Estimating the number of covering relations in a formal concept lattice

by

Aaron Michael Rodriguez

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Applied Mathematics

Program of Study Committee:

Jennifer Newman, Major Professor

Clifford Bergman

Derrick Stolee

Iowa State University

Ames, Iowa

2015

DEDICATION

I dedicate this thesis to my parents and grandparents, who never stopped telling me to keep moving forward.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I thank Dr. Jennifer Newman for her guidance and support thoughout this project. I also thank Volodymr Sukhinin for his help and insight, my committee members for their suggested revisions, and my peers for their ongoing encouragement.

ABSTRACT

Given data organized in a tabular form called a *cross table*, the multi-dimensional relationships in the data are represented by a directed acyclic graph called a *formal concept lattice*. For large sets of data, drawing the graph is not reasonable. We focus on quantitative characteristics of the lattice structure that lead to decreased computation time in the determination of nodes and edge sets on large lattices from big data.

This work addresses the problem of recovering the covering relations, of a concept lattice, given the set of nodes. We implement one existing algorithm and formally prove that it correctly computes the covering relations of a given concept lattice from its nodes. We also discuss methods for estimating the number of covering relations in a lattice and offer a conjecture for an upper bound for this number. We present experimental results to predict edge frequency distribution for a range of cross table densities. Finally we discuss some open problems.

CHAPTER 1. INTRODUCTION

In recent times, we have seen a substantial growth in the size of large data sets. More and more are mathematical techniques being used in the analysis and data mining of such large amounts of data. One way in which data mining has become useful is in the challenge of modeling complex systems, systems of interacting events which have different underlying processes.

Formal Concept Analysis (FCA) is a method that was introduced in the 1980s, which derives a hierarchical structure from relational data. Its original motivation was to concretely represent complete lattices, but has now become widely used for modeling complex systems in an organized and logical manner. Given data organized in a tabular form called a *cross table*, similar to a flat table in a database, the multi-dimensional relationships in the data are represented by a directed acyclic graph (or partially ordered set, or lattice). If the table is small enough, the relational structure can be depicted visually by a graph. For the size of the datasets in which we are interested - "big data" - drawing the graph is not reasonable nor expected. Hence, our work is focused on a more practical aspect of the analysis: study of quantitative characteristics of the lattice structure that lead to decreased computation time in the determination of nodes and edge sets on large lattices from big data. We define the lattices in Chapter 2.

A recent challenge is to model complex systems that are dynamic in nature. If one samples a system whose state changes with time, the lattice structure that represents the system updates at each time increment. Given a lattice representation at time t, changes are sensed and attributes for the system are updated. How can we efficiently construct the new lattice from the previous one? Many algorithms have been developed for efficiently updating the formal concept lattice for small data sets. In [8], the author presents efficient algorithms for updating

the set of nodes in the formal concept lattice. However, the algorithms do not output the relationships between elements, which are modeled by a partial order on the set. It is the relationships - the edges in the lattice - that provide the structure to the multi-dimentional correlations in the data, so it is important to know the edge set at each iteration. Most FCA algorithms construct a lattice from scratch, so any updates to a system would require the re-computation of the nodes and edges from the beginning. Our current research is ongoing to produce the node set at time $t+1$, given the node set at time t [8], in an efficient manner.

In this work we address the problem of recovering the partial order of a concept lattice and discuss methods for estimating the number of immediate precedence relationships between elements of the lattice. While our research does not provide a new algorithm to generate a new edge set from the previous iteration, we do offer a conjecture for an upper bound on the number of edges for a lattice, given the node set and some other information. We also provide some experimental results on the probability distribution of edges in a lattice. This thesis is organized as follows. In Chapter 2, we provide formal definitions and background from graph theory, poset theory, and formal concept analysis. In Chapter 3, we present an algorithm from [1] that, given the elements of the concept lattice, returns the precedence links between members. In Chapter 4, we provide some theoretical and experimental results regarding the number of immediate relationships in the lattice constructed via Algorithm 1. In Chapter 5, we present a conclusion and ideas for future research.

CHAPTER 2. PRELIMINARIES

In this chapter we provide basic definitions and examples for the work we present later. Since formal concepts form a lattice structure, which has a graph representation and in particular a directed acyclic graph representation, we discuss graphs, partially ordered sets, and lattices.

2.1 Background on Graphs

A graph is a pair $H = (V, E)$ where V is a set of vertices and E is a set of 2-element subsets of V, called edges.

A directed graph (or digraph) is a graph in which each edge is an ordered pair of vertices. Such edges are then called directed edges, arcs, or arrows. A directed graph is said be acyclic if there is no sequence of arcs such that one can begin at a vertex v and return to v by following the sequence.

2.2 Background on Lattices

Definition 2.2.1. *A partially ordered set, or poset, is a set P with a binary relation $R \subseteq P \times P$ satisfying*

1. *$(x, x) \in R$ $\forall x \in P$ (reflexivity)*
2. *if $(x, y) \in R$ and $(y, x) \in R$, then $x = y$ (antisymmetry)*
3. *if $(x, y) \in R$ and $(y, z) \in R$, then $(x, z) \in R$ (transitivity)*

For the remainder of this thesis, we assume our posets are finite.

Definition 2.2.2. *A partially ordered set in which every pair of elements has a greatest lower bound and a least upper bound is called a lattice. We denote the greatest lower bound and least*

upper bound by $\wedge$ (meet) and $\vee$ (join), respectively. The greatest lower bound is sometimes called the infimum, and the least upper bound is sometimes called the supremum.

Definition 2.2.3. *A partially ordered set $\langle P, \leq \rangle$ is a complete lattice if every subset A of P has both an infimum and supremum, denoted by $\bigwedge A$ and $\bigvee A$, respectively.*

Definition 2.2.4. *[3] Let P be a partially ordered set, and let Q be a subset of P. We say Q is join-dense if every element of P is the join of a subset of Q. We say Q is meet-dense if every element of P is the meet of a subset of Q.*

Definition 2.2.5. *Let $\langle P, \leq \rangle$ be a partially ordered set. We say $S \subseteq P$ is a chain (antichain) of P if for any two elements $x, y \in S$, x and y are comparable (incomparable).*

Definition 2.2.6. *[3] A maximum chain (antichain) is a chain (antichain) with cardinality at least as large as every other chain (antichain).*

Let P be a poset. An element $y \in P$ is said to *cover* an element $x \in P$, $y \neq x$, if $x \leq y$ and there does not exist $z \in P$, $z \neq x, y$ such that $x \leq z \leq y$. We call the pair $\{x, y\}$ where y covers x a *covering relation*. The *Hasse Diagram* of a poset P is the graph $\mathcal{H} = (V, E)$ where V is the set of elements in P, and E is the set of covering relations, i.e. $\{x, y\} \in E$ if and only if y covers x. When an element y covers an element x, we say y is in the set of upper covers of x, and dually, that x is in the set of lower covers of y.

The binary relation R on a set P gives rise to a directed graph whose vertices are the elements of P and where there is an arc $\{x, y\}$ for every ordered pair of elements that are related in R. When R is an order relation, as in a poset, the directed graph is acyclic. Since a Hasse diagram shows covering relations, a Hasse diagram is actually the transitive reduction of the directed acyclic graph that represents R.

Definition 2.2.7. *[3] The height of a lattice $\mathcal{L}$, denoted as $h(\mathcal{L})$, is the cardinality of a maximum chain of $\mathcal{L}$.*

Definition 2.2.8. *[3] The width of a lattice $\mathcal{L}$, denoted as $\omega(\mathcal{L})$, is the cardinality of a maximum antichain of $\mathcal{L}$.*

2.3 Background on Formal Concept Analysis

Definition 2.3.1. *[5] Let G be a set of objects, M a set of attributes, and $I \subseteq G \times M$ a binary relation between G and M. A triple $\mathbb{K} = (G, M, I)$ is called a formal context. The pair $(g, m) \in I$ can also be written as gIm and signifies that object g has attribute m.*

For the remainder of this work, we will use $\mathbb{K}$ to mean $\mathbb{K} = (G, M, I)$ unless otherwise stated.

A formal context can be depicted visually in a tabular form with crosses representing the pairs $(g, m) \in I$. For example, consider the cross table in Figure 2.1.

(g,m)	a	b	c	d	e
1	×		×		
2	×				
3		×	×	×	
4	×	×			
5		×	×	×	×

Figure 2.1: A tabular form, or cross table, with $G = \{1, 2, 3, 4, 5\}$ and $M = \{a, b, c, d, e\}$. The relation I is indicated by the set of crosses in the table.

The cross table in Figure 2.1 depicts a formal context in which object 1 has attributes a and c, object 2 has attribute a, etc.

Definition 2.3.2. *[5] Let $X \subseteq G$, $Y \subseteq M$ and $I \subseteq G \times M$. We define the following operators*

$$X^I := \{m \in M \mid gIm \ \forall g \in X\},$$

$$Y^I := \{g \in G \mid gIm \ \forall m \in Y\}.$$

Definition 2.3.3. *[5] Let $\mathbb{K}$ be a formal context. A pair (A, B) such that $A \subseteq G$, $B \subseteq M$, $A = B^I$, and $B = A^I$ is called a formal concept in $\mathbb{K}$. The sets A and B are called the extent and intent of the formal concept (A, B), respectively. We denote the set of all formal concepts of $\mathbb{K}$ by $\mathfrak{C}(\mathbb{K})$.*

The next proposition and theorem are results used in constructing the edge sets and proving a later theorem.

Proposition 2.3.4. *[5] Let (A_1, B_1) and (A_2, B_2) be formal concepts of a formal context $\mathbb{K}$. Then $A_1 \subseteq A_2 \Leftrightarrow B_1 \supseteq B_2$.*

Proof. Assume $A_1 \subseteq A_2$. Let $b \in B_2$. Since $B_2 = A_2^I = \{m \in M \mid gIm \; \forall g \in A_2\}$, gIb for all $g \in A_2$. By definition, $B_1 = A_1^I = \{m \in M \mid gIm \; \forall g \in A_1\}$. Since $A_1 \subseteq A_2$, gIb for all $g \in A_1$. Hence, $b \in B_1$ and therefore $B_1 \supseteq B_2$. The argument for proving the other direction is similar. □

The set $\mathfrak{C}(\mathbb{K})$ forms a partially ordered set under the order relation defined as

$$(A_1, B_1) \leq (A_2, B_2) \iff A_1 \subseteq A_2 \iff B_1 \supseteq B_2$$

which is characterized in the following theorem from [5].

Theorem 2.3.5. *[5] A partially ordered set $\langle \mathfrak{C}(\mathbb{K}), \leq \rangle$ is a complete lattice, called the concept lattice of $\mathbb{K}$. The infimum (meet) and supremum (join) for a subset of $\mathfrak{C}(\mathbb{K})$ are expressed as*

$$\bigwedge_{t \in T} (A_t, B_t) = \left(\bigcap_{t \in T} A_t, \left(\bigcup_{t \in T} B_t \right)^{II} \right),$$
$$\bigvee_{t \in T} (A_t, B_t) = \left(\left(\bigcup_{t \in T} A_t \right)^{II}, \bigcap_{t \in T} B_t \right)$$

A complete lattice $\mathcal{L}$ is isomorphic to $\mathfrak{C}(\mathbb{K})$ if and only if there are mappings $\gamma : G \to \mathcal{L}$ and $\mu : M \to \mathcal{L}$ such that $\gamma(G)$ is join-dense in $\mathcal{L}$, $\mu(M)$ is meet-dense in $\mathcal{L}$ and gIm is equivalent to $\gamma(g) \leq \mu(m)$ for all $g \in G$ and all $m \in M$. In particular, $\mathcal{L} \cong \mathfrak{C}(\mathcal{L}, \mathcal{L}, \leq)$.

The following example illustrates these results. We refer to this example throughout this work.

Example 2.3.6. *Suppose we have a formal context $\mathbb{K}$ given by the cross table in Figure 2.1. As a convenience to the reader, we display it again in Figure 2.2.*

(g,m)	a	b	c	d	e
1	×		×		
2	×				
3		×	×	×	
4	×	×			
5		×	×	×	×

Figure 2.2: Cross table from Figure 2.1.

By inspection, we can see that the see that the pair $(\{3,5\},\{b,c,d\})$ *is a formal concept. The pairs* $(\{5\},\{b,c,d\})$ *and* $(\{3\},\{b,c,d\})$ *are not concepts because they do not satisfy Definition 2.3.3. In Figure 2.3 we give the concept lattice for this context.*

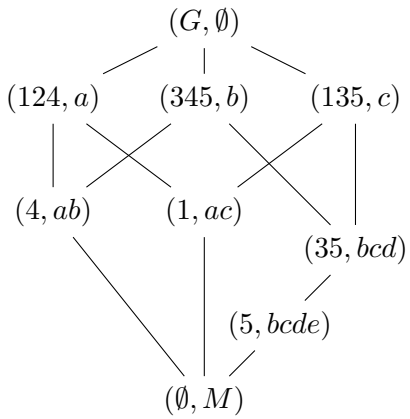


Figure 2.3: The concept lattice for the formal concext $\mathbb{K}$.

For simplicity, we omit set braces, commas, set union, and set intersection symbols for the extent and intent of formal concepts unless there is a need for clarification. For example, the concept $(\{3,5\},\{b,c,d\})$ would be written as $(35,bcd)$, and $bcd \cap ab$ and $35 \cup 4$ are b and 345, respectively.

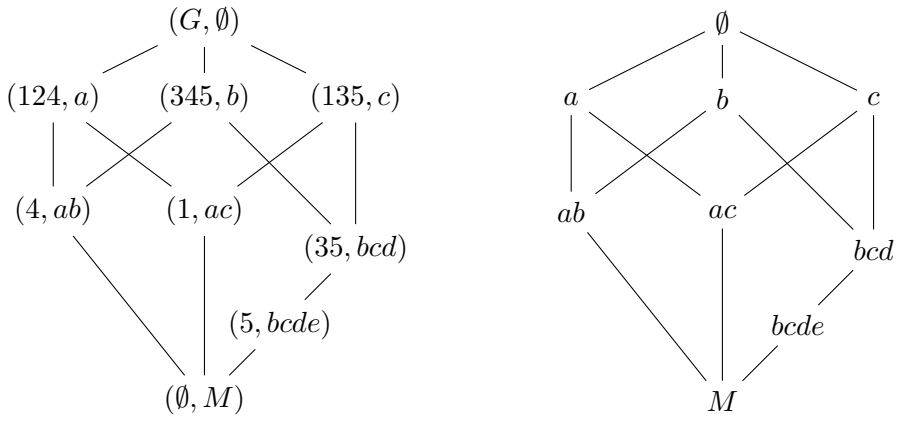
CHAPTER 3. EDGE CONSTRUCTION ALGORITHMS AND LITERATURE REVIEW

Ongoing research in [8] is creating a more efficient method for generating the set of formal concepts from a cross table, as well as providing an efficient algorithm for generating the iteration $t+1$ of a cross table from a cross table at time t. That algorithm provides only the set of concepts, or nodes, of the lattice. Part of the work of this thesis is to implement the most efficient algorithm we could find from the literature that generates the edge set of the lattice. This provides a method whereby we can compare our combined algortithm that generates nodes and edges to other algorithms in the literature, as many existing algorithms start with a cross table and output a set of nodes and edges.

The algorithm *CoveringEdges* is given in [2] and is used in [6] as a baseline for comparison with *BorderAlg*, which is introduced in [6]. CoveringEdges uses both the extent and intent of concepts to determine explicit covering relations of the concept lattice while BorderAlg achieves the same by only using the intents.

We chose the edge producing algorithm called *iPred*, published in [1], for our implementation. We describe the iPred algorithm for recovering the partial order of a concept lattice. The algorithm was introduced as an improvement on BorderAlg, introduced in [6].

We first justify a map that takes a concept lattice to a corresponding lattice of intents. Then, we explain the precedence relation and its consequences. Finally, after stating the definitions needed for understanding the algorithm, we present proofs for the main claims proposed in [1]. The authors of [1] give a proof for Proposition 3.0.16 (Proposition 1 in [1]), which serves as the main test condition in the algorithm, and a sketch of the proof of correctness for the algorithm itself. Our aim here is to present these proofs formally, filling in the details of each proof.



(a) $\mathfrak{C}(\mathbb{K})$ (b) $f(\mathfrak{C}(\mathbb{K}))$

Figure 3.2: The concept lattice mapped to its lattice of intents, and their respective Hasse diagrams.

Definition 3.0.7. *Let P and Q be partially ordered sets. A map $f : P \to Q$ is a poset isomorphism if it is bijective and order-embedding, i.e., for all $x, y \in P$, $x \leq_P y$ if and only if $f(x) \leq_Q f(y)$.*

Given a concept lattice $\langle \mathfrak{C}(\mathbb{K}), \leq \rangle$, there is a lattice isomorphism to $\langle \mathcal{C}, \supseteq \rangle$ given by $(A, B) \mapsto B$ where $\mathcal{C}$ is the set of intents of the formal context $\mathbb{K}$. Clearly this map is bijective since every formal concept has an intent to which it is mapped, and no two concepts can have the same intent without contradicting Definition 2.3.3. It is also clear that the map is order-embedding since $(A_1, B_1) \leq (A_2, B_2)$ if and only if $B_1 \supseteq B_2$. Because of this, all computation will be done using the complete lattice $\mathcal{L} = \langle \mathcal{C}, \supseteq \rangle$. We note that the join operation of $\mathcal{L}$ is set intersection. Figure 3.2 illustrates the map when applied to the lattice from Example 2.3.6.

The precedence relation is denoted by $\preceq$. If $c, \tilde{c} \in \mathcal{C}$, then $c \preceq \tilde{c} \iff \tilde{c} \subseteq c$, and we write $c \prec \tilde{c}$ if and only if c is an immediate predecessor of $\tilde{c}$ (if and only if $\tilde{c}$ is an immediate successor of c). By saying c is an immediate predecessor of $\tilde{c}$, we mean that $\tilde{c} \subseteq c$ and if there exists $\hat{c}$ such that $\tilde{c} \subseteq \hat{c} \subseteq c$, then either $\tilde{c} = \hat{c}$ or $\hat{c} = c$.

Remark 3.0.8. *Let $c, \tilde{c} \in \mathcal{C}$. The following are equivalent:*

1. $c \preceq \tilde{c}$
2. $\tilde{c} \subseteq c$

3. c is a predecessor of $\tilde{c}$

4. $\tilde{c}$ is a successor of c

Observation 3.0.9. *[1] If $c \prec \tilde{c}$, then $|c|>|\tilde{c}|$. (i.e. An immediate predecessor has more attributes.)*

We decribe BorderAlg, as this will provide motivation for iPred. Our objective is to recover the parial order of the concept lattice, which amounts to finding all of the edges on the Hasse diagram. This can be done by finding the set of upper covers for each element of the lattice. One can achieve this in the following manner: First, sort elements in $\mathcal{L}$ according to cardinality. As the elements are processed one by one, starting with the element of smallest cardinality, keep track of an updated set of elements called the *Border*. The Border set is the set of maximal elements (in the set inclusion sense) among those that have been processed. An element $\tilde{c} \in \mathcal{C}$ is maximal if $\tilde{c} \subseteq c$ implies $\tilde{c} = c$ for $c \in \mathcal{C}$. Next, determine a set of candidates for membership in the set of upper covers. The *Candidate set* is formed by taking the intersection of the current element with each element in Border. Only immediate successors of the current element can be in the set of upper covers of that element, so the maximal elements among those in the Candidate set are chosen. These form the *Cover set* of the current element. Edges are added, and Border is updated.

There are methods for determining the maximal elements of the candidate set, as shown in [6] via the BorderAlg algorithm. The running time of BorderAlg is $O\left(|\mathcal{C}| \times \omega(\mathcal{L}) \times |M|^2\right)$ [1]. The authors of iPred improve this complexity by a factor of $|M|$ by approaching the task from a different perspective. An element $\tilde{c}$ from the candidate set is an upper cover of an element c_i if and only if c_i is a lower cover of $\tilde{c}$.

Observation 3.0.10. *$\tilde{c}$ is in the candidate set of upper covers for $c_i \in \mathcal{C}$ if and only if $c_i \preceq \tilde{c}$ (if and only if $\tilde{c} \subseteq c_i$).*

Instead of checking if $\tilde{c}$ is an upper cover of c_i, the algorithm iPred checks that c_i is a lower cover of $\tilde{c}$. The following definitions are useful for understanding the iPred algorithm and proofs.

Definition 3.0.11. *[1] Let $\mathcal{C}$ be the set of elements in $\mathcal{L}$. An enumeration of $\mathcal{C}$ is an ordered listing*

$$enum(\mathcal{C}) = \{c_1, c_2, \ldots, c_n\}$$

such that $\forall i, j \leq n$: $i \leq j \Rightarrow |c_i| \leq |c_j|$. Note that $enum(\mathcal{C})$ is not necessarily a unique listing.

Definition 3.0.12. *[1] A face of $c \in \mathcal{C}$ with respect to $\tilde{c}$ is the set difference $\tilde{c} - c$ such that $\tilde{c} \prec c$.*

Definition 3.0.13. *[1] The set of faces of $c \in \mathcal{C}$ is*

$$faces(c) = \{\tilde{c} - c \mid \tilde{c} \prec c\}$$

Definition 3.0.14. *[1] An accumulation of faces (up to i) of $c = c_k \in \mathcal{C}$ with respect to $enum(\mathcal{C})$ is*

$$\Delta_c^i = \bigcup \{c_j - c \mid c_j \in enum(\mathcal{C}) \text{ and } c_j \prec c \text{ and } j < i\}$$

Note that i must be greater than or equal to $k + 1$. Also note that the set Δ_c^i contains only set differences of nodes that are immediate predecessors of c.

The accumulation of faces is used to test if an element is in the lower cover of another element. The accumulation Δ_c^i is constructed for a particular i only when the immediate predecessors c_j of c are known, $j < i$. Note that we abuse notation and write the faces and accumulation of faces as a union of the elements comprising the set defferences, omitting of course any duplicate elements that may result.

Lemma 3.0.15. *Let $c_1, c_2, \ldots, c_n$ be any collection of immediate predecessors of an element c. Then $(c_i - c) \cap (c_j - c) = \emptyset$ for all $i \neq j$.*

Proof. The proof is straightforward. Since c_i and c_j are immediate predecessors of c, $c_i \vee c_j = c$. Since the join operation on this lattice is set intersection, $c_i \cap c_j = c$, and the claim follows immediately. □

The following proposition is proven in [1]. We restate it here with a more detailed proof.

Proposition 3.0.16. *[1] Let $\mathcal{L} = \langle \mathcal{C}, \supseteq \rangle$ be the lattice of intents of a formal context $\mathbb{K}$, and enum($\mathcal{C}$) be an enumeration of $\mathcal{C}$. Let $c \in$ enum($\mathcal{C}$) and construct Δ_c^i. Then, $c_i \prec c$ if and only if $c_i \cap \Delta_c^i = \emptyset$ and $c_i \preceq c$.*

Proof. ($\Leftarrow$) Assume $c_i \cap \Delta_c^i = \emptyset$ and $c_i \preceq c$. By way of contradiction, assume $c_i \not\prec c$. Then there is a $\tilde{c} \neq c_i$ such that $c_i \preceq \tilde{c} \prec c$. This implies that $|c| < |\tilde{c}| < |c_i|$. It follows from Definition 3.0.11 of enumeration and Definition 3.0.14 of accumulation of faces that $\tilde{c} - c \subseteq \Delta_c^i$. Since $c_i \preceq \tilde{c}$, then $\tilde{c} \subseteq c_i$, and hence $\tilde{c} - c \subseteq c_i$. We then have $\tilde{c} - c \subseteq c_i \cap \Delta_c^i$, and hence $c_i \cap \Delta_c^i \neq \emptyset$, a contradiction.

($\Rightarrow$) Now assume $c_i \prec c$. Clearly, $c_i \preceq c$. Let $\hat{c}$ be any immediate predecessor of c with cardinality at most $|c_i|$. Then $\hat{c}$ must be incomparable with c_i, otherwise we would have $c \subsetneq \hat{c} \subseteq c_i$. Then c_i would not be an immediate precessor of c. By Definition 3.0.14 of accumulation of faces, we have

$$\begin{aligned} c_i \cap \Delta_c^i &= [c \cup (c_i - c)] \cap \Delta_c^i \\ &= \left[c \cap \Delta_c^i\right] \cup \left[(c_i - c) \cap \Delta_c^i\right] \end{aligned}$$

By Definition 3.0.14, $c \cap \Delta_c^i = \emptyset$, and by Lemma 3.0.15 $(c_i - c) \cap \Delta_c^i = \emptyset$. Therefore, $c_i \cap \Delta_c^i = \emptyset$.

□

Proposition 3.0.16 provides the main condition for recovering the partial order of the lattice, and is the basis for the iPred algorithm (Algorithm 1).

Input: $\mathcal{C} = \{c_1, c_2, \ldots, c_l\}$

Output: $E = \{(c_j, c_k) : c_j \prec c_k\}$

```
Sort(C)
for i = 2 : l do
    Δ[c_i] ← ∅;
end
Border ← c_1
for i = 2 : l do
    Candidate ← {c_i ∩ c̃ | c̃ ∈ Border};
    for each c̃ ∈ Candidate do
        if Δ[c̃] ∩ c_i = ∅ then
            Add edge (c_i, c̃) to E;
            Δ[c̃] = Δ[c̃] ∪ (c_i − c̃);
            Border ← Border − c̃;
        end
    end
    Border ← Border ∪ c_i;
end
```

Algorithm 1: iPred algorithm

Theorem 3.0.17. *The iPred algorithm finds all pairs* (c_k, c_j) *where* $c_k \prec c_j$ *in* $\mathcal{L}$.

Proof. First note that the algorithm will terminate because our lattices are assumed to be finite. Fix an enumeration $\mathcal{C} = \{c_1, c_2, \ldots, c_l\}$. At the beginning of the algorithm, $\Delta[\tilde{c}]$ is initialized to be empty for each $\tilde{c} \in \mathcal{C}$. Each time an immediate predessesor of $\tilde{c}$ is found, $\Delta[\tilde{c}]$ is updated. This ensures that in loop i of the algorithm, $\Delta^i_{\tilde{c}}$ is correctly computed and stored in $\Delta[\tilde{c}]$.

An edge $(c_i, \tilde{c})$ is found if and only if the condtions from Propostion 3.0.16 are satisfied. Since $\tilde{c}$ is in the Candidate set of c_i, we have that $\tilde{c} \subseteq c_i$ and hence $c_i \preceq \tilde{c}$. Thus, it remains to test whether $\Delta^i_{\tilde{c}} \cap c_i = \emptyset$. Since $\Delta[\tilde{c}] = \Delta^i_{\tilde{c}}$, it suffices to test whether $\Delta[\tilde{c}] \cap c_i = \emptyset$. If

$\Delta[\tilde{c}] \cap c_i = \emptyset$ and we already have $c_i \preceq \tilde{c}$, then $c_i \prec \tilde{c}$ and we have an edge $(c_i, \tilde{c})$. If $\Delta[\tilde{c}] \cap c_i \neq \emptyset$, then $(c_i, \tilde{c})$ is not an edge and the algorithm checks the next candidate. When the algorithm terminates, no false edges are found, and no edges are missing.

Lastly, we show that the Border set for the current elements is correctly updated. The Border set is the set of maximal elements from those that have been processed, so c_i is added to Border whether or not the condition from Propostion 3.0.16 tests positive because the elements were sorted by size for the enumeration (i.e. $|c_{i-1}| \leq |c_i|$ for $i = 2, \ldots, l$.) Suppose the condition from Propostion 3.0.16 does test positive. Then $\tilde{c}$ is removed from Border since a postive test implies that $\tilde{c} \subsetneq c_i$, in which case $\tilde{c}$ would no longer be maximal. □

Next, we demonstrate the algorithm by applying it to the set of formal concepts from Example 2.3.6.

Example 3.0.18. *Suppose we are given the set of formal concepts from Example 2.3.6. Since iPred actually finds the partial order for the lattice of intents, we will first map each concept to its repsective intent. This gives us a set of intents $\mathcal{C}$ such that*

$$enum(\mathcal{C}) = \{\emptyset, a, b, c, ab, ac, bcd, bcde, abcde\}.$$

We initialize the accumulation of faces to be empty for each element of $enum(\mathcal{C})$, and then we set Border to $\emptyset$. When $c_2 = a$, the candidate set only has the empty set, and $\Delta[\emptyset] \cap a = \emptyset$. Hence, there exists an edge between $\emptyset$ and a. We update $\Delta[\emptyset]$ to $\Delta[\emptyset] = a$, and the border is updated to $\emptyset$, then to $\{a\}$. The algorithm advances to the next element in the enumeration.

It is easy to see that, similar to the first iteration, there are edges between $\emptyset$ and b and between $\emptyset$ and c. Once the loop is completed for $i = 4$, we have Border $= \{a, b, c\}$ and $\Delta[\emptyset] = abc$. For $i = 5$, the current element is ab and Candidate $= \{a, b, \emptyset\}$. The following are the loops for each member of the candidate set.

$c_5 = ab$

$Candidate = \{a, b, \emptyset\}$

$\Delta\,[a] \cap ab = \emptyset$	$\Delta\,[b] \cap ab = \emptyset$	$\Delta\,[\emptyset] \cap ab = ab \neq \emptyset$
$Edge\ (ab, a)$	$Edge\ (ab, b)$	$No\ edge$
$\Delta\,[a] = b$	$\Delta\,[b] = a$	$\Delta\,[\emptyset] = abc$
$Border = \{b, c\}$	$Border = \{c\}$	$Border = \{c\}$
		$Border = \{c, ab\}$

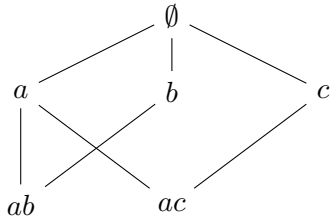
For $i = 6$, the current element is ac and $Candidate = \{c, a\}$. The following are the loops for each member of the candidate set.

$c_6 = ac$

$Candidate = \{c, a\}$

$\Delta\,[c] \cap ac = \emptyset$	$\Delta\,[a] \cap ac = \emptyset$
$Edge\ (ac, c)$	$Edge\ (ac, a)$
$\Delta\,[c] = a$	$\Delta\,[a] = bc$
$Border = \{ab\}$	$Border = \{ab\}$
	$Border = \{ab, ac\}$

At this point we have constructed the following portion of the lattice.



The following are the iterations for $i = 7$ *and* $i = 8$.

$c_7 = bcd$

$Candidate = \{b, c\}$

$\Delta[b] \cap bcd = \emptyset$	$\Delta[c] \cap bcd = \emptyset$
$Edge\ (bcd, b)$	$Edge\ (bcd, c)$
$\Delta[b] = acd$	$\Delta[c] = abd$
$Border = \{ab, ac\}$	$Border = \{ab, ac\}$
	$Border = \{ab, ac, bcd\}$

$c_8 = bcde$

$Candidate = \{b, c, bcd\}$

$\Delta[b] \cap bcde = cd \neq \emptyset$	$\Delta[c] \cap bcde = bd \neq \emptyset$	$\Delta[bcd] \cap bcde = \emptyset$
$No\ edge$	$No\ edge$	$Edge\ (bcde, bcd)$
$\Delta[b] = acd$	$\Delta[c] = abd$	$\Delta[bcd] = e$
$Border = \{ab, ac, bcd\}$	$Border = \{ab, ac, bcd\}$	$Border = \{ab, ac, bcd\}$
		$Border = \{ab, ac, bcde\}$

When $i = 9$, $c_9 = abcde$ *which is the infimum of all elements in* $\mathcal{C}$. *It is clear that* ab, ac, *and* $bcde$ *each have edges to* $abcde$. *This completes the algorithm.*

CHAPTER 4. RESULTS

It is reasonable to infer that the number of immediate relationships in the formal context influences the running time for the algorithm. This gives rise to questions about the number of edges in the concept lattice for a given context. Although the exact number cannot be known a priori, we would still like to estimate the number of edges.

In this chapter, we explore upper bounds on the number of edges of the graph $(\mathfrak{C}(\mathbb{K}), E)$ where E is the set of covering relations for the concept lattice $\langle \mathfrak{C}(\mathbb{K}), \leq \rangle$. We ran an experiment whose results we use to give an experimental frequency count (probability distribution) of all edges in a context. We describe this experiment in Section 4.2. This experiment allows one to model the frequency distribution of edges for classes of cross tables described in Section 4.2.

4.1 Bounding the Number of Edges

Recall that the lattice at hand is $\mathcal{L} = \langle \mathcal{C}, \supseteq \rangle$ where $\mathcal{C}$ is the set of intents of the formal context. Since $\mathcal{C}$ is a subset of the power set of M, the set of attributes, we can obtain an upper bound of the number of edges in $\mathcal{L}$ by counting the number of edges in the power set lattice on $|M|$ elements. The number of edges in this power set lattice can be written as

$$\sum_{i=1}^{|M|} \binom{|M|}{i} \binom{i}{i-1}$$

The first binomial term counts the number of i-element subsets of M, and the second binomial term counts the number of $(i-1)$-element subsets of each i-element subset. This is a very large number, certainly much larger than $2^{|M|}$, the cardinality of the power set itself, and much, much larger than the actual edge set. This number is too large to be useful for practical purposes, so we would like to find a more useful estimate.

Wiseman proved Theorem 4.1.1 in 1990, which expresses an upper bound in terms of the height and number of elements in the lattice.

Theorem 4.1.1. *[9] Let P be a poset with $|P| = n$ and height $k \geq 4$. Then*

$$|E| \leq \left\lfloor \frac{(n-k)^2}{4} \right\rfloor + 2n - k - 1$$

Furthermore, there exists a poset whose Hasse diagram realizes the bound.

In example 3.0.18, $n = 9$, $k = 5$, and the actual number of edges is 13.

$$\left\lfloor \frac{(9-5)^2}{4} \right\rfloor + 2(9) - 5 - 1 = 16 > 13$$

Example 4.1.2 shows, however, that while Wiseman's bound is true it does not always provide a close estimate for the number of edges.

Example 4.1.2. *Suppose we are given the following formal context $\mathbb{K}$, given by the cross table in Figure 4.1.*

(g,m)	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*k*	*l*
1		×					×	×	×	
2			×		×	×				
3	×			×					×	×
4	×								×	
5		×								
6		×	×					×	×	
7	×				×		×			
8										×
9						×				
10				×			×			

Figure 4.1: A 10×10 example cross table

Let $\mathcal{C}$ be the set of intents. Then, $enum(\mathcal{C}) = \{\emptyset, a, b, c, d, e, f, g, h, k, l, ak, dg, cef, aeg, bhk, bghk, adkl, bchk, abcdefghkl\}$. The number of elements is 20, the height is 5, and the number of edges is 36. (See Figure 4.2.)

$$\left\lfloor \frac{(20-5)^2}{4} \right\rfloor + 2(20) - 5 - 1 = 90 > 36$$

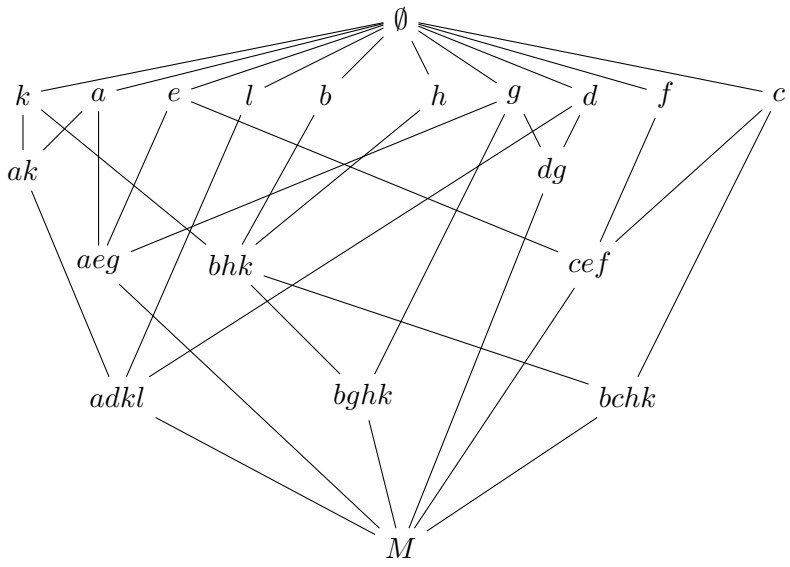


Figure 4.2: The lattice from Example 4.1.2 with 36 edges

As Theorem 4.1.1 provides a bound for general posets, we proprose that this bound can be improved for concept lattices by taking advantage of the added structure. One observation is that finite lattices display a very natural separation into pieces. These decompositions come in two forms, chains and antichains, and are characterized by the work of Dilworth and Mirsky, respectively.

Dilworth proved Theorem 4.1.3 in 1950, which relates the width of a poset to its partition into chains.

Theorem 4.1.3. *[4, 7] Let P be a partially ordered set and n a natural number. If P possesses no antichain of cardinality $n+1$, then it can be expressed as the disjoint union of n chains.*

Mirsky proved Theorem 4.1.4 in 1971 as a response to Theorem 4.1.3.

Theorem 4.1.4. *[7] Let P be a partially ordered set, and m a natural number. If P has no chain of cardinality $m+1$, then it can be expressed as the disjoint union of m antichains.*

Mirsky's theorem says that the minimum number of antichains into which a lattice may partitioned is given by the height the lattice. These results lead us to offer the following conjecture in which we express an upper bound in terms of antichains.

Conjecture 4.1.5. *Let E be the set of edges in Hasse diagram of $\mathcal{L} = \langle \mathcal{C}, \supseteq \rangle$ and enum($\mathcal{C}$) be an enumeration of $\mathcal{C}$. There exists a partition $\{S_1, S_2, \ldots, S_m\}$ of enum($\mathcal{C}$) into m antichains, where m is the height of the lattice $\mathcal{L}$, such that*

$$|E| \leq \sum_{i=1}^{m-1} |S_i||S_{i+1}|.$$

We demonstrate Conjecture 4.1.5 in the following example.

Example 4.1.6. *We use the same lattice as in Example 4.1.2. One way to partition is*

$$S_1 = \{\emptyset\}$$

$$S_2 = \{a, b, c, d, e, f, g, h, k, l\}$$

$$S_3 = \{ak, dg\}$$

$$S_4 = \{cef, aeg, bhk, \}$$

$$S_5 = \{bghk, bchk, adkl\}$$

$$S_6 = \{abcdefghkl\}$$

We then have $|S_1||S_2| + |S_2||S_3| + |S_3||S_4| + |S_4||S_5| + |S_5||S_6| = 10 + 20 + 6 + 9 + 3 = 48$. The actual number of edges for this lattice is 36. This partition gives a better estimate than Wiseman's bound. Note that the partition that produces the upper bound is not unique. Observe the following partition.

$$S_1 = \{\emptyset\}$$

$$S_2 = \{a, b, c, d, e, f, g, h, k, l\}$$

$$S_3 = \{ak, dg, cef, aeg, bhk\}$$

$$S_4 = \{adkl, bghk, bchk\}$$

$$S_5 = \{abcdefghkl\}$$

We then have $|S_1||S_2| + |S_2||S_3| + |S_3||S_4| + |S_4||S_5| = 10 + 50 + 15 + 3 = 78$.

Attempts have been made to prove Conjecture 4.1.5, but no proof has been found. However, we observe that the conjecture is true for a special class of lattices.

Definition 4.1.7. *A poset is stratified if there exists a partition $S_1, \ldots, S_t$ such that each S_i is an antichain and every cover relation $x \prec y$ has $x \in S_i$ and $y \in S_{i+1}$ for some $i \in \{1, \ldots, t-1\}$.*

The conjecture is true for stratified lattices since the maximum number of possible edges between members of S_i and S_{i+1} is $|S_i||S_{i+1}|$.

4.2 Experiments

All implementation was written in Java and carried out on a Intel Duo Core CPU 2.66 GHz machine with 4 GB RAM running in Windows. Generation of contexts and retrieval of concepts were achieved via algorithms that were conceived and implemented in [8].

The author of [8] uses the BitSet class in Java to represent the extent and intent of formal concepts. In this representation, a bit cooresponds to an object or attribute and has value 1 if the object or attribute is present and 0 otherwise. In our implementation of iPred, we kept this convention in order to utilize the bitwise operations as a way of performing the set operations used in the Algorithm 1. Our implementation takes as input a list of intents and returns a list of pairs with each intent as the first component and a list of the members that cover that intent as the second component. The reader may refer to the appendix for source code.

Let ρ denote the density of a cross table, calculated as the ratio of crosses to total available entries. With a fixed number of objects and attributes, we calculate the edge frequency distribution from a total of n randomly generated cross tables with density ρ. We categorize edges by looking at the set difference between intents:

$$\nu_i^\rho = |\{\{u, v\} : |u - v| = i\}|$$

where $u, v \in \mathcal{C}$ and $u \prec v$ for a fixed density ρ.

For each lattice $\mathcal{L}_j^\rho$, $j = 1, 2, \ldots, n$, we have $\{\nu_{i,j}^\rho\}_{i=1}^{|M|-1}$. Then the total number of edges in each density is

$$K^\rho = \sum_{i=1}^{|M|-1} \sum_{j=1}^{n} \nu_{i,j}^\rho.$$

Note that for a fixed i, if f_i is the frequency of edges in $\{\nu_{i,j}^\rho\}_{j=1}^n$, then

$$F_i^\rho = \frac{f_i}{K^\rho} = \frac{1}{K^\rho} \sum_{j=1}^{n} \nu_{i,j}^\rho$$

is the percent of all edges between intents that differ by i attributes, from our n randomly generated contexts.

In our experiment, $|M| = 10$, $n = 10^6$, and ρ ranged from five percent to ninety-five percent incremented by five. Figures 4.3-4.21 display $\{F_i\}_{i=1}^{9}$ from the experiment for each density ρ.

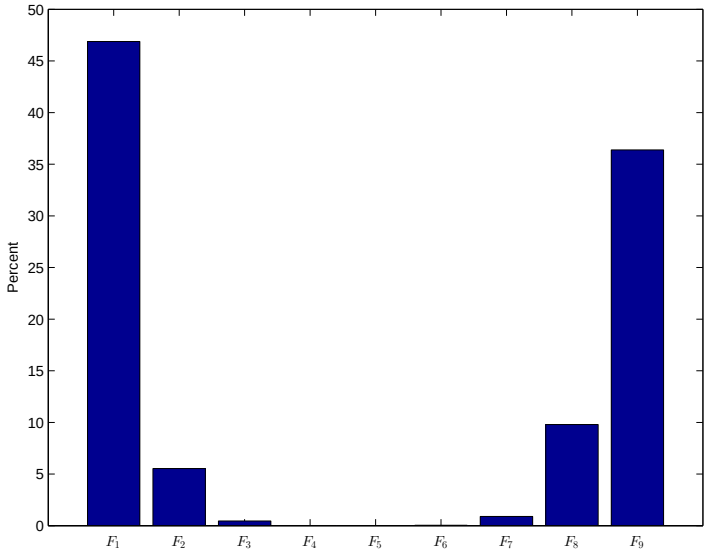


Figure 4.3: $\rho = .05$

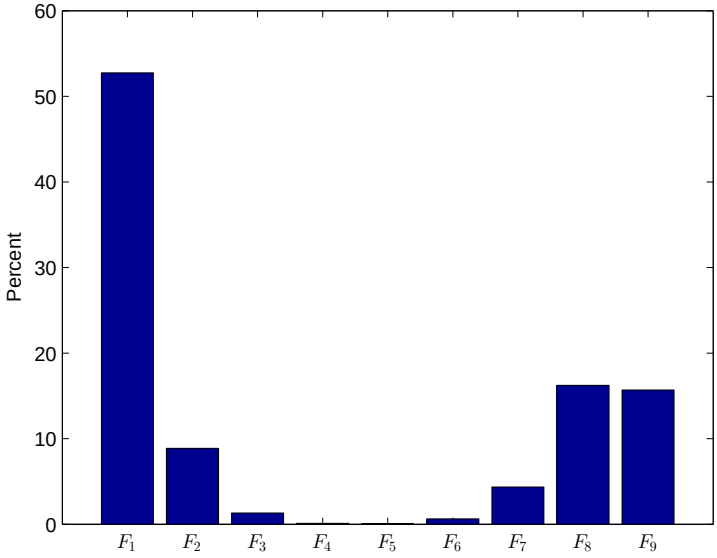


Figure 4.4: $\rho = .10$

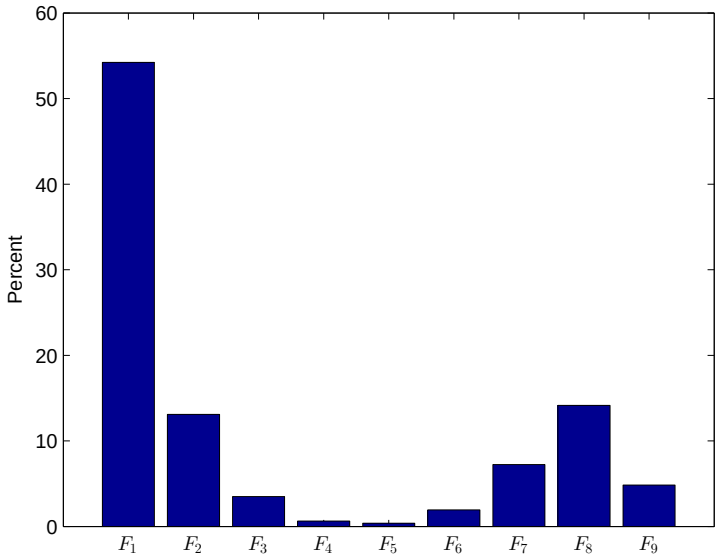


Figure 4.5: $\rho = .15$

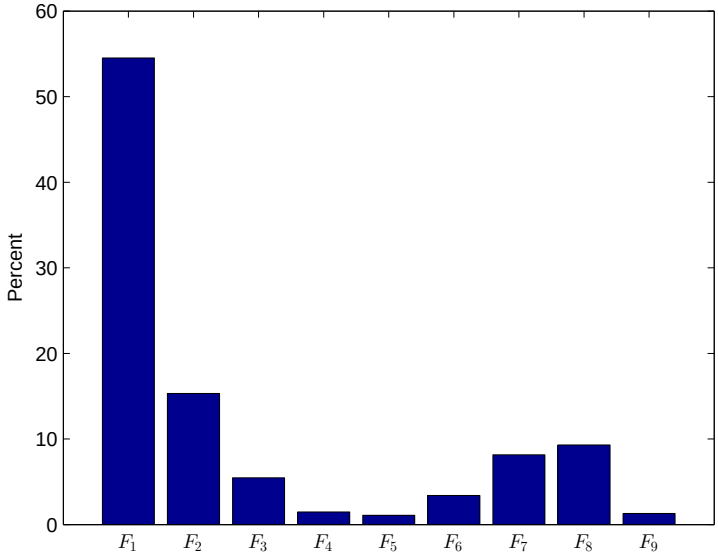


Figure 4.6: $\rho = .20$

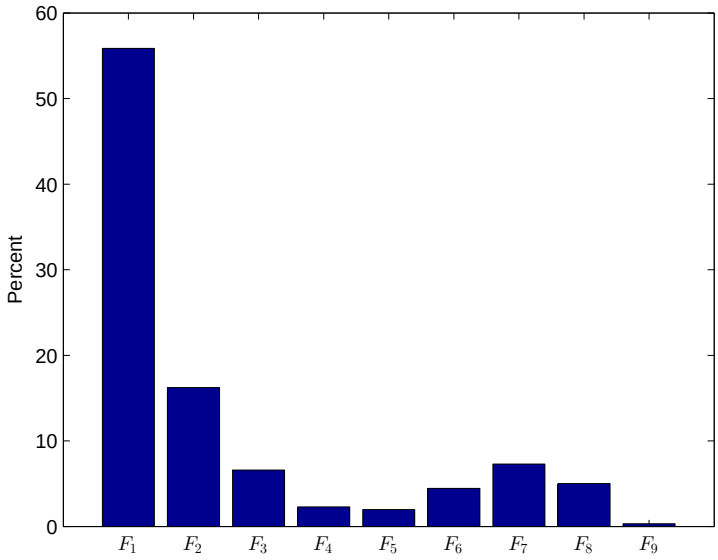


Figure 4.7: $\rho = .25$

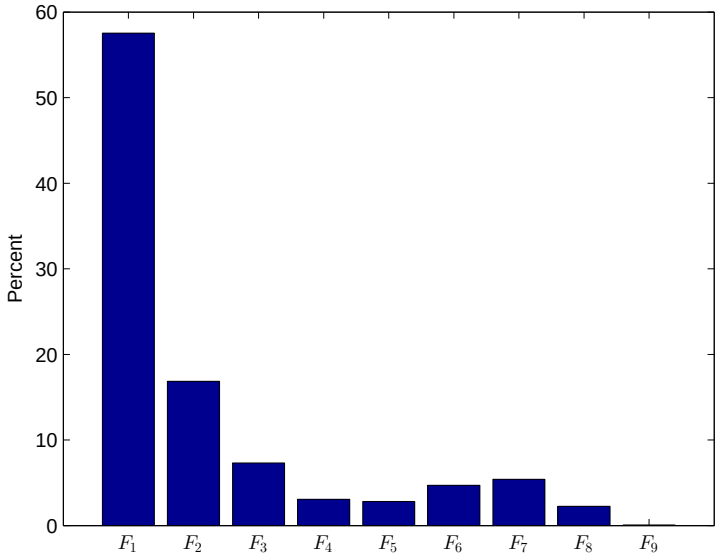


Figure 4.8: $\rho = .30$

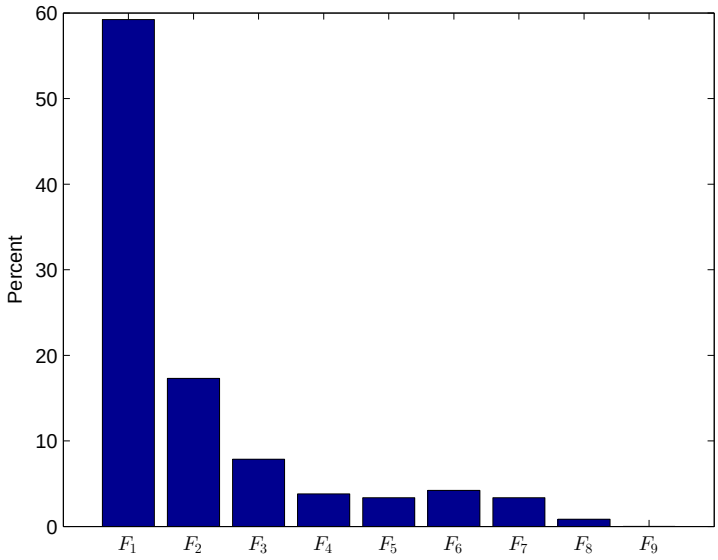


Figure 4.9: $\rho = .35$

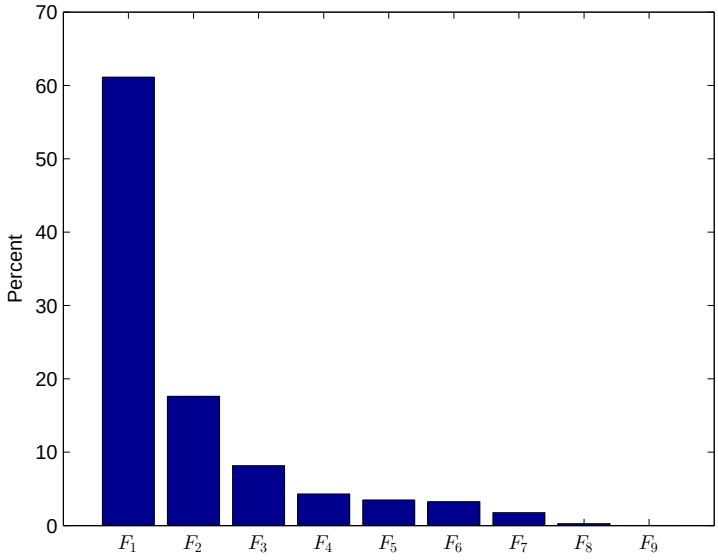


Figure 4.10: $\rho = .40$

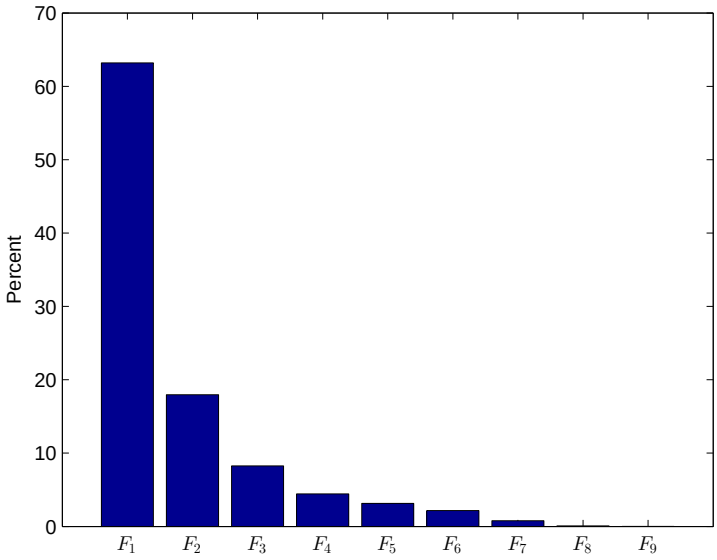


Figure 4.11: $\rho = .45$

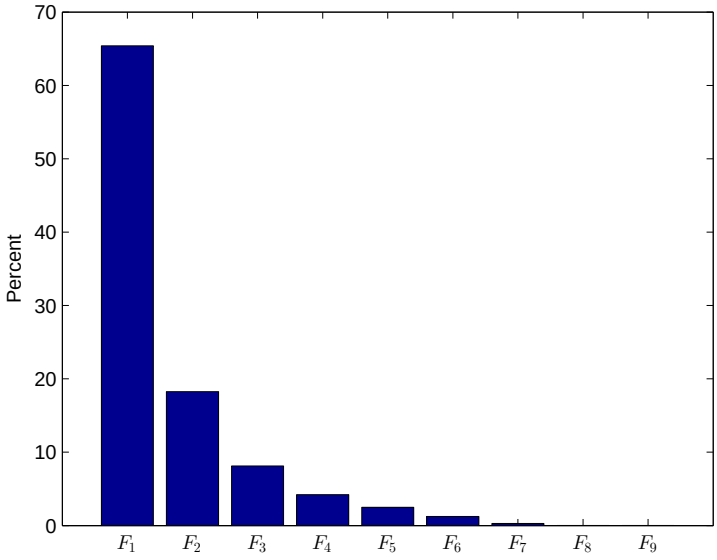


Figure 4.12: $\rho = .50$

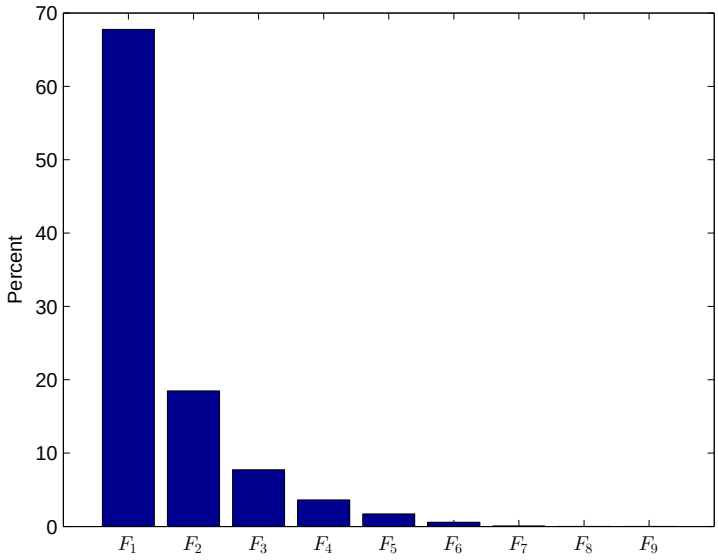


Figure 4.13: $\rho = .55$

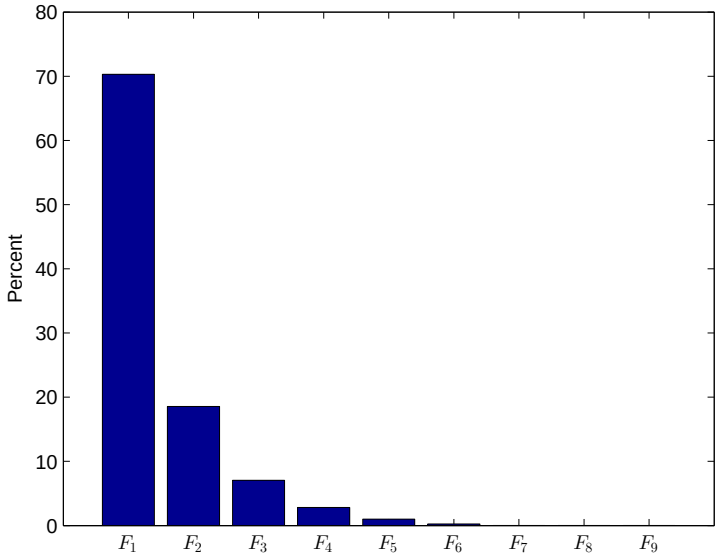


Figure 4.14: $\rho = .60$

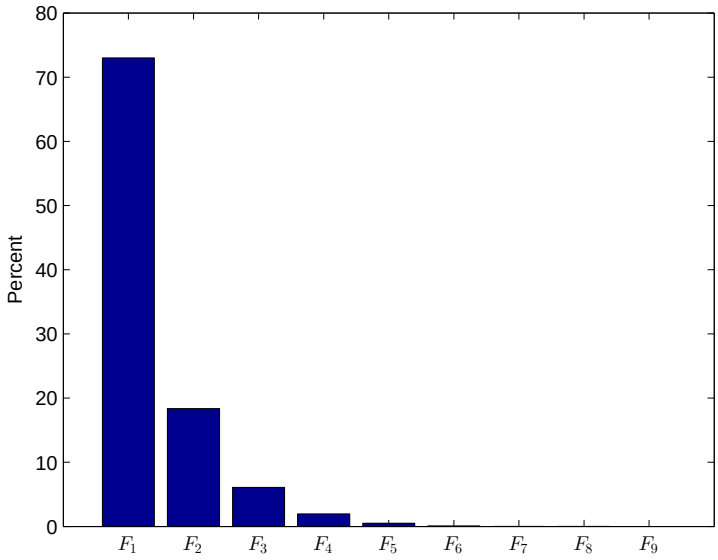


Figure 4.15: $\rho = .65$

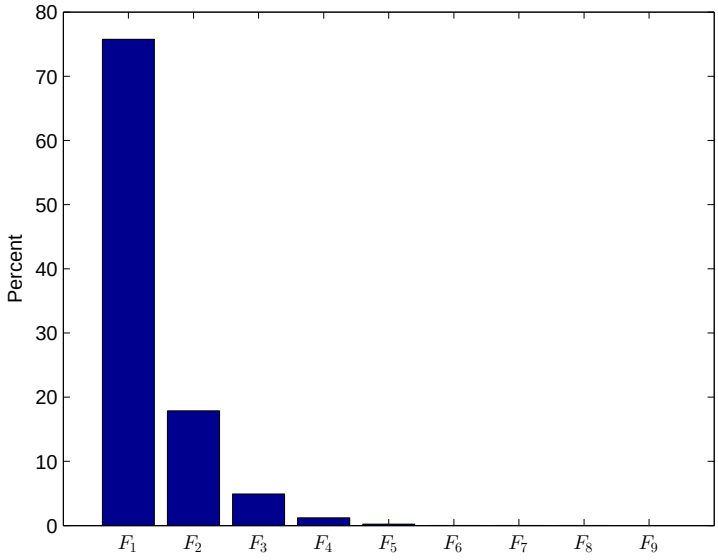


Figure 4.16: $\rho = .70$

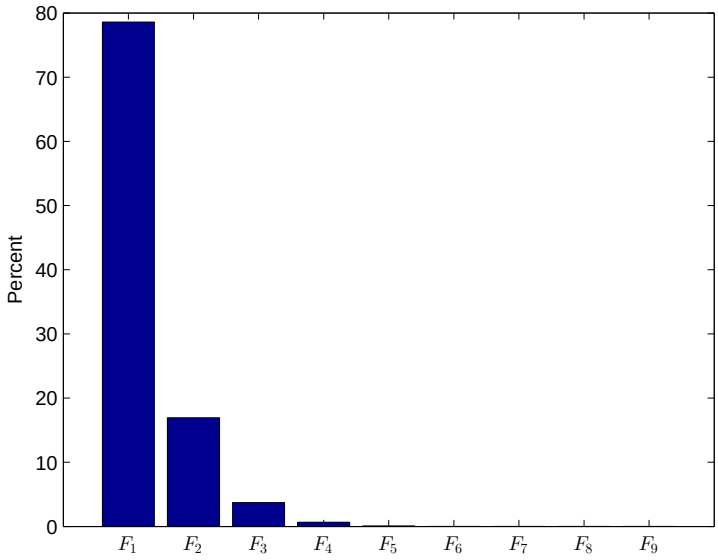


Figure 4.17: $\rho = .75$

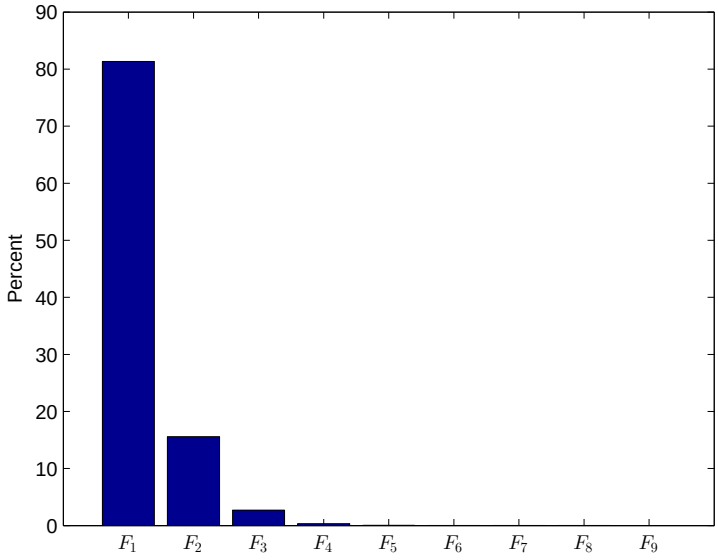


Figure 4.18: $\rho = .80$

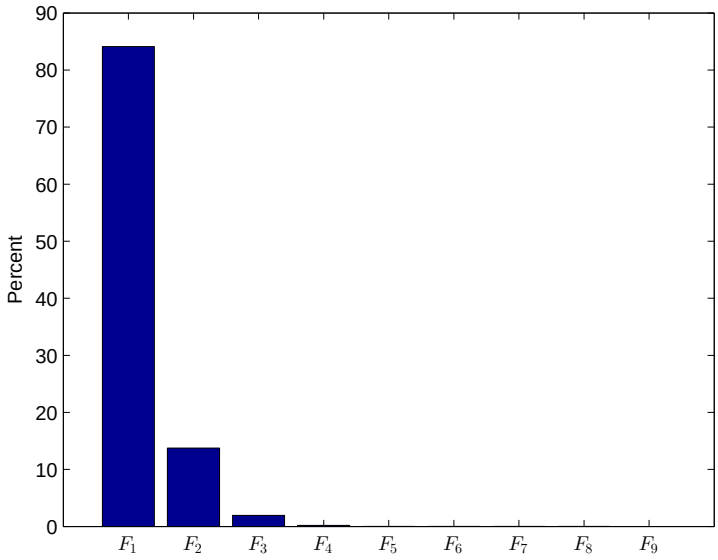


Figure 4.19: $\rho = .85$

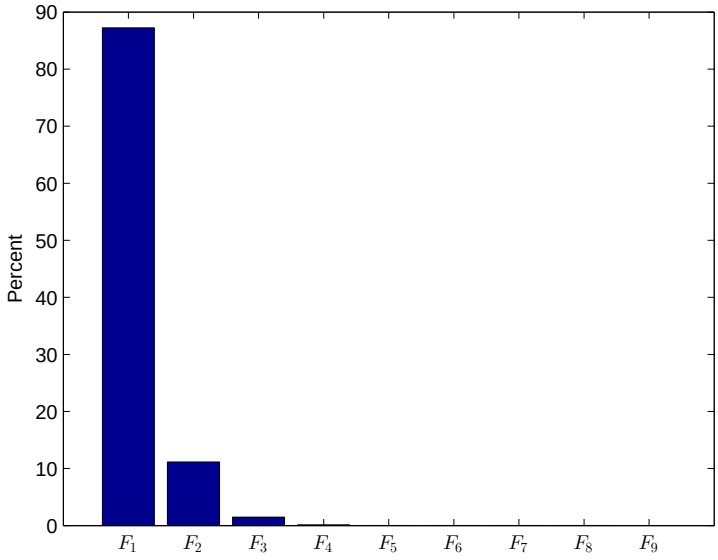


Figure 4.20: $\rho = .90$

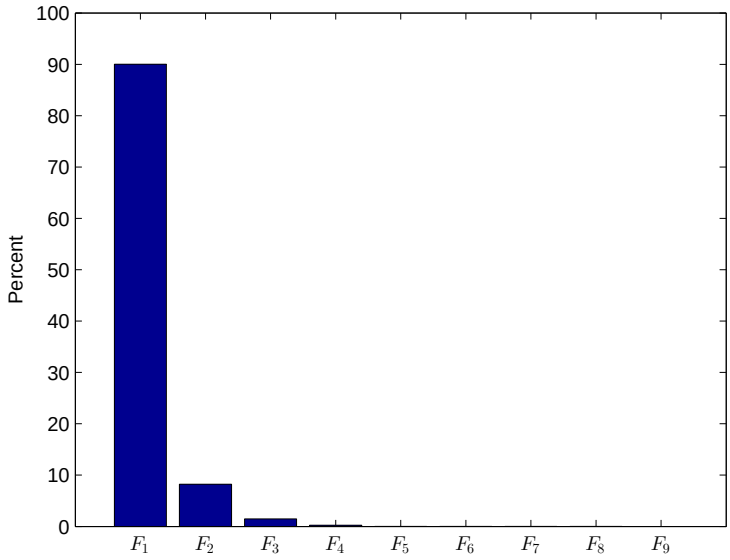


Figure 4.21: $\rho = .95$

By inspection, one can observe a general decline in the percentage of edges associated with ν_9 and a clear increase in percentage of those associated with ν_1 as the ρ increases. One might expect this outcome since, in most cases, a greater density should yield a greater number of formal concepts. For each density ρ, we see that F_1 is the highest percentage. This should be of no surprise for the following reason: If two concepts are related, then their respective intents are related by inclusion. Thus, whenever two related intents $B_1 \subseteq B_2$ differ by exactly one attribute, it is impossible for there to exist a set B_3 such that $B_1 \subseteq B_3 \subseteq B_2$. In this case there must be an edge between two such intents. For a summary of the data collected in this experiment, the reader may refer to Table 4.1.

Table 4.1: Summary of experimental data

ρ	Total Number of Concepts	Average Number of Concepts	ν_1	ν_2	ν_3	ν_4	ν_5	ν_6	ν_7	ν_8	ν_9	Total Number of Edges
.05	5739807	5.7398	3348927	394781	32276	1514	64	2599	63819	699583	2598995	7142558
.10	8589782	8.5898	6255424	1051068	156383	13829	7011	75585	517099	1924997	1859880	11861276
.15	11583224	11.5832	9613172	2322939	619640	113149	68547	345096	1282580	2507654	858241	17731018
.20	14799911	14.7999	13436853	3772887	1348092	362148	267093	838483	2006026	2291684	320057	24643323
.25	18418405	18.4184	18250406	5301706	2154661	745180	645217	1451859	2381704	1638418	103353	32672504
.30	22622120	22.6221	24447734	7159964	3111246	1305274	1196811	1996811	2292919	956855	28794	42496408
.35	27538729	27.5387	32337427	9441559	4293032	2074306	1838932	2299367	1836603	463214	6990	54591430
.40	33329988	33.3300	42444431	12236690	5667985	2988812	2414001	2259266	1230088	186867	1530	69429670
.45	40138174	40.1382	55321752	15721659	7224162	3890118	2750705	1894025	688742	61301	292	87552756
.50	48082060	48.0821	71624569	20001273	8899839	4606668	2718085	1348909	316448	16803	50	109532644
.55	57207816	57.2078	91991799	25082993	10505838	4926557	2315542	798379	118726	3693	5	135743532
.60	67427646	67.4276	116916216	30842989	11733987	4701361	1673043	390941	36861	684	2	166296084
.65	78258248	78.2582	146072526	36764146	12184405	3920290	1011525	158020	9777	126	0	200120815
.70	88564992	88.5650	177404097	41845343	11528671	2817467	512674	55852	2515	22	0	234166641
.75	96279203	96.2792	205891597	44397392	9758877	1738690	224977	18311	657	5	1	262030507
.80	98174476	98.1745	221876942	42549867	7358283	941514	90420	5938	191	1	0	272823156
.85	90710862	90.7109	212763465	34737109	4934048	477811	37815	2483	89	1	0	252952821
.90	73412180	73.4122	173397618	22129224	2939438	279995	22034	1556	82	3	0	198769950
.95	52923561	52.9236	119826089	10965744	1955821	333773	40408	3650	267	9	0	133125761

4.3 Discussion

Proving Conjecture 4.1.5 for non-stratified lattices has proven to be a difficult problem. It is not clear how to determine an optimal arrangement for the S_i in the partition so that the sum given in the conjecture produces a true upper bound. Furthermore, since a partition of a lattice into antichains is not unique, the problem of constructing antichains of a desirable size is non-trivial.

Another question arises regarding the application of Conjecture 4.1.5. If the height of the lattice gives the minimum size of a partition into antichains, then how does one obtain the height before doing any computation? Our explanation relies on the notion that the frequency distributions from the experiment provide an approximation for the probability that an edge goes between intents that differ by i attributes given a density ρ.

Conjecture 4.1.5 says that there exists a partition of the lattice into m antichains such that the number of edges satisfies the bound. The height of any concept lattice is at most $|M|+1$ where M is the set of attributes. The data suggest that, for sufficiently large ρ, it is highly likely for there to be an edge between concepts whose intents differ by exactly one attribute. If that is the case, then the height, m, is likely to equal $|M|+1$ where M is the set of attributes. If the height is equal $|M|+1$, then the partition is easily constructed by grouping together concepts having the same number of attributes. And, we are assured that the subsets in the partition are indeed antichains since any two concepts having the same number of attributes cannot be related. On the other hand, the densities for cross tables that represent real data are generally around 25 percent. Hence, the above explanation may not be as useful in practice.

The partition constructed by grouping together intents of equal cardinality does not always produce an upper bound. Consider, for example, the lattice in Figure 4.22 with 6 edges. If $S_1 = \{\emptyset\}$, $S_2 = \{a, b\}$, $S_3 = \{cd\}$, and $S_4 = \{abcd\}$, then the estimate from Conjecture 4.1.5 is 5, which is less than 6. On the other hand, if $S_1 = \{\emptyset\}$, $S_2 = \{a, b, cd\}$, and $S_3 = \{abcd\}$, then the estimate is exactly 6 edges.

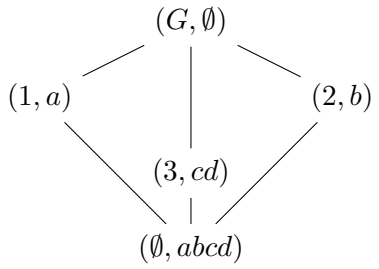


Figure 4.22: Lattice with exactly 6 edges

Although this approach to partitioning does not guarantee an upper bound, it is easily implemented. The data also suggest that exceeding the resulting estimate is unlikely for ρ greater than 20 percent. As part of the experiment, we also implemented this method of partioning and recorded the estimated number of edges and the actual number of edges for each sample. We then counted the number of lattices for which the actual number of edges exceeded the estimated number of edges in that lattice. Table 4.2 provides a summary.

Table 4.2: With larger densities, it is less likely for the number of edges to exceed the estimate

ρ	Maximum Estimate	Maximum Actual	Minimum Estimate	Minimum Actual	Percentage of Samples where Actual exceeds Estimate
.05	10	10	2	2	33.1504
.10	35	20	3	4	18.3026
.15	71	30	5	6	7.1725
.20	99	44	8	9	1.5887
.25	150	58	12	12	0.1286
.30	221	74	15	13	0.0040
.35	336	101	20	18	0.0003
.40	484	133	27	21	0
.45	677	160	30	23	0
.50	1108	231	43	31	0
.55	1565	287	43	33	0
.60	2444	367	43	31	0
.65	4112	527	47	33	0
.70	6765	716	29	25	0
.75	10435	941	27	20	0
.80	17543	1278	25	23	0
.85	33255	1888	15	13	0.00030
.90	42279	2251	8	8	0.00310
.95	10489	948	2	2	0.0830

One may observe from the data in Table 4.2 that, by partitioning according to number of attributes, the frequency with which the number of edges exceeded the estimate was small for most densities. These occurances happened less than one percent of the time with ρ as low 25 percent.

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

Some questions remain. Proving Conjecture 4.1.5 remains an open problem. Future work could require the use of techniques from extremal graph theory, an area not well-explored in this work. The bound given in [9] is true for general posets, but we wish to use the structure of the concept lattice to obtain a more useful bound. Every concept lattice is a complete lattice, and every complete lattice is isomorphic to a concept lattice [3, 5]. Can we study this open problem from the perspective of complete lattices rather than general posets?

Related to this is the problem of constructing the antichains in the partition and determining which partitions give the best upper bound. We observed that the partition for finding the upper bound on the number of covering relations is not always unique. Under what conditions is the partition unique? This is could also be re-stated as how to decompose $|\mathcal{L}|$ into a sum $\sum_{i=1}^{m} a_i$ such that $\sum_{i=1}^{m-1} a_i a_{i+1}$ is a tight upper bound, with the contraints that $a_1 = a_m = 1$ and $a_i \geq 1$ for $2 \leq i \leq m-1$. Wiseman treated a similiar question briefly in [9], but these open problems are not addressed anywhere else in the literature.

Other future work includes determining whether it is useful to have a lower bound. The trivial answer is to use the height of the lattice. However, it is not clear how to estimate the height before computing any edges. Future work includes finding ways to accurately estimate the height of a concept lattice. This might involve taking advantage of its special structure. Such estimates would also be useful since Conjecture 4.1.5 uses the height of the lattice to determine the size of the partition into antichains.

The algorithms in [8] generate information about covering relations that goes unused. Can we use this information to build on those algorithms by incorporating an efficient component that provides the covering relations explicitly? If so, could these processes be parallelized?

Lastly, we wish to test these results using real data. As our results are based on randomly generated contexts, we have less information about structure. With real data, however, one may gain knowlegde about the structure of the system to be modeled before using formal concept analysis. Or, given prior insights about the system, one could make appropriate assumptions about its structure and the relationships involved.

APPENDIX. ADDITIONAL MATERIAL

Java Implementation of iPred

The following is our source code for the iPred algorithm.

```
public void runModifiediPred(ArrayList<BitSet> c) {
    int numberOfConcepts = c.size();
    ArrayList<Cover> adjacencyList = new ArrayList<Cover>();
    ArrayList<BitSet> delta = new ArrayList<BitSet>(numberOfConcepts);
    ArrayList<BitSet> border = new ArrayList<BitSet>();
    ArrayList<BitSet> cand = new ArrayList<BitSet>();

    BitSet d = new BitSet();  //Initializing delta
    for (int i=0; i<numberOfConcepts; i++){
      delta.add(d);
    }
    border.add(c.get(0));
    for (int i = 1; i < numberOfConcepts; i++){
      BitSet ci = c.get(i);
      if (border.contains(ci)==true){
        border.remove(ci);  //Do not want ci in the border, otherwise edge
    count is inaccurate.
      }
      cand.clear();  //forming the candidate set
      for (BitSet ctilde : border){
        BitSet temp = (BitSet) ci.clone();
        temp.and(ctilde);
        if (cand.contains(temp) == false){
          cand.add(temp);
        }
      }
```

```
26        ArrayList<BitSet> upperCover = new ArrayList<BitSet>();
27
28        for (BitSet ctilde : cand){
29          int index = 0;
30          for (BitSet elt : c){
31            if (elt.equals(ctilde) == true){
32              index = c.indexOf(elt);
33            }
34          }
35          BitSet dctilde = (BitSet) delta.get(index).clone();
36          if (border.contains(ctilde) || (ci.cardinality() - ctilde.cardinality()
    ) == 1 || dctilde.intersects(ci) == false){
37            upperCover.add(ctilde);
38            BitSet temp1 = (BitSet) ctilde.clone();
39            temp1.xor(ci);
40            dctilde.or(temp1);
41            delta.set(index, dctilde);
42            border.remove(ctilde);
43          }
44        }
45        Cover cover =  new Cover(ci, upperCover);
46        adj.add(cover);
47        border.add(ci);
48      }
49      System.out.println(adjacencyList);
50    }
51  }
```

BIBLIOGRAPHY

[1] J. Baixeries, L. Szathmary, P. Valtchev, R. Godin. Yet a Faster Algorithm for Building the Hasse Diagram of a Concept Lattice. *ICFCA*, LNAI 5548, 162-177, 2009.

[2] C. Carpineto, G. Romano. *Concept Data Analysis*, Wiley, England, 2004.

[3] B. Davey, H. Priestly. *Introduction to Lattices and Orders*, 2nd ed. Cambridge University Press, Cambridge, 2002.

[4] R. P. Dilworth. A Decomposition Theorem for Partially Ordered Sets. *Annals of Mathematics*, Second Series, Vol. 51, No. 1, 161-166. Annals of Mathematics, http://www.jstor.org/stable/1969503. Accessed April 29, 2015.

[5] B. Ganter, R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin, Heidelberg 2012.

[6] B. Martin, P.W. Eklund. From Concepts to Concept Lattice: A Border Algorithm for Making Covers Explicit. *ICFCA*, LNCS 4933, 78-89. Springer, Heidelberg 2008.

[7] L. Mirsky. A Dual of Dilworth's Decomposition Theorem. *The American Mathematical Monthly*, Vol. 78, 1971, 876-877. Mathematical Association of America, http://www.jstor.org/stable/2316481. Accessed April 6, 2015.

[8] V. Sukhinin. Ph.D Dissertation (In Progress), Iowa State University, 2015.

[9] J. Wiseman. Extremal Hasse Diagrams. *Congressus Numerantium*, Vol. 77, 1990, 195-198.