

Lattice decompositions through methods using congruence relations

Jean-François Viaud, Karell Bertet, Christophe Demko, Rokia Missaoui

▶ **To cite this version:**

Jean-François Viaud, Karell Bertet, Christophe Demko, Rokia Missaoui. Lattice decompositions through methods using congruence relations. 2016. hal-01282151

HAL Id: hal-01282151
https://hal.science/hal-01282151

Preprint submitted on 3 Mar 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/279458576

Lattice decompositions through methods using congruence relations

ARTICLE · JUNE 2015

Source: arXiv

READS

14

4 AUTHORS, INCLUDING:



Jean-François Viaud

Université de La Rochelle

4 PUBLICATIONS **0** CITATIONS

SEE PROFILE



Karell Bertet

Université de La Rochelle

54 PUBLICATIONS **158** CITATIONS

SEE PROFILE

Available from: Jean-François Viaud
Retrieved on: 03 March 2016

arXiv:1506.08658v1 [cs.DM] 29 Jun 2015

Lattice decompositions through methods using congruence relations

Jean-François Viaud and Karell Bertet and Christophe Demko and Rokia Missaoui

1 Introduction

During the last decade, the computation capabilities have promoted Formal Concept Analysis (FCA) with new methods based on concept lattices. Though they are exponential in space/time in worst case, concept lattices of a reasonable size enable an intuitive representation of data organized by a context that links objects to attributes through a binary relation. Methods based on concept lattices have been developed in various domains such as knowledge discovery and management, databases or information retrieval where some relevant concepts, *i.e.* possible correspondences between objects and attributes are considered either as classifiers, clusters or representative object/attribute subsets.

With the increasing size of data, a set of methods have been proposed in order to either generate a subset (rather than the whole set) of concepts and their neighborhood in an on-line and interactive way [10, 22] or better display lattices using nested line diagrams [15]. Such approaches become inefficient when contexts are huge. However, the main idea of lattice/context decomposition into smaller ones is still relevant when the classification property of the initial lattice is maintained. Many lattice decompositions have been defined and studied, both from an algebraic

Jean-François Viaud
Laboratory L3i, University of La Rochelle, Avenue Michel Crépeau 17000 La Rochelle, France, e-mail: jviaud@univ-lr.fr

Karell Bertet
Laboratory L3i, University of La Rochelle, Avenue Michel Crépeau 17000 La Rochelle, France, e-mail: kbertet@univ-lr.fr

Christophe Demko
Laboratory L3i, University of La Rochelle, Avenue Michel Crépeau 17000 La Rochelle, France, e-mail: cdemko@univ-lr.fr

Rokia Missaoui
University of Québec in Outaouais, Canada, e-mail: rokia.missaoui@uqo.ca

point of view [8, 18] or from an FCA point of view [15, 14]. We can cite the Unique Factorisation Theorem [18], the matrix decomposition [2], the Atlas decomposition [15], the subtensorial decomposition [15], the subdirect decomposition, or the doubling convex construction.

The subdirect decomposition has been widely studied many years ago, in the field of universal algebra [8, 13, 11, 12], and even in FCA [23, 24, 25, 26, 14]. To the best of our knowledge, there is no new development or novel algorithms for subdirect decomposition of contexts. The doubling convex construction has also been widely studied [6, 19, 16, 3], mainly from a theoretical point of view, in order to characterize lattices that can be obtained by such decomposition.

In this chapter, we investigate the subdirect decomposition of a concept lattice as a first step towards an interactive exploration and mining of large contexts. The subdirect decomposition of a lattice L into factor lattices $(L_i)_{i\in\{1,\dots,n\}}$, denoted by $L \hookrightarrow L_1 \times \cdots \times L_n$, is defined by two properties (see important results in [15]): (i) L is a sublattice of the direct product $L_1 \times \cdots \times L_n$, and (ii) each projection of L onto a factor is surjective. The first property establishes that each factor lattice is the concept lattice of an arrow-closed subcontext, *i.e.* closed according to the arrow relation between objects and attributes. This means that the decomposition can be obtained by computing specific subcontexts. The second property states that there is an equivalence between arrow-closed subcontexts and congruence relations of L, *i.e.*, an equivalence relation whose equivalence classes form a lattice with elements closed by the meet and join operations. This means that the concepts of L can be retrieved from the factor lattices, and the classification property of L is maintained since each equivalence relation forms a partition of the elements. The last result establishes an equivalence between arrow-closed subcontexts and compatible subcontexts, *i.e.* subcontexts such that each concept corresponds to a concept of the initial lattice. This result gives a way to compute the morphism from L into the direct product, and thus to retrieve the concepts of L from the factor lattices. In this chapter, we deduce from these results strong links between the following notions that have not been used yet together as far as we know:

- Factors of a subdirect decomposition,
- Congruence relations,
- Arrow-closed subcontexts and
- Compatible subcontexts.

As suggested in [15], the contexts of the factors of a particular subdirect decomposition, namely the irreducible subdirect subcontexts, can be obtained by a polynomial processing of each row/object of the initial context. Therefore, the subdirect decomposition of a lattice can be extended to a subdirect decomposition of its reduced context into subdirect and irreducible subcontexts.

In this chapter, we propose a subdirect and polynomial decomposition of a context into subcontexts by extending the subdirect decomposition of a lattice into factors lattices.

After studying the subdirect decomposition, we investigate a new procedure named reverse doubling construction to reduce the size of data. It is based on the

previous work of A. Day. The doubling convex procedure was first designed by Day [5, 6], then generalized [16] and widely studied [7, 19, 3]. Intuitively, this construction consists in doubling into a lattice L a convex subset C of nodes of L. In this chapter, we propose a "reverse doubling construction" which consists in removing from a lattice L a doubling convex set until no duplicated convex set exists. Hovewer, our procedure could end with the only conclusion that there is no convex to be removed. When successively applying the reverse doubling construction to a lattice L and a convex set C, and Day's doubling construction, the lattice L is recovered. Our only hypothesis is that lattices are finite. We can deduce, at least, two interesting consequences from this second decomposition:

- Like in the first case, from an "information retrieval" point of view, the search space is thus reduced and hence easier to analyse.
- From a "knowledge discovery" point of view, learning which information is doubling and redundant is important.

More generally, as for any reduction technique, we can deduce the following consequences:

- less data to store means smaller storage space.
- less data to exploit means faster computations.

Studies concerning the doubling convex construction can be organized according to the following chronological sequence of events:

- The first one corresponds to the orginal work of Day [5, 6, 7, 19], who introduced the procedure. At the very beginning, only intervals were doubled.
- Then, further generalizations were developped that lead to the present general doubling convex method [16].
- In parallel, characterisations of lattices obtained by iterating the doubling convex construction were investigated [5, 6, 7, 19, 3].
- In this chapter, we define a reverse procedure which removes a convex from a lattice whenever it is possible.

Being able to recover the full lattice from the smaller one and a convex inside, means that all the information is contained in the small lattice. Thus we only need to consider the sub-context that defines the small lattice. In other words, only a part of the data is relevant. Consequently, one need only to access or even keep a smaller part of the data. This is obviously an interesting way to manage big data.

These two decompositions, namely the sub-direct decomposition and the reverse doubling construction, lead to data storage saving of large contexts. Indeed, the generation of the whole set of factor lattices can be avoided by providing an interactive generation of a few (but not all) concepts and their neighborhood from large contexts. Moreover, a focus on a specific factor lattice can be proposed to the user by generating, partially or entirely, the concept lattice and/or a basis of implications.

There are at least two reasons for studying this case of pattern management. The first one comes from the fact that users tend to be overwhelmed by the knowledge extracted from data, even when the input is relatively small. The second reason is

that the FCA has made progress in lattice construction and exploration, and hence existing solutions can be adapted and enriched to only target useful patterns.

This chapter is organized as follows. The next section gives the background needed to introduce the two decompositions. Then, the next two sections present the two decompositions. We conclude in the last section with some perspectives.

2 Structural framework

Throughout this paper all sets (and thus lattices) are considered to be finite.

2.1 Lattices and Formal Concept Analysis

2.1.1 Algebraic lattice

Let us first recall that a *lattice* $(L, \leq)$ is an ordered set in which every pair (x, y) of elements has a *least upper bound*, called *join* $x \vee y$, and a *greatest lower bound*, called *meet* $x \wedge y$. As we are only considering finite structures, every subset $A \subset L$ has a join and meet (e. g. finite lattices are complete).

2.1.2 Concept or Galois Lattice

A (formal) *context* (O, A, R) is defined by a set O of objects, a set A of attributes, and a binary relation $R \subset O \times A$, between O and A. Two operators are derived:

- for each subset $X \subset O$, we define $X' = \{m \in A, j\ R\ m\ \forall j \in X\}$ and dually,
- for each subset $Y \subset A$, we define $Y' = \{j \in O, j\ R\ m\ \forall m \in Y\}$.

A (formal) *concept* represents a maximal objects-attributes correspondence by a pair (X, Y) such that $X' = Y$ and $Y' = X$. The sets X and Y are respectively called *extent* and *intent* of the concept. The set of concepts derived from a context is ordered as follows:

$$(X_1, Y_1) \leq (X_2, Y_2) \Longleftrightarrow X_1 \subseteq X_2 \Longleftrightarrow Y_2 \subseteq Y_1$$

The whole set of formal concepts together with this order relation form a complete lattice, called the *concept lattice* of the context (O, A, R).

Different formal contexts can provide isomorphic concept lattices, and there exists a unique one, named the *reduced context*, defined by the two sets O and A of the smallest size.

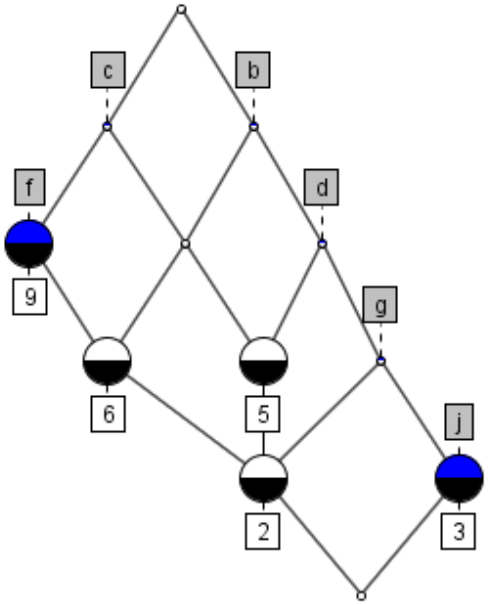
This particular context is introduced by means of special concepts or elements of the lattice L, namely irreducible elements.

An element $j \in L$ is *join-irreducible* if it is not a least upper bound of a subset not containing it. The set of join irreducible elements is noted J_L. *Meet-irreducible*

elements are defined dually and their set is M_L. As a direct consequence, an element $j \in L$ is join-irreducible if and only if it has only one immediate predecessor denoted j^-. Dually, an element $m \in L$ is meet-irreducible if and only if it has only one immediate successor denoted m^+.

In Figure 2.1.2, join-irreducible nodes are labelled with a number and meet-irreducible nodes are labelled with a letter.

Fig. 1 A lattice with its irreducible nodes



2.1.3 Fundamental Bijection

A fundamental result [1] establishes that any lattice $(L, \leq)$ is isomorphic to the concept lattice of the context $(J_L, M_L, \leq)$, where J_L and M_L are the join and meet irreducible concepts of L, respectively. Moreover, this context is a reduced one.

As a direct consequence, there is a bijection between lattices and reduced contexts where objects of the context are associated with join-irreducible concepts of the lattice, and attributes are associated with meet-irreducible concepts.

Table 2.1.3 shows the reduced context of the lattice in Figure 2.1.2.

When needed, operators $\bullet'$ introduced in subsection 2.1.2 will be written $\bullet^L$ to stress the fact that they are used with the reduced context of the lattice L.

Table 1 The reduced context of the lattice in Figure 2.1.2

	b	c	d	f	g	j
2	x	x	x	x	x	
3	x		x		x	x
5	x	x	x			
6	x	x		x		
9		x		x		

2.2 Compatible and Arrow-closed Subcontexts

This section is dedicated to the equivalence between compatible and arrow-closed subcontexts.

2.2.1 Compatible subcontexts

A *subcontext* of a formal context (O,A,R) is a triple $(J,M,R\cap J\times M)$ such that $J\subset O$ and $M\subset A$. A subcontext $(J,M,R\cap J\times M)$ of (O,A,R) is *compatible* if for each concept (H,N) of (O,A,R), $(J\cap H,M\cap N)$ is a concept of $(J,M,R\cap J\times M)$.

2.2.2 Arrow relations

The arrow-closed subcontexts involved in the equivalence are based on the arrow relations between join and meet irreducible concepts of a lattice. Consider the reduced context $(J_L,M_L,\leq)$ of a lattice $(L,\leq)$. Arrow relations [4, 17] form a partition of the relation $\not\leq$ (defined by not having $x\leq y$) by considering the immediate predecessor j^- of a join-irreducible j, and the unique immediate successor m^+ of a meet-irreducible m:

- $j\updownarrow m$ if $j\not\leq m$, $j\leq m^+$ and $j^-\leq m$.
- $j\uparrow m$ if $j\not\leq m$, $j\leq m^+$ and $j^-\not\leq m$.
- $j\downarrow m$ if $j\not\leq m$, $j\not\leq m^+$ and $j^-\leq m$.
- $j\circ m$ if $j\not\leq m$, $j\not\leq m^+$ and $j^-\not\leq m$.

In Figure 2.2.2, the reduced context of Figure 2.1.3 is enriched with the four relations $\updownarrow$, $\uparrow$, $\downarrow$, and $\circ$ in the empty cells that both correspond to the case where $j\not\leq m$:

As an illustration, let $j=5$ and $m=f$ be join-irreducible and meet-irreducible nodes respectively (see Figure 2.1.2). Then, $j^-=2$ and $m^+=c$. The relation $5\updownarrow f$ holds since $5\not\leq f$, $5\leq c$ and $2\leq f$.

Table 2 Reduced context of Table 2.1.3 filled with arrow relations

	b	c	d	f	g	j
2	×	×	×	×	×	↕
3	×	↕	×	↓	×	×
5	×	×	×	↕	↕	○
6	×	×	↕	×	↓	○
9	↕	×	○	×	○	○

2.2.3 Arrow-closed subcontext

A subcontext $(J, M, R \cap J \times M)$ of a context (O, A, R) is an *arrow-closed subcontext* when the following conditions are met:

- If $j \uparrow m$ and $j \in J$ then $m \in M$
- If $j \downarrow m$ and $m \in M$ then $j \in J$

As an example, the first subcontext of Figure 2 is an arrow-closed subcontext of the reduced context of Table 2.2.2 whereas the second one is not, due to the down-arrow $6 \downarrow g$.

Fig. 2 Arrow-closed and non-arrow-closed subcontexts of the context in Table 2.2.2

	c	d	f	g
3		x		x
5	x	x		
6	x		x	

	c	d	f	g
3		x		x
5	x	x		

2.2.4 Equivalence theorem

First let us introduce the first equivalence we need in this chapter, whose proof can be found in [15]:

Theorem 1. *Let $(J, M, R \cap J \times M)$ be a subcontext of (O, A, R). The following propositions are equivalent:*

- *The subcontext $(J, M, R \cap J \times M)$ is a compatible one.*
- *The subcontext $(J, M, R \cap J \times M)$ is an arrow-closed one.*

2.3 Congruence Relations and Factor Lattices

In this section, we introduce the equivalence between congruence relations and arrow-closed subcontexts.

2.3.1 Quotient

An *equivalence relation* is a binary relation R over a set E which is reflexive, symmetric and transitive. An equivalence class of $x \in E$ is:

$$x_R = \{y \in E \,|\, xRy\}$$

The set of equivalence classes, called the *quotient* set E/R, is:

$$E/R = \{x_R \,|\, x \in E\}$$

2.3.2 Factor lattice

A congruence relation Θ on a lattice L is an equivalence relation such that:

$$x_1 \Theta y_1 \text{ and } x_2 \Theta y_2 \Longrightarrow x_1 \wedge x_2 \Theta y_1 \wedge y_2 \text{ and } x_1 \vee x_2 \Theta y_1 \vee y_2$$

The quotient L/Θ verifies the following statement:

$$x_\Theta \leq y_\Theta \Longleftrightarrow x \Theta (x \wedge y) \Longleftrightarrow (x \vee y) \Theta y$$

With such an order, L/Θ is a lattice, called factor lattice. A standard theorem from algebra, whose proof is omitted, states that:

Theorem 2. *The projection $L \to L/\Theta$ is a lattice morphism onto.*

2.3.3 The second equivalence theorem

We are now able to formulate the second equivalence whose proof can be found in [15]:

Theorem 3. *Given a lattice L, the set of congruence relations on L corresponds bijectively with the set of arrow-closed subcontexts of the reduced context of L.*

Congruence relations will be computed with this theorem. However, other algorithms exist [11, 12].

2.4 Subdirect decompositions

In this section, we introduce the equivalence between subdirect decompositions and sets of arrow-closed subcontexts.

2.4.1 Subdirect product

Definition 1. A subdirect product is a sublattice of a direct product $L_1 \times \cdots \times L_n$ of lattices $L_i, i \in \{1,\ldots,n\}$ such that each projection onto a factor is surjective. The lattices $L_i, i \in \{1,\ldots,n\}$ are the factor lattices. A subdirect decomposition of a lattice L is an isomorphism between L and a subdirect product which can be denoted as:

$$L \hookrightarrow L_1 \times \cdots \times L_n \twoheadrightarrow L_i$$

2.4.2 The third equivalence theorem

The third and most important equivalence whose proof can be found in [15], makes a connection with sets of arrows-closed subcontexts when they cover the initial context:

Proposition 1. *Given a reduced context (O,A,R), then the subdirect decompositions of its concept lattice L correspond bijectively to the families of arrow-closed subcontexts $(J_j,M_j,R\cap J_j \times M_j)$ with $O = \cup J_j$ and $A = \cup M_j$.*

2.5 The doubling convex construction

The second decomposition, presented in this chapter is based on Alan Day's doubling convex construction [5, 6, 7, 19]. First recall that a subset $C \subset L_C$ of a lattice L_C is *convex* if it satisfies the following condition: for all $x, y \in C$ such that $x \leq y$, if $z \in L_C$ satisfies $x \leq z \leq y$ then $z \in C$. Roughly, a convex set contains all its intervals.

Now we can give the doubling convex construction of Day. Let $C \subset L_C$ be a convex set of a lattice L_C, let $L_C[C] = L_C \backslash C \cup (C \times \{0,1\})$. Define the following order on $L_C[C]$:

- $x \leq y$ if $x \leq y$ in L_C
- $(x,i) \leq y$ if $x \leq y$ in L_C
- $x \leq (y,j)$ if $x \leq y$ in L_C
- $(x,i) \leq (y,j)$ if $x \leq y$ in L_C and $i \leq j$.

With this order $L_C[C]$ is a lattice.

For instance consider the lattice L_C of Figure 2.5. A convex set C is identified with red nodes. After applying Day's construction, the lattice of Figure 2.5 is obtained. The two occurrences of the convex set are identified with green and pink nodes.

Our construction is based on the two following theorems proved by W. Geyer [16].

Fig. 3 A lattice L_C with nodes of a convex set C in red.

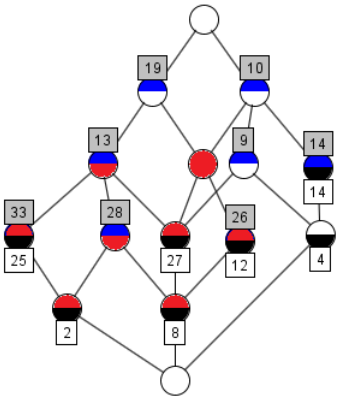
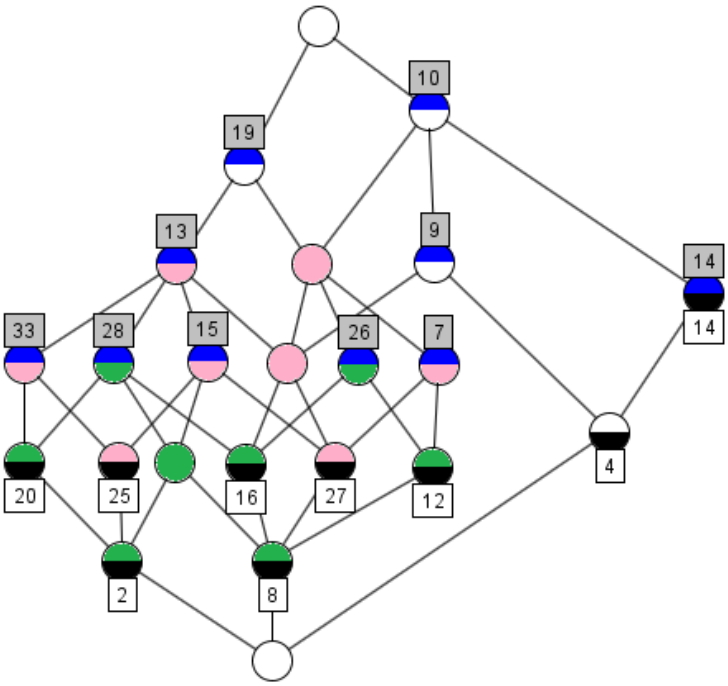


Fig. 4 The lattice of Figure 2.5 with the convex set doubled. Nodes of each convex are coloured in green and pink.



Theorem 4. *Let $L = L_C[C]$ be a lattice obtained by doubling the convex set C of the lattice L_C. Note $K = (O,A,R)$ the reduced context of L, then the reduced context $K_C = (J,M,R \cap J \times M)$ is an arrow-closed sub-context of K such that:*

$$(*)\, R \cap ((O \backslash J) \times (A \backslash M)) = \emptyset$$

Later on, the previous condition $(*)$ will be referred as the Geyer's Condition. Now we are searching for an arrow-closed sub-context that satisfies the Geyer's Condition.

Using theorem 3, it is equivalent to search for an arrow-closed sub-context or a congruence relation. Now recall that it is possible to give a lattice structure to the set of congruence relations in the following manner. First, the binary relations can be ordered by the inclusion. Then the set of congruence relations can be ordered that way. Moreover, the intersection of two congrence relations is still a congruence relation; thus we get the meet operation. As usual, the join operation is not so easy. However, given a set of congruence relations, it is sufficient to consider the smallest congruence relations containing all of them.

With this lattice structure on the congruence relations, we can state the following result of Day [6]:

Theorem 5. *Let L be a congruence normal lattice and Θ a congruence relation which is an atom, i.e. a successor of the bottom element. Then, there exists a convex set C such that L is isomorphic to $L/\Theta[C]$.*

Notice that congruence normality has not been defined because it will not be used in this chapter. Notice also that Day is more explicit about the convex C, but we will give a new construction of it.

3 Subdirect decomposition into subdirectly irreducible factors

3.1 Main Result

From the three previous equivalences found in [15], we deduce the following one:

Corollary 1. *Given a lattice L and its reduced context (O,A,R), we have an equivalence between:*

1. *The set of arrow-closed subcontexts of (O,A,R),*
2. *The set of compatible subcontexts of (O,A,R),*
3. *The set of congruence relations of L and their factor lattices.*

Corollary 2. *Given a lattice L and its reduced context (O,A,R), we have an equivalence between:*

1. *The families of arrow-closed subcontexts of (O,A,R) covering O and A,*

2. *The families of compatible subcontexts of* (O,A,R) *covering* O *and* A,
3. *The families* $(\theta_i)_{i\in I}$ *of congruence relations of* L *such that* $\cap_{i\in I}\theta_i = \Delta$ *with* $x\Delta y \Longleftrightarrow x = y$.
4. *The set of subdirect decompositions of* L *and their factor lattices.*

In the following, we exploit these four notions that, to the best of our knowledge, have not been put together yet.

1. The subdirect decomposition ensures that L is a sublattice of the factor lattice product. Moreover, each projection from L to a factor lattice is surjective.
2. The congruence relations of L indicate that factor lattices correspond to their quotient lattices, and thus preserve partitions via equivalence classes.
3. The compatible subcontexts give a way to compute the morphism from L onto its factors.
4. Arrow-closed subcontexts enable the computation of the reduced context of the factor lattices.

In the following we present the generation of a particular subdirect decomposition and show a possible usage of factor lattices.

3.2 Generation of Subdirectly Irreducible Factors

In this section, we consider subdirect decompositions of a lattice L with its reduced context (O,A,R) as input. From Corollary 2, a subdirect decomposition of a lattice L can be obtained by computing a set of arrow-closed subcontexts of (O,A,R) that have to cover O and A. There are many sets of arrow-closed subcontexts and thus many subdirect decompositions. In particular, the decomposition from a lattice L into L itself is a subdirect decomposition, corresponding to the whole subcontext (O,A,R) which is clearly arrow-closed. A subdirect decomposition algorithm has been proposed in [14]. However, all congruence relations are computed and then only pairs of relations are formed to get a decomposition. As a consequence, potentially multiple decompositions are produced with necessarily two factors.

In this chpater, we focus on the subdirect decomposition of a context into a possibly large number of small factors, *i.e.* factors that cannot be subdirectly decomposed. A factor lattice L is *subdirectly irreducible* when any subdirect decomposition of L contains L itself as a factor. A nice characterization of subdirectly irreducible lattices can be found in [15]:

Proposition 2. *A lattice* L *is subdirectly irreducible if and only if its reduced context is one-generated.*

A reduced context (O,A,R) is one-generated if it can be obtained by arrow-closing a context with only one $j \in J$. Thus (O,A,R) is the smallest arrow-closed subcontext containing $j \in J$.

Therefore, we deduce the following result:

Proposition 3. *Let L be a lattice. From L, we can deduce a product lattice* $L_1 \times ... \times L_n$ *where each lattice* L_i *is:*

- *the concept lattice of a one-generated subcontext,*
- *subdirectly irreducible,*
- *a factor lattice of the subdirectly irreducible decomposition.*

From this result, we propose an algorithm (Algorithm 1) to compute in polynomial time the contexts of the factor lattices $L_1, \ldots, ...L_n$ of a subdirectly irreducible decomposition, with a reduced context (O,A,R) as input. The one-generated subcontexts for each $j \in J$ are obtained by arrow-closing (Algorithm 2). The subdirectly irreducible decomposition of L can then be obtained by computing the concept lattices of these subcontexts.

One can notice that the closure computed from join-irreducible concepts can also be calculated from meet-irreducible concepts.

Algorithm 1: Subdirect_Decomposition

Input: A context (O,A,R)
Output: List $\mathscr{L}$ of the contexts (J_j,M_j,R_j) of the subdirectly irreducible factor lattices

$\mathscr{L} \leftarrow \emptyset$;
forall the $j \in O$ **do**
  Compute $(J_j,M_j,R_j) =$ **Arrow_Closure**$((j,\emptyset,\emptyset),(O,A,R))$, the one-generated subcontext span by j;
  if $\mathscr{L}$ *does not contain any subcontext that covers* (J_j,M_j,R_j) **then**
    add (J_j,M_j,R_j) to $\mathscr{L}$
  if $\mathscr{L}$ *contains a subcontext covered by* (J_j,M_j,R_j) **then**
    delete it from $\mathscr{L}$
  return $\mathscr{L}$;

Algorithm 2: Arrow_Closure

Input: A subcontext $(\tilde{J},\tilde{M},\tilde{R})$ of a context (J,M,R)
Output: Arrow-closure of $(\tilde{J},\tilde{M},\tilde{R})$

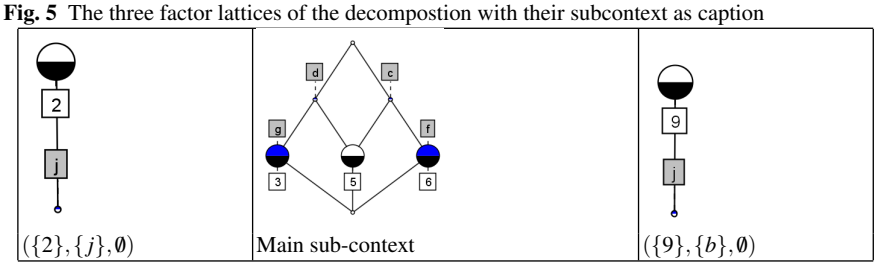
$J_c = \tilde{J}$; $M_c = \tilde{M}$;
$pred_J = 0$; $pred_M = 0$;
while $pred_M < card(M_c)$ *or* $pred_J < card(J_c)$ **do**
  $pred_J = \text{card}(J_c)$;
  $pred_M = \text{card}(M_c)$;
  forall the $j \in J_c$ **do**
    add to M_c all $m \in M$ such that $j \uparrow m$;
  forall the $m \in M_c$ **do**
    add to J_c all $j \in J$ such that $j \downarrow m$;
  Return $(J_c,M_c,R \cap J_c \times M_c)$

Table 3 Iterations of Algorithm 1 for the reduced context in Table 2.1.3

j	Arrow_Closure			Contained in $\mathscr{L}$
	Input	Output		
	$(\tilde{J},\tilde{M},\tilde{R})$	J_c	M_c	
2	$(2,\emptyset,\emptyset)$	$\{2\}$	$\{j\}$	$\times$
3	$(3,\emptyset,\emptyset)$	$\{3\}$	$\{c\}$	
5	$(5,\emptyset,\emptyset)$	$\{3,5,6\}$	$\{c,d,f,g\}$	$\times$
6	$(6,\emptyset,\emptyset)$	$\{6\}$	$\{d\}$	
9	$(9,\emptyset,\emptyset)$	$\{9\}$	$\{b\}$	$\times$

Consider the reduced context in Figure 2.1.3. Each iteration of Algorithm 1 is described by Figure 3 for each value of j, the input and output of Algorithm 2, and the three one-generated subcontexts that belong to $\mathscr{L}$ at the end of the process. Therefore we get three factor lattices (see Figure 5).

Fig. 5 The three factor lattices of the decompostion with their subcontext as caption



$(\{2\},\{j\},\emptyset)$ Main sub-context $(\{9\},\{b\},\emptyset)$

The first subcontext is the one on the left of Figure 2. The two other ones are: $(\{2\},\{j\},\emptyset)$ and $(\{9\},\{b\},\emptyset)$. The latter two subcontexts are interesting because:

- They show that the initial lattice has parts which are distributive. Indeed, these two subcontexts contain exactly one double arrow in each line and each column.
- They give us a dichotomy: any concept contains either 2 or j ; any concept contains either 9 or b
- In the reduced context, an arrow brings a deeper knowledge than a cross.

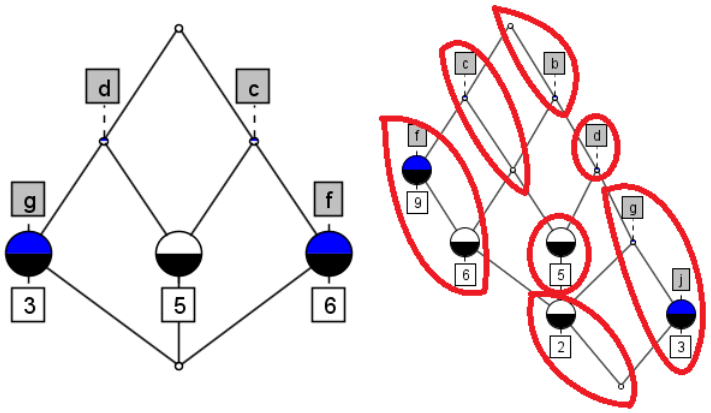
The context on the middle of Figure 2 is tricky to understand. For the other ones, we have a simple relation $2 \updownarrow j$ or $9 \updownarrow b$, which means that, for instance, 2 and j are some kind of complement or converse.

Figure 10 shows a factor lattice and its corresponding congruence.

3.3 Onto Morphism and FCA

A subdirect decomposition of a lattice L into factor lattices $L_1,\ldots,L_n$ is relevant since there exists an into morphism from L to the product lattice $L_1 \times \ldots \times L_n$. This

Fig. 6 Factor lattice and congruence



morphism is specified by the bijection between compatible subcontexts and congruence relations stated by Corollary 1:

Proposition 4. *Let* $(J,M,R\cap J\times M)$ *be a compatible subcontext, then the relation* $\Theta_{J,M}$ *defined by:*

$$(A_1,B_1)\Theta_{J,M}(A_2,B_2)\Longleftrightarrow A_1\cap J=A_2\cap J\Longleftrightarrow B_1\cap M=B_2\cap M$$

is a congruence relation, and its factor lattice is isomorphic to the concept lattice of the subcontext $(J,M,R\cap J\times M)$.

Algorithm 3 computes this morphism: each concept of L is computed as the product of concepts in each factor, and then marked in the product lattice. From Algorithm 3, we get the large tagged lattice product shown in Figure 4. Obviously, this algorithm is not intended to be used in a real application with large contexts since the product lattice is much more bigger than the original one, while the main goal of the decomposition is to get smaller lattices. We only use this algorithm in the empirical study.

Nevertheless, this morphism can be extended to develop basic FCA processing. Once the subdirectly irreducible decomposition of a reduced context (O,A,R) into the contexts $C_1,\ldots,C_n$ is computed, an interactive exploration and mining process can easily be considered by using the following basic tasks and avoiding the generation of the lattice for the whole context (O,A,R):

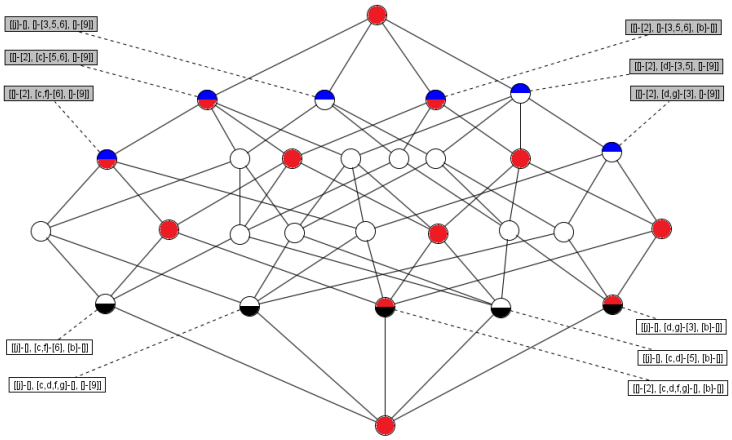
- Compute the smallest concept of L that contains a given subset of objects or attributes, and identify its neighborhood
- Compute the smallest concept c_{ij} and its neighborhood in a subset of factors that contain a given collection of objects or attributes. Each factor L_i is a specific view of data.

Algorithm 3: Into_morphism

Input: Initial lattice L;
Subcontexts (J_j, M_j, R_j);
Product lattice $P = L_1 \times \ldots \times L_n$
Output: Product lattice P with nodes coming from L marked.

```
1 forall the c = (A, B) ∈ L do
2     forall the (J_j, M_j, R_j) do
3         Compute (A ∩ J_j, B ∩ M_j);
4     Mark, in P, the product node Π_j(A ∩ J_j, B ∩ M_j);
```

Fig. 7 Lattice product in which nodes of the initial lattice appear in red



4 Reverse doubling convex

In this section, we describe the reverse doubling construction which itself uses congruence relations.

Given a lattice L, we recall that we are looking for a lattice L_C containing a convex C such that $L = L_C[C]$.

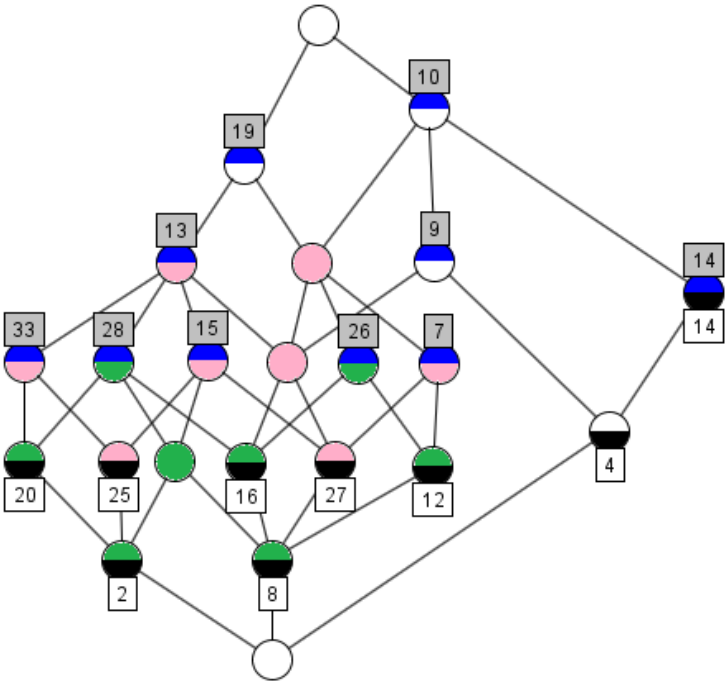
4.1 Main steps

Before giving into details about this decomposition, let us give the main steps of our construction. Given a finite lattice L, we are searching a lattice L_C and a convex set $C \subset L_C$ such that L is isomorphic to $L_C[C]$. From Theorem 5, we know that good candidates can be obtained from factor lattices, spanned by a congruence relation which is an atom. It is important to note that we only get good candidates

because we do not use the congruence normality hypothesis. Hovewer, the first step of our construction is to compute atoms of the lattice of congruence relations. From Theorem 3, we know that these congruence relations correspond to arrow-closed sub-contexts. Then from Theorem 4 of Geyer, we only need to check if these arrow-closed sub-contexts, that are just good candidates, satisfy the Geyer's condition. If none of these sub-contexts is valid, the decomposition is not possible. Otherwise, each one of these sub-contexts generates a lattice L_C that is appropriate. The last step consists in identifying the convex set C in L_C.

To illustrate the reverse doubling convex construction, we will use the lattice given in Figure 4.1 and its reduced Table 4.1, already filled with arrows.

Fig. 8 The lattice used to illustrate the reverse doubling convex decomposition. Nodes in green and pink correspond to the two occurrences of the doubled convex.



4.2 The lattice of Congruence Relations

A reduced context (O,A,R) of a lattice L is supposed to be given.

Table 4 The reduced context of the lattice given in Figure 4.1

	2	4	8	12	14	25	20	27	16
9	↑	×	×	↕	↕	○	○	×	×
10	↕	×	×	×	×	↓	↓	×	×
14	↑	×	↕	↓	×	○	○	↓	↓
19	×	↕	×	×	↓	×	×	×	×
33	×	↑	↕	○	○	×	×	↓	↓
15	×	↑	×	↑	○	×	↕	×	↕
28	×	↑	×	↑	○	↕	×	↕	×
7	↑	↑	×	×	○	○	○	×	↕
26	↑	↑	×	×	○	○	○	↕	×
13	×	↑	×	↕	○	×	×	×	×

In Subsection 2.3, we introduced congruence relations. As a particular kind of binary relation, the set of congruence relations inherits a structure of lattice: it is a sublattice of the lattice of binary relations which is ordered by inclusion. Using Theorem 3, this lattice is also the lattice of arrow-closed sub-contexts. As explained previously, we aim at finding the atoms of that lattice. To that end, we introduce a new context, namely the context of arrow-closed subcontexts, which is defined and computed as follows. We first compute one-generated arrow-closed sub-contexts.

Definition 2. A context (J,M,R) is one-generated if it can be obtained by arrow-closing a context with only one $j \in J$. Thus (J,M,R) is the smallest arrow-closed subcontext containing $j \in J$.

The set of such one-generated sub-contexts contains all join-irreducible arrow-closed sub-contexts of the lattice of arrow-closed sub-contexts. Thus, the set of one-generated sub-contexts generates the lattice of arrow-closed sub-contexts. The one-generated sub-contexts are stored as observations of a new context in the following way: an arrow-closed is characterized by its set of attributes, since it is closed. Thus the newly generated context has the same set of attributes, namely A. Since, one-generated sub-contexts are computed by closing one $j \in O$, they can be identified with that j. So the set of observations of the newly generated context is O. At last, there is a cross in the new context between j and a if and only if the arrow-closed sub-context generated by j contains the attributes a.

From this context, we can deduce the lattice of arrow-closed sub-contexts and in particular its atoms.

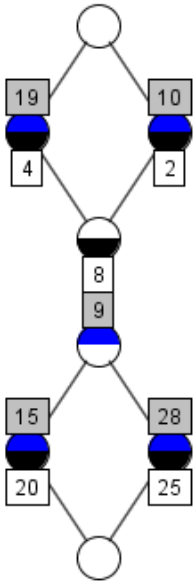
From the lattice of Figure 4.1, we compute the Table 4.2 of one-generated arrow-closed sub-contexts and then we get the lattice of congruence relations given in Figure 4.2. Notice that Table 4.2 has been reduced after computation. In this particular case, there are two atoms.

Using only one of these atoms, namely the one containing observation 25, we get the arrow-closed subcontext of Table 4.4, which satisfies Geyer's condition. Then, its concept lattice, which is also the factor lattice of the selected congruence relation, can be compute, and visualize on Figure 4.4.

Table 5 The *reduced* context computed to get the lattice of congruence relations (Figure 4.2)

	9	10	19	15	28
2		×			
4			×		
8		×	×		
25	×	×	×		×
20	×	×	×	×	

Fig. 9 The lattice of congruence relations of the lattice given in Figure 4.1



4.3 The factor lattice

We are now given a list of arrow-closed sub-contexts, which correspond to atom congruence relations.

The second step is to remove from this list the following contexts:

- The empty context. In the particular case where the initial lattice has only two congruence relations, there is only one atom, namely the bottom element. This case is excluded since it gives rise to a trivial decomposition.
- Any context that does not satisfy Geyer's Condition.

After this removal process, our list could be empty. Since the Geyer's condition is necessary and sufficient, we can conclude that the original lattice can not be processed through the reverse doubling construction.

On the opposite side, any remaining context generates a lattice L_C such that there exists a convex set C so that the initial lattice L satisfies $L = L_C[C]$.

The two atoms identified in Figure 4.2 satisfy Geyer's condition. So the lattice given in Figure 4.1 has two decompositions.

4.4 Finding the convex

Given a lattice L_C satisfying the previous conditions of subsection 4.3, the last step is to identify in L_C nodes of a convex set C such that $L = L_C[C]$.

First remark that using theorem 4 which gives a necessary and sufficient condition, we already know that L is obtained by the doubling construction from L_C. Thus, each concept of L_C gives rise to one concept in L if and only if it is not in C and two concepts in L if and only if it is in C.

Let us be more precise about these two concepts, and first let us recall some notations: (O,A,R) is the reduced context of the big lattice L. We consider $(J,M,R\cap J\times M)$ a atom in the lattice of arrow-closed sub-contexts whi ch satifies the Geyer's condition. From this context, we can deduce L_C the lattice from which a convex has been removed. Let c be a concept of L_C. Since $(J,M,R\cap J\times M)$ is a compatible sub-context, using Theorem 1, c can be written as $c=(H\cap J,N\cap M)$ with (H,N) a concept of L. Since c is a concept, we have $(H\cap J)'=N\cap M$ and $(N\cap M)'=H\cap J$. Recall, from the end of Subsection 2.1.3, that these operations are written $\bullet^L$ in the reduced context (O,A,R) of L; thus we also have $H^L=N$ and $N^L=H$. Now, we can conclude that, if $c=(H\cap J,N\cap M)$ is not in the convex, it comes from the unique concept of L, namely (H,N) and thus we must have $((H\cap J)^{LL},(H\cap J)^L)=(H,N)=((N\cap M)^L,(N\cap M)^{LL})$. In the other case, we have $((H\cap J)^{LL},(H\cap J)^L)\neq((N\cap M)^L,(N\cap M)^{LL})$.

From this point, we can state a simpler condition for a concept to be in the convex set:

Proposition 5. *A concept $c=(H\cap J,N\cap M)$ is in the convex if and only if*

$$(H\cap J)^L\subseteq(N\cap M) \text{ or } (N\cap M)^L\subseteq(H\cap J)$$

Proof. We have previously seen that if c is in the convex then $((H\cap J)^{LL},(H\cap J)^L)=(H,N)=((N\cap M)^L,(N\cap M)^{LL})$. So both inclusions are true and actually are equalities.

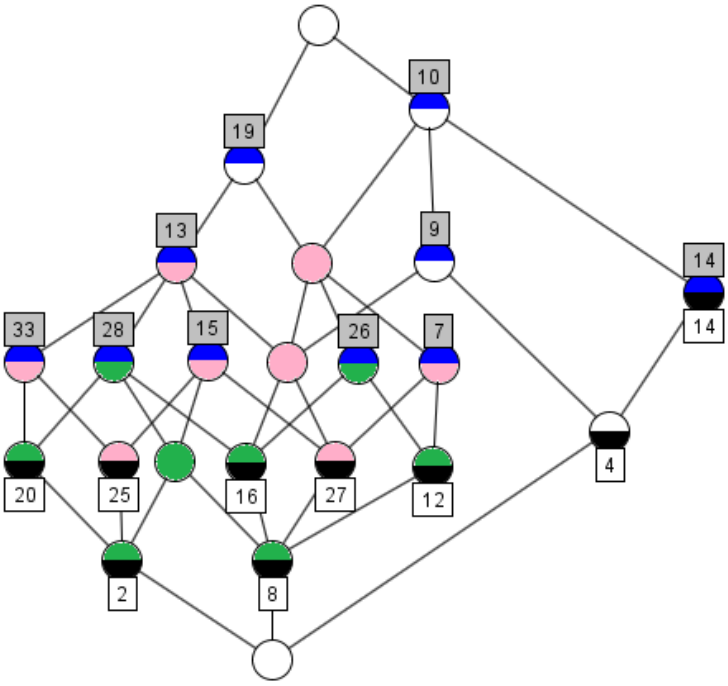
Suppose, reciprocally, that we have the first one $(H\cap J)^L\subset(N\cap M)$. We obtain the same result dually in the second case. First we prove that the inclusion is an equality. Indeed, since $(J,M,R\cap J\times M)$ is a subcontext of (O,A,R), we have $(H\cap J)'\subset(H\cap J)^L$. However $(H\cap J)'=(N\cap M)$, so $(N\cap M)\subset(H\cap J)^L$ and dually $(H\cap J)\subset(N\cap M)^L$. Therefore, we deduce that $(H\cap J)^L=(N\cap M)$, and then $(N\cap M)^L=(H\cap J)^{LL}$. Moreover, $(H\cap J)\subset(N\cap M)^L\Longrightarrow(N\cap M)^{LL}\subset(H\cap J)^L$, and with $(N\cap M)=(H\cap J)^L$, we deduce that $(N\cap M)=(N\cap M)^{LL}$, since $(N\cap M)\subset(N\cap M)^{LL}$. Now from $(H\cap J)^L=(N\cap M)=(N\cap M)^{LL}$ and $(N\cap M)^L=(H\cap J)^{LL}$, we get $((H\cap J)^L,(H\cap J)^{LL})=((N\cap M)^{LL},(N\cap M)^L)$. This means that a concept $((H\cap J),(N\cap M))$ in the small lattice L_C that satisfies the condition $(H\cap J)^L\subset$

$(N \cap M)$ or $(N \cap M)^L \subset (H \cap J)$ comes from a unique concept in the lattice L, namely the concept $((H \cap J)^{LL}, (H \cap J)^L) = ((N \cap M)^L, (N \cap M)^{LL}) = (H, N)$.

Conversely, if $((H \cap J), (N \cap M))$ does not satisfy the previous condition, it comes from two concepts $((H \cap J)^{LL}, (H \cap J)^L) \neq ((N \cap M)^L, (N \cap M)^{LL})$ in L. Thus to get L from L_C, one needs to double these concepts and then these concepts are exactly the ones of the convex.

In Figure 4.4, nodes of the convex C are in red, and when the doubling construction is applied on that lattice, with this convex set, the lattice from Figure 4.4 is obtained.

Fig. 10 The initial lattice. Nodes in green and pink correspond to the two occurrences of the doubled convex.



This lattice is obtained from the reduced context given in the table 4.4, which is one of the two atomes of the lattice 4.2.

All computations were done using the Galactic project available at :

http://thegalactic.github.io/

Lattices were drawn with ConExp available at :

http://conexp.sourceforge.net/

Fig. 11 The factor lattice obtained by the reverse doubling decomposition.

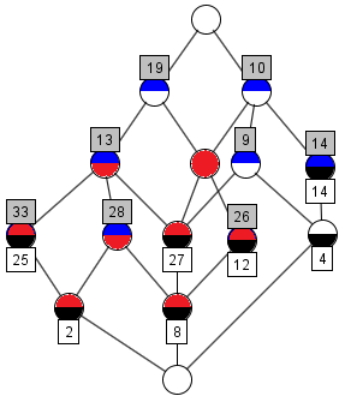


Table 6 The reduced context of the factor lattice given in Figure 4.4

	10	13	14	19	26	28	33	9
12	×			×	×			
14	×		×					
2		×		×		×	×	
25		×		×			×	
27	×	×		×				×
4	×		×					×
8	×			×	×	×		×

5 Conclusion and future work

In this chapter, we have presented two decompositions of lattices based on congruence relations. In the fisrt section, different points of view and constructions about these relations were given. Then we presented a particular case of the widely studied subdirect decomposition. To further investigate the subdirect decomposition, it would be interesting to conduct large-scale experiments on real world data to understand the semantics behind the generated irreducible contexts. In particular, attributes covered by several factors interfere in different views of data whose semantic must be interesting to understand. Moreover families of arrow-closed subcontext covering the initial context could be used as universal index. It would be useful to allow the user to interactively select a few factors of the decomposition by mixing o ur approach with the one in [14]. Since the empirical study in [20] show that many real-life contexts are subdirectly irreducible, we plan to identify cases in which a context is necessarily irreducible.

In the previous section, a new decomposition, based on Day's construction, was given which uses again congruence relations. Day's doubling construction has been widely studied [6, 19, 16, 3], but to our knowledge, this decomposition has never been done.

From a theoretical point of view, we think that there are strong links between the implication basis in a quotient lattice and the one from the initial lattice. To the best of our knowledge, this issue has never been addressed and could have significant algorithmic impacts. However, we note that [21] tackle a similar issue in case of a vertical decomposition of contexts into subcontexts.

To go further, we plan to study, compare and combine other decompositions, in particular the Fratini congruence [9] which exploits again a congruence relation and atlas decomposition [15] which uses different tools.

References

1. M. Barbut and B. Monjardet, editors. *L'ordre et la classification*. Algèbre et combinatoire, tome II. Hachette, 1970.
2. Radim Belohlavek and Vilem Vychodil. Discovery of optimal factors in binary data via a novel method of matrix decomposition. *Journal of Computer and System Sciences*, 76(1):3–20, February 2010.
3. K. Bertet and N. Caspard. Doubling convec sets in lattices: characterizations and recognition algorithms. Technical Report TR-LACL-2002-08, LACL (Laboratory of Algorithms, Complexity and Logic), University of Paris-Est (Paris 12), 2002.
4. P. Crawley and R.P. Dilworth. *Algebraic theory of lattices*. Prentice Hall, Englewood Cliffs, 1973.
5. Alan Day. Splitting lattices generate all lattices. *algebra universalis*, 7(1):163–169, 1977.
6. Alan Day. Congruence normality: The characterization of the doubling class of convex sets. *algebra universalis*, 31(3):397–406, 1994.
7. Alan Day, J.B. Nation, and Steve Tschantz. Doubling convex sets in lattices and a generalized semidistributivity condition. *Order*, 6(2):175–180, 1989.
8. Jiří Demel. Fast algorithms for finding a subdirect decomposition and interesting congruences of finite algebras. *Kybernetika (Prague)*, 18(2):121–130, 1982.
9. Vincent Duquenne. Lattice drawings and morphisms. In *Formal Concept Analysis, 8th International Conference, ICFCA 2010, Agadir, Morocco, March 15-18, 2010. Proceedings*, pages 88–103, 2010.
10. S. Ferré. *Reconciling Expressivity and Usability in Information Access from File Systems to the Semantic Web*. PhD thesis, Univeristy Rennes 1, 2014.
11. Ralph Freese. Computing congruence lattices of finite lattices. *Proceedings of the American Mathematical Society*, 125(12):3457–3463, 1997.
12. Ralph Freese. Algorithms in finite, finitely presented and free lattices. *Preprint, July*, 22:159–178, 1999.
13. Ralph Freese. Computing congruences efficiently. *Algebra universalis*, 59(3-4):337–343, 2008.
14. P. Funk, A. Lewien, and G. Snelting. Algorithms for concept lattice decomposition and their applications. Technical report, TU Braunschweig, December 1995.
15. Bernhard Ganter and Rudolf Wille. *Formal concept analysis - mathematical foundations.* Springer, 1999.
16. W. Geyer. The generalized doubling construction and formal concept analysis. *algebra universalis*, 32(3):341–367, 1994.
17. George Grätzer. *General lattice theory*. Birkhäuser-Verlag, Basel, 1978.
18. Peter Mihók and Gabriel Semanišin. Unique factorization theorem and formal concept analysis. In SadokBen Yahia, EngelbertMephu Nguifo, and Radim Belohlavek, editors, *Concept Lattices and Their Applications*, volume 4923 of *Lecture Notes in Computer Science*, pages 232–239. Springer Berlin Heidelberg, 2008.

19. J.B. Nation. Alan day's doubling construction. *algebra universalis*, 34(1):24–34, 1995.
20. Gregor Snelting. Concept lattices in software analysis. In *Formal Concept Analysis, Foundations and Applications*, pages 272–287, 2005.
21. Petko Valtchev and Vincent Duquenne. On the merge of factor canonical bases. In *International Conference on Formal Concept Analysis ICFCA*, pages 182–198. Springer Berlin Heidelberg, 2008.
22. M. Visani, K. Bertet, and J-M. Ogier. Navigala: an Original Symbol Classifier Based on Navigation through a Galois Lattice. *International Journal on Pattern Recognition and Artificial Intelligence (IJPRAI)*, 2011.
23. Rudolf Wille. Subdirekte produkte und konjunkte summen. *Journal fr die reine und angewandte Mathematik*, 0239_0240:333–338, 1969.
24. Rudolf Wille. Subdirekte Produkte vollständiger Verbände. *J. reine angew. Math.*, 283/284:53–70, 1976.
25. Rudolf Wille. Subdirect decomposition of concept lattices. *Algebra Universalis*, 17:275–287, 1983.
26. Rudolf Wille. Subdirect product construction of concept lattices. *Discrete Mathematics*, 63(2-3):305–313, 1987.