RENN, CHARLES FREDERICK, M.A. On Union-Closed Families. (2007)
Directed by Dr. Theresa Vaughan. 73pp.

A union-closed family is a non-empty finite collection of non-empty sets, $\mathcal{F}$, such that for any $A, B \in \mathcal{F}$, then $A \cup B \in \mathcal{F}$ [7]. Peter Frankl conjectured in 1979 that if $\mathcal{F}$ is union-closed, then there exists an element α such that α occurs in at least half of the sets of $\mathcal{F}$ (cf. [6]). Despite its simplicity, the conjecture has defied any general proof. In [6], Bjorn Poonen proved the result for families involving up to 7 elements, and only marginal improvement has been made on this bound.

In this paper we attempt to approach the above conjecture from a broad perspective. We begin by imposing a numbering on all possible subsets, or *collections*, of the power set on n elements. From this, we investigate relationships between the numbering of a given collection and whether or not it is union-closed. We also look for a pattern in the distribution of union-closed families within the numbering for $n \leq 5$. For this work, we use a complete listing of the union-closed families. We make use of several custom computer applications written in C++ to produce complete union-closed family listings for $n = 3, 4, 5$.

ON UNION-CLOSED FAMILIES

by

Charles Frederick Renn

A Thesis Submitted to
the Faculty of The Graduate School at
The University of North Carolina at Greensboro
in Partial Fulfillment
of the Requirements for the Degree
Master of Arts

Greensboro
2007

Approved by

Committee Chair

Dedicated to
my grandfathers,
Harland Frederick Duerk, and
Charles Francis Renn

APPROVAL PAGE

This thesis has been approved by the following committee of the
Faculty of The Graduate School at The University of North Carolina at Greensboro.

Committee Chair ________________________________

Committee Members ________________________________

____

____
Date of Acceptance by Committee

____
Date of Final Oral Examination

ACKNOWLEDGMENTS

I would like to thank the members of my committee, Sebastian Pauli and Paul Duvall, for their open doors and their willingness to listen and comment on my ideas throughout the development of this thesis. I would also like to make a special thank you to my advisor, Theresa Vaughan, who has consistently supported me throughout this process and who simply encouraged me to "play".

I would also like to thank my employer, the National Board for Certified Counselors, for their flexibility with my erratic student schedule and for their continued support over the past two years. In particular, I would like to thank my colleagues, Adrian Goulbourne and Bob Henegar, who patiently endured many math conversations and were always willing to offer a fresh perspective.

Lastly, I would like to thank my wife, Karen. Without her continuous love and understanding, this thesis would not have been possible.

TABLE OF CONTENTS

LIST OF FIGURES

Page

CHAPTER I
INTRODUCTION

A union-closed family is a non-empty finite collection of non-empty sets, $\mathcal{F}$, such that for any $A, B \in \mathcal{F}$, then $A \cup B \in \mathcal{F}$ [7]. Peter Frankl conjectured in 1979 that if $\mathcal{F}$ is union-closed, then there exists an element α such that α occurs in at least half of the sets of $\mathcal{F}$ (cf. [6]). Despite its simplicity, the conjecture has defied any general proof. In [6], Bjorn Poonen proved the result for families involving up to 7 elements. This result has been extended by Morris for families on 9 elements [5]. Gao and Yu have proved the result for families where $|\mathcal{F}| \leq 32$ [2].

In this paper we attempt to approach the above conjecture from a broad perspective. We begin by imposing a numbering on all possible subsets, or *collections*, of the power set on n elements. From this, we investigate relationships between the numbering of a given collection and whether or not it is union-closed. We also look for a pattern in the distribution of union-closed families within the numbering for $n \leq 5$. For this work, we use a complete listing of the union-closed families. We make use of several custom computer applications written in C++ to produce complete union-closed family listings for $n = 3, 4, 5$.

Chapter 2 of this paper defines many of the terms that will be used throughout the paper. This chapter also describes the numbering scheme mentioned above and a natural equivalence relation on these collections. The problem of finding a unique representative for the resultant equivalence classes is discussed. In Chapter 3 we present some consequences and results of this numbering. We provide data related to the distribution of families and corresponding graphical representations.

In Chapter 4 we apply counting theorems of Burnside and Polya to the problem of counting union-closed families for a given n.

We introduce the concept of *squashed ordering* in Chapter 5. This provides a secondary numbering on the collections, and we investigate an application of this squashed order to possibly identify non-isomorphic collections that are union-closed.

Union-closed families have a direct representation as a semilattice, as described by Poonen [6]. In Chapter 6 we briefly discuss this approach and provide several interesting examples. We also discuss the graph-generating algorithm used by the computer program. Lastly, Chapter 7 presents some notes and comments regarding the computer applications used throughout this paper.

CHAPTER II
DEFINITIONS AND NOTATION

2.1 Preliminary Definitions

Let $[n] = \{1, 2, 3, \ldots, n\}$. Throughout this paper we will be considering subsets of the power set on n elements excluding the empty set (i.e. $\mathcal{P}([n]) - \{\emptyset\}$, denoted $\mathcal{P}^*([n])$). Such subsets are called *collections*.

Definition 2.1 *A* ***collection in n***, $\mathcal{A}$, *is a non-empty subset of the set* $\mathcal{P}^*([n])$. *Often, such an object is referred to simply as a* ***collection***. *A collection in* n $\mathcal{A}$ *is called* ***proper*** *if* $\mathcal{A} \not\subset \mathcal{P}([n-1])$. *If a collection is not proper, we say it is* ***non-proper***.

We use the convention of script capitals to refer to collections, and Roman capitals to refer to subsets of $[n]$. To represent the cardinality of a collection, $\mathcal{A}$, we use the usual notation, $|\mathcal{A}|$.

Definition 2.2 *A* ***union-closed family*** *(or, simply* ***family****) is a collection* $\mathcal{F}$ *with the following property: for any* $A, B \in \mathcal{F}$, $A \cup B \in \mathcal{F}$. *That is,* $\mathcal{F}$ *is closed under unions. For a given* n*, the maximal family is* $\mathcal{P}^*([n])$.

Later, we will have need to consider sets of collections in a given n, and we will use the following definition.

Definition 2.3 *The* ***n-supercollection*** *is the set of all possible non-empty collections in* n*, which is equivalent to* $\mathcal{P}(\mathcal{P}^*([n])) - \{\emptyset\}$. *A* $\boldsymbol{(n, i)}$***-supercollection*** *is the set of all possible collections in* n *with cardinality* i.

2.2 Collection Numbering

In this section, we define a numbering, based on the binary expansion of integers, on an n-supercollection. This will allow for numerical manipulation of collections, including the ability to represent collections as a single point on the number line. In addition, our numbering acts as an encoding of the collection itself, so all information about the structure of the collection is contained within its number. This provides us with a method of efficiently handling these collections programmatically.

We begin with defining a numbering on the subsets of $[n]$. Let A be a subset of $[n]$ and let α be a function from $\mathcal{P}^*([n])$ to the set of sequences of length n, with $\alpha(A) = (a_n, a_{n-1}, \ldots, a_2, a_1)$. We define its members in the following way.

$$a_i = \begin{cases} 1 & i \in A \\ 0 & i \notin A \end{cases} \quad \text{for } i \in \{1, 2, 3, \ldots, n\} \tag{2.1}$$

If we write $\alpha(A)$ without commas and parentheses, we obtain what may be interpreted as an integer in binary representation. This number, which we will write $\#A$, is assigned to the set.

Example 2.4 *Let* $A = \{1, 2, 4, 6\}$*. Then,* $\alpha(A) = 101011$*, and* $\#A = 43$*.*

Because of unique binary representation for integers, this numbering is well-defined and reversible. For completeness, we note the following.

Corollary 2.5 *Let A be a subset of $[n]$. The* ***set number*** *for A is*

$$\#A = \sum_{i \in A} 2^{i-1}$$

Consequently, $\#A \in \{1, 2, 3, \ldots, 2^n - 1\}$.

We can extend this process to uniquely assign numbers to collections in an obvious way. We now consider sequences of length $2^n - 1$, where each element

b_i corresponds to the set with set number i. Given a collection $\mathcal{A}$, we define a function β from all collections in n to sequences of length $2^n - 1$. So we have $\beta(\mathcal{A}) = (b_{2^n-1}, b_{2^n-2}, \ldots, b_1)$ where

$$b_i = \begin{cases} 1 & i = \#A \text{ for some } A \in \mathcal{A} \\ 0 & i \neq \#A \text{ for some } A \in \mathcal{A} \end{cases} \quad \text{for } i \in \{1, 2, 3, \ldots, 2^n - 1\} \tag{2.2}$$

Again, we write $\beta(\mathcal{A})$ without commas and parentheses and interpret it as an integer in binary representation. This integer is the number assigned to the collection. We will write $\#\mathcal{A}$ for this number.

Example 2.6 *Let $\mathcal{A} = \{\{1\}, \{1,2\}, \{3\}, \{1,2,3\}\} \subset \mathcal{P}^*([3])$. First, calculating the set numbers for the member sets we obtain*

$$\begin{aligned} \#\{1\} &= 1 \\ \#\{1,2\} &= 3 \\ \#\{3\} &= 4 \\ \#\{1,2,3\} &= 7 \end{aligned}$$

This yields $\beta(\mathcal{A}) = 1001101$ and $\#\mathcal{A} = 77$.

As before, we note

Corollary 2.7 *Let $\mathcal{A}$ be a subset of $\mathcal{P}^*([n])$. The* ***collection number*** *for $\mathcal{A}$ is*

$$\#\mathcal{A} = \sum_{A \in \mathcal{A}} 2^{\#A - 1}$$

Consequently, $\#\mathcal{A} \in \{1, 2, 3, \ldots, 2^{2^n - 1} - 1\}$.

2.3 Equivalence Classes on Collections

Consider the following collections.

$$\mathcal{F}_1 = \{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}, \text{ and}$$

$$\mathcal{F}_2 = \{\{2\}, \{1,2\}, \{2,3\}, \{1,2,3\}\}$$

Although these families are visibly distinct families and have different collection numbers, they are *structurally* identical. Indeed, if the symbols '1' and '2' are interchanged in $\mathcal{F}_1$, we obtain $\mathcal{F}_2$. In general, any permutation of the symbols '1', '2', and '3' would yield a structurally identical family, preserving the property of union-closedness. We use this idea to define an equivalence relation on the set of collections in n.

First, we refine our notion of a permutation acting on a collection in n. Let $\pi \in S_n$, where S_n is the symmetric group on n elements. This permutation induces a permutation π^* of the set $\mathcal{P}^*([n])$ in the obvious way. We define the action of π^* on a set $A \in \mathcal{P}^*([n])$,

$$\pi^*(A) = \{\pi(i) \mid i \in A\}$$

Let S_n^* represent the set of all permutations induced on $\mathcal{P}^*([n])$ by S_n. If $\mathcal{F}$ is collection in n, we can abuse the notation slightly, and write what we mean by the permutation $\pi^*(\mathcal{F})$,

$$\pi^*(\mathcal{F}) = \{\pi^*(A) \mid A \in \mathcal{F}\}$$

Definition 2.8 *Let $\mathcal{A}$ and $\mathcal{B}$ be collections in n. We say that $\mathcal{A}$ is* ***permutation-equivalent*** *to $\mathcal{B}$ if and only if there exists a $\pi \in S_n$ such that $\pi^*(\mathcal{A}) = \mathcal{B}$. We write $\mathcal{A} \sim_P \mathcal{B}$.*

It is easy to check that permutation-equivalence is an equivalence relation on the set of collections. We denote the permutation-equivalence class of a collection $\mathcal{A}$

by $[\mathcal{A}]_P$. It is clear that if $\mathcal{A}$ and $\mathcal{B}$ are permutation-equivalent, then they are structurally identical, i.e. isomorphic.

Definition 2.9 *Let $\mathcal{B}$ be a collection in n. The* ***closure of*** *$\mathcal{B}$ is the collection formed by the union of $\mathcal{B}$ with the set of all possible unions of the member sets of $\mathcal{B}$. We write the closure of $\mathcal{B}$ as $\overline{\mathcal{B}}$.*

$$\overline{\mathcal{B}} = \mathcal{B} \cup \{\bigcup_{i=1}^{m} X_i | X_i \in \mathcal{B}, m \in \mathbb{Z}^+\}$$

Consider the following two collections,

$$\mathcal{F} = \{\{1\}, \{2\}, \{1,2\}, \{1,2,3\}\}, \text{ and}$$

$$\mathcal{G} = \{\{1\}, \{2\}, \{1,2,3\}\}$$

Clearly $\mathcal{F} \not\sim_P \mathcal{G}$, as $|\mathcal{F}| \neq |\mathcal{G}|$. However, we see that $\overline{\mathcal{G}} = \mathcal{F}$. That is, $\mathcal{F}$ is the closure of $\mathcal{G}$. This leads us to our next notion of equivalence.

Definition 2.10 *Let $\mathcal{A}$ and $\mathcal{B}$ be collections in n. We say that $\mathcal{A}$ is* ***union-closed equivalent*** *to $\mathcal{B}$ if and only if the closure of $\mathcal{A}$ equals the closure of $\mathcal{B}$. We write $\mathcal{A} \sim_U \mathcal{B}$.*

The relation of union-closed equivalence is also an equivalence relation on the set of collections, and we denote the equivalence class for the collection $\mathcal{A}$ by $[\mathcal{A}]_U$.

It is clear that many different collections may have the same closure, and are therefore in the same class with respect to union-closed equivalence. In an equivalence class, there exists a collection whose cardinality is less than or equal to the cardinality of any other collection in the equivalence class. We call this collection the **minimal generating collection (MGC)** [4], and it has the obvious property that none of its members are equal to unions of smaller sets that are still members of the collection. Johnson and Vaughan proved the uniqueness of the MGC for a given equivalence class in [4].

We follow with the following observation. Let $\mathcal{A}$ be a collection in n and $\pi \in S_n$. Then,

$$\overline{\pi^*(\mathcal{A})} = \pi^*(\overline{\mathcal{A}})$$

From this fact, we can combine the ideas of permutation equivalence and union-closed equivalence and realize that

$$[[\mathcal{A}]_P]_U = [[\mathcal{A}]_U]_P$$

We realize that in this case the subscripts are irrelevant, and we omit them, writing $[[\mathcal{A}]]$ instead. We say that $[[\mathcal{A}]]$ is the **UC-class for $\mathcal{A}$**. These classes define a useful partition on collections. If two union-closed families are not in the same UC-class, then the two families must be structurally distinct (ie. non-isomorphic). Consequently, counting these classes will yield the number of distinct union-closed families for a given n.

2.4 Two Counterexamples

When dealing with these equivalence classes, it would be instructive to have methods of characterizing each class based on the structure of the union-closed family. For example, if each class had a unique vector of cardinalities for its member sets, identifying union-closed equivalent collections and obtaining an inventory of non-isomorphic classes would be simpler. However, as the following counterexample shows, several possible characterizations prove to be not unique.

Example 2.11 *Let $\mathcal{A}$ and $\mathcal{B}$ be the collections in 4 below, $\#\mathcal{A} = 29856$ and $\#\mathcal{B} =$*

31746.

$$\mathcal{A} = \{\{4\}, \{2,3\}, \{1,2,4\}, \{1,3,4\}, \{2,3,4\}, \{1,2,3,4\}\} \text{ and}$$

$$\mathcal{B} = \{\{2\}, \{3,4\}, \{1,2,4\}, \{1,3,4\}, \{2,3,4\}, \{1,2,3,4\}\}$$

Let $\boldsymbol{e}$ *be the sequence where* e_i *equals the number of occurrences of the element* i*, and let* $\boldsymbol{c}$ *be the sequence where* c_i *equals the number of member sets of cardinality* i*. For both* $\mathcal{A}$ *and* $\mathcal{B}$*, we have the following:*

$$\boldsymbol{e} = (3,4,4,5)$$

$$\boldsymbol{c} = (1,1,3,1)$$

However, $\mathcal{A}$ *and* $\mathcal{B}$ *are not isomorphic. It is easiest to see this using a graphical representation for each collection, where nodes (member sets) are joined by edges indicating containment. We will discuss this representation further in Chapter 6, but provide the graphs for* $\mathcal{A}$ *and* $\mathcal{B}$ *here.*

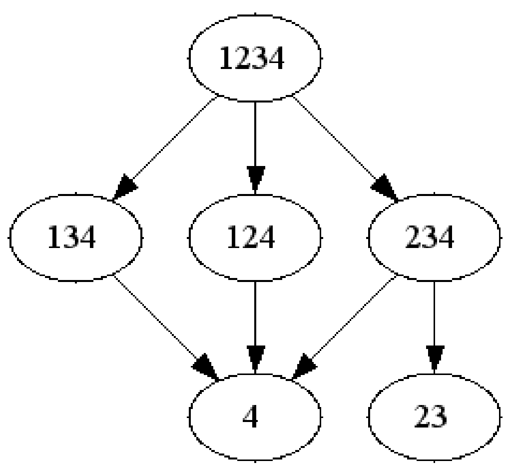

Figure 2.1: The collection $\mathcal{A}$, $\#\mathcal{A} = 29856$.

Another possible characterization uses the idea of graph ***levels****. The level of a node is the length of the minimal path from itself to the top node (i.e. the set* $[n]$*).*

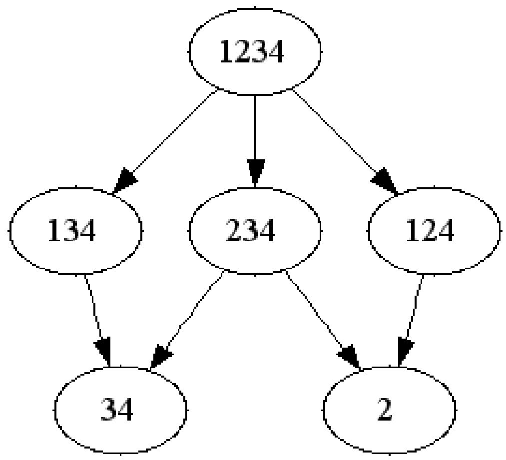

Figure 2.2: The collection $\mathcal{B}$, $\#\mathcal{B} = 31746$.

If we let $\boldsymbol{l}$ *be the sequence where* l_i *equals the number of sets at level* i, $i \geq 0$, *then we see both collections above have* $\boldsymbol{l} = (1, 3, 2)$, *disproving the uniqueness of* l.

From the previous example, we cannot characterize the isomorphism classes uniquely based on these aspects of the family's structure. Instead, we often will take the maximal collection number of all collections in an isomorphism class as the representative element.

By taking the maximal collection number for a given class, we know the corresponding collection must be union-closed. It is tempting to attempt to use the fact that the collection number is maximal to show that this family possesses an element occurring in at least half of the sets of the family, which is the conjecture of Frankl mentioned earlier. Moreover, because of the maximal nature of this number, we may think that the element n would be the element to satify the conjecture. However, the following counterexample when $n = 5$ proves otherwise. Interestingly, no similar counterexample can be found for $n < 5$.

Example 2.12 *Consider the collection* $\mathcal{A}$ *with* $\#\mathcal{A} = 1946157178$ *and the graph structure below. It can be shown that the collection number* $\#\mathcal{A}$ *is maximal for*

$[[\mathcal{A}]]$. *We can see that* $\boldsymbol{e}=(5,6,7,4,4)$ *and* $|\mathcal{A}|=9$. *In particular, although the*

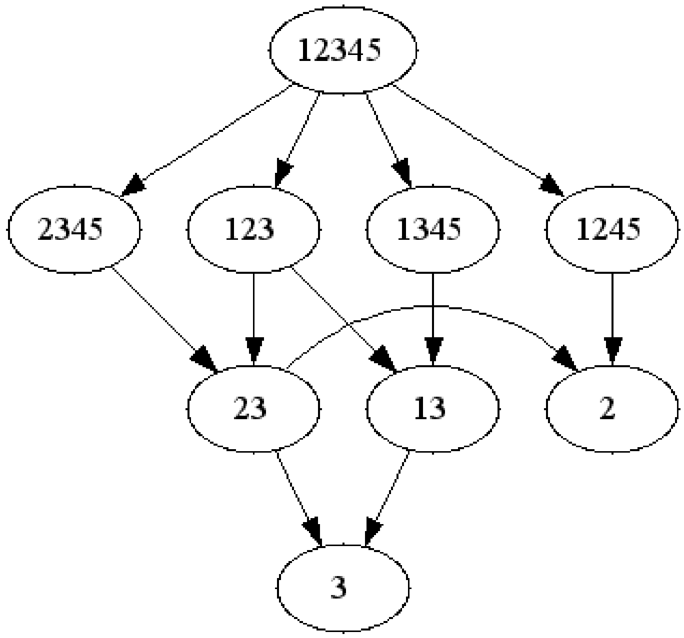

Figure 2.3: The collection $\mathcal{A}$ with $\#\mathcal{A}=1946157178$.

family does satify the Frankl conjecture, neither element 4 *or* 5 *occur in at least half of the sets of* $\mathcal{A}$.

CHAPTER III
USING COLLECTION NUMBERS

In this chapter, we investigate patterns and consequences that arise from the binary-based numbering on collections introduced in the last chapter. We are interested in methods by which a collection's union-closedness could be determined from only its collection number.

3.1 Some Easy Lemmas

We begin with several simple results about the collection numbering.

Lemma 3.1 *A collection number is odd if and only if the collection contains the singleton set* $\{1\}$*.*

Proof: A collection number is odd if and only if $b_1 = 1$. This requires that the set with set number 1 be in the collection, and for any n the set with number 1 is the singleton $\{1\}$. □

Lemma 3.2 *Let* $\mathcal{A}$ *be a collection in* n*. If* $\#\mathcal{A}$ *is odd, then* $\#\overline{\mathcal{A}}$ *is odd.*

Proof: By Lemma 3.1, $\{1\} \in \mathcal{A}$, and therefore $\{1\} \in \overline{\mathcal{A}}$. Hence, $\#\overline{\mathcal{A}}$ is odd. □

Theorem 3.3 *Let* $\mathcal{A}$ *be a collection in* n*. If* $\#\mathcal{B}$ *is odd for all* $\mathcal{B} \in [\mathcal{A}]_P$*, then* $\mathcal{A}$ *contains all singletons in* $\mathcal{P}([n])$*. The closure of* $\mathcal{A}$ *is the family* $\mathcal{P}^*([n])$*, and* $[[A]]$ *is UC-class for* $\mathcal{P}^*([n])$*.*

Proof: Assume $\#\mathcal{B}$ is odd for all $\mathcal{B} \in [\mathcal{A}]_P$. There exists a permutation π in S_n such that $\pi^*(\mathcal{A}) = \mathcal{B}$. In particular, π maps i to 1 for some $i \in [n]$. Because $\#\mathcal{B}$ is odd, it contains $\{1\}$ by Lemma 3.1, and therefore $(\pi^*)^{-1}(\{1\}) = \{i\} \in \mathcal{A}$. Since $\mathcal{B}$ is arbitrary, we have that $\mathcal{A}$ contains all singletons $\{i\}$ for $i \in [n]$. □

Corollary 3.4 *Let $\mathcal{A}$ be a collection in n. If $\#\mathcal{B}$ is even for all $\mathcal{B} \in [\mathcal{A}]_P$ then $\mathcal{A}$ contains none of the singletons in $\mathcal{P}^*([n])$.*

Here we have a result relating the preservation of a number's parity with the structure of a collection. Unfortunately, this verification is not efficient, since the work required to verify all permutations of a given collection is comparable to simply calculating the closure of the collection.

From the results above, we realize that for some collections, $[\mathcal{A}]_P = \{\mathcal{A}\}$. That is, all permutations of $\mathcal{A}$ are $\mathcal{A}$ itself. We characterize this fact in the following theorem.

Theorem 3.5 *Let $\pi \in S_n$ and $\mathcal{A}$ be a collection in n. Then $\pi^*(\mathcal{A}) = \mathcal{A}$ for all $\pi \in S_n$ if and only if, if $\mathcal{A}$ contains a set of cardinality i, then $\mathcal{A}$ contains all possible sets of cardinality i.*

Proof: Suppose $\pi^*(\mathcal{A}) = \mathcal{A}$ for all $\pi \in S_n$, and suppose $U \in \mathcal{A}$, $|U| = i$. As in Theorem 3.3, for a set $V \in \mathcal{P}^*([n])$ with $|V| = i$, there exists a permutation π such that $\pi^*(V) = U$. Therefore, by the hypothesis, $(\pi^*)^{-1}(U) = V \in \mathcal{A}$. Since V was arbitrary, $\mathcal{A}$ contains all possible sets of cardinality i. The converse is obvious. □

Corollary 3.6 *Let $\mathcal{A}$ be a collection in n and let $\pi \in S_n$. The collection $\mathcal{A}$ is union-closed and $\pi^*(\mathcal{A}) = \mathcal{A}$ if and only if $\mathcal{A}$ consists of all sets in $[n]$ with cardinality greater than or equal to k, for some fixed k.*

3.2 Special Collections

Within a n-supercollection, there are several collections which have special significance. We have already defined $\mathcal{P}^*([n])$, which is the collection containing every possible non-empty set. Clearly, this collection is maximal and union-closed, and its collection number is maximal, that is,

$$\#\mathcal{P}^*([n]) = 2^{2^n-1} - 1$$

We introduce two more special collections here.

Definition 3.7 *The* ***singleton collection in n****, denoted* $\mathrm{Sgl}(n)$*, is the collection containing only the singletons in* $\mathcal{P}^*([n])$*. It is the minimal generating collection for the family* $\mathcal{P}^*([n])$*.*

Because $\mathrm{Sgl}(n)$ is the minimal generating collection for $\mathcal{P}^*([n])$, a collection in n is proper only if its collection number is greater than $\#\mathrm{Sgl}(n)$. We provide a lemma about the collection number of $\mathrm{Sgl}(n)$ below. It is a direct consequence of the collection numbering, and we leave the proof to the reader.

Lemma 3.8 *The collection number of* $\mathrm{Sgl}(n)$ *can be calculated by the following recursive formula.*

$$\#\mathrm{Sgl}(n) = \#\mathrm{Sgl}(n-1) + \#\mathcal{P}^*([n-1]) + 1$$

Definition 3.9 *The* ***halfway collection in n****, denoted* $\mathrm{Half}(n)$*, is the collection containing only the set* $[n]$*. It has the property that its collection number is half of one more than the total number of collections:*

$$\#\mathrm{Half}(n) = \frac{\#\mathcal{P}^*([n]) + 1}{2} = 2^{2^n-2}$$

The binary nature of the numbering leads to the fact that any collection with a number greater than $\#\text{Half}(n)$ must include the set $[n]$. Thus, the collections 'repeat' in the sense that any collection in the latter half of the collection listing has exactly the same structure as a set in the first half of the listing, save for the inclusion of the set $[n]$. We state this fact slightly differently in the following lemma.

Lemma 3.10 *If $\mathcal{A}$ is a proper collection in n with $\#\mathcal{A} < \#\text{Half}(n)$, then*

$$\mathcal{A} \sim_U (\mathcal{A} \cup \text{Half}(n))$$

The sum of these facts allows us qualitatively describe the UC-classes for families in n. Moreover, are able to define numerical ranges where collection numbers for families must be. First, a family is either proper or non-proper. Indeed, for every collection number $\#\mathcal{A}$, $1 \leq \#\mathcal{A} < \#\text{Sgl}(n))$, $[[\mathcal{A}]]$ is an non-proper family class, and these are the UC-classes for collections in $n - 1$.

Now, given a proper family, it is either *degenerate* or *normal*. A degenerate family is identical to a non-proper family, with the inclusion of the set $[n]$. If $\#\mathcal{A}$, $\#\text{Half}(n) < \#\mathcal{A} < \#\text{Half}(n) + \#\text{Sgl}(n))$, then $[[\mathcal{A}]]$ is a degenerate family class. (Note that because we do not allow empty collections, $\text{Half}(n)$ is not degenerate, but nonetheless trivial.)

Every other family is considered normal. By Lemma 3.10, we observe that every normal UC-class has a representative with a collection number greater than $\#\text{Sgl}(n) + \#\text{Half}(n)$.

3.3 Summary of Results and Graphical Representations

By the results of the previous section, we obtain the number of isomorphism classes of union-closed families by using a computer to iterate through collection numbers from $\text{Half}(n)$ to $\mathcal{P}^*([n])$. We present the results for $1 \leq n \leq 5$ in Figure 3.1 below, where N_p is the number of proper union-closed family classes and N is the total number of union-closed family classes.

n	N_p	N
1	1	1
2	3	4
3	14	18
4	165	183
5	14480	14663

Figure 3.1: Counts of isomorphism classes for $1 \leq n \leq 5$.

With these counts, we wish to graph histograms showing the distribution of proper UC-classes with respect to the cardinality of the family (ie. the number of UC-classes of cardinality i). For a given cardinality i, there are $\binom{2^n-1}{i}$ possible collections. This is of course a symmetric, normal distribution, but it may be surprising that the distribution of families is also near-normal. The following Figures 3.2, 3.3, and 3.4 show the distributions for $n = 3, 4$ and 5, respectively. The numerical data for these figures is given in Appendix A.

Let the set C_n be the set of all collection numbers for the n-supercollection, and then consider the function $f_n : C_n \mapsto C_n$ where $f(\#\mathcal{A}) = \#\overline{\mathcal{A}}$. That is, each collection number is mapped to the collection number of its closure. It is interesting to notice that the resulting graph is very reminiscent of the well-known Sierpinski triangle. We provide graphs f_3 and f_4 in Figures 3.5 and 3.6. Although the graphs appear to have points on the same vertical line in numerous places, this is due to the compression of the horizontal scale.

A complete picture of f_5 would require the plotting of over two billion points and so is not included. However, we can reduce the number of points plotted by plotting every i^{th} collection. Setting $i = 15199$ (which is prime), we obtain a partial graph of f_5 containing 141291 points, shown in Figure 3.7. The Sierpinski pattern is again clearly visible.

In all the graphs f_n, the horizontal translation symmetry is very strong. Each graph has two triangles in the top half of the graph, and each time the right-hand triangle is exactly the left-hand triangle shifted by $\#\text{Half}(n)$. This is exactly for the reason given in Lemma 3.10. By the same idea, we can explain all horizontal translation symmetries.

We end this chapter with some expanded graphs for f_4. As noted above, although the graph appears to have vertical components, it is in fact a function. And, when expanded, the horizontal translation becomes more apparent and the graph could be said to be locally periodic. Two expanded portions of f_4 are provided in Figures 3.8 and 3.9.

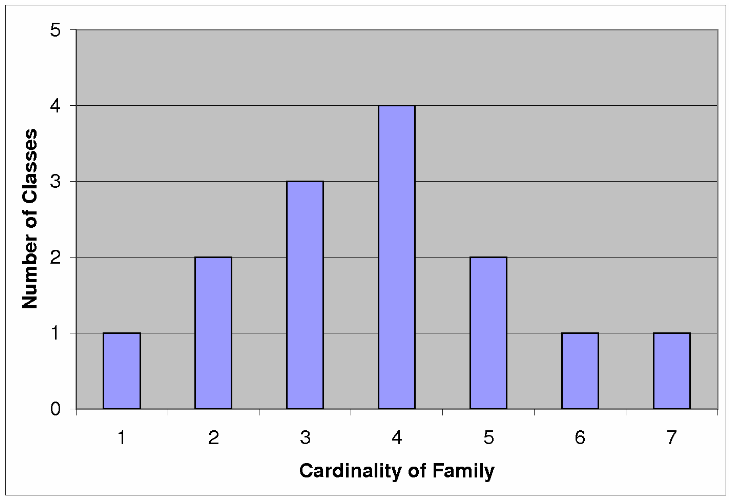

Figure 3.2: Distribution of Classes by Cardinality for $n = 3$.

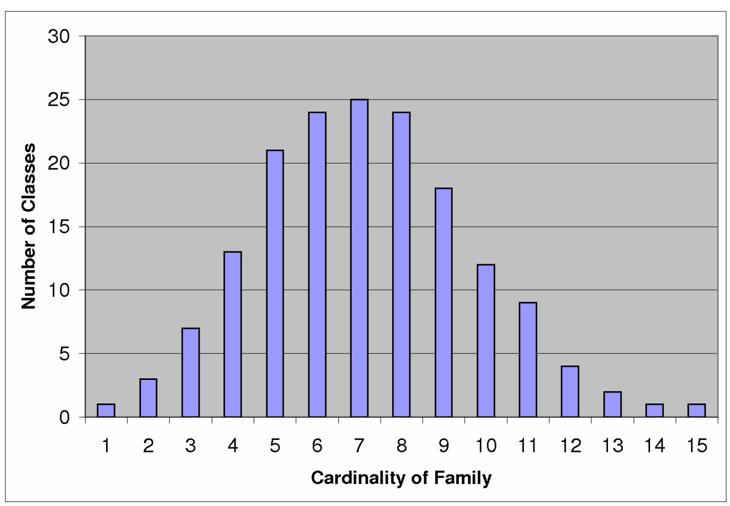

Figure 3.3: Distribution of Classes by Cardinality for $n = 4$.

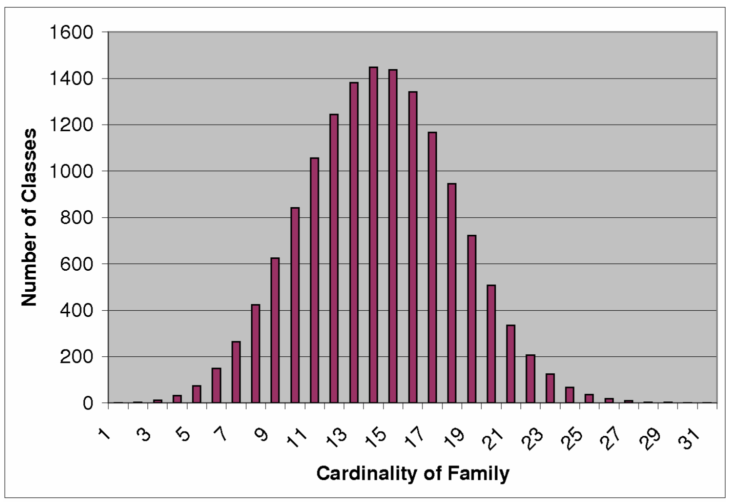

Figure 3.4: Distribution of Classes by Cardinality for $n = 5$.

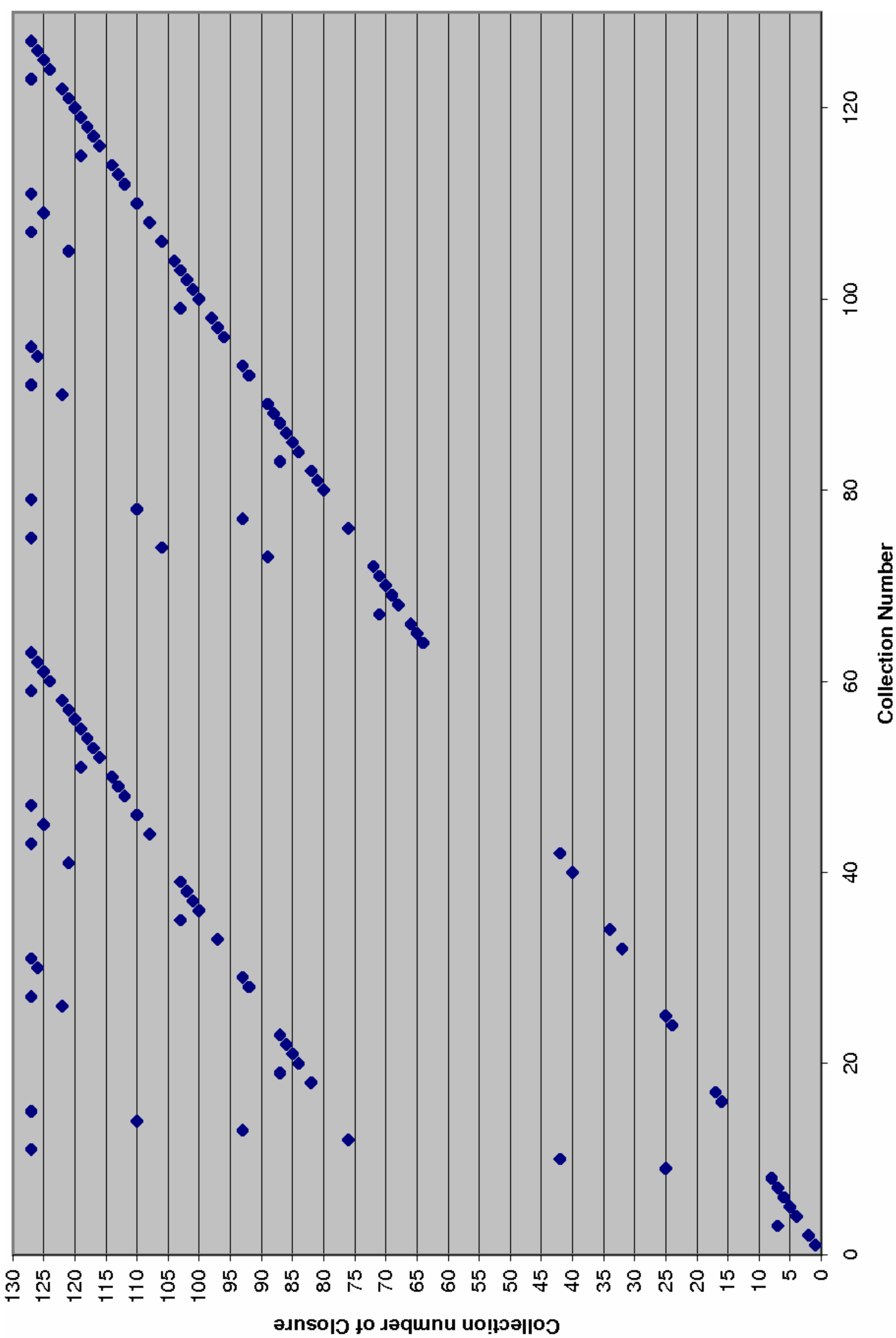

Figure 3.5: The plot of f_3.

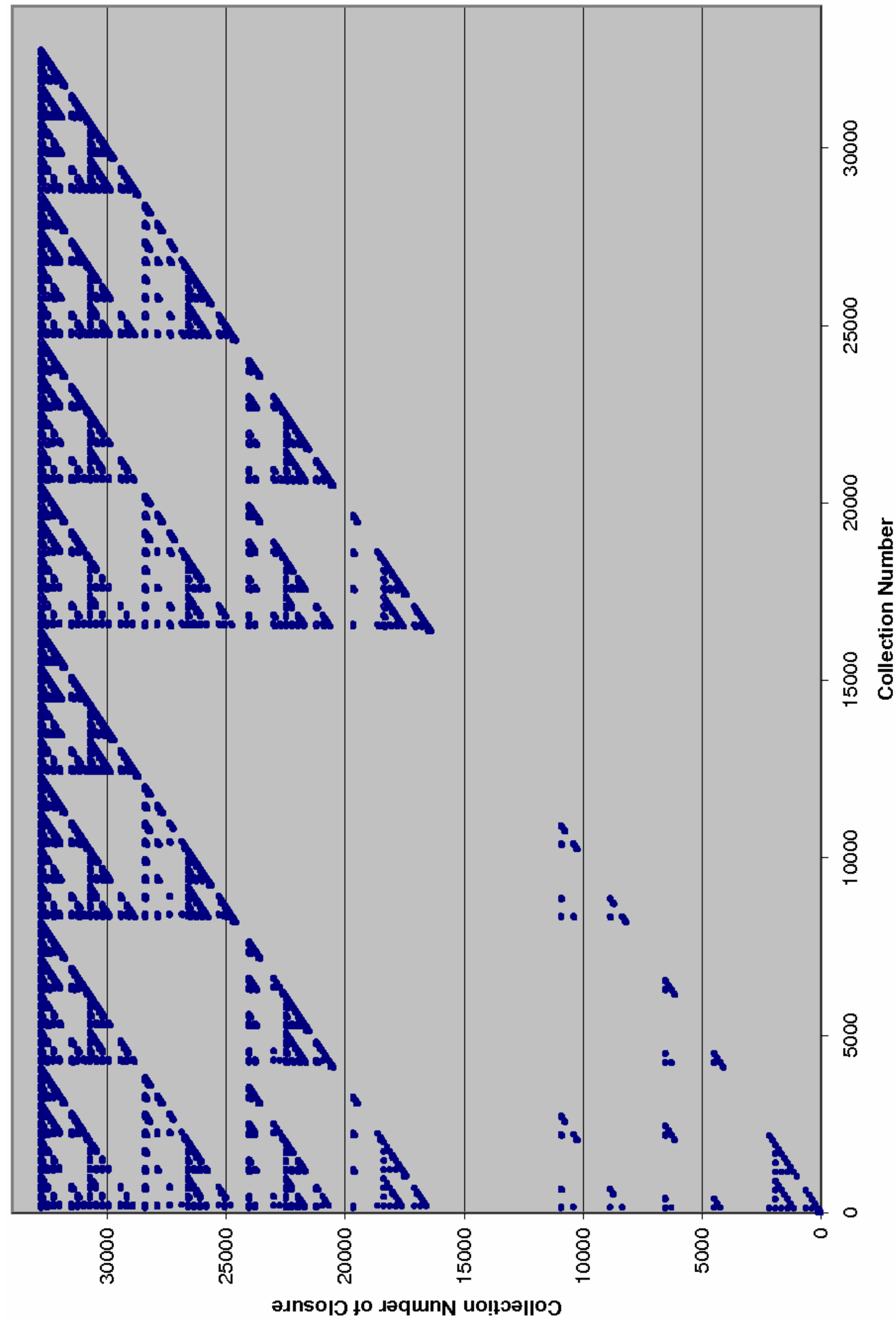

Figure 3.6: The plot of f_4.

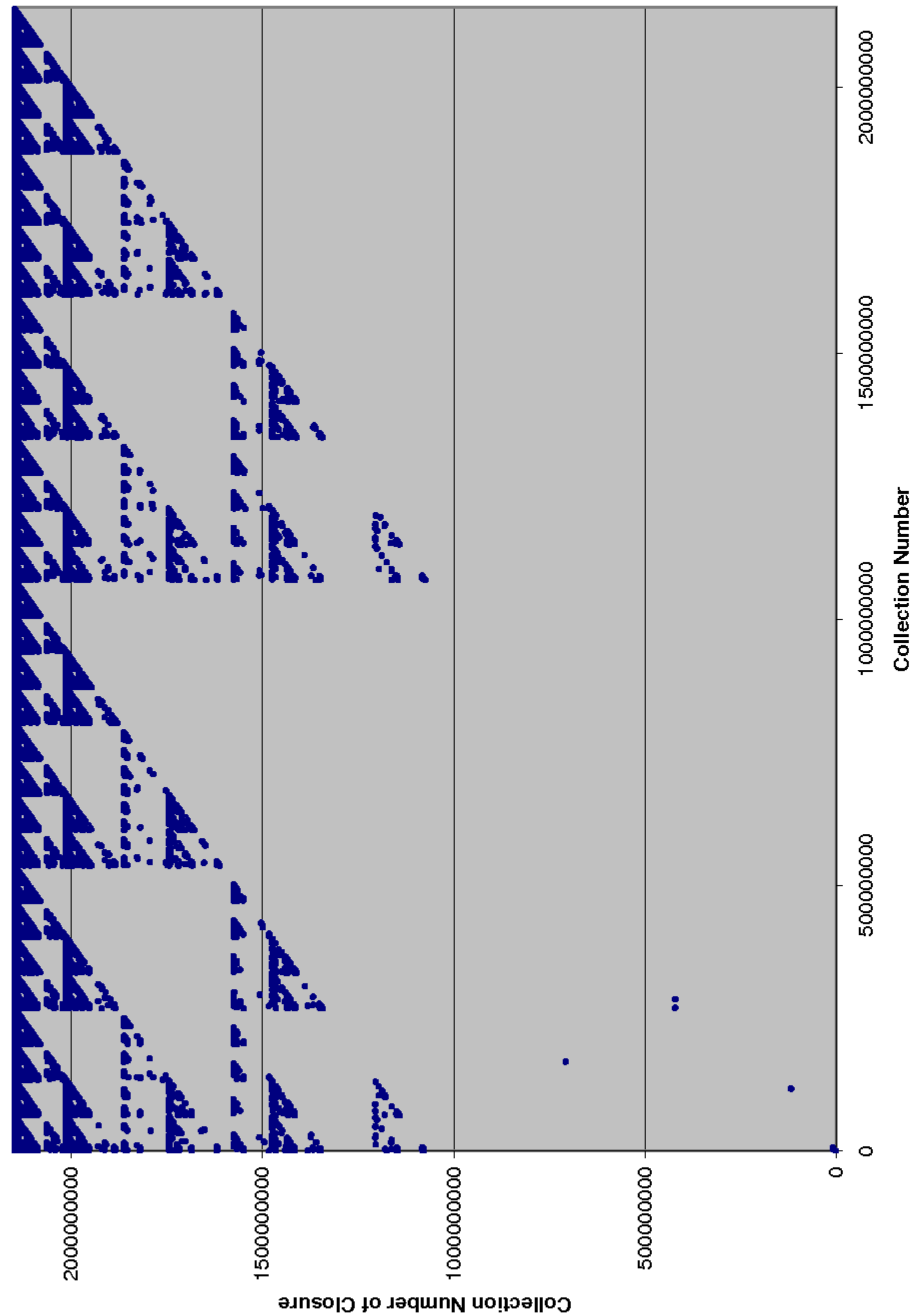

Figure 3.7: The plot of every 15199^{th} collection for f_5.

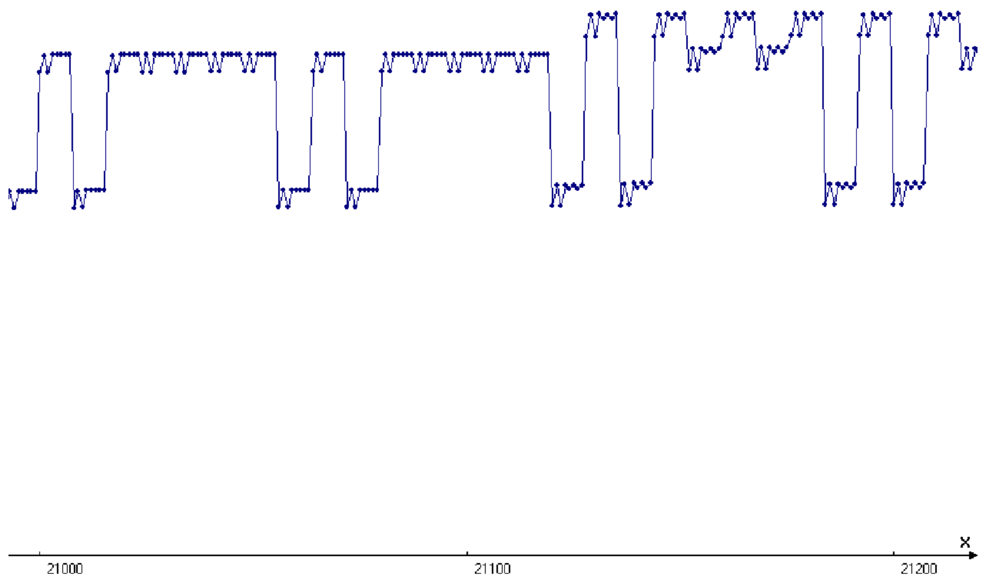

Figure 3.8: An expanded section of the graph of f_4

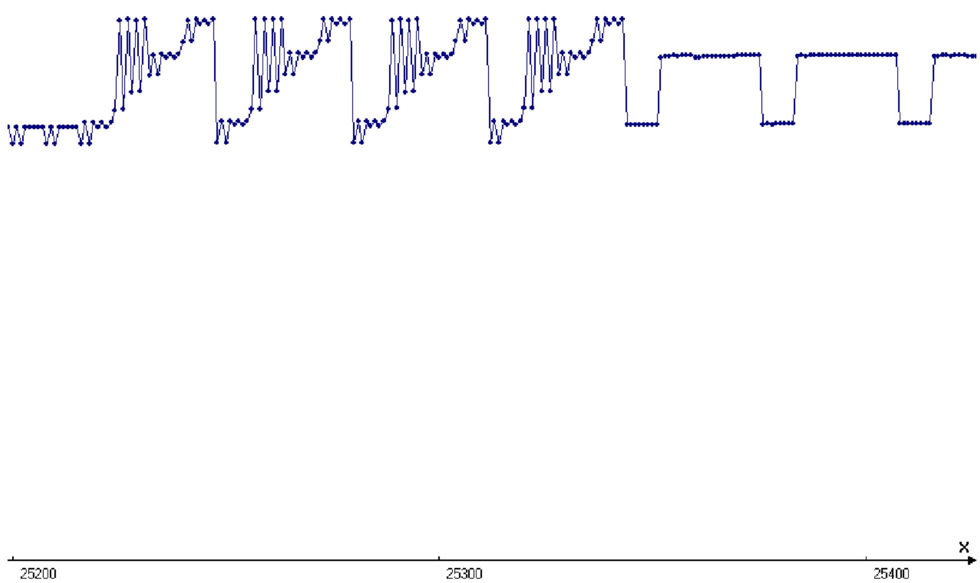

Figure 3.9: Another expanded section of the graph of f_4

CHAPTER IV
USING THEOREMS OF BURNSIDE AND POLYA

The previous chapter was focused on counting UC-classes. Here, we consider permutation-equivalence classes and ask, for a given n how many permutation equivalence classes of proper collections are there? This gives an upper bound on the number of UC-classes. We attempt to answer this question by considering 2-colorings on $\mathcal{P}([n])$ and by using the results of Burnside and Polya in [9].

4.1 Definitions and Notation

Let $\mathcal{F}$ be a proper collection in n. Let $\mathcal{C} = \{b, w\}$ be our set of colors ('black' and 'white'). We define the coloring $c_{\mathcal{F}} : \mathcal{P}([n]) \mapsto \mathcal{C}$ by the function

$$c(A) = \begin{cases} b & A \in \mathcal{F} \\ w & A \notin \mathcal{F} \end{cases}$$

Thus, b represents inclusion in $\mathcal{F}$, and that by our definitions $c_{\mathcal{F}}([n])$ is always b.

Example 4.1 *Consider the following collections.*

$$\mathcal{F}_1 = \{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}, \text{ and}$$

$$\mathcal{F}_2 = \{\{2\}, \{1,2\}, \{2,3\}, \{1,2,3\}\}$$

The families $\mathcal{F}_1$ and $\mathcal{F}_2$ are shown with their colorings in Figure 4.1, and it is clear that $\mathcal{F}_1 \sim_P \mathcal{F}_2$.

We observe that for any $A \in \mathcal{P}^*([n])$, $|\pi^*(A)| = |A|$, that is, π^* preserves cardinality. In terms of the graph representation of families, we say that π^* "preserves levels". With this in mind, we realize that each permutation in S_n induces

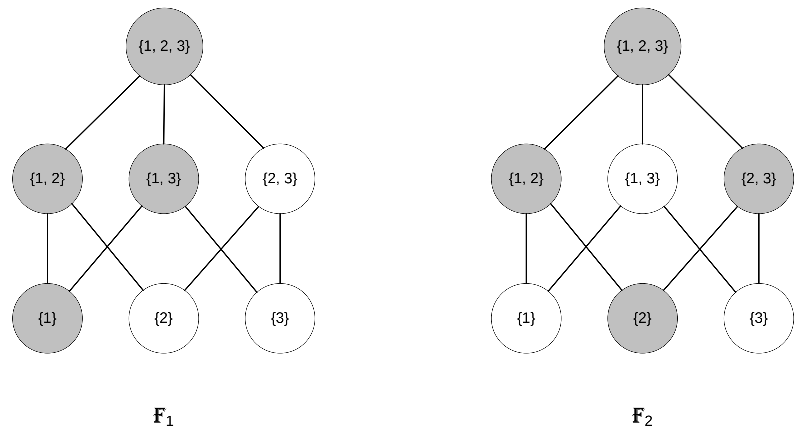

Figure 4.1: Colorings of $\mathcal{F}_1$ and $\mathcal{F}_2$.

a permutation of the $\binom{n}{k}$ k-sets in $\mathcal{P}^*([n])$. For a given $\pi \in S_n$, we define $\pi^{(k)}$ as a restriction of π^* to only those sets of cardinality k, and we let S_n^k be the set of all permutations $\pi^{(k)}$. We note that $S_n = S_n^{(1)}$.

We use the notation $Z(G)$ to refer to the cycle index for a group G of symmetries on a set. Similiarly. for a single permutation π, $Z(\pi)$ will represent the cycle index.

Now we are able to ask the question from the introduction in terms of colorings: What is the number of equivalence classes of 2-colorings on the set $\mathcal{P}^*([n]) - [n]$ induced by the group S_n^*? This is well-suited to the application of results from Burnside and Polya.

4.2 Calculations

We first answer the above question when $n = 3$, and then hope to generalize for greater n. Because we will consider the set $[n]$ to always be colored black and we ignore the empty set, we have $\binom{3}{1}$ singletons and $\binom{3}{2}$ doubletons for a total of 6 sets

$\pi \in S_3$	$Z(\pi)$	$Z(\pi^{(2)})$	$Z(\pi^*)$
(1)(2)(3)	x_1^3	x_1^3	x_1^6
(12)(3) (13)(2) (23)(1)	x_1x_2	x_1x_2	$x_1^2x_2^2$
(123) (132)	x_3	x_3	x_3^2

Figure 4.2: All elements of S_3 and their cycle indices.

to consider. Thus, we have 64 families (2^6) to consider when $n = 3$.

Consider the case of colorings containing one singleton, A, and one doubleton, B. Either $A \subset B$ or $A \not\subset B$. For any $\pi^* \in S_n^*$, if $A \subset B$, then $\pi^*(A) \subset \pi^*(B)$. Hence, containment is preserved by π^*, and it is sufficient to consider only these two cases ($A \subset B$ or $A \not\subset B$). By such logic, we deduce all possible colorings and present them in the chart in Figure 4.4. There are 20 distinct colorings.

We now approach this problem from the direction of Burnside's Lemma. There are six permutations in S_3, and we list them in Figure 4.2 with their corresponding cycle structure.

From this, we can write the cycle index for S_3^*

$$Z(S_3^*) = \frac{1}{6}\left[x_1^6 + 3x_1^2x_2^2 + 2x_3^2\right]$$

By inspection, we note that $Z(S_3) = Z(S_3^{(2)})$ and

$$Z(S_3) = Z(S_3^{(2)}) = \frac{1}{6}\left[x_1^3 + 3x_1x_2 + 2x_3\right]$$

The reason for this is clear when we note that as S_3 permutes the singletons, the complementary sets are also permuted in the same fashion. In general, we have the following lemma.

Lemma 4.2

$$Z(S_n^{(k)}) = Z(S_n^{(n-k)}) \textit{ for } i \in \{1, 2, 3, \ldots, n-1\}$$

Using Burnside's Lemma, we can calculate N, the number of distinct 2-colorings.

$$\begin{aligned} N &= \frac{1}{6}\left[2^6 + 3(2^4) + 2(2^2)\right] \\ &= \frac{1}{6}[120] \\ &= 20 \end{aligned}$$

This agrees with our work in Figure 4.4.

By the following substitution, $x_i = (b^i + w^i)$, we can invoke Polya's theorem and obtain a pattern inventory for the 2-colorings.

$$\begin{aligned} Z(S_3^*) &= \frac{1}{6}\left[(b+w)^6 + 3(b+w)^2(b^2+w^2)^2 + 2(b^3+w^3)^2\right] \\ &= b^6 + 2b^5w + 4b^4w^2 + 6b^3w^3 + 4b^2w^4 + 2bw^5 + w^6 \end{aligned}$$

Because our families are designated by the set colored black, we can weight $w = 1$ and obtain a clearer inventory of families,

$$b^6 + 2b^5 + 4b^4 + 6b^3 + 4b^2 + 2b + 1$$

Here the exponent of b is the cardinality of the family, not including the set $[n]$. Again, this result coincides with the results on our chart. However, in the above inventory we lose information about how many sets from each level are colored.

In summary, of the possible 64 families with $n = 3$, only 20 distinct 2-colorings exist. Of these families, only 14 are union-closed.

The use of Burnside's Lemma and Polya's theorem for $n > 3$ requires us to find the cycle index $Z(S_n^*)$. Since this induced permuation preserves the permutations at each level, we can decompose π^* as follows,

$$\pi^* = \prod_{i=1}^{n-1} Z(\pi^{(i)})$$

From this, we can write the cycle index for $Z(S_n^*)$,

$$Z(S_n^*) = \frac{1}{n!} \sum_{\pi \in S_n} \left(\prod_{i=1}^{n-1} Z(\pi^{(i)}) \right) \tag{4.1}$$

The remaining task is to calculate the cycle indices of the induced permutations $\pi^{(i)}$. However, this is not a straightforward calculation. As we will show, it is possible to compute the cycle index for some $S_n^{(i)}$. But, in order to make use of (4.1), we need to know the particular $\pi^{(i)}$ an individual permutation π induces. This information is lost when we write $Z(S_n^{(i)})$ due to the collapsing of terms within the expression.

It is tempting at first glance to attempt to "reverse engineer" the cycle index $Z(S_n^{(i)})$ in an effort to extract information about the permutations $\pi^{(i)}$. However, even when $i = 2$, the expression for the induced cycle index is extremely complicated, and it is very likely that the cycle indices for $i > 2$ are even more difficult - if expressible at all.

In [3], Harary provides general formulas for $Z(S_n)$ and $Z(S_n^{(2)})$. For the symmetric group S_n, we have

$$Z(S_n) = \frac{1}{n!} \sum_{(\mathbf{j})} \frac{n!}{\prod_{k=1}^{n} k^{j_k} j_k!} a_1^{j_1} a_2^{j_2} ... a_n^{j_n} \tag{4.2}$$

where the sum runs over the set of solution vectors $\mathbf{j} = (j_1, j_2, \ldots, j_n)$ to the equation

$$1j_1 + 2j_2 + \ldots + nj_n = n \tag{4.3}$$

It is clear that the vector $\mathbf{j}$ describes the cycle decomposition of a permutation in S_n and the component j_i is the number of i-cycles in the cycle decomposition. There is a interesting recursion involved with equation (4.3) that, with the definition $Z(S_0) = 1$, leads to the following recursive definition of $Z(S_n)$ (from [12])

$$Z(S_n) = \frac{1}{n} \sum_{k=1}^{n} x_k Z(S_{n-k})$$

Harary also provides a formula for the cycle index of $S_n^{(2)}$ which he names the *pair group* [3]. We include it here to demonstrate the mounting complexity of these cycle indices.

$$Z(S_n^{(2)}) = \frac{1}{n!} \sum_{(\mathbf{j})} \frac{n!}{\prod_{k=1}^{n} k^{j_k} j_k!} \prod_{k=1}^{\lfloor n/2 \rfloor} (a_k a_{2k}^{k-1})^{j_{2k}} a_k^{k\binom{j_k}{2}} \prod_{k=0}^{\lfloor (n-1)/2 \rfloor} a_{2k+1}^{kj_{2k+1}} \prod_{1 \le r < s \le n-1} a_{LCM(r,s)}^{GCD(r,s) j_r j_s}$$

A complete listing of cycle indices for $3 \le n \le 6$ can be found in Appendix B. By substituting 2 in for each x_i in the $Z(S_n^*)$ formulas, we can obtain the number of distinct 2-colorings on the set $\mathcal{P}([n]) - \{\emptyset, [n]\}$. This is also the number of permutation equivalence classes for collections in n, but each class may or may not represent a union-closed family.

4.3 Summary of Results

We begin this section by summarizing the results of our computations. In the following table, $P = 2^{|\mathcal{P}([n]) - \{\emptyset, [n]\}|}$, N is the number of equivalence classes of 2-colorings, and N_{UC} is the number of equivalence classes that represent a union-closed family.

n	$\|P\|$	N	N_{UC}	N/P	N_{UC}/N
2	4	3	3	0.75	1
3	64	20	14	0.3125	0.7
4	16384	996	165	0.06079	0.16566
5	1073741824	9333312	14480	0.008692	0.0015514
6	4611686018427390000	6406604874137940	?	0.001389	?

Figure 4.3: Calculations for $2 \leq n \leq 6$.

By observing the values for N/P we see that considering only distinct colorings of the power set greatly reduces the number of cases to verify. However, as mentioned before, we lose information regarding how many sets at each level are colored for each distinct coloring, and this makes any organized iteration through the distinct coloring impossible. In addition, despite the fact that $N \ll P$, the magnitude of N is still very large and prohibitive.

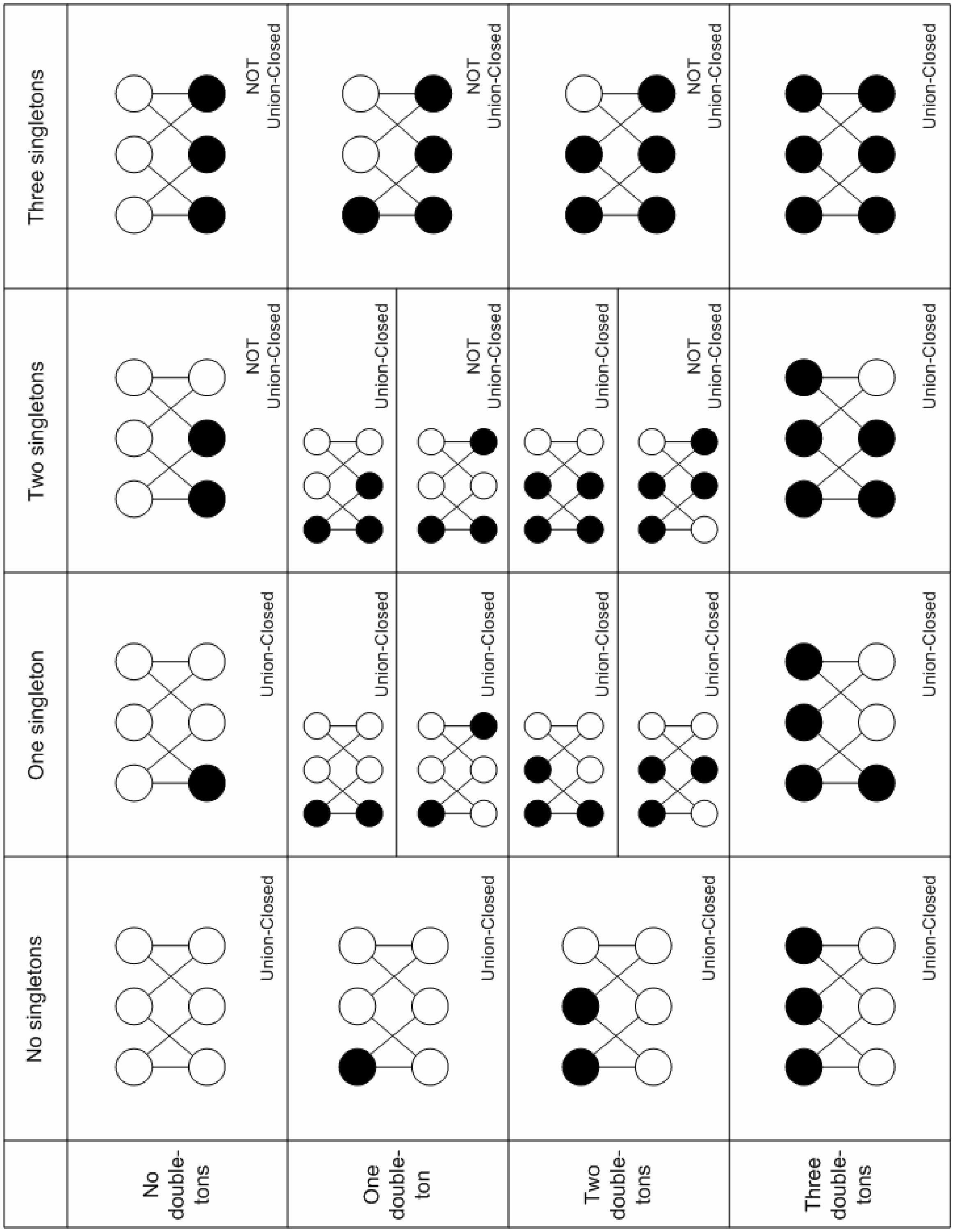

Figure 4.4: All 2-colorings of the set $\mathcal{P}^*[3] - \{1, 2, 3\}$.

CHAPTER V

SQUASHED ORDERING AND SUPERCOLLECTIONS

Up to this point, we have made use of collection numbers as an ordering on an n-supercollection, and with Lemma 3.10 we attempted to identify intervals containing at least one member of each UC-class. In this chapter we will refine our process by only considering (n, i)-supercollections. This is a natural refinement because all permutations of a family in n with cardinality i must necessarily be entirely contained within the (n, i)-supercollection. To this end, we use the *squashed ordering* of collections and some related results (see Anderson [1]).

5.1 Definition of Squashed Ordering

Let A and B be subsets of $[n]$ with cardinality i. Let $A \diamond B = (A \cap B') \cup (A' \cap B)$, where $A' = [n] - A$. The expression $A \diamond B$ is also known as the symmetric difference. Using this, Anderson provides us with the following two technical definitions for orderings on the i-sets A and B of $[n]$.

Definition 5.1 (Lexicographic Order) *We say $A <_L B$ if the smallest element of $A \diamond B$ is in A.*

Definition 5.2 (Squashed Order) *We say $A <_S B$ if the largest element of $A \diamond B$ is in B.*

The technical definitions given above are not intuitively clear. Most readers will conceptualize the lexicographic ordering as a 'dictionary' ordering where the

Lexicographic $<_L$	*Squashed* $<_S$
123	123
124	124
125	134
134	234
135	125
145	135
234	235
235	145
245	245
345	345

Figure 5.1: The 3-sets in $\{1, 2, 3, 4, 5\}$ in lexicographic and squashed order.

'alphabet' is replaced by the positive integers, and we agree to write sets in the increasing element order. The intuition behind the squashed ordering is less obvious. For comparison, we provide a listing of the 3-sets in $\{1, 2, 3, 4, 5\}$ according to both orderings in Figure 5.1.

The pattern behind the squashed order becomes clearer when we agree to write the sets in *decreasing, or reverse*, element order. In fact, we will see that it will be advantageous to represent a set A using the $\beta(A)$ defined in Chapter 2. For both the reverse notation and the β notation, we see in Figure 5.2 that the squashed order is a lexicographic ordering of these alternate notations.

With $n = 5$, there are $\binom{5}{3}$ 3-sets. Reading down from the top in Figure 5.2, the set $\{1, 2, 5\}$ is in the 5^{th} squashed order position, for example. We will make use of the following theorem [1, Theorem 7.2.1].

Theorem 5.3 *Given positive integers m and l, there exists a unique representation*

Set A	Reverse of A	$\beta(A)$
123	321	00111
124	421	01011
134	431	01101
234	432	01110
125	521	10011
135	531	10101
235	532	10110
145	541	11001
245	542	11010
345	543	11100

Figure 5.2: The 3-sets in $\{1,2,3,4,5\}$ in the squashed order.

of m in the form

$$m = \binom{a_l}{l} + \binom{a_{l-1}}{l-1} + \cdots + \binom{a_t}{t}$$

*where $a_l > a_{l-1} > \ldots > a_t \geq t \geq 1$. This representation of m is called the **l-binomial representation of** m.*

Choosing m to be the squashed order position and l to be the cardinality of the sets, we will see that this binomial representation allows us to recover the set itself. Consider the set $\{3,4,5\}$ which is the last set, in the 10^{th} position. With $m = 10$ and $l = 3$, we have $10 = \binom{5}{3}$, or, it is the last subset of 5 elements taken 3 at a time.

Recall the earlier example with set $B = \{1,2,5\}$, which is in the 5^{th} position. Taking $m = 5$ and $l = 3$, this time we obtain $5 = \binom{4}{3} + \binom{2}{2}$. Inspecting Figure 5.2 again, we see that B is preceded by the $\binom{4}{3}$ sets that have a leading 0 in the β listing, and it is the first set with a leading 1 - hence the addition of $\binom{2}{2}$.

This combinatorial interpretation of the binomial representation forms the basis of a reconstruction algorithm for the set when given the squashed order position and the set's cardinality.

Algorithm 5.4 *1. Let m be the squashed order position for a set B, $|B| = l$.*

2. Choose a_l so that $\binom{a_l}{l} \leq m$ and $\binom{a_l+1}{l} > m$.

3. Put $m_1 = m - \binom{a_l}{l}$. If $m_1 = 0$, then we are done. Otherwise, choose a_{l-1} so that, as before, $\binom{a_{l-1}}{l-1} \leq m_1$ and $\binom{a_{l-1}+1}{l-1} > m$.

4. Continue until $m_i = 0$. We now have the l-binomial representation of m as in Theorem 5.3, including the sequence $(a_l, a_{l-1}, \ldots, a_t)$.

5. Using the above sequence, we reconstruct the set B as

$$\{a_l + 1, a_{l-1} + 1, \ldots, a_t, a_t - 1, \ldots, a_t - (t-1)\}$$

That is, add 1 to all a_k, except for a_t, and this will provide a partial list of elements in the set B. The remaining elements of B are the t descending, consecutive integers beginning with a_t.

Example 5.5 *Let us reconstruct the 6^{th} 3-set of [5]. Setting $m = 6$ and $l = 3$, we obtain the l-representation of m:*

$$m = 6 = \binom{4}{3} + \binom{2}{2} + \binom{1}{1}$$

From the algorithm, the set must be $\{5, 3, 1\}$, or $\{1, 3, 5\}$.

5.2 Application of Squashed Order to Supercollections

We will apply the squashed ordering to (n, i)-supercollections, and we will be interested in the position of a collection in the squashed order. We begin with the following definition.

Definition 5.6 *Let $\mathcal{A}$ be a collection in n of cardinality i. Then* $\mathrm{Sq}_i(\mathcal{A})$ *is the position of $\mathcal{A}$ in the (n, i)-supercollection.*

As in Example 5.5, we can take $\mathrm{Sq}_i(\mathcal{A})$ and reconstruct the collection $\mathcal{A}$ and obtain $\#\mathcal{A}$. Moreover, this process can be reversed. Therefore if we know $\mathcal{A}$ belongs to the (n, i)-supercollection and only one of $\mathcal{A}$, $\#\mathcal{A}$, or $\mathrm{Sq}_i(\mathcal{A})$ is known, the other two quantities can be obtained. We present two examples of this process, one beginning with the squashed order position, and the second starting with the collection number.

Example 5.7 *Let $\mathcal{B}$ be a collection with $n = 3$ and cardinality 4, and let $Sq_4(\mathcal{B}) = 22$. Then,*

$$Sq_4(\mathcal{B}) = 22 = \binom{6}{4} + \binom{4}{3} + \binom{3}{2}$$

From this representation, we have $\beta(\mathcal{B}) = 1010110$. This gives $\#\mathcal{B} = 86$ and

$$\mathcal{B} = \{\{2\}, \{1, 2\}, \{1, 3\}, \{1, 2, 3\}\}$$

Example 5.8 *Let $\mathcal{D}$ be a collection in the $(3, 5)$-supercollection with $\#\mathcal{D} = 103$. Written in binary, 103 is 1100111. From this we know*

$$\mathcal{D} = \{\{1\}, \{2\}, \{1, 2\}, \{2, 3\}, \{1, 2, 3\}\}$$

We reconstruct the 5-binomial representation of $\mathcal{D}$ as

$$\binom{6}{5} + \binom{5}{4} + \binom{3}{3} = Sq_5(\mathcal{D}) = 12$$

This kind of calculation allows for a new way to iterate through the collections for a given n, namely by iterating through each (n, i)-supercollection independently in order to find UC-classes. By Lemma 3.10, we know that it is sufficient to only inspect collections that include the set $[n]$. So, for a (n,i)-supercollection it is sufficient to only iterate through the squashed order positions $\binom{2^n-2}{i}$ through $\binom{2^n-1}{i}$ since any collection in this range necessarily contains $[n]$. We call this subset the **topset of the (n,i)-supercollection**. At this point, the total number of sets to check is the same, 2^{2^n-2}. However, we can make use of the (n,i)-supercollections and squashed order to reduce the number of collections to iterate through.

Certain (n,i)-supercollections always contain the same number of UC-classes, for any n. A trivial example is the $(n, 2^n-1)$-supercollection which contains only one collection, $\mathcal{P}^*([n])$, and this is clearly the only UC-class in this supercollection. (Figures 3.2, 3.3, and 3.4 are convenient graphs of the number of UC-classes in the (n,i)-supercollections for $n = 3, 4, 5$, and the related data is given in Appendix A.) Equally as trivial is the $(n,1)$-supercollection. This supercollection only has one (proper) UC-class as well, namely $[[\text{Half}(n)]]$. Other supercollections can be completely analyzed as well. We consider the $(n,2)$-supercollection in the following lemma.

Lemma 5.9 *In the $(n,2)$-supercollection, there are $n-1$ proper UC-classes.*

*P*roof: Every proper UC-class in the $(n,2)$-supercollection consists of exactly two sets, one being the set $[n]$. Let $\mathcal{A} = \{[n], A\}$ and $\mathcal{A}' = \{[n], A'\}$. If $|A| = |A'|$, then $\mathcal{A} \sim_P \mathcal{A}'$. Therefore, there are only $n-1$ UC-classes in the $(n,2)$-supercollection.□

We also pose the following conjecture about (n,i)-supercollections near $\mathcal{P}^*([n])$.

Conjecture 5.10 *The number of UC-classes for the $(n,(2^n-1)-i)$-supercollections is the same for all $n \geq i$.*

For those supercollections that we cannot directly calculate, we ask if each UC-class has an element within a specific interval of the squashed order. For example, does every proper UC-class have a family $\mathcal{F}$, $|\mathcal{F}| = i$, so that the squashed order position of $\mathcal{F}$ is in the "top half" of the topset? That is,

$$\mathrm{Sq}_i(\mathcal{F}) \geq \frac{\binom{2^n-1}{i} - \binom{2^n-2}{i}}{2}$$

The above statement turns out to be false, with a counterexample when $n = 5$ and $i = 4$ or 8 (see Figure 5.5).

In a (n, i)-supercollection, each UC-class has precisely one maximal element, and of these maxima there one having the least associated value. This so-called "minimal-maximum" is by definition the boundary value that is of interest, as every UC-class must have an element with squashed order position greater than or equal to this value. Therefore, to account for all UC-classes it would be sufficient to begin at the minimal-maximum value and increment until the final squashed order position. The question remains, what is the minimal-maximum for a given (n, i)-supercollection?

For each minimal-maximum, we calculate its percentile rank within both the entire (n, i)-supercollection and the related topset. For example, let $\mathcal{G}$ be the minimal-maximum collection in the $(3, 2)$-supercollection. The collection number for $\mathcal{G}$ is 72, and its squashed order position is 19. Because there are $\binom{7}{2} = 21$ collections in this supercollection, the percentile rank for $\mathcal{G}$ in the supercollection is $\frac{19}{21}$, or 90.48%. In other words, 90.48% of the collections in the $(3, 2)$-supercollection have collection number less than or equal to $\#\mathcal{G}$. The percentile rank for $\mathcal{G}$ in the topset is 66.67%.

We present the percentiles for the minimal-maximum collection for each (n, i)-supercollection for $n = 3, 4, 5$ in Appendix C, but provide graphs of these percentiles in Figures 5.3, 5.4, and 5.5 below.

Looking at these graphs, there appears to be a pattern in the percentiles of the total squashed ordering. In particular, the supercollection with cardinality 2^{n-1}

has the lowest percentile, and for $n = 4$ and 5 the percentiles are 83.21 and 83.25, respectively. We will return to this idea in a moment.

Now, instead of graphing the percentiles for these minimal-maximum collections, we ask if there is a pattern within their structure, and we inspect the binary representation for these collections. In order to highlight any patterns, we place the binary representations in a grid and color the gridboxes containing 1. Doing this, we obtain the grids in Figures 5.6, 5.7, and 5.8.

A repeated visual pattern can be seen in these grids. Moreover, note the minimal-maximum for the $(n, 2^{n-1})$-supercollections:

$n = 3$	1101010
$n = 4$	110101010101010
$n = 5$	1101010101010101010101010101010

Each collection is a degenerate family, because the element 1 only occurs in the set $[n]$. In other words, these collections are isomorphic to $\mathcal{P}^*([n-1]) \cup [n]$. We can imitate this pattern for $n = 6$, and we get a family whose squashed order position is at the 83.3 percentile. This is in line with previous observations. This brings us to another conjecture.

Conjecture 5.11 *Let $[[\mathcal{A}]]$ be a UC-class in n with cardinality i. If $\mathcal{A}'$ is the collection with the maximal collection number in this UC-class, then*

$$\#\mathcal{A}' > .83\binom{2^n - 1}{i}$$

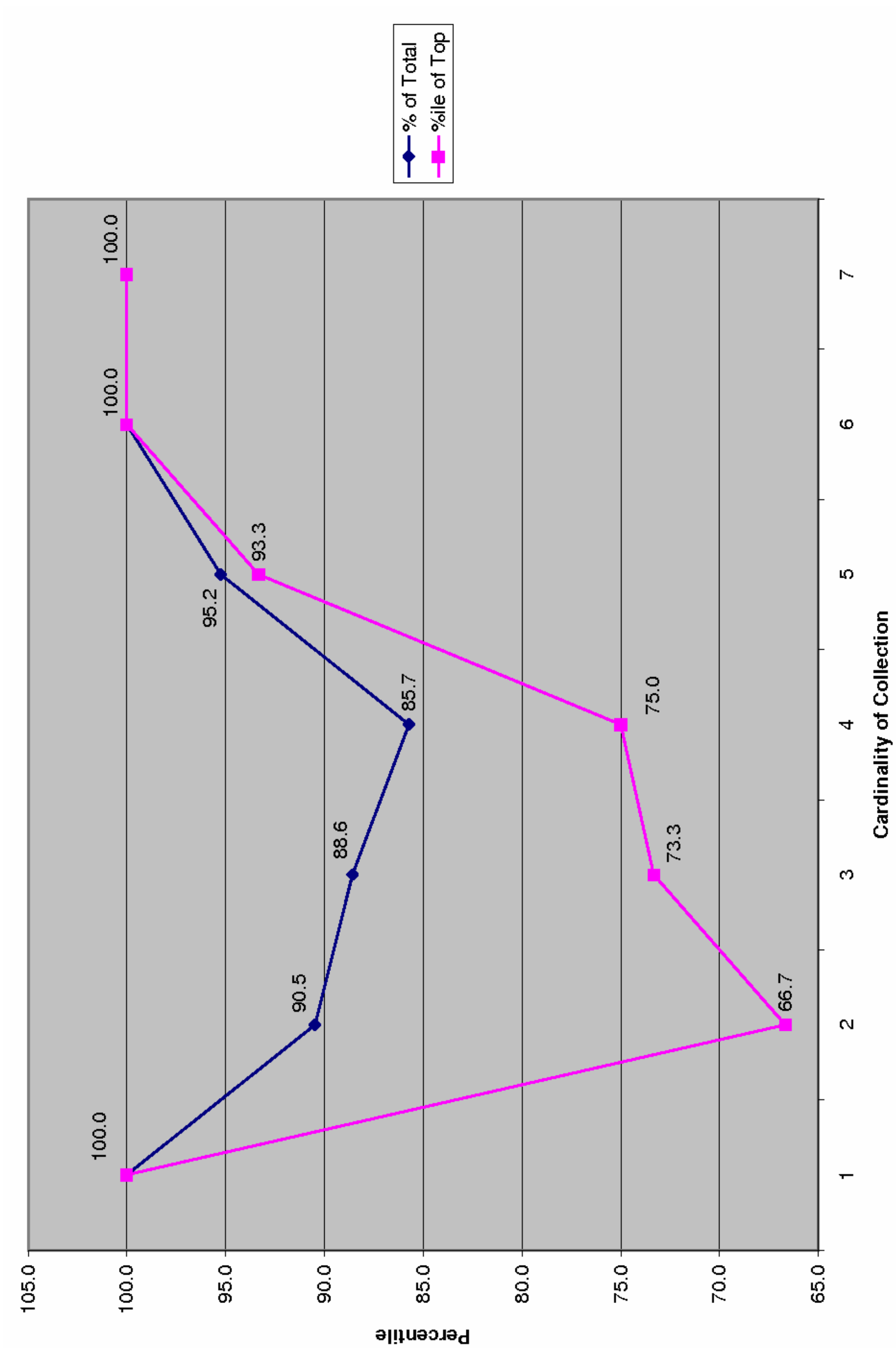

Figure 5.3: Minimal-maximum graph for $(3, i)$-supercollections.

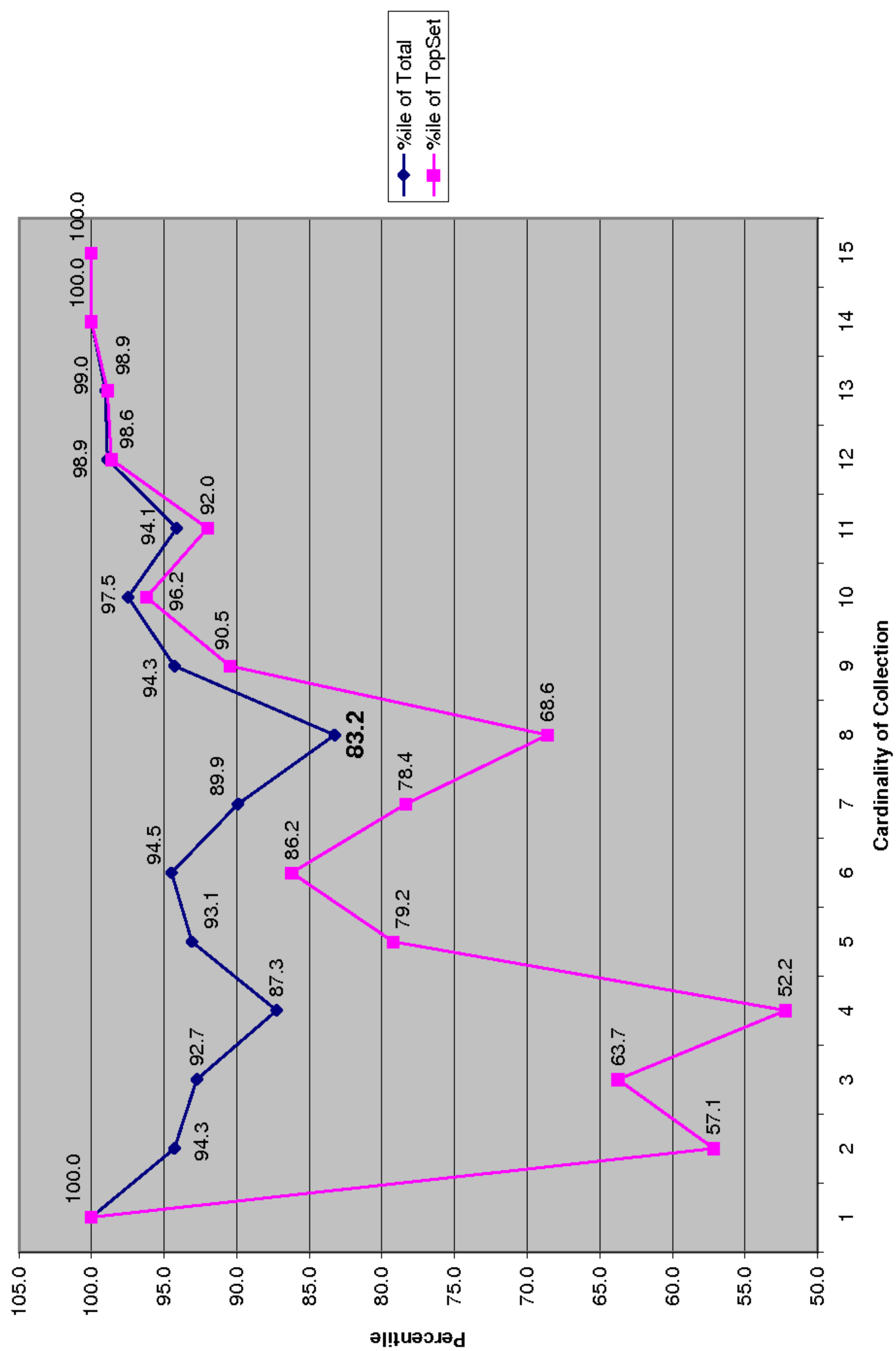

Figure 5.4: Minimal-maximum graph for $(4, i)$-supercollections.

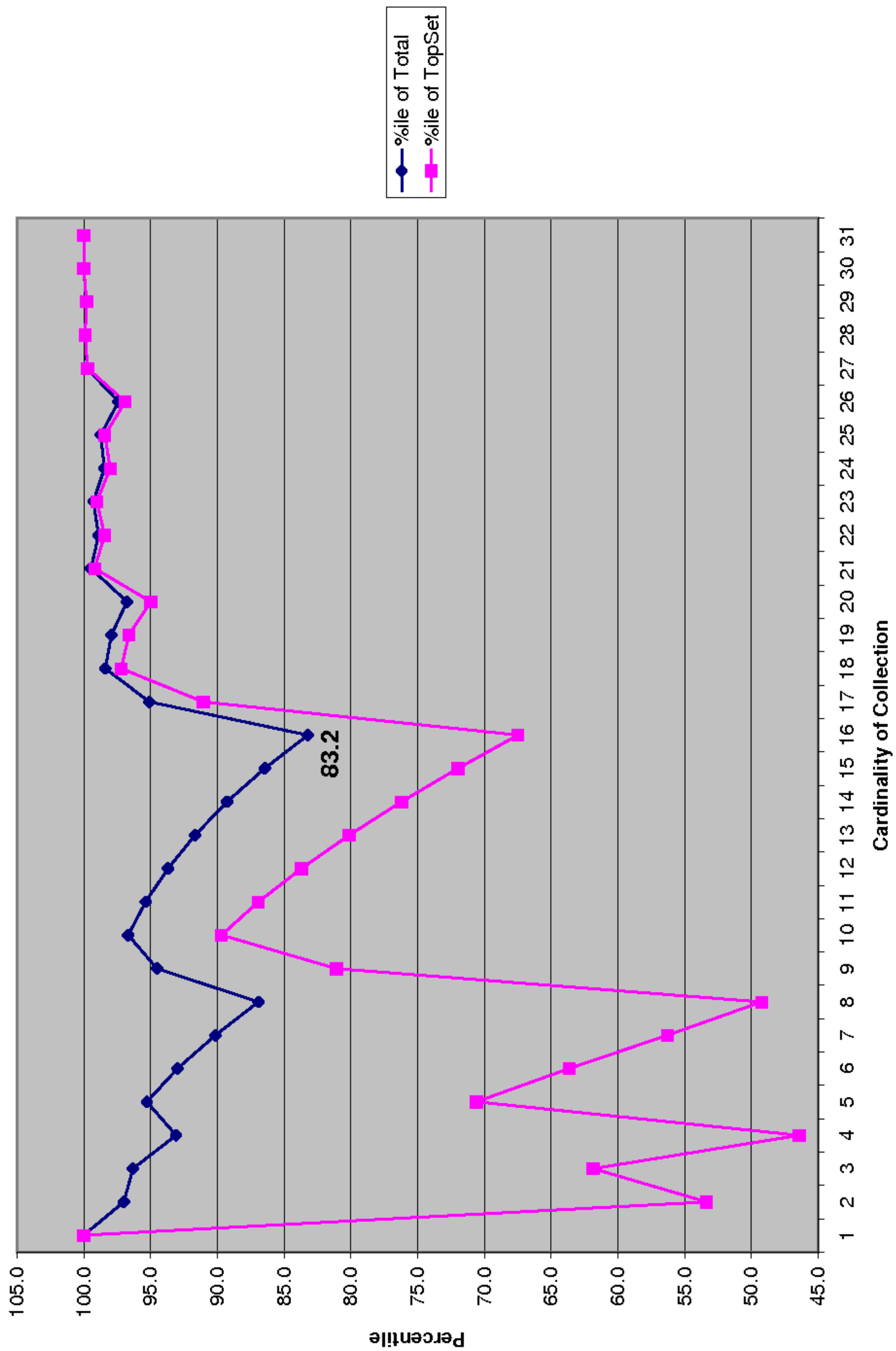

Figure 5.5: Minimal-maximum graph for $(5, i)$-supercollections.

Cardinality	Collection Number								
1	64	=	1	0	0	0	0	0	0
2	72	=	1	0	0	1	0	0	0
3	97	=	1	1	0	0	0	0	1
4	106	=	1	1	0	1	0	1	0
5	122	=	1	1	1	1	0	1	0
6	126	=	1	1	1	1	1	1	0
7	127	=	1	1	1	1	1	1	1

Figure 5.6: Binary representations for $(3, i)$-supercollections.

Cardinality	Collection Number																	
1	16384	=	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	16512	=	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
3	18436	=	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0
4	18568	=	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0
5	24961	=	1	1	0	0	0	0	1	1	0	0	0	0	0	0	1
6	27272	=	1	1	0	1	0	1	0	1	0	0	0	1	0	0	0
7	27304	=	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0
8	27306	=	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0
9	31402	=	1	1	1	1	0	1	0	1	0	1	0	1	0	1	0
10	32426	=	1	1	1	1	1	1	0	1	0	1	0	1	0	1	0
11	32476	=	1	1	1	1	1	1	0	1	1	0	1	1	1	0	0
12	32746	=	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0
13	32762	=	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0
14	32766	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
15	32767	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 5.7: Binary representations for $(4, i)$-supercollections.

Cardinality	Collection Number																																
1	1073741824	=	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1073774592	=	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1082163200	=	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1082163328	=	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
5	1216348232	=	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
6	1216905344	=	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
7	1216907392	=	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
8	1216907400	=	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
9	1635877249	=	1	1	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1
10	1787332744	=	1	1	0	1	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
11	1789429896	=	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
12	1789436576	=	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0
13	1789569160	=	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0
14	1789569672	=	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	0	1	0	0	0
15	1789569704	=	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	0
16	1789569706	=	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
17	2058005162	=	1	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
18	2125114026	=	1	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
19	2129308330	=	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
20	2129324714	=	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0
21	2146101930	=	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0
22	2146101994	=	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1	0
23	2147145682	=	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	1	0	1	1	1	1	1	0	1	0	0	1	0
24	2147151594	=	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	0	1	0
25	2147409372	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	0
26	2147417820	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	0	1	1	1	0	0
27	2147483356	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	0	0
28	2147483626	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0
29	2147483642	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0
30	2147483646	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
31	2147483647	=	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 5.8: Binary representations for $(5, i)$-supercollections.

CHAPTER VI
GRAPHS OF UNION-CLOSED FAMILIES

In Figures 4.1 and 2.2, we used graphical representations of families to easily observe similarities (and differences) in families. In this chapter we show how graphs for families are created, and we make some simple observations about these graphs. We finish with an explanation of our graph-creation algorithm.

6.1 Families and Graphs

In his work, Poonen observes that a union-closed family, with the subset relation, has the structure of a semilattice [6], a special type of partially-ordered set (or, *poset*). Posets easily lend themselves to graphical representations, called *Hasse diagrams* [8]. First, we define the notion of covering within a poset, as stated in [8]

Definition 6.1 *Let $(P, <)$ be a poset, and let $x, y \in P$. We say x* ***covers*** *y if $x < y$ and if there is no $z \in P$ such that $x < z < y$.*

In a Hasse diagram, the elements of the poset form the vertices of the graph, and the edges of the graph are the cover relations. Moreover, if $x < y$ then the vertex for y is drawn "above" x, that is, with a greater vertical coordinate [8].

For our graphs of families, we use a variation of the Hasse diagram. Instead of having the vertical positioning of the vertices (sets) be determined by the covering relation, we calculate the vertical position of a set according to its level. Recall that the level of a set is defined as the length of the minimal path from itself to the the set $[n]$. Vertices with a lower level are placed higher in the graph, and thus the set $[n]$ is at top of the graph. Consequently, it is possible that a set A may be placed lower in the graph than a set B, even though A may cover B. In order to preserve this information, we present the covering relations as directed edges where $B \rightarrow A$ if and only if $A \subset B$, and the resulting graph is a directed graph. For a vertex v in a directed graph we notate the indegree and outdegree of v as $\deg^+(v)$ and $\deg^-(v)$, respectively.

We present an example of the graph of a family in Figure 6.1. In Figure 6.2 we present an example where a set covers a set in a higher level ($\{1,3,5\}$ covers $\{3,5\}$) and also where a set covers a set in the same level ($\{1,3,4,5\}$ covers $\{1,3,4\}$).

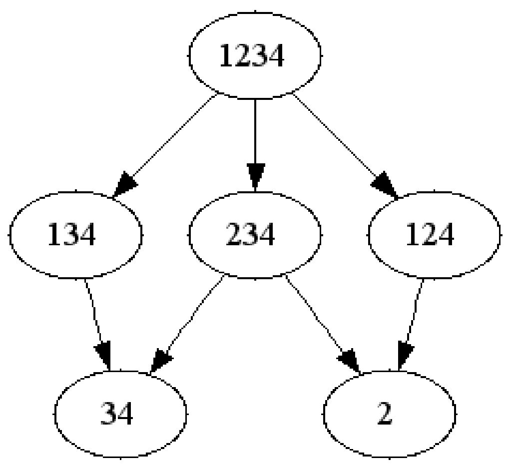

Figure 6.1: The collection $\mathcal{B}$, $\#\mathcal{B} = 31746$.

One convenient consequence of the set numbering we use throughout this paper is stated in the following lemma.

Lemma 6.2 *Let A, B be distinct sets of a family in n. If $A \subset B$, then $\#\mathcal{A} < \#\mathcal{B}$.*

Proof: Consider the binary sequences $\alpha(A) = (a_n, \ldots, a_1)$ and $\alpha(B) = (b_n, \ldots, b_1)$. Since $A \subset B$, we know if $a_j = 1$, then $b_j = 1$, and because $A \neq B$, there exists i, $1 \leq i \leq n$, where $a_i = 0$ and $b_i = 1$. Therefore $\#\mathcal{A} < \#\mathcal{B}$. $\square$

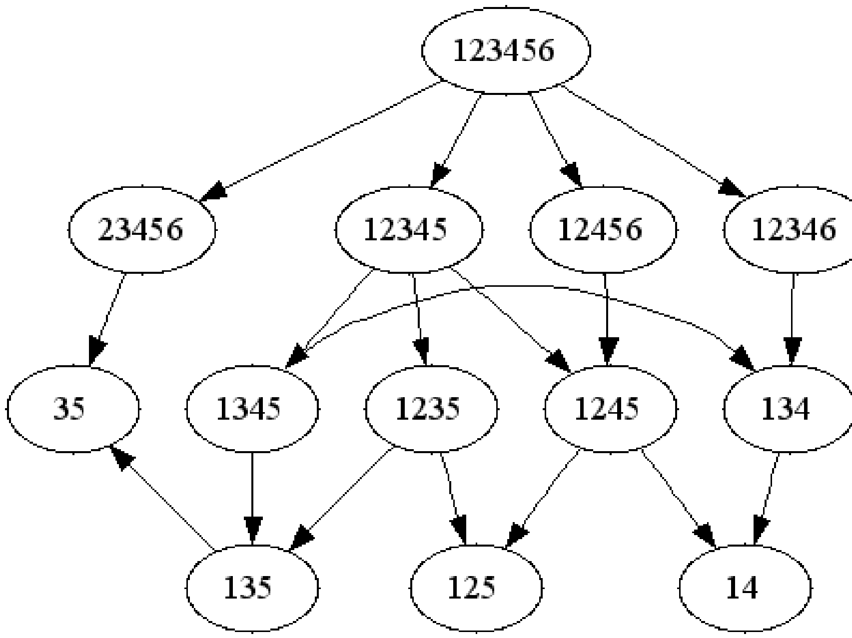

Figure 6.2: An example of cover relations that are within a single level or from a lower level to a higher level.

We can also use the graph to re-state certain properties of the family. Clearly the root node $[n]$ is the only node with indegree equal to zero. Also, in [4] Johnson and Vaughn define the minimal generating collection of a family as the collection of sets in the family having no more than one child. In terms of the graph we can restate this definition as follows.

Definition 6.3 *Given the graph of a family $\mathcal{F}$, the minimal generating collection for $\mathcal{F}$ is the collection formed by the sets with outdegree less than or equal to* 1*.*

The following is also a rephrasing of a lemma of Johnson and Vaughan [4].

Lemma 6.4 *Let A be a set of a family in n, $\mathcal{F}$, and consider the graph of $\mathcal{F}$. If $|A| = i$, then $deg^{-}(A) \leq i$.*

6.2 Constructing the Graph Programmatically

In this section (and in the next chapter) we shift our focus to the programmatic structures used to model collections, families, and their graphs.

When constructing the graph of a family, there are several important questions to consider. The first of these is deciding how to iterate through the sets in the family. It is simplest to begin with the root node $[n]$ and construct the branches below. Fortunately, the sets of a family are stored in memory as elements within a map container, and this map container maintains all the sets in numeric order according to set number. We can then make good use of Lemma 6.2 and know that all subsets of a given set will occur earlier in the numeric listing. Therefore, we iterate through the sets in reverse set number order.

It is important to note that the graph is not a tree; there may be more than one path from the root node to a given node. This fact raises the next few questions, one of which is, how do we construct an efficient and accurate algorithm to place each node? We choose to use a recursive algorithm, and it necessarily must be exhaustive. That is, it must determine if the node being placed is a subset of each existing node in the graph. Of course, it is only necessary to visit a node once when placing a new node. For this reason, as we search through the entire graph we "color" each existing node as visited so that if it happens to be the child of multiple nodes it is still only checked once. After a node is placed, then the graph is reset and all nodes appear unvisited so that the next node can be placed.

Another issue is the calculation of levels in a graph. Because there may be multiple paths from a node to the root, we must have a way to determine the minimal path in order to find the level of a node. As an example, in Figure 6.2 the node $\{1, 4\}$ has four distinct paths from itself to the root, three of length 3 and one of length 4. In order to calculate levels, we choose to have each node maintain a listing of its parents. Then, we iterate in reverse set number order as before, calculating the level of each node and setting the level for the root equal to zero. Therefore when calculating the level of an arbitrary set A, the levels for all potential parents have been calculated, and the level for A is simply one more than the minimum level for its parent nodes.

In the following chapter we present the pseudocode for the recursive graph construction algorithm.

CHAPTER VII
NOTES ON TECHNOLOGY USED

Throughout this paper, nearly all calculated results pertaining to sets, collections, and families were obtained through the use of custom programs written in the C++ language. To model a collection programmatically, three custom classes were implemented, one for sets, one for collections, and one for graph nodes. A separate library of functions was also written to generate permutation-equivalent collections from a given collection.

In this chapter we consider only on portions of these custom code libraries. For those interested, the complete source code is available for use under the terms of the GNU General Public License at `http://www.uncg.edu/~cfrenn/ucf`.

7.1 The MSET and MFAMILY Classes

The MSET and MFAMILY classes represent a set and collection respectively. The name ‘mset’ (for “mathematical set”) was chosen due to the fact that the word ‘set’ is a reserved word in the C++ language. An mfamily is a set of msets.

The mset class is straightforward, as it is essentially an unsigned integer variable that stores the set number along with several other basic properties (the cardinality, for example) and member functions (or, *methods*). A convenient feature of the class is the inclusion of several overloaded operators for common set operations, namely union, intersection, and containment.

The mfamily class is much more complex. The main data structure in an mfamily is a map container that stores the mset objects for the family. A map stores objects in $< key,\ value >$ pairs, and specific objects can be retrieved simply from the unique associated key value. Since set numbers are unique, we use these set numbers as the key values for the msets in the family. Also, as mentioned in the previous chapter, the map container automatically maintains its elements in order according to their key values.

In addition to the map container for the family itself, an mfamily also stores the related graph structure for a union-closed family. The graph is stored as a vector of "gnodes", another custom object discussed below but are essentially the set numbers for the msets. An mfamily also stores some other secondary information, such as the size of the family, the minimum and maximum cardinalities of member sets, and the collection number.

There are many methods related to an mfamily, and we mention only a few here. One method will create the appropriate collection given a collection number. Another will return the minimal generating collection for a union-closed family after the graph has been constructed. The two main methods are to return the closure of a collection and to build the graph of a union-closed family.

To form the closure of the collection, we follow the logic provided below in pseudocode.

`given a collection,` $\mathcal{F}$

`declare the mfamily,` $temp$ `(which is empty)`

    `for each` A`, where` $A \in \mathcal{F}\{$

        `for each` B`, where` $B \in temp\{$

            `insert` $A \cup B$ `into` $temp$

        $\}$

        `insert` A `into` $temp$

    $\}$

`return` *temp*

Example 7.1 *Let $\mathcal{F} = \{A, B, C\}$. The closure obtained by the above algorithm is listed below, in order of insertion.*

$$\overline{\mathcal{F}} = \{A, A \cup B, B, A \cup C, A \cup B \cup C, B \cup C, C\}$$

The algorithm used to build the graph consists of two parts. The first, which we will call nodeInsert, is a recursive method in the gnode class and places a given set in graph as a descendant of the calling node. The second method is in the mfamily class and simply iterates through the sets of the mfamily and places them into the graph using the nodeInsert method from the root node. It then resets the coloring of all nodes in the graph in preparation the insertion of the next set. We discuss the nodeInsert method in the next section.

7.2 The GNODE class

As mentioned earlier, the GNODE ("graph node") class is the structure used for the creation of the graph of a given family. Each gnode stores the set number for the set it represents, along with the level of the node, a boolean 'visited' value, and two vectors that store the set numbers for its parent and child nodes. The critical method for this class is the nodeInsert method which recursively seeks to place a node where appropriate as a descendant of the calling node. For our purposes, we always insert nodes from the root and assume that all nodes are 'unvisited' according to out coloring. We provide the pseudocode for the nodeInsert method below.

```
Let X be the calling node and A be the node to be inserted.
if (A ⊂ X) and (X is not visited){
     if A is a subset of a child of X{
          for each child C of X{
             call nodeInsert to place A as descendant of C **Recursion
          }
          mark X as visited
     }
     else{
          store A as child of X
          store X as parent of A
     }
}
return
```

7.3 Finding the Permutations of a Collection

In addition to the classes discussed above, another code library was developed to realize the permuation equivalent collections of a given collection. This is not a trivial computation, for even when $n = 5$, a collection may have up to 120 collections that are permutation-equivalent. We implement code that, given the collection number for a collection in n $\mathcal{A}$, returns the collection numbers of all collections permutation-equivalent to $\mathcal{A}$ in a set container. (A set container is similar to the map container, only that the value stored also serves as the key.)

To accomplish all permutations in n, we first define a vector of set numbers, $permVector$, where the i^{th} element is the set number for the set $\{i + 1\}$. (The difference of 1 is due to the fact that the vector index begins at 0, and we have not considered the empty set in our work.) This original definition serves in fact as the identity permutation. We then use a built-in C++ function to iterate through all permutations of these n elements of $permVector$. At each stage, the current permutation π is defined by mapping the vector index to the set contained in that position. Then, given a set A, the set $\pi(A)$ is found by the following:

$$\bigcup_{i \in A} permVector[i-1]$$

Given a collection number, we first extract the collection to a vector of its member sets. Then, we build a new collection by permuting each set according to the current permutation defined by $permVector$. Once all sets are permuted, then the collection for the new collection is calculated and stored. We repeat this process until all permutions of $permVector$ are considered. We provide the pseudocode for this process below.

```
Given a collection in n, B
declare collection T
declare set temp
for each permutation π of permVector{
    clear T
    for each set A in B{
        clear temp
        for each element i ∈ {1, 2, ..., n}{
            if i ∈ A{
                temp = temp ∪ permVector[i − 1]
            }
        }
        T = T ∪ {temp}
    }
    store #T
}
```

7.4 Other Technology Used

All programming was done in the C++ language, using Microsoft VisualStudio .NET. Many of the graphs provided in Chapters 3 and 5 were accomplished using Microsoft Excel, with the exception of the expanded graphs in Figures 3.8 and 3.9. For these, the GraphSight (`http://www.graphsight.com`) software program was used. The graph images were created using the *dot* graph visualization program (`http://www.graphviz.org`).

BIBLIOGRAPHY

[1] Anderson, Ian. *Combinatorics of Finite Sets.* Mineola, NY: Dover Publications, Inc. 1987.

[2] Gao, W. and Yu, H. *Note on the Union-Closed Sets Conjecture.* Ars Combinatoria. **49** (1998):280-288.

[3] Harary, Frank. *Graph Theory.* Reading, MA: Addison-Wesley, 1969.

[4] Johnson, Robert T. and Theresa P. Vaughan. *On Union-Closed Families, I.* J. Combinatorial Theory, Series A. **84** (1998):242-249.

[5] Morris, Robert. *FC-families and improved bounds for Frankl's conjecture.* European Journal of Combinatorics. **27** (2006): 269282.

[6] Poonen, Bjorn. *Union-Closed Families.* J. Combinatorial Theory, Series A. **59** (1992):253-268.

[7] Sarvate, D.G. and J-C. Renaud. *On the Union-Closed Sets Conjecture.* Ars Combinatoria. **27** (1989):149-154.

[8] Stanley, Richard P. *Enumerative Combinatorics, Volume I.* Wadsworth & Brooks/Cole. Monterey, CA. 1986.

[9] Tucker, Alan. *Applied Combinatorics.* New York: Jon Wiley & Sons, Inc. 2002.

[10] Weisstein, Eric W. "Symmetric Group." From *MathWorld* - A Wolfram Web Resource.
`http://mathworld.wolfram.com/SymmetricGroup.html`

[11] Weisstein, Eric W. "Pair Group." From *MathWorld* - A Wolfram Web Resource.
`http://mathworld.wolfram.com/PairGroup.html`

[12] "Cycle index." *Wikipedia, The Free Encyclopedia.* 24 Nov 2006, 17:33 UTC. Wikimedia Foundation, Inc. 25 Nov 2006
`http://en.wikipedia.org/w/index.php?title=Cycle_index&oldid=89855205`

APPENDIX A
NUMERICAL DATA FOR UC-CLASSES

Below we provide the numerical data used to create the histograms in Figures 3.2, 3.3, and 3.4, which show the distribution of proper UC-classes with respect to the cardinality of the family.

Cardinality of Family	Number of UC Classes
1	1
2	2
3	3
4	4
5	2
6	1
7	1

Figure A.1: Count of UC-Classes by Cardinality when $n = 3$.

Cardinality of Family	Number of UC Classes
1	1
2	3
3	7
4	13
5	21
6	24
7	25
8	24
9	18
10	12
11	9
12	4
13	2
14	1
15	1

Figure A.2: Count of UC-Classes by Cardinality when $n = 4$.

Cardinality of Family	Number of UC Classes
1	1
2	4
3	12
4	32
5	73
6	149
7	264
8	423
9	625
10	841
11	1057
12	1244
13	1382
14	1448
15	1437
16	1341
17	1167
18	946
19	722
20	507
21	335
22	207
23	124
24	67
25	36
26	19
27	9
28	4
29	2
30	1
31	1

Figure A.3: Count of UC-Classes by Cardinality when $n = 5$.

APPENDIX B
CALCULATED CYCLE INDICES

In Chapter 4, we made use of cycle indices of the elements of the symmetric group S_n. Here we present tables showing cycle indices for permutations $\pi \in S_n$ and the permutations π induces, for $3 \leq n \leq 6$. In the following tables, the cycle indices for the groups $S_n^{(i)}$ and S_n^* can be computed in the following manner,

$$Z(S_n^{(i)}) = \sum \alpha_\pi Z(\pi^{(i)})$$

$\pi \in S_3$	α_π	$Z(\pi)$	$Z(\pi^{(2)})$	$Z(\pi^*)$
(1)(2)(3)	1	x_1^3	x_1^3	x_1^6
(12)(3)	3	x_1x_2	x_1x_2	$x_1^2x_2^2$
(123)	2	x_3	x_3	x_3^2

Figure B.1: Cycle indices for $n = 3$.

$\pi \in S_4$	α_π	$Z(\pi)$	$Z(\pi^{(2)})$	$Z(\pi^{(3)})$	$Z(\pi^*)$
(1)(2)(3)(4)	1	x_1^4	x_1^6	x_1^4	x_1^{14}
(12)(3)(4)	6	$x_1^2x_2$	$x_1^2x_2^2$	$x_1^2x_2$	$x_1^6x_2^4$
(123)(4)	8	x_1x_3	x_3^2	x_1x_3	$x_1^2x_3^4$
(12)(34)	3	x_2^2	$x_1^2x_2^2$	x_2^2	$x_1^2x_2^6$
(1234)	6	x_4	x_2x_4	x_4	$x_2x_4^3$

Figure B.2: Cycle indices for $n = 4$.

$\pi \in S_5$	α_π	$Z(\pi)$	$Z(\pi^{(2)})$	$Z(\pi^{(3)})$	$Z(\pi^{(4)})$	$Z(\pi^*)$
(1)(2)(3)(4)(5)	1	x_1^5	x_1^{10}	x_1^{10}	x_1^5	x_1^{30}
(12)(3)(4)(5)	10	$x_1^3x_2$	$x_1^4x_2^3$	$x_1^4x_2^3$	$x_1^3x_2$	$x_1^{14}x_2^8$
(123)(4)(5)	20	$x_1^2x_3$	$x_1x_3^3$	$x_1x_3^3$	$x_1^2x_3$	$x_1^6x_3^8$
(1234)(5)	30	x_1x_4	$x_2x_4^2$	$x_2x_4^2$	x_1x_4	$x_1^2x_2^2x_4^6$
(12)(34)(5)	15	$x_1x_2^2$	$x_1^2x_2^4$	$x_1^2x_2^4$	$x_1x_2^2$	$x_1^6x_2^{12}$
(12)(345)	20	x_2x_3	$x_1x_3x_6$	$x_1x_3x_6$	x_2x_3	$x_1^2x_2^2x_3^4x_6^2$
(12345)	24	x_5	x_5^2	x_5^2	x_5	x_5^6

Figure B.3: Cycle indices for $n = 5$.

$\pi \in S_6$	α_π	$Z(\pi)$	$Z(\pi^{(2)})$	$Z(\pi^{(3)})$	$Z(\pi^{(4)})$	$Z(\pi^{(5)})$	$Z(\pi^*)$
(1)(2)(3)(4)(5)(6)	1	x_1^6	x_1^{15}	x_1^{20}	x_1^{15}	x_1^6	x_1^{62}
(12)(3)(4)(5)(6)	15	$x_1^4x_2$	$x_1^7x_2^4$	$x_1^8x_2^6$	$x_1^7x_2^4$	$x_1^4x_2$	$x_1^{30}x_2^{16}$
(123)(4)(5)(6)	40	$x_1^3x_3$	$x_1^3x_3^4$	$x_1^2x_3^6$	$x_1^3x_3^4$	$x_1^3x_3$	$x_1^{14}x_3^{16}$
(1234)(5)(6)	90	$x_1^2x_4$	$x_1x_2x_4^3$	$x_2^2x_4^4$	$x_1x_2x_4^3$	$x_1^2x_4$	$x_1^6x_2^4x_4^{12}$
(12)(34)(5)(6)	45	$x_1^2x_2^2$	$x_1^3x_2^6$	$x_1^4x_2^8$	$x_1^3x_2^6$	$x_1^2x_2^2$	$x_1^{14}x_2^{24}$
(12345)(6)	144	x_1x_5	x_5^3	x_5^4	x_5^3	x_1x_5	$x_1^2x_5^{12}$
(123)(45)(6)	120	$x_1x_2x_3$	$x_1x_2x_3^2x_6$	$x_1^2x_3^2x_6^2$	$x_1x_2x_3^2x_6$	$x_1x_2x_3$	$x_1^6x_2^4x_3^8x_6^4$
(12)(34)(56)	90	x_2^3	$x_1^3x_2^6$	x_2^{10}	$x_1^3x_2^6$	x_2^3	$x_1^6x_2^{28}$
(1234)(56)	90	x_2x_4	$x_1x_2x_4^3$	$x_2^2x_4^4$	$x_1x_2x_4^3$	x_2x_4	$x_1^2x_2^6x_4^{12}$
(123)(456)	40	x_3^2	x_3^5	$x_1^2x_3^6$	x_3^5	x_3^2	$x_1^2x_3^{20}$
(123456)	120	x_6	$x_3x_6^2$	$x_2x_6x_{12}$	$x_3x_6^2$	x_6	$x_2x_3^2x_6^7x_{12}$

Figure B.4: Cycle indices for $n = 6$.

APPENDIX C
NUMERICAL DATA FOR MINIMAL-MAXIMA

Here we present the numerical data used to generate Figures 5.3, 5.4, and 5.5, as well as the collection number and squashed order position for each minimal-maximum collection.

Cardinality	Minimal-Maximum Collection Number	Squashed Order Position	%ile of Total	%ile of Topset
1	64	7	100	100
2	72	19	90.48	66.67
3	97	31	88.57	73.33
4	106	30	85.71	75
5	122	20	95.24	93.33
6	126	7	100	100
7	127	1	100	100

Figure C.1: Minimal-maximum data for $(3, i)$-supercollections.

Cardinality	Minimal-Maximum Collection Number	Squashed Order Position	%ile of Total	%ile of Topset
1	16384	15	100	100
2	16512	99	94.29	57.14
3	18436	422	92.75	63.74
4	18568	1191	87.25	52.20
5	24961	2795	93.07	79.22
6	27272	4729	94.49	86.21
7	27304	5785	89.90	78.35
8	27306	5357	83.25	68.59
9	31402	4719	94.29	90.48
10	32426	2927	97.47	96.20
11	32476	1285	94.14	92.01
12	32746	450	98.90	98.63
13	32762	104	99.05	98.90
14	32766	15	100	100
15	32767	1	100	100

Figure C.2: Minimal-maximum data for $(4, i)$-supercollections.

Cardinality	Minimal-Maximum Collection Number	Squashed Order Position	%ile of Total	%ile of Topset
1	1073741824	31	100	100
2	1073774592	451	96.99	53.33
3	1082163200	4329	96.31	61.84
4	1082163328	29289	93.08	46.40
5	1216348232	161846	95.25	70.57
6	1216905344	684442	92.96	63.62
7	1216907392	2369853	90.12	56.26
8	1216907400	6855085	86.90	49.23
9	1635877249	19052157	94.50	81.07
10	1787332744	42874925	96.67	89.67
11	1789429896	80738314	95.35	86.91
12	1789436576	132207330	93.68	83.68
13	1789569160	189049792	91.66	80.11
14	1789569672	236667607	89.25	76.19
15	1789569704	259765150	86.43	71.96
16	1789569706	250070306	83.21	67.46
17	2058005162	252154796	95.09	91.04
18	2125114026	202883046	98.37	97.19
19	2129308330	138189130	97.92	96.61
20	2129324714	81928847	96.76	94.98
21	2146101930	44104841	99.44	99.18
22	2146101994	19935666	98.89	98.43
23	2147145682	7831342	99.27	99.02
24	2147151594	2588876	98.45	98.00
25	2147409372	726850	98.72	98.41
26	2147417820	165463	97.38	96.88
27	2147483356	31385	99.75	99.71
28	2147483626	4490	99.89	99.88
29	2147483642	464	99.78	99.77
30	2147483646	31	100	100
31	2147483647	1	100	100

Figure C.3: Minimal-maximum data for $(5, i)$-supercollections.