Formal Concept Analysis: Methods and Applications in Computer Science

Bernhard Ganter
TU Dresden
ganter@math.tu-dresden.de

Summer 2002

Adapted and extended by:
Gerd Stumme
Otto–von–Guericke–Universität Magdeburg
stumme@aifb.uni-karlsruhe.de

Summer 2003

Contents

Chapter 1

Formal Concept Analysis in a nutshell

This section is meant to be an ‘appetizer’. It provides a brief overview over Formal Concept Analysis, in order to allow for a better understanding of the overall picture. This section introduces the most basic notions of Formal Concept Analysis, namely formal contexts, formal concepts, concept lattices, many-valued contexts and conceptual scaling. These definitions are repeated and discussed in more detail in the remainder of the book.

1.1 Formal contexts, concepts, and concept lattices

Formal Concept Analysis (FCA) was introduced as a mathematical theory modeling the concept of ‘concepts’ in terms of lattice theory. To allow a mathematical description of extensions and intensions, FCA starts with a *(formal) context*.

Definition 1 A *(formal) context* is a triple $\mathbb{K} := (G, M, I)$, where G is a set whose elements are called *objects*, M is a set whose elements are called *attributes*, and I is a binary relation between G and M (i.e. $I \subseteq G \times M$). $(g, m) \in I$ is read “object g *has* attribute m”. $\Diamond$

Figure 1.1 shows a formal context where the object set G comprises all airlines of the Star Alliance group and the attribute set M lists their destinations. The binary relation I is given by the cross table and describes which destinations are served by which Star Alliance member.

Definition 2 For $A \subseteq G$, let

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

and, for $B \subseteq M$, let

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

A *(formal) concept* of a formal context (G, M, I) is a pair (A, B) with $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The sets A and B are called the *extent* and the *intent* of the formal concept (A, B), respectively. The *subconcept–superconcept relation* is formalized by

$$(A_1, B_1) \leq (A_2, B_2) :\Longleftrightarrow A_1 \subseteq A_2 \quad (\Longleftrightarrow B_1 \supseteq B_2) \ .$$

	Latin America	Europe	Canada	Asia Pacific	Middle East	Africa	Mexico	Caribbean	United States
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
All Nippon Airways		×		×					×
Ansett Australia				×					
The Austrian Airlines Group		×	×	×	×	×			×
British Midland		×							
Lufthansa	×	×	×	×	×	×	×		×
Mexicana	×		×				×	×	×
Scandinavian Airlines	×	×		×		×			×
Singapore Airlines		×	×	×	×	×			×
Thai Airways International	×	×		×				×	×
United Airlines	×	×	×	×			×	×	×
VARIG	×	×		×		×	×		×

Figure 1.1: A formal context about the destinations of the Star Alliance members

The set of all formal concepts of a context $\mathbb{K}$ together with the order relation $\leq$ is always a complete lattice,[1] called the *concept lattice* of $\mathbb{K}$ and denoted by $\underline{\mathfrak{B}}(\mathbb{K})$. $\Diamond$

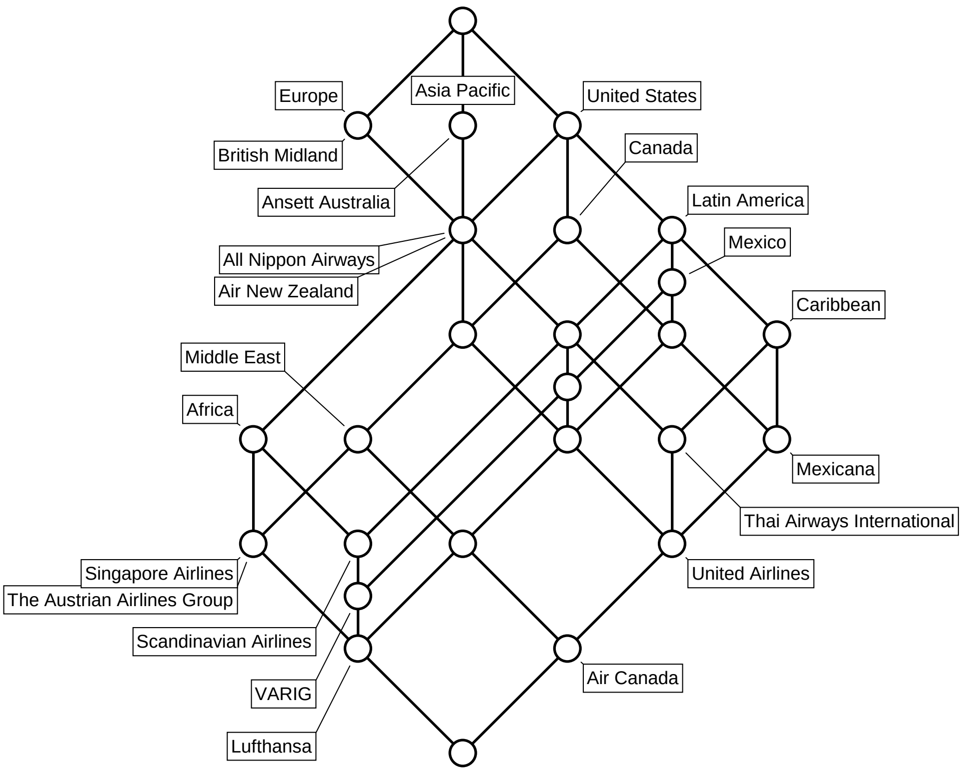


Figure 1.2: The concept lattice of the context in Figure 1.1

[1] I. e., for each subset of concepts, there is always a unique greatest common subconcept and a unique least common superconcept.

Figure 1.2 shows the concept lattice of the context in Figure 1.1 by a *line diagram.* In a line diagram, each node represents a formal concept. A concept $\mathfrak{c}_1$ is a subconcept of a concept $\mathfrak{c}_2$ if and only if there is a path of descending edges from the node representing $\mathfrak{c}_2$ to the node representing $\mathfrak{c}_1$. The name of an object g is always attached to the node representing the smallest concept with g in its extent; dually, the name of an attribute m is always attached to the node representing the largest concept with m in its intent. We can read the context relation from the diagram because an object g has an attribute m if and only if the concept labeled by g is a subconcept of the one labeled by m. The extent of a concept consists of all objects whose labels are attached to subconcepts, and, dually, the intent consists of all attributes attached to superconcepts. For example, the concept labeled by 'Middle East' has {Singapore Airlines, The Austrian Airlines Group, Lufthansa, Air Canada} as extent, and {Middle East, Canada, United States, Europe, Asia Pacific} as intent.

In the top of the diagram, we find the destinations which are served by most of the members: Europe, Asia Pacific, and the United States. For instance, beside British Midland and Ansett Australia, all airlines are serving the United States. Those two airlines are located at the top of the diagram, as they serve the fewest destinations — they operate only in Europe and Asia Pacific, respectively.

The further we go down in the concept lattice, the more globally operating are the airlines. The most destinations are served by the airlines at the bottom of the diagram: Lufthansa (serving all destinations beside the Caribbean) and Air Canada (serving all destinations beside Africa). Also, the further we go down in the lattice, the fewer served are the destinations. For instance, Africa, the Middle East, and the Caribbean are served by relatively few Star Alliance members.

Dependencies between the attributes can be described by implications. For $X, Y \subseteq M$, we say that the *implication* $X \rightarrow Y$ *holds* in the context, if each object having all attributes in X also has all attributes in Y. For instance, the implication {Europe, United States} $\rightarrow$ {Asia Pacific} holds in the Star Alliance context. It can be read directly in the line diagram: the largest concept having both 'Europe' and 'United States' in its intent (i. e., the concept labeled by 'All Nippon Airways' and 'Air New Zealand') also has 'Asia Pacific' in its intent. Similarly, one can detect that the destinations 'Africa' and 'Canada' together imply the destination 'Middle East' (and also 'Europe', 'Asia Pacific', and 'United States').

Concept lattices can also be visualized in *nested line diagrams.* For obtaining a nested line diagram, one splits the set of attributes in two parts, and obtains thus two formal contexts. For each formal context, one computes its concept lattice and a line diagram. The nested line diagram is obtained by enlarging the nodes of the first line diagram and by drawing the second diagram inside. The second lattice is used to further differentiate each of the extents of the concepts of the first lattice. Figure 1.3 shows a nested line diagram for the Star Alliance context. It is obtained by splitting the attribute set as follows: $M =$ { Europe, Asia Pacific, Africa, Middle East } $\cup$ { United States, Canada, Latin America, Mexico, Caribbean }. The order relation can be read by replacing each of the lines of the large diagram by eight parallel lines linking corresponding nodes in the inner diagrams. The concept lattice given in Figure 1.2 is embedded (as a join–semilattice) in this diagram, it consists of the bold nodes. The concept mentioned above (labeled by 'Middle East') is for instance represented by the left-most bold node in the lower right part.

The bold concepts are referred to as 'realized concepts', as, for each of them, the set of all attributes labeled above is an intent of the formal context. The non-realized concepts are not only displayed to indicate the structure of the inner scale, but also because they indicate implications: Each non-realized concept indicates that the attributes in its intent imply the attributes contained in the largest realized concept below. For instance, the first implication discussed above is indicated

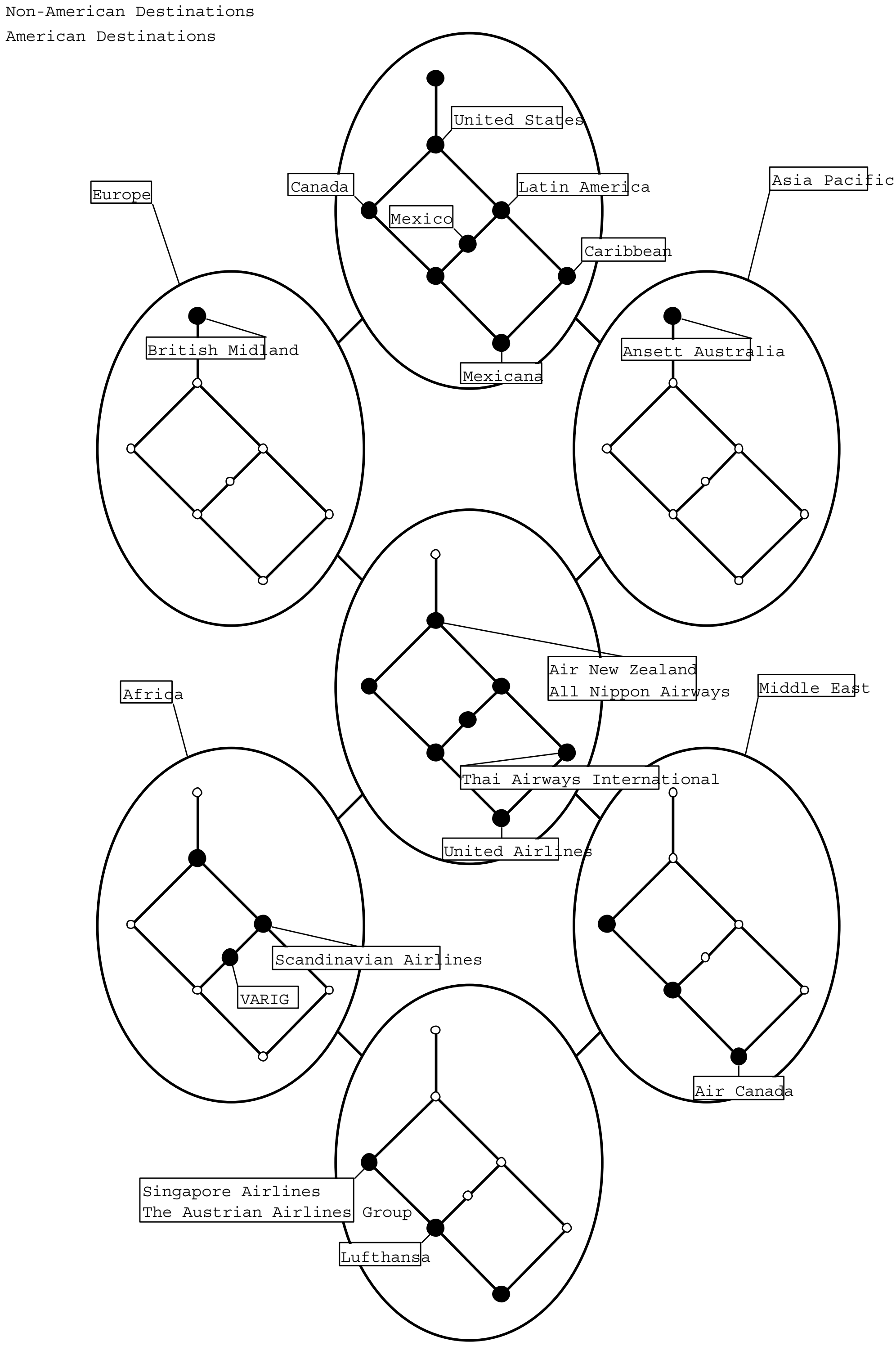


Figure 1.3: A nested diagram of the concept lattice in Figure 1.2

by the non-realized concept having as intent 'Europe' and 'United States', it is represented by the empty node below the concept labeled by 'British Midland'. The largest realized sub-concept below is the one labeled by ‘All Nippon Airways’ and ‘Air New Zealand’ — which additionally has ‘Asia Pacific’ in its intent. Hence the implication { Europe, United States } $\rightarrow$ { Asia Pacific } holds. The second implication from above is indicated by the non-realized concept left of the concept labeled by ‘Scandinavian Airlines’, and the largest realized concept below, which is the one labeled by ‘Singapore Airlines’ and ‘The Austrian Airlines Group’.

1.2 Many-valued contexts and conceptual scaling

In most applications, there are not only Boolean attributes in the databases. The conceptual model we use for this is a *many-valued context*. In database terms, a many-valued context is a relation of a relational database with one key attribute whose domain is the set G of objects. Here, we consider only one (denormalized) database relation at a time.

In order to obtain a concept lattice from a many-valued context, it has to be translated into a one-valued (formal) context. The translation process is described by *conceptual scales*:

Definition 3 A *many-valued context* is a tuple $(G, M, (W_m)_{m\in M}, I)$ where G and M are sets of *objects* and *attributes*, resp., W_m is a set of *values* for each $m \in M$, and $I \subseteq G \times \bigcup_{m\in M}(\{m\} \times W_m)$ is a relation such that $(g, m, w_1) \in I$ and $(g, m, w_2) \in I$ imply $w_1 = w_2$.

A *conceptual scale* for an attribute $m \in M$ is a one-valued context $\mathbb{S}_m := (G_m, M_m, I_m)$ with $W_m \subseteq G_m$. The context $\mathbb{R}_m := (G, M_m, J_m)$ with $gJ_mn :\iff \exists w{\in}W_m\colon (g, m, w){\in}I \wedge (w, n){\in}I_m$ is called the *realized scale* for the attribute $m \in M$.

A *Conceptual Information System* consists of a many-valued context together with a set of conceptual scales. $\Diamond$

The set M consists of all attributes of the database scheme, while the sets M_m contain the attributes which are shown to the user by the Conceptual Information System.

Example 1 The information system INFO-80 supports planning, realization, and control of business transactions related to flight movements at Frankfurt Airport. A Conceptual Information System has been established, based on the TOSCANA software, in order to facilitate access to the data of INFO-80 ([**?**]).

Here we consider 18389 outbound flight movements effectuated at Frankfurt Airport during one month. For each of these flight movements 110 attributes are registered automatically and stored in a database (e. g., destination, estimated time of departure, actual time of departure, number of passengers, terminal position, ...). 77 conceptual scales have been designed (some of the 110 database attributes were of minor importance for data analysis). By combining the scales and zooming into them with other scales, the analyst can explore dependencies and irregularities.

Figure 1.4 shows the realized scale *Destination Southern Europe/Mediterranean Sea*. Because of the large number of flight movements, TOSCANA displays only the number of flight movements assigned to each concept. If desired, the user can drill-down to the flight number and to more detailed information. For instance, the concept labeled by ‘Marokko’ (Morocco) has 53 objects (i. e., flight numbers) in its extent, and the attributes ‘Marokko’ and ‘Nordafrika’ (Northern Africa) in its intent. Its upper neighbor (which is labeled by ‘Nordafrika’) has $53+14+111 = 178$ objects in its extent, and ‘Nordafrika’ in its intent.

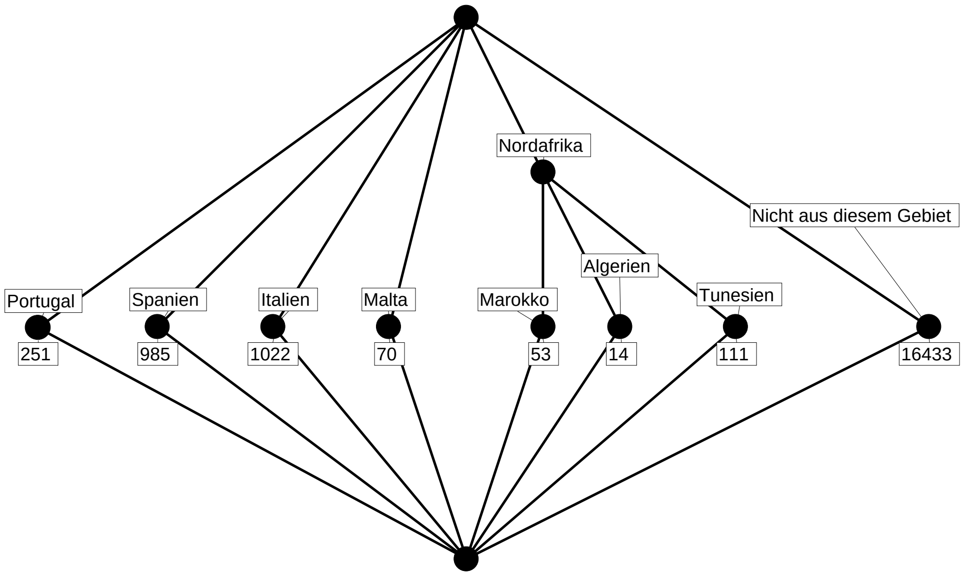


Figure 1.4: Realized scale *Destination Southern Europe/Mediterranean Sea*

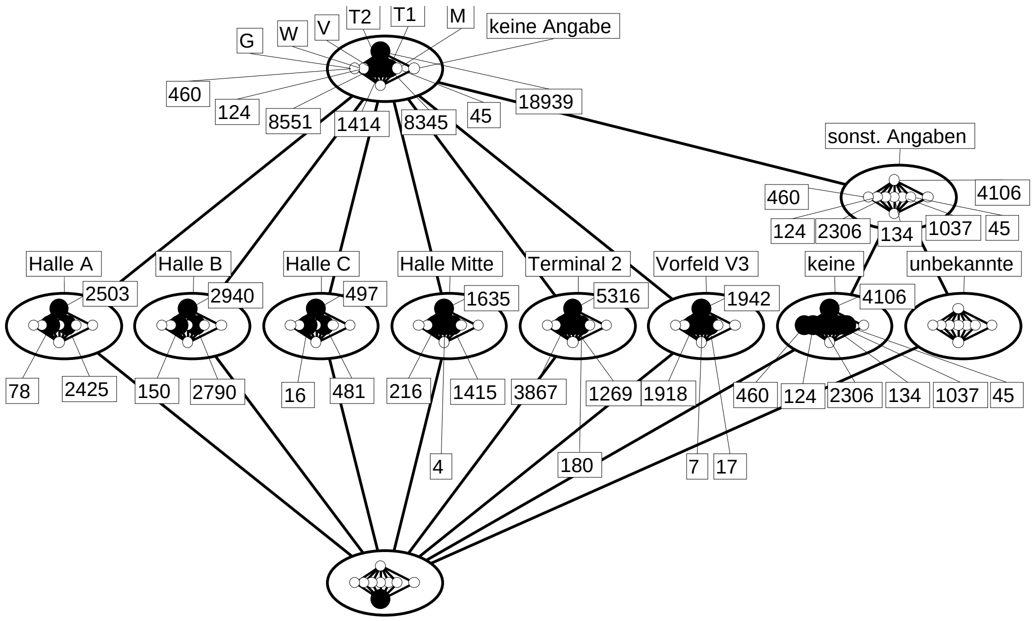


Figure 1.5: Nested line diagram of the scales *Position of Baggage Conveyor* and *Position of Aircraft*

For analyzing how objects are distributed over two different scales, one can combine the two conceptual scales. The result of combining the two conceptual scales *Position of Baggage Conveyor* and *Position of Aircraft* is displayed in the nested line diagram in Figure 1.5.

Each of the 17 lines of the outer scale represents seven parallel lines linking corresponding concepts of the inner scale. The concept lattice we are interested in is embedded in this *direct product*. The embedding is indicated by the bold circles.

Again, each non-realized concept (i. e., the empty circles) indicates an implication. For example, the leftmost circle in the leftmost ellipse has the attribute 'Halle A' (hall A) and 'G' (general aviation, i. e., private airplanes) in its intent. These two attributes form the premise of the implication. The conclusion of the implication is given by the intent of the largest realized concept below. In this case, it is the bottom concept which has all attributes in its extent. Hence hall A as position of the baggage conveyor and the general aviation apron as position of the aircraft implies everything else. This means that the premise is not realized by any object. There are no general aviation aircraft with a baggage conveyor assigned at hall A. There are concepts in the nested line diagram which are expected to be non-realized as well. For instance, an aircraft at Terminal 1 should not have a baggage conveyor assigned at Terminal 2. But in the diagram, we see that the concept with these two attributes is realized. There are 180 flight movements having the attribute 'Terminal 2' of the outer scale and the attribute 'T1' of the inner scale.

We can detect some more unnormal combinations. There are four aircraft that docked at Terminal 2, while their assigned baggage conveyors are at Terminal 1. To seven aircraft at Terminal 2 and 17 aircraft at Terminal 1, conveyors on the 'Vorfeld V3' (Apron V3) were assigned. In all these cases, one can drill down to the original data by clicking on the number to obtain the flight movement numbers, which in turn lead to the data sets stored in the INFO-80 system.

The knowledge that the combinations are unnormal, is not coded explicitly in INFO-80, but is only implicitly present as expert knowledge. Since these combinations happen at the airport, they cannot be forbidden by database constraints. But once such conflicts are discovered, one can implement an alarm which informs the operator of the baggage transportation system about such cases.

If there are too many scales involved in the knowledge discovery process, then zooming into the outer scale reduces the size of the displayed diagram. For instance, by zooming into the concept labeled by 'Terminal 2' in Fig. 1.5, we obtain the diagram of the conceptual scale *Position of Aircraft*, but with the objects being restricted to those flight movements where the assigned baggage conveyor is at Terminal 2. Then one can continue the exploration by adding a new conceptual scale on the inner level. This interactive human-centered knowledge discovery process is described in detail in [**?**].

This section gave a short introduction to the core notions of FCA. We will discuss them (and more advanced topics) in more detail in the remainder of this book.

Part I

Contexts, Concepts, and Concept Lattices

Chapter 2

Concept lattices

Formal Concept Analysis studies how *objects* can be hierarchically grouped together according to their common *attributes*. One of the aspects of FCA thus is *attribute logic*, the study of possible attribute combinations. Most of the time, this will be very elementary. Those with a background in Mathematical Logic might say that attribute logic is just Propositional Calculus, and thus Boolean Logic, or even a fragment of this. Historically, the name *Propositional Logic* is misleading: Boole himself used the intuition of attributes ("signs") rather than of propositions. So in fact, attribute logic goes back to Boole.

But our style is different from that of logicians. Our logic is *contextual*, which means that we are interested in the logical structure of concrete data (of the *context*). Of course, the general rules of mathematical logic are important for this and will be utilized.

2.1 Basic notions

2.1.1 Formal contexts and cross tables

Definition 4 A **Formal Context** (G, M, I) consists of two sets G and M and of a binary relation $I \subseteq G \times M$. The elements of G are called the **objects**, those of M the **attributes** of (G, M, I). If $g \in G$ and $m \in M$ are in relation I, we write $(g, m) \in I$ or $g\ I\ M$ and read this as *"object g* **has** *attribute M"*. $\Diamond$

The simplest format for writing down a formal context is a **cross table**: we write a rectangular table with one row for each object and one column for each attribute, having a cross in the intersection of row g with column m iff $(g, m) \in I$. The simplest data type for computer storage is that of a bit matrix[1].

Note that the definition of a formal context is very general. There are no restrictions about the nature of objects and attributes. We may consider physical objects, or persons, numbers, processes, structures, etc., etc., virtually everything. Anything that is a *set* in the mathematical sense may be taken as the set of objects or of attributes of some formal context. We may interchange the rôle of objects and attributes: if (G, M, I) is a formal context, then so is the **dual context** (M, G, I^{-1}) (with $(m, g) \in I^{-1} :\iff (g, m) \in I$). It is also not necessary that G and M are disjoint, they need not even be different.

On the other hand, the definition is rather restrictive when applied to real world phenomena. Language phrases like "all human beings" or "all chairs" do not denote

[1]It is not easy to say which is the *most efficient* data type for formal contexts. This depends, of course, on the operations we want to perform with formal contexts. The most important ones are the derivation operators, to be defined in the next subsection.

sets in our sense. There is no "set of all chairs", because the decision if something is a chair is not a matter of fact but a matter of subjective interpretation. The notion of "formal concept" which we shall base on the definition of "formal context" is much, much narrower than what is commonly understood as a concept of human cognition. The step from "context" to "formal context" is quite an incisive one. It is the step from "real world" to "data". Later on, when we get tired of saying "formal concepts of a formal context", we will sometimes omit the word "formal". But we should keep in mind that it makes a big difference.

2.1.2 The derivation operators

Given a selection $A \subseteq G$ of objects from a formal context (G, M, I), we may ask which attributes from M are common to all these objects. This defines an operator that produces for every set $A \subseteq G$ of objects the set $A^{\uparrow}$ of their common attributes.

Definition 5 For $A \subseteq G$, we let

$$A^{\uparrow} := \{m \in M \mid g\ I\ m \text{ for all } g \in A\} \ .$$

Dually, we introduce for a set $B \subseteq M$ of attributes

$$B^{\downarrow} := \{g \in G \mid g\ I\ m \text{ for all } m \in B\} \ .$$

These two operators are the **derivation operators** for (G, M, I). ◊

Then set $B^{\downarrow}$ denotes thus the set consisting of those objects in G that have all the attributes from B.

Usually, we do not distinguish the derivation operators in writing and use the notation A', B' instead. This is convenient, as long as the distinction is not explicitly needed.

If A is a set of objects, then A' is a set of attributes, to which we can apply the second derivation operator to obtain A'' (more precisely: $(A^{\uparrow})^{\downarrow}$), a set of objects. Dually, starting with a set B of attributes, we may form the set B'', which is again a set of attributes. We have the following simple facts:

Proposition 1 *For subsets $A, A_1, A_2 \subseteq G$ we have*

1. $A_1 \subseteq A_2 \Rightarrow A_2' \subseteq A_1'$,

2. $A \subseteq A''$,

3. $A = A'''$.

Dually, for subsets $B, B_1, B_2 \subseteq M$ we have

1'. $B_1 \subseteq B_2 \Rightarrow B_2' \subseteq B_1'$,

2'. $B \subseteq B''$,

3'. $B = B'''$.

The elementary **proof** is omitted here. The reader may confer to [**?**] for details. The mathematically interested reader may notice that the derivation operators constitute a **Galois connection** between the (power sets of the) sets G and M.

The not so mathematically oriented reader should try to express the statements of the Proposition in comman language. We give an example: Statement 1.) says that *if a selection of objects is enlarged, then the attributes which are common to all objects of the larger selection are among the common attributes of the smaller selection.* Try to formulate 2.) and 3.) in a similar manner!

2.1.3 Formal concepts, extent and intent

In what follows, (G, M, I) always denotes a formal context.

Definition 6 (A, B) is a **formal concept** of (G, M, I) iff

$$A \subseteq G, \quad B \subseteq M, \quad A' = B, \quad \text{and } A = B'.$$

The set A is called the **extent** of the formal concept (A, B), and the set B is called its **intent**. $\Diamond$

According to this definition, a formal concept has two parts: its extent and its intent. This follows an old tradition in philosophical concept logic, as expressed in the *Logic of Port Royal, 1654* [**?**], and in the International Standard ISO 704 (translation of the German Standard DIN 2330).

The description of a concept by extent and intent is redundant, because each of the two parts determines the other (since $B = A'$ and $A = B'$). But for many reasons this redundant description is very convenient.

When a formal context is written as a cross table, then every formal concept (A, B) corresponds to a (filled) rectangular subtable, with row set A and column set B. To make this more precise, note that in the definition of a formal context there is no *order* on the sets G or M. Permuting the rows or the columns of a cross table therefore does not change the formal context it represents. A *rectangular subtable* may, in this sense, omit some rows or columns; it must be rectangular after an appropriate rearrangement of the rows and the columns. It is then easy to characterize the rectangular subtables that correspond to formal contexts: they are full of crosses and maximal with respect to this property:

Lemma 1 *(A, B) is a formal concept of (G,M,I)iff $A \subseteq G$, $B \subseteq M$, and A and B are each maximal (with respect to set inclusion) with the property $A \times B \subseteq I$.*

A formal context may have many formal concepts. In fact, it is not difficult to come up with examples where the number of formal concepts is exponential in the size of the formal context. The set of all formal concepts of (G, M, I) is denoted

$$\mathfrak{B}(G, M, I).$$

Later on we shall discuss an algorithm to compute all formal concepts of a given formal context.

2.1.4 Conceptual hierarchy

Formal concepts can be (partially) ordered in a natural way. Again, the definition is inspired by the way we usually order concepts in a *subconcept–superconcept hierarchy*: "Pig" is a subconcept of "mammal", because every pig is a mammal. Transferring this to formal concepts, the natural definition is as follows:

Definition 7 Let (A_1, B_1) and (A_2, B_2) be formal concepts of (G, M, I). We say that (A_1, B_1) is a **subconcept** of (A_2, B_2) (and, equivalently, that (A_2, B_2) is a **superconcept** of (A_1, B_1) iff $A_1 \subseteq A_2$. We use the $\leq$-sign to express this relation and thus have

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

The set of all formal concepts of (G, M, I), ordered by this relation, is denoted

$$\underline{\mathfrak{B}}(G, M, I)$$

and is called the **concept lattice** of the formal context (G, M, I). $\Diamond$

This definition is natural, but irritatingly asymmetric. What about the intents? Well, a look at Proposition 1 shows that for concepts (A_1, B_1) and (A_2, B_2)

$$A_1 \subseteq A_2 \quad \text{is equivalent to} \quad B_2 \subseteq B_1.$$

Therefore

$$(A_1, B_1) \leq (A_2, B_2) :\Longleftrightarrow A_1 \subseteq A_2 \quad (\Longleftrightarrow B_2 \subseteq B_1).$$

The concept lattice of a formal context is a partially ordered set:

Definition 8 A partially ordered set is a pair $(P, \leq)$ where P is a set, and $\leq$ is a binary relation on P (i. e.,$\leq$ is a subset of $P \times P$) which is

1. reflexive ($x \leq x$ for all $x \in P$),
2. anti-symmetric ($x \leq y \wedge x \neq y \Rightarrow \neg y \leq x$ for all $x \in P$),
3. and transitive ($x \leq y \wedge y \leq z \Rightarrow x \leq z$ for all $x, y, z \in P$),

$x \geq y$ stands for $y \leq$, and $x < y$ for $x \leq y$ with $x \neq y$. $\Diamond$

Beside concept lattices, partially ordered sets appear frequently in Mathematics and Computer Science. We provide some examples.

Example 2
- Every tree is a partially ordered set.
- The set $\mathbb{R}$ of all reel numbers together with the ordinary $\leq$–relation.
- On any set M, the equality $=$ is also a (trivial) order relation.
- For a set M, the powerset $\mathfrak{P}(M)$ with set inclusion $\subseteq$ is also a partially ordered set.
- The set $\mathbb{N}$ of all natural numbers with the divides–relation $|$.

Observe that we do not assume a **total order**, which would require the additional condition $x \leq y$ or $y \leq x$ for $x, y \in P$. Observe also that partially ordered sets allow for 'multiple inheritance', as the two last examples show.

Concept lattices have additional properties beside being partially ordered sets, that it why we call them 'lattices'. This will be the topic of the next section.

2.2 The algebra of concepts

2.2.1 Reading a concept lattice diagram

The concept lattice of (G, M, I) is the set of all formal concepts of (G, M, I), ordered by the subconcept–superconcept order. Ordered sets of moderate size can conveniently be displayed as **order diagram**s. We explain how to *read* a concept lattice diagram by means of an example given in Figure 2.1. Later on we will discuss how to *draw* such diagrams.

Figure 2.1 refers to the following situation: Think of two squares, of equal size, that are drawn on paper. There are different ways to arrange two squares: they may be disjoint (= have no point in common), may overlap (= have a common interior point), may share a vertex, an edge or a line segment of the boundary (of length > 0), they may be parallel or not.

Figure 2.1 shows a concept lattice unfolding these possibilities. It consists of twelve formal concepts, these being represented by the twelve small circles in the

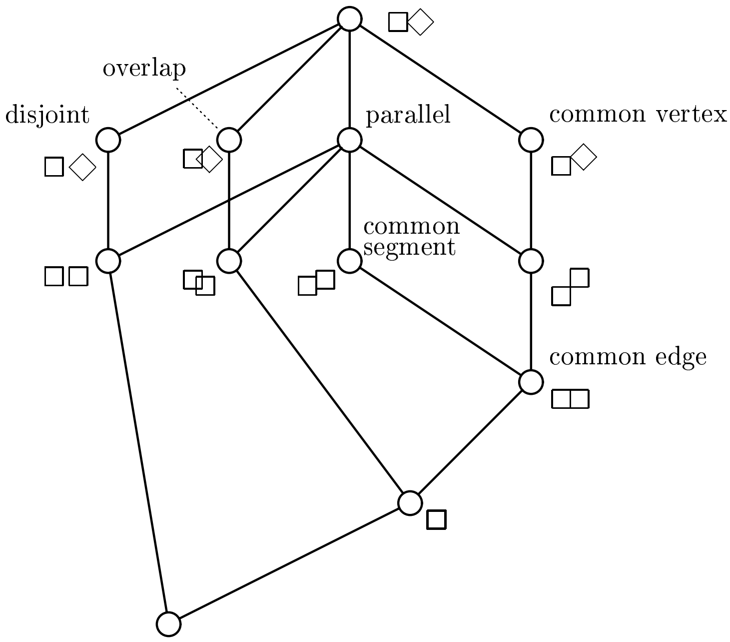


Figure 2.1: A concept lattice diagram. The objects are pairs of unit squares. The attributes describe their mutual position. See also Figure 6.6 (p. 107).

diagram. The names of the six attributes are given. Each name is attached to one of the formal concepts and is written slightly above the respective circle. The ten objects are represented by little pictures; each showing a pair of unit squares. Again, each object is attached to exactly one formal concept; the picture representing the object is drawn slightly below the circle representing the object concept.

Some of the circles are connected by edges. These express the concept order. With the help of the edges we can read from the diagram which concepts are sub-concepts of which other concepts, and which objects have which attributes. To do so, one has to follow *ascending paths* in the diagram.

For example, consider the object . From the corresponding circle we can reach, via ascending paths, four attributes: “common edge”, “common segment”, “common vertex”, and “parallel”. does in fact have these properties, and does not have the other ones: the two squares are neither “disjoint” nor do they “overlap”.

Similarly, we can find those objects that have a given attribute by following all descending paths starting at the attribute concept. For example, to find all objects which “overlap”, we start at the attribute concept labeled “overlap” and follow the edges downward. We can reach three objects (namely , , and , the latter symbolizing two squares at the same position). Note that we cannot reach , because only at concept nodes it is allows to make a turn.

With the same method, we can read the intent and the extent of every formal concept in the diagram. For example, consider the concept circle labeled . Its extent consists of all objects that can be reached from that circle on an descending path. The extent therefore is { , }. Similarly, we find by an inspection of the ascending paths that the intent of this formal concept is {overlap, parallel}.

The diagram contains all necessary information. We can read off the objects, the attributes, and the incidence relation I. Thus we can perfectly reconstruct the formal context from the diagram ("the original data").[2] Moreover, for each formal concept we can easily determine its extent and intent from the diagram.

So in a certain sense, concept lattice diagrams are perfect. But there are, of course, limitations. Take another look at Figure 2.1. Is it *correct*? Is it *complete*? The answer is that, since a concept lattice faithfully unfolds the formal context, the information displayed in the lattice diagram can be only as correct and complete as the formal context is. In our specific example it is easy to check that the given examples in fact do have the properties as indicated. But a more difficult problem is if our selection of objects is representative. Are there possibilities to combine two squares, that lead to an attribute combination not occurring in our sample?

We shall come back to that question later.

2.2.2 Supremum and infimum

Can we compute with formal concepts? Yes, we can. The concept operations are however quite different from addition and multiplication of numbers. They resemble more of the operations *greatest common divisor* and *least common multiple*, that we know from integers.

Definition 9 Let $(M, \leq)$ be a partially ordered set, and A be a subset of M. A **lower bound** of A is an element s of M with $s \leq a$, for all $a \in A$. An **upper bound** of A is defined dually. If there exists a largest element in the set of all lower bounds of A, then it is called the **infimum** (or **join**) of A. It is denoted *inf* A or $\bigwedge A$. The **supremum** (or **meet**) of A (*sup* A, $\bigvee A$) is defined dually. For $A = \{x, y\}$, we write also $x \wedge y$ for their infimun, and $x \vee y$ for their supremum. $\Diamond$

Similarly as for arbitrary sums we use the symbol $\sum$ instead of $+$, we use the large symbols $\bigvee$ and $\bigwedge$ for arbitrary suprema and infima.

Lemma 2 *For any two formal concepts (A_1, B_1) and (A_2, B_2) of some formal context we obtain*

- *the infimum (***greatest common subconcept***) of (A_1, B_1) and (A_2, B_2) as*

$$(A_1, B_1) \wedge (A_2, B_2) := (A_1 \cap A_2, (B_1 \cup B_2)''),$$

- *the supremum (***least common superconcept***) of (A_1, B_1) and (A_2, B_2) as*

$$(A_1, B_1) \vee (A_2, B_2) := ((A_1 \cup A_2)'', B_1 \cap B_2).$$

It is not difficult to prove (Exercise 3) that what is suggested by this definition is in fact true: $(A_1, B_1) \wedge (A_2, B_2)$ is in fact a formal concept (of the same context), $(A_1, B_1) \wedge (A_2, B_2)$ is a subconcept of both (A_1, B_1) and (A_2, B_2), and any other common subconcept of (A_1, B_1) and (A_2, B_2) is also a subconcept of $(A_1, B_1) \wedge (A_2, B_2)$. Similarly, $(A_1, B_1) \vee (A_2, B_2)$ is a formal concept, it is a superconcept of (A_1, B_1) and of (A_2, B_2), and it is a subconcept of any common superconcept of these two formal concepts.

With some practice, one can read off infima and suprema from the lattice diagram. Choose any two concepts from Figure 2.1 and follow the descending paths from the corresponding nodes in the diagram. There is always a highest point where these paths meet, that is, a highest concept that is below both, namely, the infimum. Any other concept below both can be reached from the highest one on a

[2] This reconstruction is assured by the Basic Theorem given below.

descending path. Similarly, for any two formal concepts there is always a lowest node (the supremum of the two), that can be reached from both concepts via ascending paths. And any common superconcept of the two is on an ascending path from their supremum.

2.2.3 Complete lattices

The operations for computing with formal concepts, infimum and supremum, are not as weird as one might suspect. In fact, we obtain with each concept lattice an algebraic structure called a "lattice", and such structures occur frequently in mathematics and computer science. "Lattice theory" is an active field of research in mathematics. A *lattice* is an algebraic structure with two operations (called "meet" and "join" or "infimum" and "supremum") that satisfy certain natural conditions:[3]

Definition 10 A partially ordered set $\mathbb{V} := (V, \leq)$ is a **lattice**, if their exists, for every pair of elements $x, y \in V$, their infimum $x \wedge y$ and their supremum $x \vee y$. $\Diamond$

We shall not discuss the algebraic theory of lattices in this lecture. Many universities offer courses in lattice theory, and there are excellent textbooks.[4]

Concept lattices have an additional nice property: they are **complete lattice**s. This means that the operations of infimum and supremum do not only work for an input consisting of two elements, but for arbitrary many. In other words: each collection of formal concepts has a greatest common subconcept and a least common superconcept. This is even true for infinite sets of concepts. The operations "infimum" and "supremum" are not necessarily binary, they work for any input size.

Definition 11 A partially ordered set $\mathbb{V} := (V, \leq)$ is a **complete lattice**, if for every set $A \subseteq V$ exist its infimum $\bigwedge V$ and its supremum $\bigvee A$. $\Diamond$

The definition requests the existence of infimum and supremum for every set A, hence also for the empty set $A := \emptyset$. Following the definition, we obtain that $\bigwedge \emptyset$ has to be the (unique) largest element of the lattice. It is denoted by $\mathbf{1_V}$. Dually, $\bigvee \emptyset$ has to be the smallest element of the lattice; it is denoted by $\mathbf{0_V}$.

The arbitrary arity of infimum and supremum is very useful, but will make essentially no difference for our considerations, because we shall mainly be concerned with finite formal contexts and finite concept lattices. Well, this is not completely true. In fact, although the concept lattice in Figure 2.1 is finite, its ten objects are representatives for *all* possibilities to combine two unit squares. Of course, there are infinitely many such possibilities. It is true that we shall consider finite concept lattices, but our examples may be taken from an infinite reservoir.

2.2.4 The Basic Theorem

We give now a mathematically precise formulation of the algebraic properties of concept lattices. The theorem below is not difficult, but basic for many other results. Its formulation contains some technical terms that we have not mentioned so far.

To understand the notion of a "supremum-dense" set, think of the positive integers with the least common multiple (LCM) in the rôle of the supremum operation. Some numbers can be written as a LCM of other integers. For example,

[3] Unfortunately, the word "lattice" is used with different meanings in mathematics. It also refers to generalized grids.

[4] *An introduction to lattices and order* by B. Davey and H. Priestley is particularly popular among CS students.

12 = LCM$\{3,4\}$. Therefore, 12 is called *LCM-reducible*. Prime numbers cannot be written as an LCM of other numbers. Therefore, prime numbers are LCM-irreducible. But prime numbers are not the only LCM-irreducible numbers. For example, 25 is not the least common multiple of other numbers, and thus is LCM-irreducible. It is easy to see that the LCM-irreducible integers are precisely the *prime powers*[5]. The set of prime powers is *LCM-dense*, which means that every positive integer can be written as a LCM of prime powers. Any larger set, that is, any set containing all prime powers, is also LCM-dense, of course.

There are no GCD-irreducible numbers. Every positive integer can be written as the GCD of other numbers. But there are GCD-dense sets of numbers. For example, the set of all integers greater than 1000 is GCD-dense.

In a complete lattice, an element is called **supremum-irreducible** if it cannot be written as an supremum of other elements, and **infimum-irreducible** if it can not be expressed as an infimum of other elements. It is very easy to locate the irreducible elements in a diagram of a finite lattice: the supremum-irreducible elements are precisely those from which there is exactly one edge going downward. An element is infimum-irreducible if and only if it is the start of exactly one upward edge. In Figure 2.1 there are precisely nine supremum-irreducible concepts and precisely five infimum-irreducible concepts. Exactly four concepts have both properties, they are **doubly irreducible**.

A set of elements of a complete lattice is called **supremum-dense**, if every lattice element is a supremum of elements from this set. Dually, a set is called **infimum-dense**, if the infima that can be computed from this set exhaust all lattice elements.

Definition 12 Two complete lattices $\mathbb{V}$ and $\mathbb{W}$ are **isomorphic** ($\mathbb{V} \cong \mathbb{W}$), if there exists a bijective mapping $\varphi: V \to W$ with $x \leq y \iff \varphi(x) \leq \varphi(y)$. The mapping φ is then called **lattice isomorphism** between $\mathbb{V}$ and $\mathbb{W}$. $\Diamond$

Example 3 The lattice $(\{1,2,3,5,6,10,12,15,30\}, |)$ is isomorphic to the powerset $(\mathfrak{P}(\{2,3,5\}), \subseteq)$ with the isomorphism $\varphi: \{1,2,3,5,6,10,12,15,30\} \to \mathfrak{P}(\{2,3,5\})$ with $\varphi(x) := \{y \in \{2,3,5\} \mid y \text{ divides } x\}$.

Now we have defined all the terminology necessary for stating the main theorem of Formal Concept Analysis.

Theorem 1 (The basic theorem of Formal Concept Analysis.) *The concept lattice of any formal context (G, M, I) is a complete lattice. For an arbitrary set $\{(A_i, B_i) \mid i \in I\} \subseteq \mathfrak{B}(G, M, I)$ of formal concepts, the supremum is given by*

$$\bigvee_{i \in I} (A_i, B_i) = \left((\bigcup_{i \in I} A_i)'', \bigcap_{i \in I} B_i \right)$$

and the infimum is given by

$$\bigwedge_{i \in I} (A_i, B_i) = \left(\bigcap_{i \in I} A_i, (\bigcup_{i \in I} B_i)'' \right).$$

A complete lattice L is isomorphic to $\underline{\mathfrak{B}}(G, M, I)$ iff there are mappings $\tilde{\gamma} : G \to L$ and $\tilde{\mu} : M \to L$ such that $\tilde{\gamma}(G)$ is supremum-dense and $\tilde{\mu}(M)$ is infimum-dense in L, and

$$g \; I \; m \iff \tilde{\gamma}(g) \leq \tilde{\mu}(m).$$

In particular, $L \cong \underline{\mathfrak{B}}(L, L, \leq)$.

[5]The number 1 requires special treatment. We shall not go into this.

The theorem is less complicated as it first may seem. We give some explanations below. Readers in a hurry may skip these and continue with the next section.

The first part of the theorem gives the precise formulation for infimum and supremum of arbitrary sets of formal concepts. The second part of the theorem gives (among other information) an answer to the question if concept lattices have any special properties. The answer is "no": every complete lattice is (isomorphic to) a concept lattice. This means that for every complete lattice we must be able to find a set G of objects, a set M of attributes and a suitable relation I, such that the given lattice is isomorphic to $\underline{\mathfrak{B}}(G, M, I)$. The theorem does not only say how this can be done, it describes in fact *all* possibilities to achieve this.

In Figure 2.1, every object is attached to a unique concept, the corresponding *object concept*. Similarly for each attribute there corresponds an *attribute concept*. These can be defined as follows:

Definition 13 Let (G, M, I) be some formal context. Then

- for each object $g \in G$ the corresponding **object concept** is

$$\gamma g := (\{g\}'', \{g\}'),$$

- and for each attribute $m \in M$ the **attribute concept** is given by

$$\mu m := (\{m\}', \{m\}'').$$

The set of all object concepts of (G, M, I) is denoted γG, the set of all attribute concepts is μM. $\Diamond$

Using Definition 6 and Proposition 1 it is easy to check that these expressions in fact define formal concepts of (G, M, I).

We have that $\gamma g \leq (A, B) \iff g \in A$. A look at the first part of the Basic Theorem shows that each formal concept is the supremum of all the object concepts below it (Exercise 4). Therefore, the set γG of all object concepts is supremum-dense. Dually, the attribute concepts form am infimum dense set in $\underline{\mathfrak{B}}(G, M, I)$. The Basic Theorem says that, conversely, any supremum-dense set in a complete lattice L can be taken as the set of objects and any infimum-dense set be taken as a set of attributes for a formal context with concept lattice isomorphic to L.

We conclude with a simple observation that often helps to find errors in concept lattice diagrams. The fact that the object concepts form an supremum-dense set implies that every supremum-irreducible concept must be an object concept (the converse is not true). Dually, every infimum-irreducible concept must be an attribute concept. This yields the following rule for concept lattice diagrams:

Proposition 2 *Given a formal context (G, M, I) and a finite order diagram, labeled by the objects from G and the attributes from M. For $g \in G$ let $\tilde{\gamma}(g)$ denote the element of the diagram that is labeled with g, and let $\tilde{\mu}(m)$ denote the element labeled with m. Then the given diagram is a correctly labeled diagram of $\underline{\mathfrak{B}}(G, M, I)$ if and only if it fulfills the following conditions:*

1. *The diagram is a correct lattice diagram,*

2. *every supremum-irreducible element is labeled by some object,*

3. *every infimum-irreducible element is labeled by some attribute,*

4. $g \, I \, m \iff \tilde{\gamma}(g) \leq \tilde{\mu}(m)$.

2.2.5 The duality principle

The definitions of lattices and complete lattices are self-dual: If $(V, \leq)$ is a (complete) lattice, then $(V, \leq)^d := (V, \geq)$ is also a (complete) lattice. If a theorem holds for a (complete) lattice, then the 'dual theorem' also holds, i. e., the theorem where all occurences of $\leq, \vee, \wedge, \bigvee, \bigwedge, \mathbf{0_V}, \mathbf{1_V}$ etc. are replaced by $\geq, \wedge, \vee, \bigwedge, \bigvee, \mathbf{1_V}, \mathbf{0_V}$, resp.

For concept lattices, their dual can be obtained by "permuting" the formal context:

Lemma 3 *Let (G, M, I) be a formal context. Then $(\underline{\mathfrak{B}}((G, M, I)))^d \cong \underline{\mathfrak{B}}(M, G, I^{-1})$, where $I^{-1} := \{(m, g) \in M \times G \mid (g.m) \in I\}$.*

2.3 Computing and drawing concept lattices

2.3.1 Computing all concepts

There are several algorithms that help drawing concept lattices. We shall discuss some of them below. But we find it instructive to start by some small examples that can be drawn by hand. For computing concept lattices, we will investigate a fast algorithm later. We start with a naive method, and proceed then to a method which is suitable for manual computation.

In principle, it is not difficult to find all the concepts of a formal context. The following proposition summarizes the naive possibilities of generating all concepts:

Lemma 4 *Each concept of a context (G, M, I) has the form (X'', X') for some subset $X \subseteq G$ and the form (Y', Y'') for some subset $Y \subseteq M$. Conversely, all such pairs are concepts.*

Every extent is the intersection of attribute extents and every intent is the intersection of object intents.

The first part of the lemma suggests the following, first algorithm for computing all concepts:

1. $\underline{\mathfrak{B}}(G, M, I) := \emptyset$;
2. For all subsets X of G, add (X'', X') to $\underline{\mathfrak{B}}(G, M, I)$;

However, this is rather inefficient, and not practicable even for relatively small contexts. The second part of the proposition at least yields the possibility to calculate the concepts of a small context by hand.

The following method is more efficient, and is recommended for computations by hand. It is based on the following observations:

1. It suffices to determine all concept extents [or all concept intents] of (G, M, I),

   since we can always determine the other part of a formal concept with the help of the derivation operators.

2. The intersection of arbitrary many extents is an extent [and the intersection of arbitrary intents is an intent].

   This follows easily from the formulae given in the Basic Theorem. By the way: a convention that may seem absurd on the first glance allows to include in "arbitrary many" also the case "zero". The convention says that the intersection of zero intents equals M and the intersection of zero extents equals G.

3. One can determine all concept extents from knowing all **attribute extent**s $\{m\}'$, $m \in M$ [and all concept intents from all **object intent**s $\{g\}'$, $g \in G$] because
   - every extent is an intersection of attribute extents,
   - [and every intent is the intersection of object intents].

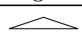
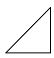
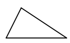
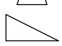
   This is a consequence of the fact that the attribute concepts are infimum-dense and the object concepts are supremum-dense.

This leads to the following

Instruction how to determine all formal concepts of a small formal context

1. Initialize a list of concept extents. To begin with, write for each attribute $m \in M$ the attribute extent $\{m\}'$ to this list (if not already present).
2. For any two sets in this list, compute their intersection. If the result is a set that is not yet in the list, then extend the list by this set. With the extended list, continue to build all pairwise intersections.
3. If for any two sets of the list their intersection is also in the list, then extend the list by the set G (provided it is not yet contained in the list). The list then contains all concept extents (and nothing else).
4. For every concept extent A in the list compute the corresponding intent A' to obtain a list of all formal concepts (A, A') of (G, M, I).

An example We illustrate the method by means of an example from elementary geometry. The objects of our example are seven triangles. The attributes are five standard properties that triangles may or may not have:

Triangles				
abbreviation	coordinates			diagram
T1	(0,0)	(6,0)	(3,1)	
T2	(0,0)	(1,0)	(1,1)	
T3	(0,0)	(4,0)	(1,2)	
T4	(0,0)	(2,0)	$(1,\sqrt{3})$	
T5	(0,0)	(2,0)	(5,1)	
T6	(0,0)	(2,0)	(1,3)	
T7	(0,0)	(2,0)	(0,1)	

Attributes	
symbol	property
a	equilateral
b	isoceles
c	acute angled
d	obtuse angled
e	right angled

We obtain the following formal context

	a	b	c	d	e
$T1$		×		×	
$T2$		×			×
$T3$			×		
$T4$	×	×	×		
$T5$				×	
$T6$		×	×		
$T7$					×

Following the above instruction, we proceed:

1. Write the attribute extents to a list.

No.		extent	found as
e_1	:=	$\{T_4\}$	$\{a\}'$
e_2	:=	$\{T_1, T_2, T_4, T_6\}$	$\{b\}'$
e_3	:=	$\{T_3, T_4, T_6\}$	$\{c\}'$
e_4	:=	$\{T_1, T_5\}$	$\{d\}'$
e_5	:=	$\{T_2, T_7\}$	$\{e\}'$

2. Compute all pairwise intersections, and

3. add G

No.		extent	found as
e_1	:=	$\{T_4\}$	$\{a\}'$
e_2	:=	$\{T_1, T_2, T_4, T_6\}$	$\{b\}'$
e_3	:=	$\{T_3, T_4, T_6\}$	$\{c\}'$
e_4	:=	$\{T_1, T_5\}$	$\{d\}'$
e_5	:=	$\{T_2, T_7\}$	$\{e\}'$
e_6	:=	$\emptyset$	$e_1 \cap e_4$
e_7	:=	$\{T_4, T_6\}$	$e_2 \cap e_3$
e_8	:=	$\{T_1\}$	$e_2 \cap e_4$
e_9	:=	$\{T_2\}$	$e_2 \cap e_5$
e_{10}	:=	$\{T_1, T_2, T_3, T_4, T_5, T_6, T_7\}$	step 3

4. Compute the intents

Concept No.	(extent	,	intent)
1	$(\{T_4\}$	,	$\{a, b, c\})$
2	$(\{T_1, T_2, T_4, T_6\}$	,	$\{b\})$
3	$(\{T_3, T_4, T_6\}$	,	$\{c\})$
4	$(\{T_1, T_5\}$	,	$\{d\})$
5	$(\{T_2, T_7\}$	,	$\{e\})$
6	$(\emptyset$	,	$\{a, b, c, d, e\})$
7	$(\{T_4, T_6\}$	,	$\{b, c\})$
8	$(\{T_1\}$	,	$\{b, d\})$
9	$(\{T_2\}$	,	$\{b, e\})$
10	$(\{T_1, T_2, T_3, T_4, T_5, T_6, T_7\}$	,	$\emptyset)$

We have now computed all ten formal concepts of the triangles–context. The last step can be skipped if we are not interested in an explicit list of all concepts, but just in computing a line diagram.

2.3.2 Drawing concept lattices

Based on one of the lists 3. or 4., we can start to draw a diagram. Before doing so, we give two simple definitions.

Definition 14 Let (A_1, B_1) and (A_2, B_2) be formal concepts of (G, M, I).

We say that (A_1, B_1) is a **proper subconcept** of (A_2, B_2), if $(A_1, B_1) \leq (A_2, B_2)$ and, in addition, $(A_1, B_1) \neq (A_2, B_2)$ holds. As an abbreviation, we write $(A_1, B_1) < (A_2, B_2)$.

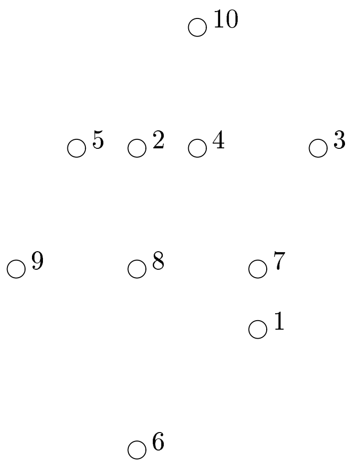
We say that (A_1, B_1) is a **lower neighbour** of (A_2, B_2), if $(A_1, B_1) < (A_2, B_2)$, but no formal concept (A, B) of (G, M, I) exists with $(A_1, B_1) < (A, B) < (A_2, B_2)$. The abbreviation for this is $(A_1, B_1) \prec (A_2, B_2)$. $\Diamond$

Instruction how to draw a line diagram of a small concept lattice

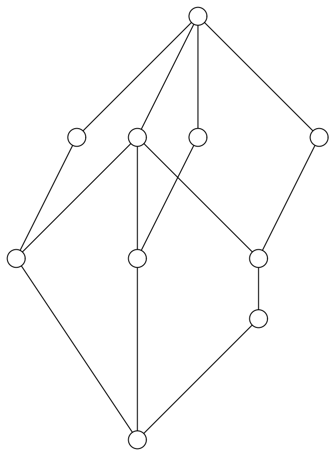
5. Take a sheet of paper and draw a small circle for every formal concept, in the following manner: a circle for a concept is always positioned higher than the all circles for its proper subconcepts.
6. Connect each circle with the circles of its lower neighbors.
7. Label with attribute names: attach the attribute m to the circle representing the concept $(\{m\}', \{m\}'')$.
8. Label with object names: attach each object g to the circle representing the concept $(\{g\}'', \{g\}')$.

We follow these instructions.

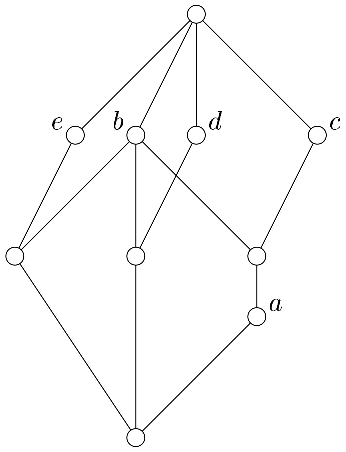
5. Draw a circle for each of the formal concepts:



6. Connect circles with their lower neighbours:



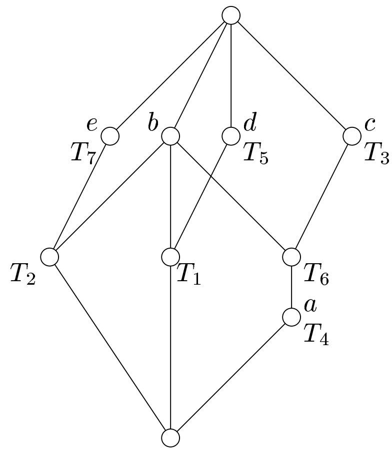
7. Add the attribute names:



8. Determine the object concepts

Object g	object intent $\{g\}'$	No. of concept
T_1	$\{b, d\}$	8
T_2	$\{b, e\}$	9
T_3	$\{c\}$	3
T_4	$\{a, b, c\}$	1
T_5	$\{d\}$	4
T_6	$\{b, c\}$	7
T_7	$\{e\}$	5

and add the object names to the diagram:



Ready! Usually it takes some tries before a nice, readable diagram is achieved. Finally we can make the effort to avoid abbreviations and to increase the readability. The result is shown in Figure 2.2.

2.3.3 Clarifying and reducing a formal context

There are context manipulations that simplify a formal context without changing the diagram, except for the labellig. It is usually advisable to do these manipulations first, before starting computations.

The simplest operation is **clarification**, which refers to indentifying "equal rows" of a formal context, and "equal columns" as well. What is meant is that

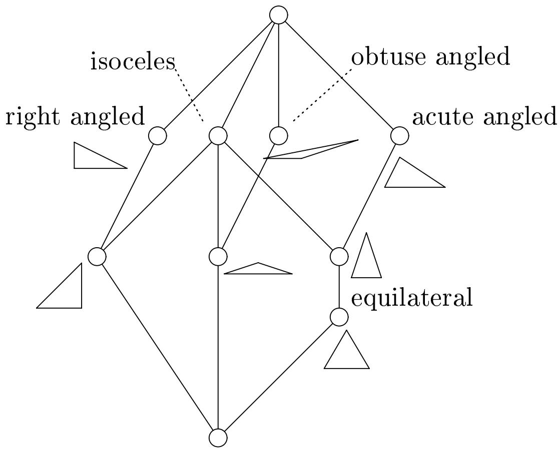


Figure 2.2: A diagram of the concept lattice of the triangle context.

if a context contains objects $g_1, g_2, \ldots$ with $\{g_i\}' = \{g_j\}'$ for all i, j, that is, objects which have exactly the same attributes, then these can be replaced by a single object, the name of which is just the list of names of these objects. The same can be done for attributes with identical attribute extent.

Definition 15 We say that a formal context is **clarified** if no two of its object intents are equal and no two of its attribute extents are equal. $\Diamond$

A stronger operation is **reduction**, which refers to omitting attributes that are equivalent to combinations of other attributes (and dually for objects). For defining reduction it is convenient to work with a clarified context.

Definition 16 An attribute m of a clarified context is called **reducible** if there is a set $S \subseteq M$ of attributes with $\{m\}' = S'$, otherwise it is **irreducible**. Reduced objects are defined dually. A formal context is called **reduced**, if all objects and all attribues are irreducible. $\Diamond$

$\{m\}' = S'$ means that an object g has the attribute m if and only if it has all the attributes from S. If we delete the column m from our cross table, no essential information is lost because we can reconstruct this column from the data contained in other columns (those of S). Moreover, deleting that column does not change the number of concepts, nor the concept hierarchy, because $\{m\}' = S'$ implies that m is in the intent of a concept if and only if S is contained in that intent. The same is true for reducible objects and concept extents. Deleting a reducible object from a formal context does not change the structure of the concept lattice.

It is even possible to remove several reducible objects and attributes simultaneously from a formal context without any effect on the lattice structure, as long as the number of removed elements is finite.

Definition 17 Let (G, M, I) be a finite context, and let G_{irr} be the set of irreducible objects and M_{irr} be the set of irreducible attributes of (G, M, I). The context $(G_{irr}, M_{irr}, I \cap G_{irr} \times M_{irr})$ is the **reduced context** corresponding to (G, M, I).

For a finite lattice L let $J(L)$ denote the set of its supremum-irreducible elements and let $M(L)$ denote the set of its infimum-irreducible elements. Then $(J(L), M(L), \leq)$ is the **standard context** for the lattice L. $\Diamond$

Proposition 3 *A finite context and its reduced context have isomorphic concept lattices. For every finite lattice L there is (up to isomorphism) exactly one reduced context, the concept lattice of which is isomorphic to L, namely its standard context.*

2.3.4 Other algorithms for computing concept lattices

The algorithm which we shall use in what follows to generate all formal concepts of a given formal context can be formulated without referring to concept lattices. It will be an *algorithm to generate all closed sets of a closure system*, called NEXTCLOSURE. It can be applied to concept lattices because for each formal context (G, M, I), the set $\mathbf{Ext}(G, M, I)$ of all concept extents is a closure system, as well as the set $\mathbf{Int}(G, M, I)$ of all the concept intents of (G, M, I).

———(to be written)———

- Titanic
- Nourine
- ...

2.3.5 Computer programs for concept generation and lattice drawing

There are several free or conditionally free programs available for concept lattice generation and/or drawing. A well known one, old fashioned but reliable, is CONIMP (for DOS), available under `www.mathematik.tu-darmstadt.de`. S. Yevtushenko provides the Concept Explorer, which allows for editing formal contexts and computing their concept lattices. It can be downloaded from `http://sourceforge.net/projects/conexp/`. A rather advanced package, invented for applications in data analysis, is the NAVICON DECISION SUITE by NAVICON GmbH. A demo version is available under `www.navicon.de`.

2.4 Additive and Nested Line Diagrams

In this section, we discuss possibilities to generate line diagrams both automatically or by hand. A list of some dozens of concepts may already be quite difficult to survey, and it requires practice to draw good line diagrams of concept lattices with more than 20 elements.

The best and most versatile form of representation for a concept lattice is a well drawn line diagram. It is however tedious to draw such a diagram by hand and one would wish an automatic generation by means of a computer. We know quite a few algorithms to do this, but none which provides a general satisfactory solution. It is by no means clear which qualities make up a *good* diagram. It should be transparent, easily readable and should facilitate the interpretation of the data represented. How this can be achieved in each individual case depends however on the aim of the interpretation and on the structure of the lattice. Simple optimization criteria (minimization of the number of edge crossings, drawing in layers, etc.) often bring about results that are unsatisfactory. Nevertheless, automatically generated diagrams are a great help: they can serve as the starting point for drawing by hand. Therefore, we will describe simple methods of generating and manipulating line diagrams by means of a computer.

2.4.1 Additive line diagrams

We will now explain a method where a computer generates a diagram and offers the possibility of improving it interactively. Programming details are irrelevant in this context. We will therefore only give a **positioning rule** which assigns points in the plane to the elements of a given ordered set $(P, \leq)$. If a and b are elements of P with $a < b$, the point assigned to a must be lower than the point assigned to b (i.e., it must have a smaller y-coordinate). This is guaranteed by our method. We will leave the computation of the edges and the checking for undesired coincidences of vertices and edges to the program. We do not even guarantee that our positioning is injective (which of course is necessary for a correct line diagram). This must also be checked if necessary.

Definition 18 A **set representation** of an ordered set $(P, \leq)$ is an order embedding of $(P, \leq)$ in the power-set of a set X, i.e., a map

$$\text{rep} : P \to \mathfrak{P}(X)$$

with the property

$$x \leq y \iff \text{rep}\, x \subseteq \text{rep}\, y.$$

$\Diamond$

An example of a set representation for an arbitrary ordered set $(P, \leq)$ is the assignment

$$X := P, \quad a \mapsto (a].$$

In the case of a concept lattice,

$$X := G, \quad (A, B) \mapsto A$$

is a set representation.

$$X := M, \quad (A, B) \mapsto M \setminus B$$

is another set representation, and both can be combined to

$$X := G \,\dot{\cup}\, M, \quad (A, B) \mapsto A \cup (M \setminus B).$$

It is sufficient to limit oneself to the irreducible objects and attributes.

For an **additive line diagram** of an ordered set $(P, \leq)$ we need a set representation $\text{rep} : P \to \mathfrak{P}(X)$ as well as a **grid projection**

$$\text{vec} : X \to \mathbb{R}^2,$$

assigning a real vector with a positive y-coordinate to each element of X. By

$$\text{pos}\, p := n + \sum_{x \in \text{rep}\, p} \text{vec}\, x$$

we obtain positioning of the elements of P in the plane. Here, n is a vector which can be chosen arbitrarily in order to shift the entire diagram. By only allowing positive y-coordinates for the grid projection we make sure that no element p is positioned below an element q with $q < p$.

Every finite line diagram can be interpreted as an additive diagram with respect to an appropriate set representation. For concept lattices we usually use the representation by means of the irreducible objects and/or attributes. The resulting diagrams are characterized by a great number of parallel edges, which improves their readability. Experience shows that the set representation by means of the irreducible attributes is most likely to result in an easily interpretable diagram. Figure 2.2 for instance was obtaining by selecting the irreducible attributes for the set representation.

It is particularly easy to manipulate these diagrams: If we change –the set representation being fixed– the grid projection for an element $x \in X$, this means that all images of the order filter $\{p \in P \mid x \in \operatorname{rep} p\}$ are shifted by the same distance and that all other points remain in the same position. In the case of the set representation by means of the irreducibles these order filters are precisely principal filters or complements of principal ideals, respectively. This means that we can manipulate the diagram by shifting principal filters or principal ideals, respectively, and leaving all other elements in position.

Even carefully constructed line diagrams loose their readability from a certain size up, as a rule from around 50 elements up. One gets considerably further with *nested line diagrams* which will be introduced next. However, these diagrams do not only serve to represent larger concept lattices. They offer the possibility to visualize how the concept lattice changes if we add further attributes.

2.4.2 Nested line diagrams

Nested line diagrams permit a satisfactory graphical representation of somewhat larger concept lattices. The basic idea of the nested line diagram consists of clustering parts of an ordinary diagram and replacing bundles of parallel lines between these parts by one line each. Thus, a nested line diagram consists of ovals, which contain clusters of the ordinary line diagram and which are connected by lines. In the simplest case two ovals which are connected by a simple line are congruent. Here, the line indicates that corresponding circles within the ovals are direct neighbors, resp.

Furthermore, we allow that two ovals connected by a single line do not necessarily have to be congruent, but they may each contain a part of two congruent figures. In this case, the two congruent figures are drawn in the ovals as a "background structure", and the elements are drawn as bold circles if they are part of the respective substructures. The line connecting the two boxes then indicates that the respective pairs of elements of the background shall be connected with each other. An example is given in Figure 2.3. It is a screenshot of a library information system which was set up for the library of the Center on Interdisciplinary Technology Research of Darmstadt University of Technology.

Nested line diagrams originate from partitions of the set of attributes. The basis is the following theorem:

Theorem 2 *Let (G, M, I) be a context and $M = M_1 \cup M_2$. The map*

$$(A, B) \mapsto (((B \cap M_1)', B \cap M_1), ((B \cap M_2)', B \cap M_2))$$

is a $\bigvee$-preserving order embedding of $\mathfrak{B}(G, M, I)$ in the direct product of $\underline{\mathfrak{B}}(G, M_1, I \cap G \times M_1)$ and $\underline{\mathfrak{B}}(G, M_2, I \cap G \times M_2)$. The component maps

$$(A, B) \mapsto ((B \cap M_i)', B \cap M_i)$$

are surjective on $\underline{\mathfrak{B}}(G, M_i, I \cap G \times M_i)$.

Proof If (A, B) is a concept of (G, M, I), then $B \cap M_i$ is the set of all attributes common to the objects of A in the context $(G, M_i, I \cap G \times M_i)$, i.e., it is an intent of this context. Hence the above-mentioned assignment is really a map into the product. The union of the intents $B \cap M_1$ and $B \cap M_2$ again yields B, i.e., the map is injective. The fact that it is furthermore $\bigvee$-preserving (and thus an order-embedding) can again be seen from the concept intents. It remains to be shown

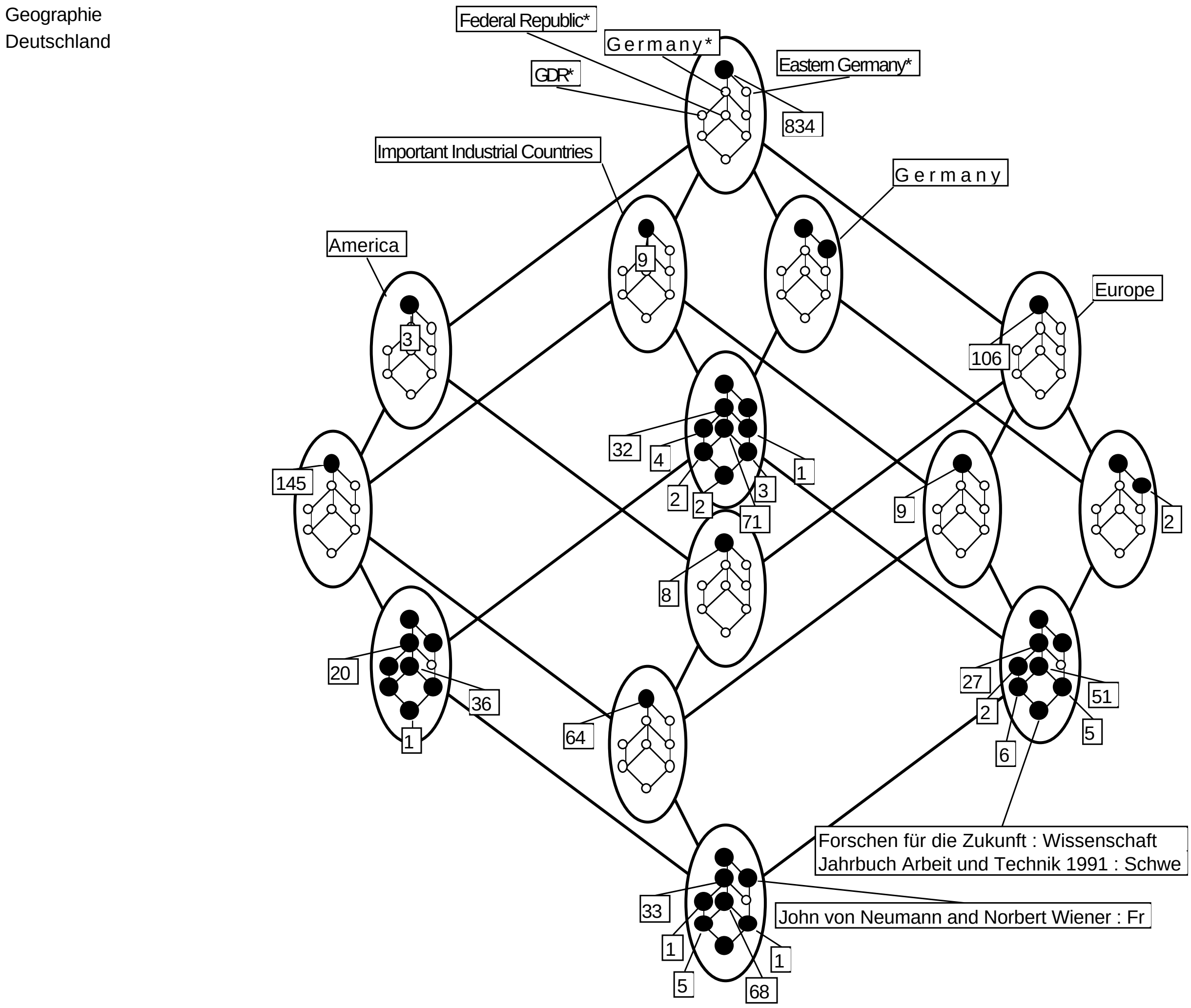


Figure 2.3: Nested line diagram of a library information system

that the component maps are surjective. Let C be an intent of $(G, M_i, I \cap G \times M_i)$. Then $B := C^{II}$ is an intent of (G, M, I) with $B \cap M_i = C$, i.e., the image of the concept (B', B) of (G, M, I) under the ith component map is the concept with the intent C. $\square$

In order to sketch a nested line diagram, we proceed as follows: First of all we split up the attribute set: $M = M_1 \cup M_2$. This splitting up does not have to be disjoint. More important for interpretation purposes is the idea that the sets M_i bear meaning. Now, we draw line diagrams of the subcontexts $\mathbb{K}_i := (G, M_i, I \cap G \times M_i)$, $i \in \{1, 2\}$ and label them with the names of the objects and attributes, as usual. Then we sketch a nested diagram of the product of the concept lattices $\underline{\mathfrak{B}}(\mathbb{K}_i)$ as an auxiliary structure. For this purpose we draw a large copy of the diagram of $\underline{\mathfrak{B}}(\mathbb{K}_1)$, representing the lattice elements not by small circles but by congruent ovals, which contain each a diagram of $\underline{\mathfrak{B}}(\mathbb{K}_2)$.

By Theorem 2 the concept lattice $\underline{\mathfrak{B}}(G, M, I)$ is embedded in this product as a $\bigvee$-semilattice. If a list of the elements of $\underline{\mathfrak{B}}(G, M, I)$ is available, we can enter them into the product according to their intents. If not, we enter the object concepts the intents of which can be read off directly from the context, and form all suprema.

This at the same time provides us with a further, quite practicable method of determining a concept lattice by hand: split up the attribute set as appropriate, determine the (small) concept lattices of the subcontexts, draw their product in form of a nested line diagram, enter the object concepts and close it against suprema. This method is particularly advisable in order to arrive at a useful diagram quickly.

2.5 Examples

——–(to be written)——–

2.6 Exercises

1. Give the formal context for the concept lattice shown in Figure 2.1.
2. Give a proof of Proposition 1.
3. Let (A_1, B_1) and (A_2, B_2) be formal concepts of a formal context (G, M, I). Show that the expression 'greatest common subconcept' in Lemma 2 is justified by showing
   (a) $(A_1, B_1) \wedge (A_2, B_2)$ is a formal concept (of the same context).
   (b) $(A_1, B_1) \wedge (A_2, B_2)$ is a subconcept of both (A_1, B_1) and (A_2, B_2).
   (c) Any other common subconcept of (A_1, B_1) and (A_2, B_2) is also a subconcept of $(A_1, B_1) \wedge (A_2, B_2)$.
4. Show that, in a concept lattice, the set of all object concepts is supremum-dense. Hint: Show that each formal concept is the supremum of the set of all its subconcepts which are also object concepts.
5. Let $(V, \leq)$ be a partially ordered set where every subset A of V has an infimum. Show that $(V, \leq)$ is a complete lattice.
6. Show that every non-empty finite lattice is a complete lattice.
7. Show that in every lattice the following equivalences hold:

$$x \leq y \iff x \wedge y = x \iff x \vee y = y$$

   Remark: From the knowledge about infima and suprema of a lattice, we are thus able to reconstruct its order relation.
8. Prove Lemma 3: For any formal context (G, M, I), the concept lattices $(\underline{\mathfrak{B}}((G, M, I)))^d$ and $\underline{\mathfrak{B}}(M, G, I^{-1})$ are isomorphic.
9. Let $\mathbb{V}$ and $\mathbb{V}$ be two complete lattices, and $\varphi: V \to W$ be a bijective mapping. Show that φ is an isomorphism of complete lattices if and only if, for all subsets X of V,

$$\varphi\left(\bigvee X\right) = \bigvee \{\varphi(x) \mid x \in X\}$$

   and

$$\varphi\left(\bigwedge X\right) = \bigwedge \{\varphi(x) \mid x \in X\} \quad .$$