A Generalized Next-Closure Algorithm – Enumerating Semilattice Elements from a Generating Set

Daniel Borchmann
TU Dresden, Institute of Algebra
daniel.borchmann@mailbox.tu-dresden.de

September 11, 2012

Abstract

A generalization of the well known Next-Closure algorithm is presented, which is able to enumerate finite semilattices from a generating set. We prove the correctness of the algorithm and apply it on the computation of the intents of a formal context.

1 Introduction

Next-Closure is one of the best known algorithms in Formal Concept Analysis [8] to compute the concepts of a formal context. In its general form it is able to efficiently enumerate the closed sets of a given closure operator on a finite set. This generality might be a drawback concerning efficiency compared to other algorithms like Close-by-One [1, 9, 11]. On the other hand, the general formulation of Next-Closure widens its field of application. However, there are still applications where Next-Closure might be useful, but is not applicable, because a closure operator on a finite set is not explicitly available. One such example might be the computation of concepts of a fuzzy formal context [3]. In those cases most often an ad hoc variation of Next-Closure can be constructed. The aim of this paper is to provide a generalization of Next-Closure which covers those cases, and may even go beyond them.

As it turns out, Next-Closure is not about enumerating closed sets of a closure operator, even not on an abstract ordered set. The algorithm is merely about enumerating elements of a certain semilattice, given as an operation together with a generating set. This observation shall turn out to be quite natural.

It has to be noted that there have been prior attempts to generalize Next-Closure to a more general setting [7]. But this approach is, as far as the author can tell, not related to the one presented in this paper.

This paper is organized as follows. First of all we shall revisit the original version of Next-Closure, together with the basic definitions. Then we present our generalized version working on semilattices, together with a complete proof of its correctness. Then we

show how this generalized form is indeed a generalization of the original Next-Closure. Additionally, we present another algorithm for enumerating the intents of a given formal context, which is very similar to Close-by-One. Finally, we give some outlook on further questions which might be interesting within this line of research.

2 The Next-Closure Algorithm

Before we are going to discuss our generalized form of Next-Closure, let us revisit the original version as it is given in [6, 8]. To make our discussion a bit more consistent, we shall allow ourselves a minor deviation from the standard description of the algorithm, which we shall mention explicitly.

Let M be a finite set and let $c : \mathfrak{P}(P) \longrightarrow \mathfrak{P}(P)$ be a function such that

a) c is idempotent, i.e. $c(c(A)) = c(A)$ for all $A \subseteq M$,

b) c is monotone, i.e. if $A \subseteq B$, then $c(A) \subseteq c(B)$ for all $A, B \subseteq M$, and

c) c is extensive, i.e. $A \subseteq c(A)$ for all $A \subseteq M$.

The mapping c is then said to be a *closure operator* on P. A set $A \subseteq M$ is called *closed (with respect to c)* if $A = c(A)$, and the *image of c* is defined as

$$c[\mathfrak{P}(M)] := \{\, c(A) \mid A \subseteq M \,\}.$$

Without loss of generality, let $M = \{\, 1, \ldots, n \,\}$ for some $n \in \mathbb{N}$. For two sets $A, B \in c[\mathfrak{P}(M)]$ with $A \neq B$ and $i \in M$ we say that A is *lectically smaller* than B *at position* i if and only if

$$i = \min(A \,\Delta\, B) \quad \text{and} \quad i \in B,$$

where $A \,\Delta\, B = (A \backslash B) \cup (B \backslash A)$ denotes the *symmetric difference* of A and B. We shall write $A \prec_i B$ if A is lectically smaller than B at position i. Finally, we say that A is *lectically smaller* than B, for $A, B \in c[\mathfrak{P}(M)]$, if $A = B$ or $A \prec_i B$ for some $i \in M$. We shall write $A \preceq B$ in this case.

It has to be noted that, in contrast to our definition, the lectic order is normally defined for all sets $A, B \subseteq M$ in the very same spirit as given above. However, as we shall see, this is not necessary, which is why we have restricted our definition to closed sets only.

Now let us define for $A \in c[\mathfrak{P}(M)]$ and $i \in M$

$$A \oplus i := c(\{\, j \in A \mid j < i \,\} \cup \{\, i \,\}).$$

Then we have the following result.

2.1 Theorem (Next-Closure [6]) *Let $A \in c[\mathfrak{P}(M)]$. Then the next closed set $A^+ \in c[\mathfrak{P}(M)]$ after A with respect to the lectic order $\preceq$, if it exists, is given by*

$$A^+ = A \oplus i$$

with $i \in M$ being maximal with $A \prec_i A \oplus i$.

This is the original version of Next-Closure, as it is given in [6, 8]. Therein, term "next closed set" has the obvious meaning, namely

$$A^+ = \min B \in c[\mathfrak{P}(M)] \mid A \prec B.$$

Our generalization now starts with the following observation: the set $A \oplus i$ can be seen as the smallest closed set containing both $\{ j \in A \mid j < i \}$ and $\{ i \}$, or equivalently, both $c(\{ j \in A \mid j < i \})$ and $c(\{ i \})$. This means that we can rewrite $A \oplus i$ as

$$A \oplus i = c(\{ j \in A \mid j < i \}) \vee c(\{ i \}),$$

where $X \vee Y$ is the smallest closed set containing both $X, Y \in c[\mathfrak{P}(M)]$, the *supremum of* X *and* Y, which is simply given by $X \vee Y = c(X \cup Y)$. This observation suggests to consider Next-Closure on abstract algebraic structures with a binary operation $\vee$ with some certain properties, namely on *semi-lattices*. To do so we need a more general notion of $c(\{ i \})$, since we do not necessarily deal with subsets, and a more general notion of $\{ j \in A \mid j < i \}$, which likewise might not be expressible in a more general setting. Finally, we need to find a starting point for our enumeration, which is $c(\varnothing)$ in the original description of Next-Closure, but may vary in other cases. Luckily, all this is possible and quite natural, as we shall see in the next section.

3 Generalizing Next-Closure for Semilattices

The aim of this section is to present a generalization of the Next-Closure algorithm that works on semilattices. For this recall that a *semilattice* $\underline{L} = (L, \vee)$ is an algebraic structure with a binary operation $\vee$ which is associative, commutative and idempotent. It is well known that by

$$x \leqslant_{\underline{L}} y \ :\Longleftrightarrow \ x \vee y = y, \quad (x, y \in L)$$

an order relation on L is defined in such a way that for every two elements $a, b \in L$ the element $a \vee b$ is the least upper bound of both a and b with respect to $\leqslant_{\underline{L}}$.

For the remainder of this section let $\underline{L} = (L, \vee)$ be an arbitrary but fixed semilattice. Furthermore, let $(x_i \mid i \in I)$ be an enumeration of a finite generating set $\{ x_i \mid i \in I \} \subseteq L$ of $\underline{L}$. Finally, let $\leqslant_I$ be a total order on I.

3.1 Definition Let $a, b \in L$ and let $i \in I$. Set

$$\Delta_{a,b} := \{ j \in I \mid (x_j \leqslant_{\underline{L}} a \text{ and } x_j \not\leqslant_{\underline{L}} b) \text{ or } (x_j \not\leqslant_{\underline{L}} a \text{ and } x_j \leqslant_{\underline{L}} b) \}.$$

We then define

$$a <_i b \ :\Longleftrightarrow \ i = \min \Delta_{a,b} \text{ and } x_i \leqslant_{\underline{L}} b.$$

Furthermore we write $a < b$ if $a <_i b$ for some $i \in I$ and write $a \leqslant b$ if $a = b$ or $a < b$. $\diamondsuit$

One can see the similarity of this definition to the one of the lectic order. Here, the set $\Delta_{a,b}$ generalizes $a \, \Delta \, b$ and $x_i \leqslant_{\underline{L}} b$ somehow represents the fact that $i \in b$, or equivalently $\{ i \} \subseteq b$, in the special case of $L = \mathfrak{P}(M)$ and $i \in M$.

Note that if $a <_i b$ and $k \in I$ with $k <_I i$, then

$$x_k \leqslant_{\underline{L}} a \iff x_k \leqslant_{\underline{L}} b.$$

This observation is quite useful and will be used in some of the proofs later on.

The first thing we want to consider now are two easy results stating that $\leqslant$ is a total order relation on L extending $\leqslant_{\underline{L}}$.

3.2 Lemma *The relation $<$ is irreflexive and transitive. Furthermore, for every two elements $a, b \in L$ with $a \neq b$, it is either $a < b$ or $b < a$.*

Proof If $a = b$, then the set $\Delta_{a,b}$ defined above is empty, therefore we cannot have $a <_i a$ for some $i \in I$. This shows the irreflexivity of $<$. Let us now consider the transitivity of $<$. For this let $a, b, c \in L$, $i, j \in I$ and suppose that $a <_i b$ and $b <_j c$. We have to show that $a < c$. Let us consider the following cases.

Case $i <_I j$. We have $x_i \not\leqslant_{\underline{L}} a$ and $x_i \leqslant_{\underline{L}} b$ because of $a <_i b$. Due to $i <_I j$ it follows that $x_i \leqslant_{\underline{L}} c$. Suppose that there exists $k \in I$, $k <_I i$ with $x_k \leqslant_{\underline{L}} a$ and $x_k \not\leqslant_{\underline{L}} c$. Then if $x_k \not\leqslant_{\underline{L}} b$ we would have $x_k \not\leqslant_{\underline{L}} a$ because of $k <_I i$, a contradiction. But if $x_k \leqslant_{\underline{L}} b$, then $x_k \leqslant_{\underline{L}} c$ because of $k <_I i <_I j$, again a contradiction. Thus we have shown that $a < c$.

Case $j <_I i$. We have $x_j \not\leqslant_{\underline{L}} b$, $x_j \leqslant_{\underline{L}} c$ because of $b <_j c$. Due to $j <_I i$ it follows that $x_j \not\leqslant_{\underline{L}} a$. Now if there were a $k \in I$, $k <_I j$ with $x_k \leqslant_{\underline{L}} a$ and $x_k \not\leqslant_{\underline{L}} c$, then $x_k \leqslant_{\underline{L}} b$ would imply $x_k \leqslant_{\underline{L}} c$ and $x_k \not\leqslant_{\underline{L}} b$ would imply $x_k \not\leqslant_{\underline{L}} a$, analogously to the first case, a contradiction. Hence such a k cannot exist and $a < c$.

Case $i = j$. This cannot occur since otherwise $x_i \leqslant_{\underline{L}} b$, because of $a <_i b$, and $x_i \not\leqslant_{\underline{L}} b$, because of $b <_i c$, a contradiction.

Overall we have shown that $a < c$ in any case and therefore $<$ is a transitive relation.

Finally let $a, b \in L$ with $a \neq b$. Then because $\{ x_i \mid i \in I \}$ is a generating set, the set $\Delta_{a,b}$ is not empty, since otherwise $a = b$. With $i := \min \Delta_{a,b}$ we either have $a <_i b$ if $x_i \leqslant_{\underline{L}} b$ and $b <_i a$ otherwise. ▫

3.3 Lemma *Let $a, b \in L$ with $a \leqslant_{\underline{L}} b$. Then $a \leqslant b$. In particular, if $a \leqslant_{\underline{L}} c$ and $b \leqslant_{\underline{L}} c$ for $a, b, c \in L$, then $a \vee b \leqslant c$.*

Proof We show $x_i \leqslant_{\underline{L}} a \implies x_i \leqslant_{\underline{L}} b$ for all $i \in I$. This shows $b \not< a$, hence $a \leqslant b$ by Lemma 3.2. Now if $x_i \leqslant_{\underline{L}} a$, then because of $a \leqslant_{\underline{L}} b$ we see that $x_i \leqslant_{\underline{L}} b$ and the claim is proven. ▫

The next step towards a general notion of Next-Closure is to provide a generalization of $\oplus$.

3.4 Definition Let $a \in L$ and $i \in I$. Then define

$$a \oplus i := \bigvee_{\substack{j <_I i \\ x_j \leqslant_{\underline{L}} a}} x_j \vee x_i. \qquad \diamondsuit$$

With all these definitions at hand we are now ready to formulate and prove the promised generalization. For this, we generalize the proof of Next-Closure as it is given in [8, page 67].

3.5 Lemma *Let $a, b \in L$ and $i, j \in I$. Then the following statements hold:*

i) $a <_i b, a <_j c, i <_I j \implies c <_i b$.

ii) $a < a \oplus i$ *if* $x_i \not\leqslant_{\underline{L}} a$.

iii) $a <_i b \implies a \oplus i \leqslant b$.

iv) $a <_i b \implies a <_i a \oplus i$.

Proof i) It is $x_i \not\leqslant_{\underline{L}} a$ and due to $i <_I j$ we get $x_i \not\leqslant_{\underline{L}} c$ as well. Furthermore, $x_i \leqslant_{\underline{L}} b$ because of $a <_i b$. Now if there would exist a $k \in I$ with $k <_I i$ such that $x_k \leqslant_{\underline{L}} c, x_k \not\leqslant_{\underline{L}} b$, then $x_k \leqslant_{\underline{L}} a$ because of $k <_I i$ and $x_k \not\leqslant_{\underline{L}} a$ because of $k <_I i <_I j$, a contradiction. With the same argumentation a contradiction follows from the assumption that there exists a $k \in I, k <_I i$ with $x_k \not\leqslant_{\underline{L}} c, x_k \leqslant_{\underline{L}} b$. In sum we have shown $c <_i b$, as required.

ii) We have $x_i \not\leqslant_{\underline{L}} a$ and $x_i \leqslant_{\underline{L}} a \oplus i$. Furthermore, for $k \in I, k <_I i$ and $x_k \leqslant_{\underline{L}} a$ we have $x_k \leqslant_{\underline{L}} a \oplus i$ by definition. This shows $a < a \oplus i$.

iii) Let $a <_k b$ for some $k \in I$. Then $\bigvee_{j<_I k, x_j \leqslant_{\underline{L}} a} x_j \leqslant_{\underline{L}} b$ and $x_k \leqslant_{\underline{L}} b$, hence with Lemma 3.3 we get $a \oplus k \leqslant b$.

iv) Let $a <_i b$. Then $x_i \not\leqslant_{\underline{L}} a$ and with (ii) we get $a < a \oplus i$. By (iii), $a \oplus i \leqslant b$. If for $k \in I$, $k <_I i$ it holds that $x_k \leqslant_{\underline{L}} a \oplus i$ and $x_k \not\leqslant_{\underline{L}} a$, then we also have $x_k \leqslant_{\underline{L}} a \oplus i \leqslant_{\underline{L}} b$, i.e. $x_k \leqslant_{\underline{L}} b$, contradicting the minimality of i. □

3.6 Theorem (Next-Closure for Semilattices) *Let $a \in L$. Then the next element $a^+ \in L$ with respect to $<$, if it exists, is given by*

$$a^+ = a \oplus i$$

with $i \in I$ being maximal with $a <_i a \oplus i$.

Proof Let

$$a^+ = \min_< b \in L \mid a < b$$

be the next element after a with respect to $<$. Then $a <_i a^+$ for some $i \in I$ and by Lemma 3.5.iv we get $a <_i a \oplus i$ and with Lemma 3.5.iii we see $a \oplus i \leqslant a^+$, hence $a \oplus i = a^+$. The maximality of i follows from Lemma 3.5.i. □

To find the correct element $i \in I$ such that $a^+ = a \oplus i$ we can utilize Lemma 3.5.ii. Because of this result, only elements $i \in I$ with $x_i \not\leqslant_{\underline{L}} a$ have to be considered, a technique which is also known for the original form of Next-Closure.

However, to make the above theorem practical for enumerating the elements of a certain semilattice, one has to start with some element, preferably the smallest element in L with respect to $\leqslant$. This element must also be minimal in L with respect to $\leqslant_{\underline{L}}$, by Lemma 3.3. Since $\{ x_i \mid i \in I \}$ is a generating set of $\underline{L}$, and $a \leqslant a \vee b$ for all $a, b \in L$, the minimal elements of L with respect to $\leqslant_{\underline{L}}$ must be among the elements $x_i, i \in I$. So to find the first

element of $\underline{L}$ with respect to $\leqslant$, find all minimal elements in $\{ x_i \mid i \in I \}$ and choose the smallest element with respect to $\leqslant$ from them. But because of $x_i \leqslant x_j$ if and only if $j <_I i$, one just has to take the largest index j of all minimal elements among the x_i to find the smallest element in L with respect to $\leqslant$.

As a final remark for this section note that the set $\{ x_i \mid i \in I \}$ must always include the $\vee$-irreducible elements of $\underline{L}$. These are all those elements $a \in L$ that cannot be represented as a join of other elements, or, equivalently,

$$\{ b \in L \mid b <_{\underline{L}} a \} = \varnothing \quad \text{or} \quad \bigvee_{b <_{\underline{L}} a} b <_{\underline{L}} a.$$

It is also easy to see that the set of $\vee$-irreducible elements of $\underline{L}$ is also sufficient, i. e. that it is a generating set of $\underline{L}$.

4 Computing the Intents of a Formal Context

We have seen an algorithm that is able to enumerate the elements of a semilattice from a given generating set. We have also claimed that this is a generalization of Next-Closure, which we want to demonstrate in this section. Furthermore, we want to give another example of an application of this algorithm, namely the computation of the intents of a given formal context.

Firstly, let us reconstruct the original Next-Closure algorithm from Theorem 3.6 and the corresponding definitions. For this let M be a finite set and let c be a closure operator on $M = \{ 0, \ldots, n-1 \}$, say. We then apply Theorem 3.6 to the semilattice $\underline{P} = (c[\mathfrak{P}(M)], \vee)$. We immediately see that $\leqslant_{\underline{P}} = \subseteq$ and that $<_i$ is the usual lectic order on $\underline{P}$. Then the set

$$\{ c(\{ i \}) \mid i \in M \} \cup \{ c(\varnothing) \}$$

is a finite generating set of $\underline{P}$ and we can define $x_i := c(\{ i \})$ and $x_n := c(\varnothing)$, i.e. $I = \{ 0, \ldots, n \}$. For a closed set $A \subseteq M$ and $i \in I$ then follows

$$\begin{aligned}
A \oplus i &= \bigvee_{\substack{j<i \\ x_j \subseteq A}} x_i \vee x_i \\
&= c\Big(\bigcup_{\substack{j<i \\ x_j \subseteq A}} x_j \Big) \vee x_i \\
&= c\Big(\bigcup \{ c(\{ j \}) \mid j < i, j \in A \} \Big) \vee c(\{ i \}) \\
&= c(\{ j \mid j < i, j \in A \}) \vee c(\{ i \}) \\
&= c(\{ j \mid j < i, j \in A \} \cup \{ i \})
\end{aligned}$$

which is the original definition of $\oplus$ for Next-Closure. Furthermore, it is $A \oplus n = A$ since $c(\varnothing) \subseteq A$ for each closed set A. We therefore do not need to consider x_n when looking for the next closed set, and indeed, the only reason why $x_n = c(\varnothing)$ has been included is

that it is the smallest closed set in $\underline{P}$. All in all, we see that Next-Closure is a special case of Theorem 3.6.

However, for a closure operator c on a finite set M it seems more natural to consider the semilattice $\underline{P} = (c[\mathfrak{P}(M)], \cap)$, because the intersection of two closed sets of c again yields a closed set of c. One sees that $\leqslant_{\underline{P}} = \supseteq$. As a generating set we take the set of $\cap$-irreducible elements $\{ X_i \mid i \in G \}$ for some index set G. Let $A, B \in c[\mathfrak{P}(M)]$ and let $<_G$ be a linear ordering on G. Then $A < B$ if and only if there exists $i \in G$ such that

$$i = \min\{ j \in G \mid (X_i \supseteq A, X_i \not\supseteq B) \text{ or } (X_i \not\supseteq A, X_i \supseteq B) \} \quad \text{and} \quad X_i \supseteq B$$

and $\oplus$ is just given by

$$A \oplus i = \bigcap_{\substack{j <_G i \\ X_j \supseteq A}} X_j \cap X_i.$$

Now note that $\oplus$ does not need the closure operator c anymore. This means that if the computation of c is very costly and the $\cap$-irreducible elements (or a superset thereof) is known, this approach might be much more efficient. In general, however, it is not known how to efficiently determine the $\cap$-irreducible closed sets of c. But if c is given as the $\cdot''$ operator of a formal context, these irreducible elements can be determined quickly [8]. We shall describe this idea in more detail.

Let G and M be two finite sets and let $J \subseteq G \times M$. We then call the triple $\mathbb{K} := (G, M, J)$ a *formal context*, G the *objects* of the formal context and M the *attributes* of the formal context. For $g \in G$ and $m \in M$ we write $g \; J \; m$ for $(g, m) \in J$ and say that object g *has* attribute m.

Let $A \subseteq G$ and $B \subseteq M$. We then define the *derivations* of A and B to be

$$A' := \{ m \in M \mid \forall g \in A : g \; J \; m \}$$
$$B' := \{ g \in G \mid \forall m \in B : g \; J \; m \}.$$

Then the $\cdot''$ operator is just the twofold derivation of a given set of attributes. It turns out that this is indeed a closure operator, and that every closure operator can be represented as a $\cdot''$ operator of a suitable formal context [4, 8]. The closed sets of $\cdot''$, i.e. all sets $B \subseteq M$ with $B = B''$, are called the *intents* of $\mathbb{K}$ and shall be denoted by $\mathrm{Int}(\mathbb{K})$. It is clear from the previous remarks that $(\mathrm{Int}(\mathbb{K}), \cap)$ is a semilattice.

The advantage of representing a closure operator is that the $\cap$-irreducible elements of $(\mathrm{Int}(\mathbb{K}), \cap)$ can be directly read off from the format context. As discussed in [8], the set

$$\{ \{ g \}' \mid g \in G \}$$

contains the irreducible elements we are looking for, except M (note that the order relation on $(\mathrm{Int}(\mathbb{K}), \cap)$ is $\supseteq$.) Furthermore, it is possible to omit certain objects g from $\mathbb{K}$ without changing $\mathrm{Int}(\mathbb{K})$. Every object $g \in G$ can be omitted from $\mathbb{K}$ for which the set $\{ g \}'$ is either equal to M or can be represented as a proper intersection of other sets $\{ g_1 \}', \ldots, \{ g_n \}'$ for some elements $g_1, \ldots, g_n \in G$. It is also clear that if there exist two distinct objects g_1 and g_2 with $\{ g_1 \}' = \{ g_2 \}'$, that we can remove one of them without changing $\mathrm{Int}(\mathbb{K})$. A formal context for which no such objects exist is called *object clarified* and *object reduced*.

Listing 1: Compute the Next Intent of a Formal Context

```
define next-intent(𝕂 = (G, M, J), A)
  for g ∈ G, descending
    if {g}′ ⋣ A then
      let (B := A ⊕ g)
        if ∀h ∈ G, h <_G g, {h}′ ⋣ A : {h}′ ⋣ B then
          return B
        end if
      end let
    end if
  end for
  return nil
end
```

If $\mathbb{K} = (G, M, J)$ is an object clarified and object reduced formal context, then the set $\{ \{g\}' \mid g \in G \}$ is exactly the set of $\cap$-irreducible intents of $\mathbb{K}$, except for the set M.

The algorithm described above now takes the following form when applied to the semilattice $(\mathrm{Int}(\mathbb{K}), \cap)$. As index set we choose the set G of object of the given formal context, ordered by $<_G$. For every object $g \in G$ we set $x_g := \{g\}'$. Then the set $\{ x_g \mid g \in G \}$ is a generating set of the semilattice $(\mathrm{Int}(\mathbb{K})\setminus\{M\}, \cap)$, which we want to enumerate (since we get the set M for free). For A being an intent of $\mathbb{K}$ and $g \in G$ we have

$$A \oplus g := \bigcap_{\substack{\{h\}' \supseteq A \\ h <_G g}} \{h\}' \cap \{g\}'$$

and as the first intent we take M. For an intent $A \subseteq M$ of $\mathbb{K}$ we then have to find the maximal object $g \in G$ (with respect to $<_G$) such that $A <_g A \oplus g$. This is equivalent to g being maximal with $\{g\}' \not\supseteq A$ and $\forall h \in G, h <_G g : \{h\}' \supseteq A \iff \{h\}' \supseteq A \oplus g$. However, the direction "$\Longrightarrow$" is clear, hence we only have to ensure

$$\forall h \in G, h <_G g : \{h\}' \not\supseteq A \implies \{h\}' \not\supseteq A \oplus g.$$

All these considerations yield the algorithm shown in Listing 1. Of course, the derivations of the form $\{g\}'$ should not be computed every time they are needed but rather stored somewhere for reuse.

Let us simplify Listing 1 a bit further. If A is an intent of $\mathbb{K}$, then for $g \in G$ it is true that

$$\{g\}' \not\supseteq A \iff g \notin A'.$$

This is because

$$\begin{aligned} \{g\}' \supseteq A &\implies \{g\}'' \subseteq A' \\ &\implies g \in A' \end{aligned}$$

Listing 2: Simplified version of Listing 1

```
0  define next-intent(𝕂 = (G, M, I), A)
1    for g ∈ G, descending
2      if g ∉ A′ then
3        let (B := A ⊕ g)
4          if B′ ∩ G_g ⊆ A′ ∩ G_g then
5            return B
6          end if
7        end let
8      end if
9    end for
10   return nil
11 end
```

and

$$\begin{aligned} g \in A' &\implies \{g\} \subseteq A' \\ &\implies \{g\}' \supseteq A'' = A, \end{aligned}$$

and therefore $\{g\}' \supseteq A \iff g \in A'$, i. e. $\{g\}' \not\supseteq A \iff g \notin A'$. Using this, we can simplify the condition

$$\forall h \in G, h <_G g, \{h\}' \not\supseteq A : \{h\}' \not\supseteq B$$

to

$$\forall h \in G, h <_G g : h \in B' \implies h \in A'$$

or equivalently to $B' \cap G_g \subseteq A' \cap G_g$, where $G_g := \{h \in G \mid h <_G g\}$. This leads to the algorithm of Listing 2.

Curiously enough, this algorithm is now very similar to Close-by-One. To make this similarity more apparent, let us give a brief description of the algorithm Close-by-One. In contrast to Next-Closure, which enumerates the intents of the formal context $\mathbb{K}$, Close-by-One enumerates the *formal concepts* of $\mathbb{K}$. These are pairs (C, D) of sets $C \subseteq G, D \subseteq M$ such that $C' = D$ and $D' = C$. The set of all formal concepts of $\mathbb{K}$ is denoted by $\mathfrak{B}(\mathbb{K})$. The formal concepts $\mathfrak{B}(\mathbb{K})$ of $\mathbb{K}$ can be ordered by

$$(C_1, D_1) \leqslant (C_2, D_2) \iff C_1 \subseteq C_2 (\iff D_2 \subseteq D_1).$$

It turns out that the set $(\mathfrak{B}(\mathbb{K}), \leqslant)$ forms a *(complete) lattice*, i. e. an ordered set such that for each set of formal concepts there exists a smallest upper and a largest lower bound. This lattice is also called the *concept lattice* of $\mathbb{K}$.

Close-by-One now performs a depth-first search in this lattice to compute all formal concepts of $\mathbb{K}$. Following the description of [10], the main part of the work is done by the

Listing 3: Close-by-One [10]

```
0  define generate-from(𝕂 = (G, M, J), (C, D), ℓ)
1    output (C, D)
2
3    if D = M or ℓ > |M| then
4      return
5    end if
6
7    for m ∈ {ℓ, ..., n}, ascending
8      if m ∉ D then
9        let (E := C ∩ {m}′,
10            F := E′)
11         if F ∩ M_m ⊆ D ∩ M_m then
12           generate-from((E, F), m + 1)
13         end if
14       end let
15     end if
16   end for
17 end
18
19 define all-concepts(𝕂 = (G, M, J))
20   generate-from(𝕂, (∅′, ∅″), 1)
21 end
```

function `generate-from`, as it is described in Listing 3. To simplify the description of this function, we again assume that the set M of attributes of $\mathbb{K}$ is just the set $M = \{1, \ldots, n\}$. Furthermore, if $m \in M$, we define $M_m := \{1, \ldots, m-1\}$.

We can now spot some similarities between the functions `next-intent` from Listing 2 and `generate-from`. The most striking similarity to note is the occurrence of the same tests in both functions: "$B' \cap G_g \subseteq A' \cap G_g$" in line 4 of Listing 2, and "$F \cap M_m \subseteq D \cap M_m$" in line 11 of Listing 3. If we substitute the definitions of the involved variables, these tests get the following form:

$$\begin{array}{ll} \texttt{next-intent:} & (A \cap \{g\}')' \cap G_g \subseteq A' \cap G_g \\ \texttt{generate-from:} & (C \cap \{m\}')' \cap M_m \subseteq C' \cap M_m \end{array}$$

So except that we consider sets of objects in the function `next-intent` and sets of attributes in the function `generate-from`, both test conditions are the same.

Still, the functions `next-intent` and `generate-from` are very different in how they compute the next intent and formal concept, respectively. While `next-intent` computes for a given intent just a new one, the function `generate-from` performs a depth-first search and enumerate *all* formal concepts of $\mathbb{K}$.

5 Conclusion

We have seen a natural generalization of the Next-Closure algorithm to enumerate elements of a semilattice from a generating set. We have proven the algorithm to be correct and applied it to the standard task of computing the intents of a given formal context, yielding a another algorithm to accomplish this. This algorithm turned out to have a lot of similarity to Close-by-One.

There are still some interesting ideas one might want to look at.

Firstly, a variation of the original Next-Closure algorithm is able to compute the canonical base of a formal context, a very compact representation of its implicational knowledge. It would be interesting to know whether the generalization given in this paper gives more insight into the computation, and therefore into the nature of the canonical base. This is, however, quite a vague idea.

Secondly, the algorithm discussed above to compute the intents of a formal context enumerates them in a certain order, which might or might not be a lectic one. Understanding this order relation might be fruitful, especially with respect to the complexity results obtained recently which consider enumerating *pseudo-intents* of a formal context in lectic order [5].

Thirdly, there exists a variant of the Next-Closure algorithm that is able to compute intents of a formal context *up to symmetry* [8, Theorem 51]. Given a set Γ of *automorphisms* of a formal context, this variant is able to compute for each orbit of intents under Γ exactly one representative. It would be interesting to know whether we can find a variant of our generalized Next-Closure algorithm that accomplishes the same thing.

Finally, and more technically, it might be interesting to look for other applications of our general algorithm. As already mentioned in the introduction, the enumeration of fuzzy concepts of a fuzzy formal context might be worth investigating (and it might be interesting comparing it to [2]).

Acknowledgments

The author is grateful for the useful comments of the anonymous reviewers, especially for pointing out the similarity of Listing 1 to Close-by-One, which the author had overlooked.

References

[1] Simon Andrews. In-Close, a fast algorithm for computing formal concepts. In Sebastian Rudolph, Frithjof Dau, and Sergei O. Kuznetsov, editors, *ICCS Supplementary Proceedings*, Moscow, 2009.

[2] Radim Belohlávek, Bernard De Baets, Jan Outrata, and Vilém Vychodil. Computing the Lattice of All Fixpoints of a Fuzzy Closure Operator. *IEEE T. Fuzzy Systems*, 18(3):546–557, 2010.

[3] Radim Belohlávek and Vilém Vychodil. Attribute Implications in a Fuzzy Setting. In Rokia Missaoui and Jürg Schmid, editors, *ICFCA*, volume 3874 of *Lecture Notes in Computer Science*, pages 45–60. Springer, 2006.

[4] Daniel Borchmann. Decomposing Finite Closure Operators by Attribute Exploration. In *ICFCA 2011, Supplementary Proceedings*, 2011.

[5] Felix Distel. Hardness of Enumerating Pseudo-intents in the Lectic Order. In Léonard Kwuida and Baris Sertkaya, editors, *ICFCA*, volume 5986 of *Lecture Notes in Computer Science*, pages 124–137. Springer, 2010.

[6] Bernhard Ganter. Two basic algorithms in concept analysis. FB4-Preprint Nr. 831, 1984.

[7] Bernhard Ganter and Klaus Reuter. Finding all closed sets: A general approach. *Order*, 8(3):283–290, 1991.

[8] Bernhard Ganter and Rudolph Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin-Heidelberg, 1999.

[9] Sergei O. Kuznetsov. A fast algorithm for computing all intersections of objects in a finite semi-lattice. *Automatic Documentation and Mathematical Linguistics*, 27:11–21, 1993.

[10] Jan Outrata and Vilém Vychodil. Fast algorithm for computing fixpoints of galois connections induced by object-attribute relational data. *Inf. Sci.*, 185(1):114–127, 2012.

[11] Vilém Vychodil, Petr Krajča, and Jan Outrata. Advances in algorithms based on CbO. In Marzena Kryszkiewicz and Sergei Obiedkov, editors, *Concept Lattices and Their Application*, pages 325–337, 2010.