5-2013

Universal Cycles for Some Combinatorial Objects

Andre A. Campbell
East Tennessee State University

Universal Cycles for Some Combinatorial Objects

A thesis

presented to

the faculty of the Department of Mathematics

East Tennessee State University

In partial fulfillment

of the requirements for the degree

Master of Science in Mathematical Sciences

---

by

André Campbell

May 2013

Anant Godbole, Ph.D., Chair

Rick Norwood, Ph.D.

Debra Knisley, Ph.D.

Keywords: Universal Cycle, de Bruijn Cycle, Posets, Boolean Lattice

ABSTRACT

Universal Cycles for Some Combinatoral Objects

by

André Campbell

A de Bruijn cycle commonly referred to as a universal cycle (*u-cycle*), is a complete and compact listing of a collection of combinatorial objects. In this paper, we show the power of de Bruijn's original theorem, namely that the cycles bearing his name exist for *n-letter* words on a *k-letter* alphabet for all values of k, n, to prove that we can create de Bruijn cycles for multi-sets using natural encodings and *M-Lipschitz n-letter* words and the assignment of elements of $[n] = \{1, 2, ..., n\}$ to the sets in any labeled subposet of the Boolean lattice; de Bruijn's theorem corresponds to the case when the subposet in question consists of a single ground element. In this paper, we also show that de Bruijn's cycles exist for words with weight between s and t, where these parameters are suitably restricted.

DEDICATION

I would like to dedicate this thesis to my grandfather Alphonso Alexander Tyndale. He was a loving, caring man and a man of strong positive words. He always mentioned that education is the key to success and explained to us (grandchildren) how important it was to respect time because time waits on no man. I didn't understand what he meant back then, but I surely do now. Finally, a special feeling of gratitude to my loving parents, Owen and Ruth Campbell whose words of encouragement and push for tenacity ring in my ears. My sister Krystina Campbell, who was always asking about the progress of my thesis. I appreciate all that they have done and thank you for being my best cheerleaders. Without you all, this would not be possible.

ACKNOWLEDGMENTS

This thesis was done by André Campbell under the keen supervision of Anant Godbole. I wish to thank my committee members, Dr. Anant Godbole, Dr. Rick Norwood and Dr. Debra Knisley who were more than generous with their precious time. Thank you Dr. Rick Norwood and Dr. Debra Knisley for agreeing to serve on my committee and for proofreading the thesis. Finally, a special thank you to Dr. Anant Godbole, my committee chairman, for his countless hours of reflecting, proofreading and most of all patience throughout the entire thesis process, from the introduction to the meetings over the summer break, winter break and spring break, to the final defense.

TABLE OF CONTENTS

ABSTRACT 2

DEDICATION 4

ACKNOWLEDGMENTS 5

1 INTRODUCTION 7

2 POSETS 16

3 WORDS 28

4 CONCLUDING REMARKS 44

BIBLIOGRAPHY 45

VITA 46

1 INTRODUCTION

A more rigorous definition of a *u-cycle* is a cyclic sequence which contains each element of a collection of combinatorial objects exactly once. An example of such a cycle is the de Bruijn cycle of order n. Let $n = 3$, then the binary cyclic string 11101000 contains all eight possible binary strings of length 3 exactly once ($111 \rightarrow 110 \rightarrow 101 \rightarrow 010 \rightarrow 100 \rightarrow 000 \rightarrow 001 \rightarrow 011$). Another example of a u-cycle is 1234524135. This contains all 2-subsets of [5], where $[n] = \{1, 2, 3, ..., n\}$, exactly once ($12 \rightarrow 23 \rightarrow 34 \rightarrow 45 \rightarrow 52 \rightarrow 24 \rightarrow 41 \rightarrow 13 \rightarrow 35 \rightarrow 51$), where the notation ij (or ji) is the shorthand for the set $\{i, j\}$ for $i < j$. Do universal cycles of k-subsets of n always exist? For $k = 2$, the answer is yes, if $n \geq 3$ and is odd! For $k \geq 4$, very little is known. See the Hurlbert paper [5] for results along these lines. Now this brings us to the de Bruijn Theorem.

Theorem 1 *(de Bruijn) A u-cycle exists for n-letter words on a k-letter alphabet for all* $k, n \in \mathbb{Z}^+$.

Theorem 2 *A connected digraph is Eulerian if and only if the in-degree of each vertex is the same as its out-degree. More generally, a digraph G is Eulerian if it has one non-trivial weakly connected component and if i(v) = o(v) for each v.*

In order to prove the De Bruijn Theorem we use the above stated theorem in its weak form.

de Bruijn Proof:

First we create a digraph, G, that has vertices that consist of all $n-1$ letter words on a *k-letter* alphabet. In other words, vertices v have one less letter than the words we are trying to *u-cycle*, which then appear as edge labels between vertices as follows: a directed edge is drawn from v_1 to v_2 if the last $n-2$ letters of v_1 are the same as the first $n-2$ letters of v_2, and is then labelled with the corresponding concatenated *n-letter* word. For example, let $k = 5$ and $n = 5$, then the edge from $ANDR \rightarrow NDRE$ will be labelled $ANDRE$. Therefore, it is then easy to see that the conditions of Theorem 2 are satisfied, and that the Eulerian circuit generates the required *u-cycle*. □

Now this bring us to one of the three main theorems of this thesis. We call this theorem the Lipschitz Theorem.

Theorem 3 *Given a modular k-letter alphabet, a u-cycle of M-Lipschitz n-letter words exists if condition C holds.*

Before presenting the proof of the Lipschitz Theorem, we define what a Lipschitz word is, specify condition C, and show an example of a *u-cycle* of all Lipschitz words. Firstly, a Lipschitz word can be defined as follows: a real-valued function ϕ is Lipschitz if there exists a real number M such that for all numbers x and y, $|\phi(x) - \phi(y)| \leq M|x-y|$. Or similarly, a word on an ordered alphabet such as (A,B,...,Z) is Lipschitz if $|f_i - f_j| \leq M|i-j|$ where f_i is the ith letter of the word. We let $M = 1$, though the definition of M-Lipschitz words still holds true even if $M \neq 1$. Secondly, condition C is: $|f_{i+1} - f_i| \leq M$. This implies the M-Lipschitz condition since $|f_i - f_j| \leq$

$|f_i - f_{i+1}| + |f_{i+1} - f_{i+2}| + ... + |f_{j-1} - f_j| \leq M + M + M + M + ... + M = M|i - j|$. The converse however is not true. Finally, for example, let $n = 2$ and let $k = 3$, i.e. A, B, C. Then the word $AABCCBB$ contains all possible strings of length 2 exactly once ($AA \to AB \to BC \to CC \to CB \to BB \to BA$). Note that AC and CA are 2-Lipschitz but not 1-Lipschitz which then explains why they were not included in the *u-cycle*.

Proof of Theorem 3:

If $n = 1$, then all words are adjacent, then trivially a *u-cycle* exists. Suppose that the alphabet is modular so that, e.g., $|A - Y| = 2$. First we create a graph G whose vertices consist of M-Lipschitz words with one less letter than the word we are trying to *u-cycle*, which then appears as an edge label between the vertices as follows: a directed edge is drawn v_1 to v_2 if the last $n - 2$ letters of v_1 are the same as the first $n-2$ letters of v_2, and is then labelled with the corresponding concatenated *Lipschitz n-letter* word. For this to happen, the letters must be at most length M apart. For example, let $k = 5$ and $n = 5$ as in the proof before and let $M = 1$. Then the edge from $ABCD \to BCDE$ will be labelled $ABCDE$ which is a legitimate Lipschitz word. A graph is said to be connected if there exists a $u - v$ path between any two arbitrary vertices u, v. In the previous example, if the letters in the vertex word are not at most length one apart, then the *Lipschitz n-letter* word does not exist as an edge. For example, let $k = 4$ and $n = 4$, then the edge from $BDA \to DAC$ would be labelled $BDAC$ which is not a Lipschitz word with $M = 1$. Thus, the vertices

must be Lipschitz as well. We know that in order for the conditions of Theorem 1 to be satisfied, the in-degree of v and the out-degree of v must be equal for it to be Eulerian and a *u-cycle* of *M-Lipschitz n-letter* words on an *k-letter* alphabet exists. Notice that only condition C and the modularity of the alphabet makes this true; $i(v) = o(v) = 2M + 1$ for every v. This finishes the proof. □

There are some very important references that must be mentioned. First, we have the paper by Leitner and Godbole titled "Universal Cycles of Classes of Restricted Words." In this paper [3] they proved that Universal Cycles (U-Cycles) exists for several restricted classes of words, which included non-bijections, equitable words (under suitable restrictions), ranked permutations, and passwords. Leitner and Godbole in [3] defined a word as equitable if for all letters i, j, $||i| - |j|| \leq 1$ where $|i|$ is the number of times i occurs. They also said that words formed must follow ordinary rankings in a tournament. A trivial example to explain this is: the ranking 113 is allowed, but the ranking 112 is not allowed since second place is already taken in the tie with first. They also defined a password as an n-letter word of length k on $[n]$ where there are q distinct classes of symbols in $[n]$, $q \leq k < n$ distinct classes of symbols in n, and each word must contain at least one element of each class. In other words, you can think of this as a security measure that protects one's personal accounts, where passwords must contain at least one number, one lower case letter, one symbol, etc.

This brings us to our other key reference, Blanca and Godbole [4], one of the most important references of this thesis. This paper is titled "On Univeral Cycles for new Classes of Combinatorial Structures". In this paper [4], the authors used natural encodings of these objects to show that there exist u-cycles for collections of subsets, matroids, restricted multisets, chains of subsets, multichains, and lattice paths. They showed that for subsets, there exists a *u-cycle* for k-subsets of an n-set if we let the k vary in a non-zero length interval. They also showed that there exists u-cycles for all $n - length$ words over some alphabet Γ, which contain all characters from $A \subset \Gamma$.

Blanca and Godbole mentioned that the subset $\{1, 2\}$ does not have to be coded as 12, $\{2, 5\}$ does not have to be coded as 52, and so on. Before getting into the different codings, let me first explain the two types of coding. The first type of coding for multisets is called the *k-coding* and the second type of coding for multisets is called the *n-coding*. The authors of [4] defined a universal cycle of a set A of k-element multisets of $[n]$ as a cyclic sequence of length $|A|$ such that for all $a \in A$ its corresponding string appears exactly once in the sequence. For example, consider the sequence 112233. This is a legal *u-ycle* of all the multisets of size $k = 2$ from $\{1, 2, 3\}$ ($[n] = 3$), namely $\{1, 1\}, \{1, 2\}, \{2, 2\}, \{2, 3\}, \{3, 3\}, \{3, 1\}$. This is a *k-coding*. Throughout the next few pages, blue represents the *k-coding* and red represents *n-coding*, then we proceed as follows:

$$\{1,1\} = < 2, 0, 0 >$$

$$\downarrow$$

$$\{1,2\} = < 1, 1, 0 >$$

$$\downarrow$$

$$\{2,2\} = < 0, 2, 0 >$$

$$\downarrow$$

$$\{2,3\} = < 0, 1, 1 >$$

$$\downarrow$$

$$\{3,3\} = < 0, 0, 2 >$$

$$\downarrow$$

$$\{3,1\} = < 1, 0, 1 >$$

In the first case, for example, both representations indicate that the multiset contains 1 twice and does not contain 2 or 3. However, the first coding uses $k = 2$ letters and the second uses $n = 3$ letters. For the same example, a *u-cycle* does not exist for the *n-coding* since we get a cycle with $< 1, 1, 0 > \rightarrow < 1, 0, 1 > \rightarrow < 0, 1, 1 > \rightarrow < 1, 1, 0 >$. In general, both k and n codings may give a *u-cycle*, or just one, or none. However, the *n-coding* of **multisets** of any size between 0 and $n(k-1)$ of an n element set, are words of weight k from an alphabet $\{1, 2, ..., k\}$, where the weight is the sum of the elements in the *n-coding*. And the *n-coding* of **sets** of any size between 0 and $n(k-1)$ of an n element set are words of weight k from an alphabet $\{0, 1\}$. Let us look at another example 1100011000, where we are attempting to use the *n-coding* for all

k-subsets of $[n]$ with $k = 2$ and $n = 5$. We are forced into a cycle and hence there does not exist a *u-cycle* using the types of *n-codings* mentioned above for $n = 5$ and $k = 2$.

$$< 1, 1, 0, 0, 0 > = \{1,2\}$$

$$\downarrow$$

$$< 1, 0, 0, 0, 1 > = \{1,5\}$$

$$\downarrow$$

$$< 0, 0, 0, 1, 1 > = \{4,5\}$$

$$\downarrow$$

$$< 0, 0, 1, 1, 0 > = \{3,4\}$$

$$\downarrow$$

$$< 0, 1, 1, 0, 0 > = \{2,3\}$$

$$\downarrow$$

$$< 1, 1, 0, 0, 0 > = \{1,2\}$$

However, a *u-cycle* does exist for the *k-coding* as seen by 1234524135. However, the $\{0,1\}$ *n-coding* does work if we can change the question, and this is one of the main contributions of [4].

Another important fact noticed in [4] was that a binary word of weight k is equivalent to a subset of size k. They used *n-coding* to show that there exists a *u-cycle* of all subsets of size between s and t, $s < t$, if it is constructed correctly, even though

it is not a trivial thing to do. For $s = t$ there is no guarantee of a *u-cycle* in either coding, and if one coding does produce a *u-cycle*, the other coding may not. For example, given the binary coding 1110011010 with $n = 4, s = 2, t = 3$:

$$< 1, 1, 1, 0 > = \{1,2,3\}$$

$$\downarrow$$

$$< 1, 1, 0, 0 > = \{1,2\}$$

$$\downarrow$$

$$< 1, 0, 0, 1 > = \{1,4\}$$

$$\downarrow$$

$$< 0, 0, 1, 1 > = \{3,4\}$$

$$\downarrow$$

$$< 0, 1, 1, 0 > = \{2,3\}$$

$$\downarrow$$

$$< 1, 1, 0, 1 > = \{1,2,4\}$$

$$\downarrow$$

$$< 1, 0, 1, 0 > = \{1,3\}$$

$$\downarrow$$

$$< 0, 1, 0, 1 > = \{2,4\}$$

$$\downarrow$$

$$< 1, 0, 1, 1 > = \{1,3,4\}$$

$$\downarrow$$

$$< 0, 1, 1, 1 > = \{2,3,4\}.$$

As can be seen once again, this is a complete list and there exists a u-cycle. Hopefully theorem 5 will generalize all of the above for words with weights between s and t ($s < t$) i.e., multisets of size between s and t, where $t \geq s + (d - 1)$, where d is the size of the alphabet.

In the next two chapters (Posets and Words) we will be defining some very important terminology that will be used to prove the main theorems of this thesis.

Theorem 4 *There exists a u-cycle of all assignment of the elements of* $\{1, 2, 3, ..., n\}$ *to the sets in an ordered subposet of the Boolean Lattice whose Hasse Diagram has any given shape.*

Theorem 5 *Given* $s < s + (d - 1) \leq t$ *there exists a u-cycle of all words with weight between* s *and* t*.*

2 POSETS

In this chapter we analyze another very interesting set of combinatorial objects. Blanca and Godbole in [4] talked about universal cycles of chains of subsets. They mentioned that given any $[n]$, where $[n] = \{1, 2, 3, ..., n\}$, a *k-chain* of $[n]$ is a sequence of sets $A_1 \subseteq A_2 \subseteq A_3 \subseteq ... \subseteq A_k \subseteq [n]$. As can be seen, the number of distinct *k-chains* of $[n]$ is $(k+1)^n$, where the 1 arises from the fact that a subset may be empty. After figuring out the number of distinct *k-chains*, they created a natural encoding for these distinct chains of $[n]$, and then finally analyzed the existence of universal cycles (*u-cycles*) [4].

In [4], let each A_i of some *k-chain* be coded by a binary string of length n. In this case they thought of the binary string as an ordered binary n-tuple denoted by s_i. They also defined an operation $\oplus$ as follows: $s_i \oplus s_j = s'$ if and only if $s'(k) = s_i(k) + s_j(k)$ for all k such that $1 \leq k \leq n$. For example, if $a = \{1, 2, 3\}, b = \{2, 2, 2\}$, and $c = \{1, 1, 1\}$. Then, $a \oplus b \oplus c = \{4, 5, 6\}$.

A Partially Ordered Set $(P, <)$ commonly referred to as a Poset is a set P and a relation $<$ on P that satisfies the follwing properties:

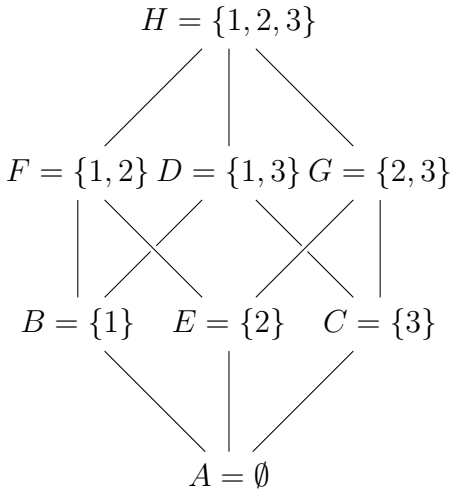
(i) Reflexive, i.e. $S < S$,

(ii) Transitive, i.e., $S < T$ and $T < U$, then $S < U$.

(iii) Antisymmetric, i.e. if $S < T$ and $T < S$, then $S = T$.

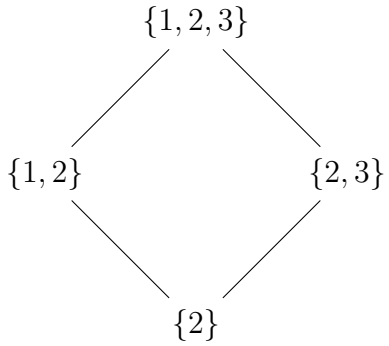
A Hasse Diagram of P is a diagram that represents the transitive reduction

and it exists if $A < B$, $A \neq B$ and there doesn't exist a C such that $A < C < B$. Now that we have defined a Poset and a Hasse Diagram, it is time to define what the Boolean lattice is, and give an example of what one looks like. We denote the Boolean Lattice (B_n) and we define it to be the collection of **ALL** subsets of $[n]$, where $[n] = \{1, 2, 3, 4, ..., n\}$ and where $A < B$ if $A \subseteq B$. For instance, look at the Hasse Diagram of the Boolean Lattice (B_3).

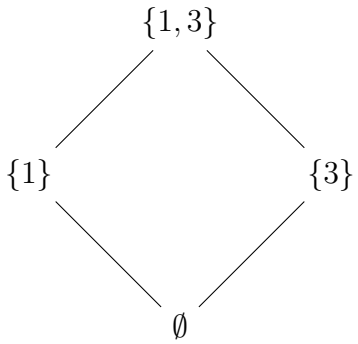


Shown here are all the collections of subsets of $[n] = [3]$ including the emptyset, where $A \subseteq B, E, C$; $B \subseteq F, D$; $E \subseteq F, G$; $C \subseteq D, G$; and $F, D, G \subseteq H$. Notice that for example $B \subseteq H$ but there are no arrows from $B \to H$ since it follows from the property of transitivity. Trivially, the $\emptyset$ is a subset of all sets $\{A, B, C, ..., H\}$. There are many different subposets of B_3. Before we get into subposets, we define what a subposet is. A subposet Q consists of a Hasse diagram that is consistent with the Boolean lattice, and an assignment of subsets of $\{1, 2, \ldots, n\}$ to the sets in Q. For

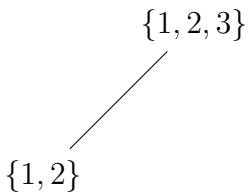
example, take the diamond subposet:



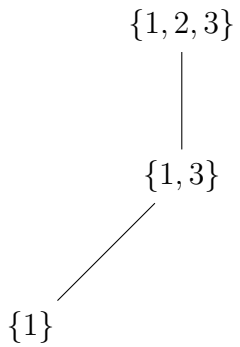
This is the subposet $\{A, B, C, D\}$ where $A \subseteq B, C$ and $B, C \subseteq$ D, and the sets A, B, C, D follow the rules of inclusion dictated by the Hasse Diagram. In this example $A = \{2\}, B = \{1, 2\}, C = \{2, 3\}$ and $D = \{1, 2, 3\}$. Or we could have



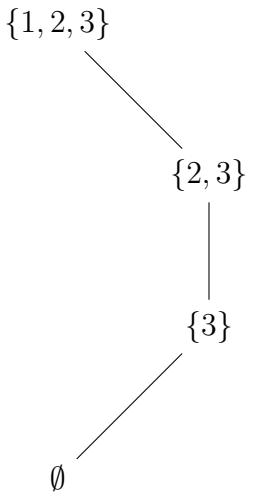
Alternately, the subposet could be any *k-chain*. For instance, we could have taken any 2-*chain*:



or 3-*chain*:



or 4-*chain*:



It just depends on which subposet you choose from the Boolean Lattice. Now that we have the general idea of what a subposet is, let's look at an example of a *u-cycle* of a 2-*chain*.

Given the above 2-*chain*, first we figure out all the allowable configurations for any elements $j \in \{1, 2, 3, ..., n\}$. For instance,

B 0 1 1

A 0 0 1

are all allowable, where

1

0

indicates that $j \in B$ but $j \notin A$. Note, however, that we cannot have a configuration of

B 0

A 1

because $A \subseteq B$ and we cannot have $j \in A, j \notin B$. Note that the allowable two vectors of $\{A, B\}$ are $(0,0), (0,1), (1,1)$. Since we have figured out all our allowable configurations, it is time to code these configurations. C is used to denote the coding for these allowable configurations.

B 0 1 1

A 0 0 1

C a b c

Hence, we are only allowed 3^n possibilities, where n is length of the word. Notice that we coded 00 as an a, 01 as a b and 11 as a c. Now, let $n = 2$ and we end up with all 2-*chains* which are listed below:

$$\begin{array}{cccccccccc} 1 & \emptyset & 2 & 1,2 & 1 & 1,2 & 2 & 1,2 & 1,2 \\ | & | & | & | & | & | & | & | & | \\ \emptyset & \emptyset & \emptyset & \emptyset & 1 & 1 & 2 & 2 & 1,2 \end{array}$$

Now that we have all 2-*chains*, we use the binary coding to represent the sets as a characteristic vector.

B 10 00 01 11 10 11 01 11 11

A 00 00 00 00 10 10 01 01 11

W *ba aa ab bb ca cb ac bc cc*

Given the above characteristic coding for each vector, we now have all 2-*letter* words on a 3-*letter* alphabet and we can now use the de Bruijn Theorem to show the existence of a *u-cycle*. Figuring out the *u-cycle* for such words isn't always trivial. However, for this above example, we get the u-cycle $cc \to ca \to aa \to ab \to bb \to bc \to cb \to ba \to ac$ then back to *cc* where we started. This is a *u-cycle* of 2-*letter* words on a 3-*letter* alphabet. Hence, **ccaabbcba** is the u-cycle of words formed. We could have given our binary allowable configurations a different coding that would yield the u-cycle of words **220011210**. Notice that we coded *c* as a 2, *a* as a 0 and a *b* as a 1, and the u-cycle would now be $22 \to 20 \to 00 \to 01 \to 11 \to 12 \to 21 \to 10 \to 02$ then back to 22 where we started. Given any allowable configuration, coding is strictly up to the user.

B 11 10 00 01 11 11 11 10 01

A 11 10 00 00 00 01 10 00 01

Above we have the correspondence between the *u-cycle* of words and all binary coding representing the sets as characteristic vectors. Notice that for example, if we take

$$11$$

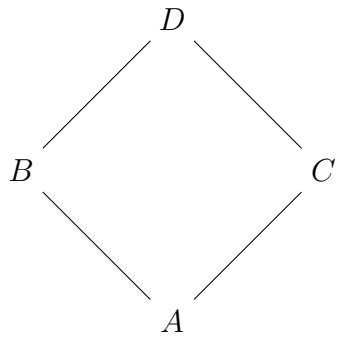
$$01$$

it is equal to

$$\{1,2\}$$
$$|$$
$$\{2\}$$

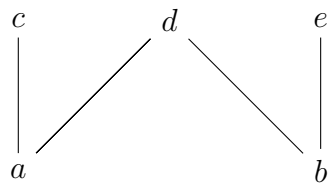
What does this poset mean? The above poset means that the element 1 does not belong to A but it does belong to B and the element 2 belongs to both A and B and the other posets can be created the same way. Notice that the 2-*chain-lattice* has 3 antichains, namely $\emptyset, A$ and B.

A set S of elements in a subposet Q is an anti-chain if there does not exist x and y in S with $x < y$ and $x \neq y$. For instance, in the diamond poset below,

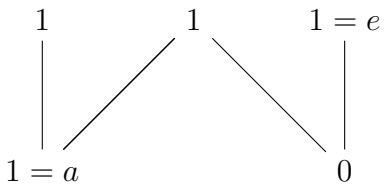


$\{A\}, \{B\}, \{C\}, \{D\}, \{B, C\}$, yield all 5 antichains; and if we include one more case representing the empty set, there are 6 possibilities for each j and thus 6^n assignments of elements to the subposet. Note that $\{B, D\}$, $\{C, D\}$ and $\{A, D\}$ are not antichains as they violate the definition.

The same can be done for all subposets, no matter how complicated they are, such as:

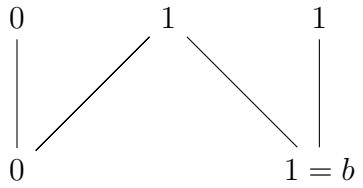


We call this subposet the W-subposet mainly because it takes the shape of the letter W. Given the W-subposet, now we have to find all the antichains. The antichains are $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{a, b\}, \{c, d\}, \{a, e\}, \{b, c\}, \{c, e\}, \{d, e\}, \{c, d, e\} + 1 = 13$ possibilities. Thus there are 13^n possible assignments of $\{1, 2, 3, ..., n\}$ to the 5 sets, as given below. Now that we have found all the anitchains, we can make a correspondence between the assignments of the elements to a subposet P and all *n-letter* words on a *k-letter* alphabet, where k is the number of antichains +1. Given that we have the anitchains, we now have the freedom to choose whichever anitchain we want to represent the element 1. For instance, say for the element 1 we choose the antichain $\{a, e\}$. Then, we let a and e contain 1 and everything that is above it gets a 1 also, otherwise it gets a 0 and we get the assignments of element 1 to the W subposet



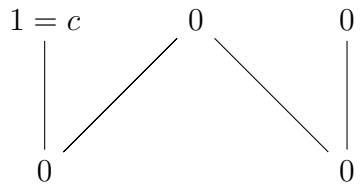
This W subposet now tells us in what sets the element 1 goes.

Now for the element 2, let's choose the set $\{b\}$ to be the antichain. Then we let b be a 1, and everything above it gets a 1, everything else gets a 0 and we get the W subposet

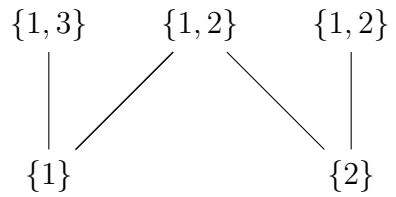


Now, this W subposet tells us in what sets the element 2 goes.

Also, for the element 3 suppose we choose the set $\{c\}$ to be the antichain. Then we let c be a 1, and everything above it gets a 1. However, there is nothing above $\{c\}$ so everything else gets a 0 and we get the W subposet



This W subposet now tell us in what sets the element 3 goes. We repeat for elements $4, 5, ..., n$. Finally, if we were to let $[n] = [3]$, then we get the following assignment of $\{1, 2, 3\}$ to the 5 sets in the W subposet below.



Lemma *There are* $(\alpha + 1)^n$ *assignments of the elements of* $[n]$ *to the elements of* P*, where* α *is the number of antichains in* P*. Furthermore, there exists a de Bruijn cycle of these assignments.*

Proof: Given any element $j \in [n]$ it is clear that if $j \in A$ for some $A \in P$, we must have $j \in B$ for each $B > A$, since $<$ is the inclusion relation. It thus follows that the number of ways to assign element j to the sets in P is equal to the number of ways of labeling the elements of P with zeros and ones so that if A is labeled by a 1, then so is each $B \supseteq A$; this includes the case where all the elements of P are labeled with zeros. We call a coloring $c : P \to \{0, 1\}$ *unitarily up-closed* if $c(A) = 1 \Rightarrow c(B) = 1$ for each $B \in P$ with $B > A$. We will prove below that the set of unitarily up-closed colorings is in bijection with the antichains of $(P, <)$; note that $\emptyset$ is not considered to be an antichain in P, but all one-element sets are; this empty set leads to the "+1" in the Lemma.

For one direction, let c be a unitarily up-closed coloring on $(P, <)$. Let $M_c = \{A \in P : c(A) = 1$ and $c(B) = 0$ for every $B < A\}$ i.e., the set of elements of P minimal with respect to receiving the value 1 under c. Clearly, M_c forms an antichain, since if there exist $A, B \in M_c$ with $A < B$, then A does not meet the constructive criteria of M_c. This provides a map from colorings to antichains. To see that the map is injective, let c and d be two different unitarily up-closed colorings, and suppose that

$M_c = M_d$. Fix $A \in P$. Suppose that $c(A) = 1$. If A is minimal, then we have that $d(A) = 1$ since $M_c = M_d$. If A is not minimal, it is above some $B \in M_c$. $B \in M_d$ by hypothesis, and since $B < A$, $d(A) = 1$ as d is unitarily up-closed. Now suppose that $c(A) = 0$. Since A is not above *any* element of M_c, A is also not above any element of M_d. Hence $d(A) = 0$, as if $d(A) = 1$ we would have that A was above some minimal element under d.

To see the reverse, notice that for a fixed antichain $\mathbf{A}$, we can define a unitarily closed up-coloring $c = c_{\mathbf{A}}$ by the rule $c(A) = 1$ iff $B < A$ for some $B \in \mathbf{A}$, 0 otherwise. We claim that $M_c = \mathbf{A}$, and hence this provides a $1-1$ inverse to our $1-1$ function. Since no element below any element of $\mathbf{A}$ receives color 1, clearly $\mathbf{A} \subseteq M_c$. On the other hand, every element A with $c(A) = 1$ is above some element of $\mathbf{A}$, and since M_c contains every element of $\mathbf{A}$, no other element receiving color 1 can belong to M_c, and we are done.

It follows that the "fate" of each element of $[n]$, i.e., which sets in P it belongs to, can be determined in α ways, and thus there are α^n assignments of the elements of $[n]$ to the sets in P. □

Theorem 4 *There exists a u-cycle of all assignment of the elements of* $\{1, 2, 3, ..., n\}$ *to sets in an ordered subposet of the Boolean Lattice whose Hasse Diagram has any given shape.*

Proof: Given any Boolean Lattice B_n and a subposet P of B_n, we start by computing the number of antichains in P. For example, if P is a *k-chain*, there are $(k+1)-$antichains; if P is a diamond, there are 5 antichains; if P is a W-poset there

are 12 antichains; and if P is a multichain M with *k-chain* $M_{k_1}, M_{k_2}, ..., M_{k_r}$, then there are $k_1 \cdot k_2 \cdot k_3 \cdot ... \cdot k_r$ antichains. Bill Kay, our collaborator, proved that the number of assignments of the elements of $\{1, 2, 3, ..., n\}$ to P is $(A+1)^n$, where A is the number of antichains. This holds true for any ordered P. The idea of the proof is as above; label each element of an antichain (and anything above it) by 1 and all other elements by 0. Then, the resulting configuration of $0's$ and $1's$ is a legal assignment of the element j to the sets in P. Repeating for each $j \in [n]$ we get $(A+1)^n$ possibilities overall. The key ingredient is that there is a bijection between antichains and legal assignments. Finally, De Bruijn's theorem in [3] is used to complete the proof, as illustrated in the 2-*chain* example. □

3 WORDS

A word of length n on an alphabet of size d is $(a_1, a_2, a_3, ..., a_n)$, where $a_i \in \{0, 1, ..., d-1\}$ and the $\sum a_i$ = weight of the word. The length a vertex is $n-1$, edges are of length n, s is the minimum weight of the word, t is the maximum weight of the word and $s < t$. If $d = 2$, then we get results proven in [4] that there exists a u-cycle of all binary words of weight between s and t. Notice that in [4] Blanca and Godbole proved the smaller case of a binary alphabet while in Theorem 5, we will be doing the proof for a larger alphabet.

Some *u-cycles* were found in the paper written by Ariel Leitner and Anant Godbole [3] for different kinds of restrictions on d. An example of such a *u-cycle*, a *password* is a *length-n* word over an alphabet d which contains at least one character from q distinct subsets (classes) of d. It was also shown in the same paper that *u-cycles* of passwords existed for as long as the condition $2q \leq n$ was satisfied. In the case where each class is of size 1, we notice that such strong passwords are not necessary.

This now brings us to one of our main theorems.

Theorem 5 *Given $s < s + (d-1) \leq t$ there exists a u-cycle of all words with weight between s and t.*

The objective of this proof is to show that G is Eulerian by showing that $i(v) = o(v)$ for each v and that the graph is weakly connected, i.e., that there exists a path between each $v \in G$ and some "sink" vertex denoted sv. The sink vertex is defined as the vertex of weights that contains a desired number of letters that are all either

$x's$ or $(x+1)'s$. Now that we have defined the sink vertex, we explain how the sink vertex is calculated.

The sink vertex is calculated by first calculating the quantity $\frac{s}{n-1}$, where s is the minimum weight of an edge or word formed between any two pair of connected vertices, d is the size of the alphabet, $n-1$ is the length of the vertex and n is the length of the edge or word formed between any two pair of connected vertices. This gives the "average letter in a vertex of weight s."

After $\frac{s}{n-1}$ has been calculated, if we get a whole number, w, then that number w represents what the weight of each letter of the sink vertex should be. On the other hand, if we get a mixed fraction (do not simplify), then the whole number (w) tells us that our sink vertex is going to contain letters that are either w or $(w+1)$. The numerator of the mixed fraction tells us how many letters of weight $(w+1)$ will be in our sink vertex and the difference between the denominator and the numerator tells us how many letters of weight w our sink vertex will contain. Finally, if we have a proper fraction (again do not simplify), then our sink vertex is going to contain $0's$ and $1's$ with the numerator telling us how many $1's$ our sink vertex is going to contain and the difference between denominator and numerator tells how many $0's$ our sink vertex will contain. Below is an example of how to calculate our target vertex or sink vertex:

$s = 16, t = 20, d = 7, n = 4.$

$$\frac{s}{n-1} = \frac{16}{4-1}$$
$$= \frac{16}{3}$$

$$= 5\tfrac{1}{3}$$

Our sink vertex contains letters of weight $5's$ and $6's$. Hence, our sink vertex is $(5, 5, 6)$.

After doing several examples to try and generalize the idea behind showing weak connectedness, we came up with the following three examples that gave us the idea of showing weak connectedness in general. First, we start out with a big enough n, and in all examples we let $n = 11$, $d = 6$, $s = 25$ and $t \geq s + (d - 1) = 30$. First we calculate our target or sink vertex as shown in the previous paragraphs. Our target or sink vertex denoted $s.v. = (2, 2, 2, 2, 2, 3, 3, 3, 3, 3)$. Then we find our vertex weights and edge weights conditions. For this example, the weights of our edges has to be between $25 \rightarrow 30$ and our vertices has to be between weights: $20 \rightarrow 30$. Now one of our later lemmas that will be used to prove the main theorem states that, any vertex v of weight equal to s can be taken to another vertex v_1 that contains the desired number of weights $x's$ and $x + 1's$ of our target vertex.

We choose v to be $(0, 0, 0, 2, 2, 5, 5, 5, 3, 3)$. Throughout the next few pages, red numbers represent vertex weights and blue numbers represent edge weights, then we proceed as follows:

0,0,0,2,2,5,5,5,3,3 25

↓ 28

0,0,2,2,5,5,5,3,3,3 28

↓ 28

0,2,2,5,5,5,3,3,3,0 28

↓ 28

2,2,5,5,5,3,3,3,0,0 28

↓ 30

2,5,5,5,3,3,3,0,0,2 28

↓ 30

5,5,5,3,3,3,0,0,2,2 28

↓ 30

5,5,3,3,3,0,0,2,2,2 25

↓ 27

5,3,3,3,0,0,2,2,2,2 22

↓ 27

3,3,3,0,0,2,2,2,2,5 22

↓ 25

3,3,0,0,2,2,2,2,5,3 22

↓ 25

3,0,0,2,2,2,2,5,3,3 22

↓ 25

0,0,2,2,2,2,5,3,3,3 22

↓ 25

0,2,2,2,2,5,3,3,3,3 25

↓ 28

2,2,2,2,5,3,3,3,3,3 28

$\downarrow 30$

2,2,2,5,3,3,3,3,3,2 28

$\downarrow 30$

2,2,5,3,3,3,3,3,2,2 28

$\downarrow 30$

2,5,3,3,3,3,3,2,2,2 28

$\downarrow 30$

5,3,3,3,3,3,2,2,2,2 28

$\downarrow 30$

3,3,3,3,3,2,2,2,2,2 25

$\downarrow 28$

3,3,3,3,2,2,2,2,2,3 25

$\downarrow 28$

3,3,3,2,2,2,2,2,3,3 25

$\downarrow 28$

3,3,2,2,2,2,2,3,3,3 25

$\downarrow 28$

3,2,2,2,2,2,3,3,3,3 25

$\downarrow 28$

2,2,2,2,2,3,3,3,3,3 25

as desired.

For our second example, we choose u to be $(5, 5, 5, 5, 5, 0, 0, 0, 0, 0)$, then we proceed as follows:

5,5,5,5,5,0,0,0,0,0 25

↓ 30

5,5,5,5,0,0,0,0,0,5 25

↓ 30

5,5,5,0,0,0,0,0,5,5 25

↓ 30

5,5,0,0,0,0,0,5,5,5 25

↓ 30

5,0,0,0,0,0,5,5,5,5 25

↓ 30

0,0,0,0,0,5,5,5,5,5 25

↓ 28

0,0,0,0,5,5,5,5,5,3 28

↓ 28

0,0,0,5,5,5,5,5,3,0 28

↓ 28

0,0,5,5,5,5,5,3,0,0 28

↓ 28

0,5,5,5,5,5,3,0,0,0 28

↓ 28

5,5,5,5,5,3,0,0,0,0 28

↓ 30

5,5,5,5,3,0,0,0,0,2 25

↓ 27

5,5,5,3,0,0,0,0,2,2 22

↓ 27

5,5,3,0,0,0,0,2,2,5 22

↓ 27

5,3,0,0,0,0,2,2,5,5 22

↓ 27

3,0,0,0,0,2,2,5,5,5 22

↓ 25

0,0,0,0,2,2,5,5,5,3 22

↓ 25

0,0,0,2,2,5,5,5,3,3 25

↓ 28

0,0,2,2,5,5,5,3,3,3 28

↓ 28

0,2,2,5,5,5,3,3,3,0 28

↓ 28

2,2,5,5,5,3,3,3,0,0 28

↓ 30

2,5,5,5,3,3,3,0,0,2 28

↓ 30

5,5,5,3,3,3,0,0,2,2 28

↓ 30

5,5,3,3,3,0,0,2,2,2 25

↓ 27

5,3,3,3,0,0,2,2,2,2 22

↓ 27

3,3,3,0,0,2,2,2,2,5 22

↓ 25

3,3,0,0,2,2,2,2,5,3 22

↓ 25

3,0,0,2,2,2,2,5,3,3 22

↓ 25

0,0,2,2,2,2,5,3,3,3 22

↓ 25

0,2,2,2,2,5,3,3,3,3 25

↓ 28

2,2,2,2,5,3,3,3,3,3 28

↓ 30

2,2,2,5,3,3,3,3,3,2 28

↓ 30

2,2,5,3,3,3,3,3,2,2 28

↓ 30

2,5,3,3,3,3,3,2,2,2 28

↓ 30

5,3,3,3,3,3,2,2,2,2 28

↓ 30

3,3,3,3,3,2,2,2,2,2 25

↓ 28

3,3,3,3,2,2,2,2,2,3 25

↓ 28

3,3,3,2,2,2,2,2,3,3 25

↓ 28

3,3,2,2,2,2,2,3,3,3 25

↓ 28

3,2,2,2,2,2,3,3,3,3 25

↓ 28

2,2,2,2,2,3,3,3,3,3 25

as desired.

The final example we are going to look at is when our starting vertex w is the vertex $(5, 0, 1, 0, 5, 5, 4, 2, 0, 3)$, then we proceed as follows as in the previous example:

5,0,1,0,5,5,4,2,0,3 25

↓ 28

0,1,0,5,5,4,2,0,3,3 23

↓ 26

1,0,5,5,4,2,0,3,3,3 26

↓ 29

0,5,5,4,2,0,3,3,3,3 28

↓ 28

5,5,4,2,0,3,3,3,3,0 28

↓ 30

5,4,2,0,3,3,3,3,0,2 25

↓ 27

4,2,0,3,3,3,3,0,2,2 22

↓ 26

2,0,3,3,3,3,0,2,2,4 22

↓ 25

0,3,3,3,3,0,2,2,4,3 23

↓ 25

3,3,3,3,0,2,2,4,3,2 25

↓ 28

3,3,3,0,2,2,4,3,2,3 25

↓ 28

3,3,0,2,2,4,3,2,3,3 25

↓ 28

3,0,2,2,4,3,2,3,3,3 25

↓ 28

0,2,2,4,3,2,3,3,3,3 25

↓ 27

2,2,4,3,2,3,3,3,3,2 27

↓ 29

2,4,3,2,3,3,3,3,2,2 27

↓ 29

4,3,2,3,3,3,3,2,2,2 27

↓ 29

3,2,3,3,3,3,3,2,2,2,2 25

↓ 27

2,3,3,3,3,3,2,2,2,2,2 24

↓ 27

3,3,3,3,2,2,2,2,2,3 25

↓ 28

3,3,3,2,2,2,2,2,3,3 25

↓ 28

3,3,2,2,2,2,2,3,3,3 25

↓ 28

3,2,2,2,2,2,3,3,3,3 25

↓ 28

2,2,2,2,2,3,3,3,3,3 25

as desired. Clearly, we can see that there exists a path between three chosen vertices and our sv. We need to prove this in general.

Proof of Theorem 5: If $s = t = 0$, then trivially a *u-cycle* exists. Since $s < t$ where $t \geq s + (d-1)$, it follows that if s is positive so is t. Now let V be the set of all vertices of length $n-1$ and weight w, where $s-(d-1) \leq s \leq t$ over a alphabet $[d] = \{0, 1, 2, ..., d-1\}$ and let $D =< V, E >$ be a digraph defined as follows. Suppose we let $v_1 = a_1a_2...a_{n-1}$ and $v_2 = b_1b_2...b_{n-1}$ which are both in V. A directed edge

is drawn from $v_1 \to v_2$ if the last $n-2$ letters of v_1 are the same as the first $n-2$ letters of v_2 and the edge labeled with the corresponding concatenated *n-letter* word of weight w satisfies $s-(d-1) \leq 2 \leq t$. If we let A be the set of all *n-letter* words, then the *Eulerian* circuit in D will determine the existence of a universal cycle for A. And once again the proof reduces to showing that D is *Eulerian*.

Suppose we let $v_1 = a_1 a_2 ... a_{n-1} \in V$ where

$$\sum_{i=1}^{n-1} a_i = w.$$

If $s \leq w < t$, then we will have edges (words) coming in and out of v_1 for every vertex weight $x \geq 0$, where $x \in [d]$, such that $s \leq w + x \leq t$, hence $i(v) = o(v)$. If $w = t$ then it is only possible to add a weight 0 since adding anything else to v_1 will make the edge (word) greater than t which is illegal. Hence $i(v) = o(v) = 1$. If $s-(d-1) \leq w < s$, then the valid edges (words) will be those corresponding value of x such that $s \leq w + x \leq t$, and hence $i(v) = o(v) \geq 1$. Therefore, $i(v) = o(v)$ for all vertices in the vertex set V. Establishing weak connectedness is no trivial step. However, we will prove weak connectedness using Lemmas $6-9$ below.

Lemma 6 *Any vertex v of weight equal to s, can be taken to v' = (x,x,x,x,...,x+1,x+1,...,x+1), where the numbers of $x's$ and $x+1's$ are correct, but they do not necessarily have to be in the right order.*

Proof: The claim is that once we have a vertex of weight equal to s, then we can reduce it to the desired number of weight $x's$ and $(x+1)'s$. First, we figure out what our target or sink vertex is. We then know that the weight of our vertices has to be between $s-(d-1) \leq |v| \leq t$ and the weight of our edges has to be between

$s \leq |e| \leq t$. Since we are starting with a vertex of weight equal to s, then our vertex consists of $a_i \in \{0, 1, 2, ..., d-1\}$ where the $\sum a_i = |v| = s$. Recall that vertices are of length $n-1$.

Case 1: Suppose our vertex is of weight equal to s and does not contain weights of $x's$ or $(x+1)'s$ or both. Then, the ultimate goal is to end up at a vertex that has the desired number of weights $x's$ and $(x+1)'s$. If the first element of our vertex is greater than weight $(x+1)$, then we can cycle and add a weight of $(x+1)$. Adding a weight of $(x+1)$ decreases our vertex weight and increases our edge weight while still keeping our edge and vertex weights legal. If this new vertex weight is within x of the maximum vertex weight, then we cycle until we get to a weight greater than $(x+1)$ and add weight x. We repeat until we get to a weight less than x. If we get to a weight less than x, then we can cycle and add weight x as long as our vertex and edge weights are legal. Also, if we get to a weight less than x but the weight of our previous vertex is within x of the maximum vertex weight, then we need to cycle which keeps our vertex and edge weights legal. Again, we repeat until we get to a weight greater than $(x+1)$ (where the first part of this paragraph already took care of this case). However, on the other hand, if our vertex weight is within x of the minimum vertex weight, then we need to add a weight of $(x+1)$ which will increase our vertex and edge weight while still keeping them legal. And we repeat all of the above until we have our desired number of weight $x's$ and $(x+1)'s$ which does not have to be in order.

Case 2: If our starting vertex v does have weights $x's$ or $(x+1)'s$ or both then whenever we get to these weights, we cycle which keeps our vertex and edge weights

legal. If we get to a vertex that is within x of the minimum vertex weight, then we need to add a weight of $(x+1)$ to keep our vertex and edge weights legal. On the other hand, if our vertex is within x of the maximum vertex weight, then we need to add a weight of x which keeps both edge and vertex weights legal. If we get to a vertex greater than $(x+1)$, and we are within our x of our minimum vertex weight, then we need to cycle until we get to a weight less than x while still keeping our edge and vertex weights legal. Furthermore, if we get to a weight greater than $(x+1)$, and we are within x of the maximum vertex weight, then we need to cycle and add weight x which still keeps our edge and vertex weights legal. We repeat until we have our vertex of desired weights $x's$ and $(x+1)'s$. There may we cases where we have a vertex with more of weight x than $(x+1)$. We rectify this by cycling until we get to a weight of x and then we cycle and add a weight of $(x+1)$. We repeat until we have our desired number of weights $x's$ and $(x+1)$. On the other hand, if we have a vertex with more of weight $(x+1)'s$ than $x's$, we rectify this by cycling until we get to a weight of $(x+1)$ and then we cycle and add a weight of x. Finally, we repeat until we have our desired number of weights $x's$ and $(x+1)$. □

The idea is if we start with a vertex of weight greater than s or less than s, then we can get it to a vertex of weight s. After getting it to a vertex of weight s, then we know that we can get our desired number of weights $x's$ and $(x+1)'s$ not necessarilly in order. Lemmas 7 and 8 deal with vertices of this kind and the proofs will follow and Lemma 9 deals with getting the weights $x's$ and $(x+1)'s$ in order.

Lemma 7 *Any vertex v of weight $\geq s+1$ can be reduced to one with a weight s.*

Proof: Suppose our vertex $v_1 = a_1a_2a_3...a_{n-1}$ and $a_1 = 0$, then we cycle and we get $v_2 = a_2a_3...a_{n-1}a_1$. Notice that this cycling does not change the vertex or the edge weight. However, if $a_1 \geq 1$, then we set $v_2 = a_2a_3...a_{n-1}(a_1 - 1)$. Notice that $|v_2| = |v_1|-1$ and the weight of the edge $|e_1| = a_1+a_2+a_3+...+a_{n-1}+(a_1-1)$. If $|e_1|$ is not legal, then we have to add the minimum letter that will keep our edge weight legal while still reducing our vertex weight. Worse case scenario is, if $|v_i| = |e_i| = t$, then we cycle and add a weight of 0. Now our vertex $v_{i+1} = a_{r+1}...a_{n-1}a_r$, where $a_r = 0$ and v_{i+1} has weight $t - a_r \geq t - (d-1) \geq s$ where $r \leq d-1$. If $|v_i| = t - r$, then we add any weight x that reduces the vertex weight to some weight greater than or equal to s while still keeping the edge weight legal. However, if we get to some vertex $v_i = a_1a_2a_3...a_{n-1}$, where $|v_i| = s+1$ and $a_1 \neq 0$, then our $v_{i+1} = a_2a_3...a_{n-1}(a_1 - 1)$ and $|v_{i+1}| = s$. The weight of our edge $|e_i| = s+1+(a_1-1) \leq s+1+d-2 = s+d-1 \leq t$, where $a_r \leq d-1$ and our edge weight is legal. □

Lemma 8 *Any vertex v of weight $\leq s-1$ can be increased to one with a weight s.*

Proof: The proof of this lemma is very similar to the previous lemma, but instead of reducing the vertex weight, we are increasing the vertex weight to s. Suppose $|v_1| = a_1a_2...a_{n-1} \leq s-1$ where $a_1 \leq d-1$, then we cycle and $v_2 = a_2a_3...a_{n-1}(a_1+1)$ which increases our vertex weight. Now our edge weight $|e_1| = |v_1| + (a_1 + 1) \leq |v_1| + d - 1 + 1 \leq s - 1 + d - 1 + 1 = s - 1 + d \leq t$.

If $a_1 = d-1$, then $v_2 = a_2a_3...a_{n-1}(d-1)$ and $|v_1| = |v_2|$ which is legal. Our edge weight $|e_1| = |v_1| + (d-1) \leq t$. Sooner or later there will exist an $a_j \leq d-2$ which

will take a vertex v, $|v| \leq s-1$. Finally, if $v_i = a_1a_2...a_{n-1}$, where $|v_i| = s-1$ and $a_1 \neq 0$. Then $v_{i+1} = a_2a_3...a_{n-1}(a_1+1)$ where $|v_{i+1}| = s$. The edge weight $|e_i| = |v_i| + a_1 + 1 \leq |v_i| + d - 1 + 1 \leq s - 1 + d - 1 + 1 = s - +d \leq t$. Therefore, the edge weight is legal and the weight of our vertex is s. □

Lemma 9 *Any vertex v of weight s with the desired number of weights $x's$ and $x+1's$ (not necessarily in order) can be arranged in order.*

Proof: The proof is very straight forward. Since we have our vertex weight of s and our desired number of weight $x's$ and $(x+1)'s$, then we can cycle until they are in order. If $v_1 = a_1a_2...a_{n-1}$ where $a_1 = x$ and $a_{n-1} = x+1$, then we cycle and add weight $(x+1)$ and our $v_2 = a_2a_3...a_{n-1}(x+1)$. And we repeat until we have our desired number of weight $(x+1)'s$. Then we cycle and add our weight $x's$ to the end. Notice that our edge and vertex weight may increase but still remain legal. Finally we cycle until our weight $x's$ and $(x+1)'s$ are in order. If $v_1 = a_1a_2...a_{n-1}$ where $a_1 = x+1$ and $a_{n-1} = x$, then we cycle and add weight x and our $v_2 = a_2a_3...a_{n-1}x$. And we repeat until we have our desired number of weight $x's$. Then we cycle and add our weight $(x+1)'s$ to the end. Notice that our edge and vertex weights are still legal even though they might increase or decrease. □

Therefore, given any vertex in D there exists a path p such that we can get to our target or sink vertex. This establishes weak connectedness, and this completes the proof of Theorem 5. □

4 CONCLUDING REMARKS

We showed that given any modular *k-letter* alphabet, there exists *u-cycles* for *M-Lipschitz n-letter* word if condition C holds and all assignment of the elements of $[n]$ to the sets in an ordered subposet of the Boolean Lattice whose Hasse Diagram has any given shape. We also showed that there exists a *u-cycle* of all words with weight between s and t.

For future research, it would be interesting to improve Theorem 5 so that the range of the weight of the word is as small as possible, or else to prove that the range in Theorem 5 is the best possible. Also, one might ask how and to what extent one can show the existence of de Bruijn cycles for unordered posets. Last but not least, can results be proved for (labelled as well as unlabelled) subposets of mother posets other than the Boolean Lattice?

BIBLIOGRAPHY

[1] Miklos Bona (2002). "*A Walk Through Combinatorics, An Introduction to Enumeration and Graph Theory*," Published by World Scientific Co. pte. Ltd.

[2] Gary Chartrand, Linda Lesniak, Ping Zhang (2011). "*Graphs & Digraphs*," 5th edition, Published CRC Press.

[3] A. Leitner and A. P. Godbole (2010). *"Universal Cycles of Classes of Restricted Words,"* to appear in *Discrete Math.*

[4] A. Blanca and A. P. Godbole (2011). *"On Universal Cycles for New Classes of Combinatorial Structures," SIAM J. Discrete Math.* **25**, 1832-1842.

[5] G. Hulbert (1994). *"On Universal Cycles for k-subsets of an n-set," SIAM J. Discrete Math.* **7**, 598-604.

[6] G. Hulbert, T. Johnson, and J. Zahl (2009). *"On Universal Cycles for Multisets," Discrete Math.* **309**, 5321-5327.

[7] A. Bechel, B. LaBounty-Lay and A. P. Godbole (2008). *"Universal Cycles of Discrete Functions," Congressum Numerantium.* **189**, 121-128.

[8] F. Chung, P. Diaconis and R. Graham (1992). *"Universal Cycles for Combinatorial Structures," Discrete Math.* **110**, 43-59.

[9] B. Jackson (1993). *"Universal Cycles of k-subsets and k-permutations," Discrete Math.* **117**, 114-150.

VITA

ANDRÉ ALEXANDER CAMPBELL

Education:	B.A. Mathematics and Computer Science, Tusculum College, Greeneville, Tennessee, 2009 M.S. Mathematics, East Tennessee State University (ETSU), Johnson City, Tennessee, 2013
Professional Experience:	Teaching Assistant, East Tennessee State University, Johnson City, Tennessee, 2012–present Graduate Assistant, East Tennessee State University University, Johnson City, Tennessee, 2011–2012
Publications:	A. Campbell, A. Godbole and B. Kay, "*Contributions to the Theory of de Bruijn Cycles*," Submitted for Publication, April 2013
Honor Societies:	Kappa Mu Epsilon