An Abstract of the Thesis of

Robert A. Rowley for the degree of Doctor of Philosophy in Computer Science presented on December 2, 1993.

Title: Fault-Tolerant Ring Embedding in De Bruijn Networks

Abstract approved: _______ Redacted for Privacy __________

Bella Bose

A fault-tolerant embedding of a ring in an interconnection network involves finding a cycle in the network that avoids a set of faulty processors or links. This thesis deals primarily with fault-tolerant ring embedding in a d-ary De Bruijn network with d^n processors. It is shown that

(a) A fault-free cycle of length at least $d^n - f \cdot n$ can always be found in the event of $f \leq d-2$ processor failures.

(b) A fault-free Hamiltonian cycle can always be found if there are at most $d-2$ link failures and d is a prime power.

Both of these results are optimal when a worst-case fault distribution is assumed. The results on ring embedding in the presence of link failures can be extended, in some cases, to butterfly networks.

It is also shown that the d-ary De Bruijn digraph admits $d-1$ disjoint Hamiltonian cycles when d is a power of 2 and at least $(d-1)/2$ disjoint Hamiltonian cycles when d is an odd prime power.

Fault-Tolerant Ring Embedding in De Bruijn Networks

by

Robert A. Rowley

A Thesis

submitted to

Oregon State University

in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

Completed December 2, 1993

Commencement June 1994

Approved:

Redacted for Privacy

Professor of Computer Science in charge of major

Redacted for Privacy

Head of Department of Computer Science

Redacted for Privacy

Dean of Graduate School

Date Thesis is presented December 2, 1993

Typed by Robert Rowley for Robert Rowley .

Acknowledgements

I would like to thank my advisor Professor Bella Bose for his support and kindness over a period of several years. I am grateful to Professors Walter Rudd, Paul Cull, Burton Fein and David Sullivan for serving on my committee.

I would also like to acknowledge my officemates, past and present, whose friendship greatly enriched my sojourn at OSU.

Lastly, I thank my parents for their unlimited encouragement and support.

Table of Contents

List of Figures

Figure

List of Tables

Table

Fault-Tolerant Ring Embedding in De Bruijn Networks

Chapter 1

Introduction

Physical limitations on processing speeds of sequential computers have led high performance systems to rely increasingly on exploiting the inherent parallelism in problems and implementing their solutions on parallel machines. Recent advances in very large scale integrated (VLSI) technology make the design of multicomputers with thousands of processors feasible. Processors can be viewed as individual chips, several chips on a circuit board or even individual computers interconnected suitably to form networks. The study of interconnection networks and their underlying topologies have consequently become an integral part of designing high performance architectures.

As the number of processors in a system increases so to does the probability of a processor or communication link malfunctioning. It is therefore important to evaluate network performance in the presence of component failures. There are many ways to assess the reliability of a network. One widely-used measure is based on the network's connectivity. In this approach, a network is assumed to have tolerated a given set of component failures if it remains connected (i.e., all nonfaulty processors can still communicate with each other). However, even if the faulty network is connected it is not always clear how a parallel computation can be performed efficiently since the faulty network may bear little resemblance to the original network. A more general approach to fault-tolerance involves arranging the nonfaulty processors in some useful pattern such as a binary tree or a linear array.

In this thesis we investigate how the nonfaulty processors in a De Bruijn network can be joined in a ring. The goal of the work is to allow a faulty De Bruijn network to efficiently support algorithms that make use of a ring or linear array.

1.1 Problem Definition

We assume that an interconnection network is modeled by a graph, the nodes being the processors and the edges being the physical links between processors. Henceforth we will not make a distinction between a network and its underlying graph, e.g., node and processor will be used interchangeably. Component failures are assumed to be total, i.e., faulty nodes can neither perform computations nor route messages, and are modeled by removing the faulty nodes and/or edges from the graph.

Let R_k denote a cycle (or ring) of length k. An *embedding* of R_k into graph G is a one-to-one mapping τ that takes the nodes of R_k to the nodes of G and the edges of R_k to paths in G. The principal measures of an embedding are its dilation and congestion. The *dilation* of τ is the length of the longest path $\tau(e)$ taken over all edges e in R_k. The *congestion* of τ is the largest number of paths $\tau(e)$ using a single edge in G.

This thesis addresses the problem of embedding the largest possible ring R_k in a De Bruijn graph so that nodes in R_k are mapped to nonfaulty nodes and edges in R_k are mapped to fault-free paths. The proposed embeddings have unit dilation and congestion implying that the embedded ring is a subgraph of the faulty graph.

1.2 The De Bruijn Network

The De Bruijn interconnection network is modeled by either the directed or undirected De Bruijn graph. The d-ary directed De Bruijn graph B(d,n) has nodes corresponding to n-tuples over a d-letter alphabet Λ, and directed edges from each node $x_1 \ldots x_n$ to nodes $\{x_2 \ldots x_n\alpha \mid \alpha \in \Lambda\}$. Each node has indegree and outdegree d, and nodes of the form α^n have loops.

The undirected De Bruijn graph, denoted UB(d,n), is obtained from B(d,n) by deleting loops, removing the orientation of the edges and merging any resulting parallel edges. This results in a graph possessing d nodes of degree 2d–2, d·(d–1) nodes of degree 2d–1 and $d^n - d^2$ nodes of degree 2d [PR82]. The 8-node and 16-node De Bruijn graphs on the alphabet {0,1} are shown in Figure 1.1, and the undirected De Bruijn graph UB(2,3) is shown in Figure 1.2.

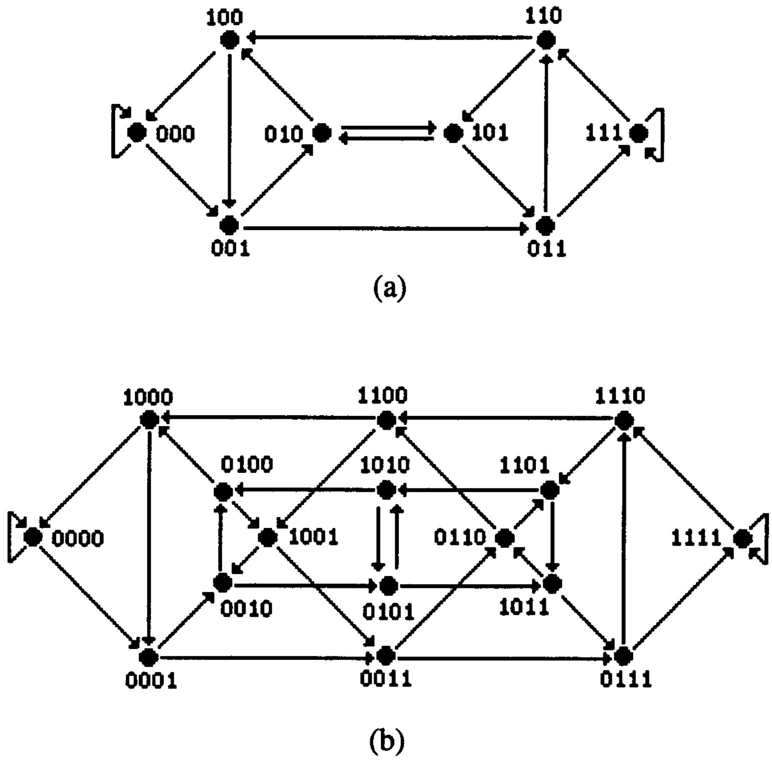

Figure 1.1. Binary De Bruijn graphs (a) B(2,3) and (b) B(2,4).

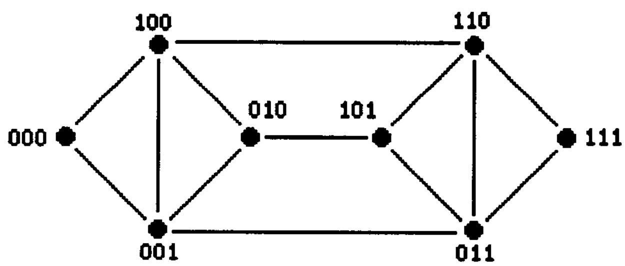

Figure 1.2. Undirected binary De Bruijn graph UB(2,3).

The De Bruijn graph[1] derives its name from N. G. de Bruijn who used it in 1946 to solve a combinatorial problem [Bru46]. The graph was first studied as a communication network by Schlumberger [Sch74], and was proposed as a processor interconnection network suitable for VLSI implementation by Pradhan in 1981 [Pra81]. The De Bruijn topology has also been proposed for implementing large optical-based networks that employ lightwave division multiplexing [SR91a, Muk92]. Surveys of De Bruijn graphs and networks can be found in papers by Bermond and Peyrat [BP89], Samatham and Pradhan [SP89], and in Section 3.3 of Leighton's book [Lei92].

The De Bruijn network has much to recommend it as a general purpose architecture. For example, any T-step computation on a mesh-of-trees, a k-dimensional mesh, or butterfly network can be simulated in O(T) steps by a like-sized De Bruijn graph [Sch92, KLM+89, Sch90]. In addition, B(d,n) contains such useful topologies as a complete d-ary tree and a shuffle-exchange as subgraphs.

While the De Bruijn network cannot simulate the popular hypercube network with constant slowdown on arbitrary computations, it can implement the class of highly parallel *normal* or *composite* hypercube algorithms with a small constant delay [Ull84, PV81, SP89]. The paradigm of these algorithms is the iterative version of the divide-and-conquer method, in which a problem is divided into subproblems of equal size, with a one-to-one correspondence between the results of the subproblems. A rich set of problems, such as permuting, sorting, the fast Fourier transform, and prefix computation have efficient solutions in this class.

The De Bruijn network can also efficiently simulate an idealized parallel computation model. Idealized models, such as a PRAM, assess a unit cost for communication between processors, making it possible to design parallel algorithms without regard to the network topology. Proposed simulations of a PRAM by a De Bruijn network involve probabilistic routing techniques developed in [VB81, LMR88].

---

[1] We follow Van Lint and Wilson in capitalizing the word "de" when omitting the initials of N.G. de Bruijn [LW92].

The applications of De Bruijn graphs are not limited to parallel processing. The graphs arise in such diverse areas as the search for dense graphs and minimal broadcast graphs [DF84, BHL+92, BP88], and the study of cellular automata [Wol84, Sut91].

De Bruijn graphs also play an important role in decoding convolutional codes. Maximum-likelihood decoding of these codes requires the decoder to find the best match between a received stream of symbols and a path in a De Bruijn graph with weighted edges. This has led to the implementation of an 8192-processor De Bruijn network in VLSI by the Jet Propulsion Laboratory. The network is to be used in conjunction with NASA's Galileo mission to Jupiter [DKM92].

1.3 Previous Work

The fact that the De Bruijn graph admits a Hamiltonian cycle is well-established. The properties of these cycles have received much attention, due in large part to their relationship to full length shift register sequences (also known as De Bruijn sequences). Background on this subject can be found in the surveys by Fredricksen [Fre82] and Ralston [Ral82].

The ability of the De Bruijn network to tolerate faults has been the subject of numerous papers. Work has been done on fault-tolerant routing [EH85, HP89, ISO85, ISO86, Pra81, Lyu93, DLH93], fault-diagnosis [PR82, SR91b], fault-tolerant VLSI design [Obr91], and emulation of a nonfaulty network by a faulty network [Ann89, BCH92]. The work most closely related to the topic of this thesis is that of Samatham and Pradhan who investigated fault-tolerant embeddings of complete d-ary trees [SP89].

1.4 Notation and Terminology

For the most part, our graph-theoretic terminology follows Harary [Har69]. The *indegree* and *outdegree* of a node X in a directed graph (digraph) G refer to the number of edges terminating at and originating from X. If there is an edge from X to Y we say that Y is the *successor* of X, and X is the *predecessor* of Y. A path is a sequence of vertices v_1, ..., v_k, such that (v_i, v_{i+1}) is an edge, $1 \le i \le k-1$. A *cycle* is a closed path in which all nodes are distinct, and a cycle of length k is called a *k-cycle*. A *circuit* is a closed path in which all edges are distinct. A cycle that visits all nodes in the graph is said to be *Hamiltonian*, and a circuit that traverses all edges is *Eulerian*. A digraph is (strongly) connected if there is an oriented path between every pair of nodes. A *component* of a digraph is a maximal connected subgraph.

The following notation terminology is used throughout this thesis.

Symbol	
B(d,n)	The d-ary De Bruijn digraph with d^n nodes.
UB(d,n)	The undirected De Bruijn graph.
Λ	The d-ary alphabet over which the nodes of B(d,n) are defined.
Λ^n	The set $\{x_1 \ldots x_n \mid x_i \in \Lambda\}$ of words of length n (n-tuples) over Λ.
Λ^*	The set of all words over Λ, including the empty string.
Z	The set of integers.
Z_d	The ring of integers modulo d.
GF(q)	The Galois field with q elements.
$\phi(n)$	The Euler function denoting the number of positive integers relatively prime to and not exceeding n.
$a \mid b$	a divides b.
LCM(a,b)	The least common multiple of integers a and b.
α^n	The n-tuple $\alpha \ldots \alpha$.
$\underline{\alpha\beta}$	The n-tuple $\alpha\beta \ldots \alpha\beta$ when n is even and the n-tuple $\alpha\beta \ldots \alpha\beta\alpha$ when n is odd (the value of n is implicit).
wt(X)	The sum of the digits in X.
$wt_\alpha(X)$	The number of α's in X.
$\pi^i(X)$	The left rotation of X by i positions.
N(X)	The cycle in B(d,n) obtained by rotating the letters of a node.
maximal cycle	A cycle of length q^n-1 in B(q,n) corresponding to a linear recurrence of period q^n-1.

Chapter 2

Ring Embedding With Node Failures

In this chapter, we describe a network-level distributed algorithm that finds a fault-free cycle in B(d,n) in the presence of an arbitrary number of node failures. We assume that the location of the faulty nodes is not known in advance.

Our approach involves partitioning the nodes of B(d,n) into small cycles called necklaces. A necklace is deemed faulty if it contains a faulty node. A large fault-free cycle is constructed by joining nonfaulty necklaces.

The time required to find the fault-free cycle is $O(\kappa + n)$, where κ is the diameter of the largest component in the graph obtained by removing the faulty necklaces from B(d,n). The length of the fault-free cycle is at least $d^n - n \cdot f$ when the number of faults f is at most d–2. In addition, a fault-free cycle of length at least $2^n - (n+1)$ can be found in the binary (d=2) De Bruijn graph when f = 1. In both cases, the number of communication steps required to find the cycle is $\Theta(n)$.

Our results indicate that the De Bruijn graph is competitive with the hypercube when the number of faults is small. It is known that a fault-free cycle of length $2^n - 2 \cdot f$ exists in the 2^n-node hypercube when $f \leq n-2$ [WC92, CL91a]. For example, a fault-free cycle of length 4092 can be found in the 4096-node hypercube when f = 2. By comparison, when there are two faults in the 4096-node De Bruijn graph B(4,6), a fault-free cycle of length at least 4084 can be found. It is worth mentioning that the hypercube has 50% more edges (24,576) than the De Bruijn graph (16,384) in this instance.

The remainder of this chapter is organized as follows. In Section 2.1 we introduce notation and review some relevant concepts from graph theory. The algorithm is described in Section 2.2, followed by a detailed example. A formal proof of correctness appears in Section 2.3. The algorithm's implementation and complexity are discussed in Sections 2.4 and 2.5.

2.1 Preliminaries

We assume that the nodes of B(d,n) correspond to n-tuples over Z_d. Throughout the chapter, W and V refer to elements of Z_d^{n-1}, α and β to d-ary digits, and x_i to the i'th digit of X.

A directed tree T rooted at R is a directed graph such that: (a) every node other than R is the terminal node of some edge in T; (b) R is the terminal node of no edge; (c) for each node X ≠ R there is a directed path from R to X. If (X,Y) is an edge in T, we say that X is the *parent* of Y and Y is the *child* of X. T is a *spanning tree* of G if T includes every node in G.

We use N(X) to denote the cycle $(x_1 \ldots x_n, x_2 \ldots x_n x_1, \ldots, x_n x_1 \ldots x_{n-1})$ in B(d,n) formed by rotating the digits of node X. Cycles of this form are called *necklaces*, and are represented by [Y] where Y is the minimal[1] node in the necklace. For example, N(1120) = [0112] = (1120, 1201, 2011, 0112). The set of necklaces partition the nodes of B(d,n) into disjoint cycles of length at most n. The *weight* of a node $X = x_1 \ldots x_n$ is $x_1 + x_2 + \ldots + x_n$, and is denoted wt(X). In addition, we use $wt_\alpha(X)$ to denote the number of α's in X. For example, when X = 1120, wt(X) = 4, $wt_0(X) = 1$, $wt_1(X) = 2$ and $wt_2(X) = 1$. Note that if N(X) = N(Y) then wt(X) = wt(Y) and $wt_\alpha(X) = wt_\alpha(Y)$ for any $\alpha \in \{0, \ldots, d-1\}$.

2.2 The Fault-Free Cycle Algorithm

In this section we present a high-level description of an algorithm to construct a cycle in B(d,n) that avoids a set $\{F_1, \ldots, F_f\}$ of faulty nodes. Say that necklace [X] is faulty if one or more nodes in [X] are faulty. Let B* denote the largest component in the graph B(d,n) – $\{N(F_1), \ldots, N(F_f)\}$ obtained by removing the faulty necklaces from B(d,n). The fault-free cycle will correspond to a Hamiltonian cycle in B*.

Our technique is suggested by a parallel algorithm proposed by Atallah and Vishkin to find an Euler tour in an arbitrary graph [AV84]. In that algorithm, an Euler tour was constructed by stitching together disjoint circuits. The process was guided by the

---

[1] N-tuples are ordered by viewing them as base-d numbers.

connections in a spanning tree of an auxiliary graph in which the disjoint circuits were viewed as nodes.

In our case, we construct a Hamiltonian cycle in B* by stitching together necklaces. We use a spanning subgraph of a necklace adjacency graph to determine whether, and how, two necklaces should be joined

Definition: A *necklace adjacency graph* N* is a directed graph with nodes corresponding to necklaces in B*. There is a directed edge labeled W from [X] to [Y] if and only if αW is in [X] and βW is in [Y], for some $\alpha, \beta \in Z_d$, $\alpha \neq \beta$. □

An edge labeled W (a W-edge) from [X] to [Y] can be viewed as exiting [X] at node αW and entering [Y] at node $W\beta$ (see Figure 2.1). Note that if there is a W-edge from [X] to [Y] then there is a corresponding W-edge from [Y] to [X]. (Antiparallel edges are depicted as a single edge with arrows at both ends.)

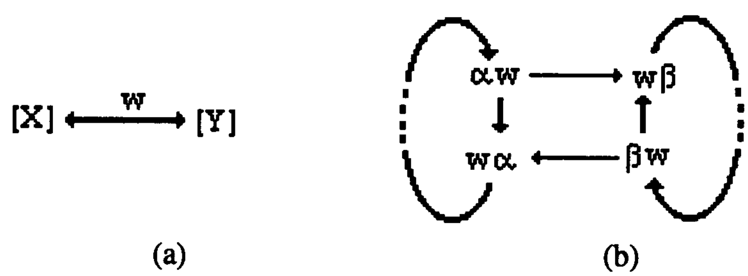

Figure 2.1. (a) Nodes in N* and (b) corresponding necklaces in B*.

Node αW (or $W\alpha$) in a given necklace is uniquely determined by W since αW and βW, $\alpha \neq \beta$, cannot be on the same necklace.

What follows is a high-level description of the proposed fault-free cycle (FFC) algorithm. The network-level implementation is described in Section 2.4.

FFC Algorithm

Step 1. Find a spanning tree T of N* such that, for every $W \in Z_d^{n-1}$, the W-edges in T, denoted T_W, have a common initial node, i.e., T_W is a subtree of height one.

Step 2. Modify T by changing every T_W from a parent and one or more children to a directed cycle with edges labeled W. Let D denote the modified tree. The modification of an edge-labeled tree is illustrated in Figure 2.2.

Figure 2.2. (a) Spanning tree T and (b) modified tree D.

Note that the modified tree is a spanning subgraph of N* because every pair of necklace-nodes [X] and [Y] in T_W are connected by a W-edge ([X], [Y]) in N*.

Step 3. Construct a Hamiltonian cycle H in B* by defining the successor of an arbitrary node αW in B* as follows. Assume that αW is in necklace [X]. If there is a W-edge in D leading from [X] to [Y] then the successor of αW is Wβ in [Y]; otherwise the successor of αW is Wα.

A formal proof of the correctness of the algorithm appears in the next section. We conclude this section with an example.

Example 2.1 Suppose that nodes 020 and 112 fail in the 27-node De Bruijn graph B(3,3). In this case B* = B(3,3) – {N(020), N(112)} since the graph remains connected after the removal of the faulty necklaces. B* contains 21 nodes so we should be able to construct a fault-free cycle of length 21.

The necklace adjacency graph N* is shown in Figure 2.3.

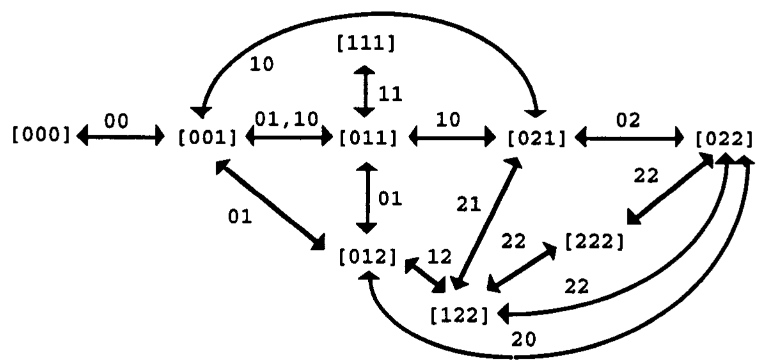

Figure 2.3. Adjacency graph N* for B(3,3) – {N(020), N(112)}.

A spanning tree of N* and the modified tree are shown in Figures 2.4 (a) and (b).

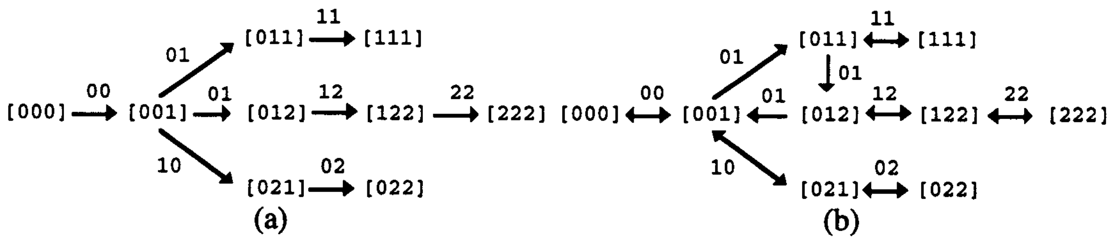

Figure 2.4. (a) Spanning tree and (b) modified tree.

The successor of each node in H can be readily obtained from 2.4 (b). For instance, node 120 is followed by its necklace successor 201 in H because [012] does not have an outgoing edge labeled 20. Conversely, node 101 is in [011] which has an outgoing 01-edge leading to [012], so 101 is followed in H by its successor in [012], namely 012.

The entire cycle is H = (000, 001, 011, 111, 110, 101, 012, 122, 222, 221, 212, 120, 201, 010, 102, 022, 220, 202, 021, 210, 100).

2.3 Proof of Correctness

In this section we prove that cycle H constructed in Step 3 of the FFC algorithm is indeed a Hamiltonian cycle in B*.

For each label W, a necklace-node in D either has no incident W-edges, or it has exactly one outgoing and one incoming W-edge. If a necklace-node [X] has an outgoing and an incoming edge W-edge in D, then nodes αW and Wα in [X] are said to be *outgoing* and *incoming* nodes respectively.

Lemma 2.1 *Every node in B* lies on exactly one necklace path from an incoming node to the next outgoing node.*

PROOF. Every necklace in B* has a corresponding necklace-node in D, so every necklace contains at least one outgoing node. If αW is an outgoing node then its necklace successor Wα is an incoming node, so outgoing and incoming nodes are alternately encountered as a necklace is traversed. (If Wα is also an outgoing node, it is counted first as an incoming node.) Consequently, each node in B* lies on exactly one path between an incoming node and the next outgoing node. □

Example 2.2 Suppose that [0122] = (0122, 1220, 2201, 2012) has incident edges labeled {012, 201, 220} in D. The incoming nodes are {<u>012</u>2, <u>201</u>2, <u>220</u>1} and the outgoing nodes are {$2\underline{012}$, $2\underline{201}$, $1\underline{220}$}. The alternating incoming and outgoing nodes in the traversal of [0122] are $\underline{012}2$, $1\underline{220}$, $\underline{220}1$, $2\underline{201}$, $\underline{201}2$, $2\underline{012}$, so the necklace is partitioned into three paths: (0122, 1220), (2201) and (2012). □

Any path in B* can be viewed as a path in D by considering only those edges from outgoing to incoming nodes, i.e. that lead from one necklace to another. Replacing each edge (αW, Wβ), $\alpha \neq \beta$, by the W-edge ([X], [Y]) where αW is in [X] and Wβ is in [Y] yields the desired path in D.

For instance, the path corresponding to H in Example 2.1 is ([000], [001], [011], [111], [011], [012], [122], [222], [122], [012], [001], [021], [022], [021], [001]). Note that this path forms an Eulerian circuit in D (see Figure 2.4.b). Lemma 2.2 shows that this is true in general.

Let H be the path in B* constructed in Step 3 of the FFC algorithm, and let J denote the corresponding path in D.

Lemma 2.2 *J is an Eulerian circuit in D.*

We will prove that (a) J is a circuit, (b) if one W-edge is in J then all W-edges are in J, (c) if a necklace-node is in J then all of the edges incident to it are in D are in J, and (d) all necklace-nodes in D are in J. Properties (a), (c) and (d) insure that every edge appears exactly once in J.

Proof of (a). Let {L_0, ..., L_{m-1}} be the labels of the edges incident to a necklace-node [X] in D, and assume that the outgoing and incoming nodes in [X] are $\alpha_0 L_0$, $L_0\alpha_0$, $\alpha_1 L_1$, $L_1\alpha_1$..., $L_{m-1}\alpha_{m-1}$ in the order in which they are encountered when traversing [X]. For instance, in Example 2.2 we could let $\alpha_0 L_0$ = 1220, $\alpha_1 L_1$ = 2201 and $\alpha_2 L_2$ = 2012. This ordering implies that the (L_i)-edge entering [X] is followed in J by the (L_{i+1})-edge exiting [X] (subscripts are reduced modulo m). Now, let the sequence of edges in J be E_0, E_1, E_2, ..., and assume that t is the smallest integer such that $E_i = E_t$, $0 \leq i \leq t-1$. From the above discussion, it follows that $E_{i-1} = E_{t-1}$ if $i > 0$. This contradicts the minimality of t, so we conclude that i = 0 and that J = E_0, E_1, ..., E_{t-1} is a circuit.

Proof of (b). Assume that T is rooted at [R]. Follow H beginning at some outgoing edge of [R]. If the first W-edge in J is ([X], [Y]), then the remaining W-edges must be traversed to get from [Y] back to [R] because this path must go through [X] (see, e.g., Figure 2.2.b).

Proof of (c). Assume that [X] is in J, and that the outgoing and incoming nodes in [X] are arranged as in the proof of (a). If the (L_i)-edge entering [X] is in J then so is the (L_{i+1})-edge exiting [X]. In addition, the (L_{i+1})-edge entering [X] is also in J, since all (L_{i+1})-edges are in J by Property (b). This implies that the (L_{i+2})-edge exiting [X] is in J, so the (L_{i+2})-edge entering [X] is in J, and so on.

Proof of (d). Suppose that [X] is not in J. Then there is some edge ([Y], [Z]) on the path from [R] to [X] such that [Y] is in J, and [Z] is not in J. Property (c) implies that all edges incident to [Y] are in J, so [Z] must also be in J, a contradiction. □

Proposition 2.1 *H is a Hamiltonian cycle in B*.*

PROOF. Every incoming node is in H since J includes every edge in D by Lemma 2.2. If an incoming node, say $W\alpha$, appears in H then H traverses the necklace containing $W\alpha$ until it encounters an outgoing node. Thus, by Lemma 2.1, every node in B* is in H. In addition, H is a cycle since a node does not reappear until all other necklace paths from incoming to outgoing nodes have been traversed. Consequently, H is Hamiltonian. □

2.4 Implementation

In this section we describe the network-level implementation of the FFC algorithm and analyze its complexity in terms of the number of communication steps needed to compute H. We assume that each node can communicate with all of its successors in one time step, i.e., multi-port communication. (If single-port communication is used, the time complexity increases by at most a factor of d.) The computation is complete when each node in B* has computed its successor in H.

We assume that the nonfaulty nodes in faulty necklaces do not participate in the computation. Each node can determine if its necklace is faulty by attempting to pass a message around the necklace. If a node does not receive its own message in n or fewer steps the necklace is assumed to be faulty. This process can be carried out simultaneously by each node, so only n steps are required to identify the nodes in B*.

Step 1.1 Find a spanning tree T′ of B*.

Select a distinguished node R in B* such that N(R) = [R]. Let T′ denote the spanning tree of B* corresponding to the propagation pattern of a message M broadcast from R. In the first step of the broadcast, R sends M to all of its successors. During the next step, the successors pass M along to their successors, and so on. The parent of a node X in T′ is the predecessor from which X first receives M. In the case of a tie, the minimal predecessor is chosen to be the parent. Note that the nodes $W\alpha$ and $W\beta$ have a common parent in T′ (assuming neither is R). This is because they have a common set of predecessors in B*, and hence receive M at the same time.

The number of steps required by the broadcast is equal to the maximum distance from the root R to any node in B* (the *eccentricity* of R). The eccentricity of R is bounded by the diameter of B* which we will denote by K.

Step 1.2 Use T′ to find a spanning tree T of N*.

A spanning tree T of N* rooted at [R] can be readily obtained from T′. For each necklace [X] choose a node Y in [X] such no other node in [X] received M prior to Y. Then let the parent of [X] in T be the necklace, say [Z], containing the parent (in T′) of Y. If $Y = W\alpha$ then the parent of Y in T′ is βW, for some $\beta \in Z_d$. The edge ([Z], [X]) is labeled W.

The tree T_W consisting of all W-edges in T has height one. To see this, let [X] and [Y] be the children in two edges labeled W. Then nodes $W\alpha$ and $W\beta$ received M first on [X] and [Y] respectively, for some $\alpha,\beta \in Z_d$. Since $W\alpha$ and $W\beta$ have a common parent in T′, [X] and [Y] also have a common parent in T.

The processing in this step is carried out at the necklace level, i.e., nodes communicate with nodes in the same necklace. The maximum length of a necklace is n, so O(n) steps are needed. At the end of this step (every node in) each necklace [X] knows its incident edges in T. We can also assume that for each W-edge {[X], [Y]}, [X] knows the identity of the node αW in [Y].

Step 2. Compute the edges in D.

Each necklace in T_W can determine the identity of the other necklaces in T_W in O(n) steps. Let $S = \{\alpha \mid \alpha W \text{ is in a necklace in } T_W\}$. The nodes in $\{\alpha W \mid \alpha \in S\}$ can inform all of the nodes in $\{W\alpha \mid \alpha \in S\}$ of the necklace that contains them in one step. The information can then be passed around the necklaces in at most n steps. The necklaces in T_W can be ordered according to their representative, i.e., [X] > [Y] if X > Y. Then, for each [X] in T_W, there is a W-edge from [X] to [Y] in D where [Y] is the next largest necklace in T_W. For example, the necklaces in T_{01} in Figure 2.4 (a) are ordered [001],

[011], [012]. The cycle is closed by inserting a W-edge between the largest and smallest necklaces.

Step 3. Compute, for each node X in B^*, the successor of X in H.

After Step 3 is completed every node has sufficient information to determine if it is an outgoing node and, if so, which of its successors in B^* is located in necklace [Y] where ([X], [Y]) is a W-edge in D.

The total number of steps required to compute H is therefore $O(K + n)$. The length of H is equal to the size of B^*.

2.5 Complexity Analysis

In this section we derive bounds on the length of the fault-free cycle and the amount of time required by the FFC algorithm to find it. It was shown in the previous section that these values correspond to the size and diameter of B^* respectively.

If $B(d,n) - \{N(F_1), \ldots, N(F_f)\}$ is connected then $B^* = B(d,n) - \{N(F_1), \ldots, N(F_f)\}$ and the size of B^* is $d^n - N_F$, where N_F denotes the total number of nodes in faulty necklaces. The maximum length of a necklace is n, so $N_F \leq n \cdot f$.

Let P be a path in B(d,n), and let S_P denote the necklaces encountered on path P excluding the initial and final nodes of P, i.e., $S_P = \{N(Z) \mid Z \text{ is an intermediate node in } P\}$. Two paths P and Q from node X to node Y are said to be *necklace-disjoint* if $S_P \cap S_Q = \emptyset$.

Proposition 2.2 *The FFC algorithm computes a fault-free cycle of length at least $d^n - n \cdot f$ in $\Theta(n)$ steps in the event of $f \leq d-2$ node failures.*

PROOF. For any two nodes X and Y we will prove the existence of a path of length $\Theta(n)$ from X to Y that avoids the faulty necklaces. First, consider the d paths $\{P_\alpha \mid \alpha \in Z_d\}$ where P_α is

$$X = x_1 \ldots x_n \to x_2 \ldots x_n\alpha \to x_3 \ldots x_n\alpha\alpha \to \ldots \to x_n\alpha \ldots \alpha \to \alpha \ldots \alpha.$$

Suppose that $N(A) = N(B)$ for nodes A and B in P_α and P_β respectively, $\alpha \neq \beta$. Let $A = x_s \ldots x_n\alpha \ldots \alpha$ and $B = x_t \ldots x_n\beta \ldots \beta$ for some $2 \leq s,t \leq n$. Without loss of generality assume that $s \leq t$. Then

$$wt_\alpha(A) = wt_\alpha(x_s \ldots x_{t-1}) + wt_\alpha(x_t \ldots x_{n-1}) + s-1, \text{ and}$$

$$wt_\alpha(B) = wt_\alpha(x_t \ldots x_{n-1})$$

must be equal since A and B are in the same necklace. However, this implies that $wt_\alpha(x_t \ldots x_{n-1}) + s = 0$; a contradiction because $s \geq 2$. Thus, the paths are necklace-disjoint, guaranteeing that at least one path, say P_α, is fault-free in the event of d–2 faults.

Now consider the d–1 paths $\{Q_i \mid 1 \leq i \leq d-1\}$ from $\alpha\ldots\alpha$ to Y where Q_i is the path

$$\alpha\ldots\alpha \rightarrow \alpha\ldots\alpha(\alpha{+}i) \rightarrow \alpha\ldots\alpha(\alpha{+}i)y_1 \rightarrow \ldots \rightarrow (\alpha{+}i)y_1\ldots y_{n-1} \rightarrow y_1\ldots y_n = Y$$

and $(\alpha+i)$ is reduced modulo d. Suppose that $N(U) = N(V)$ for nodes U and V in Q_i and Q_j respectively, $i \neq j$. Let $U = \alpha\ldots\alpha(\alpha+i)y_1\ldots y_s$ and $V = \alpha\ldots\alpha(\alpha+j)y_1\ldots y_t$, for some $0 \leq s,t \leq n-1$. Without loss of generality assume that $s \leq t$. Then

$$wt_\alpha(U) = (n - s - 1) + wt_\alpha(y_1\ldots y_s) \text{ and}$$

$$wt_\alpha(V) = (n - t - 1) + wt_\alpha(y_1\ldots y_s) + wt_\alpha(y_{s+1}\ldots y_t),$$

must be equal. So, $wt_\alpha(y_{s+1}\ldots y_t) = t-s$, i.e., $y_{s+1}\ldots y_t = \alpha\ldots\alpha$. In addition,

$$wt(U) = \alpha\cdot(n - s - 1) + wt(y_1\ldots y_s) + (\alpha+i) \text{ and}$$

$$wt(V) = \alpha\cdot(n - t - 1) + wt(y_1\ldots y_s) + \alpha\cdot(t - s) + (\alpha+j)$$

must also be equal. This contradicts the assumption that $i \neq j$, so Q_i and Q_j are necklace-disjoint.

A fault-free path can therefore be found between X and Y by combining a fault-free path P_α of length n from X to $\alpha\ldots\alpha$, with a fault-free path Q_i of length n+1 from $\alpha\ldots\alpha$ to Y. It is actually not necessary to route through $\alpha\ldots\alpha$ because there is an edge from $x_n\alpha\ldots\alpha$ to $\alpha\ldots\alpha(\alpha+i)$ in B(d,n). Hence, the total length of the path from X to Y is at most 2n. Thus, when $f < d-1$, the size of B* is at least $d^n - n\cdot f$ and the diameter of B* is at most 2n. □

The length of the cycle found by the FFC algorithm is optimal in a worst case analysis. In other words, for some distributions of $f \leq d-2$ faults, no fault-free cycle longer than $d^n - n\cdot f$ exists. The line graph property of De Bruijn graphs can be used to verify this assertion.

A *line graph* L(G) of a directed graph G is a graph whose nodes correspond to edges in G. There is an edge from node (a,b) to node (c,d) in L(G) if and only if (a,b) and (c,d) are adjacent edges in G (i.e., b = c). If the edge from $x_1 \ldots x_{n-1}$ to $x_2 \ldots x_n$ in B(d,n−1) is labeled $x_1 \ldots x_n$, it is easily be seen that B(d,n) is the line graph of B(d,n−1). This gives rise to a natural correspondence between circuits in B(d,n−1) and cycles in B(d,n). For example, the cycle (012, 122, 221, 212, 120, 201) in B(3,3) corresponds to the circuit (01, 12, 22, 21, 12, 20, 01) in B(3,2).

Assume that the nodes $F = \{\alpha^{n-1}(d-1) \mid 0 \leq \alpha \leq f-1\}$, $f \leq d-2$, fail in B(d,n). Let C be any cycle in B(d,n) that does not include the faulty nodes and let C′ be the corresponding circuit in B(d,n−1). Consider the graph H = B(d,n−1) − C′. A necessary and sufficient condition for a digraph to be Eulerian is that it be connected and balanced (i.e., the indegree of each node is equal to its outdegree). The components of H are therefore Eulerian because a balanced graph remains balanced after the removal of a circuit. Hence, the edges in H can be partitioned into circuits that correspond to Eulerian circuits in each component of H. This implies that the nodes in B(d,n) − C can be partitioned into cycles.

Suppose that the nodes in B(d,n) − C are partitioned into m cycles $D = \{D_1, \ldots, D_m\}$. If cycle D_i contains k_i faulty nodes, then the length of D_i must be at least $k_i \cdot n$. This follows from the observations that the distance between every pair of faulty nodes is n, and that the length of the smallest cycle containing a faulty node is also n. Every faulty node is contained in some cycle since C avoids the faulty nodes, so the combined lengths of all of the cycles in D is at least $n \cdot f$. Consequently, the number of nodes in B(d,n) − C is at least $n \cdot f$, and the length of C is at most $d^n - n \cdot f$.

In the best case scenario it is possible to find a cycle that includes all of the nonfaulty nodes. This follows from the fact that the De Bruijn graph is pancyclic, i.e., it contains a cycle of length t for any $1 \leq t \leq d^n$ [Lem71]. Thus, if the faults are favorably distributed, a fault-free cycle of length $d^n - f$ exists for $0 \leq f \leq d^n - 1$.

2.5.1 Binary De Bruijn Graphs

The binary De Bruijn graph may be disconnected in the event of a single faulty necklace. However, at most one node can be isolated in this case.

Proposition 2.3 *The FFC algorithm computes a fault-free cycle of length at least $2^n - (n+1)$ in $\Theta(n)$ steps in the binary De Bruijn graph B(2,n) in the event of a single node failure.*

PROOF. We will show that B(2,n) has a component with at least $2^n-(n+1)$ nodes and diameter $\Theta(n)$ when at most one necklace is removed.

Let $x = x_1 \ldots x_n$ be a faulty node in B(d,n), and let wt(x) = k. Let $y = y_1 \ldots y_n$ and $z = z_1 \ldots z_n$ be nodes in B(d,n) – N(x), and suppose that both wt(y) and wt(z) are greater than k. Then it is always possible to find a path of length at most 2n from y to z of the form

$$y \rightarrow y_2 \ldots y_{n-1}1 \rightarrow y_3 \ldots y_{n-1}11 \rightarrow \ldots \rightarrow 1 \ldots 1 \rightarrow 1 \ldots 1z_1 \rightarrow 1 \ldots 1z_1z_2 \rightarrow \ldots \rightarrow z$$

in which every node has weight greater than k. Conversely, if both wt(y) and wt(z) are less than k then there is a path of length at most n from y to z in which every node has weight less than k. It follows that if wt(x) = 0 (or n) then the 2^n-1 nodes of weight > 0 (or < n) form a connected subgraph of diameter $\Theta(n)$. Similarly, if wt(x) = 1 (or n–1) the $2^n - n - 1$ nodes of weight >1 (or < n–1) form a connected subgraph of diameter $\Theta(n)$.

Now suppose that $2 \le k \le n-2$, and let $W_1 = \{y \mid wt(y) < k\}$ and $W_2 = \{y \mid wt(y) > k\}$. We will show that B(d,n) – N(x) is connected. Let u be an arbitrary node of weight k such that $N(u) \neq N(x)$. Node u contains at least one 1 and one 0, so there are edges connecting nodes in N(u) and nodes of weight $k\pm1$. Thus, each $N(u) \neq N(x)$ forms a bridge between W_1 and W_2. We can always find at least one $N(u) \neq N(x)$ because $N(0\ldots01^k) \neq N(0\ldots0101^{k-1})$ for $n \ge k+2$. It follows that, in this case, B(d,n) – N(x) is connected, contains at least 2^n-n nodes, and has diameter $\Theta(n)$. □

2.5.2 Simulation Results

For an arbitrary number of faults it is difficult to give precise bounds on the size and diameter of the faulty graph. However, based on the results of simulations, it seems that a fault-free cycle can often be found efficiently even when the number of faults greatly exceeds the bounds of Propositions 2.2 and 2.3. The simulations were carried out by selecting a fixed source node R and, for each simulation, generating a set of f randomly distributed faults. The necklaces containing the faulty nodes were then removed from the graph, and the size of the component containing R and the eccentricity of R within the component was calculated. (If R was in a faulty necklace, a neighboring node was used instead.) These values correspond to the length of the fault-free cycle and the number of steps required to form the spanning tree in Step 1.1 respectively.

The results of the simulations for 1024-node De Bruijn graphs B(2,10) and B(4,5) are summarized in Tables 2.1 and 2.2. As can be seen from the tables, the fault-free cycle has length approximately $d^n - f \cdot n$ even when f is quite large. In fact, the average length of the cycle begins to noticeably exceed $d^n - n \cdot f$ as f increases. This behavior can be attributed the fact that the probability of a faulty necklace containing multiple faulty nodes increases with the number of faults. The simulation results also suggest that B(d,n) does not become severely fragmented even when a large number of necklaces are removed.

f	Avg. Size	Max. Size	Min. Size	$d^n-n \cdot f$	Avg. Ecc.	Max. Ecc.	Min. Ecc.
0	1024.00	1024	1024	1024	10.00	10	10
1	1014.13	1019	1013	1014	10.30	12	10
2	1004.48	1014	1003	1004	10.76	14	10
3	994.66	1004	993	994	11.10	13	10
4	985.03	994	982	984	11.40	14	10
5	975.79	994	972	974	11.65	14	10
6	966.35	984	963	964	11.95	14	11
7	956.61	974	952	954	12.28	15	11
8	948.41	978	942	944	12.45	15	11
9	938.02	969	933	934	12.68	17	11
10	928.97	949	922	924	12.81	16	11
20	843.14	873	822	824	14.59	19	12
30	762.55	833	723	724	16.50	21	14
40	686.16	744	11	624	18.48	26	10
50	622.75	679	565	524	20.28	26	16

Table 2.1. Size of the component containing node R = 0000000001 and the eccentricity of R in B(2,10) with f randomly distributed faulty necklaces.

f	Avg. Size	Max. Size	Min. Size	$d^n-n \cdot f$	Avg. Ecc.	Max. Ecc.	Min. Ecc.
0	1024.00	1024	1024	1024	5.00	5	5
1	1019.00	1019	1019	1019	5.72	6	5
2	1014.07	1019	1014	1014	5.96	6	5
3	1009.24	1014	1009	1009	5.99	6	5
4	1004.35	1009	1003	1004	5.99	6	5
5	999.33	1004	999	999	6.00	6	6
6	994.47	1002	994	994	6.01	7	6
7	989.66	994	989	989	6.01	7	6
8	984.80	994	984	984	6.01	7	6
9	979.79	989	979	979	6.03	7	6
10	975.07	984	974	974	6.08	7	6
20	928.14	949	924	924	6.41	7	6
30	882.88	902	874	874	6.82	8	6
40	840.39	864	824	824	7.15	8	6
50	798.07	828	779	774	7.38	9	7

Table 2.2. Size of the component containing node R = 00001 and the eccentricity of R in B(4,5) with f randomly distributed faulty necklaces.

2.6 Remarks

Several authors have proposed sequential algorithms that construct Hamiltonian cycles in a De Bruijn graph by linking together disjoint cycles [Hua90, Etz86, FM78, EL84]. The goal of these efforts is to efficiently generate a subset of De Bruijn sequences. Many, but not all, of these approaches exploit the cyclic partition of the De Bruijn graph yielded by necklaces. The results in this chapter differ from previous work in two important respects: we assume the presence of faulty nodes, and our reconfiguration algorithm operates in a distributed manner at the network level.

The idea of constructing a Hamiltonian cycle in a digraph by joining smaller cycles is discussed in a more general context in a paper by Cull which describes a class of digraphs for which this approach is applicable [Cul80]. This suggests that the reconfiguration algorithm given in this chapter can be adapted to work with other interconnection topologies based on digraphs in this class [BP89].

Chapter 3

Ring Embedding With Edge Failures

One way to cope with edge failures is to assume that nodes with faulty incident edges are themselves faulty, and then use the method described in Chapter 2 for finding a fault-free cycle in the presence of node failures. The drawback to this approach is that an unnecessarily large number of nonfaulty nodes may be excluded from the cycle. One way to overcome this problem as follows. Suppose that B(d,n) contains t edge-disjoint k-cycles. An edge failure can affect at most one k-cycle, so a fault-free k-cycle is guaranteed to exist in the presence of up to t–1 edge failures. Most of the effort in this chapter is devoted to establishing the existence of disjoint Hamiltonian cycles (HCs) in B(d,n).

This effort can be viewed as an extension of the original problem investigated by N. G. De Bruijn [Bru46]. De Bruijn was interested in circular d-ary sequences of length d^n in which every subsequence of length n is distinct. These sequences, which are sometimes known as De Bruijn sequences, correspond to Hamiltonian cycles in B(d,n). Finding a set of disjoint Hamiltonian cycles in B(d,n) amounts to finding a set of De Bruijn sequences in which every subsequence of length n+1 is distinct.

Disjoint HCs can also be beneficial in a fault-free environment. Their presence allows computations that use ring-structured communications to spread the message traffic more evenly across communication links.

Consider, for example, the problem of all-to-all broadcasting in which each node sends an identical message to all other nodes in the network. A simple all-to-all broadcasting algorithm using a ring (Hamiltonian cycle) requires every node to receive a new message from its ring predecessor and pass the previous message to its ring successor at each step. After N–1 steps, each node in an N-node network will have received messages from all other nodes. If the communication time depends on the length of the message, then the algorithm can be improved if the network contains t disjoint HCs. In this case, each message can be divided into t parts, and each submessage transmitted along a different HC. A related all-to-all broadcasting algorithm using disjoint HCs and wormhole routing is described in [LS90].

The remainder of this chapter is organized as follows. In Section 3.1 we review the relationship between cycles of length k in B(d,n) and d-ary sequences of period k. In Section 3.2 we present a method of constructing large edge-disjoint cycles in B(d,n) and apply these results to fault-tolerant embedding in Section 3.3. The results obtained in Section 3.3 are extended to d-ary butterfly graphs in Section 3.4.

3.1 Cycles and Sequences

Throughout this section we will assume that d is a prime power p^e and that Λ = GF(d), where GF(d) is the Galois field of order d.

We use the circular sequence $C = [c_0, c_1, \ldots, c_{k-1}]$ to denote the closed path of length k in B(d,n) in which node $c_ic_{i+1}\ldots c_{i+n-1}$ is followed by node $c_{i+1}c_{i+2}\ldots c_{i+n}$ (subscripts are reduced modulo k). For instance, [0,1,2,1,2] denotes the 5-cycle (012, 121, 212, 120, 201) in B(3,3). In this representation, n–tuples correspond to nodes in B(d,n) and (n+1)-tuples to edges. As a consequence, C is a cycle if and only if the n-tuples $\{c_ic_{i+1}\ldots c_{i+n-1} \mid 0 \le i \le k-1\}$ are distinct. In addition, C and $D = [d_0, d_1, \ldots, d_{m-1}]$ are edge-disjoint if and only if the sets $\{c_ic_{i+1}\ldots c_{i+n} \mid 0 \le i \le k-1\}$ and $\{d_id_{i+1}\ldots d_{i+n} \mid 0 \le i \le m-1\}$ are disjoint. If C and D are both HCs in B(d,n) we will simply say that they are "disjoint" rather than "edge-disjoint".

Let C be the sequence defined by the recurrence

$$c_{n+i} = a_{n-1}c_{n-1+i} + \ldots + a_0c_i, \quad i \ge 0, \tag{3.1}$$

for $a_i \in$ GF(d), $a_0 \ne 0$, and nonzero initial conditions. The *period* of C is the least k > 0 such that $c_i = c_{i+k}$ for $i \ge 0$. A sequence of period k corresponds to the k-cycle $[c_0, c_1, \ldots, c_{k-1}]$ in B(d,n).

The *characteristic polynomial* of C is

$$p(x) = x^n - a_{n-1}x^{n-1} - \ldots - a_0 \tag{3.2}$$

If p(x) is irreducible, then the period of C is equal to the order of p(x), where the *order* of p(x) is the least positive integer k such that p(x) divides $1 - x^k$. If p(x) is irreducible and has order d^n-1 then it is said to be *primitive* over GF(d). A sequence with a primitive characteristic polynomial corresponds to a cycle of length d^n-1 in B(d,n) that includes every node except 0^n.

Definition: A cycle $C = [c_0, c_1, \ldots, c_{d^{n-1}-1}]$ in B(d,n) is said to be a *maximal cycle* if it satisfies a recurrence of the form (3.1) and its characteristic polynomial (3.2) is primitive over GF(d).

Primitive polynomials of degree $n \geq 1$ are known to exist for every finite field [LP84].

Example 3.1 Suppose that we want to construct a cycle of length 5^2-1 in B(5,2). We start with a primitive polynomial of degree 2 over GF(5), say $p(x) = x^2 - x - 3$. The sequence with characteristic polynomial p(x) is $s_{2+i} = s_{1+i} + 3 \cdot s_i$. For initial conditions $s_0 = 0$ and $s_1 = 1$, the corresponding maximal cycle in B(5,2) is [0, 1, 1, 4, 2, 4, 0, 2, 2, 3, 4, 3, 0, 4, 4, 1, 3, 1,0, 3, 3, 2, 1, 2]. □

3.2 Disjoint Hamiltonian Cycles

An upper bound on the number of disjoint HCs is d–1 since some nodes in B(d,n) have indegree and outdegree d–1 (excluding loop edges). In this section we give a constructive proof that d–1 disjoint HCs exist when d is a power of two. We also show that at least $2^{-k} \Pi (p_i^{e_i} - 1)$, $1 \leq i \leq k$, disjoint HCs exist in general when $p_1^{e_1} p_2^{e_2} \ldots p_k^{e_k}$ is the prime factorization of d.

We approach the problem by first considering the case when d is a prime power and then generalize the construction to handle arbitrary values of d. We also propose a modification of the De Bruijn graph that allows it to admit d disjoint HCs. Related work on disjoint HCs in De Bruijn graphs is discussed in 3.2.4.

3.2.1 Disjoint HCs When d is a Prime Power

In this subsection we present a method for constructing at least (d–1)/2 disjoint Hamiltonian cycles in B(d,n) when $d = p^e$ is a prime power. Our approach involves first constructing d edge-disjoint cycles of length d^n-1, and then modifying these cycles to make them Hamiltonian.

Let

$$s + C \overset{\text{def}}{=} [s{+}c_0, s{+}c_1, \ldots, s{+}c_{k-1}]$$

for $s \in GF(d)$.

Lemma 3.1 *If C is a cycle in B(d,n) then s + C is also a cycle in B(d,n).*

PROOF. Assume that C is a cycle and that s + C is not a cycle. Then s + C must contain a repeated n-tuple. Suppose that $(s{+}c_i)(s{+}c_{i+1}) \ldots (s{+}c_{i+n-1}) = (s{+}c_j)(s{+}c_{j+1}) \ldots (s{+}c_{j+n-1})$. This implies that $c_i c_{i+1} \ldots c_{i+n-1} = c_j c_{j+1} \ldots c_{j+n-1}$, contradicting the assumption that C is a cycle. □

Let C be a maximal cycle in B(d,n) satisfying recurrence (3.1).

Lemma 3.2 *For any $s \in GF(d)$, $s + C = [d_0, \ldots, d_{m-1}]$ satisfies the recurrence*

$$d_{n+i} = a_{n-1}d_{n-1+i} + \ldots + a_0 d_i + s\cdot(1-\omega), \quad i \geq 0,$$

where $\omega = a_0 + \ldots + a_{n-1}$.

PROOF. Observe that $d_{n+i} = s + c_{n+i} = s + a_{n-1}c_{n-1+i} + \ldots + a_0 c_i$. Substituting $d_j - s$ for c_j yields

$$\begin{aligned} d_{n+i} &= a_{n-1}(d_{n-1+i} - s) + \ldots + a_0(c_i - s) + s \\ &= a_{n-1}d_{n-1+i} + \ldots + a_0 d_i - s(a_0 + \ldots + a_{n-1}) + s \\ &= a_{n-1}d_{n-1+i} + \ldots + a_0 d_i + s(1-\omega) \end{aligned}$$ □

Lemma 3.3 *The cycles in $\{s + C \mid s \in GF(d)\}$ are pairwise edge-disjoint.*

PROOF. Suppose that the (n+1)-tuple $v_0 \ldots v_n$ appears in both y + C and z + C, $y \neq z$. Then

$$v_n = a_{n-1}v_{n-1+i} + \ldots + a_0 v_0 + y\cdot(1-\omega), \text{ and}$$

$$v_n = a_{n-1}v_{n-1+i} + \ldots + a_0 v_0 + z\cdot(1-\omega).$$

So, $y\cdot(1-\omega) = z\cdot(1-\omega)$. This implies that $(1-\omega) = 0$ since $y \neq z$. However, if this were the case then $p(1) = 0$ and $(x-1)$ would divide (3.2). But $p(x)$ is assumed to be primitive, and hence irreducible. □

Note that every node except s^n appears in s + C because every node except 0^n appears in C. Node s^n can be inserted into s + C by replacing any (n+1)-tuple of the form $\alpha s^{n-1}\hat{\alpha}$ by the (n+2)-tuple $\alpha s^n\hat{\alpha}$. In terms of B(d,n) this is equivalent to replacing edge $\alpha s^{n-1}\hat{\alpha}$ by two edges: αs^n and $s^n\hat{\alpha}$ (see Figure 3.1).

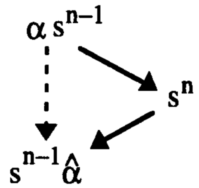

Figure 3.1 Edges used for inserting s^n into cycle s + C.

Node s^n can be inserted into cycle s + C in d−1 different ways since every node in the set $\{s^{n-1}\hat{\alpha} \mid \hat{\alpha} \neq s\}$ appears in s + C. Note that α is fixed once $\hat{\alpha}$ is selected because, by Lemma 3.2, α must satisfy

$$\hat{\alpha} = s\cdot(\omega - a_0) + a_0\alpha + s\cdot(1-\omega) \tag{3.3}$$

The cycles $\{x + C \mid x \in GF(d)\}$ partition the $d(d^n-1)$ non-loop edges of B(d,n), so the edges αs^n and $s^n\hat{\alpha}$ used to make s + C Hamiltonian must appear in cycles k + C and k′ + C respectively, for some k and k′ not equal to s. We next derive a relationship between k and k′. By Lemma 3.2,

$$s = s\cdot(\omega - a_0) + a_0\alpha + k\cdot(1-\omega) \tag{3.4}$$

and

$$\hat{\alpha} = s\omega + k'\cdot(1-\omega). \tag{3.5}$$

Subtracting (3.4) from (3.3) yields $\beta - s = s\cdot(1-\omega) - k\cdot(1-\omega)$, so

$$\hat{\alpha} = s\cdot(1-\omega) - k\cdot(1-\omega) + s. \tag{3.6}$$

It follows from (3.5) and (3.6) that $s\omega + k'\cdot(1-\omega) = s\cdot(1-\omega) - k\cdot(1-\omega) + s$, so $k' = 2s - k$. This implies that, for any $k \neq s$, cycle s + C can be made Hamiltonian by adding two edges such that one edge is in k + C and the other edge is in (2s − k) + C.

Let f: GF(d) → GF(d) be any function such that $f(x) \neq x$ for all x.

Definition: For a given f, define H_s to be the Hamiltonian cycle obtained by replacing $\alpha s^{n-1}\hat{\alpha}$ with $\alpha s^n\hat{\alpha}$ in s + C, where $\hat{\alpha} = s\omega + f(s)(1-\omega)$.

Lemma 3.4 *H_x and H_y have a common edge if and only if $y \in \{f(x), 2x - f(x)\}$ or $x \in \{f(y), 2y - f(y)\}$.*

PROOF. Assume that H_x and H_y have edge E in common. By Lemma 3, x + C and y + C are edge-disjoint, so E must be one of the edges used to extend x + C or y + C. Suppose that E was used to extend x + C, i.e., $E \in \{\alpha x^n, x^n\hat{\alpha}\}$. Since $\hat{\alpha} = x \cdot \omega + f(x)(1-\omega)$, we know from Lemma 2 that $x^n\hat{\alpha}$ is in f(x) + C. From the above discussion, this implies that αx^n is in (2x – f(x)) + C. The edges used to extend x + C and y + C are distinct when n > 1, so E must also be in y + C. Hence, $y \in \{f(x), 2x - f(x)\}$. Similarly, $x \in \{f(y), 2y - f(y)\}$ if E was used to extend y + C. Conversely, if $y \in \{f(x), 2x - f(x)\}$ or $x \in \{f(y), 2y - f(y)\}$ it is easily seen that H_x and H_y have a common edge. □

Our task now is to choose a function f so that the set $\{H_s \mid s \in GF(d)\}$ contains a large number of (pairwise) disjoint HCs. We will present three techniques for accomplishing this. The methods vary according to properties of p (recall that $d = p^e$). We begin with a lemma designed to show that, for every prime p, at least one of our strategies applies.

An element λ of Z_p is said to be a *primitive root* if every nonzero element of Z_p can be expressed as a power of λ.

Lemma 3.5 *Let p be an odd prime and let λ be any primitive root of Z_p. At least one of the following is true in Z_p:*

(a) $2 = \lambda^A$ and A is odd.

(b) $2 = \lambda^A + \lambda^B$ and both A and B are odd.

PROOF. Suppose that $2 = \lambda^A$ and that A is even. Let $T = \{1, 2, \ldots, p-1\} \setminus \{A\}$, and define $\sigma: T \to T$ such that $\lambda^i + \lambda^{\sigma(i)} = 2$. Since A is assumed to be even, T contains more odd than even elements. Function σ is one-to-one, so σ must map an odd to an odd. □

Condition (a) is equivalent to saying that 2 is a quadratic nonresidue of p. This situation occurs if and only if $p \equiv \pm 3 \pmod 8$ [Ros84, Theorem 9.4]. Consequently, Condition (b) holds if $p \equiv \pm 1 \pmod 8$. A prime can satisfy both conditions or only one. For instance, when p is 13 both (a) and (b) are satisfied since 7 is a primitive root of Z_{13}, and $2 \equiv 7^{11} \equiv 7 + 7^9 \pmod{13}$; conversely, in Z_5 only (a) is satisfied.

Strategy 1 Assume that p = 2, and let

$$f(x) = 0, \quad x \neq 0.$$

(The value of $f(0)$ is left undefined because H_0 will not be included in the set of disjoint HCs.) By Lemma 3.4, if H_x and H_y have a common edge then $x \in \{2y, 0\}$ or $y \in \{2x, 0\}$. However, $2 = 0$ in a field of characteristic 2, so H_x and H_y have a common edge only if x or y is zero. Hence, the d–1 HCs $\{H_s \mid s \in GF(d)\backslash\{0\}\}$ are disjoint.

Example 3.2 Let $GF(2^2) = \{0, 1, \zeta, \zeta^2\}$ where ζ is a root of $x^2 + x + 1$ over GF(2). In this event, $1 + \zeta = \zeta^2$, $1 + \zeta^2 = \zeta$, $\zeta + \zeta^2 = 1$ and $\zeta^3 = 1$. To construct a cycle C of length $4^2–1$ in B(4,2) we can use the recurrence

$$c_{2+i} = c_{1+i} + \zeta c_i, \ \ i \geq 0,$$

because $x^2 - x - \zeta$ is primitive over GF(4). With initial conditions $c_0 = 0$ and $c_1 = 1$ we get

$$C = [0,1,1,\zeta^2,1,0,\zeta,\zeta,1,\zeta,0,\zeta^2,\zeta^2,\zeta,\zeta^2].$$

In this case, $\omega = \zeta + 1 = \zeta^2$, $1 - \omega = \zeta$, and $a_0^{-1} = \zeta^{-1} = \zeta^2$. So, H_s is obtained by replacing $\alpha s\hat{\alpha}$ with $\alpha s s\hat{\alpha}$ in $s + C$ where $\hat{\alpha} = s\cdot\zeta^2$ and $\alpha = 0$. Specifically, $01\zeta^2$ is replaced by $011\zeta^2$ in $1 + C$, $0\zeta 1$ is replaced by $0\zeta\zeta 1$ in $\zeta + C$ and $0\zeta^2\zeta$ is replaced by $0\zeta^2\zeta^2\zeta$ in $\zeta^2 + C$. All of the replacement edges are in C, so

$$\begin{aligned} H_1 &= [1,0,0,\zeta,0,1,1,\zeta^2,\zeta^2,0,\zeta^2,1,\zeta,\zeta,\zeta^2,\zeta] \\ H_\zeta &= [\zeta,\zeta^2,\zeta^2,1,\zeta^2,\zeta,0,0,\zeta^2,0,\zeta,\zeta,1,1,0,1] \\ H_{\zeta^2} &= [\zeta^2,\zeta^2,\zeta,\zeta,0,\zeta,\zeta^2,1,1,\zeta,1,\zeta^2,0,0,1,0] \end{aligned}$$

are disjoint HCs in B(4,2). □

Strategy 2 Assume that $2 = \lambda^A + \lambda^B$ in Z_p for some primitive root λ and odd integers A and B. Let

$$f(x) = \begin{cases} \lambda^A\cdot x & \text{if } x \neq 0 \\ \lambda & \text{if } x = 0. \end{cases}$$

Assume that neither x nor y is zero. By Lemma 3.4, if H_x and H_y have a common edge then either $y \in \{\lambda^A\cdot x, (2-\lambda^A)\cdot x = \lambda^B\cdot x\}$ or $x \in \{\lambda^A\cdot y, \lambda^B\cdot y\}$. Hence, $y \in \{x\cdot\lambda^{\pm A}, x\cdot\lambda^{\pm B}\}$.

Let $J = \{1, \lambda, \lambda^2, \ldots, \lambda^{p-2}\}$. The nonzero elements of GF(d) can be partitioned into $r = (d-1)/(p-1)$ cosets $\{g_1\cdot J, \ldots, g_r\cdot J\}$ where $g\cdot J = \{g, g\cdot\lambda, g\cdot\lambda^2, \ldots, g\cdot\lambda^{p-2}\}$. Let $\Delta_i = \{H_x \mid x \in g_i\cdot J\}$ denote the HCs corresponding to the elements of $g_i\cdot J$, $1 \leq i \leq r$.

If $H_x \in \Delta_i$ has an edge in common with $H_y \in \Delta_j$ then $i = j$. This is due to the fact that $x = g_i \cdot \lambda^m$ for some m. Consequently, $y \in \{g_i \cdot \lambda^{m \pm A}, g_i \cdot \lambda^{m \pm B}\} \subseteq g_i \cdot J$. In addition, if m is even (resp. odd) then both $m \pm A$ and $m \pm B$ are odd (resp. even) since A and B are odd. Thus, H_x and H_y are disjoint if $x = g_i \cdot \lambda^j$, $y = g_i \cdot \lambda^k$, and j and k have the same parity. Therefore, the $(p-1)/2$ HCs

$$L_i = \{H_x \mid x = g_i \cdot \lambda^{2k}, k \in \{1, \ldots, (p-1)/2\}\}$$

are disjoint. The HCs in L_i are disjoint to the HCs in L_j, $i \neq j$, because $L_i \subseteq \Delta_i$ and $L_j \subseteq \Delta_j$, so we can obtain $r \cdot (p-1)/2 = (d-1)/2$ disjoint HCs by taking the union $L = L_1 \cup L_2 \cup \ldots \cup L_r$.

If $(p-1)/2$ is even, H_0 can be added to L. By Lemma 3.4, H_0 has edges in common only with H_λ and $H_{-\lambda}$ since $f(0) = \lambda$. Note that $\lambda^{(p-1)/2} = -1$ since $\lambda^{p-1} = 1$. Hence, $-\lambda = (-1)\lambda = \lambda^{(p-1)/2+1}$ is an odd power of λ when $(p-1)/2$ is even. If $g_1 = 1$, then L does not include any H_x where x is an odd power of λ, so H_0 does not conflict with any HC in L. The addition of H_0 brings the total number of disjoint HCs to $(d+1)/2$.

Example 3.3 Let $d = 13$. In Z_{13}, 7 is a primitive root and $2 = 7 + 7^9$. In addition 12/2 is even, so we can find 7 disjoint HCs by taking $f(x) = 7x$, $x \neq 0$, and $f(0) = 7$. In this case, H_x has an edge in common with H_y, $y \in \{7x, 7^9x, 7^{-1}x, 7^{-9}x\}$. This relationship is depicted graphically in Figure 3.2 where vertices x and y are connected if H_x and H_y have a common edge.

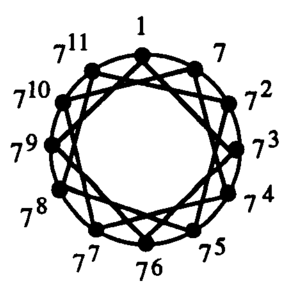

Figure 3.2. Graphical representation of non-disjoint HCs in B(13,n).

In addition, H_0 has an edge in common only with S_7 and $S_{-7} = S_{77}$. Thus, $\{H_0, H_1, H_{72}, H_{74}, H_{76}, H_{78}, H_{710}\}$ are disjoint. □

Strategy 3. Assume that $2 = \lambda^A$ in Z_p for some primitive root λ and odd integer A. Let

$$f(x) = \begin{cases} \lambda^{A} \cdot x & \text{if } x \neq 0 \\ \lambda & \text{if } x = 0. \end{cases}$$

This approach is almost identical to Strategy 2, except that H_x, $x = g_i \cdot \lambda^m$, has an edge in common with H_y, where $y \in \{g_i \cdot \lambda^{m \pm A}, 0\}$. As in Strategy 2, the (d–1)/2 HCs

$$L = \{H_x \mid x = g_i \cdot \lambda^{2k}\}, \quad 1 \leq k \leq (p-1)/2, \quad 1 \leq i \leq r$$

are disjoint. Note that Strategy 2 is superior to Strategy 3 when (p–1)/2 is even because H_0 cannot be added to L in the latter approach.

Example 3.4 Suppose that d = 5 and n = 2. Let C be the maximal cycle of length 24 in B(5,2) from Example 3.1. Since 3 is a primitive root of Z_5 and $2 = 3^3$, we can find 2 disjoint HCs in B(5,2) using f(x) = 3x, $x \neq 0$. H_s is obtained by replacing $\alpha s \hat{\alpha}$ by $\alpha s s \hat{\alpha}$ in s + C, where $\hat{\alpha} = s\omega + 2s(1-\omega)$ and α satisfies equation (3). In this example, $\omega = 4$ and $a_0^{-1} = 3^{-1} = 2$, so $\hat{\alpha} = 4s + 2s(2) = 3s$ and $\alpha = 0$. Since $J = \{3, 3^2 = 4, 3^3 = 2, 3^4 = 1\}$, the HCs

$$H_1 = [1,2,2,0,3,\underline{0,1,1,3},3,4,0,4,1,0,0,2,4,2,1,4,4,3,2,3],$$
$$H_4 = [4,0,0,3,1,3,4,1,1,2,3,2,4,3,3,0,2,\underline{0,4,4,2},2,1,0,1]$$

are disjoint. □

The number of disjoint HCs found by the strategies presented in this section is summarized in Proposition 3.1.

Proposition 3.1 *The number of disjoint HCs in B(p^e,n) is at least $\psi(p^e)$ where $\psi(p^e)$ is*

(i) $p^e - 1$ when p = 2,

(ii) $(p^e + 1)/2$ when (p–1)/2 is even and p satisfies condition (b) of Lemma 3.5,

(iii) $(p^e - 1)/2$ in all other cases.

3.2.2 Disjoint HCs in the General Case

In this section we use the disjoint HCs of the previous section to construct disjoint HCs in B(d,n) for any value of d. We now assume that the alphabet over which the De Bruijn graph is defined is Z_d. The cycles of the previous section can be readily mapped to this representation using any one-to-one mapping of the elements of GF(d) to Z_d.

Let $A = [a_0, \ldots, a_{s^n-1}]$ be an HC in $B(s,n)$ and let $B = [b_0, \ldots, b_{t^n-1}]$ be an HC in $B(t,n)$. Let $\langle A,B\rangle$ denote the cycle whose i'th element is $a_i \cdot t + b_i$, $0 \le i \le (s \cdot t)^n - 1$, where the subscripts of the a_i's and b_i's are reduced modulo s^n and t^n respectively. The following lemma is due to Rees ([Ree46], Lemma 3).

Lemma 3.6 *If s and t are relatively prime then $\langle A,B\rangle$ is a Hamiltonian cycle in $B(s \cdot t,n)$.*

Example 3.5 $A = [0,0,1,1]$ and $B = [0,0,2,2,1,2,0,1,1]$ are Hamiltonian cycles in $B(2,2)$ and $B(3,2)$ respectively, so

$\langle A,B\rangle =$ [0,0,5,5,1,2,3,4,1,0,3,5,2,1,5,3,1,1,3,3,2,2,4,5,0,1,4,3,0,2,5,4,2,0,4,4]

is a Hamiltonian cycle in $B(6,2)$. □

Let A and A' be HCs in $B(s,n)$ and let B and B' be HCs in $B(t,n)$, with s relatively prime to t.

Lemma 3.7 *$\langle A, B\rangle$ and $\langle A', B'\rangle$ are disjoint if A and A′ are disjoint or if B and B′ are disjoint.*

PROOF. Suppose that A and A′ or B and B′ are disjoint. If $\langle A, B\rangle$ and $\langle A', B'\rangle$ have a common edge then, for some i and j,

$$\begin{aligned} a_i \cdot t + b_i &= a_j' \cdot t + b_j' \\ a_{i+1} \cdot t + b_{i+1} &= a'_{j+1} \cdot t + b'_{j+1} \\ &\cdots \\ a_{i+n} \cdot t + b_{i+n} &= a'_{j+n} \cdot t + b'_{j+n}. \end{aligned}$$

This implies that the edge $a_i a_{i+1} \ldots a_{i+n} = a_j' a'_{j+1} \ldots a'_{j+n}$ appears in both A and A′, and the edge $b_i b_{i+1} \ldots b_{i+n} = b_j' b'_{j+1} \ldots b'_{j+n}$ appears in both B and B′, contradicting the assumption that at least one pair is disjoint. □

Let $d = p_1^{e_1} p_2^{e_2} \ldots p_k^{e_k}$ be the prime factorization of d, and let $\psi(d) = \psi(p_1^{e_1})\psi(p_2^{e_2})\ldots\psi(p_k^{e_k})$, where $\psi(\cdot)$ is defined for prime powers in Proposition 3.1.

Proposition 3.2 *There are at least $\psi(d)$ disjoint HCs in $B(d,n)$.*

PROOF. We use induction on k. When $k = 1$, $d = p^e$ and we can find $\psi(p^e)$ disjoint HCs by definition. Suppose that $k > 1$, and let $d' = p_1^{e_1} p_2^{e_2} \ldots p_{k-1}^{e_{k-1}}$. By induction there are $\psi(d')$ disjoint HCs in $B(d',n)$, say $\{A_i \mid 1 \le i \le \psi(d')\}$. There are also $\psi(p_k^{e_k})$ disjoint HCs in $B(p_k^{e_k},n)$, say $\{B_j \mid 1 \le j \le \psi(p_k^{e_k})\}$. Consider the set $\Gamma = \{\langle A_i,B_j\rangle \mid 1 \le i$

$\le \psi(d'), 1 \le j \le \psi(p_k^{e_k})\}$. The elements of Γ are HCs in B(d,n) by Lemma 3.6. In addition, if $\langle A_i,B_j\rangle$ and $\langle A_u,B_v\rangle$ are not equal then they are disjoint by Lemma 3.7 since either A_i and A_u, or B_j and B_v, are disjoint. We can therefore find $\psi(d')\cdot\psi(p_k^{e_k}) = \psi(d)$ disjoint HCs in B(d,n). □

Corollary 3.1 *The number of disjoint Hamiltonian cycles in B(d,n) is at least*

$$\frac{1}{2^k}\prod_{i=1}^{k}(p_i^{e_i} - 1).$$

PROOF. The proof follows directly from Propositions 3.1 and 3.2. □

Corollary 3.2 *The number of disjoint Hamiltonian cycles in B(d,n) is at least $\phi(d)/2^k$ where $\phi(\cdot)$ is the Euler function.*

PROOF. Corollary 3.2 follows from Corollary 3.1 and the fact that $\phi(d) = \phi(p_1^{e_1})\phi(p_2^{e_2})\ldots\phi(p_k^{e_k})$ and $\phi(p^e) = p^e - p^{e-1} \le p^e - 1$. □

We conclude this section by listing the values of $\psi(d)$ for d = 2, ..., 38 in Table 3.1.

d	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ψ(d)	1	1	3	2	1	3	7	4	2	5	3	7	3	2	15	9	4	9	6

d	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
ψ(d)	3	5	11	7	12	7	13	9	15	2	15	31	5	9	6	12	19	9

Table 3.1. Values of $\psi(d)$ for $2 \le d \le 38$.

3.2.3 Hamiltonian Decompositions

A digraph is said to admit a *Hamiltonian decomposition* if its edge set can be partitioned into disjoint HCs [ABS90]. This occurs if and only if every node has indegree and outdegree d, and there exist d disjoint HCs.

It is impossible to partition the edges of B(d,n) into HCs because of the presence of loop edges. Even in the best of circumstances only d–1 disjoint HCs exist, accounting for $d^n(d-1)$ edges. Consequently, at least d^n edges do not appear in any HC. In this section, we propose to modify B(d,n) to enable it to admit d disjoint HCs when n > 1 and $d = p^e$ for any odd prime p. We also propose modifying the binary De Bruijn graph B(2,n) to enable it to admit two disjoint HCs. As in B(d,n), the nodes of the modified graph

MB(d,n) will have indegree and outdegree d, so MB(d,n) will admit a Hamiltonian decomposition. The modified graph retains most of the nice graph-theoretic properties of B(d,n); in fact, the undirected version of MB(d,n) contains UB(d,n) as a subgraph.

THE CASE WHEN D IS AN ODD PRIME POWER.

Assume that $d = p^e$ for some odd prime p and let C be a maximal cycle in B(d,n). Recall that the cycles $\{s + C \mid s \in GF(d)\}$ are pairwise edge-disjoint and that s + C includes every node save s^n.

Let $\underline{\alpha\beta}$, $\alpha, \beta \in GF(d)$, denote the n-tuple $\alpha\beta\ldots\alpha\beta$ when n is even and the n-tuple $\alpha\beta\ldots\alpha\beta\alpha$ when n is odd. When $\alpha \neq \beta$, we say that $(\underline{\alpha\beta}, \underline{\beta\alpha})$ is a *parallel edge* (or *p-edge*) in B(d,n) since there is a corresponding edge from $\underline{\beta\alpha}$ to $\underline{\alpha\beta}$.

Let $E = (\underline{\alpha\beta}, \underline{\beta\alpha})$ be any p-edge in cycle C, and let E_s denote the p-edge $(\underline{(\alpha+s)(\beta+s)}, \underline{(\beta+s)(\alpha+s)})$ in s + C. Let H_s denote the Hamiltonian cycle obtained from s + C by replacing E_s by edges $(\underline{(\alpha+s)(\beta+s)}, s^n)$ and $(s^n, \underline{(\beta+s)(\alpha+s)})$. Note that these new edges may not be in B(d,n). Define MB(d,n) to be the directed graph obtained by taking the union of the edges in $\{H_s \mid s \in GF(d)\}$.

Clearly MB(d,n) admits a Hamiltonian decomposition because the new edges are distinct when $n > 1$. We claim additionally that (i) every node in MB(d,n) has indegree and outdegree d, and (ii) the undirected graph UM(B(d,n) obtained by removing the orientation of the edges in MB(d,n) contains UB(d,n).

The first claim follows from the fact that every node appears exactly once in each of the d disjoint HCs. To prove that UMB(d,n) contains UB(d,n), we will argue that every pair of nodes that are adjacent in UB(d,n) are also adjacent in UMB(d,n). The only edges replaced in the modification of B(d,n) are p-edges, so it is sufficient to prove that at most one of each pair of p-edges is replaced, e.g., if $(\underline{01}, \underline{10})$ is replaced then $(\underline{10}, \underline{01})$ must be left intact.

Suppose that both $(\underline{xy}, \underline{yx})$ and $(\underline{yx}, \underline{xy})$ are replaced in cycles H_s and H_t respectively for some s and t, $s \neq t$. This implies that $\underline{(\alpha+s)(\beta+s)} = \underline{(\beta+t)(\alpha+t)} = \underline{xy}$. So, $\alpha+s = \beta+t = x$ and $\beta+s = \alpha+t = y$. Then, $(\alpha+s) + (\beta+s) = (\beta+t) + (\alpha+t) \Rightarrow 2s = 2t$, a contradiction since GF(d) does not have characteristic 2. Thus, there is at least one undirected edge between $\underline{xy}$ and $\underline{yx}$ in UMB(d,n)[1].

THE CASE WHEN D = 2.

When d = 2 the modification is slightly different. Let C be a maximal cycle in B(2,n). Note that C contains the edge $(10^{n-1}, 0^{n-1}1)$ because it omits node 0^n. We first add 0^n to C by inserting it between 10^{n-1} and $0^{n-1}1$; then remove 0^n from 1+C. Without loss of generality, assume that 1+C contains the p-edge $(\underline{01}, \underline{10})$. If this edge is replaced with new edges $(\underline{01}, 0^n)$, $(0^n, 1^n)$, and $(1^n, \underline{10})$, then C and 1+C will form disjoint HCs. When loops are deleted, each node has indegree and outdegree 2. As before, UMB(2,n) contains UB(2,n) as a subgraph.

Example 3.6 Let d=2, and n=3. The maximal cycle C that satisfies the recurrence $c_{i+3} = c_{i+2} + c_i$, with initial conditions $c_0 = c_1 = 0$ and $c_2 = 1$, is [0, 0, 1, 1, 1, 0, 1]. C is extended by inserting 000 between 100 and 001. In 1+C = [1, 1, 0, 0, 0, 1, 0], node 000 is removed and the p-edge (010, 101) is replaced by new edges (010, 000), (000, 111) and (111, 101). The disjoint HCs in UMB(2,3) are shown in Figure 3.3. □

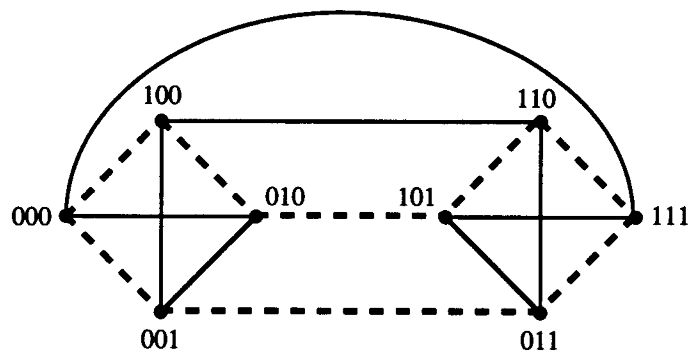

Figure 3.3. Hamiltonian decomposition of UMB(2,3).

3.2.4 Related Work

We have presented a constructive proof that most De Bruijn graphs contain multiple disjoint Hamiltonian cycles. Our results are provably optimal only when d is a power of 2, in which case we are able to find d–1 disjoint HCs in B(d,n).

---

[1]There may be two edges between $\underline{xy}$ and $\underline{yx}$, so UMB(d,n) is actually a multigraph.

Related work in this area exploits the fact that B(d,1) is the complete digraph on d nodes with loops at each node, denoted K_d^{**}. In [BBR93], Barth, Bond and Raspaud proved that there are at least $\phi(d)$ pairwise compatible Eulerian circuits in K_d^{**}. Two circuits are said to be compatible if they do not share any pair of consecutive edges. Compatible Eulerian circuits in K_d^{**} correspond to disjoint HCs in B(d,2) since B(d,2) is the line graph of B(d,1). Consequently, B(d,2) admits $\phi(d)$ disjoint HCs for any choice of d. This approach yields d–1 disjoint HCs when d is prime.

3.3 Ring Embedding

We now apply the results of the previous section to the problem of embedding a fault-free Hamiltonian cycle in B(d,n) in the presence of edge failures. At best, d–2 failures can be tolerated since it is possible to render B(d,n) non-Hamiltonian by removing a set of d–1 edges (e.g., the d–1 non-loop edges terminating at node 0…0).

Let

$$\varphi(d) \stackrel{\text{def}}{=} p_1^{e_1} + p_2^{e_2} + \ldots + p_k^{e_k} - 2k$$

where $p_1^{e_1} p_2^{e_2} \ldots p_k^{e_k}$ is the prime factorization of d.

Proposition 3.3 *There is a fault-free Hamiltonian cycle in B(d,n) when $f \le \varphi(d)$.*

PROOF. The proof is through induction on k. When k = 1, d is a prime power so there exists a maximal (d^n-1)-cycle C in B(d,n). The cycles $\{s + C \mid s \in GF(d)\}$ are pairwise edge-disjoint so at least one, say s + C, is fault-free when $f \le \varphi(d) = d-2$. As was noted in Section 3.2.1, s + C can be made Hamiltonian by replacing any edge of the form $\alpha s^{n-1}\hat{\alpha}$ by edges αs^n and $s^n\hat{\alpha}$, where α and $\hat{\alpha}$ satisfy Equation (3.3) and $\alpha \ne s$. Let $\{\alpha_i s^n,\ s^n\hat{\alpha}_i \mid 1 \le i \le d-1\}$ be the set of pairs of replacement edges. If n > 1 then $i \ne j \Rightarrow \alpha_i s^n \ne s^n\hat{\alpha}_j$, so a faulty edge affects at most one pair. Thus when $f \le \varphi(d)$, at least one fault-free pair of edges can be found to make s + C Hamiltonian.

Now assume that k > 1, and let $s = d/p_k^{e_k}$ and $t = p_k^{e_k}$. It was shown in Section 3.2.2 that an HC ⟨A,B⟩ in B(d,n) can be formed from HCs A and B in B(s,n) and B(t,n) respectively. Every edge in ⟨A,B⟩ corresponds to a unique pair of edges; one from A and one from B. More precisely, edge $v_0 \ldots v_n$ in ⟨A,B⟩ corresponds to $a_0 \ldots a_n$ in A and $b_0 \ldots b_n$ in B, where $v_i = a_i \cdot t + b_i$. Consequently, we can construct an HC in B(d,n) that avoids $v_0 \ldots v_n$ by finding HCs A and B in B(s,n) and B(t,n) respectively such that (i) A does not include $a_0 \ldots a_n$ or (ii) B does not include $b_0 \ldots b_n$. By induction, A can avoid any set of $\varphi(s)$ edges and B can avoid any set of $\varphi(t)$ edges. Hence, we can construct a Hamiltonian cycle in B(d,n) that avoids any set of $\varphi(s) + \varphi(t) = \varphi(d)$ edges. □

Proposition 3.3 implies that B(d,n) admits an HC in the presence of the maximum number of edge faults when d is a prime power because $\varphi(p^e) = p^e - 2$. Note also that $\varphi(d) \geq 1$ when $d > 2$, so every non-binary De Bruijn graph admits an HC in the presence of a single edge failure. For some values of d it is possible to tolerate a larger number of edge faults by directly applying the results of Proposition 3.2. The two approaches are combined in the following proposition.

Proposition 3.4 *B(d,n) admits a fault-free Hamiltonian cycle in the event of at most* $\mathrm{MAX}\{\psi(d)-1, \varphi(d)\}$ *edge failures.*

PROOF. B(d,n) admits $\psi(d)$ disjoint HCs by definition, so at least one HC is guaranteed to be fault-free in the event of $\psi(d) - 1$ edge faults. Proposition 3.3 insures that a fault-free HC exists in the event of $\varphi(d)$ edge faults. □

We conclude this section by listing the values of $\mathrm{MAX}\{\psi(d)-1, \varphi(d)\}$ for d = 2, ..., 35 in Table 3.2. For most of the tabulated values, $\mathrm{MAX}\{\psi(d)-1, \varphi(d)\} = \varphi(d)$. The sole exception occurs when d = 28.

d	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\mathrm{MAX}\{\psi(d)-1, \varphi(d)\}$	0	1	2	3	1	5	6	7	3	9	3	11	5	4	14	15	7

d	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
$\mathrm{MAX}\{\psi(d)-1, \varphi(d)\}$	17	5	6	9	21	7	23	11	25	8	27	4	29	30	10	15	8

Table 3.2. Values of $\mathrm{MAX}\{\psi(d)-1, \varphi(d)\}$ for $2 \leq d \leq 35$.

3.4 Extensions to the Butterfly Graph

In this section we exploit a structural relationship between De Bruijn graphs and butterfly graphs to obtain results on fault-tolerant ring embedding in butterflies.

The d-ary butterfly digraph F(d,n) has node set $Z_n \times Z_d^n$ and edges from each node (k, $x_0x_2 \ldots x_{n-1}$) to (k+1 (mod n), $x_0x_1 \ldots x_{k-1}\alpha x_{k+1} \ldots x_{n-1}$), for all $\alpha \in Z_d$. It is conventional to think of vertex (k, X) as being at level k and column X. The binary butterfly digraph F(2,3) is depicted in Figure 3.4 (the first row is replicated to aid in visualization).

In [ABR90] it was demonstrated that the node set of F(d,n) can be partitioned into d^n subsets in such a way that when the nodes in each subset are combined into a single node, and parallel edges are merged, the resulting graph is (isomorphic to) B(d,n).

More precisely, each node X in B(d,n) is associated with the set of butterfly nodes $S_X = \{(0,X), (1, \pi^{-1}(X)), (2, \pi^{-2}(X)), \ldots, (n-1, \pi^{-(n-1)}(X))\}$, where $\pi^i(X)$ denotes the left rotation of X by i positions (e.g., $\pi^3(1202) = \pi^{-1}(1202) = 2120$). Each De Bruijn edge is consequently associated with n butterfly edges. This is formally proven in Lemma 3.8, and illustrated in Figure 3.5.

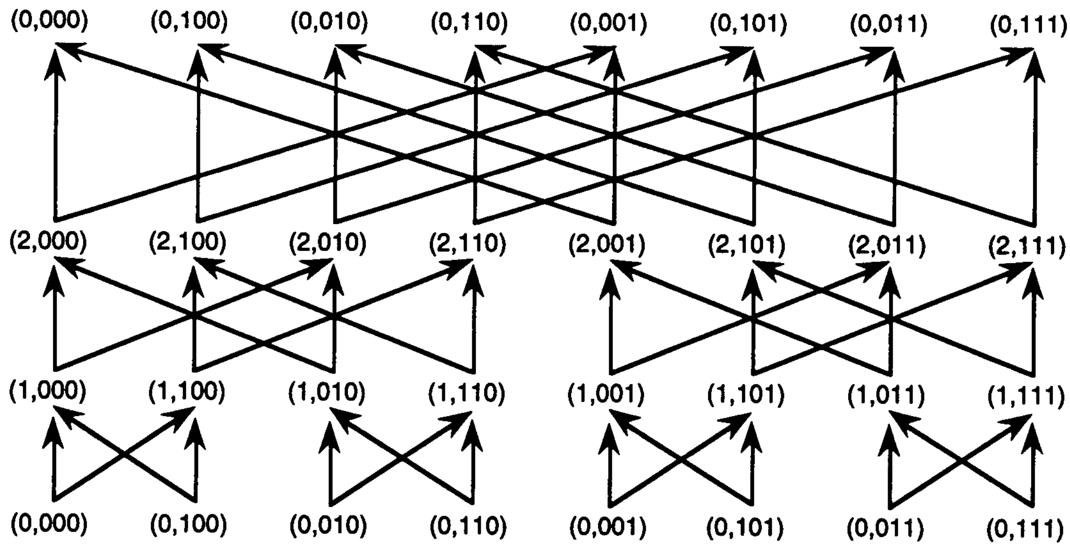

Figure 3.4. Butterfly digraph F(2,3).

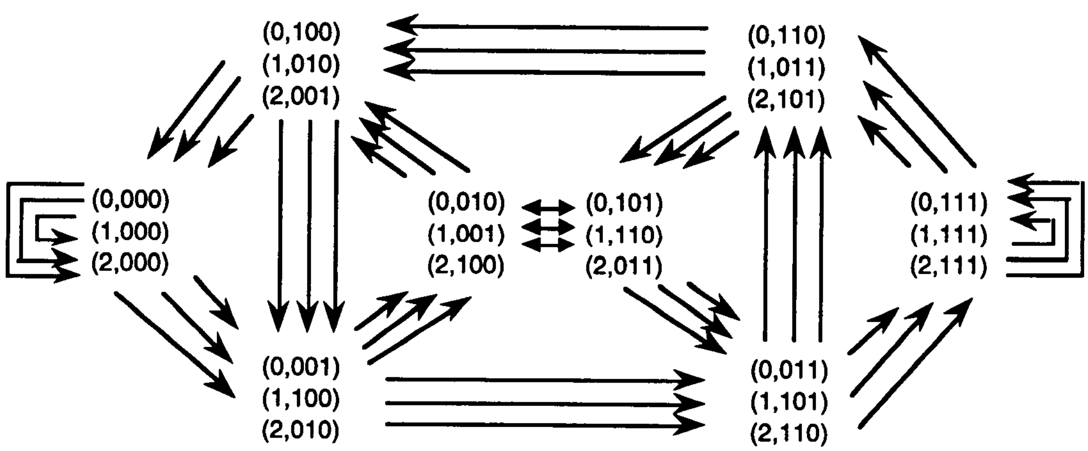

Figure 3.5. Butterfly graph F(2,3) partitioned to resemble B(2,3).

Lemma 3.8 *If there is an edge from node X to node Y in B(d,n) then there is an edge from the level i butterfly node in S_X to the level i+1 butterfly node in S_Y for i = 0, ..., n–1.*

PROOF. It suffices to show that $\pi^{-i}(X)$ and $\pi^{-(i+1)}(Y)$ differ at most in the i'th digit. (X,Y) is an edge in B(d,n), so $Y = x_2 \ldots x_n \alpha$ for some α, and $\pi^{-1}(Y) = \alpha x_2 \ldots x_n$. Since X and $\pi^{-1}(Y)$ differ at most in the first digit, it follows that $\pi^{-i}(X)$ and $\pi^{-i}(\pi^{-1}(Y))$ differ at most in the i'th digit. □

Let $S_X^i = (i, \pi^{-i}(X))$ denote the butterfly node at level i in the set corresponding to De Bruijn node X. Note that every butterfly node can be expressed as S_X^i for some X and i. For any k-cycle $C = (V_0, \ldots, V_{k-1})$ in B(d,n) let

$$\Phi(C) \overset{\text{def}}{=} (S_{V_0}^0, S_{V_1}^1, \ldots, S_{V_i}^i, \ldots, S_{V_{t-1}}^{t-1})$$

where t = LCM(k,n). Recall that the superscripts (butterfly levels) are reduced mod n and the subscripts of the V's are reduced mod k.

Lemma 3.9 *Φ(C) is a cycle in F(d, n).*

PROOF. Since t = LCM(k,n), we have $t-1 \equiv n-1 \pmod n$ and $t-1 \equiv k-1 \bmod k$. So, $S_{V_{t-1}}^{t-1} = S_{V_{k-1}}^{n-1}$. Then by Lemma 3.8, there is an edge from $S_{V_{k-1}}^{n-1}$ to $S_{V_0}^0$, so Φ(C) is a closed path in F(d,n). If Φ(C) is not a cycle then $S_{V_i}^i = S_{V_j}^j$ for some $0 \le i < j \le t-1$. In this case, we have $j \equiv i \pmod n$ and $j \equiv i \pmod k \Rightarrow j-i \equiv 0 \pmod n$ and $j-i \equiv 0 \pmod k$. So, $j-i \ge \text{LCM}(k,n)$, a contradiction since $j-i < t$. □

To illustrate Lemma 3.9, consider the 4-cycle C = (110, 100, 001, 011) in B(2,3). Since LCM(4,3) = 12, C is mapped by Φ to a 12-cycle in F(2,3): ((0,110), (1,010), (2,010), (0,011), (1,011), (2,001), (0,001), (1,101), (2,101), (0,100), (1,100), (2,110)).

Lemma 3.10 *A fault-free cycle of length LCM(k,n) exists in F(d,n) in the presence of f faulty edges if a fault-free cycle of length k exists in B(d,n) in the presence of f faulty edges.*

PROOF. Suppose edges $\{E_1, \ldots, E_f\}$ fail in F(d,n), where E_i denote the butterfly edge $S_{U_i}^i \rightarrow S_{V_i}^{i+1}$. Assume that it is always possible to find a cycle of length k in B(d,n) that avoids any set of f faulty edges. If a cycle C in B(d,n) does not include edge $U \rightarrow V$ then Φ(C) does not include edge $S_U^r \rightarrow S_V^{r+1}$ for any r. Consequently, if C is a k-cycle in

B(d,n) that avoids edges $\{U_1 \rightarrow V_1, \ldots, U_f \rightarrow V_f\}$, then Φ(C) avoids edges $\{E_1, \ldots, E_f\}$. The length of Φ(C) is LCM(k,n) by Lemma 3.9 . □

The preceding development allows the results on ring embedding in the presence of edge faults to be extended to butterfly graphs when d and n are relatively prime. Note that in this case Φ maps HCs in B(d,n) to HCs in F(d,n) since LCM(d^n,n) = $n \cdot d^n$.

Proposition 3.5 *F(d,n) admits a fault-free Hamiltonian cycle in the event of at most MAX{$\psi(d)-1$, $\varphi(d)$} edge failures when d and n are relatively prime.*

PROOF. The proof follows from Proposition 3.4, and Lemma 3.10. □

It is also a simple matter to show that Φ maps edge-disjoint cycles in B(d,n) to edge-disjoint cycles in F(d,n). To see this, let C and D be edge-disjoint cycles in B(d,n). If Φ(C) and Φ(D) have a common edge, say $S^i_U \rightarrow S^{i+1}_V$, then C and D have edge $U \rightarrow V$ in common, contradicting the assumption that C and D are edge-disjoint.

Proposition 3.6 *F(d,n) admits $\psi(d)$ disjoint Hamiltonian cycles when d and n are relatively prime.*

PROOF. The proof follows from Proposition 3.2 and the above discussion. □

Chapter 4

Counting Necklaces

The necklace structure of De Bruijn and shuffle-exchange graphs has been studied and exploited by several authors. The results of these investigations include:

- A permutation routing scheme for the N-node shuffle-exchange graph that requires O(logN) steps with high probability [LMR88]. In this approach, necklaces are used as an analogue to the levels of a butterfly network.
- An optimal $O(N^2/\log^2 N)$ area VLSI layout for the N-node shuffle-exchange graph which involves mapping necklaces to the complex plane [Lei83]. The same approach can be used for the De Bruijn graph [SP89].
- An efficient algorithm for constructing Hamiltonian cycles in De Bruijn graphs by joining necklaces is described in references [FM78, Ral81]. The technique is similar to that used in Chapter 3 to construct a fault-free cycle. Related work includes an attempt to count the number of distinct Hamiltonian cycles that can be constructed by joining necklaces [LHC89].

In [LHC89] the authors derive a recurrence for the number of necklaces made up of nodes of a given weight in the binary De Bruijn graph, and in [PI92] Prasad and Iyengar compute asymptotic bounds on the number of necklaces of a given length. In this chapter we derive exact formulae for both of these values en route to developing a general technique for counting necklaces consisting exclusively of nodes that satisfy a wide range of given conditions.

It should be noted that the problem of counting necklaces is not new; P.A. MacMahon derived an expression for the number of d-ary necklaces more than a hundred years ago [Mac92]. In addition, Polya's theorem provides a powerful mechanism for counting necklaces by weight [Liu68].

4.1 Preliminaries

We assume that the nodes of B(d,n) correspond to n-tuples over Z_d. If X is a d-ary n-tuple, then $|X|$ denotes the length of X, and $\pi^i(X)$ denotes the left rotation of X by i positions, e.g., $\pi^2(0001) = 0100$.

Note that $\pi^{i+j}(X) = \pi^i(\pi^j(X))$ and, if $X = W^k$ and $|W| = t$, then $\pi^t(X) = X$. An n-tuple (or node) X is in a necklace of length t if and only if t is the smallest positive integer such that $\pi^t(X) = X$. In this case we say that the *period* of X is t, and write period(X) = t. If period(X) = $|X|$ we say that X is *aperiodic.*

Suppose that X is in a necklace of length t. We can write n as tq + r with $0 \le r < t$, so $\pi^n(X) = \pi^{tq+r}(X) = \pi^r(\pi^{tq}(X)) = \pi^r(X) = X$. It follows that r = 0 because of the minimality of t. Consequently, the length of any necklace in B(d,n) must divide n.

Observation. An n-tuple X is in a necklace of length t if and only if $X = W^{n/t}$ and W is aperiodic.

To verify this, first suppose that X is in a necklace of length t. Then $\pi^t(X) = X$ and $t|n$. We can write X as $W_1W_2 \ldots W_{n/d}$ where $|W_i| = t$ for all i. Then $\pi^t(X) = W_2 \ldots W_{n/d}W_1 = W_1W_2 \ldots W_{n/t}$, so $W_1 = W_2 = \ldots = W_{n/t} = W$, and $X = W^{n/t}$. If $W = Z^k$ for $|Z| < |W|$ then $X = Z^{kn/t}$ and $\pi^{|Z|}(X) = X$, which contradicts the minimality of t. Thus, W is aperiodic. Next, suppose that $X = W^{n/t}$ and that W is aperiodic. If X is in a necklace of length k we must show that t is equal to k. If d = kq + r with $0 \le r < k$, we have $\pi^t(X) = \pi^r(\pi^{kq}(X)) = \pi^r(X) = X$. But k is minimal, so r = 0, and k divides t. We know that $X = Z^{n/k}$ for some Z. But X is also equal to $W^{n/t}$ and $k|t$, so $W = Z^{t/k} = Z^q$. Since W is aperiodic, q must be 1 and hence t = k.

Our counting technique makes use of the *Möbius inversion* which is defined as

$$f(n) = \sum_{t|n} g(t) \;\Rightarrow\; g(n) = \sum_{t|n} f(t)\cdot\mu(n/t)$$

where

$$\mu(x) = \begin{cases} 1 & \text{if } x = 1 \\ (-1)^k & \text{if } x \text{ is a product of } k \text{ distinct primes} \\ 0 & \text{if } x \text{ has a repeated prime factor.} \end{cases}$$

4.2 Counting Necklaces

In this section we derive a formula for the number of necklaces of length $t|n$ in a given subgraph of $B(d,n)$.

Specifically, let f be any function defined on Z_d^* and let g be any function defined on positive integers. Our goal is to count the number of necklaces in the subgraph of $B(d,n)$ consisting of the nodes $\Gamma(n)$ that satisfy $f(X) = g(n)$, i.e.,

$$\Gamma(n) = \{X \in Z_d^n \mid f(X) = g(n)\}.$$

We assume that f and g satisfy the following conditions:

Condition A. If X and Y are in the same necklace then $f(X) = g(n) \Rightarrow f(Y) = g(n)$.

Condition B. If $X \in Z_d^n$ such that $X = W^{n/t}$ and W is aperiodic, then $f(X) = g(n) \Leftrightarrow f(W) = g(t)$.

Condition A insures that $\Gamma(n)$ can be partitioned into necklaces. The importance of Condition B will be demonstrated shortly.

Let

$$A(n,t) = \{X \in Z_d^n \mid f(X) = g(n);\ \text{period}(X) = t\}$$

denote the nodes in $\Gamma(n)$ of period t. Since $A(n,t) = \varnothing$ if t does not divide n, we have the following identity

$$\sum_{t|n} \#A(n,t) = \#\Gamma(n)$$

Applying the Möbius inversion yields an expression

$$\sum_{t|n} \#\Gamma(t)\cdot\mu(n/t) = \#A(n,n). \tag{4.1}$$

for the number of aperiodic nodes in $\Gamma(n)$.

Proposition 4.1 *The number of necklaces of length $t|n$ in $B(d,n)$ containing nodes that satisfy $f(X) = g(n)$ is*

$$\frac{1}{t}\sum_{j|t} \#\Gamma(j)\cdot\mu(t/j). \tag{4.2}$$

PROOF. The desired value can be obtained by dividing #A(n,t) – the number of nodes in necklaces of length t – by t. Using Equation 4.1 we can compute

$$\#A(t,t) = \sum_{j|t} \#\Gamma(j)\cdot\mu(t/j)$$

where $A(t,t) = \{X \in Z_d^t \mid f(X) = g(t);\ \text{period}(X) = t\}$. We conclude the proof by showing that #A(t,t) = #A(n,t) when t|n.

If $X \in A(n,t)$ then $X = W^{n/t}$ for some aperiodic t-tuple W. For each $X \in A(n,t)$ let the appropriate W be denoted h(X), and let $H = \{h(X) \mid X \in A(n,t)\}$. Clearly if X and Y are in A(n,t) and $X \neq Y$, then $h(X) \neq h(Y)$. Thus, #A(n,t) = #H. Note that f(W) = g(t) for all W $\in$ H, since $f(X) = g(n) \Rightarrow f(h(X)) = g(t)$ by Condition B. Therefore, $H \subseteq A(t,t)$. In addition, $W \in A(t,t) \Rightarrow f(W^{n/t}) = g(n)$, by Condition B, so $A(t,t) \subseteq H$. Consequently, #A(t,t) = #H = #A(n,t). □

Proposition 4.2 *The total number of necklaces in B(d,n) containing nodes that satisfy f(X) = g(n) is*

$$\frac{1}{n}\sum_{j|n} \#\Gamma(j)\cdot\phi(n/j) \tag{4.3}$$

where ϕ is the Euler function.

PROOF. To find the total number of necklaces we sum 1/t·#A(n,t) over all t dividing n. By Proposition 4.2 this value is

$$\sum_{t|n} \frac{1}{t}\sum_{j|t} \#\Gamma(j)\cdot\mu(t/j). \tag{4.4}$$

To simplify Equation (4.4) we will use the following identities, the proofs of which can be found in [McE87].

(i) $\displaystyle\sum_{t|n}\sum_{j|t} f(j,t) = \sum_{j|n}\ \sum_{t|n/j} f(j,t\cdot j)$,

(ii) $\displaystyle\sum_{t|n} \mu(t)/t = \phi(n)/n$.

Applying first Identity (i) and then Identity (ii) to Equation (4.4) yields

$$\sum_{j|n} \frac{1}{j}\#\Gamma(j)\sum_{t|n/j} \mu(t)/t = \frac{1}{n}\sum_{j|n} \#\Gamma(j)\cdot\phi(n/j).$$ □

4.3 Examples

In this section we illustrate the usefulness of the Propositions 4.1 and 4.2 with some concrete examples.

Counting by Length

To determine the number of necklaces of length t in B(d,n) we can let f(X) = 0 for all X, so that $\#\Gamma(m) = \#\{X \in Z_d^m\} = d^m$. Then, by Proposition 4.1, the number of necklaces of length t|n in B(d,n) is

$$\frac{1}{t}\sum_{j|t} d^j \cdot \mu(t/j),$$

and, by Proposition 4.2, the total number of necklaces in B(d,n) is

$$\frac{1}{n}\sum_{j|n} d^j \cdot \phi(n/j).$$

For instance, the number of necklaces of length 6 in B(2,12) is

$$\begin{aligned}\frac{1}{6}\sum_{j|6} 2^j \cdot \mu(6/j) &= \frac{1}{6}[2\cdot\mu(6)+2^2\cdot\mu(3)+2^3\cdot\mu(2)+2^6\cdot\mu(1)]\\ &= \frac{1}{6}[2 - 4 - 8+64] = 9,\end{aligned}$$

and the total number of necklaces in B(2,12) is

$$\begin{aligned}\frac{1}{12}\sum_{j|12} 2^j \cdot \phi(12/j) &= \frac{1}{12}[2\cdot\phi(12)+2^2\cdot\phi(6)+2^3\cdot\phi(4)+2^4\cdot\phi(3)+2^6\cdot\phi(2)+2^{12}\cdot\phi(1)]\\ &= \frac{1}{12}[8 + 8 + 16 + 32 + 64 + 4096] = 352.\end{aligned}$$

Counting by Weight in B(2,n)

The number of necklaces of length t in B(2,n) made up of nodes of weight k can be counted by choosing f(X) to be wt(X) and g(m) to be (k/n)·m.

The weight function clearly satisfies Property A. To verify that Property B also holds, suppose that wt(X) = g(m) for some m-tuple X. If $X = W^{m/t}$, then wt(W) = (t/m)wt(X) = (t/m)(k/n)·m = (k/n)t = g(t). Conversely, if wt(W) = g(t) then wt(X) = g(m).

The number of binary m-tuples of weight km/n is

$$\#\Gamma(m) = \#\{X \in Z_2^m \mid wt(X) = mk/n\} = \binom{m}{mk/n}.$$

Thus, by Proposition 4.1, the number of necklaces of weight k and length t is

$$\frac{1}{t}\sum_{j|t} \binom{j}{jk/n}\cdot\mu(t/j),$$

and by Proposition 4.2, the total number of necklaces of weight k

$$\frac{1}{n}\sum_{j|n} \binom{j}{jk/n}\cdot\phi(n/j).$$

For instance, the number of necklaces of weight 4 and length 6 in B(2,12) is

$$\frac{1}{6}\sum_{j|6} \binom{j}{j/3}\cdot\mu(6/j) = \frac{1}{6}\left[\binom{6}{2}\mu(1) + \binom{3}{1}\mu(2)\right] = \frac{1}{6}\,[15 - 3] = 2,$$

and the total number of necklaces of weight 4 in B(2,12) is

$$\frac{1}{12}\left[\binom{12}{4}\phi(1) + \binom{6}{2}\phi(2) + \binom{3}{1}\phi(4)\right] = \frac{1}{12}[495 + 15 + 6] = 43.$$

Counting by Weight in B(d,n)

When $d > 2$ we use the same f and g; however, $\#\Gamma(m)$ is different. Let $\zeta_d(n,k)$ denote the number of d-ary n-tuples of weight k, so that $\#\Gamma(m) = \#\{X \in Z_d^m \mid wt(X) = km/n\} = \zeta_d(m,km/n)$.

An expression for $\zeta_d(n,k)$ can be found by observing that $\zeta_d(n,k)$ is the number of ways to choose k out of n objects subject to the restriction that each object may be chosen at most d–1 times. The generating function for this value is given in [Knu73] as

$$g(z) = \sum_{k=0}^{n(d-1)} \zeta_d(n,k)\cdot z^k = (1 + z + \ldots + z^{d-1})^n = (1-z^d)^n\,(1-z)^{-n}.$$

Recalling that $(1-z^d)^n = \sum_{i=0}^{n}(-1)^i\binom{n}{i}\cdot z^{di}$ and $(1-z)^{-n} = \sum_{i=0}^{\infty}\binom{n-1+i}{i}\cdot z^i$, and equating the coefficients of like powers, we get

$$\zeta_d(n,k) = \sum_{i=0}^{k/d} (-1)^i \binom{n}{i}\binom{n-1+k-di}{n-1}.$$

For instance, the number of necklaces of weight 4 and length 4 in B(3,4) is

$$\frac{1}{4}\sum_{j|4} \zeta_3(j,j)\cdot\mu(4/j) \;=\; \frac{1}{4}\,[\zeta_3(4,4)\mu(1)+ \zeta_3(2,2)\mu(2)+ \zeta_3(1,1)\mu(4)] = \frac{1}{4}\,[19-3] = 4.$$

Counting by Type

Nodes of a given weight in B(d,n) can be further partitioned by defining the *type* of a node X to be a d-tuple $K = [k_0, \ldots, k_{d-1}]$ where σ appears k_σ times in X. For example, 312211 is of type [0,3,2,1]. The number of d-ary n-tuples of type K is $\frac{n!}{k_0!\ldots k_{d-1}!}$.

We can count the number of necklaces of type K in B(d,n) in much the same way that we counted necklaces of a given weight. Let f(X) denote the type of X, and let $g(m) = [mk_0/n, \ldots, mk_{d-1}/n]$. It is easy to verify that f and g satisfy Conditions A and B.

By Proposition 4.1, the number of necklaces in B(d,n) of length t and type K is

$$\frac{1}{t}\sum_{j|t} \#\Gamma(j)\cdot\mu(t/j).$$

where $\#\Gamma(j) = \dfrac{j!}{(jk_0/n)!\ldots(jk_{d-1}/n)!}$.

Note that when d = 2, type(X) = [n – k, k] if and only if wt(X) = k. In this event, $\#\Gamma(j)$ becomes $\binom{j}{jk/n}$ as expected.

Chapter 5

Future Work

Numerous authors have assessed the reliability of De Bruijn networks using connectivity-based arguments. In this approach the d-ary De Bruijn network is said to be able to tolerate at most d–2 node or edge failures because the network may be disconnected if more failures occur. The results of this thesis indicate that the De Bruijn network can also efficiently support algorithms requiring ring-structured communication in the event of d–2 component failures.

In particular, it has been shown that when $f \le d-2$ nodes fail, a fault-free cycle of length at least $d^n - n \cdot f$ can always be found in B(d,n). It was also shown that a fault-free Hamiltonian cycle exists in the event of d–2 edge failures when d is a prime power. Both results are optimal when a worst-case distribution of faults is assumed.

The results on ring embedding in the presence of edge failures are difficult to generalize because they rely on properties of finite fields of size d. A different approach may be required to answer the following questions:

1) Does B(d,n) admit a fault-free Hamiltonian cycle in the presence of d–2 edge failures for all values of d?

2) Does B(d,n) admit d–1 disjoint Hamiltonian cycles?

Question 2 was answered affirmatively in Section 3.2 for the case when d is a power of 2.

Another line of inquiry is motivated by the fact that the undirected De Bruijn graph UB(d,n) is more appropriate than B(d,n) as a model for interconnection networks with bidirectional communication links. This raises the question of whether significantly longer fault-free cycles can be embedded in UB(d,n) in the presence of component failures? Of particular interest in light of the fact that the connectivity of UB(d,n) is twice that of B(d,n) [EH85]:

3) Does UB(d,n) admit a fault-free cycle of length at least $d^n - n \cdot f$ in the presence of $f < 2(d-1)$ node failures? This is the undirected version of Proposition 2.2.

4) Does UB(d,n) admit a fault-free Hamiltonian cycle in the presence of 2(d–2) edge failures? If more than 2(d–2) edges fail it is possible that a node may not have at least two nonfaulty incident edges.

Other areas of research include determining the maximum number of disjoint Hamiltonian cycles present in other bounded degree graphs, such as butterfly graphs and Kautz graphs [BP89].

Bibliography

[ABR90] F. Annexstein, M. Baumslag, and A. L. Rosenberg, "Group action graphs and parallel architectures," *SIAM J. Computing*, 19 (June 1990) pp. 544-569.

[ABS90] B. Alspach, J.-C. Bermond, and D. Sotteau, "Decomposition into cycles I: Hamiltonian decompositions," in *Cycles and Rays*, G. Hahn, G Sabidussi, and R. E. Woodrow (eds.), Kluwer Academic, 1990, pp. 9-18.

[Ann89] F. Annextein, "Fault-tolerance of hypercube-derivative networks," in *Proc. 1st ACM Symposium on Parallel Algorithms and Architectures* (1989), pp. 179-188.

[AV84] M. Atallah and U. Vishkin, "Finding Euler tours in parallel," *J. Systems and Sciences*, vol. 29 (1984), pp. 330-337.

[BBR93] D. Barth, J. Bond, and A. Raspaud, "Compatible Eulerian Circuits in K_n^{**}," Technical Report 93-13, LaBri-Université Bordeaux I.

[BCH92] J. Bruck, R. Cypher, and C.-T. Ho, "Fault-tolerant De Bruijn and shuffle exchange networks," *Proc. of the 1992 International Conference on Parallel Processing*, vol. III (1992), pp. 46-50.

[BHL+92] J-C. Bermond, P. Hell, A. L. Liestman and J. G. Peters, "Broadcasting in bounded degree graphs," *SIAM J. Discrete Math.* 5 (1992), pp. 10-24.

[BP88] J-C. Bermond and C. Peyrat, "Broadcasting in de Bruijn networks," in *Proc. 19-th Southeastern Conference on Combinatorics, Graph Theory and Computing,* 1988.

[BP89] J-C. Bermond and C. Peyrat, "De Bruijn and Kautz networks: a competitor for the hypercube?," in *Hypercube and Distributed Computers,* edited by F. André and J.P. Verjus, Holland: Elsevier, 1989.

[Bru46] N. G. de Bruijn, "A combinatorial problem," *Proc. Koninklijke Nederlandsche Akademie van Wetenschappen*, vol. 49 (1946) pp. 758-764.

[CL91a] M.Y. Chan and S.-J. Lee, "Distributed fault-tolerant embeddings of rings in hypercubes," *J. Parallel and Distributed Computing* 11 (1991), pp. 63-71.

[CL91b] M.Y. Chan and S.-J. Lee, "On the existence of Hamiltonian circuits in faulty hypercubes," *SIAM J. Discrete Math.* vol. 4 (November 1991), pp. 511-527.

[Cul80] P. Cull, "Tours of graphs, digraphs, and sequential machines," *IEEE Trans. Computers*, vol. C-29 (January 1980), pp. 50-54.

[DF84] C. Delorme and G. Farhi, "Large graphs with given degree and diameter - Part 1," *IEEE Trans. Computers*, vol. C-33 (May 1984), pp. 857-860.

[DKM92] S. Dolinar, T.-M. Ko and R. McEliece, "Some VLSI decompositions of the de Bruijn graph," *Discrete Mathematics* 106/107 (1992), pp. 189-198.

[DLH93] D-.Z. Du, Y-.D. Lyuu, D. F. Hsu, "Line graph iterations and connectivity analysis of De Bruijn and Kautz graphs," *IEEE Trans. Computers*, vol. 42 (May 1993), pp. 612-616.

[EH85] A.-H. Esfahanian and S. L. Hakimi, "Fault-tolerant routing in De Bruijn communication networks," *IEEE Trans. Comput.*, C-34 (September 1985) pp. 777-788.

[EL84] T. Etzion and A. Lempel, "Algorithms for the generation of full-length shift-register sequences," *IEEE Trans. Information Theory*, vol. 30 (May 1984), pp. 480-484.

[Etz86] T. Etzion, "An algorithm for constructing m-ary de Bruijn sequences," *J. Algorithms*, 7 (1986), pp. 331-340.

[FM78] H. Fredricksen and J. Maiorana, "Necklaces of beads in k colors and k-ary De Bruijn sequences," *Discrete Math.*, 23 (1978), pp. 207-210.

[Fre82] H. Fredricksen, "A survey of full length nonlinear shift register cycle algorithms," *SIAM Review*, 24 (1982), pp. 195-221.

[Gol82] S. W. Golomb, *Shift Register Sequences*, Laguna Hills, CA: Aegean Park Press, 1982.

[Har69] F. Harary, *Graph Theory*, Reading, Mass: Addison-Wesley, 1969.

[HP89] N. Homobono and C. Peyrat, "Fault tolerant routings in De Bruijn and Kautz networks," *Discrete Applied Mathematics*, vol. 24 (1989), pp. 179-186.

[Hua90] Y. Huang, "A new algorithm for the generation of binary de Bruijn sequences," *J. Algorithms*, 11 (1990), pp. 44-51.

[ISO85] M. Imase, T. Soneoka and K. Okada, "Connectivity of regular directed graphs with small diameters,"" *IEEE Trans. Comput.*, vol. C-34 (March 1985), pp. 267-273.

[ISO86] M. Imase, T. Soneoka and K. Okada, "Fault-tolerant processor interconnection networks," *Systems and Computers in Japan*, 17 (1986), pp. 21-30.

[Knu73] D. E. Knuth, *The Art of Computer Programming, vol. 1: Fundamental Algorithms*, 2nd ed. Reading, MA: Addison-Wesley, 1973.

[KLM+89] R. Koch, T. Leighton, B. Maggs, S. Rao, and A. Rosenberg, "Work-preserving emulations of fixed-connection networks," *Proc. 21st ACM Symp. on Theory of Computing* (1989), pp. 227-240.

[Lei83] F. T. Leighton, *Optimal Layouts for the Shuffle-Exchange Graph and Other Networks*. Cambridge, Mass.: The MIT Press, 1983.

[Lei92] F. T. Leighton, *Introduction to Parallel Algoriths and Architectures: Arrays, Trees, Hypercubes*. San Mateo, CA: Morgan Kaufmann, 1992.

[Lem71] A. Lempel, "M-ary closed sequences," *J. Combinatorial Theory* vol. 10, 1971, pp. 253-258.

[LHC89] W. Liu, T.H. Hildebrandt, and R. Cavin III, "Hamiltonian cycles in the shuffle-exchange network," *IEEE Trans. Comput.*, vol. C-38 (May 1989), pp. 745-750.

[Liu68] C. L. Liu, *Introduction to Combinatorial Mathematics*. New York: McGraw-Hill, 1968.

[LMR88] T. Leighton, B. Maggs, and S. Rao, "Universal packet routing," *Proc. 29th Symp. Foundations of Computer Science* (1988), 256-268.

[LP84] R. Lidl and G. Pilz, *Applied Abstract Algebra*, New York: Springer-Verlag, 1984.

[LS90] S. Lee and K. G. Shin, "Interleaved all-to-all reliable broadcast on meshes and hypercubes," in *Proc. 1990 Int. Conf. Parallel Processing*, vol. 3 (1990), pp. 110-113.

[LW92] J. H. van Lint and R. M. Wilson, *A Course in Combinatorics*. Cambridge University Press, 1992.

[Lyu93] Y.-D. Lyuu, "Fast fault-tolerant parallel communication for de Bruijn and digit-exchange networks using information dispersal," *Networks*, Vol. 23 (1993), 365-378.

[LZB92] S. Latifi, S.-Q. Zheng and N. Bagherzadeh, "Optimal ring embedding in hypercubes with faulty links," *Proc. Int'l Symposium on Fault-Tolerant Computing* (1992), pp. 178-184.

[Mac92] P. A. MacMahon, "Applications of a theory of permutations in circular procession to the theory of numbers," in *Proc. London Mathematical Society*, vol. XXII, pp. 305-313, 1892.

[McE87] R. J. McEliece, *Finite Fields for Computer Scientists and Engineers*, Boston, Mass: Kluwer Academic, 1987.

[Muk92] B. Mukherjee, "WDM-based local lightwave networks part II: multihop systems," *IEEE Network* (July 1992), pp. 20-32.

[Obr91] B. Obrenic , "Embedding De Bruijn and shuffle-exchange graphs in five pages," in *Proc. 3rd ACM Symposium on Parallel Algorithms and Architectures* (1991), pp. 137-146.

[Pra81] D. K. Pradhan, "Interconnection topologies for fault-tolerant parallel and distributed architectures," *Proc. 10th Int'l. Conf. Parallel Processing* (1981), pp. 238-242.

[PI92] L. Prasad and S.S. Iyengar, "An asymptotic equality for the number of necklaces in a shuffle-exchange network," *Theoretical Computer Science* 102 (1992), pp. 355-365.

[PR82] D. K. Pradhan and S. M. Reddy, "A fault-tolerant communication architecture for distributed systems," *IEEE Trans. Comput.*, vol. C-31, pp. 863-869, September 1982.

[PV81] F. P. Preparata and J. Vuillemin, "The cube-connected cycles: a versatile network for parallel computation," *Communications of the ACM*, May 1981, pp. 300-309.

[Ral81] A. Ralston, "A new memoryless algorithm for De Bruijn sequences," *J. Algorithms*, 2 (1981), pp. 50-62.

[Ral82] A. Ralston, "De Bruijn sequences – a model example of the interaction of discrete mathematics and computer science," *Mathematics Magazine*, vol. 55 (May 1982), pp. 131-143.

[RB90] R. Rowley and B. Bose, "On necklaces in shuffle-exchange and de Bruijn networks," in *Proc. 1990 Int. Conf. Parallel Processing*, vol.1, August 1990, pp. 347-350.

[RB91a] R. Rowley and B. Bose, "Edge-disjoint Hamiltonian cycles in de Bruijn networks," *Proc. Sixth Distributed Memory Computing Conference*, 1991, pp. 707-709.

[RB93a] R. Rowley and B. Bose, "On the Number of Arc-Disjoint Hamiltonian Circuits in the De Bruijn Graph," *Parallel Processing Letters, Special Issue on Algorithmic and Structural Aspects of Interconnection Networks* (December 1993).

[RB93b] R. Rowley and B. Bose, "A distributed algorithm for finding a fault-free cycle in a De Bruijn network," *Proc. ISCA International Conference on Parallel and Distributed Computing and Systems* (1993), pp. 263-266.

[RB93c] R. Rowley and B. Bose, "Fault-tolerant ring embedding in de Bruijn networks," *IEEE Trans. Computers*, vol 42 (December 1993), pp. 1480-1486. (see also *Proc. 1991 Int. Conf. Parallel Processing*, vol 1, August 1991, pp. 710-711.

[Ree46] D. Rees, "A note on a paper by I. J. Good," *J. London Math. Soc.*, vol. 21 (1946), pp. 169-172.

[Ros84] K.H. Rosen, *Elementary Number Theory and its Applications*, Reading, Mass: Addison-Wesley, 1984.

[Sch74] M. L. Schlumberger, "De Bruijn communication networks," Ph.D. dissertation, Department of Computer Science, Stanford Univ., 1974.

[Sch92] E. J. Schwabe, "Embedding meshes of trees into deBruijn graphs," *Information Processing Letters* 43 (1992), pp. 237-240.

[SP89] M. R. Samatham and D. K. Pradhan, "The De Bruijn multiprocessor network: a versatile parallel processing and sorting network for VLSI," *IEEE Trans. Comput.*, vol. C-38 (April 1989), pp. 567-581.

[SP91] M. R. Samatham and D. K. Pradhan, "Corrections to the De Bruijn multiprocessor network: a versatile parallel processing and sorting network for VLSI," *IEEE Trans. Comput.*, vol. C-40 (January 1991), pp. 122-123.

[SR91a] K. Sivarajan and R. Ramaswami, "Multihop lightwave networks based on de Bruijn graphs ," in *Proc. IEEE Infocom* (1991), pp. 1001-1011.

[SR91b] M. A. Sridhar and C. S. Raghavendra, "Fault-tolerant networks based on the de Bruijn graph," *IEEE Trans. Comput.*, vol. 40 (October 1991) pp. 1167-1174.

[Sut91] K. Sutner, "De Bruijn graphs and linear cellular automata," *Complex Systems*, vol. 5 (1991), pp. 19-30.

[Ull84] J. D. Ullman, *Computational Aspects of VLSI.* Rockville, MD: Computer Science Press, 1984.

[VB81] L. G. Valiant and G. J. Brebner, "Universal schemes for parallel communication," in *Proc. 13th Symp. Theory Comp.* (1981) 263-277.

[WC92] A. Wang and R. Cypher, "Fault-tolerant embeddings of rings, meshes, and tori in hypercubes," *Proc. 4th IEEE Symp. on Parallel and Distributed Computing*, 20-29 (1992).

[Wol84] S. Wolfram, "Computation theory of cellular automata," *Communications in Mathematical Physics*, vol. 96 (1984), pp. 15-57.