Depth-first Search for Pseudo-intents through Minimal Transversals

Alexandre Bazin

contact@alexandrebazin.com

Abstract. The enumeration of pseudo-intents is a long-standing problem in which the order plays a major role. In this paper, we present new algorithms that enumerate pseudo-intents in orders that do not necessarily respect the inclusion relation. We show that, confirming established results, this enumeration is equivalent to computing minimal transversals in hypergraphs a bounded number of times.

1 Introduction

Formal concept analysis, as a field of applied mathematics, is interested in the patterns that can be found in data taking the form of a set of objects described by attributes. Among these patterns are the *implications*, relations between attribute sets A and B representing the fact that every object described by A is also described by B. As often with this type of relation, interest revolves around a small, non redundant set of implications. Such a set, the *Duquenne-Guigues* basis, is constructed using the notion of *pseudo-intent*. Indeed, computing the Duquenne-Guigues basis, and thus all the implications in a data set, is equivalent to enumerating all the pseudo-intents. In [15], it is shown that the number of pseudo-intents can be exponential in the number of attributes. It has been proven in [6, 2] that pseudo-intents cannot be enumerated with a polynomial delay in lectic or reverse lectic order. However, to the best of our knowledge, no such result exists for other orders. The best-known algorithm, Next Closure [11], enumerates only in the lectic order. In a previous work [4], we proposed an exponential delay algorithm to enumerate in any order that extends the inclusion order. In order to continue the study of the influence of the order on the complexity of this enumeration problem, we propose here two new algorithms that, together, can enumerate pseudo-intents without respecting the inclusion.

Our algorithms enumerate pseudo-intents incrementally, i.e. given a formal context and a set of previously found implications, they return a new pseudo-intent. In order to do this, we define new conditions under which a pseudo-intent P can be recognized using the lower covers of P in the lattice of attribute sets closed under the implications already found. This allows us to build algorithms that, instead of starting from $\emptyset$ and adding attributes to construct sets, start from $\top$ and remove attributes, resulting in a depth-first search in the lattice. We show that both constructing and recognizing a pseudo-intent can be done

Local Proceedings also appeared in ISBN 978-84-606-7410-8, Universidad de Málaga (Dept. Matemática Aplicada), Spain.

by computing a bounded number of times the minimal transversals of some hypergraph, which makes it easy to bound the delay and isolate special, easier cases. This result establishes a link with the proof in [6] that enumerating pseudo-intents without order is at least as hard as computing minimal transversals.

We start by recalling in Section 2 the basic definitions of FCA and results on the enumeration of pseudo-intents and minimal transversals. In Section 3, we present the definition of a recognizable pseudo-intent that we use along with the properties needed for our algorithms. Finally, Sections 4 and 5 are dedicated to the two algorithms, an example and a brief study of their delay.

2 Preliminaries

2.1 Basics

In formal concept analysis, data takes the form of a *formal context*. A formal context is a triple $\mathcal{C} = (\mathcal{O}, \mathcal{A}, \mathcal{R})$ in which $\mathcal{O}$ is a set of objects, $\mathcal{A}$ is a set of attributes and $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$ a relation that maps attributes to objects. An object $o \in \mathcal{O}$ is said to be *described* by an attribute $a \in \mathcal{A}$ if $(o, a) \in \mathcal{R}$. Together with the context, there are two $\cdot'$ operators defined as

$$\cdot' : 2^{\mathcal{O}} \mapsto 2^{\mathcal{A}}$$

$$O' = \{a \in \mathcal{A} \mid \forall o \in O,\ (o, a) \in \mathcal{R}\}$$

$$\cdot' : 2^{\mathcal{A}} \mapsto 2^{\mathcal{O}}$$

$$A' = \{o \in \mathcal{O} \mid \forall a \in A,\ (o, a) \in \mathcal{R}\}$$

Their composition $\cdot''$ is a closure operator. A set of attributes $A = A''$ closed under $\cdot''$ is called an *intent*.

2.2 Implications

An *implication* is a relation between two attribute sets A and B, noted $A \to B$. It is said to *hold* in the context if and only if $A' \subseteq B'$ or, in other words, if and only if every object described by A is also described by B. A set $\mathcal{I}$ of implications is a *basis* if and only if every implication that holds in the context can be derived from $\mathcal{I}$ using Armstrong's rules :

$$\frac{B \subseteq A}{A \to B}, \quad \frac{A \to B}{A \cup C \to B \cup C}, \quad \frac{A \to B, B \to C}{A \to C}$$

A *pseudo-intent* (or *pseudo-closed set*) P is a set of attributes that contains the closure of all its subsets and is not an intent. The set $\mathcal{B} = \{P \to P'' \mid P$

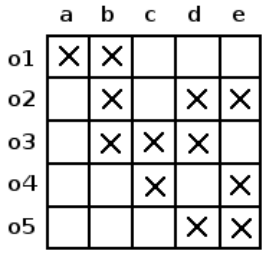

Fig. 1. A formal context

is a pseudo-intent} is the implication basis with the minimum cardinality and is called the *Duquenne-Guigues basis* [12]. As such, computing the Duquenne-Guigues basis, and thus all the implications, is enumerating all the pseudo-intents. As one of the great problems in FCA, it has been abundantly studied [12, 15, 6, 17, 4, 3, 16, 1, 19]. As the number of pseudo-intents can be exponential in the size of the context, preventing a polynomial algorithm, the attention is instead focused on the delay, i.e. the time between two pseudo-intents and the last pseudo-intent and the end of the algorithm. It has been shown that pseudo-intents cannot be enumerated with a polynomial delay in lectic [6] or reverse lectic order [2]. For the enumeration without order, it is shown in [6] that it is at least as hard as computing the minimal transversals of a hypergraph. However, the upper bound is not yet known. The problem of recognizing a pseudo-intent has been studied in [1] and found to be coNP-complete.

2.3 Logical Closures

A set of implications $\mathcal{I}$ gives rises to a *logical closure* operator $\mathcal{I}(\cdot)$ defined as

$$A^+ = A \cup \{C \mid B \to C \in \mathcal{I} \text{ and } B \subseteq A\}$$
$$\mathcal{I}(A) = A^{++\cdots+} \text{ (up to saturation)}$$

Similarly, we have the *logical pseudo-closure* operator $\mathcal{I}^-(\cdot)$ defined as

$$A^\circ = A \cup \{C \mid B \to C \in \mathcal{I} \text{ and } B \subset A\}$$
$$\mathcal{I}^-(A) = A^{\circ\circ\cdots\circ} \text{ (up to saturation)}$$

It is known that intents are closed under both $\mathcal{B}^-(\cdot)$ and $\mathcal{B}(\cdot)$, whereas a pseudo-intent P is closed under $\mathcal{B}^-(\cdot)$ and $\mathcal{I}(\cdot)$ for $\mathcal{I} \subseteq \mathcal{B} \setminus \{P \to P''\}$.

An attribute set Y is said to be a *generator* of an attribute set X under $\mathcal{I}(\cdot)$ (resp. $\mathcal{I}^-(\cdot)$) if $\mathcal{I}(Y) = X$ (resp. $\mathcal{I}^-(Y) = X$). It is a *minimal generator*

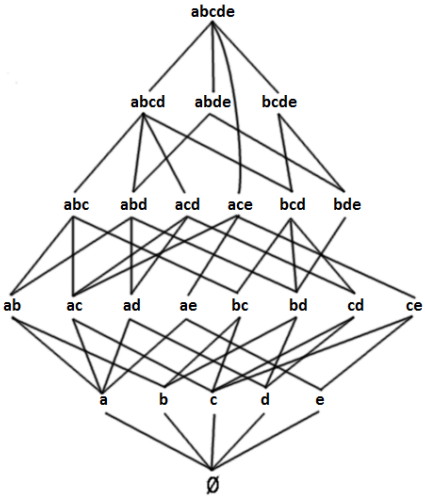

Fig. 2. $\Phi_{\mathcal{I}}$ with $\mathcal{I} = \{de \rightarrow bde, be \rightarrow bde\}$

if there is no generator Z of X such that $Z \subset Y$. We will use $Gen_{\mathcal{I}}(X)$ to denote the set of all minimal generators of a set X under $\mathcal{I}(\cdot)$. The complexity of computing the minimal generators of a set has been studied in [14]. The number of minimal generators can be exponential in the number of attributes and they can be enumerated with an incremental polynomial delay.

We shall use $\Phi_{\mathcal{I}}$ to denote the lattice of attribute sets closed under $\mathcal{I}(\cdot)$ ordered by the inclusion relation. For an attribute set A, the set of pseudo-intents P in $\mathcal{I}$ such that $P \subseteq A$ and $P'' \not\subseteq A$ will be denoted by $\mathcal{P}_{\mathcal{I}}(A)$.

2.4 Minimal Transversals

The problem of computing *minimal transversals* in hypergraphs or, equivalently, the prime conjunctive normal form of the dual of a Boolean function (otherwise called *monotone dualization*), have been largely studied in the literature [13, 10, 8, 9, 7].

Given a hypergraph $\mathcal{H} = (E, \mathcal{V})$ where E is a set of edges and $\mathcal{V} \subseteq 2^E$ is a set of vertices, a *transversal* $T \subseteq E$ is a set of edges such that $\forall V \in \mathcal{V}, T \cap V \neq \emptyset$. A transversal is *minimal* if none of its subsets is a transversal. We shall use $\mathcal{T}r(V)$ to denote the set of minimal transversals of a set of vertices V. For example, $\mathcal{T}r(\{ace, bce, cde\}) = \{c, e, abd\}$.

The problem of computing all the minimal transversals of a hypergraph can be solved in quasi-polynomial total time [9, 10, 13]. It is currently not known

whether it is possible to enumerate the transversals with a polynomial delay in the general case. However, many special cases are known to have an output-polynomial delay. Their references can be found in [9].

In this work, we cannot presuppose the nature of the hypergraph and so we are interested in the general case. The problem still being actively studied, we will suppose that we have a blackbox algorithm nextTrans that enumerates the minimal transversals of a set of vertices X. We suppose, for ease of writing, that the transversals T_i this algorithm enumerates in some arbitrary order $T_1 < T_2 < ... < T_n$ with $\text{nextTrans}(T_i, X) = T_{i+1}$ and $\text{nextTrans}(T_n, X) = \emptyset$. For example, Berge Multiplication [5] can be adapted to take the role of nextTrans. It is important to note that a first transversal can be computed in polynomial time by simply removing attributes until the set stops being a transversal.

3 Recognizing a Pseudo-Intent

We know that an attribute set is a pseudo-intent if and only if it contains the closure of all its subsets that are pseudo-intents. As for the problem of recognizing pseudo-intents, two cases have been studied. When only a context and a set A are provided, checking whether A is a pseudo-intent is coNP-complete [1]. An algorithm has been proposed in [19] that runs in $O(2^{|\mathcal{A}|})$. When the context and A are provided alongside all the pseudo-intents (and thus implications) contained in A, checking whether A is a pseudo-intent is easier as we only have to verify the closedness of A for the logical closure and $\cdot''$, which implies a runtime polynomial in the size of the implication set. It is this method that is used in algorithms such as Next Closure or the one we proposed in [4] and its drawback is that it forces an enumeration in an order that extends the inclusion order.

In this paper, we are interested in enumerating pseudo-intents in orders that do not extend the inclusion and, as such, we cannot suppose that we know the closure of all the subsets of a set before attempting to recognize its pseudo-closedness. Hence, we propose to consider the problem of recognizing a pseudo-intent A given a context and a set of subsets of A that is not necessarily complete.

Proposition 1. *An attribute set $A \in \Phi_{\mathcal{I}}$ is a pseudo-intent if $A \neq A''$ and all its lower covers in $\Phi_{\mathcal{I}}$ are closed.*

Proof. Let us suppose that A is not closed and that all its lower covers are closed. Let B be a lower cover of A and X be a subset of A. If $B \subset X$, then $\mathcal{I}(X) = A$ and $X'' = A''$. If $X \subseteq B$ and $X'' \not\subseteq B$, then B cannot be closed and we have a contradiction. Therefore, $X'' \subseteq B$. Since the closure of every subset of A is either A'' or contained in A, the set A is a pseudo-intent. □

This proposition states that some pseudo-intents can be recognized in $\Phi_{\mathcal{I}}$ only by looking at their lower covers. As such, when $\mathcal{I}$ is a subset of the Duquenne-Guigues basis, a new pseudo-intent can be found by looking in $\Phi_{\mathcal{I}}$ for sets that only have intents as lower covers.

Definition 1. *A pseudo-intent that can be recognized using Proposition 1 with a set of implication $\mathcal{I}$ is said to be recognizable. The set of all the implications that are recognizable with $\mathcal{I}$ is denoted by $Rec(\mathcal{I})$.*

Proposition 2. $\forall \mathcal{I} \subset \mathcal{B}, Rec(\mathcal{I}) \neq \emptyset$

Proof. Let P be minimal among pseudo-intents that are not premises of implications in $\mathcal{I}$. If there is a lower cover X of P in $\Phi_\mathcal{I}$ that is not closed, then there is a set $A \subset X$ in $\Phi_\mathcal{I}$ such that $A \rightarrow A''$ holds in the context. If $A'' \subseteq P$, then P is not minimal. If $A'' \not\subseteq P$, then P is not a pseudo-intent. Both cases lead to contradictions so all the lower covers of P are closed and P is recognizable from $\mathcal{I}$. Since we have $\mathcal{I} \subset \mathcal{B}$, the set of pseudo-intents that are not a premise of $\mathcal{I}$ is not empty, so it has minimal elements. Hence, the set of pseudo-intents that are recognizable from $\mathcal{I}$ is not empty if $\mathcal{I} \subset \mathcal{B}$. □

We have that, even though some unknown pseudo-intents may not be recognizable, there are always additional recognizable pseudo-intents for any subset of the Duquenne-Guigues basis. This ensures that we are able to enumerate all the pseudo-intents with an algorithm that finds recognizable ones.

Proposition 3. *Let A and B be two elements of $\Phi_\mathcal{I}$. The set B is a lower cover of A if and only if $B = A \setminus C$ with C a minimal transversal of $Gen_\mathcal{I}(A)$.*

Proof. Let C be minimal such that $\forall G \in Gen_\mathcal{I}(A), G \cap C \neq \emptyset$. For any attribute $i \in A \setminus C$, the set $(A \setminus C) \cup \{i\} = A \setminus (C \setminus \{i\})$ contains a minimal generator of A because of the minimality of C. Therefore, any B between $A \setminus C$ and A is such that $\mathcal{I}(B) = A$ and cannot be in $\Phi_\mathcal{I}$. Hence, $A \setminus C$ is a lower cover of A.

Let B be a lower cover of A in $\Phi_\mathcal{I}$. By definition, $\mathcal{I}(B \cup \{i\}) = A$ for any attribute $i \in A \setminus B$ so there is a subset C of A such that $i \in C$ and $(C \setminus \{i\}) \subseteq B$ that is a minimal generator of A. □

This proposition states that the lower covers of a set can be computed from and depend on its minimal generators. As such, the number of lower covers can be exponential in the number of minimal generators which can itself be exponential in the size of the attribute set [14].

Definition 2. *For any two attribute sets A and B, we say that B is reachable from A if and only if $B \subset A$ and there is an ordering $x_1 < x_2 < ... < x_n$ of the elements of $A \setminus B$ such that, for every $m < n$, $A \setminus \{x_1, x_2, ..., x_m\}$ is not closed under $\mathcal{I}$.*

In other words, B is reachable from A if you can obtain B by removing attributes from A and only go through sets that are not closed under $\mathcal{I}$. Evidently, minimal reachable sets are elements of $\Phi_\mathcal{I}$.

Proposition 4. *An attribute set $A \in \Phi_\mathcal{I}$ is a minimal recognizable pseudo-intent if and only if A is not an intent and all the minimal attributes sets reachable from A are intents.*

Algorithm 1 *reachableSets*(A)

Require: An attribute set A, an implication set $\mathcal{I}$
$R = \emptyset$
for every $i \in A$ **do**
  $G = \bigcup_{P \in \mathcal{P}_{\mathcal{I}}(A \setminus \{i\})} Gen_{\mathcal{I}}(P)$
  $C =$ The first transversal of G
  while $C \neq \emptyset$ **do**
    $B = A \setminus (\{i\} \cup C)$
    if $\mathcal{P}_{\mathcal{I}}(B) \neq \emptyset$ **then**
      $R = R \cup reachableSets(B)$
    else
      $R = R \cup \{B\}$
    end if
    $C = nextTrans(C, G)$
  end while
end for
Return R

Proof. If A is a minimal recognizable pseudo-intent, then every subset of A in $\Phi_{\mathcal{I}}$ is an intent. Thus every minimal reachable set is an intent.

Let us suppose that every minimal reachable subset of A is an intent and $A \neq A''$. For any set X reachable from A and any attribute set $P \subseteq X$, if $P'' \not\subset A$, then X cannot be an intent. Every minimal reachable subset of A being an intent, A contains the closure of all its subsets so it is a pseudo-intent. If P is a pseudo-intent that is not in $\mathcal{I}$, subsets of $A \setminus \{i\}$ with $i \in P'' \setminus P$ cannot be intents so either P or a superset of P in $\Phi_{\mathcal{I}}$ that is not an intent is reachable from A. Hence, A is a minimal recognizable pseudo-intent. □

A minimal recognizable pseudo-intent can therefore be recognized by computing the sets of its minimal reachable subsets. These minimal reachable sets can be computed using Algorithm 1. By definition, reachable sets can be computed by removing from A attributes that play a role in the logical closure, i.e. attributes in minimal generators of pseudo-intents $P \subset A$ such that $P'' \not\subseteq A \setminus X$ for some $A \setminus X$ reachable from A. The minimal transversals T make up all the possible ways to remove minimal generators of pseudo-intents and, thus, all the sets T such that for every $Y \subset T$, $A \setminus Y$ cannot be logically closed. As such, removing minimal transversals of the generators of pseudo-intents used in the logical closure of a set produces a reachable set. The set $A \setminus T$ is not always logically closed so the algorithm is recursively applied until an element of $\Phi_{\mathcal{I}}$ is reached.

The algorithm is not optimal as some reachable sets can be computed multiple times and the sets reachable from a set A that is not an element of $\Phi_{\mathcal{I}}$ could be computed directly from $\mathcal{P}_{\mathcal{I}}(A)$ instead of its direct subsets. However, it is sufficient to give us an upper bound of the runtime. For each recursive call, the algorithm computes $\mathcal{P}_{\mathcal{I}}(A \setminus \{i\})$ for every attribute $i \in A$. For each of these

attributes, it then computes the minimal transversals of $\bigcup_{P \in \mathcal{P}_{\mathcal{I}}(A \setminus \{i\})} Gen_{\mathcal{I}}(P)$. There is another recursive call for each pseudo-intent found missing so the runtime is bounded by $O(|\mathcal{A}| \times |\mathcal{I}| \times T)$ where T is the complexity of computing the transversals.

In the rest of this paper, we will suppose that we have an algorithm NEXTREACH that enumerates minimal reachable sets in the same fashion as NEXTTRANS.

4 Computing a Recognizable Pseudo-Intent

4.1 Algorithm

We want to incrementally compute the pseudo-intents in a formal context. In the beginning, we only have the formal context $(\mathcal{O}, \mathcal{A}, \mathcal{R})$ and an empty set of implications $\mathcal{I}$. The corresponding $\Phi_{\mathcal{I}}$ is the boolean lattice of subsets of $\mathcal{A}$. In order to find a first pseudo-intent, we can look for a recognizable set (Proposition 1) in $\Phi_{\mathcal{I}}$. We know that a non-closed set contains a pseudo-intent. Thus, we can start with $\mathcal{A}$ and recursively remove attributes as long as we obtain non-closed sets. When we cannot remove attributes without obtaining an intent, the set is a recognizable pseudo-intent and we have a first implication. This corresponds to the algorithm presented in [6] that computes a first pseudo-intent. We can then generalize the algorithm for other pseudo-intents. As $\mathcal{I}$ contains a new implication, sets disappear from $\Phi_{\mathcal{I}}$ and we now have to look in the new lattice for a second pseudo-intent. As it is not a Boolean lattice anymore, we cannot simply remove attributes to obtain lower covers so we have to use Proposition 3 to compute them. By going through the lattice using non-closed lower covers of sets until we find a set respecting Proposition 1, we can compute a second pseudo-intent. Pseudo-intents can thus be computed incrementally until $\mathcal{A}$ no longer has any non-closed lower cover. Algorithm 2, given a context, a subset $\mathcal{I}$ of the Duquenne-Guigue Basis and an attribute set $A \in \Phi_{\mathcal{I}}$, uses this method to compute a new pseudo-intent $P \subseteq A$.

Proposition 5. *$nextPI(\mathcal{A}, \mathcal{I})$ returns either $\mathcal{A}$ or a pseudo-intent that is not in $\mathcal{I}$.*

Proof. Let X be a set in $\Phi_{\mathcal{I}}$. If $X'' \neq X$, then there is a pseudo-intent $P \subseteq X$. If a lower cover of X is not closed, then there is a pseudo-intent $X \subset S$. The algorithm returns the first set it finds that has all its lower covers closed. If this set is not $\mathcal{A}$, then it is not closed and is thus a pseudo-intent. □

Algorithm 3 uses Algorithm 2 to enumerate pseudo-intents iteratively until it returns $\mathcal{A}$. It is sound but not complete, as exemplified below.

Algorithm 2 $nextPI(A, \mathcal{I})$

Require: Attribute set A, Implication set $\mathcal{I}$
1: $P = A$
2: $G = Gen_{\mathcal{I}}(A)$
3: $C =$ The first transversal of G
4: **while** $C \neq \emptyset$ **do**
5: $S = A \setminus C$
6: **if** $S'' \neq S$ **then**
7: $P = nextPI(S)$
8: $C = \emptyset$
9: **else**
10: $C = nextTrans(C, G)$
11: **end if**
12: **end while**
13: RETURN P

Algorithm 3 $allPI$

1: $\mathcal{I} = \emptyset$
2: $A = nextPI(A, \mathcal{I})$
3: **while** $A \neq \mathcal{A}$ **do**
4: $\mathcal{I} = \mathcal{I} \cup \{A \rightarrow A''\}$
5: $A = nextPI(A, \mathcal{I})$
6: **end while**
7: Return $\mathcal{I}$

4.2 Example

Let us consider an arbitrary context for which the Duquenne-Guigues basis is $\{de \rightarrow bde, bc \rightarrow bcd, ac \rightarrow abcd, be \rightarrow bde, bcde \rightarrow abcde\}$. We want to find its pseudo-intents using Algorithm 3 :

The set of implications is initially empty. We start with $abcde$.

It has a single minimal generator, itself. The first minimal transversal of $Gen(abcde)$ is a. The set $abcde \setminus a = bcde$ is not an intent so we continue recursively with it.

The set $bcde$ has a single minimal generator, itself. The first minimal transversal of $Gen(bcde)$ is b. The set $bcde \setminus b = cde$ is not an intent so we continue recursively with it.

The set cde has a single minimal generator, itself. The first minimal transversal of $Gen(cde)$ is c. The set $cde \setminus c = de$ is not an intent so we continue recursively with it.

The set de has a single minimal generator, itself. The first minimal transversal of $Gen(de)$ is d. The set $de \setminus d = e$ is an intent. The second and last minimal transversal is e. The set $de \setminus e = d$ is an intent. The algorithm returns de as the first pseudo-intent which gives us $\mathcal{I} = \{de \rightarrow bde\}$.

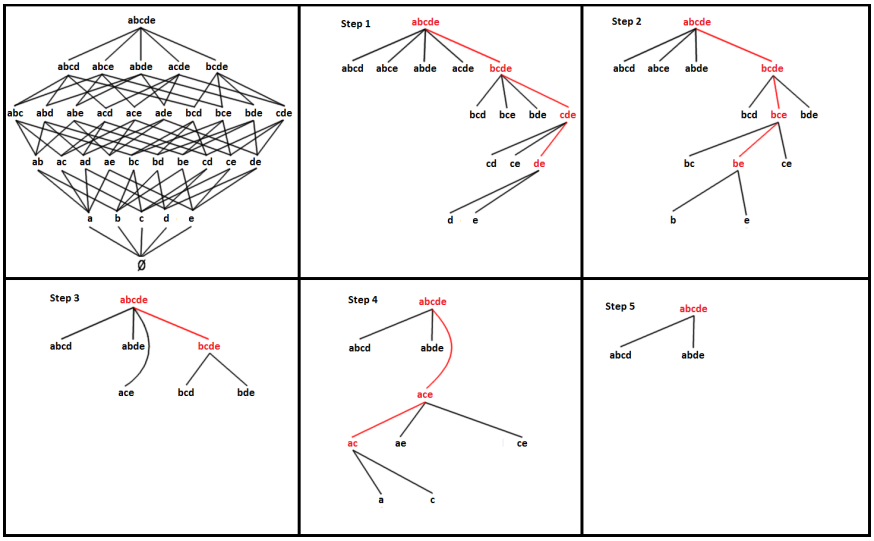

Fig. 3. The five steps of the algorithm. In red the sets considered by the recursive calls, in black the lower covers of those sets in $\Phi_{\mathcal{I}}$.

We start again with *abcde*. It has a single minimal generator, *acde*. The first minimal transversal of $Gen(abcde)$ is a. The set $abcde \setminus a = bcde$ is not an intent so we continue recursively with it.

The set *bcde* has a single minimal generator, *cde*. The first minimal transversal of $Gen(cde)$ is c. The set $bcde \setminus c = bde$ is an intent. The second minimal transversal is d. The set $bcde \setminus d = bce$ is not an intent so we continue recursively with it.

The set *bce* has a single minimal generator, itself. The first minimal transversal of $Gen(bce)$ is b. The set $bce \setminus b = ce$ is an intent. The second minimal transversal is c. The set $bce \setminus c = be$ is not an intent so we continue recursively with it.

The set *be* has a single minimal generator, itself. The first minimal transversal of $Gen(be)$ is b. The set $be \setminus b = e$ is an intent. The second and last minimal transversal is e. The set $be \setminus e = b$ is an intent. The algorithm returns *be* as the second pseudo-intent which gives us $\mathcal{I} = \{de \rightarrow bde, be \rightarrow bde\}$.

We start again with *abcde*. It has two minimal generators, *acde* and *abce*. The first minimal transversal of $Gen(abcde)$ is a. The set $abcde \setminus a = bcde$ is not an intent so we continue recursively with it.

The set *bcde* has two minimal generators, *bce* and *cde*. The first minimal transversal of $Gen(cde)$ is c. The set $bcde \setminus c = bde$ is an intent. The second

minimal transversal is bd. The set $bcde \setminus bd = ce$ is an intent. The third and last minimal transversal is e. The set $bcde \setminus e = bcd$ is an intent. The algorithm returns $bcde$ as the third pseudo-intent which gives us $\mathcal{I} = \{de \rightarrow bde, be \rightarrow bde, bcde \rightarrow abcde\}$.

We start again with $abcde$. It has two minimal generators, cde and bce. The first minimal transversal of $Gen(abcde)$ is bd. The set $abcde \setminus bd = ace$ is not an intent so we continue recursively with it.

The set ace has a single minimal generator, itself. The first minimal transversal of $Gen(ace)$ is a. The set $ace \setminus a = ce$ is an intent. The second minimal transversal is c. The set $ace \setminus c = ae$ is an intent. The third minimal transversal is e. The set $ace \setminus e = ac$ is not an intent so we continue recursively with it.

The set ac has a single minimal generator, itself. The first minimal transversal of $Gen(ace)$ is a. The set $ac \setminus a = c$ is an intent. The second and last minimal transversal is c. The set $ac \setminus c = a$ is intent. The algorithm returns ac as the fourth pseudo-intent which gives us $\mathcal{I} = \{de \rightarrow bde, be \rightarrow bde, bcde \rightarrow abcde, ac \rightarrow abcd\}$.

We start again with $abcde$. It has three minimal generators, ace, cde and bce. The first minimal transversal of $Gen(abcde)$ is c. The set $abcde \setminus c = abde$ is an intent. The second minimal transversal is e. The set $abcde \setminus e = abcd$ is an intent. The third and last minimal transversal is abd. The set $abcde \setminus abd = ce$ is an intent. The algorithm ends.

The algorithm has found four pseudo-intents but is unable to find the set bc when considering the minimal transversals, and thus the sets, **in that particular order**. Indeed, knowing $be \rightarrow bde$ and $de \rightarrow bde$ makes $bcde$ recognizable and blocks every path from $abcde$ to bc in $\Phi_{\mathcal{I}}$. Finding bc before either de or be solves the problem.

4.3 Complexity

The nature of Algorithm 2 makes it easy to bound the delay between finding two pseudo-intents. As the algorithm starts from $\mathcal{A}$ and removes attributes until it ends, it performs a maximum of $|\mathcal{A}|$ recursive calls before finding a pseudo-intent. In a call, three different tasks are performed : computing the closure of a set, computing the set of minimal generators and computing the set of all lower covers. The closure of a set is known to be computable in polynomial time. Computing $Gen_{\mathcal{I}}(A)$ is done in time polynomial in the size of the output. The computation of all the lower covers of A, as we have seen in Section 2.4, can be performed in quasi-polynomial total time, i.e. in time quasi-polynomial in the size of $n = |Gen_{\mathcal{I}}(A)|$ and $m = |\mathcal{T}r(Gen_{\mathcal{I}}(A))|$ (note that the size of m itself is bounded by $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ [18]). Hence, the delay is in $O(|\mathcal{A}| \times (C + G + T))$ with C the complexity of computing a closure, G the complexity of computing minimal generators and T the complexity of computing the lower covers.

Algorithm 4 $nextMinPI(A, \mathcal{I})$

Require: Attribute set A, Implication set $\mathcal{I}$
1: $P = A$
2: $C =$The first minimal set reachable from A
3: **while** $C \neq \emptyset$ **do**
4: **if** $C'' \neq C$ **then**
5: $P = nextMinPI(C)$
6: $C = \emptyset$
7: **else**
8: $C = nextReach(C, A)$
9: **end if**
10: **end while**
11: RETURN P

An interesting point of detail is that the algorithm does not necessarily enumerate all the intents. When all the upper covers of an intent A in $\Phi_{\mathcal{I}}$ are intents themselves, A cannot be reached by the algorithm anymore. In particular, if $A \cup \{i\} = (A \cup \{i\})''$ for every $i \in \mathcal{A} \setminus A$, we are sure that A will never be considered. In the previous example (see Figure 3), we see that the intents abd, ab, ad, bd, de and $\emptyset$ are never considered because they do not appear as lower covers of sets. We suppose that this property helps reduce the runtime on dense contexts with a high number of intents.

5 Computing a Minimal Recognizable Pseudo-Intent

5.1 Algorithm

The problem with Algorithm 3 is that there are some pseudo-intents it cannot compute. Among those leftover pseudo-intents, there is necessarily at least one that is minimal. We have shown in Proposition 4 that a minimal recognizable pseudo-intent P can be recognized through the minimal sets reachable from P. Those reachable sets can themselves be obtained by computing the transversals of the sets of minimal generators of the pseudo-intents involved in the logical closure of the different $P \setminus \{i\}$ (Algorithm 1). Algorithm 4 uses these properties to compute a minimal recognizable pseudo-intent. In a manner similar to that of Algorithm 2, the algorithm starts with $\mathcal{A}$ and enumerates the minimal reachable sets. Once one that is not an intent is found, the algorithm is recursively applied. The algorithm ends when it finds a set with only intents as minimal reachable sets. The main difference between Algorithm 2 and Algorithm 4 is that the former performs a depth-first search in the lattice $\Phi_{\mathcal{I}}$ while the latter performs it in the directed graph in which the nodes are the elements of $\Phi_{\mathcal{I}}$ and the edges are the pairs in the "reachable" relation.

Proposition 6. *Algorithm 4 returns either $\mathcal{A}$ or a minimal pseudo-intent that is not in $\mathcal{I}$.*

Algorithm 5 *allPI2*

1: $\mathcal{I} = \emptyset$
2: $A = nextminPI(A, \mathcal{I})$
3: **while** $A \neq \mathcal{A}$ **do**
4: $\quad \mathcal{I} = \mathcal{I} \cup \{A \rightarrow A''\}$
5: $\quad A = nextMinPI(A, \mathcal{I})$
6: **end while**
7: Return $\mathcal{I}$

Proof. From Proposition 4 we know that if every minimal set $T \in \Phi_{\mathcal{I}}$ reachable from $A \in \Phi_{\mathcal{I}}$ is closed and A is not, then A is a minimal recognizable pseudo-intent. If $T \in \Phi_{\mathcal{I}}$ is not closed, then T contains a minimal recognizable pseudo-intent so the algorithm can be recursively called on T. The algorithm stops when it encounters a set A with only intents as minimal reachable sets. The set $\mathcal{A}$ being the only intent on which the algorithm is called, it returns either $\mathcal{A}$ or a minimal recognizable pseudo-intent. □

We can enumerate all the pseudo-intents with Algorithm 5 using Algorithm 4 in the same fashion as Algorithm 3.

Proposition 7. *Algorithm 5 is sound and complete.*

Proof. Soundness follows from Proposition 6. For any implication set $\mathcal{I} \subset \mathcal{B}$, pseudo-intent P not used in $\mathcal{I}$ and attribute set $A \in \Phi_{\mathcal{I}}$ such that $P \subset A$, no set X such that $P \subset X$ and $P'' \not\subseteq X$ can be an intent so either P or one of its non-closed supersets is reachable from A. Hence, the algorithm does not stop until every pseudo-intent has been found. □

5.2 Example

Contrary to Algorithm 3, Algorithm 5 is complete. We can either use it to compute all the pseudo-intents or combine it with Algorithm 3. Let us suppose we have used Algorithm 3, as exemplified in Section 4, and that we know the set of implications $\mathcal{I} = \{de \rightarrow bde, be \rightarrow bde, bcde \rightarrow abcde, ac \rightarrow abcd\}$. Using Algorithm 5 from there gives us the following :

We start with $abcde$. The first minimal reachable set $abcde \setminus (a \cup bd) = ce$ is an intent. The second minimal reachable set $abcde \setminus (a \cup bc) = de$ is an intent. The third minimal reachable set $abcde \setminus (a \cup e) = bcd$ is an intent. The fourth minimal reachable set $abcde \setminus (b \cup cd) = ae$ is an intent. The fifth minimal reachable set $abcde \setminus (b \cup e) = cd$ is an intent. The sixth minimal reachable set $abcde \setminus (b \cup ce) = ad$ is an intent. The seventh minimal reachable set $abcde \setminus c = abde$ is an intent. The eighth minimal reachable set $abcde \setminus (d \cup ab) = ce$ is an intent. The ninth minimal reachable set $abcde \setminus (d \cup ae) = bc$ is not an intent so we continue recursively with it.

The set bc contains no pseudo-intents so its minimal reachable sets are $bc \backslash b = c$ and $bc \setminus c = b$. They are both intents so the algorithm returns bc as a new pseudo-intent which gives us $\mathcal{I} = \{de \rightarrow bde, be \rightarrow bde, bcde \rightarrow abcde, ac \rightarrow abcd, bc \rightarrow bcd\}$.

We start again with $abcde$. The eight first minimal reachable sets computed are the same as in the previous step. The ninth reachable set $abcde \setminus cde = ab$ is an intent. The tenth and last reachable set $abcde \setminus e = abcd$ is an intent. The algorithm ends.

In this example, the pseudo-intent bc has been found after $bcde$. Using both Algorithm 3 and Algorithm 5, it is thus possible to enumerate pseudo-intents in orders that do not extend the inclusion.

5.3 Complexity

Once again, the delay between two pseudo-intents with Algorithm 5 is easy to bound. Algorithm 4 performs at most $|\mathcal{A}|$ recursive calls before returning a pseudo-intent or $\mathcal{A}$. Each call requires the computation of, at most, all the reachable sets in $\Phi_{\mathcal{I}}$. We have shown that this can be done in $O(|\mathcal{A}| \times |\mathcal{I}| \times T)$ with T the complexity of computing minimal transversals of minimal generators of pseudo-intents. Thus, the worst case delay of Algorithm 5 is $|\mathcal{I}|$ times that of Algorithm 3.

A first minimal transversal can be computed in polynomial time. As such, for any attribute set A, we can compute its first reachable set in $\Phi_{\mathcal{I}}$ in time polynomial in $|\mathcal{A}|$ and $|\mathcal{I}|$. Hence, if, for every $A \in \Phi_{\mathcal{I}}$ that is not a pseudo-intent, the reachable sets in $\Phi_{\mathcal{I}}$ are ordered in such a way that the first one is not closed, Algorithm 5 can compute (but not necessarily recognize) a pseudo-intent in time polynomial in the size of the implication set. A minimal pseudo-intent P, by definition, does not contain any other pseudo-intent so the set of its reachable set is $\{P \setminus \{p\} \mid p \in P\}$ which can be computed in polynomial time. Thus, there are always orderings of reachable sets such that Algorithm 5 can compute minimal pseudo-intents with an incremental polynomial time between two of them. However, as shown in [6], minimal pseudo-intents cannot be enumerated in output polynomial time. Indeed, even though it is possible to compute a minimal pseudo-intent in time polynomial in the size of the implication set, confirming that they have all been found is exponential because non-minimal pseudo-intents and $\mathcal{A}$ have a potentially exponential number of reachable sets.

6 Conclusion

We have proposed two algorithms for computing pseudo-intents using the computation of minimal transversals as their main operation. The first can find pseudo-intents before some of their subsets that are themselves pseudo-intents but it is not complete for some enumeration orders. The second is complete but can only enumerate in orders that respect the inclusion relation. They can be

combined to obtain the enumeration order of the first with the completeness of the second. We expressed their delay as a function of the complexity of computing minimal transversals. We showed that, for some orderings of attribute sets, pseudo-intents can be reached, but not recognized, in incremental polynomial time. As both reaching and recognizing pseudo-intents, in our context, are related to the problem of computing minimal transversals in hypergraphs for which many special cases are known, we believe that this work may be useful in isolating more cases for which the enumeration of pseudo-intents is easier.

On the practical side, the fact that the algorithms do not necessarily enumerate the entirety of the set of intents should significantly reduce the runtime on dense contexts. Furthermore, the depth-first strategy used in the algorithms allows for some computations to be saved and reused. Further algorithmic optimizations and empirical comparisons with other algorithms for the same problem will be the subject of future investigations.

References

1. Mikhail A. Babin and Sergei O. Kuznetsov. Recognizing Pseudo-intents is coNP-complete. In *Proceedings of the 7th International Conference on Concept Lattices and Their Applications, Sevilla, Spain, October 19-21, 2010*, pages 294–301, 2010.
2. Mikhail A. Babin and Sergei O. Kuznetsov. Computing Premises of a Minimal Cover of Functional Dependencies is Intractable. *Discrete Applied Mathematics*, 161(6):742–749, 2013.
3. Konstantin Bazhanov and Sergei A. Obiedkov. Comparing Performance of Algorithms for Generating the Duquenne-Guigues Basis. In *Proceedings of The Eighth International Conference on Concept Lattices and Their Applications, Nancy, France, October 17-20, 2011*, pages 43–57, 2011.
4. Alexandre Bazin and Jean-Gabriel Ganascia. Enumerating Pseudo-Intents in a Partial Order. In *Proceedings of the Tenth International Conference on Concept Lattices and Their Applications, La Rochelle, France, October 15-18, 2013.*, pages 45–56, 2013.
5. Claude Berge. *Hypergraphs: Combinatorics of Finite Sets*, volume 45. Elsevier, 1984.
6. Felix Distel and Baris Sertkaya. On the Complexity of Enumerating Pseudo-intents. *Discrete Applied Mathematics*, 159(6):450–466, 2011.
7. Thomas Eiter and Georg Gottlob. Identifying the Minimal Transversals of a Hypergraph and Related Problems. *SIAM J. Comput.*, 24(6):1278–1304, 1995.
8. Thomas Eiter, Georg Gottlob, and Kazuhisa Makino. New Results on Monotone Dualization and Generating Hypergraph Transversals. *SIAM J. Comput.*, 32(2):514–537, 2003.
9. Thomas Eiter, Kazuhisa Makino, and Georg Gottlob. Computational Aspects of Monotone Dualization: A Brief Survey. *Discrete Applied Mathematics*, 156(11):2035–2049, 2008.
10. Michael L. Fredman and Leonid Khachiyan. On the Complexity of Dualization of Monotone Disjunctive Normal Forms. *J. Algorithms*, 21(3):618–628, 1996.
11. Bernhard Ganter. Two Basic Algorithms in Concept Analysis. In *Preprint Technische Hochschule Darmstadt*, 1984.

12. Jean-Louis Guigues and Vincent Duquenne. Familles Minimales d'Implications Informatives Résultant d'un Tableau de Données Binaires. *Mathématiques et Sciences humaines*, 95:5–18, 1986.
13. Vladimir Gurvich and Leonid Khachiyan. On Generating the Irredundant Conjunctive and Disjunctive Normal Forms of Monotone Boolean Functions. *Discrete Applied Mathematics*, 96-97:363–373, 1999.
14. Miki Hermann and Baris Sertkaya. On the Complexity of Computing Generators of Closed Sets. In *Formal Concept Analysis, 6th International Conference, ICFCA 2008, Montreal, Canada, February 25-28, 2008, Proceedings*, pages 158–168, 2008.
15. Sergei O. Kuznetsov. On the Intractability of Computing the Duquenne-Guigues Bas. *J. UCS*, 10(8):927–933, 2004.
16. Sergei O. Kuznetsov and Sergei A. Obiedkov. Counting Pseudo-intents and #P-completeness. In *Formal Concept Analysis, 4th International Conference, ICFCA 2006, Dresden, Germany, February 13-17, 2006, Proceedings*, pages 306–308, 2006.
17. Sergei O. Kuznetsov and Sergei A. Obiedkov. Some Decision and Counting Problems of the Duquenne-Guigues Basis of Implications. *Discrete Applied Mathematics*, 156(11):1994–2003, 2008.
18. Zbigniew Lonc and Miroslaw Truszczynski. On the Number of Minimal Transversals in 3-uniform Hypergraphs. *Discrete Mathematics*, 308(16):3668–3687, 2008.
19. Sebastian Rudolph. Some Notes on Pseudo-closed Sets. In *Formal Concept Analysis, 5th International Conference, ICFCA 2007, Clermont-Ferrand, France, February 12-16, 2007, Proceedings*, pages 151–165, 2007.