On The VLSI Decompositions for Complete Graphs, DeBruijn Graphs, Hypercubes, Hyperplanes, Meshes, and Shuffle-Exchange Graphs

Thesis by

Tsz-Mei Ko

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1993

(Defended August 3, 1992)

To all complex systems that process and communicate

Acknowledgements

Special thanks to Dr. Robert J. McEliece for his extraordinary supervision during my three years of graduate studies at Caltech. His directions have led me to the insights for the solutions of many problems.

Thanks to Dr. Edward Posner, Dr. Yaser Abu-Mostafa, Dr. Charles Seitz, and Dr. Alain Martin for their valuable comments that have improved the readability and correctness of this thesis.

Thanks to Dr. Sam Dolinar, Dr. Kar-Ming Cheung, Dr. Ivan Onyszchuk, Dr. Laif Swanson, and other members of the Advanced Error Correcting Code Research and Development Team at the Jet Propulsion Laboratories for their inspiring discussions. I have learnt many practical aspects of coding and decoding from them. Thanks also go to Dr. Oliver Collins, from the Johns Hopkins University, who originally formulated the VLSI decomposition problem when he was doing research at Caltech/JPL. Dr. Collins has also taught me Theorem 6.2 and its proof.

Thanks to Dr. Michael Little, Dr. Michael Yung, Dr. Michael Campbell, and other members of the 3D Microelectronics Group at the Hughes Research Laboratories. They have helped me in understanding the connectivity issue and other limitations in the current VLSI technology.

Thanks to the National Security Agency and the Hughes Aircraft Company to provide the fundings that made this research possible. This research was partially supported by the National Security Agency under Grant Number MDA904-90-H-1007 and partially supported by a Hughes Doctoral Fellowship.

Thanks to all the people in the Caltech/JPL community. They have truely created an enjoyable research environment. I have learnt a lot through the continuous interaction with all of them.

Thanks to Ms Debbie McGougan and other secretaries in the Electrical Engineering Department for their assistances. They have saved me a great deal of time.

Thanks to all my teachers, colleagues, relatives, and friends. They have made this universe a much more wonderful place to live.

Last but not the least, a googolplex of special thanks to my parents, Wa and Kwok-Lin Ko, and my brother, Tze-Man Ko, for their unlimited support.

Abstract

A *C-chip VLSI decomposition* of a graph G is a collection of C vertex-disjoint subgraphs of G which together contain all of G's vertices and a subset of its edges. If the vertex-disjoint subgraphs are isomorphic to each other, we call one of these isomorphic subgraphs a *building block*. The *efficiency* of a VLSI decomposition is defined to be the fraction of edges of G that are in the subgraphs. In this thesis, motivated by the need to construct large Viterbi decoders, we study VLSI decompositions for deBruijn graphs. We obtain some strong necessary conditions for a graph to be a building block for a deBruijn graph, and some slightly more restrictive sufficient conditions which allow us to construct some efficient building blocks for deBruijn graphs. By using the methods described in this thesis, we have found a 64-chip VLSI decomposition of the deBruijn graph B_{13} with efficiency 0.754. This decomposition is being used by JPL design engineers to build a single-board Viterbi decoder for the $K = 15$, rate 1/4 convolutional code which will be used on NASA's *Galileo* mission.

Furthermore, we study VLSI decompositions for the families of complete graphs, hypercubes, hyperplanes, meshes, and shuffle-exchange graphs. In each of these cases, we obtain very efficient or even optimal decompositions. We also prove several general theorems that can be applied to obtain bounds on the efficiencies for VLSI decompositions of other complex graphs. In general, the results presented in this thesis are useful for implementing massively parallel computers.

Table of Contents

I. Introduction

There are two major problems in designing massively parallel computers—(i) choose the appropriate interconnection network; and (ii) implement the choosen architecture in minimum cost. In this thesis, we will concentrate on the second problem.

For the first problem, the solution usually relies on the effectiveness for the different architectures to implement certain applications and algorithms. Many researches [Leig92] have been done to compare the algorithmic times for different interconnection schemes. As an example, the hypercube and the hypercubic derived networks, e.g., the shuffle-exchange graph, have efficient implementations for the Fast Fourier Transform, sorting algorithms, and many other useful applications. We will discuss the multi-chip implementation for some of these architectures in this thesis. In particular, we discuss the implementation for the deBruijn graph extensively in Chapter 4. We focus on this family of graphs because they represent the circuit diagrams for fully parallel Viterbi decoders. In fact, the binary deBruijn graph B_{K-2} (to be defined in Section 4.1) represents the fully parallel Viterbi decoder for a constraint length K, rate $1/n$ convolutional code; and NASA is using a $K = 15$, rate $1/4$ convolutional code on the Galileo mission.

For the second problem, the solution usually relies on the effectiveness on using the resources, i.e., the available chip area and the available number of pins per chip. With the recent advances in VLSI technologies, the transistor sizes have been greatly reduced and the die size, i.e., the available chip area, has simultaneously been largely increased. As a result, many processors can be built inside one single chip. Unfortunately, the slow progress in packaging technology has left us a bottleneck on the number of available pins. Thus many recent circuit designs are limited by the available number of pins and/or I/O capability. In this thesis, we attempt to find methods to take full advantage of the precious pins by minimizing the number

of external inter-chip connections. The exact problem, which we call the VLSI decomposition problem, is precisely defined in Chapter 2.

In Chapters 3–5, we reveal our attempts to solve the VLSI decomposition problem for the families of complete graphs, hypercubes, hyperplanes, meshes, deBruijn graphs, and shuffle-exchange graphs. For all these cases, we have found optimal or relatively good solutions. In Chapter 6, we present several general theorems and some general techniques that can be used to obtain bounds on the number of inter-chip connections for a given graph. These theorems and techniques can be applied to many other graphs that are not discussed in this thesis. Finally, in Chapter 7, we summarize all the important results that are presented in this thesis.

II. The VLSI Decomposition Problem

Let G be a graph that represents an interconnection network for a complex circuit, such as a high-speed parallel computer. The vertices of G correspond to arithmetic processors and the edges of G correspond to data paths connecting the processors. Furthermore, we use directed and undirected edges to represent simplex and duplex links respectively.

In modern VLSI technology, if the circuit is too large to fit on a single chip, it may be possible to build it by wiring together two or more appropriately designed chips. Each processor must then be placed on one of the chips, but the wires of the circuit may be either internal to the chips (intrachip wires) or external (interchip wires). Thus we are motivated to define a *C-chip VLSI decomposition* of a graph G as a collection of C vertex-disjoint subgraphs of G which together contain all of G's vertices, and a subset of its edges. The edges contained in the collection of subgraphs are called *internal* edges. Since we can reduce the total number of pins on the chips by including as many internal edges as possible in the decomposition, we define the *efficiency* of a C-chip VLSI decomposition of G into subgraphs $H_1, H_2, \ldots, H_C$ as

$$\text{eff}(H_1, H_2, \cdots, H_C \mapsto G) = \frac{\sum_{i=1}^{C} \text{E}(H_i)}{\text{E}(G)} \tag{2.1}$$

where $\text{E}(\mathcal{G})$ denotes the number of edges in graph $\mathcal{G}$. The number of vertices in the subgraph H_i is called the *size* (or the *chip size*) of H_i. Let k_i denote the size of H_i and $|G|$ denote the number of vertices in G. Then

$$k_1 + k_2 + \cdots + k_C = |G|.$$

An example of a three-chip VLSI decomposition for the deBruijn graph B_3 (to be defined in Section 4.1) into chip sizes 2, 3 and 3 with an efficiency of 0.625 is shown in Figure 2.1.

In a VLSI decomposition of G into subgraphs $H_1, H_2, \ldots, H_C$, it is normally desirable for the subgraphs H_i to be isomorphic since the cost of fabricating multiple

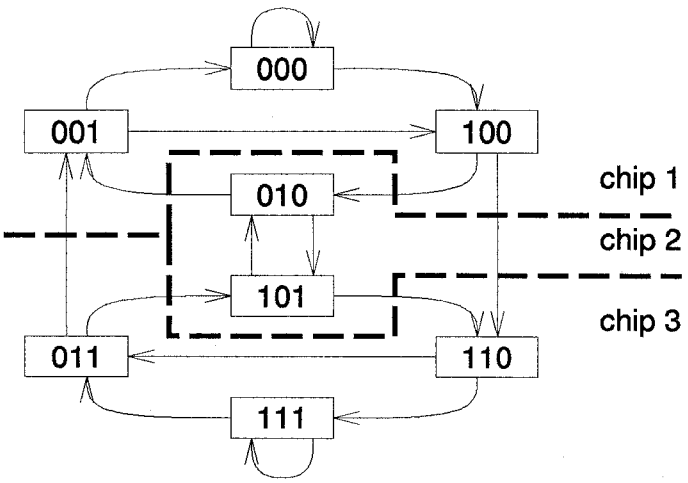

Figure 2.1. A three-chip VLSI decomposition for the de-Bruijn graph B_3 with an efficiency of 0.625.

copies of one chip is much less than designing several kinds of chips. For the case $H_1 \equiv H_2 \equiv \cdots \equiv H_C \equiv H$, we call one of these isomorphic subgraphs H a *building block* for the graph G. The size $k = |H|$ of a building block for G must be a factor of $|G|$. In addition, we use the shorthand notation $\text{eff}(H \vdash G)$ to denote the efficiency of the VLSI decomposition for G into building blocks H. Thus, by (2.1),

$$\text{eff}(H \vdash G) = \text{eff}(H, H, \cdots, H \mapsto G) = \frac{|G|}{|H|} \cdot \frac{\text{E}(H)}{\text{E}(G)}. \tag{2.2}$$

We also call $\text{eff}(H \vdash G)$ the *efficiency of the building block* H for the graph G. If a building block can be used to build any graph in a fixed set of graphs $\{G_n\}$, we call it a *universal building block* for $\{G_n\}$. The efficiency of a universal building block H for the set of graphs $\{G_n\}$ is usually a function of n, i.e.,

$$\text{eff}(H \vdash G_n) = \frac{|G_n|}{|H|} \cdot \frac{\text{E}(H)}{\text{E}(G_n)}. \tag{2.3}$$

However, there are special cases, such as the universal deBruijn building blocks (c.f. Section 4.8), that have an efficiency independent of n.

In this thesis, we consider the following VLSI decomposition problems:

[1] Given a graph G and chip sizes $k_1, k_2, \ldots, k_C$ (where $k_1 + k_2 + \cdots + k_C = |G|$), what are the most efficient VLSI decompositions for G into C chips of sizes $k_1, k_2, \ldots, k_C$?

[2] Given a graph G and a non-negative integer k (where k divides $|G|$), what are the most efficient VLSI decompositions for G into C chips (that are not required to be isomorphic) of equal sizes k?

[3] Given a graph G and a non-negative integer k (where k divides $|G|$), what are the most efficient building blocks of size k for G?

[4] Given a family of graphs $\{G_n\}_{n \geq N}$ and a non-negative integer k (where k divides $|G_n|$ for each $n \geq N$), what are the most efficient universal building blocks of size k for $\{G_n\}_{n \geq N}$?

A related problem, usually called the "pin limitation" problem, is to determine the maximum possible number $\mathrm{E}^*_G(k)$ of interconnecting edges in a subgraph of G with k vertices. If $\mathrm{E}^*_G(k)$ is known for a graph G, the efficiency for the C-chip VLSI decomposition of graph G into subgraphs $H_1(k_1), H_2(k_2), \cdots, H_C(k_C)$ of sizes $k_1, k_2, \ldots, k_C$ respectively is bounded above by

$$\text{(2.4)} \quad \mathrm{eff}(H_1(k_1), H_2(k_2), \cdots, H_C(k_C) \mapsto G) \leq \frac{\mathrm{E}^*_G(k_1) + \mathrm{E}^*_G(k_2) + \cdots + \mathrm{E}^*_G(k_C)}{\mathrm{E}(G)}.$$

In particular, for the case $k_1 = k_2 = \cdots = k_C = k$, we obtain

$$\text{(2.5)} \qquad e_{\mathrm{I}}(G; k) \leq e_{\mathrm{NI}}(G; k) \leq \frac{C \cdot \mathrm{E}^*_G(k)}{\mathrm{E}(G)},$$

where $e_{\mathrm{NI}}(G; k)$ denotes the efficiency of the most efficient VLSI decomposition into C chips (which are not required to be isomorphic) of equal sizes k and $e_{\mathrm{I}}(G; k)$ denotes the efficiency of the most efficient VLSI decomposition into C isomorphic chips (i.e., building blocks) of size k for the graph G.

In the following chapters, we reveal some efficient VLSI decompositions for several graph families. In some cases, (2.4) and (2.5) together with previous results on the studies of "pin limitations" [Harp64,Lind64,Snir81,Cyph90] provide proofs that the known VLSI decompositions are optimal.

III. Complete Graphs, Hypercubes, Hyperplanes, Meshes

3.1. An Easy Case: Complete Graphs K_n

The complete graph K_n consists of n vertices and $\binom{n}{2}$ edges, each connecting two distinct vertices. Figure 3.1 shows the complete graphs K_3, K_4, and K_5. Note that any induced subgraph† of K_n with m vertices (where $m \leq n$) is isomorphic to the complete graph K_m. Thus the most efficient C-chip VLSI decomposition of K_n into chip sizes $k_1, k_2, \ldots, k_C$ (where $k_1 + k_2 + \cdots + k_C = n$) consists of the C smaller complete graphs $K_{k_1}, K_{k_2}, \ldots, K_{k_C}$ with an efficiency of

$$\text{(3.1)} \qquad \text{eff}(K_{k_1}, K_{k_2}, \cdots, K_{k_C} \mapsto K_n) = \frac{\binom{k_1}{2} + \binom{k_2}{2} + \cdots + \binom{k_C}{2}}{\binom{n}{2}}.$$

An example of an optimal two-chip VLSI decomposition of K_9 into chip sizes 4 and 5 with an efficiency of 4/9 is shown in Figure 3.2.

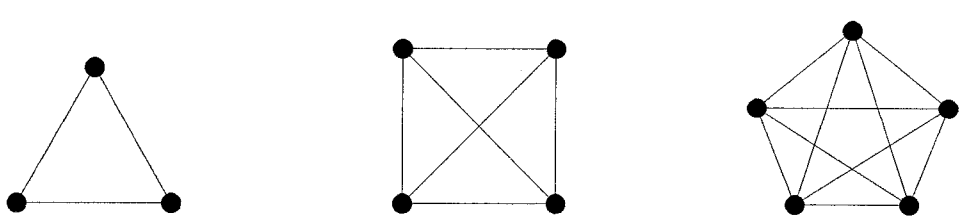

Figure 3.1. The complete graphs K_3, K_4 and K_5.

By letting $k_1 = k_2 = \cdots = k_C = k$ in (3.1), the most efficient VLSI decomposition of K_n into chips (which are not required to be isomorphic) of equal sizes k

† An induced subgraph H is a subgraph that contains all edges with both endvertices in H.

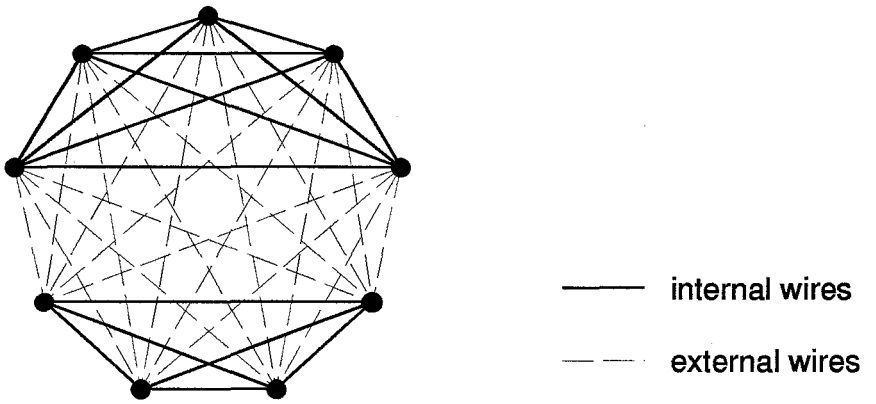

Figure 3.2. An optimal two-chip VLSI decomposition of K_9 into chip sizes 4 and 5.

(where $k|n$) consists of n/k complete graphs K_k with an efficiency of

$$e_{\text{NI}}(K_n; k) = \text{eff}(K_k, K_k, \cdots, K_k \mapsto K_n) = \frac{\binom{k}{2} \cdot \frac{n}{k}}{\binom{n}{2}} = \frac{k-1}{n-1}. \tag{3.2}$$

Consequently, the complete graph K_k is the most efficient K_n building block of size k with an efficiency of

$$e_{\text{I}}(K_n; k) = \text{eff}(K_k \vdash K_n) = \frac{\binom{k}{2} \cdot \frac{n}{k}}{\binom{n}{2}} = \frac{k-1}{n-1}. \tag{3.3}$$

Since (3.3) holds for all n that are multiples of k, the complete graph K_k is also the most efficient universal building block (of size k) for the family of complete graphs $\{K_n\}_{kN|n}$ where N is a positive integer.

3.2. Binary n-Cubes Γ_n

The binary n-cube Γ_n has 2^n vertices, labelled by the binary n-tuples $\{0,1\}^n$, and $n \cdot 2^{n-1}$ edges, each connecting two vertices with labels different in one position. Figure 3.3 shows the binary n-cubes Γ_1, Γ_2 and Γ_3.

Let $H(m)$ be the induced subgraph[†] containing m vertices of Γ_n that have labels equivalent to the binary expansion of the first m non-negative integers $0, 1, \ldots, m-1$. Harper[Harp64,Bern67,Hart76] showed that the subgraph $H(m)$ has the maximum number of edges among all subgraphs of Γ_n with m vertices (where $m \leq 2^n$). Consider a vertex of $H(m)$. Each "one" in the label indicates that the vertex with a "zero" in the same position is also in $H(m)$ and thus indicates the presence of an edge in $H(m)$. Therefore the total number of edges in $H(m)$ is

$$\mathrm{E}^*_{\Gamma_n}(m) = \mathrm{E}(H(m)) = \sum_{i=0}^{m-1} \mathrm{weight}(i) \tag{3.4}$$

where weight(i) denotes the number of "one"s in the binary expansion of i. From (3.4) and (2.4), the efficiency for a C-chip VLSI decomposition of Γ_n into chip sizes $k_1, k_2, \ldots, k_C$ (where $k_1 + k_2 + \cdots + k_C = 2^n$) is bounded above by

$$\frac{1}{\mathrm{E}(\Gamma_n)} \sum_{i=1}^{C} \left(\sum_{j=0}^{k_i - 1} \mathrm{weight}(i) \right). \tag{3.5}$$

For the case $k_1 = k_2 = \cdots = k_C = 2^k$, the expression (3.5) can be simplified to

$$e_{\mathrm{NI}}(\Gamma_n; 2^k) \leq \frac{1}{\mathrm{E}(\Gamma_n)} \sum_{i=1}^{C} \left(k \cdot 2^{k-1}\right) = \frac{(k)(2^{k-1})(2^{n-k})}{(n)(2^{n-1})} = \frac{k}{n}. \tag{3.6}$$

The upper bound (3.6) can be achieved if we decompose Γ_n into 2^{n-k} Γ_k-chips. We can show that such a decomposition is possible by construction, i.e., finding a one-to-one mapping between the vertex set V of Γ_n and the set of ordered pairs

[†] Induced subgraph is defined in the footnote on page 7.

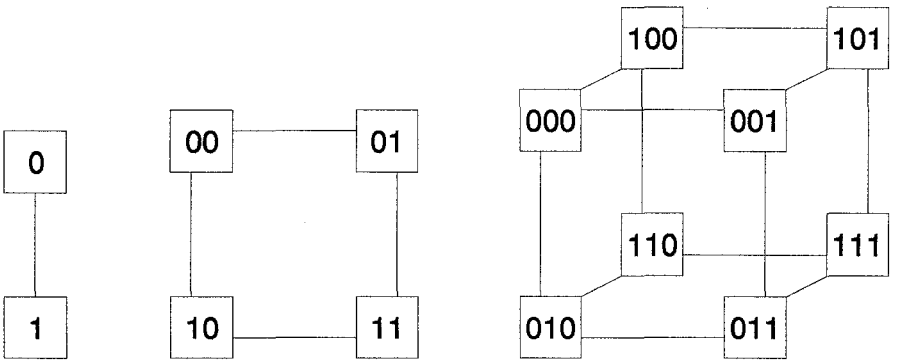

Figure 3.3. The binary n-cubes Γ_1, Γ_2 and Γ_3.

(i, j), where i denotes the chip number and j denotes a vertex inside chip i, such that if two vertices are connected inside a Γ_k-chip, then those two vertices are also connected in the graph Γ_n. Now let the 2^{n-k} Γ_k-chips be numbered in binary from $00\cdots0$ to $11\cdots1$. For each of these 2^{n-k} chips, we use conventional labels for its vertices, i.e., each vertex is labelled by a binary k-tuple and two vertices are connected iff their labels are different in exactly one position. We can map vertex $v_1v_2\cdots v_n$ of Γ_n onto the Γ_k-chip numbered $v_{k+1}v_{k+2}\cdots v_n$ at location $v_1v_2\cdots v_k$. Note that this one-to-one mapping satisfies the above mentioned requirement, i.e., if two vertices are connected inside a Γ_k-chip, then those two vertices are also connected in the graph Γ_n. Figure 3.4 shows an example of an optimal two-chip VLSI decomposition of Γ_4 into two Γ_3-chips by using the above mapping procedure.

By (3.6) and the argument above, the decomposition of Γ_n into 2^{n-k} binary k-cubes is an optimal $C = 2^{n-k}$ chip (of chip sizes $k_1 = k_2 = \cdots = k_C = 2^k$) VLSI decomposition. The efficiency is

$$(3.7) \qquad e_{\text{I}}(\Gamma_n; 2^k) = e_{\text{NI}}(\Gamma_n; 2^k) = \text{eff}(\Gamma_k \vdash \Gamma_n) = \frac{(k)(2^{k-1})(2^{n-k})}{(n)(2^{n-1})} = \frac{k}{n}.$$

Since (3.7) holds for all $n \geq k$, the binary k-cube Γ_k is also the most efficient

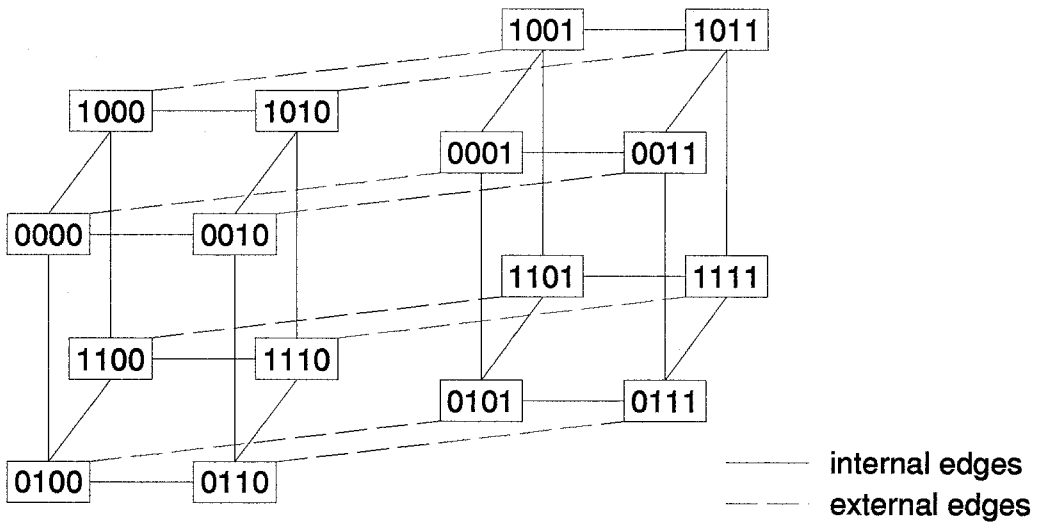

Figure 3.4. An optimal two-chip VLSI decomposition of Γ_4 into Γ_3-chips with an efficiency of 3/4.

universal building block for the family of binary n-cubes $\{\Gamma_n\}_{n \geq N}$ where $N \geq k$.

3.3. n-Dimensional Hyperplanes $\Gamma_n(l_1, l_2, \ldots, l_n)$

An n-dimensional hyperplane $\Gamma_n(l_1, l_2, \ldots, l_n)$ consists of vertices $v_1 v_2 \cdots v_n$, where $0 \leq v_i \leq l_i - 1$, and two vertices are connected iff their labels are different in one position. Thus there are $l_1 l_2 \cdots l_n$ vertices and $(l_1 + l_2 + \cdots + l_n - n)(l_1 l_2 \cdots l_n)/2$ edges in $\Gamma_n(l_1, l_2, \ldots, l_n)$. (Note that the binary n-cube is a special case of an n-dimensional hyperplane with $l_i = 2$ for $1 \leq i \leq n$.) Since the hyperplane $\Gamma_n(l_1, l_2, \ldots, l_n)$ is isomorphic to $\Gamma_n(l'_1, l'_2, \ldots, l'_n)$, where $l'_1, l'_2, \ldots, l'_n$ is a permutation of $l_1, l_2, \ldots, l_n$, we may assume

$$l_1 \geq l_2 \geq \cdots \geq l_n$$

without loss of generality.

For each vertex $v = v_1v_2\cdots v_n$ in $\Gamma_n(l_1, l_2, \ldots, l_n)$, we may assign a distinct non-negative integer $N(v)$ defined by

$$N(v) = N(v_1v_2\cdots v_n) = v_1 + l_1v_2 + l_1l_2v_3 + \cdots + l_1l_2\cdots l_{n-1}v_n.$$

Let $H(m)$ be the induced subgraph† of $\Gamma_n(l_1, l_2, \ldots, l_n)$ (where $l_1 \geq l_2 \geq \cdots \geq l_n$) with m vertices such that vertex v is in $H(m)$ iff $N(v) < m$. Lindsey [Lind64] generalized Harper's result [Harp64] and showed that the subgraph $H(m)$ has the maximum number of edges among all subgraphs of $\Gamma_n(l_1, l_2, \ldots, l_n)$ with m vertices (where $m \leq l_1l_2\cdots l_n$). Similar to the case of the binary n-cube, the number of edges in $H(m)$ is

$$\mathrm{E}^*_{\Gamma_n(l_1,l_2,\ldots,l_n)}(m) = \mathrm{E}(H(m)) = \sum_{i=0}^{m-1} \mathrm{weight}(i) \tag{3.8}$$

where weight(i) denotes the sum of the digits in $N^{-1}(i)$, i.e., if $N(v_1v_2\cdots v_n) = i$, then weight$(i) = v_1 + v_2 + \cdots + v_n$. By (3.8), the number of internal edges in a C-chip VLSI decomposition of $\Gamma_n(l_1, l_2, \ldots, l_n)$ with chip sizes $k_1, k_2, \ldots, k_C$ (where $k_1 + k_2 + \cdots + k_C = l_1l_2\cdots l_n$) is bounded above by

$$\sum_{i=1}^{C} \left(\sum_{j=0}^{k_i-1} \mathrm{weight}(i) \right). \tag{3.9}$$

For the case $k_1 = k_2 = \cdots = k_C = l_1l_2\cdots l_k$, the expression (3.9) can be simplified to

$$\sum_{i=1}^{C} \left(\frac{l_1 + l_2 + \cdots + l_k - k}{2} (l_1l_2\cdots l_k) \right) = \frac{l_1 + l_2 + \cdots + l_k - k}{2} (l_1l_2\cdots l_n).$$

This upper bound can be achieved if we decompose $\Gamma_n(l_1, l_2, \ldots, l_n)$ into k-dimensional sub-hyperplanes $\Gamma_k(l_1, l_2, \ldots, l_k)$ by mapping vertex $v_1v_2\cdots v_n$ of $\Gamma_n(l_1, l_2, \ldots, l_n)$ onto location $v_1v_2\cdots v_k$ in the chip numbered $v_{k+1}v_{k+2}\cdots v_n$. The

† Induced subgraph is defined in the footnote on page 7.

efficiency of such an optimal $C = l_{k+1}l_{k+2}\cdots l_n$ chip (of chip sizes $k_1 = k_2 = \cdots = k_C = l_1 l_2 \cdots l_k$) VLSI decomposition for $\Gamma_n(l_1, l_2, \ldots, l_n)$ is

$$\begin{aligned} & e_{\text{I}}(\Gamma_n(l_1, l_2, \ldots, l_n); l_1 l_2 \cdots l_k) = e_{\text{NI}}(\Gamma_n(l_1, l_2, \ldots, l_n); l_1 l_2 \cdots l_k) \\ & = \text{eff}(\Gamma_k(l_1, l_2, \ldots, l_k) \vdash \Gamma_n(l_1, l_2, \ldots, l_n)) = \frac{l_1 + l_2 + \cdots + l_k - k}{l_1 + l_2 + \cdots + l_n - n}. \end{aligned} \tag{3.10}$$

Since (3.10) holds for all $n \geq k$, the k-dimensional hyperplane $\Gamma_k(l_1, l_2, \ldots, l_k)$ (where $l_1 \geq l_2 \geq \cdots \geq l_k$) is also the most efficient universal building block for the family of hyperplanes $\{\Gamma_n(l_1, l_2, \ldots, l_n)\}_{n \geq N}$ where $N \geq k$ and $l_i \leq l_k$ for $k < i \leq n$.

As a final remark, if we consider only the hyperplanes $\Gamma_n(l)$ that denotes $\Gamma_n(l_1, l_2, \ldots, l_n)$ with $l_1 = l_2 = \cdots = l_n = l$, the optimal $C = l^{n-k}$ chip VLSI decomposition (with chip sizes $k_1 = k_2 = \cdots = k_C = l^k$) for $\Gamma_n(l)$ consists of l^{n-k} copies of $\Gamma_k(l)$. By letting $l_1 = l_2 = \cdots = l_k = l$ in (3.10), we obtain the efficiency as

$$e_{\text{I}}(\Gamma_n(l); l^k) = e_{\text{NI}}(\Gamma_n(l); l^k) = \text{eff}(\Gamma_k(l) \vdash \Gamma_n(l)) = \frac{kl - k}{nl - n} = \frac{k}{n}. \tag{3.11}$$

Again, since (3.11) holds for all $n \geq k$, the hyperplane $\Gamma_k(l)$ is also the most efficient universal building block for the family of hyperplanes $\{\Gamma_n(l)\}_{n \geq N}$ where $N \geq k$.

3.4. d-Dimensional Meshes $M'_d(n)$ Without Wrap-Around

A d-dimensional mesh of side length n has n^d vertices labelled by the n-ary d-tuples $\{0, 1, \cdots, n-1\}^d$. For the mesh $M'_d(n)$ without wrap-around, two vertices are connected by an edge iff their labels are different in one position and the difference of the two digits in that position is one. (The mesh $M_d(n)$ with wrap-around is defined in the next section.) Thus there are $(d)(n-1)(n^{d-1})$ edges in $M'_d(n)$. Note that a basic difference between meshes and hyperplanes is that we normally consider family of meshes $\{M'_d(n)\}_{n \geq k}$ that have different side lengths in contrast with family of hyperplanes $\{\Gamma_n(l)\}_{n \geq k}$ that have different dimensions. Figure 3.5 shows the two-dimensional meshes $M'_2(6)$ and $M_2(6)$.

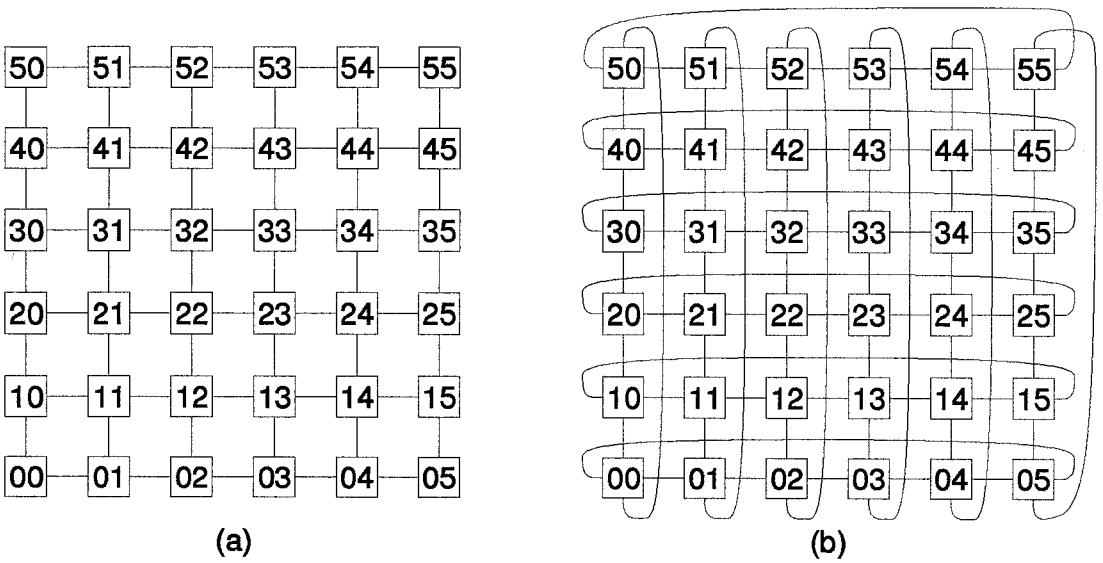

Figure 3.5. Two dimensional meshes: (a) $M'_2(6)$ without wrap-around and (b) $M_2(6)$ with wrap-around.

Let $H'_d(m)$ be a subgraph of $M'_d(n)$ with m vertices and let $V(H'_d(m))$ be the vertex set of $H'_d(m)$. Let $P_r(H'_d(m))$, where $1 \leq r \leq d$, be the set of $(d-1)$-tuples $(x_1, x_2, \cdots, x_{d-1})$ such that $(x_1, x_2, \cdots, x_{d-1}) \in P_r(H'_d(m))$ iff there exists $(x_1, x_2, \cdots, x_{r-1}, j, x_r, x_{r+1}, \cdots, x_{d-1}) \in V(H'_d(m))$ for some j. (Intuitively, we may consider $P_r(H'_d(m))$ as the $(d-1)$-dimensional projection of a d-dimensional object.) Figure 3.6 shows a subgraph $H'_2(14)$ with $P_1(H'_2(14)) = \{1,2,3,4\}$ and $P_2(H'_2(14)) = \{1,2,3,4,5\}$. Note that the number of edges that are in the rth dimension, i.e., those connecting vertices $(x_1, x_2, \cdots, x_{r-1}, j, x_r, x_{r+1}, \cdots, x_{d-1})$ and $(x_1, x_2, \cdots, x_{r-1}, j+1, x_r, x_{r+1}, \cdots, x_{d-1})$, in $H'_d(m)$ is $\leq m - |P_r(H'_d(m))|$ (where the notation $|R|$ denotes the number of elements in the set R). Thus the total number of edges in $H'_d(m)$ is bounded above by

$$\mathrm{E}(H'_d(m)) \leq dm - \sum_{r=1}^{d} |P_r(H'_d(m))|. \tag{3.12}$$

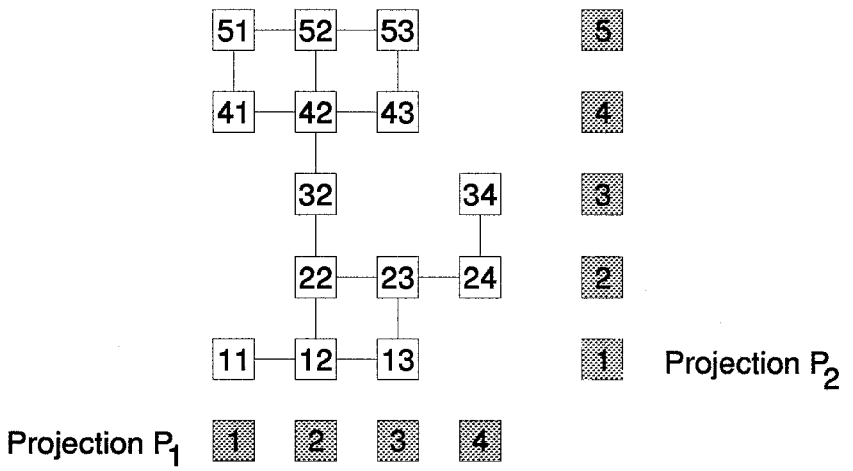

Figure 3.6. A subgraph $H_2'(14)$ of $M_2'(6)$. $P_1(H_2'(14)) = \{1,2,3,4\}$ and $P_2(H_2'(14)) = \{1,2,3,4,5\}$.

Cypher [Cyph90] proved that $\sum_{r=1}^{d} |P_r(H_d'(m))| \geq dm^{\frac{d-1}{d}}$. Thus, by (3.12), the maximum number of edges in the subgraph $H_d'(m)$ is bounded above by

$$\mathrm{E}^*_{M_d'(n)}(m) \leq d \cdot (m - m^{\frac{d-1}{d}}). \tag{3.13}$$

From (2.4) and (3.13), the efficiency for a C-chip VLSI decomposition of $M_d'(n)$ into chip sizes $k_1, k_2, \ldots, k_C$ (where $k_1 + k_2 + \cdots + k_C = n^d$) is bounded above by

$$\frac{1}{\mathrm{E}(M_d'(n))} \sum_{i=1}^{C} \left(d \cdot (k_i - k_i^{\frac{d-1}{d}})\right) = \frac{1}{(n-1)(n^d)} \sum_{i=1}^{C} \left(k_i - k_i^{\frac{d-1}{d}}\right). \tag{3.14}$$

For the case $k_1 = k_2 = \cdots = k_C = k^d$, the expression (3.14) can be simplified to

$$\begin{aligned} e_{\mathrm{NI}}(M_d'(n); k^d) &\leq \frac{1}{(n-1)(n^d)} \sum_{i=1}^{C} (k^d - k^{d-1}) \\ &= \frac{(\frac{n}{k})^d (k^d - k^{d-1})}{(n-1)(n^d)} = \frac{(n)(k-1)}{(n-1)(k)}. \end{aligned} \tag{3.15}$$

The upper bound (3.15) can be achieved if we decompose $M_d'(n)$ into $(n/k)^d$ $M_d'(k)$-chips by mapping vertex $(v_1, v_2, \cdots, v_d)$ of $M_d'(n)$ onto position $(r_1, r_2, \cdots, r_d)$ in the

chip numbered $(q_1, q_2, \cdots, q_d)$ where q_i and r_i are the quotient and the remainder respectively when v_i is divided by k. Thus the decomposition of $M'_d(n)$ into $(n/k)^d$ d-dimensional $M'_d(k)$ meshes is an optimal $C = (n/k)^d$ chip (of chip sizes $k_1 = k_2 = \cdots = k_C = k^d$) VLSI decomposition with an efficiency of

$$\begin{aligned} e_{\text{I}}(M'_d(n); k^d) &= e_{\text{NI}}(M'_d(n); k^d) \\ &= \text{eff}(M'_d(k) \vdash M'_d(n)) = \frac{(n/k)^d (d)(k^{d-1})(k-1)}{(d)(n^{d-1})(n-1)} = \frac{(n)(k-1)}{(n-1)(k)}. \end{aligned} \tag{3.16}$$

Since (3.16) holds for all n that is a multiple of k, the mesh $M'_d(k)$ is also the most efficient universal building block for the family of meshes $\{M'_d(n)\}_{kN|n}$ where N is a positive integer.

3.5. d-Dimensional Meshes $M_d(n)$ With Wrap-Around

A d-dimensional mesh $M_d(n)$ (with wrap-around) of side length n has n^d vertices labelled by the n-ary d-tuples $\{0, 1, \cdots, n-1\}^d$. Two vertices are connected by an edge iff their labels are different in one position and the difference of the two digits in that position is 1 mod n. Thus there are dn^d edges in $M_d(n)$. Figure 3.5(b) shows the mesh $M_2(6)$.

By using the same argument introduced by Cypher [Cyph90], we can prove that for meshes $M_d(n)$,

$$\text{E}^*_{M_d(n)}(m) \leq \frac{n+1}{n} \cdot dm - dm^{\frac{d-1}{d}}. \tag{3.17}$$

Thus the efficiencies for $M_d(n)$ building blocks of size k^d (where $k|n$) are bounded above by

$$\begin{aligned} e_{\text{I}}(M_d(n); k^d) &\leq e_{\text{NI}}(M_d(n); k^d) \\ &\leq \left(\frac{n+1}{n} \cdot dk^d - d \cdot (k^d)^{\frac{d-1}{d}} \right) \cdot \frac{\left(\frac{n}{k}\right)^d}{dn^d} = \frac{k-1}{k} + \frac{1}{n}. \end{aligned} \tag{3.18}$$

The upper bound (3.18) can almost be achieved if we decompose $M_d(n)$ into $(n/k)^d$ $M'_d(k)$ meshes (without wrap-around) by mapping vertex $v_1 v_2 \cdots v_d$ of $M_d(n)$ onto

position $r_1 r_2 \cdots r_d$ in the chip numbered $q_1 q_2 \cdots q_d$, where q_i and r_i are the quotient and the remainder respectively when v_i is divided by k. The efficiency of this decomposition is

$$\text{(3.19)} \qquad \text{eff}(M'_d(k) \vdash M_d(n)) = \frac{(d)(k-1)(k^{d-1})(\frac{n}{k})^d}{dn^d} = \frac{k-1}{k}.$$

By (3.18), the efficiency for a universal $M_d(n)$ building block of size k^d is bounded above by $\frac{k-1}{k}$ since a universal $M_d(n)$ building block is necessary to build $M_d(n)$ with arbitrary large n. Thus the $M'_d(k)$ mesh is the most efficient universal building block of size k^d for the family of meshes $\{M_d(n)\}_{kN|n}$ where N is a positive integer.

IV. DeBruijn Graphs

4.1. Binary DeBruijn Graphs B_n

The binary deBruijn graph B_n can be defined as a directed graph with 2^n vertices, each represented with a binary n-tuple, and 2^{n+1} edges, each labelled with a binary $(n+1)$-tuple, where the edge $x_1x_2\cdots x_{n+1}$ is a directed edge from $x_2x_3\cdots x_{n+1}$ to $x_1x_2\cdots x_n$. Figure 4.1 shows the binary deBruijn graphs B_2 and B_3.

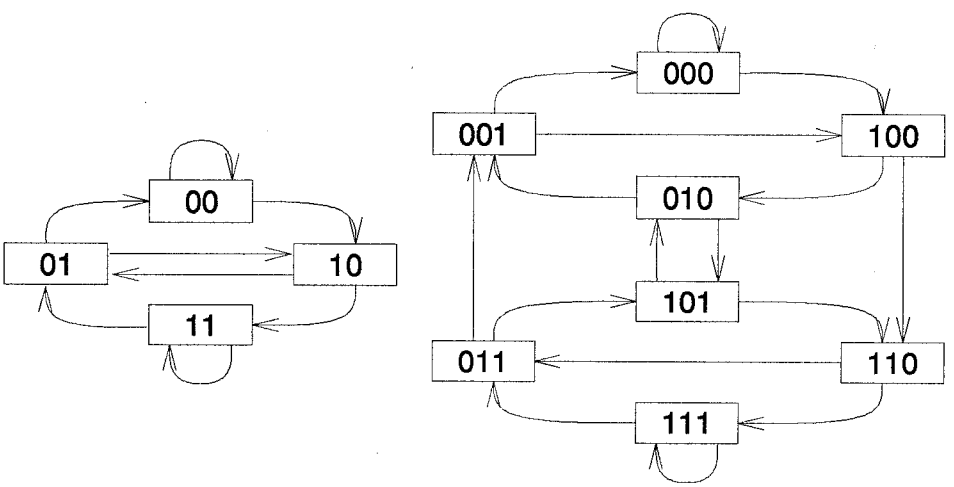

Figure 4.1. The deBruijn graphs B_2 (or B_2^2) and B_3 (or B_3^2).

Some necessary conditions and a general construction for efficient VLSI decompositions for binary deBruijn graphs are discussed in [Doli92a,Doli92b]. These results can directly be generalized for q-ary deBruijn graphs B_n^q. This chapter considers VLSI decompositions for deBruijn graphs B_n^q with emphasis on binary deBruijn graphs B_n which have practical significance in building fully parallel Viterbi

decoders. In addition, the binary deBruijn graphs B_n have strong resemblance with the shuffle-exchange graphs Ψ_n as to be shown in Chapter 5.

4.2. q-ary DeBruijn Graphs B_n^q

The q-ary deBruijn graph B_n^q can be defined as a directed graph with q^n vertices, each represented with a q-ary n-tuple $\{0, 1, \cdots, q-1\}^n$, and q^{n+1} edges, each labelled with a q-ary $(n+1)$-tuple $\{0, 1, \cdots, q-1\}^{n+1}$, where the edge $x_1 x_2 \cdots x_{n+1}$ is a directed edge from $x_2 x_3 \cdots x_{n+1}$ to $x_1 x_2 \cdots x_n$. Figures 4.1 and 4.2 show the deBruijn graphs B_2^2 (or B_2), B_3^2 (or B_3), and B_2^3.

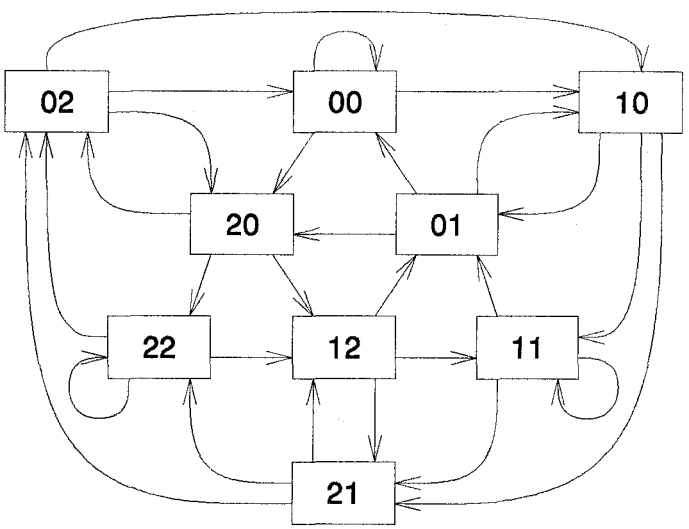

Figure 4.2. The deBruijn graph B_2^3.

In the next section, we derive some properties of deBruijn graphs which are needed to study the VLSI decomposition for these graphs. (Other important properties can be found in [Golo82: Secs. 2.2 and 6.2].)

4.3. Properties of DeBruijn Graphs B_n^q

We first define some terms for a graph G that may contain both directed and undirected edges. These terms are defined in a general way so that the same terminology may be used on the shuffle-exchange graphs in Chapter 5. Let $P = (P_1, P_2, \cdots, P_{l+1})$ be a sequence of $l+1$ vertices, and $E = (E_1, E_2, \cdots, E_l)$ be a sequence of l edges (directed or undirected), in a graph G. If, for $i = 1, 2, \ldots, l$, E_i is an edge joining P_i and P_{i+1} (in either direction if E_i is a directed edge), we call (P, E) an *ambulance path* (in resemblance with the fact that an ambulance need not to obey the street directions during emergency) of *length* l from P_1 to P_{l+1}.

By tracing along the ambulance path (P, E) in the order $P_1, P_2, \ldots, P_{l+1}$, its edge set E can be divided into three classes—undirected edges, forward edges (directed edges in the same direction as the trace) and backward edges (directed edges in the reverse direction). If there are f forward edges and b backward edges, we define the *net length* of the ambulance path to be $|f - b|$ and the *signed net length* to be $f - b$.

A *normal path* (or simply called *path*) is an ambulance path with no backward edge. A *normal trail* (or *trail*) is a normal path with distinct edges. A *cycle* is an ambulance path such that $P_1, P_2, \ldots, P_l$ are all distinct, but $P_{l+1} = P_1$. A cycle in which the number of forward edges equals the number of backward edges is called a *balanced* cycle. A cycle which is not balanced is called an *unbalanced* cycle. We also call a cycle with r forward edges and s backward edges an (r, s) *cycle*. Figure 4.3 shows an unbalanced $(3, 2)$ (or $(2, 3)$ depending on the direction of the trace) cycle of length 5 in B_3.

Theorem 4.1. If X and Y are vertices in B_n^q, and l is a positive integer with $l \leq n$, then there cannot be more than one path of length l from X to Y.

Proof: Let $X = x_1 x_2 \cdots x_n$. In a path from X to Y, the vertex immediately

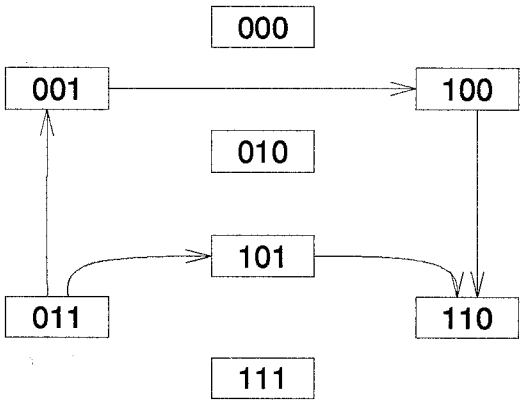

Figure 4.3. An unbalanced $(3,2)$ cycle of length 5 in B_3. The net length is 1.

following X must be $y_1x_1x_2\cdots x_{n-1}$ for some $y_1 \in \{0,1,\cdots,q-1\}$. The next vertex must then be $y_2y_1x_1\cdots x_{n-2}$, and so on. Thus Y, which is the lth vertex in the path, must be of the form $Y = y_ly_{l-1}\cdots y_1x_1x_2\cdots x_{n-l}$ (where y_1 has not been shifted out since $l \leq n$). It follows that each vertex on the path, and hence the path itself, is uniquely determined by X and Y. ∎

The following theorem shows that the number of vertex-disjoint unbalanced (r,s) cycles in B_n^q is bounded above by a number independent of n.

Theorem 4.2. In B_n^q, there are at most q^l/l vertex-disjoint unbalanced (r,s) cycles, where $l = r+s$ is the length of the cycle.

Proof: If $l \geq n$, the theorem is trivially true, since B_n^q has q^n vertices and thus at most $q^n/l \leq q^l/l$ vertex-disjoint unbalanced cycles of length l. On the other hand, if $l < n$, let $A = a_1a_2\cdots a_n$ be a vertex which lies on an unbalanced (r,s) cycle. Since a cycle may be traversed in either of two opposite directions, we may assume $r > s$ without loss of generality. After traversing the cycle and returning back to

A, the substring $a_{s+1}a_{s+2}\cdots a_{n-r}$ will have been shifted to the right $r-s$ positions and so we have

$$A = x_1x_2\cdots x_ra_{s+1}a_{s+2}\cdots a_{n-r}y_1y_2\cdots y_s.$$

Since this expression must be the same as the original label $a_1a_2\cdots a_n$, by equating the corresponding bits in positions $r+1$ through $n-s$, we obtain the equations

$$a_{r+i} = a_{s+i} \qquad (1 \le i \le n-s-r). \tag{4.1}$$

It follows from (4.1) that A is uniquely determined by the l values $a_1, a_2, \ldots, a_r$ and $a_{n-s+1}, a_{n-s+2}, \ldots, a_n$, so that there are at most $q^{r+s} = q^l$ vertices that can lie on an unbalanced (r,s) cycle. Since each (r,s) cycle contains exactly l vertices, it follows that a set of vertex-disjoint unbalanced (r,s) cycles contain at most q^l/l members. ∎

Let S_n^q be the set of all q-ary strings X of length n such that one of the longest run of zeros in X is either at the extreme left or the extreme right. For example, $0020100, 0012003, 1234123 \in S_7^5$, but $0100230 \notin S_7^5$. The following theorem shows that the number of edge-disjoint (vertex-disjoint) unbalanced cycles in B_n^q is bounded above by the number of elements in S_{n+1}^q (S_n^q). In addition, we may remove all unbalanced cycles in B_n^q by deleting all edges $E \in S_{n+1}^q$ or all vertices $V \in S_n^q$.

Theorem 4.3. In an unbalanced cycle in B_n^q, (i) there exists a vertex $V \in S_n^q$ and (ii) there exists a directed edge $E \in S_{n+1}^q$.

Proof: Let $P_1 = a_1a_2\cdots a_n$ be a vertex on an unbalanced (r,s) cycle and we assume $r > s$ without loss of generality. Let $P_1, E_1, P_2, E_2, \cdots, P_l, E_l, P_{l+1}$ (where $P_{l+1} = P_1$ and $l = r+s$) be the sequence of vertices and edges if we trace along the cycle starting from vertex P_1 and back to $P_1 = P_{l+1}$. (The directed edge E_i may

connect the vertices P_i and P_{i+1} in either direction.) Let

$$\begin{aligned} r' &= \max_{1\le i\le l+1} d(P_1, P_i) \\ s' &= \min_{1\le i\le l+1} d(P_1, P_i) \end{aligned} \tag{4.2}$$

where $d(P_1, P_i)$ denotes the signed net length of the ambulance path $P_1P_2\cdots P_i$. Then $a_{s'+1}a_{s'+2}\cdots a_{n-r'}$ is the longest substring in $P_1 = a_1a_2\cdots a_n$ that is also a substring in any other vertices P_i, $2 \le i \le l+1$, i.e., the substring $a_{s'+1}a_{s'+2}\cdots a_{n-r'}$ is shifted back and forth but never get shifted out when we trace along the cycle.

We first prove part (i) of the theorem. Suppose one of the longest run of zeros in $P_1 = a_1a_2\cdots a_n$ is the substring $a_ua_{u+1}\cdots a_v$. If $u = 1$ or $v = n$ or P_1 does not contain any zero, then $P_1 \in S_n^q$ and we are done. Otherwise the next vertex P_2 on the unbalanced cycle has the label $xa_1a_2\cdots a_{n-1}$ (if E_1 is a forward edge) or $a_2a_3\cdots a_ny$ (if E_1 is a backward edge). In both cases, either $P_2 \in S_n^q$ or the substring $a_ua_{u+1}\cdots a_v$ remains to be one of the longest run of zeros in P_2. Continuing this way, if $P_i \notin S_n^q$ $\forall 1 \le i \le j$, then either $P_{j+1} \in S_n^q$ or the substring $a_ua_{u+1}\cdots a_v$ is one of the longest run of zeros in P_{j+1}. Now we consider three cases: (a) $v \ge n - r'$, (b) $u \le s' + 1$, and (c) $s' + 1 < u \le v < n - r'$.

(a) For the case $v \ge n - r'$, let v' be the smallest positive integer such that $d(P_1, P_{v'}) = n - v$, i.e.,

$$n - v = d(P_1, P_{v'}) > d(P_1, P_i) \ \ \forall i < v'.$$

If $P_i \notin S_n^q$ $\forall 1 \le i < v'$, then $P_{v'} \in S_n^q$ since the substring $a_ua_{u+1}\cdots a_v$ is a suffix of $P_{v'}$.

(b) For the case $u \le s' + 1$, let u' be the smallest positive integer such that $d(P_1, P_{u'}) = u$, i.e.,

$$u = d(P_1, P_{u'}) < d(P_1, P_i) \ \ \forall i < u'.$$

If $P_i \notin S_n^q \ \forall 1 \leq i < u'$, then $P_{u'} \in S_n^q$ since the substring $a_u a_{u+1} \cdots a_v$ is a prefix of $P_{u'}$.

(c) For the remaining case $s' + 1 < u \leq v < n - r'$, we will show that there exists another all zeros substring $a_{t-u+v} \cdots a_t$ (which is also one of the longest run of zeros) in P_1 where $t \geq n - r'$ so that case (c) reduces to case (a). First note that vertex P_{l+1} has the form

$$P_{l+1} = x_1 x_2 \cdots x_{s'+(r-s)} a_{s'+1} a_{s'+2} \cdots a_{n-r'} y_1 y_2 \cdots y_{r'-(r-s)}.$$

By equating the corresponding bits of P_{l+1} and $P_1 = a_1 a_2 \cdots a_n$, we obtain the equations

$$a_{r-s+i} = a_i \qquad (s' + 1 \leq i \leq n - r'). \tag{4.3}$$

It follows from (4.3) and $s' + 1 < u \leq v < n - r'$ that the substring $a_{r-s+u} \cdots a_{r-s+v} = a_u \cdots a_v$ is also an all zeros substring of length $v - u + 1$. Continuing this way, the substring $a_{m(r-s)+u} \cdots a_{m(r-s)+v}$ is an all zeros substring if $(m-1)(r-s) + v \leq n - r'$. By letting $t = m(r-s) + v$ and

$$m - 1 = \left[\frac{n - r' - v}{r - s} \right] \geq \frac{n - r' - v}{r - s} - 1$$

where [ ] denotes the greatest integer function, we obtain

$$t = m(r-s) + v \geq \left(\frac{n - r' - v}{r - s} \right) (r - s) + v = n - r'$$

which is what we set out to show.

To prove (ii), we use the result in part (i) which asserts the existence of a vertex $P_i \in S_n^q$. Then at least three of the four directed edges $P_i 0$, $P_i 1$, $0P_i$, and $1P_i$ have the longest run of zeros either at the extreme left or the extreme right. Thus at least one of the two distinct edges E_{i-1} or E_i has its longest run of zeros either at the extreme left or the extreme right. ∎

Theorem 4.4. (i) In an ambulance path of net length $\geq n-2$ in B_n^q, there exists a vertex $V \in S_n^q$. (ii) In an ambulance path of net length $\geq n$ in B_n^q, there exists a directed edge $E \in S_{n+1}^q$.

Proof: Let $P_1 = a_1 a_2 \cdots a_n$ and let $P_1, E_1, P_2, E_2, \cdots, P_l, E_l, P_{l+1}$ be the sequence of vertices and edges if we trace along an ambulance path from vertex P_1 to P_{l+1}. Let $d(P_i, P_j)$ denote the signed net length of the ambulance path from P_i to P_j. Without loss of generality, we assume $d(P_1, P_{l+1})$ to be non-negative.

Suppose one of the longest run of zeros in $P_1 = a_1 a_2 \cdots a_n$ is the substring $a_u a_{u+1} \cdots a_v$. If $u = 1$ or $v = n$ or P_1 does not contain any zero, then $P_1 \in S_n^q$ and we are done. Thus we only need to consider the case $2 \leq u \leq v \leq n-1$. If $d(P_1, P_{l+1}) \geq n-2$, there must exist a positive integer v' such that v' is the smallest and $d(P_1, P_{v'}) = n - v$ since $n - v \leq n - 2 \leq d(P_1, P_{l+1})$. If $P_i \notin S_n^q \ \forall 1 \leq i < v'$, then $P_{v'} \in S_n^q$ since the substring $a_u a_{u+1} \cdots a_v$ is a suffix of $P_{v'}$.

To prove part (ii), note that if $d(P_1, P_{l+1}) \geq n$, then $d(P_2, P_l) \geq n-2$. By the result in part (i), there exists a vertex P_i $(2 \leq i \leq l)$ that is in the set S_n^q. Then at least one of the two directed edges E_{i-1} and E_i (which are in the form $P_i 0$, $P_i 1$, $0P_i$, or $1P_i$) has its longest run of zeros either at the extreme left or the extreme right. ∎

4.4. C-Chip VLSI Decompositions for DeBruijn Graphs B_n^q

In Chapter 6[†], there are some general theorems that can be used to obtain upper bounds on the efficiency for some VLSI decompositions. In particular, Theorem 6.3 together with Theorem 4.1 give a lower bound on the number of input edges w (defined in Section 6.1) for any subgraph of B_n^q.

Let $H(m)$ be an induced subgraph[‡‡] of B_n^q with m vertices. Let E_{ext} be the set of edges e_i in B_n^q such that e_i has one endvertex in $H(m)$ and the other endvertex not in $H(m)$. Let $H_{\text{ext}}(m)$ be the union of $H(m)$ and E_{ext}. (As in Section 6.1, we call $H_{\text{ext}}(m)$ a subgraph with external edges.) By letting $d_i = q$ for all the m vertices in the vertex set V of $H_{\text{ext}}(m)$ in Theorem 6.3, we obtain

$$(l+1)\sum_{v_i \in V}(q\log q) \le (wl + \sum_{v_i \in V} q)\log(wl + \sum_{v_i \in V} q) \qquad \forall l \le n$$

or

$$(l+1)(qm) \le (wl+qm)\log_q(wl+qm) \qquad \forall l \le n. \tag{4.4}$$

Since $(wl+qm)\log_q(wl+qm)$ is a monotonic increasing function of w, there exists an integer $w_{\min}(q,m,l)$ which is the smallest among all integers w that satisfy Inequality (4.4) for a given q, m and l. By noting that (4.4) holds for all $l \le n$, we obtain the lower bound

$$w \ge w_{\text{lowerbound}}(q,m,n) = \max_{l \le n} w_{\min}(q,m,l) \tag{4.5}$$

on the number of input edges on any subgraph $H_{\text{ext}}(m)$ of B_n^q. Consequently, since each vertex in $H_{\text{ext}}(m)$ has in-degree q, the number of edges in any subgraph $H(m)$ of B_n^q is bounded above by

$$\mathrm{E}^*_{B_n^q}(m) \le qm - w_{\text{lowerbound}}(q,m,n). \tag{4.6}$$

---

† Most of Chapter 6 does not use results from other chapters and can be read independently.

‡‡ Induced subgraph is defined in the footnote on page 7.

It follows from (2.4) and (4.6) that the efficiency for a C-chip VLSI decomposition of deBruijn graph B_n^q into subgraphs $H_1(k_1), H_2(k_2), \cdots, H_C(k_C)$ of sizes $k_1, k_2, \ldots, k_C$ respectively is bounded above by

$$(4.7)\quad \text{eff}(H_1(k_1), H_2(k_2), \cdots, H_C(k_C) \mapsto B_n^q) \leq 1 - \frac{1}{q^{n+1}} \sum_{1 \leq i \leq C} w_{\text{lowerbound}}(q, k_i, n).$$

m	n	$w_{\text{lowerbound}}(q{=}2, m, n)$
1	$n \geq 1$	1
2	$n \geq 2$	1
3	$n \geq 8$	2
4	$n \geq 6$	2
$5-7$	$n \geq 5$	2
8	$n \geq 15$	3
16	$n \geq 12$	4
32	$n \geq 25$	7
64	$n \geq 36$	12
128	$n \geq 45$	21
256	$n \geq 59$	38
512	$n \geq 81$	70

Table 4.1. Some values of $w_{\text{lowerbound}}(q, m, n)$ for $q = 2$ and n sufficiently large. $w_{\text{lowerbound}}(q, m, n)$ is defined in Equation (4.5).

Table 4.1 lists some values of $w_{\text{lowerbound}}(q, m, n)$ for $q = 2$ and n is sufficiently large. Table 4.2 lists some other values of $w_{\text{lowerbound}}(q, m, n)$ for $q = 2$ and $m = 128$. As an example, refering to Table 4.2, $w_{\text{lowerbound}}(2, 128, 13) = 13$. Thus if we

decompose B_{13} into 64 chips of equal sizes 128, there will be at least $13 \times 64 = 832$ external wires or efficiency ≤ 0.949 as imposed by the upper bound in (4.7). From this example, (4.7) does not seem to be a tight upper bound (since the best decomposition we are able to find has an efficiency far less than 0.949). However, we will see that (4.7) is a rather good upper bound for large values of m and n when we study the asymptotic behavior of $w_{\text{lowerbound}}(q, m, n)$ as $m \to \infty$ and compare the result with Theorem 4.12.

n	$w_{\text{lowerbound}}(q=2, m=128, n)$
8	4
9	6
10	8
11	10
12	11
$13 \leq n \leq 14$	13
15	14
16	15
$17 \leq n \leq 18$	16
$19 \leq n \leq 21$	17
$22 \leq n \leq 25$	18
$26 \leq n \leq 31$	19
$32 \leq n \leq 44$	20
$45 \leq n$	21

Table 4.2. Some values of $w_{\text{lowerbound}}(q, m, n)$ for $q = 2$ and $m = 128$. $w_{\text{lowerbound}}(q, m, n)$ is defined in Equation (4.5).

Lemma 4.5. Let X and Y be positive integers such that $X \geq b$ and $Y \geq 1$. If $X \leq Y \log_b Y$, then $Y \geq X/(\log_b X)$.

Proof: We have

$$\frac{X}{\log_b X} \log_b \left(\frac{X}{\log_b X} \right) = \frac{X}{\log_b X} (\log_b X - \log_b \log_b X)$$
$$\leq \frac{X}{\log_b X} (\log_b X) = X \leq Y \log_b Y$$

and $Y \log_b Y$ is a monotonic increasing function for $Y \geq 1$. ∎

We may apply Lemma 4.5 on Inequality (4.4) to obtain

$$w_{\text{lowerbound}}(q,m,n) \geq \frac{(l+1)(qm)}{l \log_q ((l+1)(qm))} - \frac{qm}{l}$$
$$= \frac{(qm)(l+1-\log_q(l+1)-\log_q(qm))}{(l)(\log_q(l+1)+\log_q(qm))} \qquad \forall l \leq n$$

or

$$w_{\text{lowerbound}}(q,m,n) \geq \max_{l \leq n} \frac{(qm)(l+1-\log_q(l+1)-\log_q(qm))}{(l)(\log_q(l+1)+\log_q(qm))}. \tag{4.8}$$

For $n \geq (\log_q m)^{1+\alpha}$ where α is a constant > 0, we may let $l = (\log_q m)^{1+\alpha}$ in (4.8) to obtain

$$w_{\text{lowerbound}}(q,m,n) \geq \frac{(qm)((\log_q m)^{1+\alpha}+1-\log_q((\log_q m)^{1+\alpha}+1)-\log_q(qm))}{((\log_q m)^{1+\alpha})(\log_q((\log_q m)^{1+\alpha}+1)+\log_q(qm))}$$
$$\asymp \frac{qm}{\log_q m} \qquad \text{as} \quad m \to \infty. \tag{4.9}$$

That is, if a large deBruijn graph B_n^q is decomposed into sufficiently small chips of sizes $k_1, k_2, \ldots, k_C$, then there are at least

$$\sum_{i=1}^{C} \frac{qk_i}{\log_q k_i}$$

external wires.

4.5. DeBruijn Building Blocks

From Theorems 4.1 and 4.2, we obtain two necessary conditions for deBruijn building blocks.

Theorem 4.6. If a graph H of size q^k/C is a building block for the deBruijn graph B_n^q, then

(i) For any two vertices X and Y in H, there cannot be more than one path of the same length $l \leq n$ from X to Y.

(ii) H does not contain any unbalanced cycle of length $l < \log_q C$.

Proof: Part (i) follows immediately from Theorem 4.1 since H is a subgraph of B_n^q. To prove (ii), note that Theorem 4.2 implies that $C \leq q^l/l \leq q^l$. ∎

We may use the results in Section 4.4 to obtain an upper bound for the efficiency of deBruijn building blocks. By letting $k_1 = k_2 = \cdots = k_C = q^k$ in Equation (4.7), the efficiency for a building block of size q^k for the deBruijn graph B_n^q is bounded above by

$$e_{\mathrm{I}}(B_n^q; q^k) \leq e_{\mathrm{NI}}(B_n^q; q^k) \leq 1 - \frac{w_{\text{lowerbound}}(q, q^k, n)}{q^{k+1}} \tag{4.10}$$

where

$$\liminf_{k \to \infty} w_{\text{lowerbound}}(q, q^k, n) \geq \frac{q^{k+1}}{k} \tag{4.11}$$

for $n \geq k^{1+\alpha}$ (where $\alpha > 0$) as implied by (4.9). That is, if a large deBruijn graph B_n^q is decomposed into sufficiently small subgraphs of equal sizes q^k, then the set of external edges will contain at least $1/k$ of the edges in B_n^q.

4.6. Universal DeBruijn Building Blocks

We consider *universal deBruijn building blocks* for the family of deBruijn graphs $\{B_n^q\}_{n \geq N}$. Since a universal deBruijn building block is necessary to build deBruijn graphs B_n^q with arbitrary large n, it follows from Theorem 4.6 that

(i) A universal deBruijn building block cannot contain two vertices X and Y such that there are two paths of the same length from X to Y.

(ii) A universal deBruijn building block cannot contain any unbalanced cycle.

Condition (ii) is equivalent to the statement that a universal deBruijn building block must be a *graded* digraph. In the following, we first introduce the notion of a graded digraph and then prove the equivalence between digraphs with no unbalanced cycle and graded digraphs in Lemma 4.7.

A digraph G with vertex set V is graded of rank m if there is a mapping ρ from the vertex set V to the set $\{0, 1, \ldots, m\}$, such that, for $x, y \in V$, $\rho(y) = \rho(x) + 1$ if there is a directed edge from x to y. We call $\rho(x)$ the *rank* of x.

Lemma 4.7. Let H be a digraph. H is a graded digraph of rank m iff all cycles in H are balanced and all ambulance paths in H have net length $\leq m$.

Proof: We will first prove the "if" portion and then the "only if" portion of the lemma.

$\Longrightarrow$ Suppose all cycles in H are balanced and all ambulance paths in H have net length $\leq m$. Let V be the vertex set of digraph H. We will prove that H is a graded digraph of rank m by constructing a rank function ρ that maps the vertex set V to the set $\{0, 1, \ldots, m\}$. Without loss of generality, H is assumed to be a connected graph. Otherwise, if H is the union of two disjoint subgraphs H_1 and H_2 whose rank functions (defined in this proof) are ρ_1 and ρ_2 respectively, then, for each vertex v in graph H, we may let $\rho(v)$ to be $\rho_1(v)$ (or $\rho_2(v)$) if v is in

the subgraph H_1 (or H_2).

To construct ρ, we first choose an arbitrary vertex $v_0 \in V$ and assign an arbitrary integer $N(v_0)$ to this vertex. For each of the other vertices $v \in V$ ($v \neq v_0$), there must be at least one ambulance path from v_0 to v since H is connected. If there is an ambulance path with f forward edges and b backward edges from v_0 to v, we assign the integer

$$N(v) = N(v_0) + f - b$$

to vertex v. If there are two ambulance paths, say P_1 and P_2, from v_0 to v, we obtain the same assignment $N(v)$ independent of the path P_1 or P_2 that is used. Such a consistency exists since all cycles in H are balanced by hypothesis and thus the cycle that is composed of P_1 and P_2 (P_2 being traced from v to v_0) must also be balanced.

Now let $N_{\min}$ (which may be negative) be the smallest integer in the set $\{N(v),\ v \in V\}$. (A minimum exists since there is a finite number of elements in the set $\{N(v),\ v \in V\}$.) For each $v \in V$, we assign the rank

$$\rho(v) = N(v) - N_{\min}.$$

To show that the mapping ρ is indeed a rank function, we need to prove that $\rho(w) = \rho(v) + 1$ if there is a directed edge from v to w. This can be shown by considering the cycle composed by path P_v (an ambulance path from v_0 to v), the directed edge from v to w, and path P'_w (an ambulance path from w to v_0). Let P_v (P'_w) contains f_v (f'_w) forward edges and b_v (b'_w) backward edges. Since all cycles in graph H are balanced by hypothesis,

$$f_v - b_v + 1 + f'_w - b'_w = 0.$$

Consequently,

$$\rho(v) - \rho(w) = N(v) - N(w) = (f_v - b_v) - (b'_w - f'_w) = -1$$

which is set out to be shown.

In addition, since all ambulance paths in graph H have net length $\leq m$ by hypothesis and $\ell = |N(x) - N(y)|$ indicates the existence of a path of net length ℓ between vertices x and y,

$$0 \leq \rho(v) = |\rho(v)| = |N(v) - N_{\min}| \leq m$$

for any $v \in V$. Thus ρ is a mapping from the vertex set V of graph H to the set $\{0, 1, \ldots, m\}$ and H is a graded digraph of rank m.

$\Longleftarrow$ Now suppose H (with vertex set V) is a graded digraph of rank m such that vertex $v \in V$ has rank $\rho(v)$. We need to prove that (i) all ambulance paths in H have net length $\leq m$ and (ii) all cycles in H are balanced.

To prove (i), consider any ambulance path of net length ℓ between vertices v and w. Then

$$\ell = |\rho(v) - \rho(w)| \leq \rho(v) - 0 \leq m$$

by hypothesis. Thus all ambulance paths in H have net length $\leq m$.

To prove (ii), let A be a vertex on a cycle in graph H. This cycle can also be considered as an ambulance path of net length $\ell = \rho(A) - \rho(A) = 0$ that starts from vertex A and back to A. Thus this cycle must be a balanced cycle. ∎

Theorem 4.8 summarizes the necessary conditions for universal deBruijn building blocks.

Theorem 4.8. If a digraph H is a universal deBruijn building block for the family of deBruijn graphs $\{B_n^q\}_{n \geq N}$, then

(i) H cannot contain two vertices X and Y such that there are more than one path of the same length from X to Y.

(ii) H is a graded digraph.

Proof: Condition (i) follows from Theorem 4.6 (i). Condition (ii) follows from Theorem 4.6 (ii) and Lemma 4.7. ∎

Many digraphs satisfy the two necessary conditions listed in Theorem 4.8. In particular, any spanning subgraph† H of B_k^q that is a graded digraph will definitely satisfy condition (ii). If we further restrict the rank of H to be $\leq k$, condition (i) will also be satisfied by Theorem 4.1 and Lemma 4.7. Surprisingly, these two conditions, i.e.,

(iii) H is a graded digraph.

(iv) H has rank $\leq k$.

are actually sufficient conditions for any spanning subgraph H of B_k^q to be a universal deBruijn building block for the family of deBruijn graphs $\{B_n^q\}_{n\geq k}$.

Theorem 4.9. Any spanning subgraph of B_k^q that is a graded digraph of rank $\leq k$ is a universal deBruijn building block for the family of deBruijn graphs $\{B_n^q\}_{n\geq k}$.

We first prove two lemmas that are necessary to prove Theorem 4.9. For convenience, we define V_n^q to be the set of all n-dimensional q-ary vectors. For a fixed q, we define three linear mappings L, R, and C ("left," "right," and "center") from V_n^q to V_{n-1}^q. (Technically, the mappings L, R , and C are each families of mappings, one for each $n \geq 2$.) If $x = [x_1, \ldots, x_n]$ is a q-ary vector of length n, then

$$Lx = (x_1, \ldots, x_{n-1})$$

$$Rx = (x_2, \ldots, x_n)$$

$$Cx = (x_1 - x_2, \ldots, x_{n-1} - x_n) \qquad (\text{mod } q).$$

For example, for $q = 2$, if $x = [10110]$, then $Lx = [1011]$, $Rx = [0110]$, and $Cx = [1101]$.

---

† A spanning subgraph H of graph G is a subgraph that contains all vertices of G.

Lemma 4.10. The mappings L and R commute with C , i.e., $CLx = LCx$ and $CRx = RCx$ for any q-ary vector x of length ≥ 3.

Proof: By direct computation we find that if $x = [x_1, \ldots, x_n]$, then

$$CLx = LCx = [x_1 - x_2, \ldots, x_{n-2} - x_{n-1}];$$

$$CRx = RCx = [x_2 - x_3, \ldots, x_{n-1} - x_n].$$

∎

We now define the *burst agreement* $B(x, y)$ between two n-vectors x and y as the length of the largest block of consecutive components on which x and y agree. For example if $x = [11010010]$ and $y = [01110001]$, then $B(x, y) = 3$ because x and y agree in positions 4, 5, and 6, but in no set of four or more consecutive positions.

Lemma 4.11. If x and y are two n-vectors with $C^r x = C^r y$, and $B(x, y) \geq r$, then $x = y$.

Proof: We use induction on r. For $r = 1$, the assertion is that if $Cx = Cy$, and if x and y agree in at least one coordinate, then x and y are identical. To see that this is so, note that C is a linear mapping from V_n^q to V_{n-1}^q. Its nullspace, i.e., the set of x's such that $Cx = 0$, is the set of vectors $[x_1, \ldots, x_n]$ such that $x_1 - x_2 = x_2 - x_3 = \cdots = x_{n-1} - x_n = 0$. This set contains only the q vectors $[00\cdots0]$, $[11\cdots1]$, ..., $[q-1, q-1, \cdots, q-1]$. Thus if $Cx = Cy$, then either $x = y$ or $x = y + [ii\cdots i]$ $(1 \leq i \leq q-1)$, i.e., x and y differ in all n positions. It follows that if $Cx = Cy$ and if x and y agree in at least one place, then $x = y$. This completes the proof for $r = 1$.

We now assume $r \geq 2$, and that the lemma has been proved for all $r' < r$. If $B(x, y) \geq r$, i.e., if x and y agree on r consecutive positions, then clearly Cx and Cy agree on at least $r - 1$ positions. Thus if we let $x' = Cx$ and $y' = Cy$, then $B(x', y') \geq r-1$. Also, the hypothesis $C^r x = C^r y$ is equivalent to $C^{r-1} x' = C^{r-1} y'$. Thus by the induction hypothesis, $x' = y'$, i.e., $Cx = Cy$. But also $B(x, y) \geq r \geq 1$,

so that by the $r = 1$ case of the lemma, which has already been proved, $x = y$. ∎

We now return to prove Theorem 4.9 by showing that any spanning subgraph of B_k^q that is a graded digraph of rank k builds B_n^q for all $n \geq k$.

Proof of Theorem 4.9: Let H_k^q be a spanning subgraph of B_k^q that is also a graded digraph of rank k. Let ρ be the rank function that maps the vertex set V_k^q of H_k^q to the set $\{0, 1, \ldots, k\}$. For any $X = [X_1, X_2, \ldots, X_n] \in V_n^q$, suppose that $C^{n-k}X = x \in V_k^q$, and $\rho(x) = i$. We define the $(n-k)$-bit *chip number* of X, denoted by $\text{num}(X)$, as

$$\text{num}(X) = [X_{i+1}, \ldots, X_{i+n-k}]. \tag{4.12}$$

Note that since $0 \leq i \leq k$, then $1 \leq i+1 \leq i+n-k \leq n$, so that the chip number as defined in (4.12) "fits" within the field of X. In building B_n^q with q^{n-k} copies of H_k^q ("chips"), numbered $[0, 0, \cdots, 0]$ to $[q-1, q-1, \cdots, q-1]$, we place vertex X on the chip numbered $\text{num}(X)$, at the location corresponding to $x = C^{n-k}X$. Lemma 4.11 shows that no two vertices of B_n^q can be assigned the same location on the same chip, so that each of the q^n vertices in B_n^q is assigned a unique "home" on one of the q^{n-k} chips. What remains to show is that the connections within the chips correspond to connections in the big graph B_n^q, i.e., that if $\text{num}(X) = \text{num}(Y)$ and if $C^{n-k}X$ and $C^{n-k}Y$ are connected on H_k^q, then X and Y are connected in B_n^q, i.e., $LX = RY$.

To see this, we reason as follows. Since $C^{n-k}X$ and $C^{n-k}Y$ are connected on H_k^q, then $\rho(C^{n-k}Y) = \rho(C^{n-k}X) + 1$. Thus if $\rho(C^{n-k}X) = i$, then $\rho(C^{n-k}Y) = i+1$, and so, since $\text{num}(X) = \text{num}(Y)$, we have

$$[X_{i+1}, \ldots, X_{i+n-k}] = [Y_{i+2}, \ldots, Y_{i+n-k+1}].$$

Thus LX and RY agree on $n-k$ consecutive positions, i.e.,

$$B(LX, RY) \geq n-k. \tag{4.13}$$

But also, since $C^{n-k}X$ and $C^{n-k}Y$ are connected on H_k^q, we have $LC^{n-k}X = RC^{n-k}Y$, which, by Lemma 4.10, implies

$$C^{n-k}LX = C^{n-k}RY. \tag{4.14}$$

Combining (4.13) and (4.14), using Lemma 4.11, we find that $LX = RY$, which is what we set out to prove. ∎

There is a small gap between the necessary conditions (Theorem 4.8) and the sufficient conditions (Theorem 4.9) on universal deBruijn building blocks. If H is a spanning subgraph of B_k^q such that H satisfies the necessary conditions (in Theorem 4.8) and H has rank $> k$, it is not known, in general, whether or not H is a universal deBruijn building block. All the graphs that have been found in this "gap" have been shown to be universal deBruijn building blocks. As an example, a Hamilton path of B_k^q, which falls into this "gap", is a universal deBruijn building block since all deBruijn graphs B_n^q are Hamiltonian [Golo82]. However, the construction in Theorem 4.9 fails to provide a general proof since num(X) in Equation (4.12) does not "fit" within the field of X.

4.7. An Example For Building B_n^q

In Figure 4.4, the graph $B_3(\rho)$ is both a spanning subgraph of B_3 and a graded digraph of rank 3. By Theorem 4.9, $B_3(\rho)$ is a universal deBruijn building block for the family of deBruijn graphs $\{B_n\}_{n\geq 3}$. We illustrate the construction in the proof of Theorem 4.9, by building the graph B_5 with four copies of the universal deBruijn building block $B_3(\rho)$. We begin with Table 4.3, which lists, for each of the 32 possible 5-bit vectors X, the 3-bit vector $x = C^2X$, and the corresponding rank $\rho(x)$.

We number the four copies of $B_3(\rho)$ 00, 01, 10 and 11. Table 4.3 can be used to find the chip number and the location within a chip of each 5-bit vector X, as

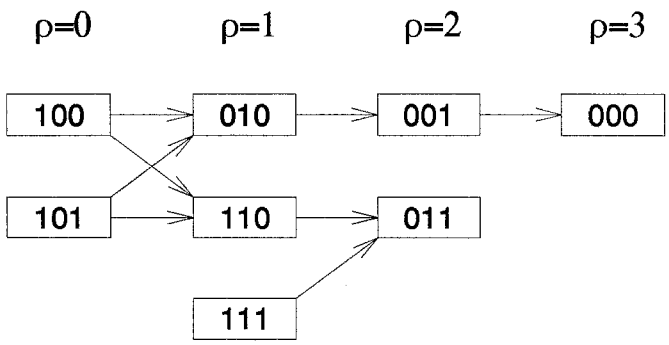

Figure 4.4. The universal deBruijn building block $B_3(\rho)$.

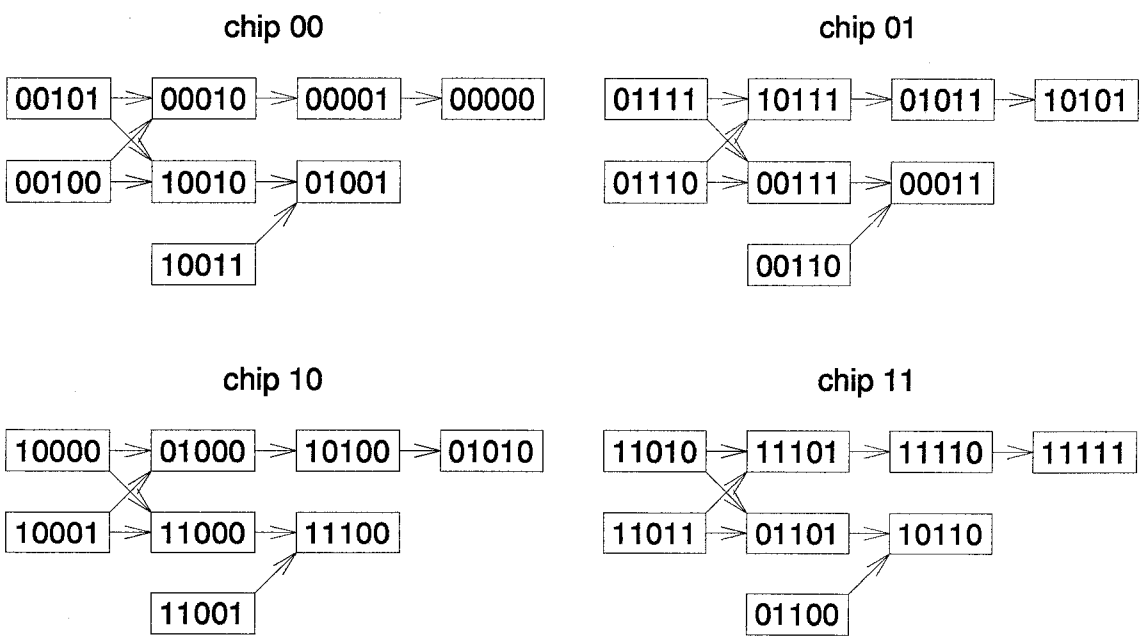

Figure 4.5. Four copies of the graph $B_3(\rho)$ in Figure 4.4 labelled to form a spanning subgraph of B_5. This is a four-chip VLSI decomposition of B_5 of efficiency 0.50.

X	$x = C^2X$	$\rho(x)$	X	$x = C^2X$	$\rho(x)$
00000	000	3	10000	100	0
00001	001	2	10001	101	0
00010	010	1	10010	110	1
00011	011	2	10011	111	1
00100	101	0	10100	001	2
00101	100	0	10101	000	3
00110	111	1	10110	011	2
00111	110	1	10111	010	1
01000	010	1	11000	110	1
01001	011	2	11001	111	1
01010	000	3	11010	100	0
01011	001	2	11011	101	0
01100	111	1	11100	011	2
01101	110	1	11101	010	1
01110	101	0	11110	001	2
01111	100	0	11111	000	3

Table 4.3. A table for building B_5 from 4 copies of the graph $B_3(\rho)$ in Figure 4.4.

follows. For a given X, the value $x = C^2X$ gives the location, and the two bits of X in positions $\rho(x) + 1$ and $\rho(x) + 2$, which are underlined in the table, give the chip number. For example, consider $X = 11000$. According to the table, $x = 110$, $\rho(x) = 1$, and the underlined bits are 10. Thus X is placed in location 110 in the chip numbered 10. The complete assignment of vertices of B_5 to the four chips is shown in Figure 4.5. Note that these four chips can be wired together to form B_5 with an efficiency of 0.50.

4.8. The Most Efficient Known Universal DeBruijn Building Blocks

From Theorem 4.9, we can obtain universal deBruijn building blocks $U_{(q,k)}$ of size q^k for the family of deBruijn graphs $\{B_n^q\}_{n \geq k}$ by searching for spanning subgraphs of B_k^q that is a graded digraph of rank $\leq k$. Figures 4.4 and 4.6–4.10 show the most efficient $U_{(2,k)}$ building blocks we have been able to find, using ad hoc methods, for $1 \leq k \leq 7$.† In particular, Figure 4.10 shows the most efficient known $U_{(2,7)}$ building block that is being used to build the single board Viterbi decoder for the Galileo code. (In Figures 4.7–4.10, the vertex labels shown are the decimal equivalents of the actual binary labels.) For each of the most efficient known $U_{(2,k)}$ building blocks, we list, in Table 4.4, the number of edges $\mathrm{E}(U_{(2,k)})$ and the efficiency

$$\mathrm{eff}(U_{(q,k)} \vdash B_n^q) = \frac{q^{n-k}\mathrm{E}(U_{(q,k)})}{q^{n+1}} = \frac{\mathrm{E}(U_{(q,k)})}{q^{k+1}} \tag{4.15}$$

which is independent of n. That is, the efficiency of a universal deBruijn building block $U_{(q,k)}$ is independent of the size of the deBruijn graph B_n^q which it is used to build.

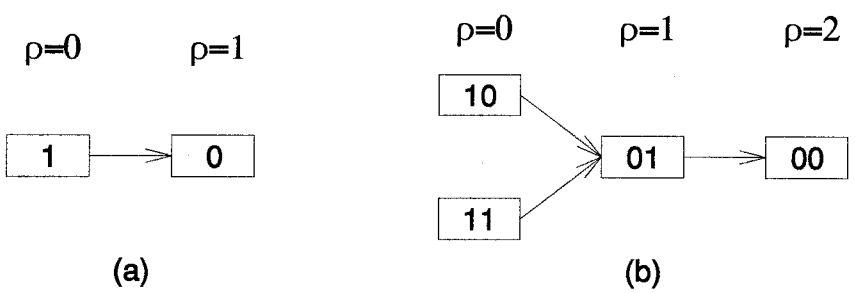

Figure 4.6. (a) The most efficient $U_{(2,1)}$ building block. (b) The most efficient $U_{(2,2)}$ building block.

† The building block for $k = 5$ was discovered by Gordon Oliver and the building blocks for $k = 6$ and $k = 7$ were discovered by Sam Dolinar, both from the Jet Propulsion Laboratories. Gordon has also found a building block for $k = 8$ with 398 edges.

k	$U_{(2,k)}$	$\mathrm{E}(U_{(2,k)})$	$\mathrm{eff}(U_{(q,k)} \vdash B_n^q)$	$c_{(2,k)}$	$\mathrm{E}_{\mathrm{ubound}}(U_{(2,k)})$
1	Fig. 4.6(a)	1	0.250	1.500	1
2	Fig. 4.6(b)	3	0.375	1.875	4
3	Fig. 4.4	8	0.500	2.000	9
4	Fig. 4.7	19	0.594	2.031	22
5	Fig. 4.8	43	0.672	1.969	48
6	Fig. 4.9	92	0.719	1.969	101
7	Fig. 4.10	193	0.754	1.969	209
8	Fig. −−	398	0.777	2.004	431

Table 4.4. The most efficient known $U_{(2,k)}$ building blocks, for $1 \leq k \leq 8$. $c_{(2,k)}$ and $\mathrm{E}_{\mathrm{ubound}}(U_{(2,k)})$ are defined in Equations (4.19) and (4.17).

We have been able to show by exhaustive search that the entries for $1 \leq k \leq 4$ are optimal in Table 4.4. For larger values of k, we can determine how much improvement may be possible by obtaining an upper bound for the number of edges in a $U_{(q,k)}$ building block. We note that there is at most one path between any two vertices X and Y in a universal deBruijn building block as implied by necessary conditions (i) and (ii) in Theorem 4.8. Thus we may apply Theorem 6.2 to obtain a lower bound on the number of input edges w (defined in Section 6.1) for any universal deBruijn building block $U_{(q,k)}$ with q^k vertices. By letting $d_i = q$ for all the q^k vertices v_i in Theorem 6.2, we obtain

$$q^{k+1} \leq w \log_q w. \tag{4.16}$$

Since $w \log_q w$ is a monotonic increasing function, there exists an integer $w_{\min}(q,k)$, for a given q and k, which is the smallest among all integers w that satify Inequal-

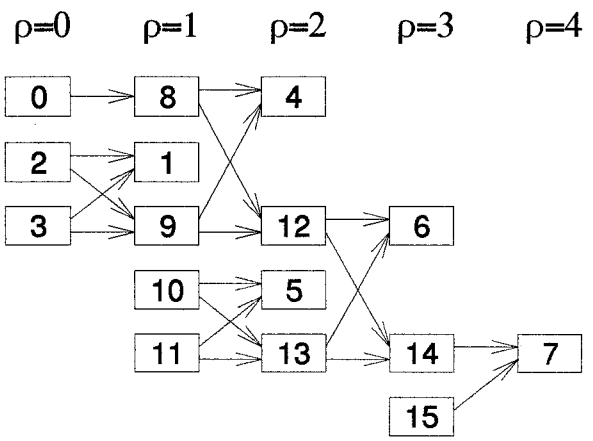

Figure 4.7. The most efficient $U_{(2,4)}$ building block. The edge count is 19.

ity (4.16). Then the number of edges in any universal deBruijn building block $U_{(q,k)}$ is bounded above by

$$\mathrm{E}(U_{(q,k)}) \leq \mathrm{E}_{\mathrm{ubound}}(U_{(q,k)}) = q^{k+1} - w_{\min}(q,k). \tag{4.17}$$

The values of $\mathrm{E}_{\mathrm{ubound}}(U_{(2,k)})$, for $1 \leq k \leq 8$, are listed in Table 4.4.

Similar to the derivation in Section 4.4, we may apply Lemma 4.5 on Inequality (4.16) to obtain

$$w \geq \frac{q^{k+1}}{k+1}.$$

Thus (4.17) becomes, after dividing both sides by q^{k+1},

$$\mathrm{eff}(U_{(q,k)} \vdash B_n^q) \leq 1 - \frac{1}{k+1}. \tag{4.18}$$

The following theorem gives a general construction for universal deBruijn building blocks that have an efficiency very close to the upper bound (4.18).

Theorem 4.12. Let S_n^q be the set of all q-ary strings X of length n such that one of the longest run of zeros in X is either at the extreme left or the extreme right.

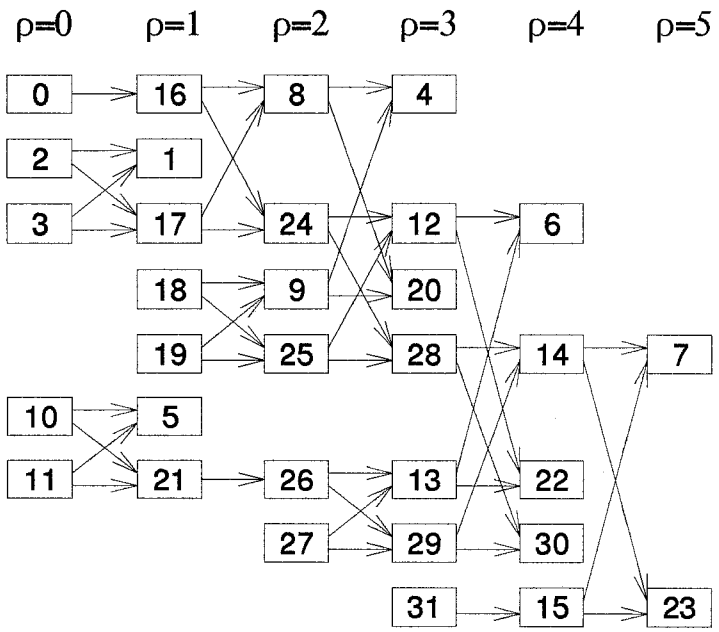

Figure 4.8. The most efficient known $U_{(2,5)}$ building block. The edge count is 43.

Let H_k^q be a spanning subgraph of B_k^q that does not contain any edge $\in S_{k+1}^q$ and contains all edges $\notin S_{k+1}^q$. H_k^q is a universal deBruijn building block for the family of deBruijn graphs $\{B_n^q\}_{n \geq k}$ with an efficiency $> 1 - 2q^2/(k+3)$.

Proof: By Theorems 4.3, 4.4, and Lemma 4.7, H_k^q is a graded digraph of rank $\leq k$. Since H_k^q is also a spanning subgraph of B_k^q, by Theorem 4.9, H_k^q is a universal deBruijn building block for the family of deBruijn graphs $\{B_n^q\}_{n \geq k}$.

To obtain the efficiency, we use an idea from Schwabe [Schw91] to count the number of elements in the set S_{k+1}^q. Let $Y \subset S_{k+1}^q$ be the set of q-ary strings X of length $k+1$ such that one of the longest run of zeros in X is at the extreme left. Consider the mapping $Y \rightarrow 0Y1$. This is a one-to-one mapping of strings $\in Y$ to strings of length $k+3$ that have a unique longest run of zeros at the extreme left

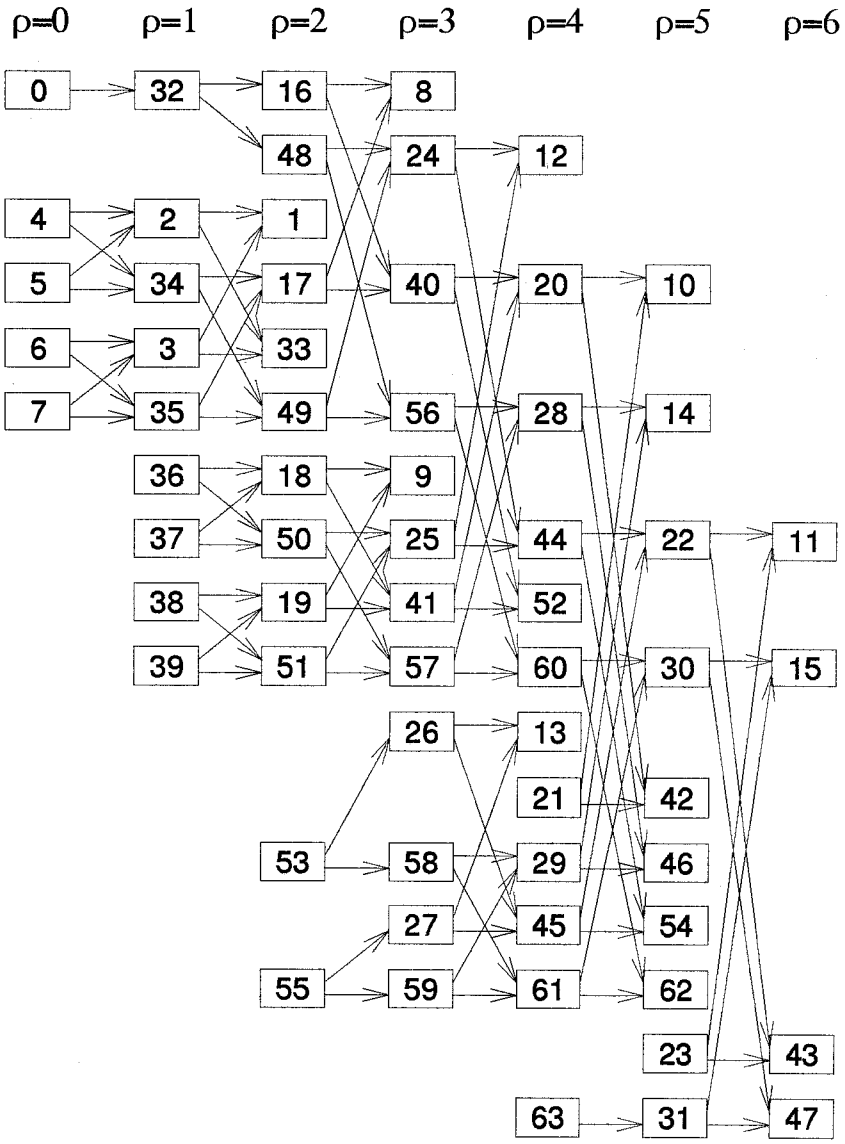

Figure 4.9. The most efficient known $U_{(2,6)}$ building block. The edge count is 92.

and are lexicographically least among their cyclic shifts. The size of this set and thus the number of elments in Y is $< q^{k+3}/(k+3)$. Similarly, the number of q-ary strings $X \in S^q_{k+1}$ such that one of the longest run of zeros in X is at the extreme right is also $< q^{k+3}/(k+3)$.

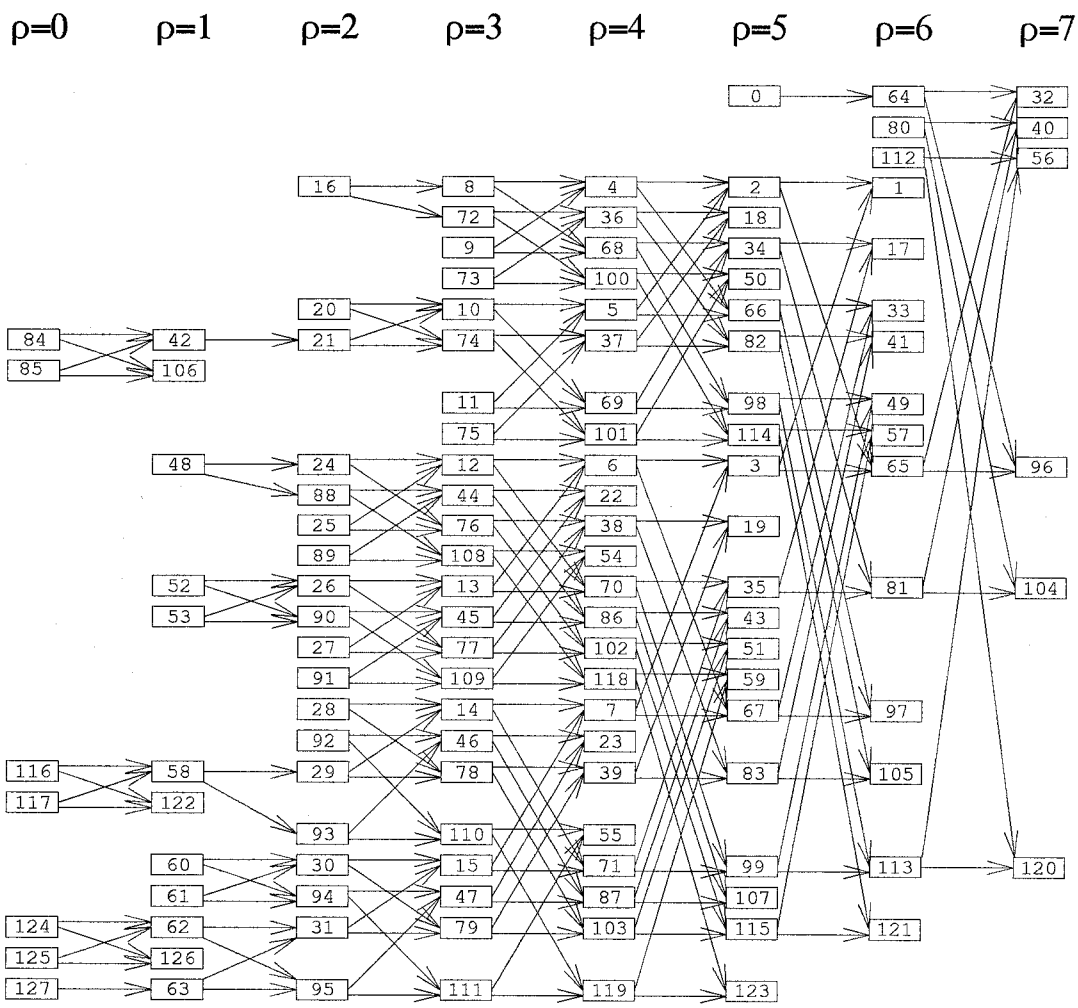

Figure 4.10. The most efficient known $U_{(2,7)}$ building block. The edge count is 193.

Consequently, the number of elements in S^q_{k+1} is $< 2q^{k+3}/(k+3)$ and the efficiency is

$$\text{eff}(H^k_q \vdash B^q_n) > 1 - \frac{2q^2}{k+3}$$

as stated in this theorem. ∎

Combining (4.18) and Theorem 4.12, we obtain

$$1 - \frac{2q^2}{k+3} < \liminf_{k\to\infty} e^*_{(q,k)} \leq \limsup_{k\to\infty} e^*_{(q,k)} \leq 1 - \frac{1}{k+1} \tag{4.19}$$

where $e^*_{(q,k)}$ denotes the efficiency for the most efficient universal deBruijn building block $U_{(q,k)}$.

We may obtain similar asymptotic behavior for general VLSI decomposition for deBruijn graphs B^q_n. Combining (4.10), (4.11) and Theorem 4.12, we obtain

$$1 - \frac{2q^2}{k+3} < \liminf_{k\to\infty} e_{\text{NI}}(B^q_n; q^k) \leq \limsup_{k\to\infty} e_{\text{NI}}(B^q_n; q^k) \leq 1 - \frac{1}{k} \tag{4.20}$$

and

$$1 - \frac{2q^2}{k+3} < \liminf_{k\to\infty} e_{\text{I}}(B^q_n; q^k) \leq \limsup_{k\to\infty} e_{\text{I}}(B^q_n; q^k) \leq 1 - \frac{1}{k}. \tag{4.21}$$

By comparing (4.19) with (4.20) and (4.21), we can conclude that it is not a severe restriction in requiring the subgraphs to be both "isomorphic" and "universal" in a VLSI decomposition of deBruijn graphs B^q_n into a large number of subgraphs.

In Table 4.4, we also list the values of $c_{(q,k)}$ which is defined by

$$\text{eff}(U_{(q,k)} \vdash B^q_n) = 1 - \frac{c_{(q,k)}}{k+1}. \tag{4.22}$$

It seems that the efficiency for the most efficient $U_{(2,k)}$ building block approaches $1 - 2/(k+1)$ as $k \to \infty$.

V. Shuffle-Exchange Graphs

5.1. Binary Shuffle-Exchange Graphs Ψ_n

The binary shuffle-exchange graph Ψ_n consists of 2^n vertices, each represented by a binary n-tuple, 2^n shuffle edges, each labelled by a binary n-tuple, and 2^{n-1} exchange edges, each labelled by a binary $(n-1)$-tuple. The shuffle edge $x_1x_2\cdots x_n$ is a directed edge from $x_2x_3\cdots x_nx_1$ to $x_1x_2x_3\cdots x_n$. The exchange edge $x_1x_2\cdots x_{n-1}$ is an undirected edge connecting $0x_1x_2\cdots x_{n-1}$ and $1x_1x_2\cdots x_{n-1}$. Figure 5.1 shows the binary shuffle-exchange graph Ψ_3.

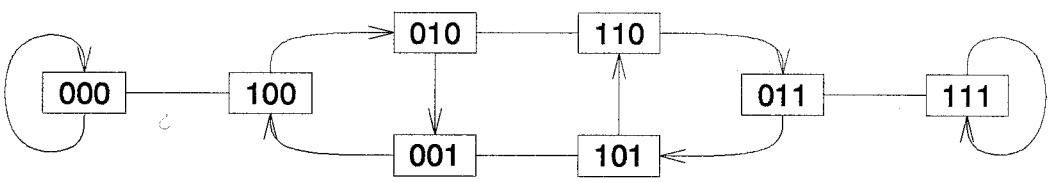

Figure 5.1. The binary shuffle-exchange graph Ψ_3.

5.2. Properties of Shuffle-Exchange Graphs Ψ_n

The binary shuffle-exchange graph Ψ_n bears a strong resemblance to the binary deBruijn graph B_n. Every edge in the deBruijn graph B_n can be obtained by either one shuffle edge or a shuffle edge followed by an exchange edge in Ψ_n. The theorems and the corresponding proofs for deBruijn graphs can be adapted to shuffle-exchange graphs as shown in this section. (The definitions for trail, net length, et al. are given in Section 4.3.)

Theorem 5.1. If X and Y are vertices in Ψ_n, and l is a positive integer with $l < n$, then there cannot be more than one trail of net length l from X to Y.

Proof: Let $X = x_0 x_{n-1} x_{n-2} \cdots x_1$. Let P_i $(0 \leq i \leq l-1)$ to be the vertex immediately preceding the $(i+1)$th forward edge in a trail from X to Y and let $P_l = Y$. The vertex P_0 must be of the form $y_0 x_{n-1} x_{n-2} \cdots x_1$ where $y_0 = x_0$ if P_0 is the same as vertex X and $y_0 = \overline{x_0}$ if the first edge is an exchange edge in the trail from X to Y. The vertex P_1 must be of the form $y_1 y_0 x_{n-1} x_{n-2} \cdots x_2$ where $y_1 = x_1$ if P_0 is connected to P_1 by a shuffle edge and $y_1 = \overline{x_1}$ if P_0 is connected to P_1 by a shuffle edge followed by an exchange edge. Continuing in this way, the vertex Y is of the form $Y = y_l y_{l-1} \cdots y_0 x_{n-1} x_{n-2} \cdots x_{l+1}$ where, for $1 \leq i \leq l$, $y_i = x_i$ if P_{i-1} is connected to P_i by a shuffle edge and $y_i = \overline{x_i}$ if P_{i-1} is connected to P_i by a shuffle edge followed by an exchange edge. It follows that the vertices P_i $(0 \leq i \leq l-1)$ on the trail, and also the trail itself, is uniquely determined by X and Y. ∎

Theorem 5.2. If X and Y are vertices in Ψ_n, and l is a positive integer with $l \leq n$, then there is at most one trail of net length l that begins with a shuffle edge from X to Y.

Proof: Let X' be the vertex immediately following X on a trail from X to Y. Since there is only one shuffle edge originated from any vertex in Ψ_n, the vertex X' is uniquely determined by X. By Theorem 5.1, the remaining trail of net length $l-1$ from X' to Y is uniquely determined by X' and Y. Therefore, the trail, if one exists, is uniquely determined by X and Y. ∎

Theorem 5.3. In Ψ_n, there are at most $2^{l+1}/l$ vertex-disjoint unbalanced (r, s) cycles, where $l = r + s$ is the total number of directed edges on the cycle.

Proof: If $l \geq n-1$, the theorem is trivially true, since Ψ_n has 2^n vertices and thus at most $2^n/l \leq 2^{l+1}/l$ vertex-disjoint unbalanced cycles of net length l. On

the other hand, if $l < n-1$, let $A = a_1 a_2 \cdots a_n$ be a vertex which lies on an (r,s) cycle. Since a cycle may be traversed in either of two opposite directions, we may assume $r > s$ without loss of generality. After traversing the cycle and returning back to A, the substring $a_{s+2}a_{s+3}\cdots a_{n-r}$ will have been shifted to the right $r-s$ positions and so we have

$$A = x_r x_{r-1} \cdots x_1 x_0 a_{s+2} a_{s+3} \cdots a_{n-r} y_1 y_2 \cdots y_s.$$

Since this expression must be the same as the original label $a_1 a_2 \cdots a_n$, by equating the corresponding bits in positions $r+2$ through $n-s$, we obtain the equations

$$a_{r+i} = a_{s+i} \qquad (2 \le i \le n-s-r). \tag{5.1}$$

It follows from (5.1) that A is uniquely determined by the $l+1$ values $a_0, a_1, \ldots, a_r$ and $a_{n-s+1}, a_{n-s+2}, \ldots, a_n$, so that there are at most $2^{r+1+s} = 2^{l+1}$ vertices that can lie on an (r,s) cycle. Since each (r,s) cycle contains at least l vertices, it follows that a set of vertex-disjoint (r,s) cycles (in Ψ_n) contain at most $2^{l+1}/l$ members independent of n. ∎

Let X and Y be sets of strings. If every string in Y has a substring in X, we say that X *covers* Y. In addition, if no string in X is a substring of any other string in X, we say that X is *irreducible*. The following theorem shows that, if T_n is a set of strings that covers $\{0,1\}^n$, we may remove all unbalanced cycles in Ψ_n by deleting all shuffle edges that have a prefix in T_n and all exchange edges that have an endvertex whose prefix is in T_n.

Theorem 5.4. If T_n is a set of strings that covers $\{0,1\}^n$, then, for every unbalanced cycle in Ψ_n,

(i) there exists a vertex V that has a prefix in T_n;

(ii) there exists an edge E such that E is either a shuffle edge with prefix in T_n or an exchange edge with an endvertex whose prefix is in T_n.

Proof: Let $P_1 = a_1 a_2 \cdots a_n$ be a vertex on an unbalanced (r, s) cycle in Ψ_n. Without loss of generality, we assume $r < s$ (i.e., there are more backward edges than forward edges) and T_n is irreducible. Let $P_1, E_1, P_2, E_2, \cdots, P_l, E_l, P_{l+1}$ (where $P_{l+1} = P_1$) be the sequence of vertices and edges if we trace along the cycle starting from vertex P_1 and back to $P_1 = P_{l+1}$. Let

$$\begin{aligned} r' &= \max_{1 \le i \le l+1} d(P_1, P_i) \\ s' &= \min_{1 \le i \le l+1} d(P_1, P_i) \end{aligned} \tag{5.2}$$

where $d(P_1, P_i)$ denotes the signed net length of the ambulance path $P_1 P_2 \cdots P_i$. Then $a_{s'+1} a_{s'+2} \cdots a_{n-r'}$ is the longest substring in $P_1 = a_1 a_2 \cdots a_n$ such that any other vertex P_i, $2 \le i \le l+1$, contains either the substring $a_{s'+1} a_{s'+2} \cdots a_{n-r'}$ or the substring $\overline{a_{s'+1}} a_{s'+2} \cdots a_{n-r'}$.

We first prove part (i) of the theorem. Since T_n covers $\{0,1\}^n$, the vertex P_1 has at least one substring that is an element in T_n. Let $a_u a_{u+1} \cdots a_v$ be the first occurrence (i.e., the leftmost) of a T_n-string in P_1. If $u = 1$, then P_1 has a prefix in T_n and we are done. Otherwise the next vertex P_2 on the unbalanced cycle has the label $a_n a_1 a_2 \cdots a_{n-1}$ (if E_1 is a forward edge), or $a_2 a_3 \cdots a_n a_1$ (if E_1 is a backward edge), or $\overline{a_1} a_2 a_3 \cdots a_n$ (if E_1 is an exchange edge). In all cases, either P_2 has a prefix in T_n or the substring $a_u a_{u+1} \cdots a_v$ remains to be the first occurrence of a T_n-string in P_2. Continuing this way, if, for all $1 \le i \le j$, P_i does not have a prefix in T_n, then either P_{j+1} has a prefix in T_n, or the substring $a_u a_{u+1} \cdots a_v$ is the first occurrence of a T_n-string in P_{j+1}. Now we consider three cases: (a) $u \le s' + 1$, (b) $v > n - r'$, and (c) $s' + 1 < u \le v \le n - r'$.

(a) For the case $u \le s' + 1$, let u' be the smallest positive integer such that $d(P_1, P_{u'}) = u$, i.e.,

$$u = d(P_1, P_{u'}) < d(P_1, P_i) \ \ \forall i < u'.$$

Then $P_{u'}$ has the prefix $a_u a_{u+1} \cdots a_v$ which is a T_n-string.

(b) For the case $v > n - r'$, let v' be the smallest positive integer such that $d(P_1, P_{v'}) = n - v + 1$, i.e.,

$$n - v + 1 = d(P_1, P_{v'}) > d(P_1, P_i) \ \forall i < v'.$$

If, for all $1 \leq i < v'$, P_i does not have a prefix in T_n, then $P_{v'}$ must have a prefix in T_n since a_v has been shifted out. That is, $P_{v'}$ has the suffix $a_u a_{u+1} \cdots a_{v-1}$ and thus $P_{v'}$ does not contain any T_n-string if the prefix of $P_{v'}$ is not in T_n. This contradicts with the fact that T_n is a cover for $\{0,1\}^n$.

(c) For the remaining case $s' + 1 < u \leq v \leq n - r'$, we will show that $a_u a_{u+1} \cdots a_v$ is not the first occurrence of a T_n-string in P_1 and thus contradicts with the original assumption. First note that vertex P_{l+1} has the form

$$P_{l+1} = x_{s'-(s-r)} x_{s'-(s-r)-1} \cdots x_1 x_0 a_{s'+2} a_{s'+3} \cdots a_{n-r'} y_1 y_2 \cdots y_{r'+(s-r)}.$$

By equating the corresponding bits of P_{l+1} and $P_1 = a_1 a_2 \cdots a_n$, we obtain the equations

$$a_{i-(s-r)} = a_i \qquad (s' + 2 \leq i \leq n - r'). \tag{5.3}$$

It follows from (5.3) and $s' + 1 < u \leq v \leq n - r'$ that the substring $a_{u-(s-r)} \cdots a_{v-(s-r)} = a_u \cdots a_v$ which has an earlier occurrence in P_1 is also a T_n-string.

To prove (ii), we use the result in part (i) which asserts the existence of a vertex P_i that has a prefix in T_n. In Ψ_n, P_i is the endvertex of two shuffle edges and an exchange edge. One of the two shuffle edges has the same label as P_i and thus also has a prefix in T_n. Then at least one of the two distinct edges E_{i-1} or E_i will satisfy the desired property, i.e., either a shuffle edge with prefix in T_n or an exchange edge with an endvertex whose prefix is in T_n. ∎

Theorem 5.5. If T_n is a set of strings that covers $\{0,1\}^n$, then

(i) In an ambulance path of net length $\geq n-1$ in Ψ_n, there exists a vertex V that has a prefix in T_n.

(ii) In an ambulance path of net length $> n$ in Ψ_n, there exists an edge E such that E is either a shuffle edge with prefix in T_n or an exchange edge with an endvertex whose prefix is in T_n.

Proof: Let $P_1 = a_1 a_2 \cdots a_n$ and let $P_1, E_1, P_2, E_2, \cdots, P_l, E_l, P_{l+1}$ be the sequence of vertices and edges if we trace along an ambulance path from vertex P_1 to P_{l+1}. Let $d(P_i, P_j)$ denote the signed net length of the ambulance path from P_i to P_j. Without loss of generality, we assume $d(P_1, P_{l+1})$ to be non-positive.

Since T_n covers $\{0,1\}^n$, there exists a T_n-string, say $a_u a_{u+1} \cdots a_v$, in P_1. If $|d(P_1, P_{l+1})| \geq n-1$, there must exist a smallest positive integer u' such that $d(P_1, P_{u'}) = -(u-1)$ since $u \leq n$ and thus $-(u-1) \geq d(P_1, P_{l+1})$. Then $P_{u'}$ has a prefix that is in T_n.

To prove part (ii), note that if $|d(P_1, P_{l+1})| > n$, then $|d(P_2, P_l)| \geq n-1$. By the result in part (i), there exists a vertex P_i $(2 \leq i \leq l)$ that has a prefix in T_n. Then at least one of the two edges E_{i-1} and E_i on the ambulance path will satisfy the desired property, i.e., either a shuffle edge with prefix in T_n or an exchange edge with an endvertex whose prefix is in T_n. ∎

5.3. C-Chip VLSI Decompositions for Shuffle-Exchange Graphs Ψ_n

As stated in Theorem 6.7, for any induced subgraph† (with external edges) $H_{\text{ext}}(m)$ with m vertices of Ψ_n,

$$2(l+1)(m) \leq (w'l + 2m)\log_2(w'l + 2m) \qquad \forall l \leq n \tag{5.4}$$

† defined in Section 6.3.

where w' denotes the number of external edges in $H_{\text{ext}}(m)$. Note that Inequality (5.4) is exactly the same as Inequality (4.4) with q substituted by 2 and w substituted by w'. Thus we may use the same derivation in Section 4.4 to obtain the lower bound

$$w' \geq w_{\text{lowerbound}}(2, m, n) = \max_{l \leq n} w_{\min}(2, m, l) \tag{5.5}$$

on the number of external edges on any subgraph $H_{\text{ext}}(m)$ of Ψ_n. ($w_{\text{lowerbound}}(q, m, n)$ and $w_{\min}(q, m, l)$ are defined in Equation (4.5) and the paragraph before (4.5) respectively.) Consequently, since each vertex in $H_{\text{ext}}(m)$ has degree 3, the number of internal edges in any subgraph $H_{\text{ext}}(m)$ of Ψ_n is bounded above by

$$\mathrm{E}^*_{\Psi_n}(m) \leq \frac{1}{2}(3m - w_{\text{lowerbound}}(2, m, n)). \tag{5.6}$$

Some values of $w_{\text{lowerbound}}(2, m, n)$ are listed in Tables 4.1 and 4.2. It follows from (2.4) and (5.6) that the efficiency of a C-chip VLSI decomposition for the shuffle-exchange graph Ψ_n into subgraphs $H_1(k_1), H_2(k_2), \cdots, H_C(k_C)$ of sizes $k_1, k_2, \ldots, k_C$ respectively is bounded above by

$$\text{eff}(H_1(k_1), H_2(k_2), \cdots, H_C(k_C) \mapsto \Psi_n) \leq 1 - \frac{1}{3 \cdot 2^n} \sum_{1 \leq i \leq C} w_{\text{lowerbound}}(2, k_i, n). \tag{5.7}$$

By (4.9), for $n \geq (\log_2 m)^{1+\alpha}$ where α is a constant > 0,

$$\begin{aligned} w_{\text{lowerbound}}(2, m, n) &\geq \frac{(2m)((\log_2 m)^{1+\alpha} + 1 - \log_2((\log_2 m)^{1+\alpha} + 1) - \log_2(2m))}{((\log_2 m)^{1+\alpha})(\log_2((\log_2 m)^{1+\alpha} + 1) + \log_2(2m))} \\ &\asymp \frac{2m}{\log_2 m} \qquad \text{as} \quad m \to \infty. \end{aligned} \tag{5.8}$$

We may interpret (5.7) and (5.8) as follows. If a large shuffle-exchange graph Ψ_n is decomposed into sufficiently small chips of sizes $k_1, k_2, \ldots, k_C$, then there are at least

$$\sum_{i=1}^{C} \frac{k_i}{\log_2 k_i}$$

external wires.

5.4. Shuffle-Exchange Building Blocks

From Theorems 5.1 and 5.3, we obtain two necessary conditions for shuffle-exchange building blocks.

Theorem 5.6. If a graph H of size q^k/C is a building block for the shuffle-exchange graph Ψ_n, then

(i) For any two vertices X and Y in H, there cannot be more than one trail of the same net length $l < n$ from X to Y.

(ii) H does not contain any unbalanced (r, s) cycle such that $r + s + 1 < \log_2 C$.

Proof: Part (i) follows immediately from Theorem 5.1 since H is a subgraph of Ψ_n. For (ii), note that Theorem 5.3 implies that $C \leq 2^{r+s+1}/(r+s) \leq 2^{r+s+1}$. ∎

We may use the results in Section 5.3 to obtain an upper bound for the efficiency of shuffle-exchange building blocks. By letting $k_1 = k_2 = \cdots = k_C = 2^k$ in (5.7), the efficiency for a building block of size 2^k for the shuffle-exchange graph Ψ_n is bounded above by

$$e_{\mathrm{I}}(\Psi_n; 2^k) \leq e_{\mathrm{NI}}(\Psi_n; 2^k) \leq 1 - \frac{w_{\mathrm{lowerbound}}(2, 2^k, n)}{3 \cdot 2^k} \tag{5.9}$$

where

$$\liminf_{k \to \infty} w_{\mathrm{lowerbound}}(2, 2^k, n) \geq \frac{2^{k+1}}{k} \tag{5.10}$$

for $n \geq k^{1+\alpha}$ (where $\alpha > 0$) as implied by (5.8). That is, if a large shuffle-exchange graph Ψ_n is decomposed into sufficiently small subgraphs of equal sizes 2^k, then the set of external edges will contain at least $2/(3k)$ of the edges in Ψ_n.

5.5. Universal Shuffle-Exchange Building Blocks

We consider *universal shuffle-exchange building blocks* for the family of shuffle-exchange graphs $\{\Psi_n\}_{n\geq N}$. Since a universal shuffle-exchange building block is necessary to build shuffle-exchange graphs Ψ_n with arbitrary large n, it follows from Theorem 5.6 that

(i) A universal shuffle-exchange building block cannot contain two vertices X and Y such that there are two trails of the same net length from X to Y.

(ii) A universal shuffle-exchange building block cannot contain any unbalanced cycle.

Similar to universal deBruijn building blocks (c.f. Section 4.6), condition (ii) is equivalent to the statement that a universal shuffle-exchange building block must be a *graded* graph. In the following, we first extend the definition of graded digraphs to graphs with both directed and undirected edges.

A graph G with vertex set V is graded of rank m if there is a mapping ρ from the vertex set V to the set $\{0, 1, \ldots, m\}$, such that, for $x, y \in V$, $\rho(y) = \rho(x) + 1$ if there is a directed edge from x to y and $\rho(y) = \rho(x)$ if there is an undirected edge connecting x and y. We call $\rho(x)$ the *rank* of x.

Lemma 5.7. A graph H is a graded graph iff all cycles in H are balanced. Furthermore, the graded graph H has rank m iff all ambulance paths in H have net length $\leq m$.

Proof: We can prove this lemma by using the same argument given in the proof of Lemma 4.7. The details are not repeated here. ∎

Theorem 5.8 summarizes the necessary conditions for universal shuffle-exchange building blocks.

Theorem 5.8. If a graph H is a universal shuffle-exchange building block for the family of shuffle-exchange graphs $\{\Psi_n\}_{n \geq N}$, then

(i) H cannot contain two vertices X and Y such that there are more than one trail of the same net length from X to Y.

(ii) H is a graded graph.

Proof: Condition (i) follows from Theorem 5.6 (i). Condition (ii) follows from Theorem 5.6 (ii) and Lemma 5.7. ∎

Many graphs satisfy the two necessary conditions listed in Theorem 5.8. By Theorems 5.3, 5.4, 5.1, and Lemma 5.7, we can obtain such a graph as follows:

(i) Find a set of strings T_k that covers $\{0,1\}^k$.

(ii) Construct the subgraph H_k by deleting from Ψ_k all shuffle edges that have a prefix in T_k and all exchange edges that have an endvertex whose prefix is in T_k.

We will prove that H_k is indeed a universal shuffle-exchange building block.

Theorem 5.9. If T_k covers $\{0,1\}^k$, then the subgraph $\Psi_k(T_k)$, obtained by deleting from Ψ_k all shuffle edges that have a prefix in T_k and all exchange edges that have an endvertex whose prefix is in T_k, is a universal shuffle-exchange building block for the family of shuffle-exchange graphs $\{\Psi_n\}_{n \geq k}$.

Proof: We will prove this theorem by showing a general construction to build Ψ_n (for all $n \geq k$) from 2^{n-k} copies of $\Psi_k(T_k)$. Without loss of generality, we assume T_k to be irreducible.

Let the 2^{n-k} copies of $\Psi_k(T_k)$-chips be numbered by $n-k$ bits $a_1 a_2 \cdots a_{n-k}$, where $a_i \in \{0,1\}$. We will give a procedure to label the vertices in each chip by n-bit labels so that the labelled chips can be viewed as isormorphic subgraphs.

We will then prove that the isormorphic subgraphs are non-overlapping and can be wired together to form Ψ_n. The labelling procedure is as follows. Consider a vertex $X = x_1x_2\cdots x_k$ in an $\Psi_k(T_k)$-chip numbered $A = a_1a_2\cdots a_{n-k}$. Suppose $x_ux_{u+1}\cdots x_v$ is the first occurrence of a T_k-string in X. We will then label vertex X in chip A as

$$N(X,A) = x_1x_2\cdots x_va_1a_2\cdots a_{n-k}x_{v+1}x_{v+2}\cdots x_{v+n-k}. \tag{5.11}$$

We will need to prove the following: (i) Each vertex among all the chips has a distinct label. That is, $N(X_1,A_1) \neq N(X_2,A_2)$ unless $X_1 = X_2$ and $A_1 = A_2$. (ii) If two vertices are connected in an $\Psi_k(T_k)$-chip, then these two vertices are also connected in the big graph Ψ_n. That is, if X and Y are connected, then $N(X,A)$ and $N(Y,A)$ are also connected.

We will prove (i) by showing that each n-bit string occurs in one of the 2^n labels $N(X,A)$. Then, by the pigeonhole principle, part (i) will be true. Consider an n-bit string $Y = y_1y_2\cdots y_n$. Suppose the first occurrence of a T_k-string in Y is the substring $y_{u'}y_{u'+1}\cdots y_{v'}$. Let $X = y_1\cdots y_{v'}y_{v'+n-k+1}\cdots y_n$, i.e., X is obtained by removing from Y the $n-k$ bits immediately following the first T_k-string. Then the first occurrence of a T_k-string in X must also be $y_{u'}y_{u'+1}\cdots y_{v'}$. Consequently, vertex X in the chip numbered $A = y_{v'+1}y_{v'+2}\cdots y_{v'+n-k}$ is labelled by $N(X,A) = Y = y_1y_2\cdots y_n$.

To prove (ii), let $X = x_1x_2\cdots x_k$ and suppose the first occurrence of a T_k-string in X is $x_ux_{u+1}\cdots x_v$. If X and Y are connected by a shuffle edge from X to Y, $Y = x_kx_1x_2\cdots x_{k-1}$. If X and Y are connected by an exchange edge, $Y = \overline{x_1}x_2x_3\cdots x_k$. In both cases, the first occurrence of a T_k-string in Y must also be $x_ux_{u+1}\cdots x_v$ since $\Psi_k(T_k)$ does not have any shuffle edge with prefix in T_k or any exchange edge with an endvertex whose prefix is in T_k. By Equation (5.11), $N(X,A)$ and $N(Y,A)$ must also be connected likewise in Ψ_n. ∎

5.6. An Example For Building Ψ_n

The set of strings $T_4 = \{10, 0000, 0001, 0011, 0111, 1111\}$ is an irreducible cover for $\{0,1\}^4$. By Theorem 5.9, the subgraph $\Psi_4(T_4)$ (shown in Figure 5.2), obtained by deleting from Ψ_4 all shuffle edges that have a prefix in T_4 and all exchange edges that have an endvertex whose prefix is in T_4, is a universal shuffle-exchange building block for the family of shuffle-exchange graphs $\{\Psi_n\}_{n\geq 4}$. We illustrate the construction in the proof of Theorem 5.9, by building the graph Ψ_6 with four copies of the universal shuffle-exchange building block $\Psi_4(T_4)$.

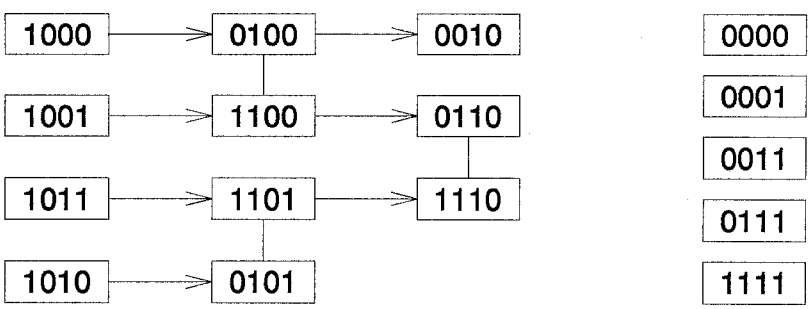

Figure 5.2. The universal shuffle-exchange building block $\Psi_4(T_4)$.

Consider the vertex 1100 in Figure 5.2. The first occurrence of a T_4-string is the substring 10. By (5.11), we assign $N(1100, 00) = 110000$ to the vertex 1100 in the chip numbered 00. Similarly, for the other three chips numbered $A = 01, 10, 11$, we assign $N(1100, A) = 110A0$ to the vertex 1100. After this labelling procedure is completed for every vertex in $\Psi_4(T_4)$, we obtain four labelled chips as shown in Figure 5.3. Note that these four chips contain all the 64 vertex labels of Ψ_6 and they can be wired together to form Ψ_6.

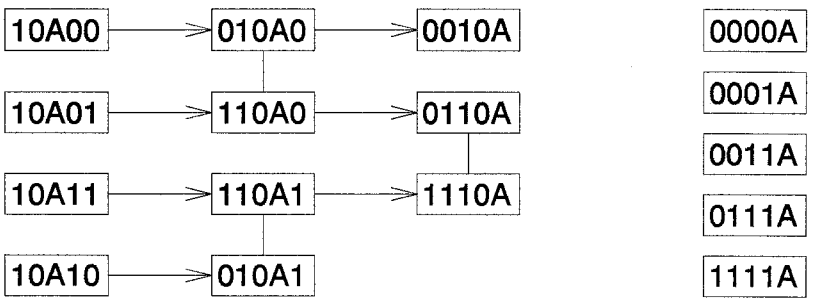

Figure 5.3. Four labelled copies of $\Psi_4(T_4)$, with $A = 00, 01, 10, 11$, that can be wired together to build Ψ_6.

5.7. The Most Efficient Known $\Psi_k(T_k)$ Building Blocks

Theorem 5.9 provides a method to obtain universal shuffle-exchange building blocks U_k of size 2^k for the family of shuffle-exchange graphs $\{\Psi_n\}_{n \geq k}$. Similar to universal deBruijn building blocks, the efficiency of a universal shuffle-exchange building block U_k is also independent of the size of the shuffle-exchange graph Ψ_n which it is used to build. Indeed,

$$\text{eff}(U_k \vdash \Psi_n) = \frac{2^{n-k}\text{E}(U_k)}{3 \cdot 2^{n-1}} = \frac{\text{E}(U_k)}{3 \cdot 2^{k-1}} \tag{5.12}$$

independent of n. Furthermore, the efficiency of the universal shuffle-exchange building block $\Psi_k(T_k)$ (obtained by using Theorem 5.9) can be determined from the set of strings T_k. If we define the *cost* of a set of binary strings X to be $\text{cost}(X) = \sum_{x \in X} 2^{-|x|}$, where $|x|$ denotes the length of string x, we can obtain a lower bound on the efficiency of $\Psi_k(T_k)$ as follows. Consider an irreducible set of strings T_k that covers $\{0,1\}^k$. In the shuffle-exchange graph Ψ_k, there are $\text{cost}(T_k) \cdot 2^k$ shuffle edges with a prefix in T_k and there are at most $\text{cost}(T_k) \cdot 2^k$ exchange edges with an endvertex whose prefix is in T_k. Therefore, by (5.12), the efficiency of the VLSI decomposition for Ψ_n into universal shuffle-exchange building blocks $\Psi_k(T_k)$

is bounded below by

$$\text{eff}(\Psi_k(T_k) \vdash \Psi_n) \geq \frac{3 \cdot 2^{k-1} - \text{cost}(T_k) \cdot 2^{k+1}}{3 \cdot 2^{k-1}} = 1 - \frac{4}{3}\text{cost}(T_k). \tag{5.13}$$

It follows from (5.13) that we can obtain efficient universal shuffle-exchange building blocks from low cost irreducible sets T_k that cover $\{0,1\}^k$. Although a minimum cost irreducible set T_k^* (that covers $\{0,1\}^k$) does not guarantee to provide the most efficient $\Psi_k(T_k)$ building block, T_k^* gives an indication on how efficient a universal shuffle-exchange building block we can get. In Table 5.1, we list some minimum cost irreducible sets T_k^*, for $1 \leq k \leq 9$, which are obtained by exhaustive computer search. For a given k, the minimum cost irreducible set T_k^* is not unique and we have chosen one randomly to be listed in the table.

For each of the irreducible sets T_k^* shown in Table 5.1, we list the cost of T_k^* and the efficiency of $\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n)$ in Table 5.2. The lower bound for the efficiency $\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n)$ is computed by using (5.13). The actual value of $\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n)$ is obtained by counting the number of edges in $\Psi_k(T_k^*)$. The upper bound for the efficiency $\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n)$ is computed by using (5.16) which will be derived later in this section. As an example, consider the case $k = 4$ with $T_4^* = \{10, 0000, 0001, 0011, 0111, 1111\}$. We find that $\text{cost}(T_4^*) = 2^{-2} + 5 \cdot 2^{-4} = 9/16$. By (5.13), $\text{eff}(\Psi_4(T_4^*) \vdash \Psi_n)$ is bounded below by $1 - (4/3) \cdot \text{cost}(T_4^*) = 1 - (4/3) \cdot (9/16) = 12/48$. By (5.16), $\text{eff}(\Psi_4(T_4^*) \vdash \Psi_n)$ is bounded above by $38/48$ since $w'_{\min}(4) = 10$. We can obtain the actual value of $\text{eff}(\Psi_4(T_4^*) \vdash \Psi_n)$ indirectly by counting the number of exchange edges that have both endvertices with prefix in T_4^*. (In general, the number of edges in $\Psi_k(T_k)$ is equal to $3 \cdot 2^{k-1} - \text{cost}(T_k) \cdot 2^{k+1} + N$ where N denotes the number of exchange edges that have both endvertices with prefix in T_k.) We note that there are four exchange edges, namely, 000, 001, 011, and 111, that have such a property in Ψ_4. Thus the number of edges in $\Psi_4(T_4^*)$ is $24 - 18 + 4 = 10$ and $\text{eff}(\Psi_4(T_4^*) \vdash \Psi_n) = 20/48$ as shown in Table 5.2.

Similar to the derivation of (4.16) for universal deBruijn building blocks, we

k	T_k^*
1	$\{1,0\}$
2	$\{1,00\}$
3	$\{1,000\}$
4	$\{10,0000,0001,0011,0111,1111\}$
5	$\{10,00000,00001,00011,00111,01111,11111\}$
6	$\{10,000000,000001,000011,000111,001111,011111,111111\}$
7	$\{10,0000000,0000001,0000011,0000111,0001111,0011111,$ $0111111,1111111\}$
8	$\{100,1101,010101,010111,011111,0000001,0000101,$ $0000111,00000000,00000110,00010110,00011110,$ $11111110,11111111\}$
9	$\{100,1101,0000001,0101011,0101111,0111111,00001011,$ $00001111,01010101,000000000,000001010,000001110,$ $000010101,000101010,000101110,000111110,$ $111111110,111111111\}$

Table 5.1. The minimum cost irreducible sets T_k^* that cover $\{0,1\}^k$, for $1 \leq k \leq 9$.

may obtain a lower bound on the number of external edges for universal shuffle-exchange building blocks. By Theorem 5.8, there is at most one trail between any two vertices in a universal shuffle-exchange building block. Thus we may apply Theorem 6.8, by letting $m = 2^k$ for a universal shuffle-exchange building block U_k of size 2^k, to obtain

$$2^{k+1} \leq w' \log_2 w' \tag{5.14}$$

where w' denotes the number of external edges. Again, since $w' \log_2 w'$ is a mono-

k	$\text{cost}(T_k^*) \cdot 2^k$	$\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n) \cdot 3 \cdot 2^k$		
		lower bound	actual value	upper bound
1	2	-2	0	3
2	3	0	2	8
3	5	4	6	17
4	9	12	20	38
5	14	40	50	80
6	23	100	112	165
7	40	224	238	337
8	72	480	520	687
9	127	1028	1080	1392

Table 5.2. The cost and the efficiency for T_k^* given in Table 5.1. The lower and upper bounds for the efficiency $\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n)$ are obtained from (5.13) and (5.16) respectively. The actual value of $\text{eff}(\Psi_k(T_k^*) \vdash \Psi_n)$ is obtained by counting the number of edges in $\Psi_k(T_k^*)$.

tonic increasing function, there exists a smallest integer $w'_{\min}(k)$ that satifies Inequality (5.14). Consequently, since each vertex in U_k has degree 3, the number of internal edges in any universal shuffle-exchange building block U_k is bounded above by

$$\text{E}(U_k) \leq \frac{1}{2}(3 \cdot 2^k - w'_{\min}(k)). \tag{5.15}$$

From (5.12) and (5.15), we obtain

$$\text{eff}(U_k \vdash \Psi_n) \leq 1 - \frac{w'_{\min}(k)}{3 \cdot 2^k}. \tag{5.16}$$

In addition, by (5.14) and Lemma 4.5,

$$w' \geq \frac{2^{k+1}}{k+1}. \tag{5.17}$$

From (5.16) and (5.17), we obtain the asymptotic upper bound

$$\mathrm{eff}(U_k \vdash \Psi_n) \leq 1 - \frac{2}{3(k+1)}. \tag{5.18}$$

We can obtain an asymptotic lower bound for the efficiency of the most efficient universal shuffle-exchange building block U_k as $k \to \infty$ as follows. As in Section 4.3, we define S_n^q to be the set of all q-ary strings X of length n such that one of the longest runs of zeros in X is either at the extreme left or the extreme right. Schwabe[Schw91] proved that the set of strings S_m^2 covers $\{0,1\}^{2m}$. In Section 4.8, we proved that the number of elements in S_m^2 is $< 2^{m+3}/(m+2)$. Thus

$$\mathrm{cost}(S_m^2) = \sum_{x \in S_m^2} 2^{-|x|} < \frac{8}{m+2}. \tag{5.19}$$

Consequently, by (5.13), for k even,

$$\mathrm{eff}(\Psi_k(S_{k/2}^2 \vdash \Psi_n) \geq 1 - \frac{4}{3}\mathrm{cost}(S_{k/2}^2) \geq 1 - \frac{4}{3} \cdot \frac{8}{\frac{k}{2}+2} = 1 - \frac{64}{3(k+4)}. \tag{5.20}$$

Combining (5.18) and (5.20), we obtain

$$1 - \frac{64}{3(k+4)} \leq \liminf_{k\to\infty} e_k^* \leq \limsup_{k\to\infty} e_k^* \leq 1 - \frac{2}{3(k+1)} \tag{5.21}$$

where e_k^* denotes the efficiency for the most efficient universal shuffle-exchange building block U_k.

For general VLSI decomposition for shuffle-exchange graphs Ψ_n, we may combine (5.20) with (5.9) and (5.10) to obtain

$$1 - \frac{64}{3(k+4)} \leq \liminf_{k\to\infty} e_{\mathrm{NI}}(\Psi_n; 2^k) \leq \limsup_{k\to\infty} e_{\mathrm{NI}}(\Psi_n; 2^k) \leq 1 - \frac{2}{3k} \tag{5.22}$$

and

$$(5.23) \qquad 1 - \frac{64}{3(k+4)} \leq \liminf_{k\to\infty} e_{\mathrm{I}}(\Psi_n; 2^k) \leq \limsup_{k\to\infty} e_{\mathrm{I}}(\Psi_n; 2^k) \leq 1 - \frac{2}{3k}.$$

Similar to deBruijn graphs, by comparing (5.21) with (5.22) and (5.23), it is not a severe restriction in requiring the subgraphs to be both "isomorphic" and "universal" in a VLSI decomposition of shuffle-exchange graphs Ψ_n into a large number of subgraphs.

VI. Some General Theorems

6.1. A Classification of Graphs

Each graph (with a non-empty edge set) may be classified as one of three categories:

(i) graphs with only directed edges;

(ii) graphs with only undirected edges;

(iii) graphs with both directed and undirected edges.

This chapter presents some theorems for graphs in each of these three categories. These theorems are useful for obtaining bounds for the efficiency of some VLSI decompositions as shown in Chapters IV and V. We first extend the definition of graphs to *graphs with external edges.*

Let H be a subgraph of G. Let E_{ext} be a set of edges in G such that each $e_i \in E_{\text{ext}}$ is an edge connecting a vertex not in H and a vertex in H. We call the union of H and E_{ext} a *graph with external edges.* Furthermore, if $e_i \in E_{\text{ext}}$ is a directed edge from a vertex not in H to a vertex in H, we call e_i an *input* edge. If $e_i \in E_{\text{ext}}$ is a directed edge from a vertex in H to a vertex not in H, we call e_i an *output* edge. For convenience, we also use *internal edges* to refer to all edges that are in the original subgraph H. Figure 6.1 shows a digraph with 8 input edges and 8 output edges (or a total of 16 external edges).

Theorem 6.1. Let G_{ext} and G'_{ext} be two graphs with external edges. Let $\mathcal{G}_{\text{ext}}$ and $\mathcal{G}'_{\text{ext}}$ be graphs obtained from G_{ext} and G'_{ext} respectively, by replacing all directed edges with undirected edges. If $\mathcal{G}_{\text{ext}}$ is isormorphic to $\mathcal{G}'_{\text{ext}}$, then G_{ext} and G'_{ext} have the same number of vertices, the same number of internal edges and the same number of external edges.

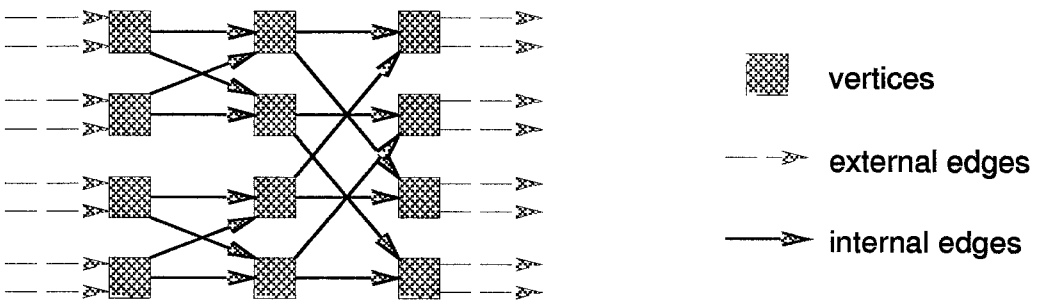

Figure 6.1. A digraph with 8 input edges and 8 output edges (or a total of 16 external edges).

Proof: The proof follows by noting that there is a one-to-one mapping on the vertices, internal edges and external edges among the four graphs G_{ext}, $\mathcal{G}_{\text{ext}}$, $\mathcal{G}'_{\text{ext}}$, and G'_{ext}. ∎

By Theorem 6.1, any theorem given in this chapter can actually be applied to any type of graphs, i.e., directed, undirected, or both. However, it usually requires a clever choice of the mapping to obtain good bounds. In addition, the proofs in Theorems 6.6–6.8 demonstrate some extensions of this mapping technique.

6.2. Directed Graph Bounds

This section introduces four theorems for directed graphs.

Theorem 6.2. (Collins [Coll92b]). Let H_{ext} be a digraph with external edges such that each vertex in H_{ext} has the same in-degree as out-degree. Let V be the vertex set of H_{ext}. If, for any two vertices $X, Y \in V$, there is at most one path from X to Y in H_{ext}, then

$$\sum_{v_i \in V} d_i \log d_i \leq w \log w$$

where d_i denotes the in-degree of vertex v_i, and w denotes the number of input edges for H_{ext}.

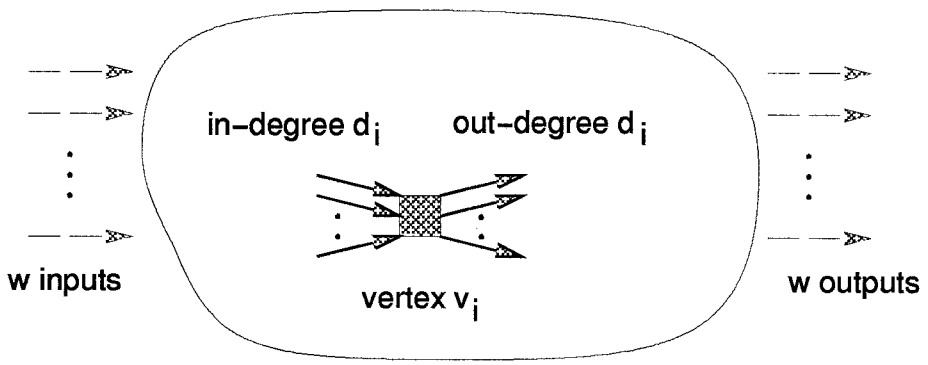

Figure 6.2. A digraph with external edges in which vertex v_i has in-degree = out-degree = d_i.

Proof (Collins [Coll92b]): Referring to Figure 6.2, let the w input edges for H_{ext} be denoted by $e_1, e_2, \ldots, e_w$. Since each vertex in H_{ext} has the same in-degree as out-degree, the number of output edges is also w. We imagine that a token is injected into H_{ext} through one of the input edges e_j and each node acts like a random switching element such that when a token arrives at the node, it is equally likely to be put out on one of its outgoing edges. The token will eventually leave H_{ext} through one of the w output edges since there is no cycle in H_{ext}. (Otherwise the existence of a cycle in H_{ext} will contradict the hypothesis that there is at most one path between two vertices in H_{ext}.) Then the entropy of the random variable that specifies the probability for the token leaving on one of the w output edges must equal $\sum_{v_i \in V} p_j(i) \log d_i$ where $p_j(i)$ is the probability of the token passing through vertex v_i (when the token is injected through input edge e_j) since there is

at most one path of getting from e_j to each one of the output edges. Thus

$$\sum_{v_i \in V} p_j(i) \log d_i \leq \log(w). \tag{6.1}$$

Now imagine that a token is injected into H_{ext} through each one of the input edges. Equation (6.1) becomes

$$\sum_{j=1}^{w} \sum_{v_i \in V} p_j(i) \log d_i \leq w \log(w)$$

$$\Rightarrow \quad \sum_{v_i \in V} \sum_{j=1}^{w} p_j(i) \log d_i \leq w \log(w)$$

$$\Rightarrow \quad \sum_{v_i \in V} d_i \log d_i \leq w \log(w)$$

since there is an average of d_i tokens passing through vertex v_i. ∎

Theorem 6.3. Let H_{ext} be a digraph with external edges such that each vertex in H_{ext} has the same in-degree as out-degree. Let V be the vertex set of H_{ext} and let l be a positive integer. If, for any two vertices $X, Y \in V$, there is at most one path of length l from X to Y in H_{ext}, then

$$(l+1) \sum_{v_i \in V} (d_i \log d_i) \leq (wl + \sum_{v_i \in V} d_i) \log(wl + \sum_{v_i \in V} d_i)$$

where d_i denotes the in-degree of vertex v_i, and w denotes the number of input edges of H_{ext}.

Proof: Given a positive integer l and a digraph (with external edges) H_{ext} with vertex set $V = \{v_1, v_2, \cdots, v_n\}$, we may construct another digraph (with external edges) $H'_{\text{ext}}(l)$ with $(l+1)(n)$ vertices, labelled by ordered pairs (v_i, y), $1 \leq i \leq n$, $0 \leq y \leq l$. There is an internal directed edge from (v_i, y_i) to (v_j, y_j) in $H'_{\text{ext}}(l)$ iff $y_j = y_i + 1$ and there exists a directed edge from v_i to v_j in H_{ext}. If there is an input edge to vertex v_i in H_{ext}, there is a corresponding input edge to each vertex (v_i, y), $1 \leq y \leq l$ in $H'_{\text{ext}}(l)$. Similarly, if there is an output edge from vertex v_i in

H_{ext}, there is a corresponding output edge from each vertex (v_i, y), $0 \leq y \leq l-1$ in $H'_{\text{ext}}(l)$. In addition, there are d_i input edges to vertex $(v_i, 0)$ and there are d_i output edges from vertex (v_i, l) in $H'_{\text{ext}}(l)$. An example is shown in Figure 6.3 with a subgraph (with external edges) H_{ext} of deBruijn graph B_3 and the corresponding $H'_{\text{ext}}(l=3)$.

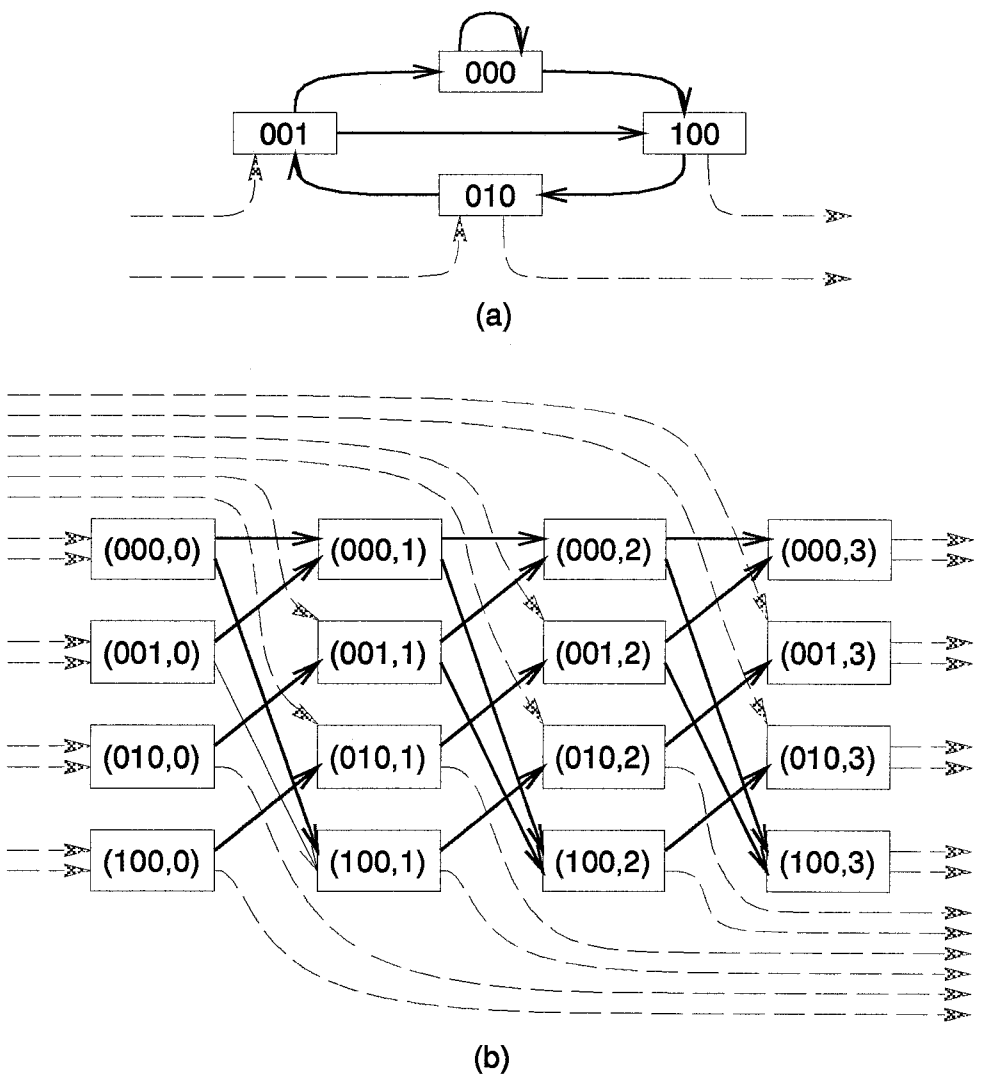

Figure 6.3. (a) A subgraph (with external edges) H_{ext} of deBruijn graph B_3. (b) The corresponding $H'_{\text{ext}}(l=3)$.

Since there is at most one path of length l from any vertex X to any vertex Y in H_{ext}, there is at most one path from vertex $(v_i, 0)$ to vertex (v_j, l), where $v_i, v_j \in V$, in graph $H'_{\text{ext}}(l)$. It follows that there is never more than one path between any two vertices in $H'_{\text{ext}}(l)$ and thus we may apply Theorem 6.2 on $H'_{\text{ext}}(l)$. By noting that there are $\sum_{v_i \in V} d_i$ input edges entering the column 0 and there are w input edges entering each of the other l columns, we obtain the desired inequality

$$(l+1)\sum_{v_i \in V}(d_i \log d_i) \leq (wl + \sum_{v_i \in V} d_i)\log(wl + \sum_{v_i \in V} d_i) \tag{6.2}$$

by Theorem 6.2. ∎

Theorems 6.2 and 6.3 can be generalized to other digraphs that contain vertices v_i with different in-degrees $d_{i_{\text{in}}}$ and out-degrees $d_{i_{\text{out}}}$. If $d_{i_{\text{in}}} < d_{i_{\text{out}}}$ we may add $d_{i_{\text{out}}} - d_{i_{\text{in}}}$ phantom input edges to vertex v_i. Similarly, if $d_{i_{\text{out}}} < d_{i_{\text{in}}}$, we may add $d_{i_{\text{in}}} - d_{i_{\text{out}}}$ phantom output edges to vertex v_i. Then we may apply Theorems 6.2 and 6.3 on this new graph with "phantom" edges. The generalized theorems are stated in the following. Note that Theorems 6.4 and 6.5 degenerate back to Theorems 6.2 and 6.3 when $d_{i_{\text{in}}} = d_{i_{\text{out}}}$ for each vertex v_i in H_{ext}.

Theorem 6.4. Let H_{ext} be a digraph with external edges such that vertex v_i in H_{ext} has in-degree $d_{i_{\text{in}}}$ and out-degree $d_{i_{\text{out}}}$. Let $d_{i\max} = \max(d_{i_{\text{in}}}, d_{i_{\text{out}}})$ and $d_{i\min} = \min(d_{i_{\text{in}}}, d_{i_{\text{out}}})$. Let w_{in} (w_{out}) denote the number of input (output) edges for graph H_{ext}. Let V be the vertex set of H_{ext}. If, for any two vertices $X, Y \in V$, there is at most one path from X to Y in H_{ext}, then

$$\sum_{v_i \in V} d_{i\max} \log d_{i\max} \leq w \log w$$

where

$$w = \frac{1}{2}\left(w_{\text{in}} + w_{\text{out}} + \sum_{v_i \in V}(d_{i\max} - d_{i\min})\right).$$

Theorem 6.5. Let H_{ext} be a digraph with external edges such that vertex v_i in H_{ext} has in-degree $d_{i_{\text{in}}}$ and out-degree $d_{i_{\text{out}}}$. Let $d_{i\max} = \max(d_{i_{\text{in}}}, d_{i_{\text{out}}})$ and

$d_{i\min} = \min(d_{i\text{in}}, d_{i\text{out}})$. Let w_{in} (w_{out}) denote the number of input (output) edges for graph H_{ext}. Let V be the vertex set of H_{ext} and let l be a positive integer. If, for any two vertices $X, Y \in V$, there is at most one path of length l from X to Y in H_{ext}, then

$$(l+1)\sum_{v_i \in V} d_{i\max} \log d_{i\max} \leq (wl + \sum_{v_i \in V} d_{i\max}) \log(wl + \sum_{v_i \in V} d_{i\max})$$

where

$$w = \frac{1}{2}\left(w_{\text{in}} + w_{\text{out}} + \sum_{v_i \in V}(d_{i\max} - d_{i\min})\right).$$

6.3. Undirected Graph Bounds

Given a set of graphs $\{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_n\}$ where $\mathcal{G}_i$ has l_i vertices with vertex set $V_i = \{(i,0),(i,1),\cdots,(i,l_i)\}$, we may form another graph G with $l_1 l_2 \cdots l_n$ vertices as follows. The vertices of G are labelled by n-tuples $a_1 a_2 \cdots a_n$, where $0 \leq a_i \leq l_i - 1$. Two vertices $x_1 x_2 \cdots x_n$ and $y_1 y_2 \cdots y_n$ are connected by an undirected edge iff their labels are different in exactly one position, say position j, such that vertices (j, x_j) and (j, y_j) are connected in $\mathcal{G}_j$. Then the graph G can be viewed as a Cartesian "product" of the graphs $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n$.

All undirected graphs can be considered as a "product" graph with $n = 1$. Besides this trivial case, many graphs are "product" graphs with $n > 1$. In particular, the n-dimensional hyperplane $\Gamma_n(l_1, l_2, \ldots, l_n)$ (defined in Section 3.3) is a product of n complete graphs K_{l_i}, $1 \leq i \leq n$. As another example, the d-dimensional mesh $M_d(n)$ with wrap-around is a product of n rings where each ring contains d vertices. In Theorem 6.6, we use the binary n-cube (defined in Section 3.2) as an example to demonstrate a technique that can be applied to any "product" graph.

We first begin with a definition. A subgraph (with external edges) H_{ext} is called an *induced subgraph with external edges* if H_{ext} contains all internal and external

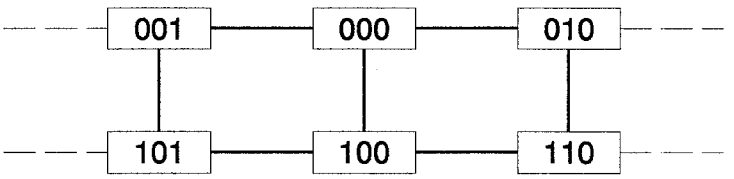

Figure 6.4. An induced subgraph with external edges of the binary n-cube Γ_3.

edges that have at least one vertex in H_{ext}. Figure 6.4 shows an induced subgraph (with external edges) of the binary n-cube Γ_3.

Theorem 6.6. Let H_{ext} be an induced subgraph with external edges of the binary n-cube Γ_n. Then

$$2(n+1)(m) \leq (w' + 2m)\log_2(w' + 2m)$$

where w' denotes the number of external edges and m denotes the number of vertices in H_{ext}.

Proof: Referring to Figure 3.3, we can classify the edges of a binary n-cube (or a "product" graph in general) into different dimensions. For example, in Γ_3, the edge connecting vertices 000 and 010 can be viewed as an edge in the second dimension. In general, an edge connecting two vertices that are different in the rth position can be classified as an edge in the rth dimension. With this classification, there is a unique path between any two vertices in Γ_n such that the traversed edges are in increasing dimensions, i.e., if the traversed edges are in the sequence $E_1, E_2, \ldots, E_l$, then $\dim(E_1) < \dim(E_2) < \cdots < \dim(E_l)$ where $\dim(E_i)$ denotes the dimension of edge E_i. We will use this idea to prove this theorem.

Given an induced subgraph (with external edges) H_{ext} with m vertices of Γ_n, we may construct another digraph (with external edges) H'_{ext} with $(n+1)(m)$ vertices as follows. Let $V = \{v_1, v_2, \cdots, v_m\}$ be the vertex set of H_{ext}. We label the $(n+1)(m)$

vertices of H'_{ext} by the ordered pairs (v_i, y), $1 \leq i \leq m$, $0 \leq y \leq n$. If there is an undirected edge connecting vertices v_i and v_j in the rth dimension in H_{ext}, then we add, in H'_{ext}, directed edges from $(v_i, r-1)$ to (v_j, r) and from $(v_j, r-1)$ to (v_i, r). If there is an external edge connecting to vertex v_i in the rth dimension in H_{ext}, then we add, in H'_{ext}, an input edge to vertex (v_i, r) and an output edge from vertex $(v_i, r-1)$. For each of the vertices $(v_i, 0)$ in H'_{ext}, we add two input edges. For each of the vertices (v_i, n) in H'_{ext}, we add two output edges. There are no other edges to be added besides the edges described above. The graph obtained by the above procedure is the desired digraph H'_{ext} which has the same in-degree as out-degree on each of its vertices. Figure 6.5 shows the digraph H'_{ext} constructed from the induced subgraph shown in Figure 6.4 using the above definition.

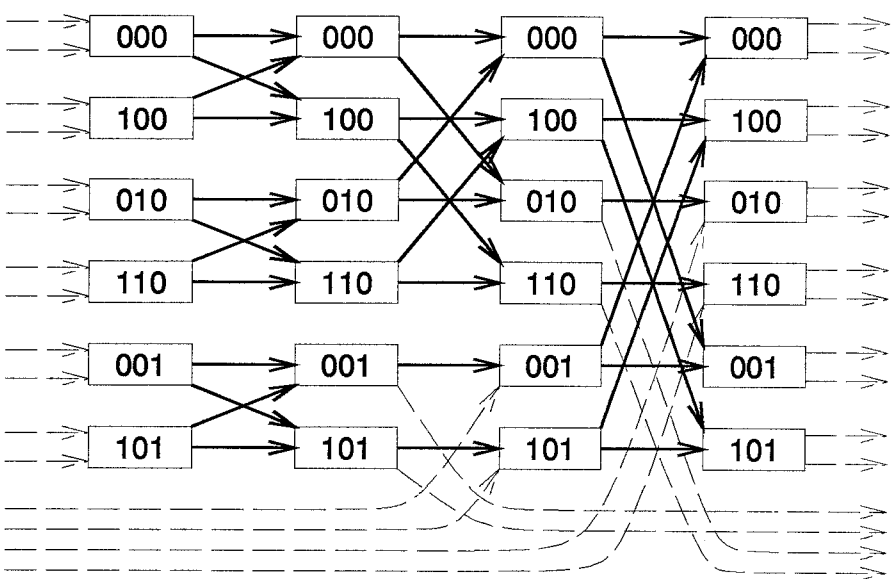

Figure 6.5. The corresponding digraph H'_{ext} constructed from the induced subgraph shown in Figure 6.4.

We mentioned at the beginning of this proof that there is a unique path between any two vertices in Γ_n such that the traversed edges are in increasing dimensions. As a result, the constructed graph H'_{ext} satisfies the requirement in Theorem 6.2 that there is at most one path between any two vertices in H'_{ext}. Thus we may apply Theorem 6.2 on H'_{ext} in which each vertex has in-degree (= out-degree) two. Note that for each external edge in H_{ext}, there is a corresponding input edge in H'_{ext}. In addition, there are 2m input edges entering column 0. Therefore we obtain the inequality

$$2(n+1)(m) \leq (w'+2m)\log_2(w'+2m) \tag{6.3}$$

by applying Theorem 6.2 on H'_{ext}. ∎

The lower bound on the number of external edges given in Theorem 6.6 is not a tight bound. It can be shown that the bound (6.3) is weaker than the bound (3.4) given in Section 3.2. However, (3.4) holds only for binary n-cubes Γ_n while the technique illustrated in the proof of Theorem 6.6 can be applied to any "product" graph.

6.4. Bounds on Graphs with Both Directed and Undirected Edges

For graphs with both directed and undirected edges, there are many ways to apply Theorem 6.2 to obtain lower bounds for the number of external edges. We will illustrate a technique similar to the one used in the proof of Theorem 6.6. In Theorems 6.7 and 6.8, we prove two lower bounds on induced subgraphs with external edges (defined in Section 6.3) of the shuffle-exchange graph Ψ_n (defined in Section 5.1). Figure 6.6 shows an induced subgraph (with external edges) of the shuffle-exchange graph Ψ_3.

Theorem 6.7. Let H_{ext} be an induced subgraph with external edges of the shuffle-exchange graph Ψ_n. Then

$$2(l+1)(m) \leq (w'l+2m)\log_2(w'l+2m) \qquad \forall l \leq n$$

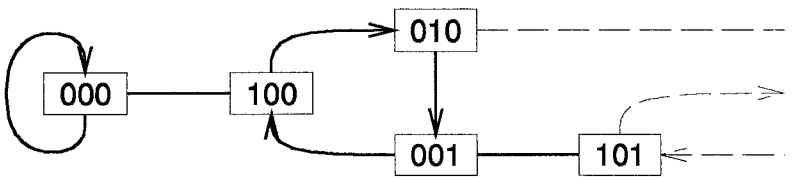

Figure 6.6. An induced subgraph with external edges of the shuffle-exchange graph Ψ_3.

where w' denotes the number of external edges and m denotes the number of vertices in H_{ext}.

Proof: By Theorem 5.2, if X and Y are vertices in Ψ_n and l is a positive integer with $l \leq n$, then there is a unique trail of net length l that begins with a shuffle edge from X to Y. By noting the similarity of this property with the digraph stated in Theorem 6.3, we will use a similar method to prove this theorem.

Given a positive integer l and an induced subgraph (with external edges) H_{ext} with m vertices of Ψ_n, we may construct another digraph (with external edges) $H'_{\text{ext}}(l)$ with $(2l+1)(m)$ vertices as follows. Let $V = \{v_1, v_2, \cdots, v_m\}$ be the vertex set of H_{ext}. We label the $(2l+1)(m)$ vertices of $H'_{\text{ext}}(l)$ by the ordered pairs (v_i, y), $1 \leq i \leq m$, $0 \leq y \leq 2l$. If there exists a shuffle edge from v_i to v_j in H_{ext}, then we add, in $H'_{\text{ext}}(l)$, directed edges from $(v_i, 2y)$ to $(v_j, 2y+1)$ and from $(v_i, 2y)$ to $(v_j, 2y+2)$, for each integer y such that $0 \leq y \leq l-1$. If there exists an exchange edge connecting vertices v_i and v_j in H_{ext}, then we add, in $H'_{\text{ext}}(l)$, directed edges from $(v_i, 2y+1)$ to $(v_j, 2y+2)$ and from $(v_j, 2y+1)$ to $(v_i, 2y+2)$, for each integer y such that $0 \leq y \leq l-1$. If there is an input edge to vertex v_i in H_{ext}, we add a corresponding input edge to each vertex (v_i, y), $1 \leq y \leq 2l$ in $H'_{\text{ext}}(l)$. If there is an output edge from vertex v_i in H_{ext}, we add two corresponding output edges from each vertex $(v_i, 2y)$, $0 \leq y \leq l-1$ in $H'_{\text{ext}}(l)$. If there is an external exchange

edge with endvertex v_i in H_{ext}, then we add an input edge to each vertex $(v_i, 2y)$, $1 \leq y \leq l$, and we add an output edge from each vertex $(v_i, 2y+1)$, $0 \leq y \leq l-1$. For each of the vertices $(v_i, 0)$ in $H'_{ext}(l)$, we add two input edges. For each of the vertices $(v_i, 2l)$ in $H'_{ext}(l)$, we add two output edges. There are no other edges to be added besides the edges described above. The graph obtained by the above procedure is the desired digraph $H'_{ext}(l)$ which has the same in-degree as out-degree on each of its vertices. Figure 6.7 shows the digraph $H'_{ext}(l=3)$ constructed from the induced subgraph shown in Figure 6.6 using the above definition.

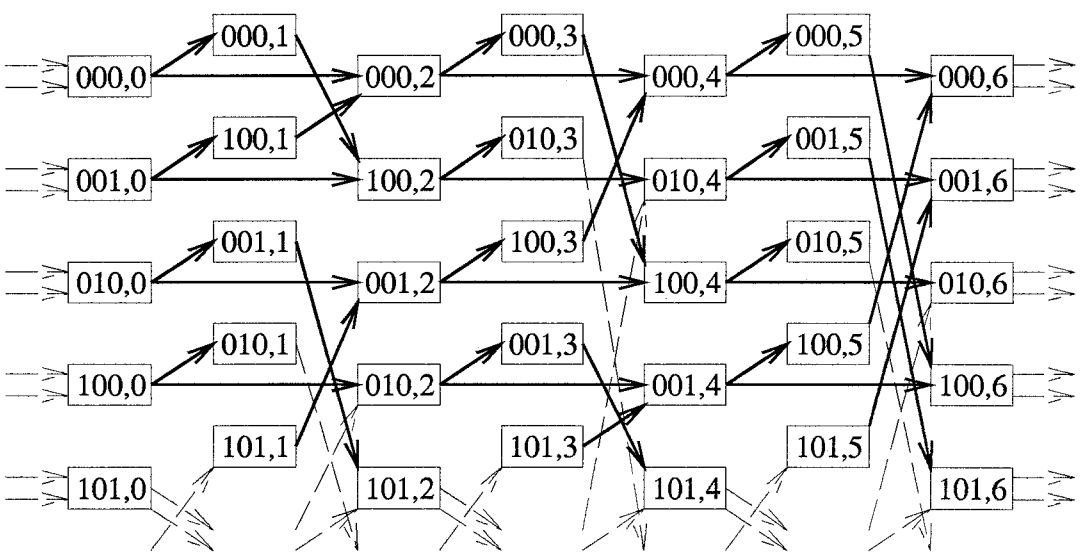

Figure 6.7. The corresponding digraph $H'_{ext}(l = 3)$ constructed from the induced subgraph shown in Figure 6.6.

By Theorem 5.2, if $l \leq n$, there is at most one path between any two vertices in $H'_{ext}(l)$. Thus we may again apply Theorem 6.2 on $H'_{ext}(l)$ since each vertex has the same in-degree as out-degree in $H'_{ext}(l)$. There are $(m)(l+1)$ vertices with in-degree two and ml vertices with in-degree one. There are 2m input edges entering column 0

and $w'l$ input edges entering the remaining $2l$ columns. Therefore we obtain the inequality

$$2(l+1)(m) \leq (w'l+2m)\log_2(w'l+2m) \qquad \forall l \leq n \tag{6.4}$$

by Theorem 6.2. ∎

The same technique shown in the proof of Theorem 6.7 can be used to modify Theorem 6.2. We again use the shuffle-exchange graph as an example for illustration.

Theorem 6.8. Let H_{ext} be an induced subgraph with external edges of the shuffle-exchange graph Ψ_n. If there is at most one trail between any two vertices in H_{ext}, then

$$2m \leq w' \log_2 w'$$

where w' denotes the number of external edges and m denotes the number of vertices in H_{ext}.

Proof: Similar to the proof of Theorem 6.7, we will construct another digraph (with external edges) H'_{ext} such that H'_{ext} will satisfy the unique path requirement in Theorem 6.2. We will use the induced subgraph (with external edges) shown in Figure 6.8 to demonstrate the construction. Note that there is at most one trail between any two vertices in the induced subgraph shown in Figure 6.8.

To begin with, let $V = \{v_1, v_2, \cdots, v_m\}$ be the vertex set of H_{ext}. We then construct H'_{ext} with $2m$ vertices as follows. We label the vertices of H'_{ext} by the ordered pairs (v_i, y), $1 \leq i \leq m$, $y \in \{0,1\}$. If there exists a shuffle edge from v_i to v_j in H_{ext}, then we add, in H'_{ext}, directed edges from $(v_i,1)$ to $(v_j,0)$ and from $(v_i,1)$ to $(v_j,1)$. If there exists an exchange edge connecting vertices v_i and v_j in H_{ext}, then we add, in H_{ext}, directed edges from $(v_i,0)$ to $(v_j,1)$ and from $(v_j,0)$ to $(v_i,1)$. If there is an input edge to vertex v_i in H_{ext}, we add a corresponding input edge to each vertex (v_i,y), $y \in \{0,1\}$ in H_{ext}. If there is an output edge from vertex

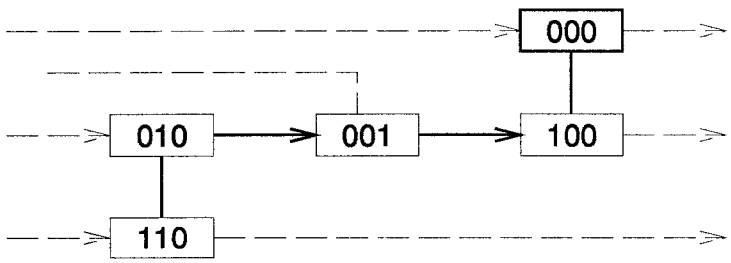

Figure 6.8. An induced subgraph (with external edges) H_{ext} of Ψ_3 such that there is at most one trail between any two vertices in H_{ext}.

v_i in H_{ext}, we add two corresponding output edges from vertex $(v_i, 1)$ in H'_{ext}. If there is an external exchange edge with endvertex v_i in H_{ext}, then we add an input edge to vertex $(v_i, 1)$ and an output edge from vertex $(v_i, 0)$. There are no other edges to be added besides the edges described above. The graph obtained by the above procedure is the desired digraph H'_{ext} in which any two vertices have at most one path between them. Figure 6.9 shows the digraph H'_{ext} constructed from the induced subgraph shown in Figure 6.8 using the above definition.

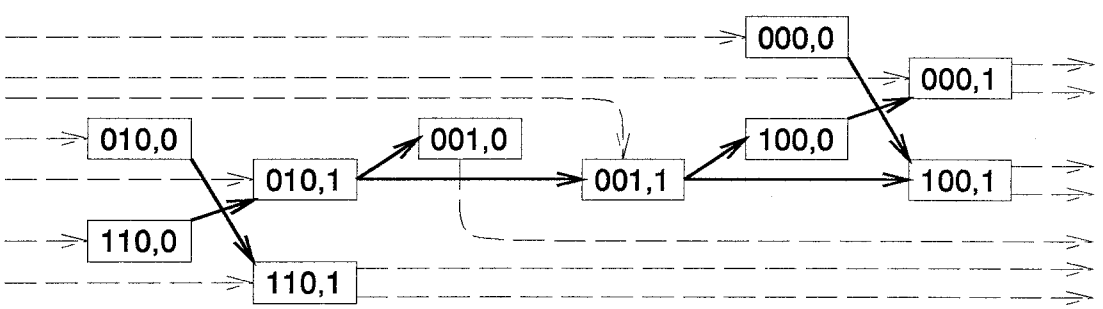

Figure 6.9. The corresponding digraph H'_{ext} constructed from the induced subgraph shown in Figure 6.8.

In H'_{ext}, there are m vertices with in-degree (= out-degree) two and another m vertices with in-degree (= out-degree) one. There are w' input edges and also w' output edges. Therefore, we obtain the desired inequality

$$2m \leq w' \log_2 w' \tag{6.5}$$

by Theorem 6.2. ∎

VII. Conclusions and Summary

With the recent advances in VLSI technology, the design of VLSI circuits is becoming limited by the number of available pins rather than by the available chip area. The results in this thesis provide a guideline on how to decompose several families of graphs to make full use of the limited number of available pins. As an example, the methods described in Chapter 4 can be used to design a 64-chip VLSI decomposition of the deBruijn graph B_{13} with efficiency 0.754. This chip, shown in Figure 4.10, is being used by JPL design engineers to build a single-board fully parallel Viterbi decoder for the *Galileo* code—a constraint length 15, rate 1/4 convolutional code. In contrast, earlier results[Coll88,Coll92a] led to a less efficient 256-chip VLSI decomposition of B_{13} with efficiency 0.563, which was used to design a multi-board decoder for the Galileo code.

In the following sections, we summarize the major results presented in this thesis.

7.1. Notations

(1) For general C-chip VLSI decompositions:

$\mathcal{D}_C^*(G \rightarrow k_1, k_2, \ldots, k_C)$ denotes the set of the most efficient C-chip VLSI decomposition for graph G into chip sizes $k_1, k_2, \ldots, k_C$.

$\mathcal{D}_C^+(G \rightarrow k_1, k_2, \ldots, k_C)$ denotes the set of the most efficient known C-chip VLSI decomposition for graph G into chip sizes $k_1, k_2, \ldots, k_C$.

$(H_1, H_2, \cdots, H_C \mapsto G)$ represents the VLSI decomposition for graph G into subgraphs $H_1, H_2, \ldots, H_C$.

$\text{eff}(H_1, H_2, \cdots, H_C \mapsto G)$ denotes the efficiency of the VLSI decomposition for graph G into subgraphs $H_1, H_2, \ldots, H_C$.

$e^*(G \rightarrow k_1, k_2, \ldots, k_C)$ denotes the efficiency of the most efficient C-chip VLSI decomposition for graph G into chip sizes $k_1, k_2, \ldots, k_C$.

(2) For VLSI decompositions into equal chip sizes:

$\mathcal{D}^*_{\text{NI}}(G \rightarrow k)$ (or $\mathcal{D}^+_{\text{NI}}(G \rightarrow k)$) denotes the set of the most efficient (or the most efficient known) VLSI decomposition for graph G into subgraphs, that are not required to be isomorphic, of equal sizes k.

$\mathcal{D}^*_{\text{I}}(G \rightarrow k)$ (or $\mathcal{D}^+_{\text{I}}(G \rightarrow k)$) denotes the set of the most efficient (or the most efficient known) VLSI decomposition for graph G into isomorphic subgraphs of size k.

$(H \vdash G)$ represents the VLSI decomposition for graph G into isomorphic subgraphs H.

$\text{eff}(H \vdash G)$ denotes the efficiency of the VLSI decomposition for graph G into isomorphic subgraphs H.

$e_{\text{NI}}(G; k)$ denotes the efficiency of the most efficient VLSI decomposition for graph G into subgraphs (that are not required to be isomorphic) of size k.

$e_{\text{I}}(G; k)$ denotes the efficiency of the most efficient VLSI decomposition for graph G into isomorphic subgraphs of size k.

(3) For universal building blocks, we use similar notations, i.e., notations obtained by replacing G by $\{G_n\}$.

7.2. Complete Graph K_n

(1) For general C-chip VLSI decompositions:

$\mathcal{D}_C^*(K_n \to k_1, k_2, \ldots, k_C) = \{(K_{k_1}, K_{k_2}, \cdots, K_{k_C} \mapsto K_n)\}$. (Note that there is an equal sign in this equation since the optimal decomposition is unique.)

$e^*(K_n \to k_1, k_2, \ldots, k_C) = \text{eff}(K_{k_1}, K_{k_2}, \cdots, K_{k_C} \mapsto K_n) = \frac{\binom{k_1}{2}+\binom{k_2}{2}+\cdots+\binom{k_C}{2}}{\binom{n}{2}}$.

(2) For VLSI decompositions into equal chip sizes:

$\mathcal{D}_{\text{NI}}^*(K_n \to k) = \mathcal{D}_{\text{I}}^*(K_n \to k) = \{(K_k \vdash K_n)\}$.

$e_{\text{I}}(K_n; k) = e_{\text{NI}}(K_n; k) = \text{eff}(K_k \vdash K_n) = \frac{k-1}{n-1}$.

(3) For VLSI decompositions into universal building blocks:

$\mathcal{D}_{\text{NI}}^*(\{K_n\}_{kN|n, \text{ where } N \in Z^+}; k)$
$= \mathcal{D}_{\text{I}}^*(\{K_n\}_{kN|n, \text{ where } N \in Z^+}; k) = \{(K_k \vdash \{K_n\}_{kN|n})\}$.

$e_{\text{I}}(\{K_n\}_{kN|n, \text{ where } N \in Z^+}; k) = e_{\text{NI}}(\{K_n\}_{kN|n, \text{ where } N \in Z^+}; k)$
$= \text{eff}(K_k \vdash \{K_n\}_{kN|n}) = \frac{k-1}{n-1}$.

7.3. n-Dimensional Hyperplane $\Gamma_n(l)$ and Binary n-Cube $\Gamma_n = \Gamma_n(l=2)$

(1) For general C-chip VLSI decompositions:

$$\mathcal{D}^*_C(\Gamma_n(l) \rightarrow k_1, k_2, \ldots, k_C) = \text{unknown in general.}$$

$$e^*(\Gamma_n(l) \rightarrow k_1, k_2, \ldots, k_C) \leq \frac{2}{(n)(l-1)(l^n)} \sum_{i=1}^{C} \left(\sum_{j=0}^{k_i - 1} \text{weight}(i) \right)$$

where weight(i) is defined in Section 3.3.

$$\mathcal{D}^*_C(\Gamma_n(l) \rightarrow l^{k_1}, l^{k_2}, \ldots, l^{k_C}) \supseteq \{(\Gamma_{k_1}(l), \Gamma_{k_2}(l), \cdots, \Gamma_{k_C}(l) \mapsto \Gamma_n(l))\}.$$

$$\text{eff}(\Gamma_{k_1}(l), \Gamma_{k_2}(l), \cdots, \Gamma_{k_C}(l) \mapsto \Gamma_n(l)) = \frac{1}{n \cdot l^n} \sum_{i=1}^{C} \left(k_i \cdot l^{k_i} \right).$$

(2) For VLSI decompositions into equal chip sizes:

$$\mathcal{D}^*_{\text{NI}}(\Gamma_n(l) \rightarrow l^k) \supseteq \mathcal{D}^*_{\text{I}}(\Gamma_n(l) \rightarrow l^k) \supseteq (\Gamma_k(l) \vdash \Gamma_n(l)).$$

$$e_{\text{I}}(\Gamma_n(l); l^k) = e_{\text{NI}}(\Gamma_n(l); l^k) = \frac{k}{n}.$$

(3) For VLSI decompositions into universal building blocks:

$$\mathcal{D}^*_{\text{NI}}(\{\Gamma_n(l)\}_{n \geq N, \text{ where } N \geq k}; l^k) \supseteq \mathcal{D}^*_{\text{I}}(\{\Gamma_n(l)\}_{n \geq N, \text{ where } N \geq k}; l^k)$$
$$\supseteq (\Gamma_k(l) \vdash \{\Gamma_n(l)\}_{n \geq N}).$$

$$e_{\text{I}}(\{\Gamma_n(l)\}_{n \geq N, \text{ where } N \geq k}; l^k) = e_{\text{NI}}(\{\Gamma_n(l)\}_{n \geq N, \text{ where } N \geq k}; l^k) = \frac{k}{n}.$$

7.4. n-Dimensional Hyperplane† $\Gamma_n(l_1, l_2, \ldots, l_n)$

(1) For general C-chip VLSI decompositions:

$$\mathcal{D}_C^*(\Gamma_n(l_1, l_2, \ldots, l_n) \rightarrow k_1, k_2, \ldots, k_C) = \text{unknown in general.}$$

$$e^*(\Gamma_n(l_1, l_2, \ldots, l_n) \rightarrow k_1, k_2, \ldots, k_C)$$
$$\leq \frac{2}{(l_1+l_2+\cdots+l_n-n)(l_1 l_2 \cdots l_n)} \sum_{i=1}^{C} \left(\sum_{j=0}^{k_i-1} \text{weight}(i) \right)$$
where weight(i) is defined in Section 3.3.

(2) For VLSI decompositions into equal chip sizes:

$$\mathcal{D}_{\text{NI}}^*(\Gamma_n(l_1, l_2, \ldots, l_n) \rightarrow l_1 l_2 \cdots l_k)$$
$$\supseteq \mathcal{D}_{\text{I}}^*(\Gamma_n(l_1, l_2, \ldots, l_n) \rightarrow l_1 l_2 \cdots l_k)$$
$$\supseteq (\Gamma_k(l_1, l_2, \ldots, l_k) \vdash \Gamma_n(l_1, l_2, \ldots, l_n)).$$

$$e_{\text{I}}(\Gamma_n(l_1, l_2, \ldots, l_n); l_1 l_2 \cdots l_k)$$
$$= e_{\text{NI}}(\Gamma_n(l_1, l_2, \ldots, l_n); l_1 l_2 \cdots l_k) = \frac{l_1+l_2+\cdots+l_k-k}{l_1+l_2+\cdots+l_n-n}.$$

(3) For VLSI decompositions into universal building blocks:

$$\mathcal{D}_{\text{NI}}^*(\{\Gamma_n(l_1, l_2, \ldots, l_n)\}_{n \geq N, \text{ where } N \geq k}; l_1 l_2 \cdots l_k)$$
$$\supseteq \mathcal{D}_{\text{I}}^*(\{\Gamma_n(l_1, l_2, \ldots, l_n)\}_{n \geq N, \text{ where } N \geq k}; l_1 l_2 \cdots l_k)$$
$$\supseteq (\Gamma_k(l_1, l_2, \ldots, l_k) \vdash \{\Gamma_n(l_1, l_2, \ldots, l_n)\}_{n \geq N}).$$

$$e_{\text{I}}(\{\Gamma_n(l_1, l_2, \ldots, l_n)\}_{n \geq N, \text{ where } N \geq k}; l_1 l_2 \cdots l_k)$$
$$= e_{\text{NI}}(\{\Gamma_n(l_1, l_2, \ldots, l_n)\}_{n \geq N, \text{ where } N \geq k}; l_1 l_2 \cdots l_k) = \frac{l_1+l_2+\cdots+l_k-k}{l_1+l_2+\cdots+l_n-n}.$$

† We assume $l_1 \geq l_2 \geq \cdots \geq l_n$ without loss of generality.

7.5. d-Dimensional Mesh $M'_d(n)$ Without Wrap-Around

(1) For general C-chip VLSI decompositions:

$$\mathcal{D}^*_C(M'_d(n) \rightarrow k_1, k_2, \ldots, k_C) = \text{unknown in general.}$$

$$e^*(M'_d(n) \rightarrow k_1, k_2, \ldots, k_C) \leq \frac{1}{(n-1)(n^d)} \sum_{i=1}^{C} \left(k_i - {k_i}^{\frac{d-1}{d}} \right).$$

$$\mathcal{D}^*_C(M'_d(n) \rightarrow {k_1}^d, {k_2}^d, \ldots, {k_C}^d) = \{(M'_d(k_1), M'_d(k_2), \ldots, M'_d(k_C) \mapsto M'_d(n))\}.$$

$$\text{eff}(M'_d(k_1), M'_d(k_2), \ldots, M'_d(k_C) \mapsto M'_d(n)) = \frac{1}{(n-1)(n^d)} \sum_{i=1}^{C} \left({k_i}^d - {k_i}^{d-1} \right).$$

(2) For VLSI decompositions into equal chip sizes:

$$\mathcal{D}^*_{\text{NI}}(M'_d(n) \rightarrow k^d) = \mathcal{D}^*_{\text{I}}(M'_d(n) \rightarrow k^d) = \{(M'_d(k) \vdash M'_d(n))\}.$$

$$e_{\text{I}}(M'_d(n); k^d) = e_{\text{NI}}(M'_d(n); k^d) = \frac{(n)(k-1)}{(n-1)(k)}.$$

(3) For VLSI decompositions into universal building blocks:

$$\mathcal{D}^*_{\text{NI}}(\{M'_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k)$$
$$= \mathcal{D}^*_{\text{I}}(\{M'_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k) = \{(M'_d(k) \vdash \{M'_d(n)\}_{kN|n})\}.$$

$$e_{\text{I}}(\{M'_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k) = e_{\text{NI}}(\{M'_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k) = \frac{(n)(k-1)}{(n-1)(k)}.$$

7.6. d-Dimensional Mesh $M_d(n)$ With Wrap-Around

(1) For general C-chip VLSI decompositions:

$$\mathcal{D}_C^*(M_d(n) \rightarrow k_1, k_2, \ldots, k_C) = \text{unknown in general.}$$

$$e^*(M_d(n) \rightarrow k_1, k_2, \ldots, k_C) \leq \frac{1}{n^d} \sum_{i=1}^{C} \left(\frac{n+1}{n} \cdot k_i - k_i^{\frac{d-1}{d}} \right).$$

(2) For VLSI decompositions into equal chip sizes:

$$\mathcal{D}_{\text{I}}^+(M_d(n) \rightarrow k^d) \supseteq \{(M'_d(k) \vdash M_d(n))\}.$$

$$\text{eff}(M'_d(k) \vdash M_d(n)) = \frac{k-1}{k}.$$

$$\frac{k-1}{k} \leq e_{\text{I}}(M_d(n); k^d) \leq e_{\text{NI}}(M_d(n); k^d) \leq \frac{k-1}{k} + \frac{1}{n}.$$

(3) For VLSI decompositions into universal building blocks:

$$\mathcal{D}_{\text{NI}}^*(\{M_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k^d)$$
$$\supseteq \mathcal{D}_{\text{I}}^*(\{M_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k^d) \supseteq \{(M'_d(k) \vdash \{M_d(n)\}_{kN|n})\}.$$

$$e_{\text{I}}(\{M_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k^d) = e_{\text{NI}}(\{M_d(n)\}_{kN|n, \text{ where } N \in Z^+}; k^d) = \frac{k-1}{k}.$$

7.7. DeBruijn Graph B_n^q

(1) For general C-chip VLSI decompositions:

$\mathcal{D}_C^*(B_n^q \to k_1, k_2, \ldots, k_C)$ = unknown in general.

$e^*(B_n^q \to k_1, k_2, \ldots, k_C) \leq 1 - \frac{1}{q^{n+1}} \sum_{1 \leq i \leq C} w_{\text{lowerbound}}(q, k_i, n)$

where $w_{\text{lowerbound}}(q, k_i, n)$ is defined in Equation (4.5).

(2) For VLSI decompositions into equal chip sizes:

$\mathcal{D}_{\text{I}}^+(B_n^q \to q^k) \supseteq \{(H_k^q \vdash B_n^q)\}$

where H_k^q is defined in the proof of Theorem 4.9.

$1 - \frac{2q^2}{k+3} < e_{\text{I}}(B_n^q; q^k) \leq e_{\text{NI}}(B_n^q; q^k) \leq 1 - \frac{w_{\text{lowerbound}}(q, q^k, n)}{q^{k+1}}$.

(3) For VLSI decompositions into universal building blocks:

$\mathcal{D}_{\text{I}}^+(\{B_n^q\}_{n \geq N, \text{ where } N \geq k}; q^k) \supseteq \{(H_k^q \vdash B_n^q)\}$.

$1 - \frac{2q^2}{k+3} < e_{\text{I}}(\{B_n^q\}_{n \geq N, \text{ where } N \geq k}; q^k) \leq 1 - \frac{w_{\min}(q,k)}{q^{k+1}}$

where $w_{\min}(q, k)$ is defined in the paragraph before Equation (4.17).

7.8. Binary Shuffle-Exchange Graph Ψ_n

(1) For general C-chip VLSI decompositions:

$\mathcal{D}_C^*(\Psi_n \to k_1, k_2, \ldots, k_C) =$ unknown in general.

$e^*(\Psi_n \to k_1, k_2, \ldots, k_C) \leq 1 - \frac{1}{3 \cdot 2^n} \sum_{1 \leq i \leq C} w_{\text{lowerbound}}(2, k_i, n)$

where $w_{\text{lowerbound}}(2, k_i, n)$ is defined in Equation (4.5).

(2) For VLSI decompositions into equal chip sizes:

$\mathcal{D}_{\text{I}}^+(\Psi_n \to 2^k) \supseteq \{(\Psi_k(T_k) \vdash \Psi_n)\}$

where $\Psi_k(T_k)$ is defined in Theorem 5.9.

$1 - \frac{64}{3(k+4)} \leq e_{\text{I}}(\Psi_n; 2^k) \leq e_{\text{NI}}(\Psi_n; 2^k) \leq 1 - \frac{w_{\text{lowerbound}}(2, 2^k, n)}{3 \cdot 2^k}$.

(3) For VLSI decompositions into universal building blocks:

$\mathcal{D}_{\text{I}}^+(\{\Psi_n\}_{n \geq N, \text{ where } N \geq k}; 2^k) \supseteq \{(\Psi_k(T_k) \vdash \Psi_n)\}$.

$1 - \frac{64}{3(k+4)} \leq e_{\text{I}}(\{\Psi_n\}_{n \geq N, \text{ where } N \geq k}; 2^k) \leq 1 - \frac{w'_{\min}(k)}{3 \cdot 2^k}$.

where $w'_{\min}(k)$ is defined in the paragraph before Equation (5.15).

VIII. References

[Bern67] A.J. Bernstein, "Maximally Connected Arrays on the N-Cube," *SIAM J. Appl. Math.*, (1967), pp. 1485–1489.

[Coll92a] O. Collins, S. Dolinar, R. McEliece, and F. Pollara, "A VLSI Decomposition of the DeBruijn Graph," *J. Assoc. Comp. Mach., in press.*

[Coll92b] O. Collins, Personal Communications.

[Coll88] O. Collins, F. Pollara, S. Dolinar, and J. Statman, "Wiring Viterbi Decoders (Splitting DeBruijn Graphs)," *JPL TDA Progress Report* 42-96 (1988), pp. 93–103.

[Cyph90] R. Cypher, "Theoretical Aspects of VLSI Pin Limitations," *Advanced Research in VLSI: Proc. Sixth MIT Conference* (1990), pp. 314–327.

[Doli92a] S. Dolinar, T.-M. Ko, and R. McEliece "Some VLSI Decompositions of the DeBruijn Graph," *Symp. Discrete Math. Appl.*, Netherlands, Sept, 1992.

[Doli92b] S. Dolinar, T.-M. Ko, and R. McEliece "VLSI Decompositions for DeBruijn Graphs," *IEEE Int. Symp. Circuits and Syst.*, San Diego, May, 1992.

[Golo82] S. Golomb, *Shift Register Sequences (Rev. Ed.).* Laguna Hills, Calif.: Aegean Park Press, 1982.

[Harp64] L.H. Harper, "Optimal Assignments of Numbers to Vertices," *SIAM J. Appl. Math.* 12 (1964), pp. 131–135.

[Hart76] S. Hart, "A Note on the Edges of the N-Cube," *Discrete Math.*, 14 (1976), pp. 157–163.

[Leig92] F. T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* San Mateo, CA: Morgan Kaufmann Publ., 1992.

[Leig83] F. T. Leighton, *Complexity Issues in VLSI: Optimal Layouts for the Shuffle-Exchange Graph and Other Networks.* Cambridge, Mass.: MIT Press, 1983.

[Lind64] J.H. Lindsey, "Assignment of Numbers to Vertices," *Am. Math. Monthly* 71 (1964), pp. 508–5164.

[Mykk72] J. Mykkeltveit, "A proof of Golomb's conjecture for the deBruijn graph," *J. Comb. Theory (B)* 13 (1972), pp. 40–45.

[Schw91] E. Schwabe, *Efficient Embeddings and Simulations for Hypercubic Networks.* MIT Thesis, MIT/LCS/TR-508, 1991.

[Snir81] M. Snir, "I/O Limitations on Multi-Chip VLSI Systems," *Proc. 19th Allerton Conf. Comm., Control and Comput.* (1981), pp. 224–233.