Mizar Formalization of Concept Lattices

Christoph Schwarzweller

Fakultät für Informatik
Universität Tübingen
Sand 13, D-72076 Tübingen, Germany
schwarzw@informatik.uni-tuebingen.de
Phone: +49 7071 78954
Fax: +49 7071 295958

Abstract – In this paper we report on the Mizar codification of formal concepts and concept lattices being part of the theory of formal concept analysis, a mathematical theory that formally describes the notion of concepts and concept hierarchies.

We give a short introduction to the theory of concept lattices and describe how we modelled these structures in the Mizar language. After that we discuss experiences and problems concerning the formalization; in particular we deal with reusing mathematical knowledge stored in the Mizar Mathematical Library.

Keywords – formalized mathematics, lattice theory, formal concept analysis, concept lattices, reusing mathematical knowledge.

1. Introduction

Concept lattices stem from the theory of formal concept analysis ([3],[9]). The aim of this theory is a formal mathematical treatment of concepts and concept hierarchies, hence to bring formal methods into the field of data analysis and knowledge processing. Its mathematical basics are in fact nothing more than applied lattice theory. Starting with an elementary representation of data --- a so-called formal context C consisting of a set of objects, a set of attributes and a relation that indicates whether an object has a particular attribute or not--- formal concepts CP are defined as a pair consisting of a set of objects and a set of attributes fulfilling some additional requirements. Now the set of formal concepts CP over a given formal context C can be equipped with a subconcept ordering and it turns out that the set of formal concepts constitutes a complete lattice with respect to this ordering. More details on what concept lattices are and how we formalized them in the Mizar language will be given in Sections 2 and 3.

The reason for choosing the theory of concept lattices to be formalized in the Mizar language is twofold. First, as already mentioned concept lattices stem from so-called formal concept analysis. This theory has a clear and rather practical intention. In fact, a great part of the theory goes into formulating theorems that allow the development of algorithms analyzing data represented as concept lattices. For example, dependencies and implications of attributes can be computed. So our work can be seen as a step towards applications of the Mizar system.

On the other hand, the theory of concept lattices is strongly based on the theory of (complete) lattices. In the Mizar Mathematical Library, there are quite a few articles dealing with these structures. So we need not build our theory from scratch; instead we can reuse the knowledge about (complete) lattices already included in the Mizar library. This is an interesting experiment because it explores the possibility of reusing knowledge of the Mizar library. Note that the definition of lattices in Mizar was intended to enable the codification of continuous lattices rather than to be used from a different point of view. We will return to this point in Section 4.

2. Basic Lattice Theory

In the following we shortly summarize the basic notations of the theory of lattices necessary for the rest of the paper (see for example [2]). The reader familiar with lattice theory may skip this section.

Lattices are special partial orders: a partially ordered set $(P,\leq)$ is a lattice, if for each two elements $x, y \in P$ both the supremum of x and y and the infimum of x and y exist. A lattice L in which for each (not necessary finite) subset $X \subseteq L$ the supremum of X and the infimum of X exist is called a complete lattice. Note that every finite lattice is complete.

There is another equivalent possibility to introduce lattices: Let L be a set and let $\wedge$ and $\vee$ be binary operations over L. Then L is a lattice if $\wedge$ and $\vee$ both are associative and commutative and fulfill the absorption law, that is for arbitrary $x, y, z \in L$ holds

$$\begin{array}{ll} x \wedge y = y \wedge x, & x \vee y = y \vee x, \\ x \wedge (y \wedge z) = (x \wedge y) \wedge z, & x \vee (y \vee z) = (x \vee y) \vee z, \\ (x \wedge y) \vee y = y, & (x \vee y) \wedge y = y. \end{array}$$

Now given such an algebraic definition of lattices one gets back the partial ordering by defining $x \leq\leq y :\Leftrightarrow x \vee y = y$. On the other hand, starting with lattices represented as partially ordered sets, one defines $x \wedge y$ to be the infimum and $x \vee y$ to be the supremum of x and y. Then one gets two binary operations fulfilling the algebraic laws mentioned. So these two representations of lattices are indeed equivalent.

In addition we need the notion of supremum density and infimum density of subsets of a complete lattice. A subset X of a complete lattice L is called supremum dense (infimum dense) if each $x \in L$ can be represented as the supremum (infimum) of a subset of X. Note that this definition is independent of the representation of the lattice just because suprema and infima are defined for both representations.

We also introduce dual lattices. Of course dual lattices exist for both representations of lattices: given a lattice $(L,\leq)$, the dual lattice of L is defined as $(L,\geq)$, that is one gets the dual lattice of L by taking L's reverse order. The corresponding definition for algebraically represented lattices is also quite simple: one only has to exchange the operations $\wedge$ and $\vee$, that is given such a lattice $(L,\wedge,\vee)$, its dual lattice is $(L,\vee,\wedge)$. It should be clear that although we defined dual lattices for both representations, the definition in fact again is independent of the representation.

3. Some Details of the Codification

In this section we describe how we formalized concept lattices in the Mizar language. The key point here is that we put our theory of concept lattices on top of the theory of lattices already defined in Mizar, that is we strongly reuse mathematical knowledge included in the Mizar library not being developed for our purpose. In addition this section is also intended to serve as a small introduction to concept lattices as given in [3].

We start with the elementary representation of data consisting of a set of objects, a set of attributes and a relation between them as already mentioned in Section 1. We modelled this as a Mizar structure called ContextStr consisting of three selectors: the Objects, the Attributes and the Information.

```
definition
struct (2-sorted) ContextStr
        (# Objects -> set,
              Attributes -> set,
              Information -> Relation of the Objects, the Attributes #);
end;
```

Now a formal context C is just a non-empty context structure. Note that a (finite) formal context can be seen as a two-dimensional table: the rows of the table correspond to the objects, the columns to the attributes. A cross in row o and column a indicates that the pair (o,a) belongs to the given relation, that is whether (o,a) ∈ the Information of C holds.

As already mentioned a formal concept CP over a given formal context C is a pair consisting of a set of objects and a set of attributes that fulfills some additional requirements. These requirements concern the so-called derivation of objects and attributes respectively. Given a set O of objects (A of attributes) one computes the set of attributes a (objects o) common to all objects (attributes) belonging to O (A).[1] Of course, the resulting set of objects (attributes) strongly depends on the given formal context C. The Mizar formalization of these two operators, which we called ObjectDerivation and AttributeDerivation, is as follows, where o is-connected-with a is just a shorthand for (o,a) ∈ the Information of C.

```
definition
let C be FormalContext;
func ObjectDerivation(C) ->
    Function of bool(the Objects of C), bool(the Attributes of C) means
for O being Element of bool(the Objects of C) holds
it.O = {a where a is Attribute of C :
              for o being Object of C st o ∈ O holds o is-connected-with a};
end;

definition
```

[1] For the reader familiar with lattice theory we mention that these operators constitute a Galois connection between the power set of objects and the power set of attributes ([3]).

```
let C be FormalContext;
func AttributeDerivation(C) ->
    Function of bool(the Attributes of C), bool(the Objects of C) means
for A being Element of bool(the Attributes of C) holds
it.A = {o where a is Object of C :
               for a being Attribute of C st a ∈ A holds o is-connected-with a};
end;
```

Using these derivation operators we now can define formal concepts: a formal concept CP consists of a set of objects and a set of attributes, called the Extent and the Intent of the concept CP, with the additional requirement that these two sets are closed with respect to the just defined derivations, that is the derivation of the Extent exactly gives the Intent and vice versa. In the Mizar language we model this in two steps: we first define a new structure followed by a Mizar attribute describing the requirement.

```
definition
let C be 2-sorted;
struct  ConceptStr over C
        (# Extent -> Subset of the Objects of C,
             Intent -> Subset of  the Attributes of C #);
end;
definition
let C be FormalContext;
let CP be ConceptStr over C;
attr CP is concept-like means
    (ObjectDerivation(C)).(the Extent of CP) = the Intent of CP &
    (AttributeDerivation(C)).(the Intent of CP) = the Extent of CP;
end;
```

Now a formal concept CP over a given formal context C simply is a concept-like non-empty concept structure over C. The set of all formal concepts CP over a given formal context C will be the carrier of the concept lattice over C. In our codification the concept lattice is given by the functor ConceptLattice(C) where C is the given formal context.

The ordering necessary to consider formal concepts C from the viewpoint of lattice theory is nothing more than an inclusion ordering: CP1 $\leq$ CP2 --- or equivalently CP1 is a subconcept of CP2 --- holds, if the Extent of CP1 is a subset of the Extent of CP2. We mention that due to the principle of duality we may also have used the intent of concepts to define the ordering.[2]

```
definition
let C be FormalContext;
let CP1, CP2 be FormalConcept of C;
pred CP1 is-SubConcept-of CP means
```

[2] To be more precise using our definition of subconcept the following theorem holds: Given CP1 and CP2 being formal concepts CP1 is a subconcept of CP2 if and only if the Intent of CP2 $\subseteq$ the Intent of CP1.

```
    the Extent of CP1 c= the Extent of CP2;
end;
```

The first important point here is that given an arbitrary formal context C the set of all formal concepts CP over C not only forms a lattice with respect to the just defined ordering, but even a complete lattice ([3],[9]) --- the so-called concept lattice over C. In the Mizar language this can be formalized as the following theorem.[3]

```
theorem
for C being FormalContext holds ConceptLattice(C) is complete;
```

Hence the mathematical foundation of formal concept analysis in fact is an application of the theory of (complete) lattices. As we will see in the following, concept lattices even capture all existing complete lattices.

The next theorem ([3],[9]) gives a characterization of concept lattices in terms of ordinary lattice theory. It states a necessary and sufficient condition for a complete lattice L to be isomorphic to a concept lattice over a given formal context C. Note that the attribute are_isomorphic is not specific with respect to concept lattices, but is the ordinary one for lattices already being part of the Mizar library.

```
theorem
for L being complete Lattice
for C being FormalContext holds
ConceptLattice(C),L are_isomorphic
iff ex g being Function of the Objects of C, the carrier of L,
            d being Function of the Attributes of C, the carrier of L
     st rng(g) is supremum-dense & rng(d) is infimum-dense &
          for o being Object of C, a being Attribute of C holds
          o is-connected-with a iff g.o ≤≤ d.a;
```

This theorem easily entails that every complete lattice is isomorphic to a concept lattice. As a by-product we get that the theory of concept lattices in fact is identical with the theory of complete lattices. We proved the following theorem in [6].

```
theorem
for L being Lattice holds
L is complete iff
ex C being FormalContext st ConceptLattice(C),L are_isomorphic;
```

We close this section by describing dual concept lattices. First we define dual formal contexts by exchanging objects and attributes and taking the reverse relation (denoted by the functor ~) as the Information of the formal context. Note that we are overloading the operator °, which already has been introduced in the Mizar article LATTICE2 [7] where it

[3] Or of course as a cluster definition.

takes a lattice as its argument.[4]

```
definition
let C be FormalContext;
func C° -> strict non empty ContextStr equals
    ContextStr (# the Attributes of C, the Objects of C, (the Information of C)~ #);
end;
```

Now because ConceptLattice(C) is a lattice, its dual lattice (ConceptLattice(C))° is well defined. It turns out that building the concept lattice over a dual context C° is the same as first building the concept lattice over C and then taking its dual lattice in the sense of Section 2. To be more precise, one can prove the following Mizar theorem.

```
theorem
for C being FormalContext holds
ConceptLattice(C°),(ConceptLattice(C))° are_isomorphic;
```

In other words building the concept lattice over a suitable formal context, namely the dual formal context, already captures all dual concept lattices. Note also that the functor ° if applied to a lattice, in particular to a concept lattice, again is the functor being already part of the Mizar library ([7]) and is not specific to concept lattices, which again demonstrates how the theory of concept lattices is put on top of the theory of (complete) lattices.

4. Experiences

First we want to say that Mizar formulation of mathematical objects is rather straightforward --- at least in the field of algebra and lattice theory --- so that even beginners can easily cope with this. Problems are caused in the environment part in which one has to import the necessary preliminaries of the article: one has to identify the articles of the Mizar library in which the notation one needs for the new article has been introduced. In addition, finding theorems in the Mizar Mathematical Library that may be helpful in proving one's own theorems is rather difficult.[5] For example, although dual lattices are defined in the article LATTICE2 using the algebraic representation, the corresponding theorem about the induced orders (compare Section 2), to be more precise the theorem stating that given an algebraically represented lattice L the order of L's dual lattice equals the reverse order of L, is part of a different article LATTICE3. It is obvious that this cannot be changed, just because some parts of LATTICE3 are necessary to formulate and prove this theorem. But, it makes finding theorems about dual lattices very difficult. Both problems could be solved by introducing more structure into the Mizar library or by extending the Mizar system with powerful search tools. This also would improve Mizar's capabilities concerning the reuse of knowledge. On the other hand, if we ignore these rather technical aspects of reusing knowledge of the Mizar library, the facilities we find are in fact satisfying, as we will see in the

---

[4] There the operator ° also denotes the dual lattice.
[5] At this point we have to mention the excellent Mizar User Service which relativizes these problems.

following.

As already mentioned lattices can be defined in two equivalent ways (compare Section 2). Consequently in the Mizar Mathematical Library there are two different representations of lattices: the first one is based on partially ordered sets[6]; the second one is the algebraic structure[7] given by a carrier and two operations usually denoted by $\wedge$ and $\vee$. We defined concept lattices using the latter representation. At a first glance this seems to be a mistake as in the Mizar library lattice theory based on the first representation is much more developed. But --- because the Mizar library comes with operators converting objects from one representation into the other one --- it turned out that it is rather easy to change the representation, apply theorems formulated for the second representation and translate back the results. Let us illustrate this with a rather trivial example:

In lattices defined using the operations $\wedge$ and $\vee$, an ordering $\leq\leq$ can be defined by a $\leq\leq$ b $:\Leftrightarrow$ a $\vee$ b = b as already has been pointed out in Section 2. Given this definition assume that we want to prove transitivity of the ordering $\leq\leq$:

theorem
for L being Lattice
for a,b,c being Element of the carrier of L holds (a ≤≤ b & b ≤≤ c) implies a ≤≤ c

Assume further that for lattices based on partially ordered sets we know that the following corresponding theorem holds. T is just a label used by the Mizar checker to apply the theorem in later proofs.

theorem T:
for L being LATTICE
for a,b,c being Element of the carrier of L holds (a ≤ b & b ≤ c) implies a ≤ c;

The transformation operator we will use to prove the theorem[8] using T is denoted by % ([1]): given an algebraically represented lattice L, an element a $\in$ L is converted into an element of the corresponding lattice based on partially ordered sets, that is theorems formulated for LATTICE are directly applicable to a%. By using this, the proof of our theorem consists of the three already mentioned steps: converting the elements (this is done using theorem LATTICE3:7 which also states that the transformation respects the orderings), applying theorem T from above (which is formulated for LATTICE) and reconverting the elements (which again is done using theorem LATTICE3:7):

proof
let L be Lattice;
let a,b,c be Element of the carrier of L;
assume A: a ≤≤ b & b ≤≤ c;
then B: a% ≤ b% by LATTICE3:7;
b% ≤ c% by A,LATTICE3:7;

[6] Called LATTICE in the Mizar Mathematical Library.
[7] Called Lattice in the Mizar Mathematical Library.
[8] In this case it is of course trivial to prove the theorem directly.

```
then a% ≤ c% by B,T;
hence a ≤≤ c by LATTICE3:7;
end;
```

A rather small problem with this proof we also want to mention is that the transformation operator % as it is defined in [1] does not exactly convert from Lattice into LATTICE, but only to partially ordered sets, or to be more precise to LattPOSet L, if L is the given Lattice. Of course L being an algebraically based lattice implies that LattPOSet L also is a lattice based on partially ordered sets, so we can formulate the following cluster about LattPOSet L.

```
definition
let L be Lattice;
cluster LattPOSet L -> with_infima with_suprema;
end;
```

The attributes with_infima and with_suprema defined in [1] state the existence of suprema and infima respectively, hence in our case that LattPOSet L indeed is a lattice.

After this cluster definition --- which in fact is nothing more than a reformulation of theorem LATTICE3:11 --- the proof from above is accepted as it is. This means that in Mizar it is possible to switch from one representation of lattices into the other without any major problem. Consequently all theorems proved for one representation are also applicable to the other one --- just as we are used to from usual mathematical lattice theory. From a technical point of view this means that reusing knowledge of the Mizar library is possible even if the actual representation slightly differs from the former one. We plan to explore this point in more detail, but we believe that the introduction of additional cluster definitions like in our example will enable us to comfortably switch between the two Mizar representations of lattices.

A last point we want to mention here is concerned with the length of our codification. The two articles that we wrote about concept lattices [5], [6] both have a length of about 4000 lines of Mizar code. The corresponding part of the textbook [3], which is the basis of our formalization, is only about eight pages.

At a first glance this seems to be an unacceptable proportion having its origins in the Mizar proof checker. But, we believe that the length of the formalization is not well suited as a measure for the quality of the formalization. First, one has to take into account the quality of the source, that is how detailed the given mathematical text and its proofs are. In our case this is rather small: many facts about formal contexts and formal concepts simply occur in the text, that is they are not formulated as a lemma or a theorem. In addition, proofs of easy lemmas are left out and even if proofs are done --- due to the duality principle of lattice theory --- one often reads “The other case analogously follows.” This is no harm in mathematical textbooks, but in Mizar (like in every formal proof or proof checking system) these things have to be made explicit.

Second, we also believe that how easy it is to construct a proof that is accepted by a proof checker is more important than how long this proof is. That is one of the main advantages of the Mizar system: its proof language and its checker are very close to the usual mathematical language and natural deduction. Consequently, though proofs sometimes tend to be a bit lengthy, to get a proof accepted one can think in ordinary mathematical terms and needs not to be aware of special tricks of the checker.

5. Conclusions and Further Work

We reported on the Mizar codification of concept lattices and our experiences in doing so. These experiences are really good: though we chose the algebraic representation of lattices whose theory is not as far developed as the theory of lattices based on partially ordered sets, we had no problems with developing the theory so far. This is due to the fact that it is possible in Mizar to apply theorems formulated for one representation of lattices to the other one as we are used to doing from ordinary mathematics. Mizar models well this aspect of mathematics. Also we had no problems in putting the theory of concept lattices on top of the Mizar codified theory of (complete) lattices, that is reusing the knowledge of the Mizar Mathematical Library can be done without major problems.

We plan to go on with the formalization of concept lattices by exploring more algorithmic aspects. This includes the already mentioned computation of dependencies and implications of attributes. In addition we want to codify the reduction of formal concepts: given a formal context, one can often eliminate objects and attributes without changing the corresponding concept lattice. A theorem in [3] states that for every (finite) formal context C there exists another formal context --- the so-called standard context of C --- defining the same concept lattice which is minimal in this sense.

Codifying these properties again is a step into a more practical use of the Mizar system as they stem from the area of knowledge processing. In addition, they are the basis for algorithms analyzing formal contexts, which means that in this sense we are also verifying algorithms using the Mizar system.

Acknowledgments: I would like to thank Rüdiger Loos, Piotr Rudnicki and Andrzej Trybulec for carefully reading earlier versions of this paper. Rüdiger Loos also was the one who pointed my attention to concept lattices and formal concept analysis.

References

[1] Grzegorz Bancerek, *Complete Lattices*, Formalized Mathematics, Vol. 4, 1992, http://www.mizar.org/JFM/vol4/lattice3.html.

[2] Garrett Birkhoff, *Lattice Theory*, third edition. Amer. Math. Soc. Coll. Publ. 25, Providence, R.I., 1973.

[3] Bernhard Ganter and Rudolf Wille, *Formale Begriffsanalyse*, (in German). Springer Verlag, Berlin-Heidelberg-New York, 1996.

[4] Piotr Rudnicki, *An Overview of the Mizar Project,* http: //mizar.org/project/bibliography.html.

[5] Christoph Schwarzweller, *Introduction to Concept Lattices*, Formalized Mathematics (to appear), Vol. 10, 1998, http://www.mizar.org/vol10/ conlat_1.html.

[6] Christoph Schwarzweller, *A Characterization of Concept Lattices; Dual Concept Lattices*, Formalized Mathematics (to appear), Vol. 11, 1999, http://www.mizar.org/vol11/conlat_2.html.

[7] Andrzej Trybulec, *Finite Join, Finite Meet and Dual Lattices*, Formalized Mathematics, Vol. 2,

1990, http://www.mizar.org/vol2/ lattice2.html.

[8] Andrzej Trybulec, *Some Features of the Mizar Language*, http://mizar.org/project /bibliography.html.

[9] Rudolf Wille, *Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts*, In: Ivan Rival (ed.), Ordered Sets, Reidel, Dordrecht-Boston, 1982.