Removing an incidence from a formal context

Martin Kauer* and Michal Krupka**

Department of Computer Science
Palacký University in Olomouc
17. listopadu 12, CZ-77146 Olomouc
Czech Republic
martin.kauer@upol.cz
michal.krupka@upol.cz

Abstract. We analyze changes in the structure of a concept lattice corresponding to a context resulting from a given context with a known concept lattice by removing exactly one incidence. We identify the set of concepts affected by the removal and show how they can be used for computing concepts in the new concept lattice. We present algorithms for incremental computation of the new concept lattice, with or without structural information.

1 Introduction

When computing concept lattices of two very similar concepts (i.e., differing only in a small number of incidences), it doesn't seem to be efficient to compute both concept lattices independently. Rather, an incremental method of computing one of the lattices using the other would be more desirable. Also, analyzing structural differences between concept lattices of two similar contexts would be interesting from the theoretical point of view.

This paper presents first results in this direction. Namely, we consider two formal contexts differing in just one incidence and develop a method of computing the concept lattice of the context without the incidence from the other one. In other words, we give a first answer to the question "What happens to the concept lattice, if we remove one cross from the context?".

Our results are the following. We consider contexts $\langle X, Y, I\rangle$ and $\langle X, Y, J\rangle$ such that J results from I by removing exactly one incidence. Further we consider the respective concept lattices $\mathcal{B}(I)$ and $\mathcal{B}(J)$. For these contexts and concept lattices we

1. identify concepts in $\mathcal{B}(I)$, affected by the removal (they form an interval in $\mathcal{B}(I)$),

* The author acknowledges support by IGA of Palacky University, No. PrF 2014 034
** The author acknowledges support by the ESF project No. CZ.1.07/2.3.00/20.0059. The project is co-financed by the European Social Fund and the state budget of the Czech Republic.

2. show how they transform to concepts in the new concept lattice (they will either vanish entirely, or transform to one or two concepts),
3. derive several further results on the correspondence between the two lattices,
4. propose two basic algorithms for transforming incrementally $\mathcal{B}(I)$ to $\mathcal{B}(J)$.

Several algorithms for incremental computation of concept lattices have been developed in the past [1, 5, 8, 6, 7, 2] (see also [4] for a comparison of some of the algorithms). In general, the algorithms build a concept lattice incrementally by modifying formal contexts by adding or removing objects one by one. Our approach is different as we focus on removing just one incidence.

2 Formal concept analysis

Formal Concept Analysis has been introduced in [9], our basic reference is [3]. A *(formal) context* is a triple $C = \langle X, Y, I\rangle$ where X is a set of objects, Y a set of attributes and $I \subseteq X \times Y$ a binary relation between X and Y. For $\langle x, y\rangle \in I$ it is said "The object x has the attribute y".

For subsets $A \subseteq X$ and $B \subseteq Y$ we set

$$A^{\uparrow_I} = \{y \in Y \mid \text{for each } x \in A \text{ it holds } \langle x, y\rangle \in I\},$$
$$B^{\downarrow_I} = \{x \in X \mid \text{for each } y \in B \text{ it holds } \langle x, y\rangle \in I\}.$$

The pair $\langle {}^{\uparrow_I}, {}^{\downarrow_I}\rangle$ is a Galois connection between sets X and Y, i.e., it satisfies for each $A, A_1, A_2 \subseteq X$, $B, B_1, B_2 \subseteq Y$,

1. If $A_1 \subseteq A_2$, then $A_2^{\uparrow_I} \subseteq A_1^{\uparrow_I}$, if $B_1 \subseteq B_2$, then $B_2^{\downarrow_I} \subseteq B_1^{\downarrow_I}$.
2. $A \subseteq A^{\uparrow_I\downarrow_I}$ and $B \subseteq B^{\downarrow_I\uparrow_I}$.

If $A^{\uparrow_I} = B$ and $B^{\downarrow_I} = A$, then the pair $\langle A, B\rangle$ is called a *formal concept* of $\langle X, Y, I\rangle$. The set A is called the *extent of* $\langle A, B\rangle$, the set B the *intent of* $\langle A, B\rangle$.

A partial order $\leq$ on the set $\mathcal{B}(X, Y, I)$ of all formal concepts of $\langle X, Y, I\rangle$ is defined by $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$ iff $A_1 \subseteq A_2$ (iff $B_2 \subseteq B_1$). $\mathcal{B}(X, Y, I)$ along with $\leq$ is a complete lattice and is called the *concept lattice* of $\langle X, Y, I\rangle$. Infima and suprema in $\mathcal{B}(X, Y, I)$ are given by

$$\bigwedge_{j\in J} \langle A_j, B_j\rangle = \left\langle \bigcap_{j\in J} A_j, \left(\bigcup_{j\in J} B_j\right)^{\downarrow_I\uparrow_I} \right\rangle, \tag{1}$$

$$\bigvee_{j\in J} \langle A_j, B_j\rangle = \left\langle \left(\bigcup_{j\in J} A_j\right)^{\uparrow_I\downarrow_I}, \bigcap_{j\in J} B_j \right\rangle. \tag{2}$$

One of immediate consequences of (1) and (2) is that the intersection of any system of extents (resp. intents) is again an extent (resp. intent).

Mappings $\gamma_I : x \mapsto \langle \{x\}^{\uparrow_I\downarrow_I}, \{x\}^{\uparrow_I}\rangle$ and $\mu_I : y \mapsto \langle \{y\}^{\downarrow_I}, \{y\}^{\downarrow_I\uparrow_I}\rangle$ assign to each object x its *object concept* and to each attribute y its *attribute concept*. We call a subset $K \subseteq L$, where L is a complete lattice, $\bigvee$-dense (resp. $\bigwedge$-dense) if and only if any element of L can be expressed by suprema (resp. infima) of some elements from K. The set of all object concepts (resp. attribute concepts) is $\bigvee$-dense (resp. $\bigwedge$-dense) in $\mathcal{B}(X,Y,I)$. This can be easily seen from (1) (resp. (2)).

We will also need a notion of an interval in lattice L. We call a subset $K \subseteq L$ an *interval*, if and only if there exist elements $a, b \in L$ such that $K = \{k \in L \mid a \leq k \leq b\}$. We denote K as $[a, b]$.

3 Problem statement and basic notions

Let $\langle X, Y, I\rangle$, $\langle X, Y, J\rangle$ be two contexts over the same sets of objects and attributes such that $\langle x_0, y_0\rangle \notin J$ and $I = J \cup \{\langle x_0, y_0\rangle\}$.

We usually denote concepts of $\langle X, Y, I\rangle$ by c, c_1, $\langle A, B\rangle$, $\langle A_1, B_1\rangle$, etc., and concepts of $\langle X, Y, J\rangle$ by d, d_1, $\langle C, D\rangle$, $\langle C_1, D_1\rangle$, etc. The respective concept lattices will be denoted $\mathcal{B}(I)$ and $\mathcal{B}(J)$.

Our goal is to find an efficient way to compute the concept lattice $\mathcal{B}(J)$ from $\mathcal{B}(I)$. We provide two solutions to this problem. First solution computes just elements of $\mathcal{B}(J)$, the second one adds also information on its structure. In this section we introduce some basic tools and prove simple preliminary results.

The following proposition shows a correspondence between the derivation operators of contexts $\langle X, Y, I\rangle$ and $\langle X, Y, J\rangle$.

Proposition 1. *For each $A \subseteq X$ and $B \subseteq Y$ it holds*

$$A^{\uparrow_J} = \begin{cases} A^{\uparrow_I} & \text{if } x_0 \notin A, \\ A^{\uparrow_I} \setminus \{y_0\} & \text{if } x_0 \in A, \end{cases} \qquad B^{\downarrow_J} = \begin{cases} B^{\downarrow_I} & \text{if } y_0 \notin B, \\ B^{\downarrow_I} \setminus \{x_0\} & \text{if } y_0 \in B. \end{cases}$$

In particular, $A^{\uparrow_J} \subseteq A^{\uparrow_I}$ and $B^{\downarrow_J} \subseteq B^{\downarrow_I}$.

Proof. Immediate. □

Formal concepts from the intersection $\mathcal{B}(I) \cap \mathcal{B}(J)$ are called *stable*. These concepts are not influenced by removing the incidence $\langle x_0, y_0\rangle$ from I. When computing $\mathcal{B}(J)$ from $\mathcal{B}(I)$, stable concepts need not be recomputed.

Proposition 2. *A concept $c \in \mathcal{B}(I)$ is not stable iff $c \in [\gamma_I(x_0), \mu_I(y_0)]$.*

Proof. If $c = \langle A, B\rangle \notin [\gamma_I(x_0), \mu_I(y_0)]$, then either $x_0 \notin A$, or $y_0 \notin B$. If, for instance, $x_0 \notin A$, then by Proposition 1, $B = A^{\uparrow_I} = A^{\uparrow_J}$, showing B is the intent of a $d \in \mathcal{B}(J)$. Now by Proposition 1,

$$B^{\downarrow_J} = \begin{cases} B^{\downarrow_I} = A & \text{if } y_0 \notin B, \\ B^{\downarrow_I} \setminus \{x_0\} = A \setminus \{x_0\} = A & \text{if } y_0 \in B \end{cases}$$

and so $d = c$. The case $y_0 \notin B$ is dual.

To prove the opposite direction it is sufficient to notice that $c \in [\gamma_I(x_0), \mu_I(y_0)]$ is equivalent to $\langle x_0, y_0\rangle \in A \times B$, excluding the case $\langle A, B\rangle \in \mathcal{B}(J)$. □

For concepts $c = \langle A, B\rangle \in \mathcal{B}(I), d = \langle C, D\rangle \in \mathcal{B}(J)$ we set

$$c^{\square} = \langle A^{\square}, B^{\square}\rangle = \langle A^{\uparrow_J\downarrow_J}, A^{\uparrow_J}\rangle, \qquad c_{\square} = \langle A_{\square}, B_{\square}\rangle = \langle B^{\downarrow_J}, B^{\downarrow_J\uparrow_J}\rangle,$$
$$d^{\boxtimes} = \langle C^{\boxtimes}, D^{\boxtimes}\rangle = \langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle, \qquad d_{\boxtimes} = \langle C_{\boxtimes}, D_{\boxtimes}\rangle = \langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle.$$

Evidently, $c^{\square}, c_{\square} \in \mathcal{B}(J)$ and $d^{\boxtimes}, d_{\boxtimes} \in \mathcal{B}(I)$. $c^{\square}$ (resp. $c_{\square}$) is called *the upper* (resp. *lower*) *child of* c. In our setting, $d^{\boxtimes} = d_{\boxtimes}$ (it would not be the case if $I \setminus J$ had more than one element). It is the (unique) concept from $\mathcal{B}(I)$, containing, as a rectangle, the rectangle represented by d.

The following theorem shows basic properties of the pairs $\langle^{\square}, ^{\boxtimes}\rangle$ and $\langle_{\square}, _{\boxtimes}\rangle$.

Proposition 3 (child operators). *The mappings $c \mapsto c^{\square}$, $c \mapsto c_{\square}$, and $d \mapsto d^{\boxtimes}$ are isotone and satisfy*

$$c \le c^{\square\boxtimes}, \qquad d \le d^{\boxtimes\square}, \qquad c^{\square\boxtimes\square} = c^{\square}, \qquad d^{\boxtimes\square\boxtimes} = d^{\boxtimes},$$
$$c \ge c_{\square\boxtimes}, \qquad d \ge d_{\boxtimes\square}, \qquad c_{\square\boxtimes\square} = c_{\square}, \qquad d_{\boxtimes\square\boxtimes} = d_{\boxtimes}.$$

Proof. Isotony follows directly from definition.

Let $c = \langle A, B\rangle$. From Proposition 1 we have $A^{\uparrow_J} \subseteq A^{\uparrow_I}$. Thus, $A = A^{\uparrow_I\downarrow_I} \subseteq A^{\uparrow_J\downarrow_I}$, whence $c \le c^{\square\boxtimes}$. Similarly, for $d = \langle C, D\rangle$, $D^{\downarrow_J} \subseteq D^{\downarrow_I}$, whence $D^{\downarrow_I\uparrow_J} \subseteq D^{\downarrow_J\uparrow_J} = D$.

To prove $c^{\square\boxtimes\square} = c^{\square}$ it suffices to show that for the extent A of c it holds $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J}$. By Proposition 1, we have two possibilities: either $A^{\uparrow_J} = A^{\uparrow_I}$, or $A^{\uparrow_J} = A^{\uparrow_I} \setminus \{y_0\}$. In the first case $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J}$ holds trivially, in the second case $A^{\uparrow_J\downarrow_I} = A^{\uparrow_J\downarrow_J}$ (by the same proposition, because $y_0 \notin A^{\uparrow_J}$) and $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J\downarrow_J\uparrow_J} = A^{\uparrow_J}$. The equality $d^{\boxtimes\square\boxtimes} = d^{\boxtimes}$ can be proved similarly.

The assertions for lower children are dual. □

Corollary 1. *The mappings $c \mapsto c^{\square\boxtimes}$ and $d \mapsto d^{\boxtimes\square}$ are closure operators and the mappings $c \mapsto c_{\square\boxtimes}$ and $d \mapsto d_{\boxtimes\square}$ are interior operators.*

Following two theorems utilize the operators $^{\square}, ^{\boxtimes}, _{\square}, _{\boxtimes}$ to give several equivalent characterizations of stable concepts. First we prove a proposition.

Proposition 4. *The following assertions are equivalent for any $c = \langle A, B\rangle \in \mathcal{B}(I)$.*

1. *c is stable,*
2. *$A^{\uparrow_I} = A^{\uparrow_J}$,*
3. *$B^{\downarrow_I} = B^{\downarrow_J}$.*

Proof. "2 ⇒ 3": by Proposition 1, $A \subseteq A^{\uparrow_J\downarrow_J} = B^{\downarrow_J} \subseteq B^{\downarrow_I} = A$.

"3 ⇒ 2": dual.

The other implications follow by definition, since c is stable iff both 2. and 3. are satisfied. □

Proposition 5 (stable concepts in $\mathcal{B}(I)$). *The following assertions are equivalent for a concept $c \in \mathcal{B}(I)$:*

1. *c is stable,*
2. $c \notin [\gamma_I(x_0), \mu_I(y_0)]$*,*
3. $c = c^{\square}$*,*
4. $c = c_{\square}$*,*
5. $c^{\square} = c_{\square}$*.*

Proof. Directly from Proposition 4. □

Proposition 6 (stable concepts in $\mathcal{B}(J)$). *The following assertions are equivalent for a concept $d \in \mathcal{B}(J)$:*

1. *d is stable,*
2. $d = d^{\boxtimes}$*,*
3. *$d^{\boxtimes}$ is stable.*

Proof. Directly from Proposition 4. □

4 Computing $\mathcal{B}(J)$ without structural information

Proposition 7. *The following holds for $c = \langle A, B\rangle \in \mathcal{B}(I)$ and $d = \langle C, D\rangle \in \mathcal{B}(J)$: If $d = c^{\square}$, then $B \in \{D, D \cup \{y_0\}\}$ and if $d = c_{\square}$, then $A \in \{C, C \cup \{x_0\}\}$.*

Proof. By definition of $^{\square}$, $D = A^{\uparrow_J}$, which is by Proposition 1 either equal to B, or to $B \setminus \{y_0\}$. Similarly for $_{\square}$. □

Proposition 8. *A non-stable concept $d \in \mathcal{B}(J)$ is a (upper or lower) child of exactly one concept $c \in \mathcal{B}(I)$. This concept is non-stable and satisfies $c = d^{\boxtimes} = d_{\boxtimes}$.*

Proof. Let $d = \langle C, D\rangle$. Since d is non-stable, then either $C^{\uparrow_I} \neq C^{\uparrow_J}$, or $D^{\downarrow_I} \neq D^{\downarrow_J}$. Suppose $C^{\uparrow_I} \neq C^{\uparrow_J}$ and set $A = C$, $B = C^{\uparrow_I}$. By Proposition 1, $x_0 \in C$, $y_0 \notin D$ and $B = D \cup \{y_0\}$. By the same proposition, $A = C = D^{\downarrow_J} = D^{\downarrow_I}$, whence A is an extent of I. Thus, $c = \langle A, B\rangle \in \mathcal{B}(I)$ and it is non-stable because $x_0 \in A$ and $y_0 \in B$ (Proposition 2). Since $D = C^{\uparrow_J} = A^{\uparrow_J}$, $d = c^{\square}$. $A = C$ yields $c = d_{\boxtimes}$.

We prove uniqueness of c. By Proposition 7, if for $c' = \langle A', B'\rangle \in \mathcal{B}(I)$ we have $d = c'^{\square}$, then either $B' = D$, or $B' = D \cup \{y_0\}$. The first case is impossible, because it would make D an intent of I and, consequently, d a stable concept. The second case means c' equals c above. There is a third case left: if $d = c'_{\square}$, then $C = B'^{\downarrow_J}$. Since $x_0 \in C$, we have $y_0 \notin B'$ (Proposition 1). Thus, $C = B'^{\downarrow_I}$ (Proposition 1 again). Consequently, $C^{\uparrow_I} = B'$ and since $y_0 \notin B'$, $B' = C^{\uparrow_J}$ (Proposition 1 for the last time). Thus, $d = c'$, which is a contradiction with non-stability of d.

The case $D^{\downarrow_I} \neq D^{\downarrow_J}$ is proved dually (in this case we obtain $d = c_{\square}$). □

The meaning of the previous theorem is that for each non-stable concept in $\mathcal{B}(J)$ there exists exactly one non-stable concept in $\mathcal{B}(I)$, such that these two are related via mappings $^{\square}, ^{\boxtimes}$ or $_{\square}, _{\boxtimes}$.

The theorem leads the following simple way of constructing $\mathcal{B}(J)$ from $\mathcal{B}(I)$. For each $c \in \mathcal{B}(I)$ the following has to be done:

1. If c is stable, then it has to be added to $\mathcal{B}(J)$.
2. If c is not stable, then each its non-stable child (i.e., each non-stable element of $\{c^{\square}, c_{\square}\}$) has to be added to $\mathcal{B}(J)$.

This method ensures that all proper elements will be added to $\mathcal{B}(J)$ (i.e., no element will be omitted) and each element will be added exactly once.

Stable (resp. non-stable) concepts can be identified by means of Proposition 11. The following proposition shows a simple way of detecting whether a child of a non-stable concept from $\mathcal{B}(I)$ is stable. It also describes the role of fixpoints of operators ${}^{\square\boxtimes}$ and ${}_{\square\boxtimes}$.

Proposition 9. *Let $c \in \mathcal{B}(I)$ be non-stable. Then*

- *$c^{\square}$ is non-stable iff c is a fixpoint of ${}^{\square\boxtimes}$,*
- *$c_{\square}$ is non-stable iff c is a fixpoint of ${}_{\square\boxtimes}$.*

Proof. If $c^{\square}$ is not stable, then $c = (c^{\square})^{\boxtimes}$ by Theorem 8. On the other hand, if $c^{\square}$ is stable, then $c^{\square\boxtimes} = c^{\square}$ by Theorem 6, which rules out $c^{\square\boxtimes} = c$, because in that case c would be equal to $c^{\square}$, which would make it stable by Theorem 5.

The proof for $c_{\square}$ is dual. □

Example 1. In Fig. 1 we can see some examples of contexts with concepts of different types w.r.t. operators ${}^{\square\boxtimes}$, ${}_{\square\boxtimes}$.

The method is utilized in Algorithm 1.

Algorithm 1 Transforming $\mathcal{B}(I)$ into $\mathcal{B}(J)$ (without structural information).

```
procedure TransformConcepts(B(I))
    B(J) ← B(I);
    for all c = ⟨A, B⟩ ∈ [γ_I(x_0), μ_I(y_0)] do
        B(J) ← B(J) \ {c};
        if c = c_□⊠ then
            B(J) ← B(J) ∪ {c_□};
        end if
        if c = c^□⊠ then
            B(J) ← B(J) ∪ {c^□};
        end if
    end for
    return B(J);
end procedure
```

Time complexity of Algorithm 1 is clearly $O(|\mathcal{B}(I)||X||Y|)$ in the worst case scenario. Indeed, the number of non-stable concepts is at most equal to $|\mathcal{B}(I)|$ and the computation of operators ${}^{\square\boxtimes}$, ${}_{\square\boxtimes}$ can be done in $O(|X| \cdot |Y|)$ time.

5 Computing $\mathcal{B}(J)$ with structural information

To analyze changes in the structure of a concept lattice after removing an incidence, we need to investigate deeper properties of the closure operator ${}^{\square\boxtimes}$ and the interior operator ${}_{\square\boxtimes}$ and the sets of their fixpoints.

	y_0	y_1	y_2
x_0	●	×	×
x_1			
x_2			

(a) The least concept is not stable and is a fixpoint of both operators.

	y_1	y_2	y_3	y_0
x_0	×	×	×	●
x_1			×	×
x_2		×		×
x_3	×			×

(b) Several non-trival non-stable concepts are fixpoints of both operators.

	y_1	y_2	y_0
x_0		×	●
x_1		×	×
x_2		×	

(c) Concept $\langle\{x_0, x_1\}, \{y_0, y_2\}\rangle$ is a fixpoint of $_{\square\boxtimes}$, but not $^{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	●	×	
x_1	×	×	×
x_2			

(d) Concept $\langle\{x_0, x_1\}, \{y_0, y_1\}\rangle$ is a fixpoint of $^{\square\boxtimes}$, but not $_{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	●		
x_1	×	×	
x_2			

(e) Concept $\langle\{x_0, x_1\}, \{y_0\}\rangle$ is not a fixpoint of any operator.

	y_1	y_2	y_3	y_4	y_0
x_0		×		×	●
x_1			×	×	×
x_2				×	
x_3	×	×			×
x_4		×			

(f) Two concepts are not fixpoints of any operator.

Fig. 1: Examples of contexts with concepts of different types w.r.t. operators $^{\square\boxtimes}$, $_{\square\boxtimes}$.

Proposition 10. *Each stable concept is a fixpoint of both* $^{\square\boxtimes}$ *and* $_{\square\boxtimes}$.

Proof. Follows directly from Theorem 5 and Theorem 6. □

Since $_{\square\boxtimes}$ is an interior operator and $^{\square\boxtimes}$ is a closure operator on $\mathcal{B}(I)$, we have for each $c \in \mathcal{B}(I)$, $c_{\square\boxtimes} \leq c \leq c^{\square\boxtimes}$. Thus, we can consider the interval $[c_{\square\boxtimes}, c^{\square\boxtimes}] \subseteq \mathcal{B}(I)$.

Proposition 11. *For any* $c \in \mathcal{B}(I)$*, each concept from* $[c_{\square\boxtimes}, c^{\square\boxtimes}] \setminus \{c\}$ *is stable.*

Proof. First we prove that either $c^{\square\boxtimes}$ equals c, or is its upper neighbor. Let $c = \langle A, B\rangle$. By definition, the intent of $c^{\square\boxtimes}$ is equal to $A^{\uparrow_J \downarrow_I \uparrow_I}$. By Proposition 1, $A^{\uparrow_J} \in \{B, B \setminus \{y_0\}\}$. Thus, $A^{\uparrow_J \downarrow_I \uparrow_I} \in \{B, B \setminus \{y_0\}\}$. If it equals B, then $c^{\square\boxtimes} = c$. Otherwise the intents of c and $c^{\square\boxtimes}$ differ in exactly one attribute, which makes c and $c^{\square\boxtimes}$ neighbors. Also notice that in this case $c^{\square\boxtimes}$ is stable because its intent does not contain y_0 (Proposition 2).

Now let $c' \leq c^{\square\boxtimes}$ be non-stable. If $c = c^{\square\boxtimes}$, then $c' \leq c$. If $c < c^{\square\boxtimes}$, then c is non-stable (Proposition 10) whereas $c^{\square\boxtimes}$ is stable. Non-stable concepts in $\mathcal{B}(I)$

form an interval (Theorem 5). Thus, $c' \vee c$ is non-stable and should be less than $c^{\square\boxtimes}$. Hence, $c' \vee c = c$ (c is a lower neighbor of $c^{\square\boxtimes}$), concluding $c' \leq c$ again.

In a similar way we obtain the inequality $c' \geq c$ for each non-stable $c' \geq c_{\square\boxtimes}$. □

The following proposition shows an important property of the sets of fixpoints w.r.t. the ordering on $\mathcal{B}(I)$: The set of fixpoints of ${}^{\square\boxtimes}$ is a lower set whereas the set of fixpoints of ${}_{\square\boxtimes}$ is an upper set.

Proposition 12. *Let $c \in \mathcal{B}(I)$ be a non-stable concept. If c is a fixpoint of ${}^{\square\boxtimes}$, then each $c' \leq c$ is also a fixpoint of ${}^{\square\boxtimes}$. If c is a fixpoint of ${}_{\square\boxtimes}$, then each $c' \geq c$ is also a fixpoint of ${}_{\square\boxtimes}$.*

Proof. Let $c = c^{\square\boxtimes}$ and $c' \leq c$. If c' is stable, then the assertion follows by Proposition 10. Suppose c' is not stable. By extensivity and isotony of ${}^{\square\boxtimes}$, $c' \leq c'^{\square\boxtimes} \leq c^{\square\boxtimes} = c$. Thus, $c'^{\square\boxtimes}$ is not stable (Proposition 2) and $c'^{\square\boxtimes} = c'$ by Proposition 11.

The case $c = c_{\square\boxtimes}$ is dual. □

The above results are used in Algorithm 2, which computes the lattice $\mathcal{B}(J)$ together with the information of its ordering. The algorithm is more complicated than the previous one. We provide a short description of the algorithm, together with some examples. Due to space limitations, we will not dwell into details. We will also leave out dual parts of similar cases.

The algorithm processes all non-stable concepts of $\mathcal{B}(I)$ in a bottom-up direction, using an arbitrary linear ordering $\sqsubseteq$ such that if $c_1 \leq c_2$, then $c_1 \sqsubseteq c_2$. Each concept is either modified (by removing x_0 from the extent or y_0 from intent), or disposed of entirely. Sometimes, new concepts are created. All concepts also get updated their lists of upper and lower neighbors.

Let $c = \langle A, B\rangle$ be an arbitrary non-stable concept from $\mathcal{B}(I)$ ($c \in [\gamma_I(x_0), \mu_I(y_0)]$).

- If $c = c^{\square\boxtimes}$, $c = c_{\square\boxtimes}$, then c will "split" into $d_1 \leq d_2$.
  - We set $d_1 = c_\square$ and $d_2 = c^\square$.
  - The concept d_1 will be a lower neighbor of d_2.
  - If for a lower neighbor c_l of c it holds $c_l = {c_l}^{\square\boxtimes}$, $c_l \neq c_{l\square\boxtimes}$, then it will be a lower neighbor of d_2. It is necessary to check whether d_1 and $c_{l\square\boxtimes}$ will be neighbors. It certainly holds $c_{l\square\boxtimes} \leq d_1$, but there can be a concept k, such that $c_{l\square\boxtimes} \leq k \leq d_1$.
  - Dually for upper neighbors.
  - If for a non-stable neighbor c_n of c it holds $c_n = {c_n}^{\square\boxtimes}$, $c_n = c_{n\square\boxtimes}$, i.e., the same conditions as for c (c_n will split into d_{n_1}, d_{n_2}), then d_1, d_{n_1} and d_2, d_{n_2} will be neighbors.
  - All other upper (resp. lower) neighbors will be neighbors of d_2 (resp. d_1).
- If $c = c^{\square\boxtimes}$ and $c \neq c_{\square\boxtimes}$, then c will lose y_0 from its intent.
  - Denote the transformed c as $d = \langle C, D\rangle = c^\square = \langle A, B \setminus \{y_0\}\rangle$.

  - If for an upper neighbor c_u it holds $c_u = c_{u\Box\boxtimes}$, $c_u \neq {c_u}^{\Box\boxtimes}$ (c_u will lose x_0 from its extent), then c_u and d will become incomparable. It is necessary to check whether $c_{\Box\boxtimes}, c_u$ and $c, {c_u}^{\Box\boxtimes}$ should be neighbors (again, there can be a concept between them).
- If $c \neq c^{\Box\boxtimes}$ and $c = c_{\Box\boxtimes}$, then c will lose x_0 from its extent.
  - Denote transformed c as $d = \langle C, D\rangle = c_{\Box} = \langle A \setminus \{x_0\}, B\rangle$.
- If $c \neq c^{\Box\boxtimes}$ and $c \neq c_{\Box\boxtimes}$, then c will vanish entirely.
  - It is necessary to check whether $c_{\Box\boxtimes}$ and $c^{\Box\boxtimes}$ should be neighbors (again, a concept can lie between them).
  - Denote by U the set of all upper neighbors of c, except for $c^{\Box\boxtimes}$. There is no fixed point of ${}^{\Box\boxtimes}$ among the elements from U.
  - Denote by L the set of all lower neighbors of c, except for $c_{\Box\boxtimes}$.
  - Concepts from U and L will not be neighbors.
    Concepts will either become incomparable or one of them or both will vanish. There is also no need for additional checks regarding neighborhood relationship between concepts from U and $c_{\Box\boxtimes}$ (resp. L and $c^{\Box\boxtimes}$) or their neighbors.
  - It holds $\forall c_l \in L: \; c_l \leq c \leq c^{\Box\boxtimes}$, but it is necessary to check if there is a concept between them.
  - Similarly, it holds $\forall c_u \in U : c_{\Box\boxtimes} \leq c \leq c_u$, but again it is necessary to check if there is a concept between them.

The number of iterations in TransformConceptLattice is at most $|\mathcal{B}(I)|$, which occurs when each concept in $\mathcal{B}(I)$ is non-stable. In each of the iterations, tests $c = c^{\Box\boxtimes}$ and $c = c_{\Box\boxtimes}$ are performed and one of the procedures SplitConcept, RelinkReducedIntent, UnlinkVanishedConcept is called. It can be easily seen that the tests can be performed quite efficiently and do not add to the time complexity.

The most time consuming among the above three procedures is SplitConcept. It iterates through all upper (which can be bounded by $|X|$) and lower (which can be bounded by $|Y|$) neighbors of the concept c. For each of the neighbors it might be necessary to check if the interval between the neighbor and certain other concept is empty (and we should make a new edge). This can be done by checking intents/extents of its neighbors.

The above considerations lead to the result that time complexity of Algorithm 2 is in the worst case $O(|\mathcal{B}| \cdot |X|^2 \cdot |Y|)$.

Example 2. In Fig. 2, we can see some examples of transformations of non-stable concepts from $\mathcal{B}(I)$ into concepts of $\mathcal{B}(J)$.

In Algorithm 2 we will assume that following functions are already defined:

- $UpperNeighbors(c)$ - returns upper neighbors of c;
- $LowerNeighbors(c)$ - returns lower neighbors of c;
- $Link(c_1, c_2)$ - introduces neighborhood relationship between c_1 and c_2;
- $Unlink(c_1, c_2)$ - cancels neighborhood relationship between c_1 and c_2.

Algorithm 2 Transforming $\mathcal{B}(I)$ with structural information into $\mathcal{B}(J)$.

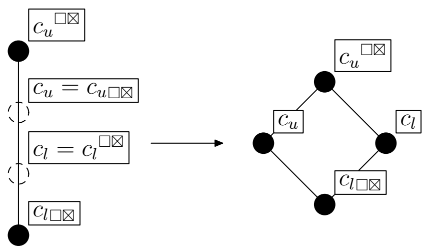
procedure LinkIfNeeded(c_1, c_2)
  if $\nexists k \in \mathcal{B}(I) : c_1 < k < c_2$ **then**
    $Link(c_1, c_2)$;
  end if
end procedure

procedure SplitConcept($c \in [\gamma_I(x_0), \mu_I(y_0)]$)
  $d_1 = c_{\square}$; $d_2 = c^{\square}$;
  $Link(d_1, d_2)$;
  for all $u \in UpperNeighbors(c)$ **do**
    $Unlink(c, u)$; $Link(d_2, u)$;
  end for
  for all $l \in LowerNeighbors(c)$ **do**
    $Unlink(l, c)$; $Link(l, d_1)$;
  end for
  for all $u \in UpperNeighbors(c)$ **do**
    if $u \neq u^{\square\boxtimes}$ **then**
      $Unlink(d_2, u)$; $Link(d_1, u)$; $LinkIfNeeded(d_2, u^{\square\boxtimes})$;
    end if
  end for
  for all $l = \langle C, D \rangle \in LowerNeighbors(c)$ **do**
    if $y_0 \notin D$ **then**
      $Unlink(l, d_1)$; $Link(l, d_2)$; $LinkIfNeeded(l_{\boxtimes\square}, d_1)$;
    end if
  end for
  return d_1, d_2;
end procedure

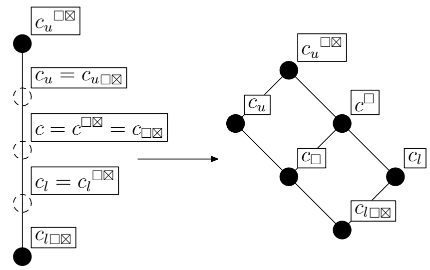
procedure RelinkReducedIntent($c \in [\gamma_I(x_0), \mu_I(y_0)]$)
  for all $u = \langle C, D \rangle \in UpperNeighbors(c)$ **do**
    if $u \neq u^{\square\boxtimes}$ **then**
      $Unlink(c, u)$;
      $LinkIfNeeded(c_{\square\boxtimes}, u)$; $LinkIfNeeded(c, u^{\square\boxtimes})$;
    end if
  end for
end procedure

procedure UnlinkVanishedConcept($c \in [\gamma_I(x_0), \mu_I(y_0)]$)
  for all $u \in UpperNeighbors(c)$ **do**
    $Unlink(c, u)$; $LinkIfNeeded(c_{\square\boxtimes}, u)$;
  end for
  for all $l \in LowerNeighbors(c)$ **do**
    $Unlink(l, c)$;
  end for
end procedure

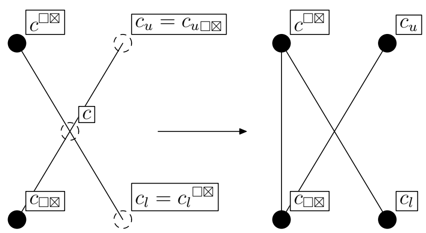
procedure TransformConceptLattice($\mathcal{B}(I)$)
  for all $c = \langle A, B \rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ from least to largest w.r.t. $\sqsubseteq$ **do**
    if $c = c^{\square\boxtimes}$ **and** $c = c_{\square\boxtimes}$ **then** ▷ Concept will split.
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \setminus \{c\}$;
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \cup SplitConcept(c)$;
    else if $c \neq c^{\square\boxtimes}$ **and** $c = c_{\square\boxtimes}$ **then** ▷ Extent will be smaller.
      $A \leftarrow A \setminus \{x_0\}$;
    else if $c = c^{\square\boxtimes}$ **and** $c \neq c_{\square\boxtimes}$ **then** ▷ Intent will be smaller.
      $RelinkReducedIntent(c)$;
      $B \leftarrow B \setminus \{y_0\}$;
    else if $c \neq c^{\square\boxtimes}$ **and** $c \neq c_{\square\boxtimes}$ **then** ▷ Concept will vanish.
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \setminus \{c\}$;
      $UnlinkVanishedConcept(c)$;
    end if
  end for
end procedure



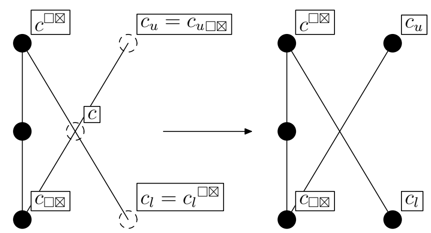
(a) Concepts become incomparable.



(b) Concept in middle "splits into two".



(c) Concept in the middle vanishes.



(d) Concept in the middle vanishes. There is already another concept between its children.

Fig. 2: Examples of transformations of non-stable concepts from $\mathcal{B}(I)$ into concepts of $\mathcal{B}(J)$.

6 Conclusion

We analyzed changes of the structure of a concept lattice, caused by removal of exactly one incidence from the associated formal context. We proved some theoretical results and presented two algorithms with time complexities $O(|\mathcal{B}| \cdot |X| \cdot |Y|)$ (Algorithm 1; without structure information) and $O(|\mathcal{B}| \cdot |X|^2 \cdot |Y|)$ (Algorithm 2; with structure information).

There exist several algorithms for incremental computation of concept lattice [1, 5, 8, 6, 7, 2], based on addition and/or removal of objects. Our approach is new in that we recompute a concept lattice based on a removal of just one incidence.

Note that the algorithm proposed by Nourine and Raynaud in [7] has time complexity $O((|Y| + |X|) \cdot |X| \cdot |\mathcal{B}|)$, which is better than complexity of our Algorithm 2. However, experiments presented in [5] indicate that this algorithm sometimes performs slower than some algorithms with time complexity $O(|\mathcal{B}| \cdot |X|^2 \cdot |Y|)$. In the case of our Algorithm 2, some preliminary experiments indicate that the size of the interval of non-stable concepts is usually relatively small, which substantially reduces the overall processing time of the algorithm.

A natural next step would be investigate adding incidences to a formal context, instead of removing. This problem, however, seems to be more difficult than the first one, namely because the set of non-stable concepts in the lattice $\mathcal{B}(J)$ has more complicated structure (it is not an interval) and also because not all non-stable concepts in $\mathcal{B}(I)$ can be computed via the operator $^{\boxtimes}$. We will try to address this issues in the future. We will also focus on the following:

– experimenting with proposed algorithms on various datasets and comparing them with other known algorithms,
– generalizing the results to allow removing and adding more incidences at the same time.

References

1. Carpineto, C., Romano, G.: Concept Data Analysis: Theory and Applications. John Wiley & Sons (2004)
2. Dowling, C.E.: On the irredundant generation of knowledge spaces. J. Math. Psychol. 37(1), 49–62 (1993)
3. Ganter, B., Wille, R.: Formal Concept Analysis – Mathematical Foundations. Springer (1999)
4. Kuznetsov, S.O., Obiedkov, S.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theoretical Artificial Intelligence 14, 189–216 (2002)
5. Merwe, D., Obiedkov, S., Kourie, D.: Addintent: A new incremental algorithm for constructing concept lattices. In: Eklund, P. (ed.) Concept Lattices, Lecture Notes in Computer Science, vol. 2961, pp. 372–385. Springer Berlin Heidelberg (2004)
6. Norris, E.M.: An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine de Mathématiques Pures et Appliquées 23(2), 243–250 (1978)
7. Nourine, L., Raynaud, O.: A fast algorithm for building lattices. Inf. Process. Lett. 71(5-6), 199–204 (1999)
8. Outrata, J.: A lattice-free concept lattice update algorithm based on *CbO. In: Ojeda-Aciego, M., Outrata, J. (eds.) CLA. CEUR Workshop Proceedings, vol. 1062, pp. 261–274. CEUR-WS.org (2013)
9. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered Sets, pp. 445–470. Boston (1982)