

本計劃之執行成果與預期成果相符，目前已經完成了三篇論文。

五、參考文獻

[1] B. W. Arden and H. Lee, Analysis of chordal ring network, IEEE Trans. Computer. 30 (1981) 291-295.

[2] L. Barriere, J. F\'abrega, E. Simo and M. Zaragora, Fault-tolerant routing in chordal ring networks, Netoworks 36 (2000) 180-190

[3] J.-C. Bermond, F. Comellas and D. F. Hsu, Distributed loop computer networks: a survey, *J. Para. Cist. Comput.* 24 (1995) 2-10.

[4] *C. Y. Chen and F. K. Hwang, Equivalent L-shapes of double-loop networks for the degenerate case, *Journal of Interconnection Networks* 1 (2000) 47-60.

[5] *C. Y. Chen, F. K. Hwang and K. Y. Lan, Equivalence of buddy networks with arbitrary number of stages, submitted (2004).

[6] S. K. Chen, F. K. Hwang and Y. C. Liu, Some combinatorial properties of mixed chordal rings," J. Inter. Networks 4 (2003) 3-16.

[7] *Z. Chen, Z. Liu, and Z. Qiu, Bidirectional shuffle-exchange network and tag-based routing algorithm, *IEEE Communication Letters* 7 (2003), 121-123.

[8] *C. Y. Chen and J. K. Luo, An efficient tag-based routing algorithm for the backward network of a bidirectional general shuffle-exchange network, preprint (2004).

[9] *Y. Cheng and F. K. Hwang, Diameters of weighted double loop networks, *J. Algorithms* 9 (1988), 401-410.

[10] F. K. Hwang, A complementary survey on double-loop networks, *Theoret. Comput. Sci*. 263 (2001), 211-229.

[11] W. Kabacinski and G. Danilewicz, Wide-sense and strict-sense nonblocking operation of multicast multi-$\log_2 N$ switching networks, *IEEE Trans. Commu.* 50 (2002) 1025-1036.

[12] * J. S. Lee, C. Y. Chen and K. Y. Lan, On degenerate double-loop L-shapes, preprint (2004).

[13] *K. Padmanabham, Design and analysis of even-sized binary shuffle-exchange networks for multiprocessors, *IEEE Trans. Parallel and Distributed Systems* 2 (1991), 385-397.

[14] C. S. Raghavendra and J. A. Sylvester, A survey of multi-connected loop topologies for local computer networks, *Comput. Netw. ISDN Syst.* 11 (1986) 29-42.

[15] Y. Tscha and K. H. Lee, Yet another result on multi-log N networks, *IEEE Trans. Commu*. 47 (1999) 1425–1431.

Equivalence of Buddy Networks with Arbitrary Number of Stages[*]

Chiuyuan Chen[†], Frank K. Hwang[‡] and Kuo-Yuan Lan

Department of Applied Mathematics
National Chiao Tung University
Hsinchu 300, Taiwan

Abstract

Equivalence of multistage interconnection networks is an important concept since it reduces the number of networks to be studied. Equivalence among the banyan networks has been well studied. Occasionally, the study was extended to networks obtained by concatenating two banyan networks (identifying the output stage of the preceding network with the input stage of the succeeding one). Recently, equivalence among the class of networks which are obtained from banyan networks by adding extra stages has also been studied. Note that all these above-mentioned networks are in the general class of buddy networks. In this paper we study equivalence of buddy networks with an arbitrary number of stages.

Keywords: Multistage interconnection networks, topological equivalence, banyan property, buddy property, bit permutation.

1 Introduction

Let $N = d^n$ be the number of inputs and outputs of a network. A *d-nary s-stage network* is a network with s columns (stages) where each column consists of N/d $d \times d$ crossbars (switches) such that links exist only between crossbars of adjacent stages (note that we do not allow multi-links between crossbars). An n-stage network is a *banyan network* if each input has a unique path to each output (see Figure 1). If a network has more than

[*]This research was partially supported by the National Science Council of the Republic of China under the grants NSC93-2115-M-009-011 and NSC93-2115-M-009-013.

[†]The corresponding author, e-mail: cychen@mail.nctu.edu.tw

[‡]This paper was written when the author was visiting Center of Mathematical Sciences, Zhejiang University, Hangzhou, Zhejiang, P.R. China.

n stages, then we say such a network has *extra stages.* In all the figures, the arcs are directed from left to right.

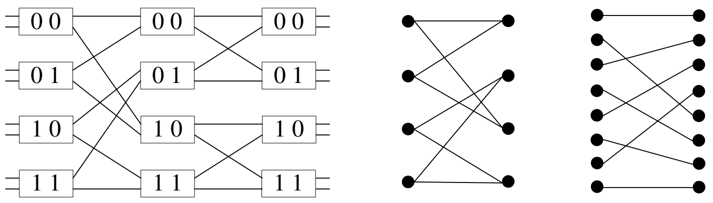


Figure 1: A binary 3-stage banyan network (the Baseline network), G_1, and $\mathcal{G}_1$.

We can associate an s-stage network with a directed graph G in which vertices represent crossbars and arcs the communication links. Throughout this paper, $G_{i,j}$ denotes the subgraph of G induced by the vertices from stage i to stage j. When there is no confusion, $G_{i,j}$ also denotes the subnetwork from stage i to stage j. Set $G_i = G_{i,i+1}$ for easy writing (see Figure 1).

Two s-stage networks are *topologically equivalent* (or simply *equivalent*) if their associated directed graphs are isomorphic. In other words, two s-stage networks are equivalent if one can be obtained from the other by permuting crossbars in the same stage. Note that equivalence in this sense preserves the connecting properties of the network. Hence once we prove a nonblocking property for a network, it extends to all equivalent networks.

Parker [10] first established the equivalence of several n-stage banyan networks including the Baseline network. Wu and Feng [13] expanded the equivalence class. Dais and Jump [6] introduced the "buddy" notation: Let v and v' be two crossbars in stage i and let V_v and $V_{v'}$ be two sets of crossbars in stage j that v and v' can reach, respectively. Then the network is a *buddy network* if for any i and $j = i + 1$, either $V_v = V_{v'}$ or $V_v \cap V_{v'} = \emptyset$. Agrawal [1] called a buddy network a *strict buddy network* if the buddy condition also holds for $j = i + 2$. In this paper, we further generalize the strict buddy network to the *universal buddy network* by allowing j to be arbitrary. In [1], Agrawal claimed that the strict buddy property characterizes the Baseline-equivalent networks. Bermond, Fourneau and Jean-Marie [2, 3] gave a counterexample to Agrawal's claim. Instead, they defined the $P(*,*)$ property for characterization: A network is a $P(*,*)$ *network* if for any two stages $i \leq j$, the number of components in the subgraph $G_{i,j}$ is $d^{n-1-(j-i)}$.

Siegel and Smith [12] proposed an extra stage to the Baseline-equivalent class of networks, and Shyy-Lea [11] considered the k-extra-stage version. Hwang, Liao and Yeh (see [8]) pointed out that the extra stage versions of Baseline-equivalent networks are not necessarily equivalent. Equivalence depends not only on the base network (Baseline or others), but also on how the extra stages are added. Previously, equivalence of extra-stage networks has been studied only for the double-concatenation type [4, 7] since it contains the famous Beneš network as a special case.

To study the equivalence of extra-stage networks for arbitrary number of stages, Chang, Hwang and Tong [5] proposed the class of bit permutation networks. Label the crossbars in a stage by distinct d-nary $(n-1)$-sequences $x_1x_2\cdots x_{n-1}$. A *bit-i group* (or simply an *i-group*) consists of the d crossbars whose labels differ only in bit i (there are d^{n-2} bit-i groups). An s-stage network is a *bit permutation network* if for every G_i, $1 \leq i \leq s-1$, the links always go from bit-u_i groups G' of stage i to bit-v_{i+1} groups G'' of stage $i+1$ for some u_i, v_{i+1}, where G'' is a permutation of G'. (A detailed definition of bit permutation networks is in Section 2.) They proved that a bit permutation network is equivalent to one whose G_i has the property that $v_{i+1} = u_i$ for all i. Such a network can be characterized by the vector $(u_1, u_2, \cdots, u_{s-1})$.

Recently, Li [9] proposed the bit permuting network. He view the outputs of stage i and the inputs of stage $i+1$ as the vertices of a bipartite graph $\mathcal{G}_i$ and label the outputs of stage i (inputs of stage $i+1$) by distinct d-nary n-sequences; see Figure 1. Then $\mathcal{G}_i$ gives a bijection from the d^n outputs to the d^n inputs and hence can be treated as a permutation. Such a permutation is called an *bit permutation* if it can be characterized by a permutation σ_i of the n bits. A network is a *bit permuting network* if each $\mathcal{G}_i$ corresponds to a σ_i. Li gave an elegant "guide" algorithm to route any n-stage bit permuting network.

The notions of universal buddy (UB), bit permutation (BP) and bit permuting (BPT) are applicable to networks with any number of stages. Since $P(*,*)$ is defined only for

n-stage networks, we generalize it to the power-of-d networks. An s-stage network is a *power-of-d network* if for any i, j, $1 \leq i \leq j \leq s$, the number of components in $G_{i,j}$ is a power of d. An s-stage network is a *power-of-d universal buddy network* if it is both power-of-d and universal buddy. The notion of power-of-d (d^P) and power-of-d universal buddy (d^PUB) are applicable to networks with any number of stages. In this paper, the notations of UB, BP, BPT, d^P and d^PUB also denote their corresponding classes of networks.

Let $A \supset B$ denote A properly contains B. Let $A = B$ denote A *is equal to* B, meaning any network in class A is a network in class B (no permutation of crossbars allowed), and vice versa. Let $A \sim B$ denote A *is equivalent to* B, meaning any network in class A is topologically equivalent to a network in class B (permutations of crossbars allowed), and vice versa. Note that the permutation of crossbars is neither unique nor one-to-one. Hence $A \sim B$ does not imply $|A| = |B|$. In particular, $A \supset B$ does not preclude $A \sim B$. In this paper, we will establish:

$$\begin{array}{l} UB \supset \\ d^P \ \supset \end{array} d^PUB \supset BP = BPT. \tag{1.1}$$

$$d^PUB \sim BP, \text{ but } UB \nsim d^P,\ UB \nsim d^PUB \text{ and } d^P \nsim d^PUB. \tag{1.2}$$

Since the BP network has the vector characterization and is defined for any number of stages, it is of interests to know whether this very useful class can be further extended with all connecting properties preserved. (1.1) shows that d^PUB generalizes BP and (1.2) shows that they are equivalent.

2 The BP and BPT classes

We now give a detailed definition of BP networks; this definition is from [5]. An s-stage network is a *bit permutation network* if for every G_i, $1 \leq i \leq s-1$, there exists a permutation ρ_i on $\{1, 2, \cdots, n\}$ such that $\rho_i(n) \neq n$ and each crossbar $x_1x_2\cdots x_{n-1}$ is adjacent to crossbar $x_{\rho_i(1)}x_{\rho_i(2)}\cdots x_{\rho_i(n-1)}$, where $x_n \in \{0, 1, \cdots, d-1\}$. Note that x_n has d values and and whenever it appears in the coordinates, d sequences are generated by running x_n through the set $\{0, 1, \cdots, d-1\}$. For example, the network in Figure 1 is a bit permutation network with $\rho_1 = (132)$ and $\rho_2 = (23)$. Since $\rho_1 = (132)$, $x_1x_2x_3$ is mapped to $x_3x_1x_2$ and the links go from bit-2 groups of stage 1 to bit-1 groups of stage 2. In particular, crossbars 00 and 01 at stage 1 are adjacent to crossbars 00 and 10 at stage 2. Since $\rho_2 = (23)$, $x_1x_2x_3$ is mapped to $x_1x_3x_2$ and the links go from bit-2 groups of stage 2 to bit-2 groups of stage 3. Thus crossbars 00 and 01 at stage 2 are adjacent to crossbars 00 and 01 at stage 3.

The stages in Figure 2 and Figure 3 are drawn horizontally to save space. These two figures are the same (they have the same connections between crossbars) except their labels. The labels in Figure 2 are outputs of stage i and inputs of stage $i+1$. The labels in Figure 3 are crossbars of stage i and crossbars of stage $i+1$. The permutation in Figure 2 illustrates a bit permutation $\sigma_i = (1234)$ in $\mathcal{G}_i$, while the permutation in Figure 3 illustrates a permutation $\rho_i = (1234)$ in G_i.

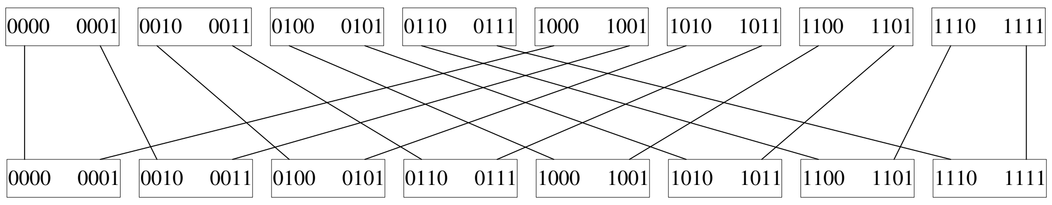


Figure 2: A bit permutation σ_i in $\mathcal{G}_i$.

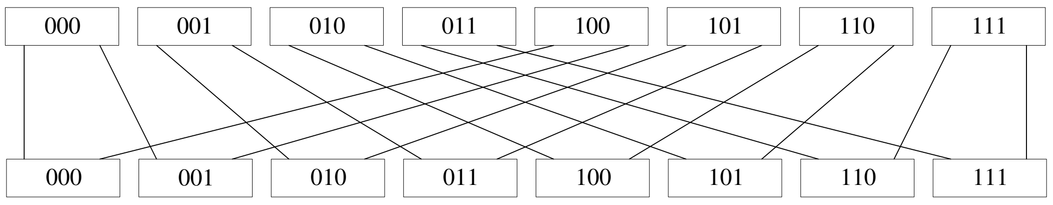


Figure 3: A permutation ρ_i of crossbars in G_i.

We now prove

Theorem 1 *$BPT = BP$.*

Proof. First consider a *BPT* network. For every $\mathcal{G}_i$, there exists a bit permutation σ_i on $\{1, 2, \cdots, n\}$ such that each output $x_1x_2\cdots x_n$ of stage i is adjacent to input $x_{\sigma_i(1)}x_{\sigma_i(2)}\cdots x_{\sigma_i(n)}$ of stage $i+1$. Note that the label of a crossbar of stage i $(i+1)$ can be obtained from the labels of its d outputs (inputs) by dropping the last bit. Thus crossbar $x_1x_2\cdots x_{n-1}$ is adjacent to crossbar $x_{\sigma_i(1)}x_{\sigma_i(2)}\cdots x_{\sigma_i(n-1)}$. Note that $\sigma_i(n) \neq n$; otherwise, there are multi-links between crossbar $x_1x_2\cdots x_{n-1}$ and crossbar $x_{\sigma_i(1)}x_{\sigma_i(2)}\cdots x_{\sigma_i(n-1)}$. Since $\sigma_i(n) \neq n$, crossbar $x_1x_2\cdots x_{n-1}$ is adjacent to crossbar $x_{\sigma_i(1)}x_{\sigma_i(2)}\cdots x_{\sigma_i(n-1)}$, where $x_n \in \{0, 1, \cdots, d-1\}$. Thus a *BPT* network is a *BP* network. On the other hand, consider a *BP* network. For every G_i, there exists a permutation ρ_i on $\{1, 2, \cdots, n\}$ such that $\rho_i(n) \neq n$ and each crossbar $x_1x_2\cdots x_{n-1}$ of stage i is adjacent to crossbar $x_{\rho_i(1)}x_{\rho_i(2)}\cdots x_{\rho_i(n-1)}$ of stage $i+1$, where $x_n \in \{0, 1, \cdots, d-1\}$. Thus each output $x_1x_2\cdots x_n$ of stage i is adjacent to input $x_{\rho_i(1)}x_{\rho_i(2)}\cdots x_{\rho_i(n)}$ of stage $i+1$. Since a permutation on $\{1, 2, \cdots, n\}$ is a bit permutation, a *BP* network is a *BPT* network. Theorem 1 now follows. ∎

We now show that a bit permutation σ_i of $\mathcal{G}_i$ defines a mapping from u-groups of stage i to v-groups of stage $i+1$. In fact, we can pinpoint u and v.

Lemma 2 *Suppose $\mathcal{G}_i$ is represented by the bit permutation σ_i. Then $\mathcal{G}_i$ induces a mapping from $\sigma_i(n)$-groups of stage i to $\sigma_i^{-1}(n)$-groups of stage $i+1$.*

Proof. Note that each output $x_1x_2\cdots x_n$ of stage i is adjacent to input $x_{\sigma_i(1)}x_{\sigma_i(2)}\cdots x_{\sigma_i(n)}$ of stage $i+1$. The label of a crossbar of stage i $(i+1)$ can be obtained from the labels of its d outputs (inputs) by dropping the last bit. Since $x_{\sigma_i(n)}$ is the last bit and get dropped in the crossbar label of stage $i+1$, the d stage-i crossbars differing only in bit $\sigma_i(n)$, i.e., the $\sigma_i(n)$-group, are mapped to the same set of stage-$(i+1)$ crossbars. On the other hand, the stage-i crossbar containing d outputs whose labels differ only in bit $\sigma_i(n)$ is mapped to the $\sigma_i^{-1}(n)$-group of stage $i+1$. Lemma 2 is proved. ∎

For the example in Figure 2, the mapping is from $(\sigma_i(4) = 1)$-groups of stage i to

$(\sigma_i^{-1}(4) = 3)$-groups of stage $i+1$. We now give a vector characterization of a *BPT* network. First a lemma.

Lemma 3 *Suppose $\mathcal{G}_i$ corresponds to a bit permutation σ_i which maps $\sigma_i(n)$-groups of stage i to $\sigma_i^{-1}(n)$-groups of stage $i+1$ and suppose $\mathcal{G}_{i+1}$ corresponds to a bit permutation σ_{i+1}. Suppose we permute the crossbars of stage $i+1$ such that the j-th crossbars of the $\sigma_i^{-1}(n)$-groups are lined up with the j-th crossbars of the $\sigma_i(n)$-groups, $j = 0, 1, \cdots, d-1$. Then after the lining-up operation, $\mathcal{G}_i$ corresponds to the bit permutation $(u_i\ n)$ and $\mathcal{G}_{i+1}$ corresponds to the bit permutation $(\sigma_{i+1}^{-1}(\sigma_i^{-1}(n))\ \ \sigma_{i+1}^{-1}(\sigma_i(n))) \circ \sigma_{i+1}$.*

Proof. Take a $\sigma_i^{-1}(n)$-group of stage $i+1$. The j-th crossbar in this group is mapped (lined up) to the j-th crossbar of the corresponding $\sigma_i(n)$-group of stage i under this lining-up operation; see Figure 4. Then the only difference is that before lining up, the bit permutation σ_i maps $\sigma_i(n)$-groups to $\sigma_i^{-1}(n)$-groups, while after lining up, the mapping is from u_i-groups to u_i-groups. Note that the mapping from u_i-groups to u_i-groups corresponds to the bit permutation $(u_i\ n)$. After lining up, $\sigma_i^{-1}(n)$-groups of stage $i+1$ become $\sigma_i(n)$-groups. Since σ_{i+1} maps $\sigma_{i+1}^{-1}(\sigma_i^{-1}(n))$ to $\sigma_i^{-1}(n)$ and $\sigma_{i+1}^{-1}(\sigma_i(n))$ to $\sigma_i(n)$, swapping bit $\sigma_i^{-1}(n)$ with bit $\sigma_i(n)$ corresponds to applying $(\sigma_{i+1}^{-1}(\sigma_i^{-1}(n))\ \ \sigma_{i+1}^{-1}(\sigma_i(n)))$ on σ_{i+1}. Thus after lining up, $\mathcal{G}_{i+1}$ corresponds to bit permutation $(\sigma_{i+1}^{-1}(\sigma_i^{-1}(n))\ \ \sigma_{i+1}^{-1}(\sigma_i(n))) \circ \sigma_{i+1}$. ∎

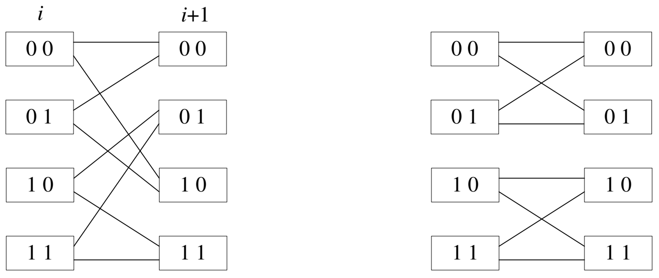


Figure 4: Lining up stage-$(i+1)$ crossbars.

By Lemma 2, we know that in every $\mathcal{G}_i$ of a *BPT* network, the links go from u_i-groups to v_{i+1}-groups for some u_i, v_{i+1}. The lining-up operation enables us to permute the crossbars of stage $i+1$ so that the links go from u_i-groups to u_i-groups. For example, in Figure 4, the links go from 2-groups to 1-groups. After lining up the stage-$(i+1)$ crossbars, the links go from 2-groups to 2-groups.

Theorem 4 *Consider an s-stage BPT network. By permuting the crossbars of stage* 2, 3, $\cdots$, *s, each $\mathcal{G}_i$ corresponds to a bit permutation which maps u'_i-groups to u'_i-groups, $i = 1, 2, \cdots, s-1$.*

Proof. We prove this theorem by induction on s. This theorem is trivially true for $s = 2$ since we can permute the crossbars of stage 2 to line up with their mates in stage 1. Then $\mathcal{G}_1$ corresponds to a bit permutation which maps u_1-groups to u_1-groups. Suppose this theorem holds for up to $s-1$ stages. We now prove for s stages. Again, permute the crossbars of stage 2 to line up with their mates in stage 1. By Lemma 3, $\mathcal{G}_2$ remains to correspond to a bit permutation. Thus we may apply induction on this (s -1)-stage BPT network such that $\mathcal{G}_i$ is characterized by a bit permutation which maps u'_i-groups to u'_i-groups, $i = 1, 2, \cdots, s-1$. ∎

Since $BPT = BP$, the above characterization is also a vector characterization of a BP network, but our proof is simpler than the original proof in [5]. Recall that an s-stage network is a BP network if for every G_i, the links always go from u_i groups G' of stage i to v_{i+1} groups G'' of stage $i+1$ for some u_i, v_{i+1}, where G'' is a permutation of G'. If we drop the requirement that G'' is a permutation of G', then the lining-up operation would not yield a vector characterization. See Figure 5 as an example. In Figure 5(a), the links in G_1 go from 1-groups to 2-groups and the links in G_2 go from 1-groups to 3-groups. In Figure 5(b), the links in G_1 go from 1-groups to 1-groups, but the links in G_2 do not go from u-groups to u-groups for any u.

3 The d^PUB class

We now show that neither $d^P \subseteq UB$ nor vice versa; hence the definition of d^PUB makes sense. Figure 6(a) shows a 2^P network which is not a UB network since C reaches $\{C'$,

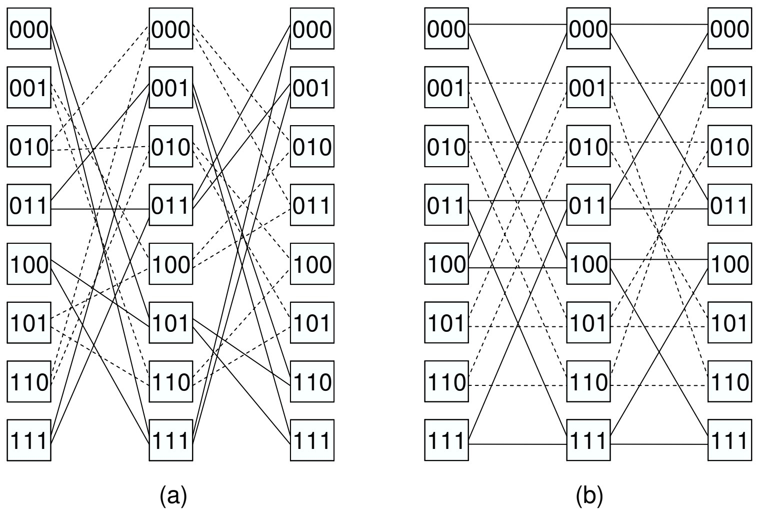


Figure 5: (a) Before lining up and (b) after lining up.

D', F', $G'\}$ and E reaches $\{C', E', F', H'\}$; the two sets intersect but are not identical. Figure 6(b) shows a UB network which is not a 2^P network since $G_{1,3}$ has 3 components.

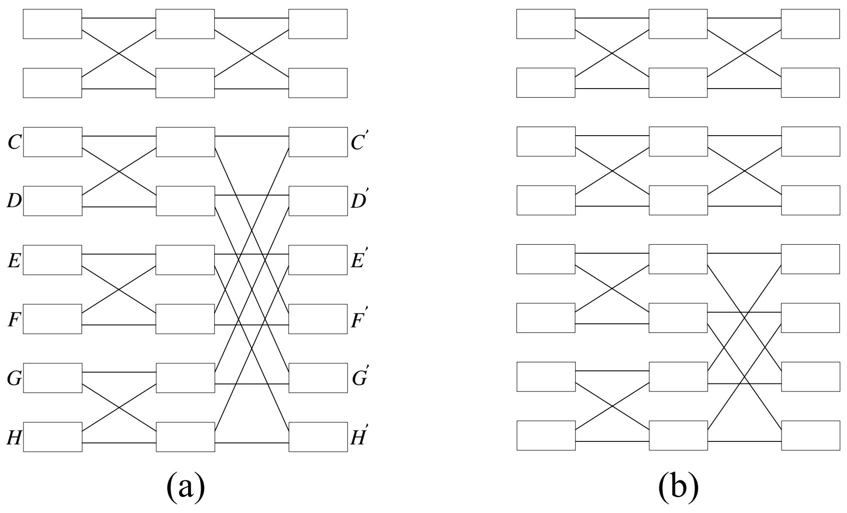


Figure 6: (a) A 2^P network and (b) a UB network.

We first quote a result of [5].

Theorem 5 *Suppose an s-stage d-nary BP network has d^n inputs, d^n outputs, and is characterized by the vector $(u_1, u_2, \cdots, u_{s-1})$ which contains k distinct elements. Then the network has d^{n-1-k} components.*

Corollary 6 *$BP \subseteq d^P$.*

Proof. It is not difficult to see that every subnetwork $G_{i,j}$ of a BP network is still a BP network. By Theorem 5, the number of components in $G_{i,j}$ is a power of d. Since i, j are arbitrary, the network is in d^P. ∎

Theorem 7 *$BP \subseteq UB$.*

Proof. Consider an s-stage BP network characterized by $(u_1, u_2, \cdots, u_{s-1})$. Let v be a crossbar in stage i which reaches a set $V_j(v)$ of crossbars in stage j. Then $V_j(v)$ consists of crossbars whose labels are the same in bits in the set $I = \{1, 2, \cdots, n-1\} \setminus \{u_i, u_{i+1}, \cdots, u_{j-1}\}$. Let v' be another crossbar in stage i. If v' differs from v in a bit in I, then clearly, $V_j(v') \cap V_j(v) = \emptyset$; if not, then $V_j(v') = V_j(v)$. Since i, j, v, v' are arbitrary, the network is in UB. ∎

Theorem 8 *$BP \subset d^P UB$.*

Proof. That $BP \subseteq d^P UB$ follows from Corollary 6 and Theorem 7. That the containment is strict follows from Figure 7 (crossbars 00 and 11 in stage 2 are connected to crossbars 00 and 11 in stage 3; so, the links from stage 2 to stage 3 do not go from u_i-groups to v_{i+1}-groups for any u_i, v_{i+1}). ∎

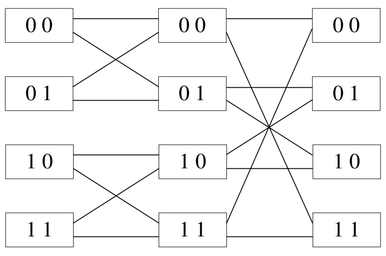


Figure 7: A $d^P UB$ network which is not a BP network.

Theorem 9 *$d^P UB \sim BP$.*

Proof. Since $d^PUB \supset BPT$, it suffices to prove that a d^PUB network is equivalent to a BP network. We prove this by induction on the number s of stages.

(1) $s = 2$. Suppose v of stage 1 is connected to the set $V_2(v)$. Let v' be another crossbar in stage 1 and connected to a given $w \in V_2(v)$. By the UB property, $V_2(v') = V_2(v)$. Since there are $d - 1$ choices of v' from w, these v' together with v form a $d \times d$ complete bipartite graph $K_{d,d}$ with $V_2(v)$. Further, $V_2(v'') \cap V_2(v) = \emptyset$ for any $v'' \notin v \cup \{v'\}$. Since v is arbitrary, $G_{1,2}$ consists of d^{n-2} $K_{d,d}$ whose equivalence to a BP network is clear.

(2) $s = 3$. By the d^P property, the network has d^{n-k} components for some $1 \leq k \leq n$. Recall that from (1) the subgraphs $G_{1,2}$ and $G_{2,3}$ must each consist of d^{n-2} $K_{d,d}$. Hence $k = 1$ is impossible.

For $k = 2$, then no two $K_{d,d}$ in $G_{1,2}$ can be connected through $G_{2,3}$. Therefore $G_{1,3}$ must consist of d^{n-2} copies of concatenation of two $K_{d,d}$, with the outputs of the former identified with the inputs of the latter (see Figure 8). Clearly, subnetwork $G_{1,3}$ is equivalent to a BP network.

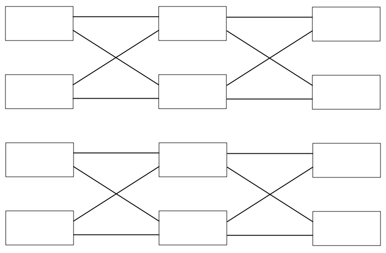


Figure 8: Concatenation of $K_{2,2}$.

For $k = 3$, first suppose $G_{1,3}$ is obtained by connecting each d-set $D = \{D_1, D_2, \cdots, D_d\}$, where each D_i is a $K_{d,d}$ in $G_{1,2}$, into one component in $G_{1,3}$. Note that the connection is done by a d-set $D' = \{D'_1, D'_2, \cdots, D'_d\}$ of $K_{d,d}$ in $G_{2,3}$. If two crossbars of the same D_i are connected to a D'_j, then one member of $D \setminus D_i$ will not be connected to D'_j, violating the UB property. Therefore, the d crossbars in a D_i must go to distinct D'_j, or all D'_j. Since we can permute the stage-2 crossbars in a D arbitrarily, and independently for each D, the stage-2 crossbars in each D can be ordered such that the k-th one goes to the k-th D', which is clearly a BP network. Figure 9 illustrates how to permute.

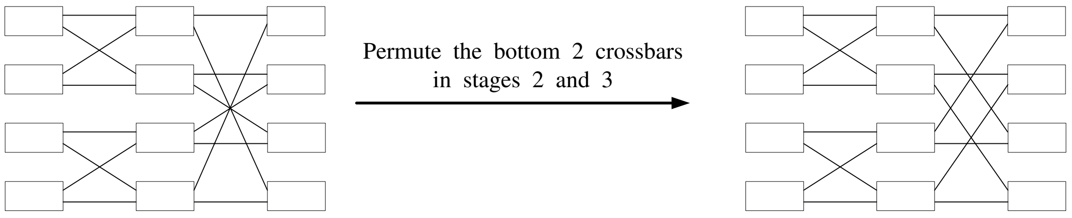


Figure 9: A permutation to achieve BP.

Suppose $G_{1,3}$ is obtained otherwise. There must exist a d'-set of $K_{d,d}$, $d' > d$, in $G_{1,2}$ connected in $G_{2,3}$ through a d'-set of $K_{d,d}$ in $G_{2,3}$. Note that an input in this component touches only d^2 among the dd' outputs. Hence there must exist another input reaching some, but not all, of these d^2 outputs, violating the UB property.

For $k \geq 4$, then the situation described in the last paragraph must also happen.

(3) $s \geq 4$. Consider the two subnetworks $G_{1,3}$ and $G_{2,s}$. By induction, $G_{1,3}$ can be represented by a vector (u_1, u_2) and $G_{2,s}$ by $(u'_1, u'_2, \cdots, u'_{s-2})$. By Lemma 3, we can permute the crossbars in stage k, $2 \leq k \leq s$, such that $u'_1 = u_2$ and $u''_k = u'_{k-1}$ for $3 \leq k \leq s-1$. Therefore the subnetwork $G_{1,s}$ is represented by the vector $(u_1, u_2, u''_3, \cdots, u''_{s-1})$, i.e., $G_{1,s}$ is a BP network.

∎

Corollary 10 *Two $d^P UB$ networks are equivalent if the characterization vector of one can be obtained from the other through a permutation.*

Figure 6(a) gives an example of a d^P network which is not equivalent to a UB network. Hence $d^P \nsim UB$. Since $UB \supset BP$, Figure 6(a) is also an example of a d^P network which is not equivalent to a BP network. Therefore the UB condition can not be dropped from Theorem 9. Since $BP \sim d^P UB$, it follows that $d^P \nsim d^P UB$. Figure 6(b) gives a UB (or strict buddy) network which is equivalent to neither a d^P nor a BP network. Hence $UB \nsim BP$. Since $BP \sim d^P UB$, it follows that $UB \nsim d^P UB$.

4 Conclusions

We established the containment relation as given in (1.1), and the equivalence relation as given in (1.2). By so doing, we achieve three desirable generalizations:

(1) We make the logical extension of the buddy network and the strict buddy network to the universal buddy network; a network with more structure but still includes all banyan-type networks and their extra-stage versions.

(2) We generalize the notion of BP to $d^P UB$ which is a larger class, yet preserves all connecting properties of BP.

(3) We generalize $P(*,*)$ which is defined only for $n = \log_d N$ stages to general s stages.

The equivalence relations we established also help in simplifying some existing proofs:

(1) The proof of vector characterization of BP in [5] is quite complicated. We gave a simple proof of vector characterization of BPT and the equality that $BPT = BP$ makes the proof valid for BP too.

(2) The proof that $P(*,*)$ characterizes the Baseline-equivalent class of banyan-type networks is very long, as admitted in [2]. Our proofs of Theorem 9 and Corollary 10 are much shorter and more general.

Acknowledgement. The author wishes to thank H. Zhou for providing a counterexample to a conjecture prior to our discovery of Theorem 9. We also thank the comments of referees which led to a better version of the paper.

References

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

[2] J.C. Bermond, J.M. Fourneau and A. Jean-Marie, Equialence of multistage interconnection networks, Inform. Proc. Lett. 26 (1987) 45-50.

[3] J.C. Bermond, J.M. Fourneau and A. Jean-Marie, A graph theoretical approach to equivalence of multistage interconnection networks, Disc. Appl. Math. 22 (1988/89) 201-217.

[4] T. Calamoneri and A. Massini, Efficient algorithm for checking the equivalence of multistage interconnection networks, J. Parallel Distrib. Comput. 64 (2004) 135-150.

[5] G.J. Chang, F.K. Hwang and L.D. Tong, Characterizing bit permutation networks, Networks 33 (1999) 261-267.

[6] D.M. Dias and J.R. Jump, Analysis and simulation of buffered delta networks, IEEE Trans. Comput. C-30 (1981) 273-282.

[7] Q. Hu, X. Shen and J. Yang, Topologies of combined $(2\log N-1)$-stage interconnection networks, IEEE Trans. Comput. 46 (1997) 118-124.

[8] F.K. Hwang, The Mathematical Theory of Nonblocking Switching Networks, World Scientific, Singapore, 1998.

[9] S.-Y.R. Li, Algebraic Switching Theory and Broadband Applications, Academic, New York, 2001.

[10] D.S. Parker, Notes on shuffle-exchange type of networks, IEEE Trans. Comput. 29 (1980) 213-222.

[11] D.J. Shyy and C.T. Lea, $Log_2(N,m,p)$ strictly nonblocking networks, IEEE Trans. Commun. 39 (1991) 1502-1510.

[12] H.J. Siegel and S.D. Smith, Study of multistage SIMD interconnection networks, Proc. 5-th Ann. Symp. Comput. Arch., 1978, 223-229.

[13] C. Wu and T. Feng, On a class of multistage interconnection networks, IEEE Trans. Comput. 29 (1980) 694-702.

On Degenerate Double-Loop L-Shapes[*]

J. S. Lee

Department of Mathematics, National Kaohshiung Normal University

Kaoshiung 802, Taiwan

James K. Lan and Jenny C. Chen[†]

Department of Applied Mathematics, National Chiao Tung University

Hsinchu 300, Taiwan

Abstract

Most of the results about the L-shapes of double-loop networks are given in terms of the four parameters ℓ, h, p, n. But these parameters are not well defined in the degenerate case. Recently, Cheng and Hwang gave an efficient algorithm to compute the four parameters ℓ, h, p, n of an L-shape which works for both the regular and the degenerate cases. On the other hand, Chen and Hwang gave a set of rules to determine the four parameters of a degenerate L-shape. Unfortunately, the solutions given by the above two methods do not always coincide. In this paper, we try to understand their respective meanings and their relations.

Keywords: Double-loop network, L-shape, degenerate.

1 Introduction

The double-loop network has been well studied (see [6] for a recent survey) as the topology for a communication network or computer network. For example, SONET (synchronous optical network) is a double-loop network. Formally, a double-loop network $DL(N; a, b)$ has N nodes $0, 1, \cdots, N-1$ and $2N$ links, $i \rightarrow i+a,\ i \rightarrow i+b \pmod{N},\ i = 0, 1, \cdots, N-$

[*]This research was partially supported by the National Science Council of the Republic of China under the grant NSC93-2115-M-009-011.

[†]e-mail: cychen@mail.nctu.edu.tw

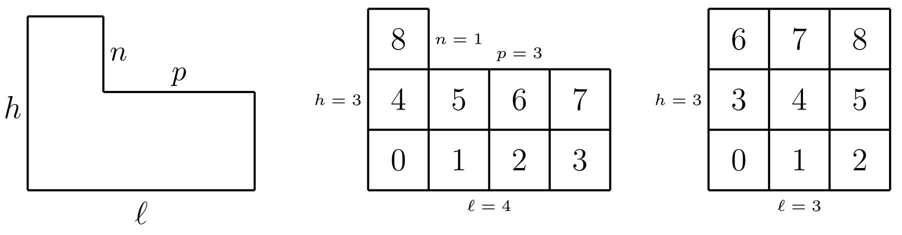
1. We assume that the weight of each of the $2N$ links is 1 and assume that $\gcd(N, a, b) = 1$ so that the network is strongly connected.

The minimum distance diagram (MDD) of $DL(N; a, b)$ is a diagram with node 0 in cell $(0, 0)$, and node v in cell (i, j) if and only if $ia + jb \equiv v \pmod{N}$ and $i + j$ is the minimum among all (i', j') satisfying the congruence. Namely, a shortest path from 0 to v is through taking i a-links and j b-links (in any order). Note that in a cell (i, j), i is the column index and j is the row index. An MDD includes every node exactly once (in case of two shortest paths, the convention is to choose the cell with the smaller row index, i.e., the smaller j). Since $DL(N; a, b)$ is clearly node-symmetric, there is no loss of generality in assuming: node 0 is the origin of a path.

Wong and Coppersmith (WC) [8] proved that the MDD of $DL(N; a, b)$ (their proof for $DL(N; 1, h)$ is easily extended to the general case) is always an L-shape which can be characterized by four parameters ℓ, h, p, n (see Fig. 1 (a)). These four parameters are the lengths of four of the six segments on the boundary of the L-shape. Clearly,

$$N = \ell h - pn.$$

In [2], Chen and Hwang showed that necessarily $\ell > n$ and $h \geq p$. Fig. 1 (b) illustrates an MDD with a regular L-shape. Fig. 1 (c) illustrates one with an L-shape degenerate into a rectangle.



(a) The four parameters (b) $a = 1$, $b = 4$ (c) $a = 1$, $b = 3$

Figure 1: Minimum distance diagrams and L-shapes.

Most of the results about the L-shape are given in terms of the four parameters ℓ, h, p, n. But these parameters are not well defined in the degenerate case. Recently, Cheng and Hwang [4] gave an $O(\log N)$-time algorithm to compute the four parameters ℓ, h, p, n of an L-shape which works for both the regular and the degenerate cases. On the other hand, Chen and Hwang [3] gave a set of rules to determine the four parameters of a degenerate L-shape. Unfortunately, the solutions given by the above two methods do not always coincide. In this paper, we try to understand their respective meanings and their relations. Since it is also of interest to know when will an L-shape degenerate, in this paper we give necessary and sufficient conditions depending on N, a, and b only.

2 Necessary and sufficient conditions for degenerate L-shapes

The following five notations will be used throughout this paper:

$$d = \gcd(N,a),\ d' = \gcd(N,b),\ N' = N/d,\ a' = a/d,\ \text{and } b' = b \pmod{N'}. \qquad (2.1)$$

Since $\gcd(N,a,b) = 1$, clearly $\gcd(d,d') = 1$. Chen and Hwang [3] proved

Lemma 1 [3] *A degenerate L-shape of height h and width ℓ satisfies one of the following three conditions:*

(1) $hb \not\equiv \ell a \equiv 0 \pmod N$.

(2) $\ell a \not\equiv hb \equiv 0 \pmod N$.

(3) $\ell a \equiv hb \equiv 0 \pmod N$.

We now prove

Theorem 2 *The L-shape of $DL(N;a,b)$ is degenerate if and only if one of the following three conditions holds:*

(C1) $d > 1$ *and there exists* $1 \le i \le \min\{d, \frac{N}{d} - 1\}$ *such that* $db \equiv ia \pmod N$.

(C2) $d' > 1$ *and there exists* $1 \le j \le \min\{d' - 1, \frac{N}{d'} - 1\}$ *such that* $d'a \equiv jb \pmod N$.

(C3) $d > 1$, $d' > 1$ *and* $d'a \equiv db \equiv 0 \pmod N$.

Moreover, (C1) $\Leftrightarrow$ (1), (C2) $\Leftrightarrow$ (2) and (C3) $\Leftrightarrow$ (3). Also, if (C1) holds, then the degenerate L-shape is of height d and width N/d; if (C2) holds, then the degenerate L-shape is of height N/d' and width d'; if (C3) holds, then the degenerate L-shape is of height d and width d'.

Proof. Necessity. Suppose the L-shape is degenerate and is a rectangle of height h and width ℓ. Then by Lemma 1, it satisfies (1) or (2) or (3). We first prove two claims.

Claim 1. If $\ell a \equiv 0 \pmod N$, then $h = d$, $\ell = N/d$ and $d > 1$.

Proof of Claim 1. Let $a = \alpha d$ for some integer α. Note that the L-shape being degenerate implies $N = \ell h$. Thus $\ell a \equiv 0 \pmod N$ implies $a \equiv 0 \pmod h$. Let $a = \beta h$ for some integer β. Then $a = \alpha d = \beta h$. Hence $d = \frac{\beta h}{\alpha}$. Since $1 = \gcd(\alpha, \frac{N}{d}) = \gcd(\alpha, \frac{\ell h}{\frac{\beta h}{\alpha}}) = \gcd(\alpha, \frac{\ell\alpha}{\beta})$, necessarily $\alpha|\beta$. Therefore $\frac{\beta}{\alpha}$ is an integer. Since $d|N$, we have $\frac{\beta}{\alpha}|\ell$. Suppose $\frac{\beta}{\alpha} > 1$. Let $\ell' = \frac{\ell}{\frac{\beta}{\alpha}}$. Then $\ell' < \ell$ and $\ell' a = \frac{\ell}{\frac{\beta}{\alpha}}\beta h = \ell h \alpha = N\alpha \equiv 0 \pmod N$. Then row 0 of the L-shape will contain two entries of 0, one at cell (0,0) and the other at cell $(\ell', 0)$, a contradiction to the definition of an L-shape (recall that an MDD includes every node exactly once). Therefore $\frac{\beta}{\alpha} = 1$. Consequently, $h = d$ and $\ell = N/d$. Since $\ell < N$ and $\ell d = N$, clearly $d > 1$. ∎

Claim 2. If the L-shape is degenerate and $hb \equiv 0 \pmod N$, then $h = N/d'$, $\ell = d'$ and $d' > 1$.

Proof of Claim 2. Since this proof is similar to that of Claim 1, we omit it. ∎

We now prove the necessity of this theorem. First, assume the L-shape satisfies condition (1). By Claim 1, we have $d > 1$, $h = d$ and $\ell = N/d$. By the definition of an MDD, hb is the first element in column 0 satisfying

$$hb \equiv ia + jb \pmod N \text{ with } i + j \leq h,\ i \geq 0,\ j \geq 0.$$

Therefore $j = 0$ for otherwise $(h-j)b$ would be the first element. Also, $i \geq 1$ for otherwise $hb \equiv 0 \pmod N$. Thus $db = hb \equiv ia \pmod N$ for $1 \leq i \leq d$. Since $\ell = N/d$, we have

$i \le \frac{N}{d} - 1$. We conclude $db \equiv ia \pmod N$ for $1 \le i \le \min\{d, \frac{N}{d} - 1\}$, which means (C1) holds. The above discussion also shows that (1) implies (C1), i.e., (1) $\Rightarrow$ (C1).

Next, assume the L-shape satisfies condition (2). Then the argument is similar except at the end we have

$$\ell a \equiv ia + jb \pmod N \text{ with } i + j < \ell,\ i \ge 0,\ j \ge 0.$$

The reason for the strict inequality that $i + j < \ell$ is by our construction on tie-breaking in defining the MDD. Thus (C2) holds. So (2) $\Rightarrow$ (C2).

Finally, assume the L-shape satisfies condition (3). By Claim 1, we have $d > 1,\ h = d$ and $\ell = N/d$. By Claim 2, we have $d' > 1,\ h = N/d'$ and $\ell = d'$. Thus $d'a = \ell a \equiv 0 \pmod N$ and $db = hb \equiv 0 \pmod N$, which means (C3) holds. So (3) $\Rightarrow$ (C3).

<u>Sufficiency.</u> Let the L-shape of $DL(N; a, b)$ be (ℓ, h, p, n). First, assume that (C1) is satisfied. Since $db \equiv ia \pmod N$ for $1 \le i \le \min\{d, \frac{N}{d} - 1\}$, we have $h \le d$. On the other hand, $\ell \le N/d$ since $(N/d)a = N(a/d) \equiv 0 \pmod N$. Therefore

$$N = \ell h - pn \le \ell h \le (N/d)d = N.$$

Necessarily,

$$\ell = N/d,\ h = d.$$

It follows

$$\ell h = N,$$

i.e., the L-shape is degenerate. Moreover, $\ell a = (N/d)a = N(a/d) \equiv 0 \pmod N$; $hb = db \equiv ia \not\equiv 0 \pmod N$ since $1 \le i \le \ell - 1$. So (C1) $\Rightarrow$ (1).

The proof of (C2) is similar to that of (C1). Finally, assume that (C3) is satisfied. Then since $d'a \equiv db \equiv 0 \pmod N$, we have $\ell \le d'$ and $h \le d$. Since $d|N$, $d'|N$ and $\gcd(d, d') = 1$, we have $d'd \le N$. Therefore

$$N = \ell h - pn \le \ell h \le d'd \le N.$$

Necessarily,

$$\ell = d',\ h = d.$$

It follows

$$\ell h = N,$$

i.e., the L-shape is degenerate. Moreover, $\ell a = d'a \equiv 0 \pmod N$; $hb = db \equiv 0 \pmod N$. So (C3) $\Rightarrow$ (3). ∎

Remarks. From the proof of Theorem 2, when an L-shape(ℓ, h, p, n) degenerates into a rectangle, it is reasonable to set ℓ to the width and h to the height of the rectangle. Moreover, it is reasonable to set $p = 0$ or $n = 0$ since $N = \ell h - pn$ and $\ell h = N$ hold simultaneously.

3 Strongly isomorphic double-loop networks and degenerate L-shapes

The following property was proved in [1].

Lemma 3 [1] *If α and β are integers, not both zero, then there exist integers x and y such that $y\alpha + x\beta = \gcd(\alpha, \beta)$ and $\gcd(x, \gcd(\alpha, \beta)) = 1$.*

Let $DL(N; a, b)$ be a double-loop network. Then

Lemma 4 *There exists an integer x such that $\gcd(x, N) = 1$ and $ax \equiv d \pmod N$.*

Proof. Since $\gcd(N, a) = d$, by Lemma 3, there exist integers x and y such that $yN + xa = d$ and $\gcd(x, d) = 1$. Hence $ax \equiv d \pmod N$. Moreover, $y(N/d) + x(a/d) = 1$ implies $\gcd(x, N/d) = 1$. It follows that $\gcd(x, N) = \gcd(x, (N/d)d) = 1$. Hence the lemma. ∎

Two double-loop networks $DL(N; a, b)$ and $DL(N; a', b')$ are *strongly isomorphic* if there exists a z prime to N such that $a' \equiv az,\ b' \equiv bz \pmod N$ or $a' \equiv bz,\ b' \equiv az$

(mod N) [7]. It is well known that two strongly isomorphic double-loop networks realize the same L-shape. The following property greatly simplifies the proofs in the remaining sections.

Theorem 5 *Let x be an integer such that $\gcd(x, N) = 1$ and $ax \equiv d \pmod{N}$. Let $b'' = bx \pmod{N}$. Then $DL(N; a, b)$ and $DL(N; d, b'')$ are strongly isomorphic.*

Proof. This theorem follows from Lemma 4. ∎

In the following, we characterize a degenerate L-shape by the four independent parameters ℓ, h, p, n. Set

$$m = \ell - p, \ q = h - n$$

for convenience; see Fig. 2(a). Then

Lemma 6 *For a degenerate L-shape, at least one of m, n, p, q is zero and at most two of m, n, p, q are zero. Moreover, it is impossible that both m and p, both n and q, or both m and q are zero.*

Proof. It is obvious that at least one of m, n, p, q is zero. Since $\ell = m+p$ and $h = n+q$, if more than two of m, n, p, q are zero, then $\ell = 0$ or $h = 0$ will happen, which is impossible. Suppose two of m, n, p, q are zero. If both m and p (n and q) are zero, then $\ell = m+p = 0$ ($h = n+q = 0$), which is impossible. If both m and q are zero, then $\ell = p$, $h = n$, and then $N = \ell h - pn = 0$, which is also impossible. Hence the lemma. ∎

Corollary 7 *There are only seven possible ways to view a degenerate L-shape. We define these shapes by identifying the parameters which are set to zero:* **(S1)**: *only $m = 0$,* **(S2)**: *only $n = 0$,* **(S3)**: *only $p = 0$,* **(S4)**: *only $q = 0$,* **(S5)**: *$m = 0$ and $n = 0$,* **(S6)**: *$p = 0$ and $q = 0$,* **(S7)**: *$n = 0$ and $p = 0$.*

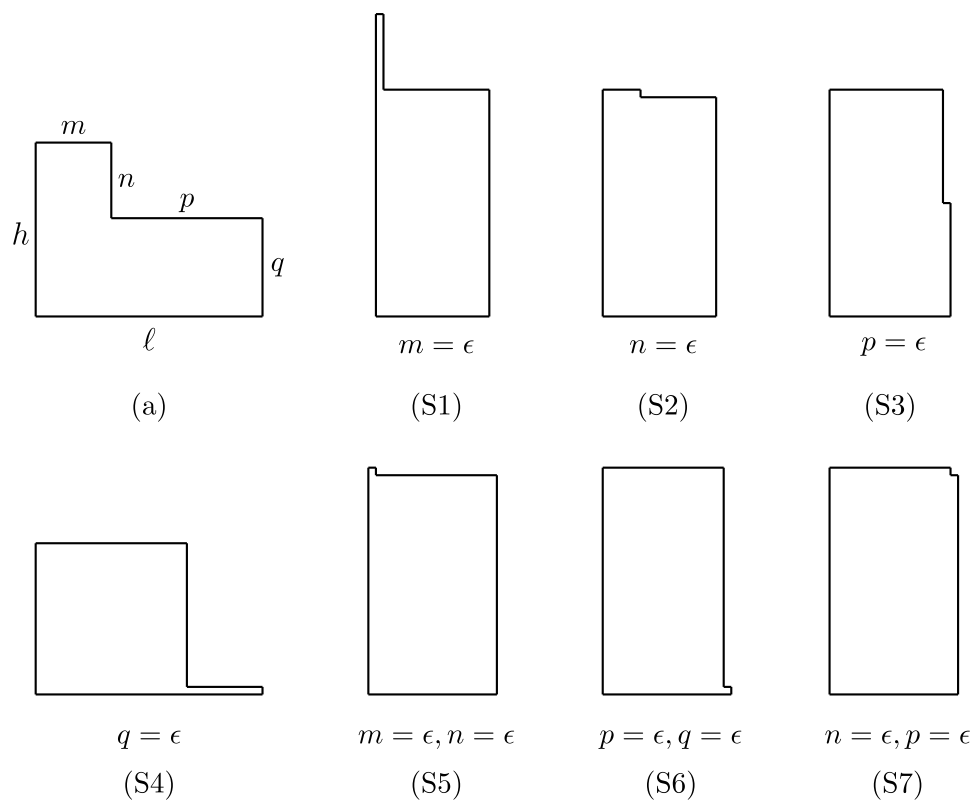


Figure 2: The ways to degenerate an L-shape.

By Corollary 7, there are seven ways to view a degenerate L-shape as the product of a limiting process operated on a regular L-shape. Fig. 2 (S2), (S3), (S5), (S6) and (S7) show five processes of shrinking a subrectangle with a side (or two sides) of length approaching zero; Fig. 2 (S1) and (S4) show two processes of cutting off a subrectangle with a side of length approaching ℓ or h. When $\epsilon = 0$, they all represent the same rectangle. But the different underlying process can induce different values of (ℓ, h, p, n).

Fiol, Yebra, Alegre, and Valero [5] pointed out that an L-shape, regular or degenerate, always tessellates the plane. Then $(\ell, -n)$ and $(-p, h)$ are simply two independent vectors characterizing the distribution of the nodes labelled by 0 (will be referred to as the 0-nodes) as seen by the equations:

$$\begin{array}{rcrcll} \ell a & - & nb & \equiv & 0 & \pmod N \\ -pa & + & hb & \equiv & 0 & \pmod N. \end{array} \tag{3.2}$$

Note that $(\ell, -n)$ is a vector in the fourth quadrant, and $(-p, h)$ one in the second. But there are other choices of two independent vectors.

4 Cheng-Hwang's algorithm

Cheng and Hwang [4] gave an algorithm (CH-ALGO in short) to solve for (ℓ, h, p, n) for $DL(N; a, b)$. The algorithm works regardless whether the L-shape is regular or not. For completeness, we give a brief review of this algorithm (note that the weight of each link in the given double-loop network is assumed to be 1).

CHENG-HWANG-ALGORITHM.

Input: $DL(N; a, b)$.

Output: (ℓ, h, p, n) of the L-shape of $DL(N; a, b)$.

Let $d,\ d',\ N',\ a'$ and b' be defined as in (2.1).

Let s_0 be the integer with

$$a's_0 + b' \equiv 0 \pmod{N'},\ 0 \le s_0 < N'.$$

Let $s_{-1} = N'$ and define q_i, s_i, recursively (by the Euclidean algorithm) as follows:

$$\begin{array}{lcll} s_{-1} & = & q_1 s_0 + s_1, & 0 \le s_1 < s_0 \\ s_0 & = & q_2 s_1 + s_2, & 0 \le s_2 < s_1 \\ s_1 & = & q_3 s_2 + s_3, & 0 \le s_3 < s_2 \\ & & \cdots & \\ s_{k-2} & = & q_k s_{k-1} + s_k, & 0 \le s_k < s_{k-1} \\ s_{k-1} & = & q_{k+1} s_k, & 0 = s_{k+1} < s_k. \end{array} \tag{4.3}$$

Define integers U_i by $U_{-1} = 0$, $U_0 = 1$, and

$$U_{i+1} = q_{i+1} U_i + U_{i-1},\ i = 0, 1, \cdots, k. \tag{4.4}$$

By induction,

$$s_i U_{i+1} + s_{i+1} U_i = N',\ i = 0, 1, \cdots, k. \tag{4.5}$$

Regard $s_{-1}/U_{-1} = \infty > x$ for real number x. Since $\{s_i\}_{i=-1}^{k+1}$ and $\{U_i\}_{i=-1}^{k+1}$ are strictly decreasing and increasing, respectively, we have

$$0 = \frac{s_{k+1}}{U_{k+1}} < \frac{s_k}{U_k} < \cdots < \frac{s_0}{U_0} < \frac{s_{-1}}{U_{-1}} = \infty.$$

Let u be the largest odd integer such that $d < \frac{s_u}{U_u}$. Define

$$v = \left\lceil \frac{s_u - dU_u}{s_{u+1} + dU_{u+1}} \right\rceil - 1.$$

Let

$$\ell' = s_u - vs_{u+1},\ h' = U_u + (v+1)U_{u+1},\ p' = s_u - (v+1)s_{u+1},\ n' = U_u + vU_{u+1}.$$

Then

$$(\ell, h, p, n) = (\ell', dh', p', dn').$$

End-of-CHENG-HWANG-ALGORITHM.

Now we characterize the (ℓ, h, p, n) obtained by CH-ALGO when $DL(N; a, b)$ has a degenerate L-shape. By Theorem 5, it suffices to consider the case that $a|N$. Since $a|N$, CH-ALGO derives

$$d = a,\ d' = \gcd(N, b),\ N' = N/d = N/a,\ a' = 1,\ b' = b \pmod{N'},\ s_{-1} = N'.$$

So we have

Lemma 8 $s_i \equiv (-1)^i U_i s_0 \pmod{N'}$ for $1 \le i \le k+1$.

Proof. By (4.3) and (4.4), $s_1 = s_{-1} - q_1 s_0 = N' - U_1 s_0$, $s_2 = s_0 - q_2 s_1 = s_0 - q_2(N' - U_1 s_0) = -q_2 N' + (1 + q_2 U_1)s_0 = -q_2 N' + U_2 s_0$. Thus $s_1 \equiv (-1)^1 U_1 s_0 \pmod{N'}$ and $s_2 \equiv (-1)^2 U_2 s_0 \pmod{N'}$. We prove the general case by induction on i. Assume this lemma holds for $i \le t$. Then, by (4.3) and (4.4), $s_{t+1} = s_{t-1} - q_{t+1} s_t$ and $U_{t+1} = U_{t-1} + q_{t+1} U_t$. Thus by induction,

$$\begin{aligned} s_{t+1} &\equiv (-1)^{t-1} U_{t-1} s_0 - q_{t+1}(-1)^t U_t s_0 \pmod{N'} \\ &= (-1)^{t+1}(U_{t-1} + q_{t+1} U_t) s_0 \pmod{N'} \\ &= (-1)^{t+1} U_{t+1} s_0 \pmod{N'}. \end{aligned}$$

∎

Theorem 9 *If $DL(N;a,b)$ satisfies*

(C1), then CH-ALGO derives an L-shape of shape (S2) with $(\ell,h,p,n)=(N',d,i,0)$*;*

(C2), then CH-ALGO derives an L-shape of shape

(S1) with $(\ell,h,p,n)=(d',j+\left\lceil\frac{d'-j}{\frac{N}{d'}}\right\rceil\frac{N}{d'},d',j+(\left\lceil\frac{d'-j}{\frac{N}{d'}}\right\rceil-1)\frac{N}{d'})$ *if* $j<\frac{N}{2d'}$*;*

(S3) with $(\ell,h,p,n)=(d',\frac{N}{d'},0,j)$ *if* $j\geq\frac{N}{2d'}$*;*

(C3), then CH-ALGO derives an L-shape of shape

(S1) with $(\ell,h,p,n)=(d',\lceil\frac{d'}{d}\rceil d,d',(\lceil\frac{d'}{d}\rceil-1)d)$ *if* $d<d'$*;*

(S5) with $(\ell,h,p,n)=(d',d,d',0)$ *if* $d>d'$*.*

Proof. First suppose $DL(N;a,b)$ satisfies (C1). Then there exists $1\leq i\leq\min\{d,N'-1\}$ such that $db\equiv ia\pmod N$. Since $a=d$, we have $b\equiv i\pmod{N'}$. Since $b'=b\pmod{N'}$ and $1\leq i\leq N'-1$, it follows that

$$b'=i.$$

By (4.3), we have $s_{-1}=q_1s_0+s_1$ and $q_1\geq 1$. Note that $s_0=N'-b'$ and $U_1=q_1$. So

$$\frac{s_1}{U_1}=\frac{s_1}{q_1}=\frac{s_{-1}}{q_1}-s_0=N'(\frac{1}{q_1}-1)+b'\leq b'=i\leq d.$$

Therefore $u=-1$. Since $b'=i\leq d$, $N'\leq(N'-b')+d$; therefore $\left\lceil\frac{N'}{(N'-b')+d}\right\rceil=1$. Thus $v=\left\lceil\frac{s_{-1}-dU_{-1}}{s_0+dU_0}\right\rceil-1=\left\lceil\frac{N'}{(N'-b')+d}\right\rceil-1=0$. Hence, $m=s_0=N'-b'>0,\ n=d(U_{-1}+vU_0)=0,\ p=s_{-1}-(v+1)s_0=b'=i>0,\ q=dU_0=d>0$. Thus the L-shape is of shape (S2) and

$$(\ell,h,p,n)=(N',d,i,0).$$

Now suppose $DL(N;a,b)$ satisfies (C2). So $DL(N;a,b)$ does not satisfy (C3). Hence $N>dd'$. Assume that $N=dd'N''$, where $N''>1$. By Theorem 2, there exists $1\leq j\leq\min\{d'-1,N/d'-1\}$ such that $d'a\equiv jb\pmod N$. Since $d=a$, we have $d'd\equiv jb\pmod N$. Since $\gcd(N,b)=d'$ and $N=dd'N''$, it follows that $d|j$. Let $j=dj'$. Then $d'd\equiv dj'b\pmod{dN'}$, which implies $d'\equiv j'b\pmod{N'}$. Thus

$$d'\equiv j'b'\pmod{N'}.$$

Note that $\gcd(N', b') = \gcd(N', b) = \gcd(N, b)$. Thus

$$\gcd(N', b') = d'.$$

We now have

$$s_k = \gcd(s_{-1}, s_0) = \gcd(N', N' - b') = \gcd(N', b') = d'$$

and $s_{k+1} = 0$. By (4.5), $s_k U_{k+1} + s_{k+1} U_k = N'$. Since $s_k = d'$ and $s_{k+1} = 0$, it follows that $d' U_{k+1} = d' N''$. Thus

$$U_{k+1} = N''.$$

By Lemma 8, $s_k \equiv (-1)^k U_k s_0 \equiv (-1)^k U_k (N' - b') \equiv (-1)^{k+1} U_k b' \pmod{N'}$. Since k is either odd or even, there are two cases:

Case 1. k is odd.

Then $s_k \equiv U_k b' \pmod{N'}$. Since $s_k = d' \equiv j'b' \pmod{N'}$, we have $U_k b' \equiv j'b' \pmod{N'}$. Thus $(U_k - j')b' \equiv 0 \pmod{N'}$. Since $U_k < U_{k+1}$, $U_k < N''$. Since $j < N/d'$, $j' < N''$. By the facts that $\gcd(N', b') = d'$ and $j' < N''$ and $U_k < N''$, it follows from $(U_k - j')b' \equiv 0 \pmod{N'}$ that

$$U_k = j'.$$

Then

$$\frac{s_k}{U_k} = \frac{d'}{j'} > d.$$

Hence $u = k$. Since $dU_k = dj' = j$ and $dU_{k+1} = dN'' = \frac{N}{d'}$,

$$v + 1 = \left\lceil \frac{s_k - dU_k}{s_{k+1} + dU_{k+1}} \right\rceil = \left\lceil \frac{d' - j}{\frac{N}{d'}} \right\rceil.$$

Thus $m = s_{k+1} = 0$, $n = d(j' + vN'') = j + v\frac{N}{d'} > 0$, $p = s_k - (v+1)s_{k+1} = d' > 0$, $q = dU_{k+1} = \frac{N}{d'} > 0$. So the L-shape is of shape (S1) and

$$(\ell, h, p, n) = (d', j + \left\lceil \frac{d' - j}{\frac{N}{d'}} \right\rceil \frac{N}{d'}, d', j + (\left\lceil \frac{d' - j}{\frac{N}{d'}} \right\rceil - 1)\frac{N}{d'}).$$

Note that since k is odd and $\{U_i\}_{i=-1}^{k+1}$ are strictly increasing, $U_{k-1} \geq 1$. Note also that $q_{k+1} \geq 2$. Thus by (4.4),

$$j = dj' = dU_k = d\frac{(U_{k+1} - U_{k-1})}{q_{k+1}} < d\frac{U_{k+1}}{2} = d\frac{N''}{2} = \frac{N}{2d'}.$$

Case 2. k is even.

Then $s_k \equiv -U_k b' \pmod{N'}$. Since $s_k = d' \equiv j'b' \pmod{N'}$, we have $-U_k b' \equiv j'b' \pmod{N'}$. Thus $(U_k + j')b' \equiv 0 \pmod{N'}$. Since $U_k < U_{k+1}$, $U_k < N''$. Since $j < N/d'$, $j' < N''$. By the facts that $\gcd(N', b') = d'$ and $j' < N''$ and $U_k < N''$, it follows from $(U_k + j')b' \equiv 0 \pmod{N'}$ that

$$U_k = N'' - j'.$$

Then by (4.3), (4.4) and the facts that $q_{k+1} \geq 2$ and $d' > j$,

$$\begin{aligned} s_{k-1} - dU_{k-1} &= q_{k+1}s_k - d(U_{k+1} - q_{k+1}U_k) \\ &= q_{k+1}d' - d(N'' - q_{k+1}(N'' - j')) \\ &= q_{k+1}(d' + \frac{N}{d'} - j) - \frac{N}{d'} > 0. \end{aligned}$$

Hence $u = k - 1$. Since $dU_k = d(N'' - j') = \frac{N}{d'} - j$,

$$v + 1 = \left\lceil \frac{s_{k-1} - dU_{k-1}}{s_k + dU_k} \right\rceil = \left\lceil \frac{q_{k+1}(d' + \frac{N}{d'} - j) - \frac{N}{d'}}{d' + \frac{N}{d'} - j} \right\rceil = \left\lceil q_{k+1} - \frac{\frac{N}{d'}}{d' + \frac{N}{d'} - j} \right\rceil = q_{k+1}.$$

Thus $m = s_k = d' > 0$, $n = d(U_{k-1} + (q_{k+1} - 1)U_k) = d(U_{k+1} - U_k) = d(N'' - (N'' - j')) = j > 0$, $p = s_{k-1} - q_{k+1}s_k = s_{k+1} = 0$, $q = dU_k = \frac{N}{d'} - j > 0$. So the L-shape is of shape (S3) and

$$(\ell, h, p, n) = (d', \frac{N}{d'}, 0, j).$$

Note that since $U_{k-1} \geq 0$ and $q_{k+1} \geq 2$,

$$j = dj' = d(N'' - U_k) = \frac{N}{d'} - d\frac{(U_{k+1} - U_{k-1})}{q_{k+1}} \geq \frac{N}{d'} - d\frac{N''}{2} \geq \frac{N}{d'} - \frac{N}{2d'} = \frac{N}{2d'}.$$

Note that when k is even, we have $j \geq \frac{N}{2d'}$. This implies that if $j < \frac{N}{2d'}$, then k is odd, which means Case 1 occurs. Therefore CH-ALGO derives an L-shape of shape (S1) if $j < \frac{N}{2d'}$ and an L-shape of shape (S3) if $j \geq \frac{N}{2d'}$.

Finally, suppose $DL(N;a,b)$ satisfies (C3). By Theorem 2, $N = dd'$; thus $N' = d'$. Since $db \equiv 0 \pmod{N}$, we have $b \equiv 0 \pmod{N'}$. Since $b' = b \pmod{N'}$, $b' = 0$. Therefore $s_0 = 0$ and

$$\frac{s_0}{U_0} = \frac{0}{1} < d.$$

Hence $u = -1$ and $v = \lceil \frac{N'}{d} \rceil - 1 = \lceil \frac{d'}{d} \rceil - 1$. Since $d \neq d'$, there are two cases:

Case 1. $d < d'$.

Then $v > 0$. So $m = s_0 = 0$, $n = d(U_{-1} + vU_0) = dv > 0$, $p = s_{-1} - s_0 = d' > 0$, $q = dU_0 = d > 0$. Thus the L-shape is of shape (S1) with

$$(\ell, h, p, n) = (d', \left\lceil \frac{d'}{d} \right\rceil d, d', (\left\lceil \frac{d'}{d} \right\rceil - 1)d).$$

Case 2. $d > d'$.

Then $v = 0$. So $m = s_0 = 0$, $n = d(U_{-1} + vU_0) = 0$, $p = s_{-1} - s_0 = N' - 0 = d' > 0$, $q = dU_0 = d > 0$. Thus the L-shape is of shape (S5) with

$$(\ell, h, p, n) = (d', d, d', 0).$$

∎

5 Chen-Hwang's rule

Chen and Hwang [3] gave a set of rules (CH-RULE in short) to determine the parameters ℓ, h, p, n for a degenerate L-shape. Their rules always set ℓ to the width and h to the height of the rectangle (the degenerate L-shape). We now briefly describe their rules.

CHEN-HWANG-RULE.

(i) Suppose $hb \not\equiv \ell a \equiv 0 \pmod{N}$. Let the zero immediately above the L-shape occurs at column j. Then

$$p = \ell - j, \; n = 0.$$

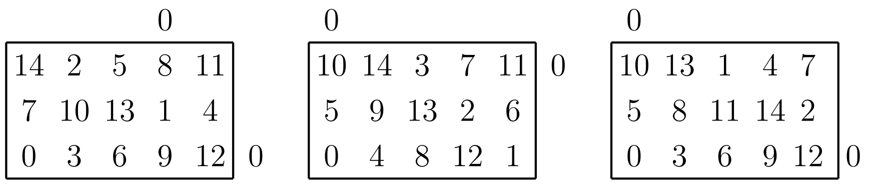
(ii) Suppose $\ell a \not\equiv hb \equiv 0 \pmod{N}$. Let the zero immediately to the right of the L-shape occurs at row i. Then

$$p = 0,\ n = h - i.$$

(iii) Suppose $\ell a \equiv hb \equiv 0 \pmod{N}$. If $h > \ell$, follow rule (i); otherwise, follow rule (ii).

End-of-CHEN-HWANG-RULE.

The ℓ, h, p, n chosen by CH-RULE satisfy the basic congruence equations in (3.2). Fig. 3 illustrates these rules.



$(\ell, h, p, n) = (5, 3, 2, 0)$ (a) rule (i) $(\ell, h, p, n) = (5, 3, 0, 1)$ (b) rule (ii) $(\ell, h, p, n) = (5, 3, 0, 3)$ (c) rule (iii)

Figure 3: The (ℓ, h, p, n) determined by CH-RULE.

W now characterize the (ℓ, h, p, n) obtained by CH-RULE when $DL(N; a, b)$ has a degenerate L-shape.

Theorem 10 *If $DL(N; a, b)$ satisfies*

(C1), then CH-RULE derives an L-shape of shape (S2) with $(\ell, h, p, n) = (N', d, i, 0)$;

(C2), then CH-RULE derives an L-shape of shape (S3) with $(\ell, h, p, n) = (d', \frac{N}{d'}, 0, j)$;

(C3), then CH-RULE derives an L-shape of shape

(S6) with $(\ell, h, p, n) = (d', d, 0, d)$ if $d < d'$;

(S5) with $(\ell, h, p, n) = (d', d, d', 0)$ if $d > d'$.

Proof. First, suppose $DL(N; a, b)$ satisfies (C1). Then there exists $1 \leq i \leq \min\{d, N' - 1\}$ such that $db \equiv ia \pmod{N}$. By Theorem 2, $\ell = N'$, $h = d$; also, (C1) $\Rightarrow$ (1). So

$hb \not\equiv \ell a \equiv 0 \pmod N$. Let the zero immediately above the L-shape occurs at column j. Since $\ell a \equiv 0 \pmod N$, $j = \ell - i$. So CH-RULE will follow rule (i) and will set $p = \ell - j = i$ and set $n = 0$. Thus $m = \ell - p = j > 0,\ n = 0,\ p = i > 0,\ q = h - n = h > 0$; so the L-shape is of shape (S2).

Next, suppose $DL(N; a, b)$ satisfies (C2). Then there exists $1 \le j \le \min\{d', \frac{N}{d'} - 1\}$ such that $d'a \equiv jb \pmod N$. By Theorem 2, $\ell = d'$ and $h = N/d'$; also, (C2) $\Rightarrow$ (2). So $\ell a \not\equiv hb \equiv 0 \pmod N$. Let the zero immediately to the right of L-shape occurs at row i. we have $i = h - j$. So CH-RULE will follow rule (ii) and will set $p = 0$ and set $n = h - i = j$. Thus $m = \ell - p = \ell > 0,\ n = j > 0,\ p = 0,\ q = h - n = N/d' - j > 0$; so the L-shape is of shape (S3).

Finally, suppose $DL(N; a, b)$ satisfies (C3). By Theorem 2, (C3) $\Rightarrow$ (Condition 3). So $\ell a \equiv hb \equiv 0 \pmod N$. Let the zero immediately above the L-shape occurs at column j and to the right of L-shape occurs at row i. Then $i = j = 0$. If $d < d'$, then $h < \ell$. So CH-RULE will follow rule (ii) and will set $p = 0$ and set $n = h - i = h = d$. Thus $m = \ell - p = \ell > 0,\ n = d > 0,\ p = 0,\ q = h - n = 0$; so the L-shape is of shape (S6). If $d > d'$, then $h > \ell$. So CH-RULE will follow rule (i) and will set $p = \ell - j = \ell = d'$ and set $n = 0$. Thus $m = \ell - p = 0,\ n = 0,\ p = d' > 0,\ q = h - n = d > 0$; so the L-shape is of shape (S5). ∎

6 The relations between CH-ALGO and CH-RULE

Both CH-ALGO and CH-RULE determine the four parameters ℓ, h, p, n for a degenerate L-shape. Unfortunately, the solution of (ℓ, h, p, n) using CH-RULE [3] does not always coincide with the values given by the CH-ALGO. For the example in Fig. 3 (b), the solution of the CH-RULE is

$$(\ell, h, p, n) = (5, 3, 0, 1)$$

and the solution of the CH-ALGO is

$$(\ell, h, p, n) = (5, 7, 5, 4)$$

(see Fig. 4). In this section, we will explain the relations between the two sets of solutions.

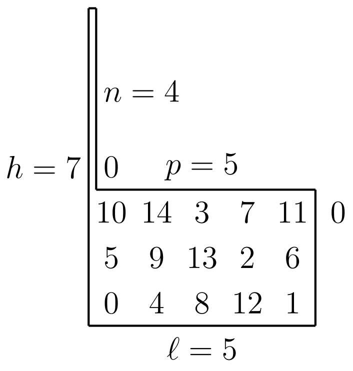


Figure 4: An alternative representation of the L-shape in Fig. 3 (b).

From Theorem 9 and Theorem 10, we know that CH-ALGO will not derive an L-shape of shape (S4) or (S6) or (S7) and CH-RULE will not derive an L-shape of shape (S1) or (S4) or (S7). We now further explain the reason below. CH-ALGO will not derive an L-shape of shape (S4) or (S6) because it always has $q = h - n = dU_{u+1} > 0$ (recall that $\{U_i\}_{i=-1}^{k+1}$ is strictly increasing and $U_{-1} = 0$). Also, CH-ALGO will not derive an L-shape of shape (S7) since if $n = d(U_u + vU_{u+1}) = 0$, then $u = -1$ and $v = 0$ and therefore $p = s_u - (v+1)s_{u+1} = s_{-1} - s_0 > 0$, a contradiction to the assumption that the L-shape is of shape (S7). CH-RULE will not derive an L-shape of shape (S1) or (S4) since it always sets ℓ to the width and h to the height of the degenerate L-shape. Also CH-RULE will not derive an L-shape of shape (S7) since it always has n and p not both zero. We now summarize the results of Theorem 9 and Theorem 10 in Table 1 and compare the degenerate shapes derived by CH-ALGO and CH-RULE in Table 2.

The following three corollaries follow from Theorem 9 and Theorem 10.

Corollary 11 *CH-ALGO and CH-RULE derive the same shape when $DL(N; a, b)$ satisfies (C1), satisfies (C2) and $j \geq \frac{N}{2d'}$ or satisfies (C3) and $d > d'$. CH-ALGO and CH-RULE derive different shapes when $DL(N; a, b)$ satisfies (C2) and $j < \frac{N}{2d'}$ or satisfies (C3) and $d < d'$.*

Table 1: The shapes derived by CH-ALGO and CH-RULE.

shape	S1	S2	S3	S4	S5	S6	S7
CH-ALGO	v	v	v		v		
CH-RULE		v	v		v	v	

Table 2: The comparison between CH-ALGO and CH-RULE.

condition	C1	C2		C3	
		$j < \frac{N}{2d'}$	$j \geq \frac{N}{2d'}$	$d < d'$	$d > d'$
CH-ALGO	S2	S1	S3	S1	S5
CH-RULE	S2	S3	S3	S6	S5
consistent	yes	no	yes	no	yes

Let $(\hat{\ell}, \hat{h}, \hat{p}, \hat{n})$ denote the solution of CH-ALGO and $(\dot{\ell}, \dot{h}, \dot{p}, \dot{n})$, the solution of CH-RULE. Corollary 12 and Corollary 13 show that when the two sets of solutions are different, one can be obtained from the other.

Corollary 12 *If $DL(N; a, b)$ satisfies (C2) and $j < \frac{N}{2d'}$, then*

$$\hat{\ell} = \hat{p} = \dot{\ell},\ \hat{h} = \dot{n} + \left\lceil \frac{\dot{\ell} - \dot{n}}{\dot{h}} \right\rceil \dot{h},\ \hat{n} = \dot{n} + (\left\lceil \frac{\dot{\ell} - \dot{n}}{\dot{h}} \right\rceil - 1)\dot{h},$$

and

$$\dot{\ell} = \hat{\ell},\ \dot{h} = \hat{h} - \hat{n},\ \dot{p} = 0,\ \dot{n} = j.$$

Corollary 13 *If $DL(N; a, b)$ satisfies (C3) and $d < d'$, then*

$$\hat{\ell} = \hat{p} = \dot{\ell},\ \hat{h} = \left\lceil \frac{\dot{\ell}}{\dot{h}} \right\rceil \dot{h},\ \hat{n} = (\left\lceil \frac{\dot{\ell}}{\dot{h}} \right\rceil - 1)\dot{h},$$

and

$$\dot{\ell} = \hat{\ell},\ \dot{h} = \dot{n} = \hat{h} - \hat{n},\ \dot{p} = 0.$$

References

[1] R. C. Chan, C. Y. Chen, and Z. X. Hong, “A simple algorithm to find the steps of double-loop networks,” Discrete Appl. Math. 121 (2002), 61-72.

[2] C. Y. Chen and F. K. Hwang, “The minimum distance diagram of double-loop networks,” IEEE Trans. Comput. 49 (2000), 977-979.

[3] C. Y. Chen and F. K. Hwang, “Equivalent L-shapes of double-loop networks for the degenerate case,” Journal of Interconnection Networks 1 (2000), 47-60.

[4] Y. Cheng and F. K. Hwang, “Diameters of weighted double loop networks,” J. Algorithms 9 (1988), 401-410.

[5] M. A. Fiol, J. L. A. Yebra, I. Alegre, and M. Valero, “A discrete optimization problem in local networks and data alignment,” IEEE Trans. Comput. C-36 (1987), 702-713.

[6] F. K. Hwang, “A complementary survey on double-loop networks,” Theoret. Comput. Sci. 263 (2001), 211-229.

[7] F. K. Hwang and W. W. Li,“Reliabilities of double-loop networks,” Probability in the Engineering and Informational Sciences 5 (1991), 255-272.

[8] C. K. Wong and D. Coppersmith, “A combinatorial problem related to multimodule organizations,” J. Assoc. Comput. Mach. 21 (1974), 392-402.

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

Jenny C. Chen[†], Frank K. Hwang and Jing-Kai Luo

Department of Applied Mathematics

National Chiao Tung University

Hsinchu 300, Taiwan

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 paper, 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.

1 Introduction

The purpose of this paper is to derive tag-based routing algorithms for the backward network of a bidirectional general shuffle-exchange network. Throughout this paper, N'

*This research was partially supported by the National Science Council of the Republic of China under the grant NSC93-2115-M-009-011.

†The corresponding author, e-mail: cychen@mail.nctu.edu.tw

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

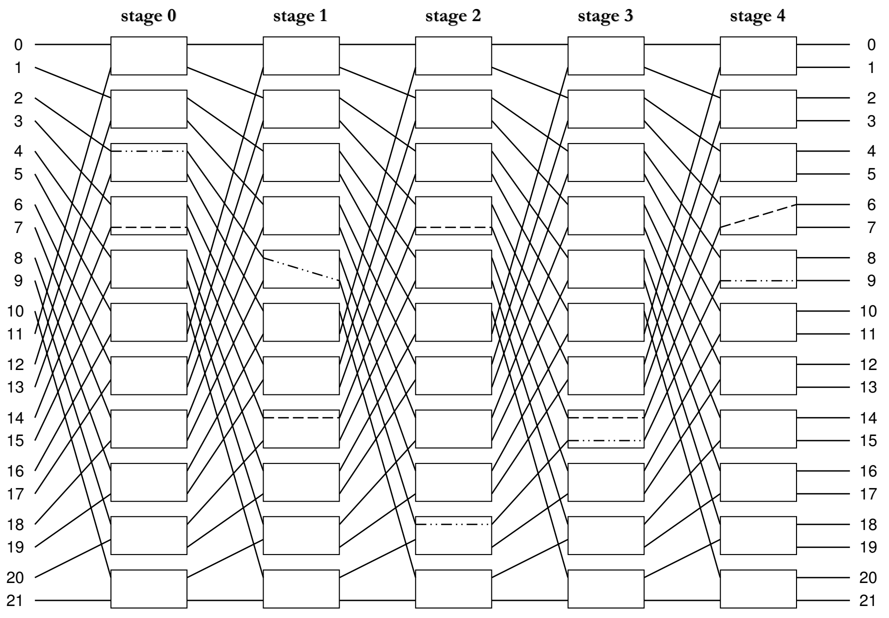
$$\pi(i) = (ki + \left\lfloor \frac{ki}{N'} \right\rfloor) \bmod N', \quad 0 \le i \le N' - 1.$$

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 discard 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 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



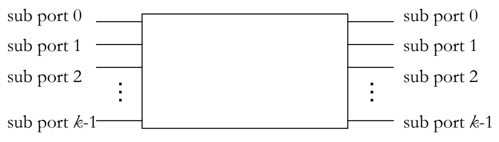


Figure 1: (a) The GSEN with $N' = 22$ and $k = 2$; this figure also shows GSEN(2,11,5). (b) A $k \times k$ switch element and its sub ports.

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 paper, 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 paper is organized as follows. In Section 2, we formally define the bidirectional GSEN and give conventions used in this paper. 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 paper

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 $0, 1, 2, \dots, n$. Each stage consists of r switch elements, labelled $0, 1, 2, \dots, r-1$. 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 totally

$$N' = k \times r$$

ports on each side of a stage, labelled $0, 1, 2, \dots, N'-1$. 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 paper, let

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

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 paper, 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 $(t_0 t_1 \dots t_n)$ 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

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

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:

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

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

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

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 $(s_0s_1 \dots s_n)$ and $(s'_0s'_1 \dots s'_n)$.

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

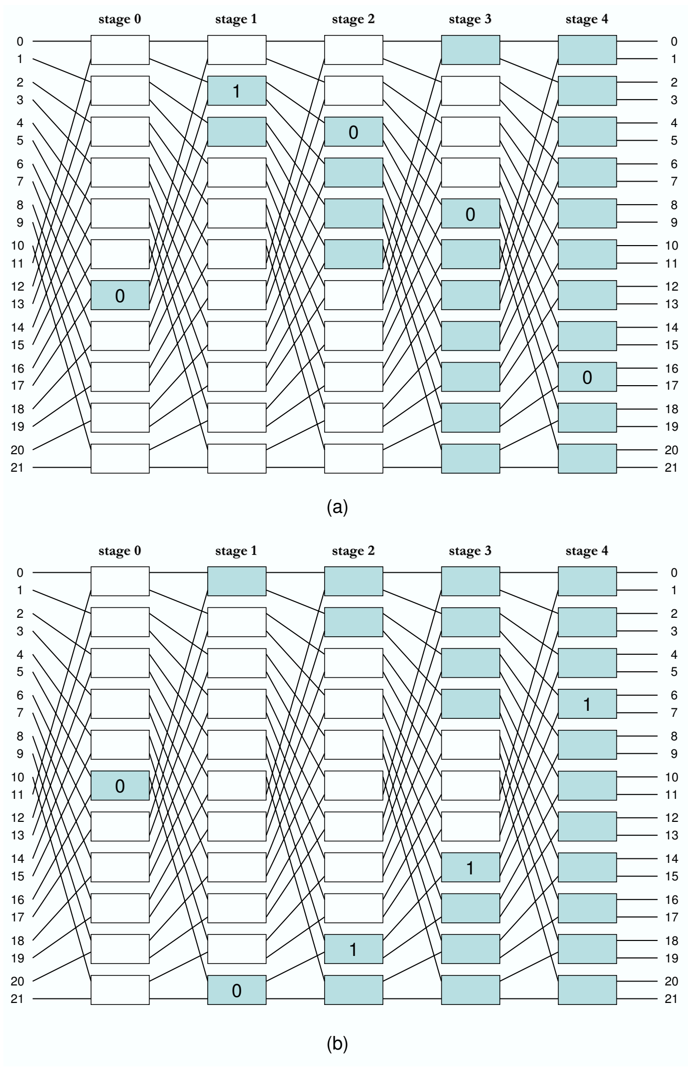


Figure 2: GSEN(2,11,5) with the switch elements that can get to (a) $i = 6$ and (b) $i = 5$ being highlighted.

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, $(s'_0s'_1s'_2s'_3s'_4) = (01000)$. After Step 4, $F_0 = 0$, $F_1 = 0$, $F_2 = 0$, $F_3 = 0$ and $F_4 = 1$. After Step 5, $(s_0s_1s_2s_3s_4) = (01001)$. It is easy to verify that: if $j < 16$, then j can get to 6 by using the tag (01000); if $j \geq 16$, then j can get to 6 by using the tag (01001). We summarize the above results in the following table.

destination i	$(s_0s_1s_2s_3s_4)$	$(s'_0s'_1s'_2s'_3s'_4)$	$v(i)$
$i = 6$	01001	01000	16

Recall that there are totally 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, $1 \le \ell \le n$, 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 $0 \le \left\lfloor \frac{d}{k^{n-\ell+1}} \right\rfloor < k^\ell$. 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 $\ell = n-1, n-2, \ldots, 0$,

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

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, $1 \le \ell \le n-1$, 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 $C_\ell + k^\ell \le r$ (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 $0 < \frac{d}{k^{n+1}} \le \frac{N'-1}{N'} < 1$. Thus $L_0 = C_0$. By Lemma 5, the tag for L_0 to get to i is s'_0. Note that $C_0 + k^0 \le r$. 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 $j \ge v(i)$, 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.

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, pp. 121-123, 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 Theroy 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, pp. 1145-1155, 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.

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

GSEN(2, 9, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 14$
$i = 2$	0 0 1 0 0	0 0 0 1 1	$v_i = 10$
$i = 3$	0 0 1 1 0	0 0 1 0 1	$v_i = 6$
$i = 4$	0 1 0 0 0	0 0 1 1 1	$v_i = 2$
$i = 5$	0 1 0 0 1	0 1 0 0 0	$v_i = 16$
$i = 6$	0 1 0 1 1	0 1 0 1 0	$v_i = 12$
$i = 7$	0 1 1 0 1	0 1 1 0 0	$v_i = 8$
$i = 8$	0 1 1 1 1	0 1 1 1 0	$v_i = 4$
$i = 9$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 10$	1 0 0 1 0	1 0 0 0 1	$v_i = 14$
$i = 11$	1 0 1 0 0	1 0 0 1 1	$v_i = 10$
$i = 12$	1 0 1 1 0	1 0 1 0 1	$v_i = 6$
$i = 13$	1 1 0 0 0	1 0 1 1 1	$v_i = 2$
$i = 14$	1 1 0 0 1	1 1 0 0 0	$v_i = 16$
$i = 15$	1 1 0 1 1	1 1 0 1 0	$v_i = 12$
$i = 16$	1 1 1 0 1	1 1 1 0 0	$v_i = 8$
$i = 17$	1 1 1 1 1	1 1 1 1 0	$v_i = 4$

GSEN(2, 10, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 12$
$i = 2$	0 0 1 0 0	0 0 0 1 1	$v_i = 4$
$i = 3$	0 0 1 0 1	0 0 1 0 0	$v_i = 16$
$i = 4$	0 0 1 1 1	0 0 1 1 0	$v_i = 8$
$i = 5$	0 1 0 0 1	0 1 0 0 0	$v_i = 0$
$i = 6$	0 1 0 1 0	0 1 0 0 1	$v_i = 12$
$i = 7$	0 1 1 0 0	0 1 0 1 1	$v_i = 4$
$i = 8$	0 1 1 0 1	0 1 1 0 0	$v_i = 16$
$i = 9$	0 1 1 1 1	0 1 1 1 0	$v_i = 8$
$i = 10$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 11$	1 0 0 1 0	1 0 0 0 1	$v_i = 12$
$i = 12$	1 0 1 0 0	1 0 0 1 1	$v_i = 4$
$i = 13$	1 0 1 0 1	1 0 1 0 0	$v_i = 16$
$i = 14$	1 0 1 1 1	1 0 1 1 0	$v_i = 8$
$i = 15$	1 1 0 0 1	1 1 0 0 0	$v_i = 0$
$i = 16$	1 1 0 1 0	1 1 0 0 1	$v_i = 12$
$i = 17$	1 1 1 0 0	1 1 0 1 1	$v_i = 4$
$i = 18$	1 1 1 0 1	1 1 1 0 0	$v_i = 16$
$i = 19$	1 1 1 1 1	1 1 1 1 0	$v_i = 8$

GSEN(2, 11, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 10$
$i = 2$	0 0 0 1 1	0 0 0 1 0	$v_i = 20$
$i = 3$	0 0 1 0 1	0 0 1 0 0	$v_i = 8$
$i = 4$	0 0 1 1 0	0 0 1 0 1	$v_i = 18$
$i = 5$	0 1 0 0 0	0 0 1 1 1	$v_i = 6$
$i = 6$	0 1 0 0 1	0 1 0 0 0	$v_i = 16$
$i = 7$	0 1 0 1 1	0 1 0 1 0	$v_i = 4$
$i = 8$	0 1 1 0 0	0 1 0 1 1	$v_i = 14$
$i = 9$	0 1 1 1 0	0 1 1 0 1	$v_i = 2$
$i = 10$	0 1 1 1 1	0 1 1 1 0	$v_i = 12$
$i = 11$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 12$	1 0 0 1 0	1 0 0 0 1	$v_i = 10$
$i = 13$	1 0 0 1 1	1 0 0 1 0	$v_i = 20$
$i = 14$	1 0 1 0 1	1 0 1 0 0	$v_i = 8$
$i = 15$	1 0 1 1 0	1 0 1 0 1	$v_i = 18$
$i = 16$	1 1 0 0 0	1 0 1 1 1	$v_i = 6$
$i = 17$	1 1 0 0 1	1 1 0 0 0	$v_i = 16$
$i = 18$	1 1 0 1 1	1 1 0 1 0	$v_i = 4$
$i = 19$	1 1 1 0 0	1 1 0 1 1	$v_i = 14$
$i = 20$	1 1 1 1 0	1 1 1 0 1	$v_i = 2$
$i = 21$	1 1 1 1 1	1 1 1 1 0	$v_i = 12$

GSEN(2, 12, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 8$
$i = 2$	0 0 0 1 1	0 0 0 1 0	$v_i = 16$
$i = 3$	0 0 1 0 1	0 0 1 0 0	$v_i = 0$
$i = 4$	0 0 1 1 0	0 0 1 0 1	$v_i = 8$
$i = 5$	0 0 1 1 1	0 0 1 1 0	$v_i = 16$
$i = 6$	0 1 0 0 1	0 1 0 0 0	$v_i = 0$
$i = 7$	0 1 0 1 0	0 1 0 0 1	$v_i = 8$
$i = 8$	0 1 0 1 1	0 1 0 1 0	$v_i = 16$
$i = 9$	0 1 1 0 1	0 1 1 0 0	$v_i = 0$
$i = 10$	0 1 1 1 0	0 1 1 0 1	$v_i = 8$
$i = 11$	0 1 1 1 1	0 1 1 1 0	$v_i = 16$
$i = 12$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 13$	1 0 0 1 0	1 0 0 0 1	$v_i = 8$
$i = 14$	1 0 0 1 1	1 0 0 1 0	$v_i = 16$
$i = 15$	1 0 1 0 1	1 0 1 0 0	$v_i = 0$
$i = 16$	1 0 1 1 0	1 0 1 0 1	$v_i = 8$
$i = 17$	1 0 1 1 1	1 0 1 1 0	$v_i = 16$
$i = 18$	1 1 0 0 1	1 1 0 0 0	$v_i = 0$
$i = 19$	1 1 0 1 0	1 1 0 0 1	$v_i = 8$
$i = 20$	1 1 0 1 1	1 1 0 1 0	$v_i = 16$
$i = 21$	1 1 1 0 1	1 1 1 0 0	$v_i = 0$
$i = 22$	1 1 1 1 0	1 1 1 0 1	$v_i = 8$
$i = 23$	1 1 1 1 1	1 1 1 1 0	$v_i = 16$

GSEN(2, 13, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 6$
$i = 2$	0 0 0 1 1	0 0 0 1 0	$v_i = 12$
$i = 3$	0 0 1 0 0	0 0 0 1 1	$v_i = 18$
$i = 4$	0 0 1 0 1	0 0 1 0 0	$v_i = 24$
$i = 5$	0 0 1 1 1	0 0 1 1 0	$v_i = 4$
$i = 6$	0 1 0 0 0	0 0 1 1 1	$v_i = 10$
$i = 7$	0 1 0 0 1	0 1 0 0 0	$v_i = 16$
$i = 8$	0 1 0 1 0	0 1 0 0 1	$v_i = 22$
$i = 9$	0 1 1 0 0	0 1 0 1 1	$v_i = 2$
$i = 10$	0 1 1 0 1	0 1 1 0 0	$v_i = 8$
$i = 11$	0 1 1 1 0	0 1 1 0 1	$v_i = 14$
$i = 12$	0 1 1 1 1	0 1 1 1 0	$v_i = 20$
$i = 13$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 14$	1 0 0 1 0	1 0 0 0 1	$v_i = 6$
$i = 15$	1 0 0 1 1	1 0 0 1 0	$v_i = 12$
$i = 16$	1 0 1 0 0	1 0 0 1 1	$v_i = 18$
$i = 17$	1 0 1 0 1	1 0 1 0 0	$v_i = 24$
$i = 18$	1 0 1 1 1	1 0 1 1 0	$v_i = 4$
$i = 19$	1 1 0 0 0	1 0 1 1 1	$v_i = 10$
$i = 20$	1 1 0 0 1	1 1 0 0 0	$v_i = 16$
$i = 21$	1 1 0 1 0	1 1 0 0 1	$v_i = 22$
$i = 22$	1 1 1 0 0	1 1 0 1 1	$v_i = 2$
$i = 23$	1 1 1 0 1	1 1 1 0 0	$v_i = 8$
$i = 24$	1 1 1 1 0	1 1 1 0 1	$v_i = 14$
$i = 25$	1 1 1 1 1	1 1 1 1 0	$v_i = 20$

GSEN(2, 14, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 4$
$i = 2$	0 0 0 1 1	0 0 0 1 0	$v_i = 8$
$i = 3$	0 0 1 0 0	0 0 0 1 1	$v_i = 12$
$i = 4$	0 0 1 0 1	0 0 1 0 0	$v_i = 16$
$i = 5$	0 0 1 1 0	0 0 1 0 1	$v_i = 20$
$i = 6$	0 0 1 1 1	0 0 1 1 0	$v_i = 24$
$i = 7$	0 1 0 0 1	0 1 0 0 0	$v_i = 0$
$i = 8$	0 1 0 1 0	0 1 0 0 1	$v_i = 4$
$i = 9$	0 1 0 1 1	0 1 0 1 0	$v_i = 8$
$i = 10$	0 1 1 0 0	0 1 0 1 1	$v_i = 12$
$i = 11$	0 1 1 0 1	0 1 1 0 0	$v_i = 16$
$i = 12$	0 1 1 1 0	0 1 1 0 1	$v_i = 20$
$i = 13$	0 1 1 1 1	0 1 1 1 0	$v_i = 24$
$i = 14$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 15$	1 0 0 1 0	1 0 0 0 1	$v_i = 4$
$i = 16$	1 0 0 1 1	1 0 0 1 0	$v_i = 8$
$i = 17$	1 0 1 0 0	1 0 0 1 1	$v_i = 12$
$i = 18$	1 0 1 0 1	1 0 1 0 0	$v_i = 16$
$i = 19$	1 0 1 1 0	1 0 1 0 1	$v_i = 20$
$i = 20$	1 0 1 1 1	1 0 1 1 0	$v_i = 24$
$i = 21$	1 1 0 0 1	1 1 0 0 0	$v_i = 0$
$i = 22$	1 1 0 1 0	1 1 0 0 1	$v_i = 4$
$i = 23$	1 1 0 1 1	1 1 0 1 0	$v_i = 8$
$i = 24$	1 1 1 0 0	1 1 0 1 1	$v_i = 12$
$i = 25$	1 1 1 0 1	1 1 1 0 0	$v_i = 16$
$i = 26$	1 1 1 1 0	1 1 1 0 1	$v_i = 20$
$i = 27$	1 1 1 1 1	1 1 1 1 0	$v_i = 24$

GSEN(2, 15, 5)			
$i = 0$	0 0 0 0 1	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 1 0	0 0 0 0 1	$v_i = 2$
$i = 2$	0 0 0 1 1	0 0 0 1 0	$v_i = 4$
$i = 3$	0 0 1 0 0	0 0 0 1 1	$v_i = 6$
$i = 4$	0 0 1 0 1	0 0 1 0 0	$v_i = 8$
$i = 5$	0 0 1 1 0	0 0 1 0 1	$v_i = 10$
$i = 6$	0 0 1 1 1	0 0 1 1 0	$v_i = 12$
$i = 7$	0 1 0 0 0	0 0 1 1 1	$v_i = 14$
$i = 8$	0 1 0 0 1	0 1 0 0 0	$v_i = 16$
$i = 9$	0 1 0 1 0	0 1 0 0 1	$v_i = 18$
$i = 10$	0 1 0 1 1	0 1 0 1 0	$v_i = 20$
$i = 11$	0 1 1 0 0	0 1 0 1 1	$v_i = 22$
$i = 12$	0 1 1 0 1	0 1 1 0 0	$v_i = 24$
$i = 13$	0 1 1 1 0	0 1 1 0 1	$v_i = 26$
$i = 14$	0 1 1 1 1	0 1 1 1 0	$v_i = 28$
$i = 15$	1 0 0 0 1	1 0 0 0 0	$v_i = 0$
$i = 16$	1 0 0 1 0	1 0 0 0 1	$v_i = 2$
$i = 17$	1 0 0 1 1	1 0 0 1 0	$v_i = 4$
$i = 18$	1 0 1 0 0	1 0 0 1 1	$v_i = 6$
$i = 19$	1 0 1 0 1	1 0 1 0 0	$v_i = 8$
$i = 20$	1 0 1 1 0	1 0 1 0 1	$v_i = 10$
$i = 21$	1 0 1 1 1	1 0 1 1 0	$v_i = 12$
$i = 22$	1 1 0 0 0	1 0 1 1 1	$v_i = 14$
$i = 23$	1 1 0 0 1	1 1 0 0 0	$v_i = 16$
$i = 24$	1 1 0 1 0	1 1 0 0 1	$v_i = 18$
$i = 25$	1 1 0 1 1	1 1 0 1 0	$v_i = 20$
$i = 26$	1 1 1 0 0	1 1 0 1 1	$v_i = 22$
$i = 27$	1 1 1 0 1	1 1 1 0 0	$v_i = 24$
$i = 28$	1 1 1 1 0	1 1 1 0 1	$v_i = 26$
$i = 29$	1 1 1 1 1	1 1 1 1 0	$v_i = 28$

GSEN(2, 16, 5)			
$i = 0$	0 0 0 0 0	0 0 0 0 0	$v_i = 0$
$i = 1$	0 0 0 0 1	0 0 0 0 1	$v_i = 0$
$i = 2$	0 0 0 1 0	0 0 0 1 0	$v_i = 0$
$i = 3$	0 0 0 1 1	0 0 0 1 1	$v_i = 0$
$i = 4$	0 0 1 0 0	0 0 1 0 0	$v_i = 0$
$i = 5$	0 0 1 0 1	0 0 1 0 1	$v_i = 0$
$i = 6$	0 0 1 1 0	0 0 1 1 0	$v_i = 0$
$i = 7$	0 0 1 1 1	0 0 1 1 1	$v_i = 0$
$i = 8$	0 1 0 0 0	0 1 0 0 0	$v_i = 0$
$i = 9$	0 1 0 0 1	0 1 0 0 1	$v_i = 0$
$i = 10$	0 1 0 1 0	0 1 0 1 0	$v_i = 0$
$i = 11$	0 1 0 1 1	0 1 0 1 1	$v_i = 0$
$i = 12$	0 1 1 0 0	0 1 1 0 0	$v_i = 0$
$i = 13$	0 1 1 0 1	0 1 1 0 1	$v_i = 0$
$i = 14$	0 1 1 1 0	0 1 1 1 0	$v_i = 0$
$i = 15$	0 1 1 1 1	0 1 1 1 1	$v_i = 0$
$i = 16$	1 0 0 0 0	1 0 0 0 0	$v_i = 0$
$i = 17$	1 0 0 0 1	1 0 0 0 1	$v_i = 0$
$i = 18$	1 0 0 1 0	1 0 0 1 0	$v_i = 0$
$i = 19$	1 0 0 1 1	1 0 0 1 1	$v_i = 0$
$i = 20$	1 0 1 0 0	1 0 1 0 0	$v_i = 0$
$i = 21$	1 0 1 0 1	1 0 1 0 1	$v_i = 0$
$i = 22$	1 0 1 1 0	1 0 1 1 0	$v_i = 0$
$i = 23$	1 0 1 1 1	1 0 1 1 1	$v_i = 0$
$i = 24$	1 1 0 0 0	1 1 0 0 0	$v_i = 0$
$i = 25$	1 1 0 0 1	1 1 0 0 1	$v_i = 0$
$i = 26$	1 1 0 1 0	1 1 0 1 0	$v_i = 0$
$i = 27$	1 1 0 1 1	1 1 0 1 1	$v_i = 0$
$i = 28$	1 1 1 0 0	1 1 1 0 0	$v_i = 0$
$i = 29$	1 1 1 0 1	1 1 1 0 1	$v_i = 0$
$i = 30$	1 1 1 1 0	1 1 1 1 0	$v_i = 0$
$i = 31$	1 1 1 1 1	1 1 1 1 1	$v_i = 0$