

# DE BRUIJN GRAPHS AND THEIR APPLICATIONS TO FAULT TOLERANT NETWORKS

JOEL BAKER

ABSTRACT. The goal of this expository paper is to introduce De Bruijn graphs and discuss their applications to fault tolerant networks. We will begin by examining N.G. de Bruijn's original paper and the proof of his claim that there are exactly  $2^{2^{n-1}-n}$  De Bruijn cycles in the binary De Bruijn graph  $\mathcal{B}(2, n)$ . In order to study fault tolerance we explore the properties of Hamiltonian and Eulerian cycles that occur on De Bruijn graphs and the type of redundancy that occurs as a result. Lastly, in this paper we seek to provide some guidance into further research on De Bruijn graphs and their potential applications to other areas.

## 1. INTRODUCTION

De Bruijn graphs are an interesting class of graphs that have applications in a variety of different areas. A De Bruijn graph is a directed graph with  $d^n$  nodes labeled by  $n$ -tuples over a  $d$ -character alphabet (denoted by juxtaposition). The edges are defined to be ordered pairs of the form  $((\alpha_1 \dots \alpha_n), (\alpha_2 \dots \alpha_n \alpha_{n+1}))$  where  $\alpha_{n+1}$  is any character in the alphabet. We will denote the De Bruijn graph  $\mathcal{B}(d, n)$ . In Figure 1 below we see two examples of these graphs. Properties discussed in this paper show that they are a natural choice for constructing interconnection networks, which are networks for which fault tolerance is important. Fault tolerance is a networks ability to maintain communication even if certain communication paths are destroyed. We will study the ability of networks to handle the failure of both nodes and edges.

In this paper we will first present some of the properties of De Bruijn graphs, including ways in which they can be applied both in theory and in practice. Before we examine applications of De Bruijn graphs, we will discuss the original theorem which De Bruijn proved which is the beginning of the study of the structure of De Bruijn Graphs. De Bruijn proved that a De Bruijn graph exhibits a predictable amount of Hamiltonian cycles. His proof centers around the fact that in the binary De Bruijn graph with  $2^n$  nodes there will exist  $2^{2^{n-1}-n}$  different Hamiltonian cycles. This trait discovered by De Bruijn is useful for applications of De Bruijn graphs to fault tolerant networks.

De Bruijn graphs are natural choices for a network's connection layout. When examining a network, i.e. a structure with objects connected by communication paths, there are a few desirable traits which make De Bruijn graphs a logical choice for investigation. Esfahanian and Hakimi [EH85] discuss these traits; they point out that we want the number of connections from each processor to be relatively small, but we also want there to be

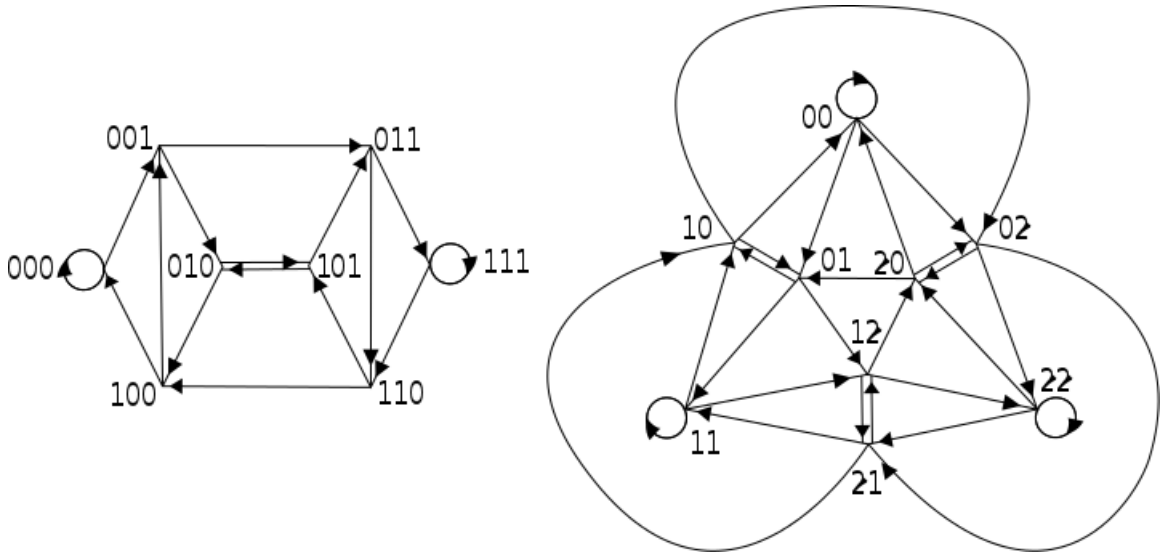


FIGURE 1. Two different De Bruijn graphs.

short paths between any two processors. We will examine this problem in terms of graphs which use nodes to represent processors and edges to represent the communication paths between these processors. We will see that the De Bruijn graphs make excellent choices for a network's structure since they:

- (1) have a relatively large number of nodes
- (2) have few connections at each node and
- (3) still maintain short paths between nodes.

These three traits make De Bruijn graphs excellent graphs for further investigation.

The remainder of this paper is organized as follows. In Section 2 we introduce the De Bruijn graph and other relevant terms. In Section 3 we will examine N.G. de Bruijn's original proof that there are  $2^{2^{n-1}-n}$  Hamiltonian cycles in the De Bruijn graph  $\mathcal{B}(2, n)$ . Following that, in Section 4 we will focus on fault tolerance. This section has been divided up into two parts. The first is aimed at understanding the type of connectivity that we have if there are node failures and the second is focused on edge failure.

## 2. PRELIMINARIES

**2.1. Terminology and Definitions.** In order to define De Bruijn graphs, we must first begin with the definition of graph.

**Definition 1.** A **graph** is a an ordered pair  $G = (V, E)$  where  $V$  is a set of vertices called nodes, together with a set  $E$  of edges which are pairs of nodes. Unless otherwise stated, the graphs mentioned will be **directed graphs**, i.e. graphs whose edges are an ordered pair of nodes.

*Example.* See Figure 2 below for an example of both a directed and undirected graph.

**Definition 2.** A **subgraph**,  $G' = (V', E')$ , of a graph  $G = (V, E)$ , is a graph on a subset  $V'$  of  $V$  with the property that every edge  $e \in E$  with endpoints in  $V'$  is also in  $E'$ .

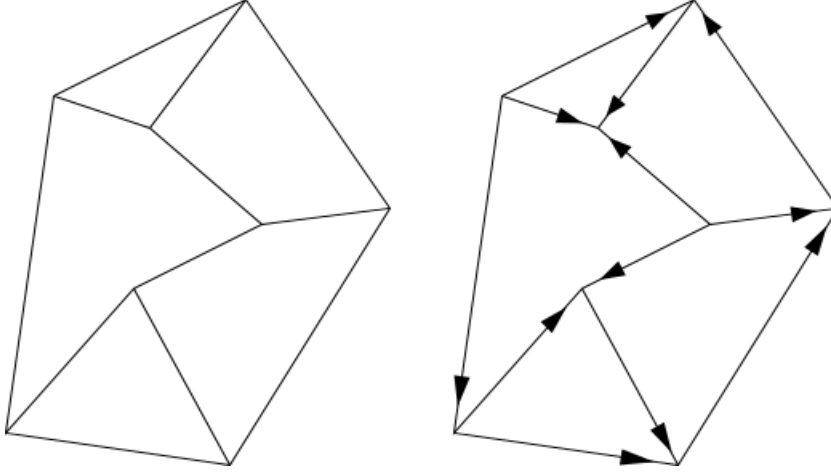


FIGURE 2. An undirected graph (left) and a directed version (right).

**Definition 3.** A  $(\beta_1 \dots \beta_n)$  is called a **successor** of node  $(\alpha_1 \dots \alpha_n)$  if there is an edge from  $(\alpha_1 \dots \alpha_n)$  to  $(\beta_1 \dots \beta_n)$ . Likewise,  $(\alpha_1 \dots \alpha_n)$  is said to be a **predecessor** of  $(\beta_1 \dots \beta_n)$ . We will say two nodes are **adjacent nodes** if there is an edge between them and that two edges are **adjacent edges** if they share a common node.

**Definition 4.** A **De Bruijn graph** is a directed graph with  $d^n$  nodes labeled by  $n$ -tuples over a  $d$ -character alphabet (denoted by juxtaposition). The edges are defined to be ordered pairs of the form  $((\alpha_1 \dots \alpha_n), (\alpha_2 \dots \alpha_n \alpha_{n+1}))$  where  $\alpha_{n+1}$  is any character in the alphabet. We will denote the De Bruijn graph  $\mathcal{B}(d, n)$ .

*Example.* The De Bruijn graph  $\mathcal{B}(3, 4)$  has the node  $(1022)$  (a 4-tuple with alphabet  $\{0, 1, 2, 3\}$ ) which has successors  $(0220)$ ,  $(0221)$ ,  $(0222)$  and  $(0223)$ .

*Example.* The De Bruijn graph  $\mathcal{B}(2, 2)$  is given below in Figure 3.

**Definition 5.** In [EH85], Esfahanian and Hakimi discuss a modified version of a DeBruijn graph called an **undirected De Bruijn graph**, denoted  $\mathcal{UB}(d, n)$ . An undirected De Bruijn graph is a De Bruijn graph modified so that:

- (1) All edges which are self loops are removed.
- (2) If  $\overrightarrow{\alpha\beta}$  is an edge in  $\mathcal{B}(d, n)$ , then the undirected edge  $\overline{\alpha\beta}$  is an edge in  $\mathcal{UB}(d, n)$ .
- (3) If  $\overrightarrow{\alpha\beta}$  and  $\overrightarrow{\beta\alpha}$  are both edges in  $\mathcal{B}(d, n)$  then there is only the single undirected edge  $\overline{\alpha\beta}$  in  $\mathcal{UB}(d, n)$ .

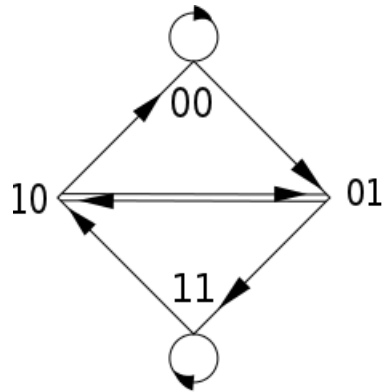


FIGURE 3. The De Bruijn graph with alphabet  $\{0, 1\}$  whose nodes are 2-tuples.

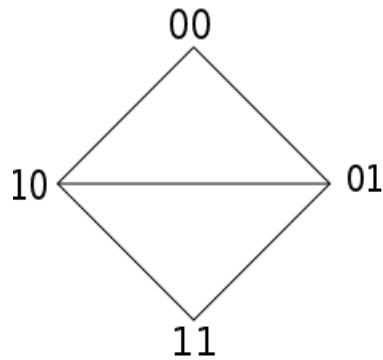


FIGURE 4. The undirected De Bruijn graph  $\mathcal{UB}(2, 2)$ .

The undirected De Bruijn graph  $\mathcal{UB}(2, 2)$  is given below in Figure 4

In order to understand applications to fault tolerant networks, we need to understand some of the terms used to describe the structure of graphs. In particular, we need to define the notions of both in-degree and out-degree, and also Hamiltonian and Eulerian cycles.

**Definition 6.** The **in-degree** of a node in a directed graph is the number of edges that end at that node and similarly, the **out-degree** of a node will be the number of edges that start at that node.

Upon examining Definition 4, we outline a few basic properties of De Bruijn graphs:

- (1) The number of nodes in  $\mathcal{B}(d, n)$  is  $d^n$  since each  $n$ -tuple is a node.

- (2) The number of edges in  $\mathcal{B}(d, n)$  is  $d^{n+1}$  since there are  $d^n$  nodes and out-degree of  $d$  on each node.
- (3) Every node in  $\mathcal{B}(d, n)$  has out-degree  $d$  since every successor of  $(\alpha_1 \dots \alpha_n)$  has the form  $(\alpha_2 \alpha_3 \dots \alpha_n \beta)$  and there are  $d$  choices for  $\beta$ .
- (4) Every node in  $\mathcal{B}(d, n)$  has in-degree  $d$  since if we examine the predecessors of  $(\alpha_1 \dots \alpha_n)$  we see that they will be those which look like  $(\beta \alpha_1 \dots \alpha_{n-1})$  with any of the  $d$  choices for  $\beta$ .
- (5) Since the edges are defined explicitly, we see that the graph  $\mathcal{B}(d, n)$  is unique for fixed  $d$  and  $n$ .

**Definition 7.** A **path** is a sequence of edges for which each edge begins where the previous edge in the sequence ended. We will denote a path

$$(\alpha_1 \dots \alpha_n)(\alpha_2 \dots \alpha_{n+1}) \dots (\alpha_i \dots \alpha_{n+i-1})$$

of length  $i$  instead by  $\alpha_1 \alpha_2 \alpha_3 \dots \alpha_{n+i-1}$ . A **cycle** is a path whose start node and end node are the same. We will denote the corresponding cycle on  $\mathcal{B}(d, n)$  whose path is  $\alpha_1 \dots \alpha_{n+i-1} \alpha_1 \alpha_2 \dots \alpha_n$  by

$$[\alpha_1 \dots \alpha_{n+i-1}].$$

*Example.* In Figure 5 we see there is a path 000101 which we can see comes from the long form of the path  $(\underline{000})(\underline{001})(\underline{010})(\underline{101})$  of  $\mathcal{B}(2, 3)$  (using underlines to highlight which characters are in our short form).

**Definition 8.** A graph is **connected** if for every pair of nodes,  $\alpha$  and  $\beta$ , there exists a path from  $\alpha$  to  $\beta$  and a path from  $\beta$  to  $\alpha$ .

**Definition 9.** A subgraph of a graph  $G$  is called a **component** of  $G$  if

- (1) Any two nodes in the subgraph are connected.
- (2) Any node in the graph which is connected to a node in the subgraph is also in the subgraph.

*Example.* In Figure 5, the cycle [000101] is the path 000101000, written in long form as

$$(000)(001)(010)(101)(010)(100)(000).$$

**Definition 10.** A **Hamiltonian cycle** of a graph  $G$  is a cycle of  $G$  which visits every node exactly once. An **Eulerian cycle** of  $G$  is a cycle of  $G$  which traverses every edge exactly once.

*Example.* The highlighted cycle in Figure 5 is the Hamiltonian cycle [11010001] which is described by starting at the node (110). We also see that [1111000010011010] is an Eulerian cycle on  $\mathcal{B}(2, 3)$ , starting at the node (111).

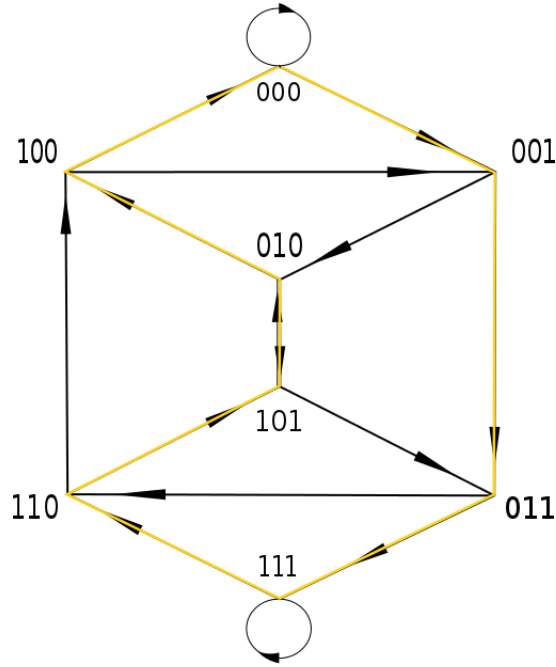


FIGURE 5. The graph above is the De Bruijn graph  $\mathcal{B}(2, 3)$  with alphabet  $\{0,1\}$  and nodes which are 3-tuples. The highlighted cycle is a Hamiltonian cycle on the graph.

**Definition 11.** We will call the sequence of characters  $[\alpha_1, \alpha_2, \dots, \alpha_{m+n}]$  a **De Bruijn sequence** when  $[\alpha_1 \dots \alpha_{n+m}]$  is a Hamiltonian cycle. Since there are  $d^n$  nodes in a De Bruijn graph, then a De Bruijn sequence will necessarily be  $d^n$  characters long.

Without use of the geometric representation, a De Bruijn sequence can also be described as a  $d^n$  character cyclic string in which each possible  $n$ -tuple from  $d$  characters occurs exactly once. We will say that two sequences  $[\alpha_1, \dots, \alpha_{d^n}]$  and  $[\beta_1, \dots, \beta_{d^n}]$  are equal if there exists some  $c$  such that for all  $i \in 1, \dots, d^n$  we have that  $\alpha_i = \beta_{i+c} \pmod{d^n}$ . This is equivalent to saying that the sequences correspond to the same cycle on the De Bruijn graph.

*Example.* Using the highlighted Hamiltonian cycle of  $\mathcal{B}(2, 3)$  in Figure 5 above, we see that  $[0, 0, 0, 1, 1, 1, 0, 1]$  is a De Bruijn sequence.

**2.2. The Doubling of a De Bruijn Graph.** An additional interesting property of De Bruijn graphs is that we can expand the De Bruijn graph  $\mathcal{B}(d, n)$  to the graph  $\mathcal{B}(d, n+1)$  with the following procedure.

**Procedure 1.** Using our standard notation as defined above we can create the De Bruijn graph  $\mathcal{B}(d, n+1)$  by the following steps:

- (1) Construct nodes in  $\mathcal{B}(d, n + 1)$  from  $\mathcal{B}(d, n)$  by allowing the edges of  $\mathcal{B}(d, n)$  to correspond to nodes of  $\mathcal{B}(d, n + 1)$ . Specifically, the adjacent nodes  $(\alpha_1 \dots \alpha_n)$  and  $(\alpha_2 \dots \alpha_{n+1})$  form the edge  $(\alpha_1 \dots \alpha_n)(\alpha_2 \dots \alpha_{n+1})$  in  $\mathcal{B}(d, n)$  and correspond to node  $(\alpha_1 \dots \alpha_{n+1})$  in  $\mathcal{B}(d, n + 1)$ .

*Example.* In the doubling from  $\mathcal{B}(3, 3)$  to  $\mathcal{B}(3, 4)$  the edge  $(201)(011)$  in  $\mathcal{B}(3, 3)$  gives the node  $(2011)$  in  $\mathcal{B}(3, 4)$

- (2) Construct one edge in  $\mathcal{B}(d, n + 1)$  for each pair of adjacent edges in  $\mathcal{B}(d, n)$ . Here we will say that edges are adjacent if the ending node of one edge is the start node of the next edge.

The node  $(\alpha_1 \dots \alpha_{n+1})$  in  $\mathcal{B}(d, n+1)$  corresponds to the edge  $(\alpha_1 \dots \alpha_n)(\alpha_2 \dots \alpha_{n+1})$  in  $\mathcal{B}(d, n)$ ; the edges leaving  $(\alpha_1 \dots \alpha_{n+1})$  are of the form  $(\alpha_1 \dots \alpha_{n+1})(\alpha_2 \dots \alpha_{n+2})$ . These  $d$  edges correspond to the  $d$  adjacent edges  $(\alpha_1 \dots \alpha_n)(\alpha_2 \dots \alpha_{n+1})$  and  $(\alpha_2 \dots \alpha_{n+1})(\alpha_3 \dots \alpha_{n+2})$  in  $\mathcal{B}(d, n + 1)$ .

*Example.* In the doubling from  $\mathcal{B}(3, 3)$  to  $\mathcal{B}(3, 4)$  the edge  $(1222)(2220)$  in  $\mathcal{B}(3, 4)$  came from the adjacent edges  $(122)(222)$  and  $(222)(220)$  in  $\mathcal{B}(3, 3)$ .

See Figure 6 to see an example of the doubling process from  $\mathcal{B}(2, 2)$  to  $\mathcal{B}(2, 3)$ .

*Remark.* By Definition 4 and Procedure 1, the number of nodes in  $\mathcal{B}(d, n + 1)$  (and thus the number of edges in  $\mathcal{B}(d, n)$ ) ought to be  $d^{n+1}$ . The number of edges in  $\mathcal{B}(d, n)$  is the number of nodes,  $d^n$ , multiplied by the out-degree,  $d$ , resulting in  $d^{n+1}$  nodes in  $\mathcal{B}(d, n + 1)$  as desired. Similarly, the number of pairs of adjacent edges in  $\mathcal{B}(d, n)$  is the product of the number of nodes, the in-degree and the out-degree giving  $d^n \cdot d \cdot d = d^{n+2}$  which is the number of edges that are in  $\mathcal{B}(d, n + 1)$ .

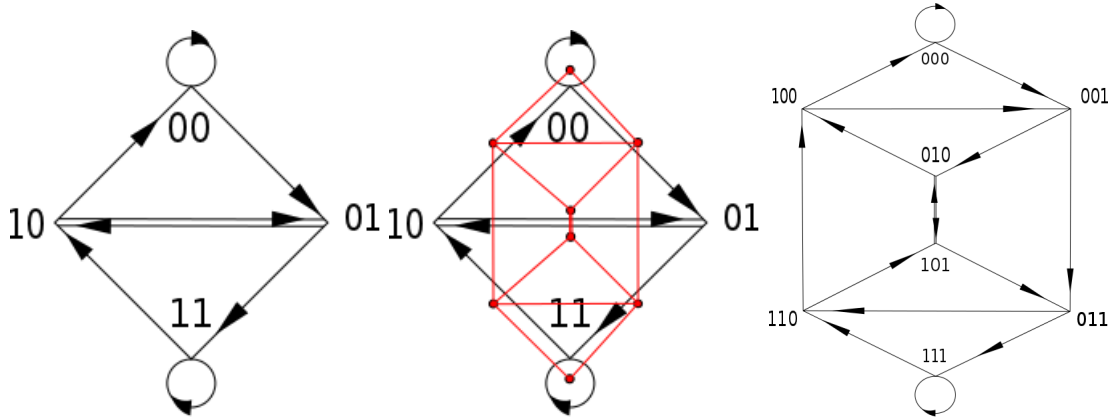


FIGURE 6. The graph  $\mathcal{B}(2, 2)$  (left) getting doubled with intermediate step (middle) to  $\mathcal{B}(2, 3)$  (right).

## 3. A COMBINATORIAL PROBLEM

De Bruijn's original paper [dB46] arose from the investigation of a problem concerning the number of unique De Bruijn sequences with respect to  $n$  on the De Bruijn graph  $\mathcal{B}(2, n)$ . De Bruijn was presented [dB46] with the conjecture (see Theorem 1 below) and in his paper he proves the conjecture using an inductive argument on a type of graph which is a more general version of the De Bruijn graph.

**Theorem 1.** *The number of De Bruijn sequences contained in  $\mathcal{B}(2, n)$  is  $2^{2^{n-1}-n}$  for all  $n \geq 1$ .*

Our goal in this section is to prove the number of De Bruijn sequences contained in  $\mathcal{B}(2, n)$  is  $2^{2^{n-1}-n}$ . However, we must first establish a few important facts.

**Definition 12.** A **T-net of order  $m$**  is a graph with  $m$  nodes and  $2m$  directed edges with the property that both the in-degree and out-degree of each node is 2.

For example, notice that  $\mathcal{B}(2, n)$  is a T-net of order  $2^n$  since it has  $2^n$  nodes and  $2^{n+1}$  edges. Let  $N$  be a T-net and denote the number of Eulerian cycles of  $N$  by  $|N|$  (If the T-net is not connected then  $|N| = 0$ ). We can now construct a unique doubled T-net by using Procedure 1. This process creates a T-net  $N^*$  of order  $2m$ .

**Theorem 2.** *If  $N$  is a T-net of order  $m$  and  $N^*$  is its doubled T-net, then*

$$|N^*| = 2^{m-1}|N|.$$

*Proof.* We first consider two important cases:

CASE 1:  $N$  is disconnected.

If  $N$  is not connected then there will be two nodes with no path between them. If there was a path in  $N^*$  between every two nodes, then this would imply that there is a path in  $N$  from every edge (these correspond to a node in  $N^*$ ) to every other edge. Thus using this "edge path" we could construct a path of nodes between every pair of nodes in  $N$ . Therefore if  $N^*$  is connected then  $N$  is connected and so the contrapositive is true as well. Thus, if  $N$  is disconnected,  $N^*$  is disconnected. This implies that if  $N$  is disconnected, then  $|N| = |N^*| = 0$  and Theorem 2 holds in this case.

CASE 2:  $N$  is connected and each node has a loop edge.

There is only one such connected T-net for each  $m$  since after the  $m$  loop edges are counted, the only way to ensure connectivity is to have the remaining edges form a cycle over all  $m$  nodes.

Looking at Figure 7 and observing that for any  $m$  we will be looking at a regular  $m$ -gon with loops at every edge, we see that for each such T-net, a path that visits each edge will never have a choice in which edge is taken, thus  $|N| = 1$  in this case.

Next, we will show that  $|N^*| = 2^{m-1}$ . For  $m > 1$  we see that since  $N$  has  $m$  edges which are self loops, then there will be  $m$  nodes in  $N^*$  which have self loops. This is a direct result from the doubling procedure. We also see that there will be  $m$  nodes in  $N^*$



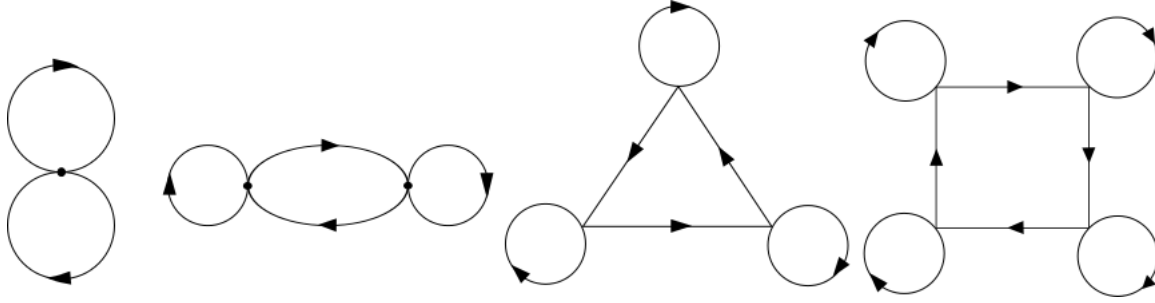


FIGURE 7. This illustrates Case 2 for  $m = 1, 2, 3$  and  $4$

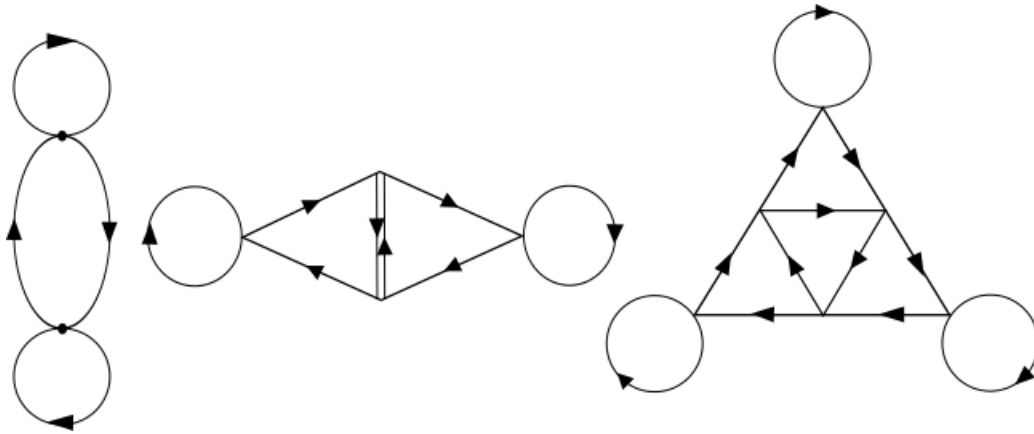


FIGURE 8. The corresponding  $N^*$  graphs for  $m = 1, m = 2$  and  $m = 3$ .

which do not have self loops, these will correspond to the edges in  $N$  which are not self looping edges. We see they will not have these self loops since these edges are not adjacent to themselves since they have different start and end nodes. We observe that any Eularian cycle on  $N^*$  will only vary on their choice of edge at the  $m$  nodes which do not have self loops. Since each of these  $m$  nodes has degree 2 then there will be  $2^m$  Eularian cycles, one for each set of edge choice at the  $m$  nodes. However we have double counted these cycles since any two cycles which take the opposite edge at these  $m$  nodes is really the same cycle written in a different order. Thus there are  $2^{m-1}$  Eularian cycles on  $N^*$ . Figure 8 shows this case for the first few  $m$ 's.

Thus Theorem 2 holds in this case as well.

We now proceed by inducting on the order of the T-net,  $m$ . When  $m = 1$  notice that  $N$  is a connected graph and each node has a loop edge, thus it falls into the second case above, so Theorem 2 holds when  $m = 1$ . We make note that since  $m = 1$ , we have that  $|N| = |N^*|$ .

Now assume inductively that for any T-net,  $N$ , of order 1 to  $m - 1$  for  $m - 1 \geq 1$  that  $|N^*| = 2^{m-1}|N|$ .

Consider a T-net,  $N$ , of order  $m$ . If  $N$  is disconnected or if each node has a loop edge, then we are done. If we are not in either of these two cases, we know that there is some node which does not have a loop edge. Call this node  $\alpha$ . Since  $N$  has in-degree and out-degree 2, let the two edges which end at  $\alpha$  be called  $ie_1$  and  $ie_2$  and let the two edges which start at  $\alpha$  be called  $oe_1$  and  $oe_2$ . We will also refer to the associated predecessors and successors of  $\alpha$  by  $A, B, C$  and  $D$  as in Figure 9.

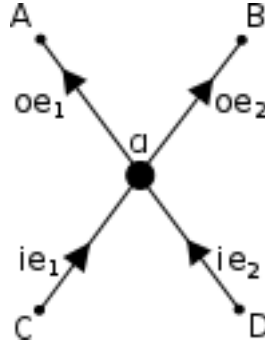


FIGURE 9. The graph of the node  $\alpha$  before removal.

We now create a new graph  $N_1$  from  $N$  by removing the node  $\alpha$  and all of the edges to and from it, but add an edge from  $C$  to  $A$  and another from  $D$  to  $B$  as in Figure 10.

Similarly, construct  $N_2$  by removing the node  $\alpha$  and all of the edges to and from it, but add an edge from  $C$  to  $B$  and another from  $D$  to  $A$ , also in Figure 10.

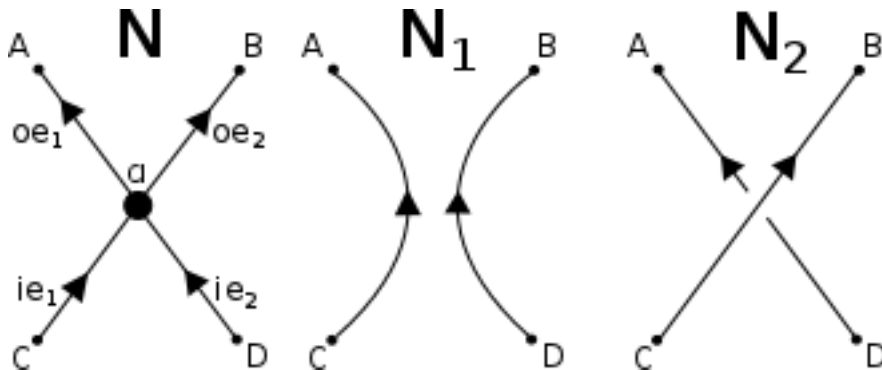


FIGURE 10. The partial graphs of  $N$ ,  $N_1$  and  $N_2$ .

Since our goal is to count Eulerian cycles on the graph, we should note that any Eulerian cycle on our T-net  $N$  corresponds to an Eulerian cycle on  $N_1$  or on  $N_2$ , but not both. This

gives us that

$$(1) \quad |N| = |N_1| + |N_2|.$$

Using Procedure 1 we double the graphs  $N_1$  and  $N_2$  to create  $N_1^*$  and  $N_2^*$  respectively. Recall our goal is to show that  $|N^*| = 2^{m-1}|N|$ . We have by induction that,

$$\begin{aligned} 2^{m-1}|N| &= 2^{m-1}(|N_1| + |N_2|) \\ &= 2 \cdot 2^{m-2}|N_1| + 2 \cdot 2^{m-2}|N_2| \\ &= 2|N_1^*| + 2|N_2^*| \end{aligned}$$

since both  $N_1$  and  $N_2$  have  $m - 1$  nodes.

All that remains is to show that,

$$(2) \quad |N^*| = 2|N_1^*| + 2|N_2^*|.$$

Procedure 1 constructs  $N^*$  from  $N$  and Figures 10 and 11 show parts of these two graphs. The four edges  $ie_1, ie_2, oe_1$  and  $oe_2$  in  $N$  are replaced by nodes  $I_1, I_2, O_1$  and  $O_2$  respectively in  $N^*$ . There are two edges entering the new nodes  $I_1$  and  $I_2$  and also two edges exiting the nodes  $O_1$  and  $O_2$  by construction. Additionally, in  $N^*$  there will be edges  $I_1O_1, I_1O_2, I_2O_1$  and  $I_2O_2$ .

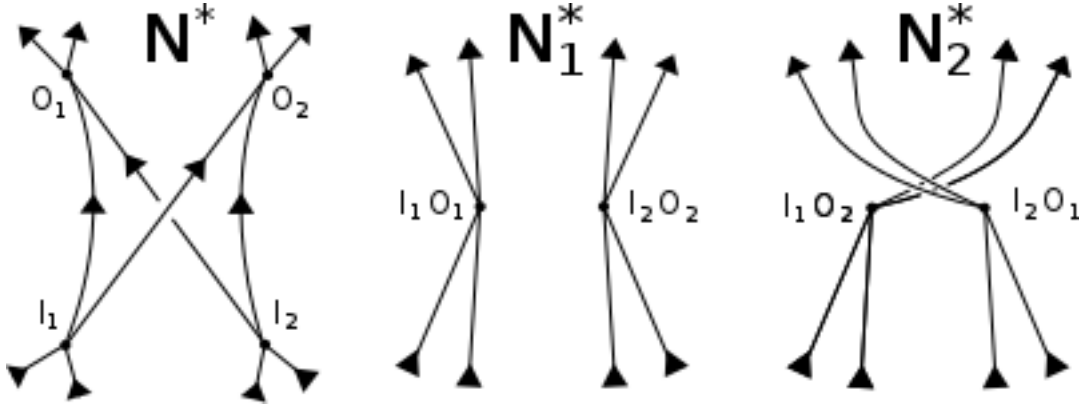


FIGURE 11. The partial graphs of  $N^*$ ,  $N_1^*$  and  $N_2^*$ .

We can construct  $N_1^*$  and  $N_2^*$  from  $N_1$  and  $N_2$  in a similar manner and once again it will be helpful to reference Figures 10 and 11. The process of combining the edges  $ie_1$  and  $oe_1$  and the edges  $ie_2$  and  $oe_2$  while creating  $N_1$  from  $N$  corresponds to merging the two nodes  $I_1$  and  $O_1$  and the two nodes  $I_2$  and  $O_2$  in  $N_1^*$ . The construction of  $N_2^*$  is similar only instead we combine the edges  $ie_1$  and  $oe_2$  and the edges  $ie_2$  and  $oe_1$  to create  $N_2$  from  $N$  and thus merge the nodes  $I_1$  and  $O_2$  and the nodes  $I_2$  and  $O_1$ . Again, the construction of these graphs can be seen in Figure 11.

Notice now that an Eulerian cycle on  $N^*$  is defined by four paths which have no common edges, each of which begins at either  $O_1$  or  $O_2$  and ends at  $I_1$  or  $I_2$ . These paths are edge disjoint because an Eulerian cycle visits each edge exactly once and these paths are all part of the same Eulerian cycle on  $N$ . We are assured that each edge except the four edges  $I_1O_1$ ,  $I_1O_2$ ,  $I_2O_1$  and  $I_2O_2$  will be included in exactly one of these paths since an Eulerian cycle traverses every edge. We will be counting the number of Eulerian cycles on  $N^*$ ,  $N_1^*$  and  $N_2^*$  which contain all four of those paths but differ in their choice of connections of the nodes  $I_1$ ,  $I_2$ ,  $O_1$  and  $O_2$ .

We will adopt N.G. de Bruijn's notation and construct three more graphs  $N^{**}$ ,  $N_1^{**}$  and  $N_2^{**}$  which represent these four paths (discussed above) by single edges instead. Notice that these new graphs are not the doubled versions of  $N^*$ ,  $N_1^*$  and  $N_2^*$ . The additional graphs here will help count the number Eulerian cycles in  $N^*$  as we proceed by induction. Also, we will now denote  $|N^*|$ ,  $|N_1^*|$ , and  $|N_2^*|$  as  $n$ ,  $n_1$  and  $n_2$ .

In order to show (2) with the new notation, we need to show

$$(3) \quad n = 2n_1 + 2n_2.$$

We consider the following cases:

- (1) Both paths in  $N^*$  starting from  $O_1$  end at different nodes.

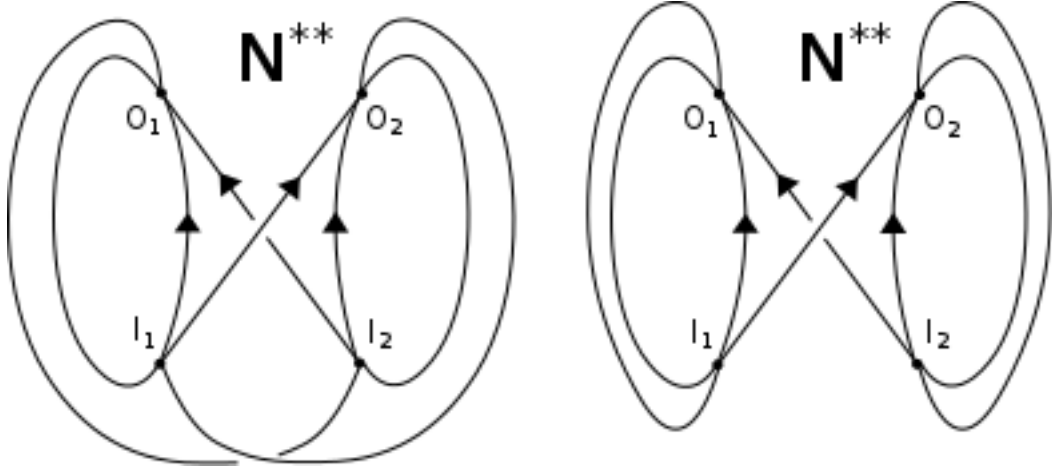


FIGURE 12. Case 1 (left) and Case 2 (right)

As is indicated by the Figure 12 we see that the four paths are: the path from  $O_1$  to  $I_1$ , from  $O_1$  to  $I_2$ , from  $O_2$  to  $I_1$  and from  $O_2$  to  $I_2$ .

Upon observing the graph  $N^{**}$  in Figure 12 we see that there are four Eulerian cycles. Examining  $N_1^{**}$  and  $N_2^{**}$  in Figure 13, which are the same graph, we see that there is exactly one Eulerian cycle. Thus, in this case we establish our claim, since

$$n = 4 = 2 \cdot 1 + 2 \cdot 1 = 2n_1 + 2n_2.$$



FIGURE 13. Case 1:  $N_1^{**}$  (left) and  $N_2^{**}$  (right)

(2) Both paths in  $N^*$  starting from  $O_1$  end at the same node.

Notice that it doesn't matter if the two paths end at  $I_1$  or  $I_2$  as the resulting graphs  $N_1^{**}$  and  $N_2^{**}$  will be the same except they will change names. Here we assume that the two paths starting at  $O_1$  end at  $I_1$ .



FIGURE 14. Case 2:  $N_1^{**}$  (left) and  $N_2^{**}$  (right)

In this case, we notice observe from Figure 14 that since  $N_1^{**}$  is disconnected, then  $n_1 = 0$ . We see number of Euler cycles in  $N_2^{**}$  is two and so again we establish the equality that  $n = 4 = 2 \cdot 0 + 2 \cdot 2 = 2n_1 + 2n_2$ . This completes our claim and we see that the equality is established.

With the truth of this claim established for any set of four paths, we see that (2) is now also clear, thus establishing Theorem 2. □

We can now establish the truth of Theorem 1 with a few observations.

- (1) A De Bruijn graph  $\mathcal{B}(2, n)$  is a T-net of order  $2^n$
- (2) There is only one Hamiltonian cycle on  $\mathcal{B}(2, 1)$ , namely [0011].
- (3) The doubling procedure on  $\mathcal{B}(2, n)$  gives  $\mathcal{B}(2, n + 1)$ .
- (4) Since an Eularian cycle visits every edge and a Hamiltonian cycle visits every node, an Eularian cycle on  $N$  represents a Hamiltonian cycle on  $N^*$ , the doubled graph of  $N$ .

*Proof.* Let  $H(N)$  denote the number of Hamiltonian cycles on  $N$ . We prove Theorem 1 by induction on  $n$ . Let  $n = 1$ . By item (2) above,

$$H(B(2, 1)) = 1 = 2^0 = 2^{2^{1-1}-1}.$$

Now to proceed inductively, let  $n \geq 1$  and assume that  $H(\mathcal{B}(2, n)) = 2^{2^{n-1}-n}$ . Then,

$$\begin{aligned}
 H(\mathcal{B}(2, n+1)) &= |\mathcal{B}(2, n)| \\
 &= |\mathcal{B}(2, n-1)^*| \\
 &= 2^{2^{n-1}-1} |\mathcal{B}(2, n-1)| \\
 &= 2^{2^{n-1}-1} H(\mathcal{B}(2, n)) \\
 &= 2^{2^{n-1}-1} 2^{2^{n-1}-n} \\
 &= 2^{2^{n-1}-1+2^{n-1}-n} \\
 &= 2^{2^n-n-1} \\
 &= 2^{2^{(n+1)-1}-(n+1)}
 \end{aligned}$$

Thus our induction holds and the proof of Theorem 1 is complete.  $\square$

#### 4. FAULT TOLERANCE

This section is motivated by applications of De Bruijn graphs, and while we will be examining the mathematical properties relevant to these applications, it is worthwhile to understand some motivation. De Bruijn graphs have a few properties which make them natural choices for the structure of a communication network. Esfahanian and Hakimi [EH85] describe applications of De Bruijn graphs to multiprocessor systems and Collins [CDMP92] describes how the De Bruijn graph gives the structure for a fully parallel Viterbi decoder. De Bruijn graphs have been natural choices for these types of network topologies for a few important reasons as follows.

We mentioned in the introduction the first two following properties and now with our focus on fault tolerance we find this third property to be valuable as well. For a De Bruijn graph with  $d^n$  nodes there is a

- (1) small in-degree and out-degree, which means fewer connections to each node (processor)
- (2) small distance between nodes (between any two nodes there does not need to exist more than  $n$  edges)
- (3) large number of paths between nodes.

These properties are the motivation for investigating applications of De Bruijn graphs, and furthermore, since we are interested in the graphs for communication network applications the following questions will be important to study.

- (1) What sort of *connectivity* can we expect to see if a given node is removed?

- (2) What sort of *connectivity* can we expect to see if an edge is removed?
- (3) How many edges can be removed while still maintaining communication paths?

These are the main questions that inspire our mathematical investigation of De Bruijn graphs. As we study the application of De Bruijn graphs we will make these questions more precise.

**4.1. Node failure.** We will first investigate the types of graph structures that result from node failures. A node failure is represented by a De Bruijn graph which has that node removed as well as all of the edges which begin or end at that particular node.

To investigate these structures, we will prove a few lemmas.

**Lemma 1.** *For any node,  $\alpha$  of a De Bruijn graph  $\mathcal{B}(d, n)$ , all of the predecessors of  $\alpha$  have the same set of successors.*

*Proof.* Let  $\alpha$  be the node  $(\alpha_1 \dots \alpha_n)$ , then any predecessor of  $\alpha$  looks like  $(x \alpha_1 \dots \alpha_{n-1})$  for  $x$  in our alphabet, and then the successor of any node of this form is  $(\alpha_1 \dots \alpha_{n-1} y)$ , with  $y$  in our alphabet.  $\square$

*Example.* In  $\mathcal{B}(3, 3)$  if we look at the node  $(201)$  then the set of predecessors will be the set  $\{(020), (120), (220)\}$  and the set of successors for each of those nodes is the same, they are  $\{(200), (201), (202)\}$ .

**Lemma 2.** *Let  $\alpha$  be a node in  $\mathcal{B}(d, n)$  of the form  $(\alpha \dots \alpha)$ , then if a node  $\beta$  is neither a successor nor a predecessor of  $\alpha$ , then  $\alpha$  and  $\beta$  have no common successors and they have no common predecessors (however, a node which is a successor of  $\alpha$  could be a predecessor of  $\beta$  and vice versa).*

*Proof.* Any successor of  $\alpha$  is of the form  $(\alpha \dots \alpha x)$  where  $x$  is a character in our alphabet. If  $\alpha$  and  $\beta$  share the successor  $(\alpha \dots \alpha x)$  then  $\beta$  is a predecessor of this node. Any predecessor of  $(\alpha \dots \alpha x)$  is of the form  $(y \alpha \dots \alpha)$  for  $y$  also a character in our alphabet. Since  $\beta$  is a predecessor of  $(\alpha \dots \alpha x)$ , then  $\beta = (y \alpha \dots \alpha)$  for some choice of  $y$ . However, we see by construction of  $\mathcal{B}(d, n)$  that  $\beta = (y \alpha \dots \alpha)$  is a predecessor of  $\alpha$ . Thus if  $\alpha$  and  $\beta$  share a successor, then  $\beta$  is a predecessor of  $\alpha$ .

The proof that if  $\alpha$  and  $\beta$  share a predecessor then  $\beta$  is a successor of  $\alpha$  is similar.  $\square$

**Observation 1.** Most of the nodes of a De Bruijn graph have disjoint sets of successors and predecessors. The exception to this is if for a given node  $(\alpha_1 \dots \alpha_n)$  it is the case that for some choice of  $x$  and  $y$ ,  $(x \alpha_1 \alpha_2 \dots \alpha_{n-1}) = (\alpha_2 \dots \alpha_n y)$ . There are three cases where this occurs:

- (1) When we examine the node  $(\alpha \dots \alpha)$ .  
Ex)  $(111)$  in  $\mathcal{B}(3, 3)$  has itself as both a predecessor and a successor.
- (2) When  $n$  is even, then when we examine the node  $(\alpha\beta\alpha\beta \dots \alpha\beta)$ .  
Ex)  $(2020)$  in  $\mathcal{B}(3, 4)$  has the node  $(0202)$  as both a predecessor and a successor.

(3) When  $n$  is odd, then when we examine the node  $(\alpha\beta\alpha\beta\dots\alpha)$ .

Ex) (404) in  $\mathcal{B}(5,3)$  has the node (040) as both a predecessor and a successor.

**Definition 13.** **Faulty nodes** in a graph will be those which are removed from the graph. We will denote the set of faulty nodes in a graph to be the set  $F$ . We will refer to the De Bruijn graph  $\mathcal{B}(d,n)$  with these nodes removed as  $\mathcal{B}(d,n)_F$ .

*Remark.* When removing  $F$  from a De Bruijn graph, we will find that this removal of the individual nodes might result in a graph on which there exists no Hamiltonian cycles. We see that Figure 15 is an example of this since after removing the points (001), (010) and (100) we no longer even have a connected graph.

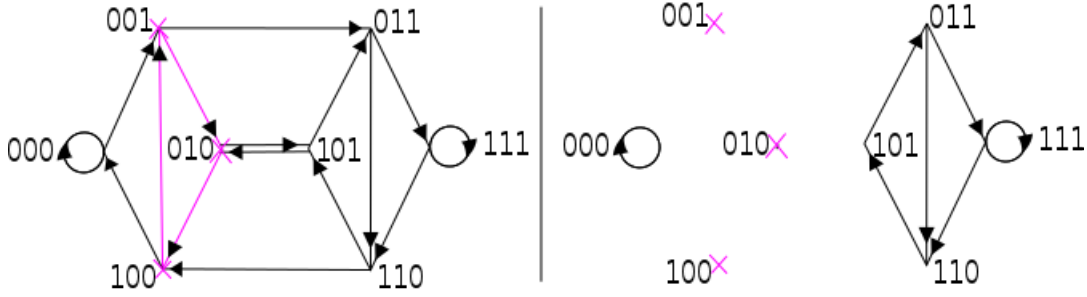


FIGURE 15.  $\mathcal{B}(2,3)$  (left) with the removal of the points (100), (010) and (001) (right).

It is with this motivation that we introduce this next definition and will later examine a procedure that constructs a cycle on  $\mathcal{B}(d,n)_F$  for a given  $F$ , which covers the maximum number of nodes.

**Definition 14.** A **necklace** of a node, denoted  $Neck((\alpha_1 \dots \alpha_n))$ , is the set of nodes which are in the cycle  $[\alpha_1 \dots \alpha_{n+n}]$

*Example.* If we examine  $\mathcal{B}(3,5)$  then

$$Neck(21102) = \{21102, 11022, 10221, 02211, 22110\}.$$

We notice that this is a set of  $n$  nodes, but if we examine instead the necklace  $Neck(2121)$  in  $\mathcal{B}(3,4)$  we see that this set contains only the nodes (2121) and (1212).

**Definition 15.** The **node connectivity** of a graph is defined to be the minimum number of nodes whose removal results in a disconnected graph.

*Example.* We notice from Figure 15 that the node connectivity of  $\mathcal{B}(2,3)$  is at most 3, but upon examination we see that we still have a disconnected graph if only (100) and (001) are removed. This implies that the node connectivity of  $\mathcal{B}(2,3)$  is 2 since there is no node whose removal results in a disconnected graph.

With the definition of node connectivity and the properties of De Bruijn graphs explored, we are now equipped to answer the question, “What kind of node connectivity can we expect to see on an undirected De Bruijn graph?”



**Theorem 3** (Thm 2 in [EH85]). *The node connectivity of an undirected De Bruijn graph,  $\mathcal{UB}(d, n)$ , is  $2d - 2$ . Furthermore, for any arbitrary set of node failures  $F$  with  $|F| \leq 2d - 3$ , the maximum length of the shortest path between any two nodes is less than or equal to  $2n$ .*

A proof of this theorem can be found in [EH85], but we omit it here for brevity, and instead look at some examples which actualize some of these bounds.

*Remark 1.* For most nodes of  $\mathcal{B}(d, n)$  we see that the set of successors and the set of predecessors are disjoint (see Observation 1) but when we look at the node  $(\alpha \dots \alpha)$  we see that it has  $d$  successors and  $d$  predecessors. However, since we are examining the undirected graph we see that a node is not considered adjacent to itself, so the node  $(\alpha \dots \alpha)$  has  $2d - 2$  adjacent nodes. If we remove all of them, then we are clearly left with a disconnected graph as there are no nodes remaining which are adjacent to  $(\alpha \dots \alpha)$ , thus exhibiting the lower bound,  $2d - 2$ .

*Remark 2.* The actualization of the second bound presented on the maximum length of the shortest path between nodes is a bit harder construct. The difficulty in construction is explained when Esfahanian and Hakima [EH85] present a slightly better bound of  $\min(2n, n + 5 + \log n)$  which has an extensive proof that will not be presented in this paper.

We will now shift our focus back to directed De Bruijn graphs and the approaches taken to find bounds on node connectivity.

**Lemma 3.** *Let  $x$  be a node in  $\mathcal{B}(2, n)$ , then  $\mathcal{B}(2, n)$ - $Neck(x)$  has a component with at least  $2^n - n - 1$  nodes.*

Before beginning the proof of this lemma, it is helpful to examine the following properties of necklaces and binary De Bruijn graphs (i.e. those of the form  $\mathcal{B}(2, n)$ ).

**Definition 16.** Let  $x$  be a node in  $\mathcal{B}(2, n)$ , then the **weight** of  $x$ , denoted  $wt(x)$ , is the number of 1's in  $x$ .

With this in mind, we notice the following observations on necklaces and weights.

**Observation 2.** For every node  $y \in Neck(x)$  we see that  $wt(y) = wt(x)$ .

**Observation 3.** There is only one necklace of weight 0 and  $n$  as

$$Neck(0 \dots 0) = \{(0 \dots 0)\}$$

and similarly

$$Neck(1 \dots 1) = \{(1 \dots 1)\}.$$

**Observation 4.** There is only one necklace of weight 1 and  $n-1$  as  $Neck((1 \ 0 \dots 0))$  is the set of nodes which contain a single 1 and  $Neck((0 \ 1 \dots 1))$  is the set of nodes which contain a single 0.

**Observation 5.** There can be multiple necklaces of weights between 1 and  $n-1$ .

*Example.* In  $\mathcal{B}(2, 4)$  we see that  $Neck((1100))$  is not the same as  $Neck((1010))$

With these facts in mind, we can proceed with the proof of Lemma 3.

*Proof.* Let  $x = (x_1 \dots x_n)$  be a faulty node in  $\mathcal{B}(2, n)$  with  $wt(x) = k$ . We observe that all of the nodes of weight greater than  $k$  are connected since if we have two such nodes  $y$  and  $z$  with  $wt(y), wt(z) > k$  then there is a path from  $y$  to  $z$  which avoids  $Neck(x)$ . One such path is the path:

$$(y_1 \dots y_n)(y_2 \dots y_n 1)(y_3 \dots y_n 11) \dots (1 \dots 1)(1 \dots 1 z_1)(1 \dots 1 z_1 z_2) \dots (1 z_1 \dots z_{n-1})(z_1 \dots z_n).$$

We see that every node in that path has weight greater than  $k$  since the weight is non-decreasing while we add 1's to the end and then non-increasing when we replace them with the  $z_i$ 's until we get to  $z$  which has weight greater than  $k$ . Similarly, there is a path between any two nodes which have weight less than  $k$  using the same method with 0's instead.

We will now prove Lemma 3 in cases on the weight of the faulty node.

- Case  $wt(x) = 0$  (or  $n$ ): There is only one node of weight 0 and therefore all of the nodes have weight greater than this faulty node. We therefore see that  $\mathcal{B}(2, n) - Neck(x)$  is entirely connected, and thus there is a connected component of size  $2^n - 1$ .
- Case  $wt(x) = 1$  (or  $n - 1$ ): We see that  $\mathcal{B}(2, n) - Neck(x)$  has no path from  $(0 \dots 0)$  to any other node but that each other node is connected to any other node since again all of the remaining nodes have weight greater than 1. Thus the largest component has size

$$2^n - |Neck(10 \dots 0)| - 1 = 2^n - n - 1$$

- Case  $1 < wt(x) < n - 1$ : We see that this case only arises when  $n \geq 4$  since for  $n < 4$  each weight is either in case 1 or case 2 above. By construction we can see there are at least two necklaces of weight  $k$ , for example  $Neck((0 \dots 01 \dots 1))$  and  $Neck((0 \dots 0101 \dots 1))$ . Now let  $y$  with  $wt(y) = k$  and  $y$  not in  $Neck(x)$ . Then there is a path from any node with weight less than  $k$  to  $y$  and there is a path from any node of weight greater than  $k$  to  $y$  using the same pathing method as above. We see that both these paths avoid  $Neck(x)$  since our path from lesser weight has non-decreasing weight and once it has reached a node of weight  $k$  it will be a node in  $Neck(y)$ . Thus  $\mathcal{B}(2, n) - Neck(x)$  is connected and then,

$$2^n - |Neck(x)| = 2^n - n.$$

This completes our cases and we see that for all  $x \in \mathcal{B}(2, n)$ , the largest connected component of  $\mathcal{B}(2, n) - Neck(x)$  has at least  $2^n - n - 1$  nodes.  $\square$

**Theorem 4.** *The De Bruijn graph  $\mathcal{B}(2, n)$  with a single node failure has a cycle with at least  $2^n - n - 1$  nodes.*

*Remark.* Notice that this statement is very similar to Lemma 3 but it says something stronger. We claim instead that not only is there a connected component of size  $2^n - n - 1$  but that this component actually contains a cycle.

We will not discuss this proof in detail, but the method is to apply Lemma 3 and then to apply the following algorithm to construct the cycle. The proof of the correctness of this algorithm is in [RB93b].

Let  $x$  be a faulty node and let  $F = Neck(x)$ . The following algorithm constructs the largest cycle in the component of  $\mathcal{B}(2, n)_F$  which contains the node  $z$ . This node is a parameter of the algorithm and so it is important that we choose  $z$  to be in the largest component before starting the algorithm. This can be done by choosing  $wt(z) \neq 0, n$ .

---

**Algorithm 1** Constructs a Hamiltonian cycle in the component of  $\mathcal{B}(2, n)_F$  which contains the node  $z$

---

```

 $C_z := (z)$ 
mark  $z$ 
 $x := z$ 
while there is an edge from  $x$  to an unmarked successor  $y$  do
    insert  $y$  after  $x$  in  $C_z$ 
    mark  $y$ 
     $x := y$ 
end while
for each node  $x$  in  $C_z$  with an unmarked successor  $y$  do
     $C_y := Cycle(y)$ 
     $C_z := C_z \text{ JOIN } C_y$ 
end for
return  $C_z$ 

```

---

*Example.* In order to understand this algorithm a little better, we will go through an example of it running on  $\mathcal{B}(2, 4)$  with faulty node (1100). First, we will remove the necklace of the faulty node (1100). This set is

$$F = Neck((1100)) = \{(1100), (1001), (0011), (0110)\}.$$

We know from the proof of Lemma 3 that for this particular  $F$ ,  $\mathcal{B}(2, 4)_F$  is connected and so we can choose any node  $z$  at which to start the algorithm.

We start at the node  $z = (1101)$ , and then build  $C_z$  by successively appending nodes to the end of  $C_z$  always choosing from among the successors the first node lexicographically which is unmarked and also not in  $F$ . We will continue in this manner until we are at a point in which there is no successor which is both unmarked and not in  $F$ .

The successors of (1101) are (1010) and (1011). At this point only (1101) is marked and neither of these successors are in  $F$ , thus our first step will be to add the node (1010) to  $C_z$  as it is the first, lexicographically, among the successors of (1101). We notice here that at this point, choosing the successor will generally mean choosing the node which adds a 0 to the end because of the lexicographical ordering. This gives,

$$C_z = (1101)(1010).$$

After this we continue adding nodes to  $C_z$  in this fashion so that we obtain

$$\begin{aligned} C_z &= (1101)(1010)(0100) \\ &= (1101)(1010)(0100)(1000) \\ &= (1101)(1010)(0100)(1000)(0000). \end{aligned}$$

We notice here that the successors of (0000) are (0000) and (0001). The node (0000) has already been added to  $C_z$  and so it is considered marked and we must instead choose to add the node (0001). This gives us

$$C_z = (1101)(1010)(0100)(1000)(0000)(0001)$$

and then we continue adding nodes again to obtain

$$\begin{aligned} C_z &= (1101)(1010)(0100)(1000)(0000)(0001)(0010) \\ &= (1101)(1010)(0100)(1000)(0000)(0001)(0010)(0101) \\ &= (1101)(1010)(0100)(1000)(0000)(0001)(0010)(0101)(1011). \end{aligned}$$

It is at this point that since the successors of (1011) are (0110) and (0111), we choose to add (0111) since (0110) is in the removed set  $F$ . We now have

$$C_z = (1101)(1010)(0100)(1000)(0000)(0001)(0010)(0101)(1011)(0111).$$

Continuing this process, we add (1110) and then see that with the addition of (1110) that of the two successors, (1101) is marked and (1100) is in  $F$ . Thus we reach the end of the “while” loop with

$$C_z = (1101)(1010)(0100)(1000)(0000)(0001)(0010)(0101)(1011)(0111)(1110).$$

We now enter into the “for” loop and try to find the first node which has an unmarked successor not in  $F$ . The first few nodes do not have this property, but we find that the node (0111) has successor (1111) which is unmarked and not in  $F$ .

We now use Algorithm 1 on the node (1111) to construct  $C_y$ . We will set  $y = (1111)$  and then construct  $C_y$  while maintaining the same list of marked nodes as before. Since the successors of (1111) are (1111) which is now marked and (1110) which is also marked we see that  $C_y = (1111)$  after the “while loop” completes. Now, after completing the construction of  $C_y$ , we then go back to our  $C_z$  and join it with  $C_y$  by inserting  $C_y$  after (0111) in  $C_z$ . This gives us the cycle

$$C_z = (1101)(1010)(0100)(1000)(0000)(0001)(0010)(0101)(1011)(0111)(1111)(1110).$$

This  $C_z$  is a cycle on the 12 remaining nodes in  $\mathcal{B}(2, 4)$  and is therefore a Hamiltonian cycle on the component containing  $z = (1101)$ .

*Remark.* It may appear to be coincidence that this algorithm returns a cycle instead of a path, but if we recall that each predecessor of a node has all same successors, then since each of the predecessors of our start node have been visited, then so, too, must all the successors of those predecessors. This means that since the first node in  $C_z$  is the only one which has not had an in-going edge and the last node is the only one without an out-going edge, then this last node must be a predecessor of the original node.

**4.2. Edge Failure.** We will now turn our attention away from node failure to the topic of edge failure. Notice that it is possible to remove an edge from a De Bruijn graph and still have a connected graph (for example remove one of the loop edges) so we will be expecting the bounds on edge failure to be better in the sense that we can remove more edges and still have connectedness and the existence of Hamiltonian cycles. In this section, we will explore edge failure and the connectivity that we can expect under these failures (i.e. questions 2 and 3 from the beginning of the Fault Tolerance section).

In order to clarify the expression of cycles in this section it will be useful to modify our notation; from here on the cycle  $C = [\alpha_0 \dots \alpha_{k-1}]$  will be expressed as

$$C = [\alpha_0, \alpha_1, \dots, \alpha_{k-1}]$$

using commas instead of juxtaposition. Also, our subscripts will be considered modulo  $k$  where  $k$  is the length of the cycle. We will choose  $d$ , the size of our alphabet, to either be a prime, or a prime power and thus our alphabet will now be  $\mathbb{F}_d$  and so we have that the characters of the alphabet are the elements of a field. This will allow us to perform some calculations on nodes and cycles which will be very useful in proving some of the theorems in this section.

One of the first calculations we will be doing is a character permutation of a cycle. If  $C = [c_0, c_1, \dots, c_{k-1}]$ , then we will say

$$C_\alpha = [c_0 + \alpha, c_1 + \alpha, \dots, c_{k-1} + \alpha].$$

Additionally, we will adopt the notation of Rowley and Bose [RB93b] and define the cycle

$$\psi(C) = [c_0 - c_1, c_1 - c_2, \dots, c_{k-1} - c_0].$$

**Definition 17.** Two cycles are **edge disjoint** if the set of edges traversed by the first cycle is disjoint from the set of edges traversed by the second.

**Definition 18.** A cycle  $C$  is **edge primitive** if the cycles  $\{C_\alpha \mid 0 \leq \alpha \leq d-1\}$  are pairwise edge disjoint.

**Lemma 4.** A cycle  $C$  in  $\mathcal{B}(d, n)$  is edge primitive if and only if  $\psi(C)$  is a cycle in  $\mathcal{B}(d, n)$  with no repeated nodes.

*Proof.* We will just prove the first direction ( $\Rightarrow$ ) here. The proof of the opposite direction uses a similar method (see [RB91]).

Assume  $C = [c_0, c_1, \dots, c_{k-1}]$  is edge primitive. By definition  $\psi(C)$  is a cycle and so we will show that no node can occur twice in  $\psi(C)$ .

Suppose that

$$(c_i - c_{i+1}, \dots, c_{i+n-1} - c_{i+n}) = (c_j - c_{j+1}, \dots, c_{j+n-1} - c_{j+n}).$$

We see that if we let  $\delta = c_j - c_i$  then,

$$\begin{aligned}
c_i - c_{i+1} = c_j - c_{j+1} &\rightarrow c_{i+1} = c_{j+1} - (c_j - c_i) = c_{j+1} - \delta \\
c_{i+1} - c_{i+2} = c_{j+1} - c_{j+2} &\rightarrow c_{i+2} = c_{j+2} - (c_{j+1} - c_{i+1}) = c_{j+2} - \delta \\
&\vdots \\
c_{i+n-1} - c_{i+n} = c_{j+n-1} - c_{j+n} &\rightarrow c_{i+n} = c_{j+n} - \delta
\end{aligned}$$

This gives us that the cycles  $C$  and  $C_\delta$  have the edge  $(c_i c_{i+1} \cdots c_{i+n-1})(c_{i+1} \cdots c_{i+n})$  in common and thus it must be the case that  $\delta = 0$ . This implies that

$$\begin{aligned}
c_i &= c_j \\
c_{i+1} &= c_{j+1} \\
&\vdots \\
c_{i+n} &= c_{j+n}
\end{aligned}$$

and therefore we see that  $\psi(C)$  can have no repeated nodes. □

*Example.* Consider the cycle  $C = [2, 1, 1, 0, 1]$  in  $\mathcal{B}(3, 2)$ . We generate

$$\psi(C) = [2 - 1, 1 - 1, 1 - 0, 0 - 1, 1 - 2] = [1, 0, 1, 2, 2]$$

We see that  $\psi(C)$  repeats no nodes and thus we know that the set of cycles  $\{C_\alpha | 0 \leq \alpha \leq d - 1\}$  are pairwise edge disjoint. We can list these cycles,

$$\begin{aligned}
C_0 &= [2, 1, 1, 0, 1] \\
C_1 &= [0, 2, 2, 1, 2] \\
C_2 &= [1, 0, 0, 2, 0]
\end{aligned}$$

and verify that no edge (3-tuple) is used more than once. Thus we see that  $C$  is edge primitive.

*Example.* We can instead consider the cycle  $C = [1, 2, 0, 0, 1]$  in  $\mathcal{B}(3, 2)$ . First, we generate

$$\psi(C) = [1 - 2, 2 - 0, 0 - 0, 0 - 1, 1 - 1] = [2, 2, 0, 2, 0]$$

Notice that the node (20) shows up two times and therefore by Lemma 4 the set of cycles  $\{C_\alpha | 0 \leq \alpha \leq d - 1\}$  are not pairwise edge disjoint. Here we will list these cycles,

$$\begin{aligned}
C_0 &= [1, 2, 0, 0, 1] \\
C_1 &= [2, 0, 1, 1, 2] \\
C_2 &= [0, 1, 2, 2, 0]
\end{aligned}$$

We see that the edge (11)(12) shows up in both  $C_0$  and in  $C_1$  and thus verifies that  $C$  is not edge primitive.

The next lemma comes from examining the relationship between cycles in  $\mathcal{B}(d, n)$  and linear recurrence relations over  $\mathbb{F}_d$ .

If we have a sequence  $S$  defined by the recurrence relation,

$$(4) \quad s_{n+i} = a_{n-1}s_{n-1+i} + \cdots + a_0s_i \text{ for } i \geq 0$$

with  $a_i \in \mathbb{F}_d$  and  $a_0 \neq 0$ , then the **period** of  $S$  is the smallest  $k > 0$  such that  $s_i = s_{i+k}$  for  $i \geq 0$ . We see then that a sequence with period  $k$  corresponds to a cycle of length  $k$  in the De Bruijn graph  $\mathcal{B}(d, n)$ , specifically the cycle  $[s_0, s_1, \dots, s_{k-1}]$ .

We will call the polynomial

$$(5) \quad p(x) = x^n - a_{n-1}x^{n-1} - \cdots - a_1x - a_0$$

the **characteristic polynomial** of  $S$ .

**Definition 19.** If  $p(x)$  is a polynomial over  $\mathbb{F}_d$  then the **order** of  $p(x)$  is the least positive integer  $k$  such that  $p(x)$  divides  $1 - x^k$ . An irreducible polynomial of degree  $n$  with coefficients in  $\mathbb{F}_d$  is **primitive** if its order is  $d^n - 1$ .

*Remark.* If  $p(x)$  is irreducible then from [RB91] the period of  $S$  is exactly the order of  $p(x)$ . From [HP03] we know that there is a primitive polynomial for every choice of  $n$  and  $d$  (a prime power).

**Definition 20.** We will say that  $C = [c_0, c_1, \dots, c_{k-1}]$  is a **maximal cycle** if it is a cycle with length  $k = d^n - 1$  and it satisfies a recurrence relation whose characteristic polynomial is primitive.

**Lemma 5.** *If  $C$  is a maximal cycle, then  $C$  is edge primitive.*

*Proof.* Let  $C$  be a maximal cycle which is not edge primitive. Since it is not edge primitive, there exist two cycles,  $C_\alpha$  and  $C_\beta$  which share an edge. This means that the node  $x = (x_0, x_1, \dots, x_{n-1})$  is followed by the node  $(x_1, \dots, x_{n-1}, x_n)$  in both  $C_\alpha$  and  $C_\beta$ . Then since  $C$  is generated by the recurrence relation (4) we see that in  $C_\alpha$  this implies,

$$\begin{aligned} x_n &= a_{n-1}(x_{n-1} - \alpha) + \cdots + a_0(x_0 - \alpha) + \alpha \\ &= a_{n-1}x_{n-1} + \cdots + a_0x_0 + (-a_{n-1}\alpha - \cdots - a_0\alpha + \alpha)d \end{aligned}$$

which implies

$$\begin{aligned} x_n - a_{n-1}x_{n-1} - \cdots - a_0x_0 &= -a_{n-1}\alpha - \cdots - a_0\alpha + \alpha \\ &= \alpha(1 - (a_{n-1} + \cdots + a_0)). \end{aligned}$$

Likewise, in  $C_\beta$

$$\begin{aligned} x_n &= a_{n-1}(x_{n-1} - \beta) + \cdots + a_0(x_0 - \beta) + \beta \\ &= a_{n-1}x_{n-1} + \cdots + a_0x_0 + (-a_{n-1}\beta - \cdots - a_0\beta + \beta). \end{aligned}$$

and so

$$\begin{aligned} x_n - a_{n-1}x_{n-1} - \cdots - a_0x_0 &= -a_{n-1}\beta - \cdots - a_0\beta + \beta \\ &= \beta(1 - (a_{n-1} + \cdots + a_0)). \end{aligned}$$

Now from  $C_\alpha$  and  $C_\beta$  we get that

$$\alpha(1 - (a_{n-1} + \cdots + a_0)) = \beta(1 - (a_{n-1} + \cdots + a_0))$$

and since  $\alpha \neq \beta$  then we see

$$1 - (a_{n-1} + \cdots + a_0) = 0$$

and then from (5) we see that

$$p(1) = 1 - (a_{n-1} + \cdots + a_0) = 0$$

and thus  $(1 - x)$  divides  $p(x)$  which contradicts the fact that  $p(x)$  is irreducible. Thus  $C_\alpha$  and  $C_\beta$  have no common edges and therefore  $C$  is edge primitive.  $\square$

This gives us some cool hardware, so let's look at an example of this. We would like to construct a maximal cycle in  $\mathcal{B}(3, 3)$ . We know that  $p(x) = x^3 + 2x^2 + 1$  is primitive over  $\mathbb{F}_d$  and so the linear recurrence

$$(6) \quad \begin{aligned} x_{i+3} &= -2x_{i+2} - x_i \\ &= x_{i+2} + 2x_i \end{aligned}$$

with seeds  $x_0 = x_1 = 0$  and  $x_2 = 1$  should generate our cycle. We continue

$$\begin{aligned} x_3 &= x_2 + 2x_0 \\ &= 1 + 2 \cdot 0 \\ &= 1 \\ x_4 &= x_3 + 2x_1 \\ &= 1 + 2 \cdot 0 \\ &= 1 \\ x_5 &= x_4 + 2x_2 \\ &= 1 + 2 \cdot 1 \\ &= 0 \\ &\dots = \dots \end{aligned}$$

and so on, continuing to use our recurrence relation to get the next values. This generates the cycle

$$C_0 = [00111021121010022201221202].$$

Then we can construct  $C_1$  and  $C_2$  by adding 1 and 2 to each value, respectively, giving

$$\begin{aligned} C_1 &= [11222102202121100012002010] \\ C_2 &= [22000210010202211120110121]. \end{aligned}$$

We see by inspection that  $C_0, C_1$  and  $C_2$  are edge disjoint.



**Theorem 5.** *The De Bruijn graph  $\mathcal{B}(d, n)$  with at most  $d - 1$  faulty edges admits a cycle of length  $d^n - 1$  when  $d$  is a prime power.*

*Proof.* We know that every maximal cycle is a cycle of length  $d^n - 1$  and also that there are  $d$  edge disjoint cycles that we can generate in this manner. This means that these cycles cover

$$d(d^n - 1) = d^{n+1} - d$$

edges. We also notice that none of them use the loop edges since then they would visit a node twice.

Additionally, we know from Procedure 1 in Section 2 that there are  $d^{n+1}$  total edges and  $d$  loop edges, thus these maximal cycles partition all the  $d^{n+1} - d$  edges which are not loop edges. This means that since we have  $d$  edge disjoint cycles, that removing  $d - 1$  edges from the whole graph cannot break all of these cycles and therefore when  $d$  is a prime power we see that  $\mathcal{B}(d, n)$  still admits at least one of these cycles. □

Rowley and Bose expand this result to provide a lower bound for De Bruijn graphs which do not have alphabets with prime power sizes in [RB93a]. They prove by similar means in the following Lemmas and Theorem.

**Lemma 6.** *The graph  $\mathcal{B}(d, n)$  admits  $d - 1$  disjoint Hamiltonian cycles when  $d$  is a power of 2.*

**Lemma 7.** *The graph  $\mathcal{B}(d, n)$  admits  $\frac{d-1}{2}$  disjoint Hamiltonian cycles when  $d$  is an odd prime power.*

We notice that these differ from our previous result in that they are full Hamiltonian cycles and not just cycles of length  $d^n - 1$  as before. It is this small difference which reduces the disjoint cycle count.

**Lemma 8.** *If  $\mathcal{B}(d, n)$  admits  $k$  disjoint Hamiltonian cycles,  $\mathcal{B}(d', n)$  admits  $k'$  Hamiltonian cycles and  $d$  and  $d'$  are relatively prime, then  $\mathcal{B}(dd', n)$  admits at least  $kk'$  Hamiltonian cycles.*

**Theorem 6.** *The number of disjoint Hamiltonian cycles in  $\mathcal{B}(d, n)$  is at least*

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

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

*Proof.* The proof of this theorem follows directly from Lemmas 6, 7 and 8. □

## 5. DE BRUIJN MODIFICATIONS

We have now seen a variety of modifications to De Bruijn graphs. One particularly simple modification described in Section 2 was the undirected version of De Bruijn graphs. In Section 3 we saw a generalized version of De Bruijn graphs called a T-net, and in Section 4 we examined modifications which result from removing a set of nodes from  $\mathcal{B}(2, n)$ . In this section we will consider a few more variants.

**5.1. The  $\mathcal{Q}_n$  Cycle.** This modification is understood more naturally through the idea of De Bruijn sequences, rather than through De Bruijn graphs. At the end of N.G. de Bruijn's paper [dB46], he presents a problem analogous to that of counting De Bruijn sequences as discussed in Section 3. He introduces the following sequence which has some unique properties.

**Definition 21.** A  $n$ -tuple is said to be **admissible** if no two consecutive digits are the same. A  $\mathcal{Q}_n$  **cycle** is a string with alphabet  $\{0,1,2\}$  in which every admissible  $n$ -tuple occurs exactly once in the string.

*Example.* We see that  $[0,1,2,0,1,0,2,0,2,1,2,1]$  is a sequence where each admissible 3-tuple occurs exactly once and thus it fits the criteria to be a  $\mathcal{Q}_3$  cycle.

We see that like De Bruijn sequences,  $\mathcal{Q}_n$  cycles require each valid  $n$ -tuple to occur. However, unlike the De Bruijn sequences presented in [dB46], these revised sequences have a different validity requirement and they also use the alphabet  $\{0,1,2\}$  instead of  $\{0,1\}$ . These cycles do still have a closed formula; In [dB46], De Bruijn gives that the number of  $\mathcal{Q}_n$  cycles is

$$3 \cdot 2^{3 \cdot 2^{n-2} - n - 1}.$$

This proof here is analogous to the proof presented in Section 3 of Theorem 1.

**5.2. Rowley and Bose Modifications.** Rowley and Bose cover two additional types of graphs in their study of fault tolerance. The first is discussed in [RB91] and it is created by removing some of the edges and adding them back in other places in order to allow the graph to have  $d$  Hamiltonian cycles instead of the  $d - 1$  in standard De Bruijn graphs. The second graph is discussed in [RB93b], and is an undirected version of the first.

Rowley and Bose give a lengthy description of the procedure for the first modification in [RB91] that will not be presented here. However we will briefly outline the process. In the procedure, every self loop is removed and also exactly one edge in every pair of edges of the form  $(abab \dots)(baba \dots)$  for  $a \neq b$ . Then, in order to preserve the in-degree and out-degree of the nodes, some edges of the form  $(aa \dots)(bb \dots)$  and also  $(abab \dots)(cc \dots)$  are added.

The new graph will maintain the in-degree and out-degree of the original De Bruijn graph  $\mathcal{B}(d, n)$  and more importantly will allow us to construct  $d$  Hamiltonian cycles on each graph instead of merely  $d - 1$ . This modification is motivated by the practical application of De Bruijn graphs where self loops would be unimportant but better connectivity is. However, through it we lose some of the algebraic structure that was exhibited in the standard De Bruijn graphs.

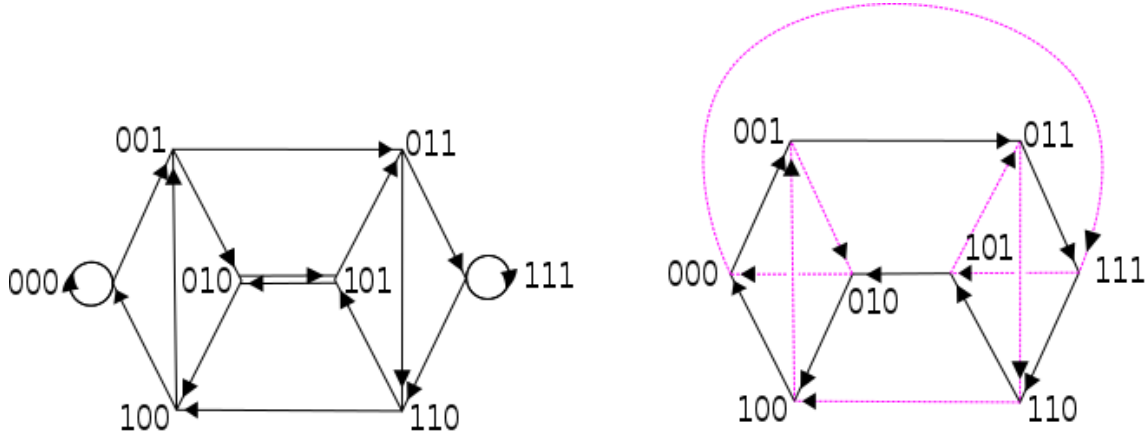


FIGURE 16.  $\mathcal{B}(2, 3)$  (left) and the modification (right).

The algebraic structure we exhibited in Section 4.2 fails because of the modifications to the choice of successors for a node. In the original De Bruijn graph we saw that an edge from  $(\alpha_1\alpha_2 \cdots \alpha_n)$  to  $(\alpha_2\alpha_3 \cdots \alpha_{n+1})$  was naturally described using the  $n + 1$  length string  $\alpha_1 \cdots \alpha_n\alpha_{n+1}$ . This property allowed a recurrence relation to choose the next node. However, in this modified graph  $n + 1$  characters are no longer sufficient to describe every edge, for example we see this in the case of the edge from  $(aa \cdots a)$  to  $(bb \cdots b)$ . It is this property that will prohibit our recurrence relations from being able to accurately construct De Bruijn cycles.

*Example.* We can look at Figure 16 as an example of this modification on the original De Bruijn graph  $\mathcal{B}(2, 3)$ . The dotted and solid lines each represent edge disjoint Hamiltonian cycles on the graph. We notice here that the modified graph now has 2 such cycles rather than the single Hamiltonian cycle that can be constructed on  $\mathcal{B}(2, 3)$ .

The second modification (discussed in [RB93b]) is done with the exact same process except to the undirected De Bruijn graph  $\mathcal{UB}(d, n)$  rather than the directed De Bruijn graph  $\mathcal{B}(d, n)$ . Rowley and Bose discuss how this version of the De Bruijn graph is perhaps a stronger model of actual multiprocessor networks since most communication paths between processors are naturally able to transmit information in both directions.

*Example.* We can look at Figure 17 below as an example of this modification on the undirected De Bruijn graph  $\mathcal{UB}(2, 3)$ . Again, the dotted and solid lines represent edge disjoint cycles on the graph.

## 6. CONCLUDING STATEMENTS

In Section 3 we examined N. G. de Bruijn’s original proof in [dB46] that there are  $2^{2^{n-1}-n}$  Hamiltonian cycles in the De Bruijn graph  $\mathcal{B}(2, n)$ . We saw in Section 4.1 that in the case

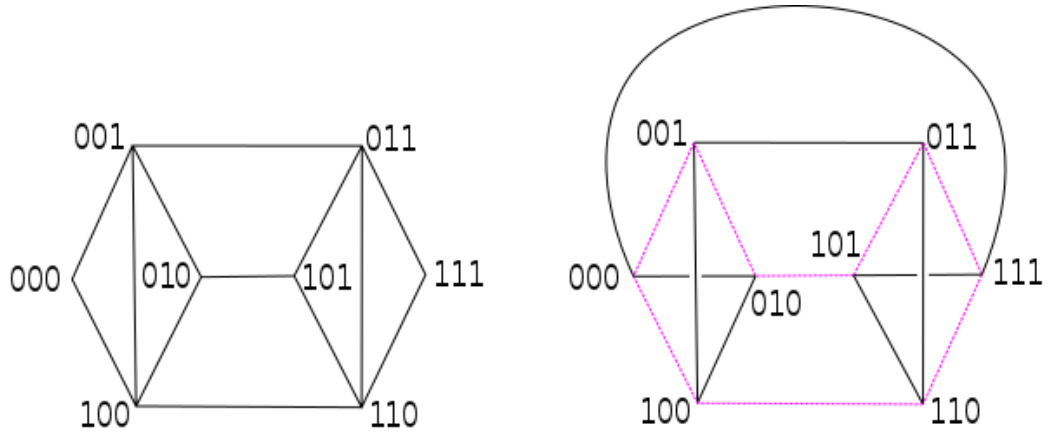


FIGURE 17.  $\mathcal{UB}(2, 3)$  (left) and the modification (right).

of a single node failure on  $\mathcal{B}(2, n)$  that we could use the technique of removing a necklace to show that these graphs admit cycles of length greater than or equal to  $2^n - n - 1$ .

We then saw in Section 4.2 that the standard De Bruijn graph  $\mathcal{B}(d, n)$  admits at least one cycle of length  $d^n - 1$  under  $d - 1$  edge failures when  $d$  is a prime power. In order to prove this bound we used irreducible polynomials and recurrence relations in order to construct these cycles. We then saw from [RB93a] that we could use this knowledge to provide a lower bound for the number of Hamiltonian cycles on  $\mathcal{B}(d, n)$  for any  $d$ , based on its prime decomposition.

Lastly, we examined some modifications to De Bruijn graphs and sequences. The  $\mathcal{Q}_n$  cycles gave us an example of a different type of structure that we could examine using the techniques developed in Section 3. Finally, we looked at the modifications presented by Rowley and Bose in [RB91] and [RB93b] that yield slightly stronger results but lost some of the original structure.

**6.1. Further Directions.** There are a few different directions that one could take for further research on De Bruijn graphs and their applications. First, we note that the method discussed in Section 4.2 for finding edge disjoint Hamiltonian cycles on De Bruijn graphs is dependent on the fact that our alphabet was chosen with  $d$  elements, where  $d$  is a prime or prime power. If we choose  $d$  to be composite then the current bounds have room for improvement. Z. Kása proposes in [Kás10] that the number of edge disjoint Hamiltonian cycles is  $d - 1$ . This bound differs greatly from the one proposed by Rowley and Bose in [RB93a] discussed in Section 4.2.

Another direction one could take is to expand the study of node failure to De Bruijn graphs other than binary De Bruijn graphs. Our investigation in Section 4.1 made use of the concept of weight which is not immediately applicable to De Bruijn graphs,  $\mathcal{B}(d, n)$ , where  $d > 2$ .

Lastly, we see that one could investigate possible uses for De Bruijn graphs outside of fault tolerance. We see from the work of Renault, Scarsini and Tomala [RST07] that De Bruijn graphs can be used to understand the Minority Game in the field of game theory. In addition Delvenne and Carli [DC07] use the De Bruijn graph in their approach to the Average Consensus Problem, a problem involving communication among independent agents. With these varied applications, we see that De Bruijn graphs are versatile and useful graphs which are worthy of further investigation.

## REFERENCES

- [CDMP92] Oliver Collins, Sam Dolinar, Robert McEliece, and Fabrizio Pollara. A vlsi decomposition of the De Bruijn graph. *J. ACM*, 39:931–948, October 1992.
- [dB46] N. G. de Bruijn. A combinatorial problem. *Nederl. Akad. Wetensch., Proc.*, 49:758–764 = *Indagationes Math.* 8, 461–467 (1946), 1946.
- [DC07] J.-C. Delvenne and R. Carli. Optimal strategies in the average consensus problem. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2498–2503, dec. 2007.
- [EH85] Abdol-Hossein Esfahanian and S. Louis Hakimi. Fault-tolerant routing in De Bruijn communication networks. *IEEE Trans. Comput.*, 34(9):777–788, 1985.
- [HP03] W. C. Huffman and V. Pless. *Fundamentals of error-correcting codes*. Cambridge Univ. Press, 2003.
- [Kás10] Zoltán Kása. On arc-disjoint hamiltonian cycles in De Bruijn graphs. *CoRR*, abs/1003.1520, 2010.
- [RB91] R. Rowley and B. Bose. Edge-disjoint hamiltonian cycles in De Bruijn networks. In *Distributed Memory Computing Conference, 1991. Proceedings., The Sixth*, pages 707–709, apr-1 may 1991.
- [RB93a] Robert Rowley and Bella Bose. On the number of arc-disjoint Hamiltonian circuits in the De Bruijn graph. *Parallel Process. Lett.*, 3(4):375–380, 1993.
- [RB93b] Robert A. Rowley and Bella Bose. Fault-tolerant ring embedding in De Bruijn networks. *Computers, IEEE Transactions on*, 42(12):1480–1486, dec 1993.
- [Ros] Vladimir Raphael Rosenfeld. Enumerating De Bruijn sequences. In *MATCH Communications in Mathematical and in Computer Chemistry*, page 2002.
- [RST07] Jérôme Renault, Marco Scarsini, and Tristan Tomala. A minority game with bounded recall. *Math. Oper. Res.*, 32(4):873–889, 2007.