

Automated Layout of Concept Lattice Using Layer Diagrams and Additive Diagrams

Richard Cole
r.cole@gu.edu.au
Griffith University, Gold Coast, Australia

Abstract

Drawings of concept lattices provide the most common mechanism for the communication of structure extracted from data via the process of formal concept analysis. To communicate structure, diagrams of concept lattices are usually made to be additive. Additive diagrams however suffer from some unaesthetic properties. Alternatively a common graph drawing approach to the layout of partially ordered sets (of which lattices are a sub-class) is via a layer diagram. This paper presents a mechanism for the layout of concept lattices that combines ideas from both additive and layer diagrams. These new hybrid diagrams preserve the structural aspects of additive diagrams while achieving the aesthetic quality of layer diagrams. Furthermore a search method is presented that optimizes the layout with respect to structural and aesthetic objective functions.

1 Introduction

Previous approaches to the layout of concept lattices can be divided into two groups; automatic methods and user assisted methods. The main automatic method is via chain-decomposition[5] and is well suited to distributive lattices. User assisted methods include the geometric method[8] and direct manipulation of an additive diagram[10]. This paper presents a method for the automated layout of non-distributive lattices that merges ideas of additive diagrams with the layered diagram approach.

Section 2 briefly recalls the basic definitions of formal concept analysis. Sections 3 and 4 explain the nature of additive diagrams and layered diagrams respectively. Section 5 presents the hybrid diagram and a search algorithm for finding an optimal hybrid diagram. The results, in terms of diagrams and efficiency, are presented in Sections 6 and 7. Section 8 presents some conclusions and an outlook for further work.

2 Formal Concept Analysis

Formal Concept Analysis is a successful technique of data analysis that has been applied in many disciplines from understanding building regulations[3], to political science[9] and psychology[7]. In formal concept analysis data is formulated into binary tables called *formal contexts*. A *formal context* is a triple (G, M, I) where G is a set of objects, M is a set of attributes, and I is a relation saying which objects have which attributes. Fig. 1 shows a context represented by a *cross-table*. The attributes in the table are labelled A to I and the objects are labelled **leech** to **maize**.

	A	B	C	D	E	F	G	H	I
leech	x	x	x	.	.
bream	x	x	x	x	.
frog	x	x	x	.	.	.	x	x	.
dog	x	.	x	.	.	.	x	x	x
weed	x	x	.	x	.	.	x	.	.
reed	x	x	x	x	.	x	.	.	.
bean	x	.	x	x	x
maize	x	.	x	x	.	x	.	.	.

Figure 1: Example Formal Context. Attributes are: A. needs water to live, B. lives in water, C. lives on land, D. needs chlorophyl, E. has two seed leaves, F. has one seed leaves, G. can move around, H. can move around, I. suckles its young.

A concept lattice is calculated from a formal context and consists of pairs (A, B) called concepts, where A is called the extent of the concept and is a subset of G , and B is called the intent of the concept and is a subset of M . In order to be a concept of a context (G, M, I) , a pair (A, B) , must additionally satisfy:

$$A' = B \quad \text{and} \quad B' = A \quad (1)$$

where the derivation operation $'$ is given by:

$$A' = \{m \in M \mid \forall g \in A : (g, m) \in I\} \quad (2)$$

$$B' = \{g \in G \mid \forall m \in B : (g, m) \in I\} \quad (3)$$

A specialization ordering is defined over the concepts of a formal context. A concept (A, B) is more specific than another concept (C, D) if $D \subseteq B$. This same ordering is also obtained by taking $A \subseteq C$. The set of all concepts of a formal context, together with this specialization ordering forms a lattice, called the *concept lattice* of the context. Fig. 2 is a diagram of the concept lattices calculated from the context in Fig. 1. The intent of a concept, x , may be obtained from the diagram by collecting all attributes attached to concepts in the upset of x . Similarly the extent can be determined by collecting all object labels attached to concepts in the down-set of x .

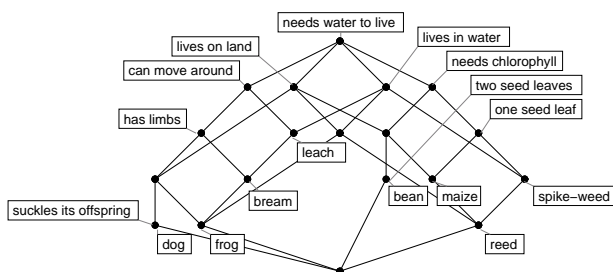


Figure 2: Concept Lattice of the Context in Fig. 1

3 Additive Line Diagram

The concept of an additive line diagram was introduced by Ganter and Wille [2]. For a lattice L we introduce a representation set X , and a function $\text{rep} : L \rightarrow \wp(X)$, that maps each element of the lattice to a subset of a representation set called X . For $\text{rep} : L \rightarrow \wp(X)$ to be a valid representation of L the following restriction is imposed:

$$\forall x, y \in L : x \leq y \text{ implies } \text{rep}(y) \subseteq \text{rep}(x)$$

Each element of the representation set X is mapped to a vector in \mathbb{R}^2 . The position on the page for each element of L is then given by the function $\text{pos} : L \rightarrow \mathbb{R}^2$ where:

$$\text{pos}(a) = \sum_{x \in \text{rep}(a)} \text{vec}(x) + n \quad (4)$$

and n is a vector used to displace the diagram on the page.

Commonly for the layout of concept lattice the attribute set is used as the representation set. In this case we have:

$$\text{rep}(x) = \text{Int}(x) \quad (5)$$

where $\text{Int}(x)$ is the intent of the concept x .

Using attributes for the representation set has the advantage that in order to determine which attributes are in the intent of a concept one can decompose the position of the concept into a sum of the component vectors. Additive diagrams based on attributes have the disadvantage however that some concepts are *distended*. An element is distended if the vertical distance from that concepts to all of its parents is large compared with the average vertical distance between concepts in the diagram. Consider for example the additive diagram of $M3$ shown in Fig. 3(a). The bottom element in this diagram is distended because its intent, $\{A, B, C\}$, contains three attributes while the intents of the parents each contain only a single attribute.

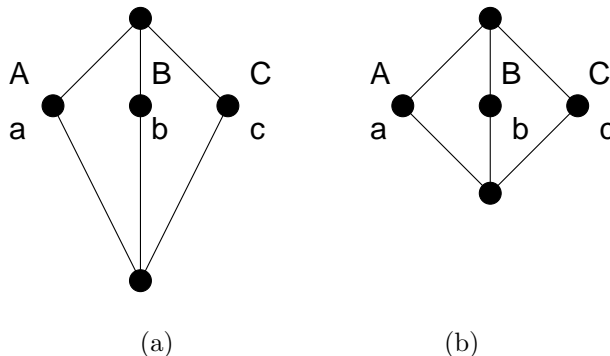


Figure 3: Additive Diagram of the lattice $M3$

One possible resolution is to allow objects to be members of the the representation set, i.e: $X = G \dot{\cup} M$. Then the representation function is given by:

$$\text{rep}(a) = (G \setminus \text{Ext}(a)) \dot{\cup} \text{Int}(a) \quad (6)$$

Using this representation it is possible to obtain the diagram in Fig. 3(b). In general however for more complex lattices, even with the use of object vectors, the optimum diagram may not necessarily be drawn. In this case, a non-additive diagram is often drawn by hand using the geometrical technique[8].

4 Layered Diagram

A common approach to the drawing of a partially ordered set is to use a layered diagram[1]. The layer diagram approach can be divided into three steps:

1. Layer Assignment; each vertex is assigned to a horizontal layer.
2. Crossing Reduction; vertices within a layer are ordered to reduce edge crossings.
3. Horizontal Coordinate Assignment; an x coordinate is assigned to each vertex.

Two popular approaches to layer assignment are: longest path layering, and Coffman-Graham layering. Coffman-Graham layering may be employed to reduce the width of a layered diagrams by restricting the maximum number of vertices allowed in any layer. Commonly after layer assignment dummy vertices are added for lines which cross layers. The horizontal coordinate assignment is performed with the objective of minimizing degree of bend at each dummy node.

5 Hybrid Diagram

A hybrid diagram of a concept lattice may be constructed by using layer assignment to determine the vertical positioning of concepts and using the additive method to determine the horizontal placement of concepts. The algorithm may be divided into two steps:

1. Layer assignment; longest path layering is used.
2. Determine additive vectors; each attribute is assigned a horizontal vector and these sum to give the horizontal position of each concept.

Longest path layering was chosen in favor of Coffman-Graham layering because an inappropriate promotion of a concept to the next layer can seriously degrade the quality of the resulting diagram. In addition there was no clear criteria to decide for a given lattice what the maximum number of concepts allowed in each layer should be.

A set of additive vectors are chosen so as to maximize a set of objective functions applied to the resulting diagram. No dummy nodes are used because the recognition of structure in a concept lattice requires the reader to visually decompose displacement vectors between concepts and this task is made difficult by bends which distract the viewer and may give a false impression.

In order to further explain the horizontal positioning of concepts the following notation is introduced.

Let (G, M, I) be a formal context, and L be the concept lattice resulting from this context. The representation function $\mathbf{rep} : L \rightarrow \wp(M)$ is given by:

$$\mathbf{rep}(a) = \text{Int}(a)$$

The horizontal positions are then determined by the function, $\mathbf{vec} : M \rightarrow \mathbb{Z}$, which now maps from attributes to integer values. This function can be encoded by a vector; let $M = \{m_1, \dots, m_k\}$ and $V \in (\mathbb{Z}_n)^k$. Then the function $\mathbf{pos} : L \rightarrow \mathbb{Z}$ is given by:

$$\mathbf{pos}(a) = \sum_{i=1}^k v_i c_{i,a}$$

where $V = (v_1, \dots, v_k)$ and $c_{i,a} = 1$ if $m_i \in \text{Int}(a)$ and 0 otherwise.

For a hybrid diagram to be satisfactory we require that no two concepts assigned to the same layer have the same x -coordinates and that no edge crosses the coordinates of a concept. This requirement was formulated as a constraint satisfaction problem and backtracking was employed to produce a list of satisfactory diagrams. Each satisfactory diagram is then assessed for diagram quality using the following objective functions.

1. Symmetry; count the number of pairs of siblings which are in the same layer and are symmetrical about a vertical axis running through their common parent. Concepts directly below their parent contribute to this measure.
2. Minimize lines; each edge is considered as a vector, and the total number of distinct vectors is minimized.
3. Maximize chains; count the number of grandparent, parent, child combinations in which the grandparent-parent vector is the same as the parent-child vector.

Often these objective functions will conflict. For example a diagram that optimizes symmetry may not optimize the total number of lines. Consider a vector $A = (\mathbb{Z})^n$ in which each component a_i is the result of applying the i 'th objective function to the diagram. We call such a vector an *objective vector*. The following ordering was used to select the best diagrams:

$$(a_1, \dots, a_n) \leq (b_1, \dots, b_n) :\Leftrightarrow \forall i \in 1 \dots n : a_i \leq b_i$$

The user is then shown all diagrams whose objective vectors are in the top of the partial order of the objective vectors of satisfactory diagrams found by local search.

5.1 Backtracking

The search for a satisfactory diagrams can be formulated as a constraint satisfaction problem[6] and solved using an algorithm based on backtracking[4].

In order explain the backtracking method we introduce a lexical ordering over the vectors used to determine diagrams. The lexical ordering is represented by a successor function $\text{succ} : (\mathbb{Z}_n)^k \rightarrow (\mathbb{Z}_n)^k$ as:

$$\text{succ}(v_1, \dots, v_i, \dots, v_k) = \begin{cases} (v_1, \dots, v_{k-1}, v_k + 1) & \text{if} \\ \quad v_k \neq n - 1 & \\ (v_1, \dots, v_i + 1, 0, \dots, 0) & \text{if} \\ \quad v_{i+1}, \dots, v_k = n - 1 & \end{cases} \quad (7)$$

The following result allow us to employ backtracking to generate partial solutions and test to see if they can form part of a whole solution.

Proposition 1 *Let $K = (G, M, I)$ be a purified context, M_s be a subset of the attributes of K , i.e. $M_s \subseteq M$, M_s'' is the set attributes, M_s with the derivation operator applied twice (see Sec.2 for a description of the derivation operator), L be the concept lattice of K , i.e. $L = \mathfrak{B}(K)$, F be the sublattice of L corresponding to a sub-context of K , i.e. $F = \mathfrak{B}(G, M_s'', I \cap (G \times M_s''))$. Let $V \subseteq \mathbb{Z}_n^k$ be a vector used to determine the horizontal coordinates of the concepts of F and L . Then if the diagram of F is unsatisfactory, so too with the diagram of L .*

This result can be used to prune the search space in the following manner. Define factor lattices F_1, \dots, F_k of L via $F_i = (G, \{m_1, \dots, m_i\}, J_i)$, with $J_i = I \cap (G \times \{m_1, \dots, m_i\})$. Determine a satisfactory diagram for F_i before elaborating the diagram to search for a satisfactory diagram for F_{i+1} . This algorithm is summarized in Fig.4.

The algorithm presented in Fig. 4 returns the first satisfactory diagram found. A small modification to this algorithm however enables each satisfactory diagram to be assessed with a number of objective functions and the best diagrams to be saved.

6 Layout Results

Two boolean lattice, three non distribute lattices, and two distributive lattices are chosen to demonstrate the diagrams produced by the hybrid approach.

```

function search(L, layer: L → ℤn)
  i ← 0
  V ← (0, ..., 0)
  while i ≠ k loop
    while ( unsat(Fi, V) ) loop
      V = incr(V, i)
    end
    i ← i + 1
  end
  return V
end

function incr((v1, ..., vk), i)
  while ( i > 1 and vi = n - 1 ) loop
    vi ← 0
    i ← i - 1
  end
  return (v1, ..., next(vi), ..., vk)
end

function next(vi)
  if (vi < 0) then
    return -(vi + 1)
  else
    return -vi
  end
end

```

Figure 4: Search Algorithm

6.1 Boolean Lattices

The hybrid layout of boolean lattices is interesting because of two properties of boolean lattices. Firstly the hybrid layout of boolean lattices shows the limitation of hybrid diagrams. Secondly because attribute vectors for boolean lattices can be permuted without modifying the diagram they search space is very much smaller for boolean lattices than the search space of diagrams of non-distributive non-modular lattices.

Fig. 5 shows the layout of the boolean lattice B4, while Fig. 6 is the layout of the boolean lattice B6. Boolean lattice have a very large ratio of width to height which is a problem for maximum path length layer assignment. The diagram of the lattice B6 suffers from being too wide and this problem becomes much worse in larger boolean lattices.

Distributive lattices, of which boolean lattices are a subclass are generally efficiently drawn using an additive diagram as described in section 3 and a technique called chain-decomposition[5]. Hybrid di-

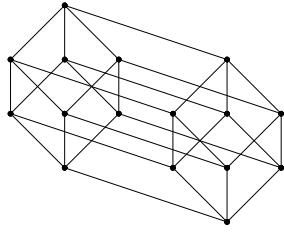


Figure 5: Boolean Lattice B4 with 4 components

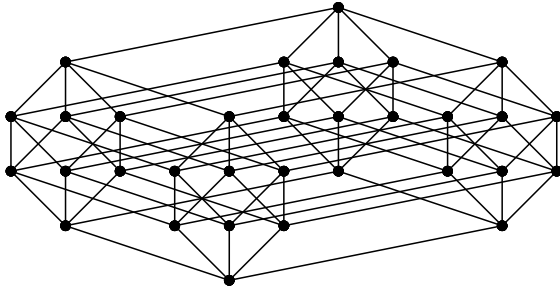


Figure 6: Boolean Lattice B5 with 5 components

agrams therefore should be employed in situations where the concept lattice is not distributive.

6.2 Non-Distributive Lattices

Diagrams of non-distributive lattices drawn with additive diagrams suffer from the problem of a distended base. The hybrid diagram solves this problem by assigning a vertical position to concepts using layer diagram method.

Fig. 7 is a diagram of a concept lattice of data concerning leisure activities supported in various national parks in California. The attribute and object labels have been left off the diagram to focus attention on layout of the lattice, rather than the placement of labels. The layout is quite good separating the regular distributive part from the non-distributive part.

The concept lattice in Fig. 8 is from data concerning different types of lakes and rivers. The lattice does not contain much regular sub-structure, although the top three layers are almost symmetrical about a vertical axis. This symmetry indicates attributes which play similar roles in the conceptual structure in the lattice.

The concept lattice in Fig. 9 is from data concern-

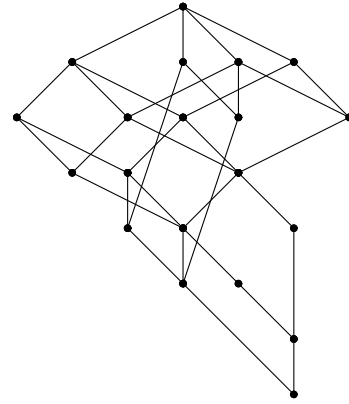


Figure 7: Concept Lattice from data about National Parks in California

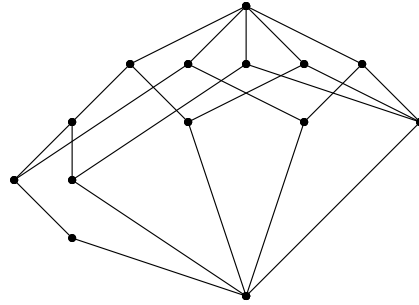


Figure 8: Concept Lattice from data about lakes and rivers

ing tasters preferences between different brands of cognac. This lattices has been previously cited as a difficult candidate for layout. The diagram in Fig. 9 shows clearly the doubled sub-lattice M3. Any lattice containing M3 as a sub-lattice is non-distributive, and thus difficult for a purely additive diagram to layout because of the problem of a distended base. The layout in Fig. 9 doesn't suffer from this problem.

6.3 Distributive Lattices

Section 6.2 showed that the hybrid diagram is suitable for non-distributive lattices, while Section 6.1 showed that hybrid diagrams are only suitable for small boolean lattices. This section examines the layout of some common distributive lattices that aren't as wide as boolean lattices.

Fig. 10 is the layout of the concept lattice generated

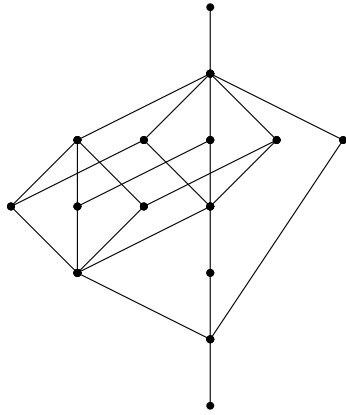


Figure 9: Concept lattice of preference judgments by tasters of cognac

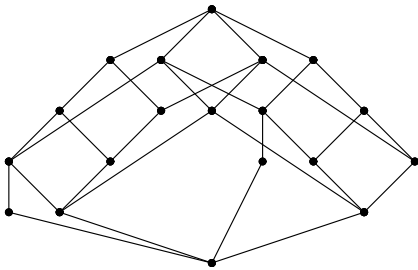


Figure 10: Concept Lattice of animals, plants and their properties mentioned in a film for children.

from the context in Fig. 1. The layout is similar in structure to that of the hand drawn diagram often used in introductory texts explaining formal concept analysis.

The diagram in Fig. 11 is good displaying a high degree of regularity. The structure of this diagram is also close to the hand-drawn of diagrams of this example lattice.

7 Efficiency Results

The backtracking algorithm was limited to considering a fixed number diagrams, commonly 1000. Since the backtracking algorithm gives preference to varying vector components corresponding to attributes at the end of the sequence m_1, \dots, m_n , rather than components the start, the ordering of the attributes was permuted. The permutations however are con-

b

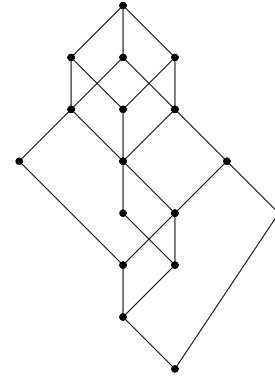


Figure 11: Concept Lattice from data on pocket knives

strained, in accordance with Proposition 1 to produce a topological sequence of attributes. The backtracking algorithm was run with up to 100 such permutations of m_1, \dots, m_n . The vector components were restricted to be integers between -10 and 10, giving 21 possible values. The number of attributes varies between 4 attributes for B4, and 9 attributes for the film example. Thus to total search space of diagrams varies between 20^4 and 20^9 diagrams.

In our experiments the algorithm produced between 1 and 5 diagrams for the non-distributive lattices, and between 1 and 100 diagrams for the distributive and boolean lattices. The large number of diagrams for the boolean and distributive cases arise because these lattices have a large number of context auto-morphisms which lead to diagrams that either look the same, or are symmetrical about a vertical axis, but are generated from different vector components.

8 Conclusion

The process of formal concept analysis produces a large number of lattices and while hand drawing of these lattices still produces the best diagrams it is a very time consuming endeavor. Automated methods for distributive lattices are used to help the formal concept analysis practitioner by suggesting a good layout. Non-distributive lattices however up until now have been difficult to layout automatically. This paper has proposed a new technique for the layout of both distributive and non-distributive lattices that

can be added to the techniques currently available for the automated layout of concept lattices.

Further work on these hybrid diagrams could compare backtracking performance with integer programming approach. Further refinement could be made to the objective functions. We intend to expose the program to a wider range of concept lattices used in practice by making the program available to practitioners of formal concept analysis and inter-operable with tools such as ANACONDA and TOSCANA [10].

[10] F. Vogt and R. Wille. TOSCANA – a graphical tool for analyzing and exploring data. *Graph Drawing*, pages 226–233, 1995.

References

- [1] G. Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph Drawing: Algorithms for the visualisation of graphs*. Prentice Hall, 1999.
- [2] B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer Verlag, 1999.
- [3] W. Kollwe, M. Skorsky, F. Vogt, and R. Wille. TOSCANA - ein werkzeug zur begrifflichen analyse und erkundung von daten. *Begriffliche Wissensverarbeitung: Grundfragen und Aufgaben*, pages 267–288, 1994.
- [4] V. Kumar. Algorithms for constraint satisfaction problems: a survey. *AI Magazine*, 13(1):32–44, 1992.
- [5] P. Luksch, M. Skorsky, and R. Wille. On drawing concept lattices with a computer. In W. Gaul and M. Schader, editors, *Classification as a tool of research*, pages 269–274, North Holland, Amsterdam, 1986.
- [6] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.
- [7] S. Strahringer and R. Wille. Conceptual clustering via convex-ordian scales. *Information and classification*, pages 85–98, 1993.
- [8] G. Stumme and R. Wille. A geometrical heuristic for drawing concept lattice. *graph drawing*, pages 452–459, 1995.
- [9] F. Vogt, C. Wachter, and R. Wille. Data analysis based on a conceptual file. *Classification, data analysis and knowledge organisation*, pages 131–140, 1991.