To appear in the *International Journal of General Systems*
Vol. 00, No. 00, Month 20XX, 1–18

NextClosures: Parallel Computation of the Canonical Base with Background Knowledge

Francesco Kriegel* and Daniel Borchmann

Institute of Theoretical Computer Science,
Technische Universität Dresden, Dresden, Germany

{francesco.kriegel,daniel.borchmann}@tu-dresden.de

(*Received 00 Month 20XX; accepted 00 Month 20XX*)

The canonical base of a formal context plays a distinguished role in Formal Concept Analysis, as it is the only minimal implicational base known so far that can be described explicitly. Consequently, several algorithms for the computation of this base have been proposed. However, all those algorithms work sequentially by computing only one pseudo-intent at a time – a fact that heavily impairs the practicability in real-world applications. In this paper, we shall introduce an approach that remedies this deficit by allowing the canonical base to be computed in a parallel manner with respect to arbitrary implicational background knowledge. First experimental evaluations show that for sufficiently large data sets the speed-up is proportional to the number of available CPU cores.

Keywords: Formal Concept Analysis · Parallel Algorithm · Implication · Canonical Base · Background Knowledge

1. Introduction

Formal Concept Analysis is a branch of mathematical order theory that allows to draw connections between otherwise unrelated research fields like logic and data mining. In particular, it provides methods to investigate problems from the realm of data mining under a mathematical point of view. For this purpose, data sets are represented as *formal contexts*, a mathematical abstraction of relational databases that allows to apply mathematical reasoning. Formal Concept Analysis then allows to study *valid implications* in those formal contexts, which loosely correspond to functional dependencies in databases. The set of all those valid implications of a formal context, called the *implicational theory*, is of interest in a large variety of applications, e.g., in learning knowledge bases from examples (Distel 2011; Borchmann, Distel, and Kriegel 2016; Kriegel 2016a). In such applications, computing a *minimal* set of implications that captures the whole implicational theory is often desirable. Those *minimal bases* provide a minimal representation of all implicational knowledge contained in a given data set. One of those bases is the *canonical base*, which is the only minimal implicational base that is known to be describable explicitly.

Conducting the computation of the canonical base may impose a major challenge, endangering the practicability of the underlying approach of extracting implicational knowledge from data sets. There are two known algorithms for computing the canonical base of a formal context (Ganter 2010; Obiedkov and Duquenne 2007). Both algorithms work sequentially, i.e., they compute one implication after the other. Moreover, both algorithms compute in addition to the implications of the canonical base all *formal concepts*

*Corresponding author

of the given context. This is a disadvantage, as the number of formal concepts can be exponential in the size of the canonical base. On the other hand, the size of the canonical base can be exponential in the size of the underlying formal context anyway (Kuznetsov 2004), and so this blowup cannot be avoided completely. Additionally, up to today it is not known whether the canonical base can be computed in output-polynomial time, and certain complexity results hint at a negative answer (Distel and Sertkaya 2011). For the algorithm from Ganter (2010), and indeed for any algorithm that enumerates the canonical base in a so-called *lectic order*, it was shown by Distel (2010) that it cannot conduct the computation with polynomial delay between two successive implications, unless $\mathsf{P} = \mathsf{NP}$.

However, the impact of theoretical complexity results for practical applications is often hard to access, and it may thus be worth investigating faster algorithms for theoretically intractable results nonetheless. A popular approach in this direction is to explore the possibilities of *parallelizing* known sequential algorithms. This is also true for Formal Concept Analysis, as can be seen by the development of parallel versions for computing the set of all formal concept of a formal context (Vychodil, Krajča, and Outrata 2008; Fu and Nguifo 2004).

In this work we want to investigate a parallel algorithm for computing the canonical base of a formal context $\mathbb{K}$. The underlying idea is actually quite simple, and was used by Lindig (2000) to (sequentially) compute the concept lattice of a formal context. In a nutshell, to compute the canonical base, we compute the closure system of all *intents* and *pseudo-intents* of $\mathbb{K}$. This set can be computed bottom up, in a level-wise order, and this computation can be done in parallel provided that the corresponding ordered set has a certain "width" at a particular level. The crucial observation is that the upper neighbors of an intent or pseudo-intent B can be easily computed by just iterating over all attributes $m \notin B$ and computing the closure of $B \cup \{m\}$. In the approach of Lindig mentioned above, this closure is just the usual double-prime operation $B \mapsto B^{II}$ of the underlying formal context $\mathbb{K}$. In our approach it is the closure operator the closures of which are exactly the intents and pseudo-intents of $\mathbb{K}$. Experimental results presented in this work indicate that for suitably large data sets the computation of the canonical base can be sped up by a factor proportional to the number of available CPU cores.

Surprisingly, despite the simplicity of our approach, we are not aware of any prior work on computing the canonical base of a formal context in a parallel manner.

The paper is structured as follows. After recalling all necessary notions of Formal Concept Analysis in Section 2 and of background implications in Section 3, we shall describe in Section 4 our approach of computing the canonical base in parallel. Benchmarks of this algorithm are presented in Section 5, and we shall close this work with some conclusions in Section 6.

2. Preliminaries

This section gives a brief overview on the notions of Formal Concept Analysis (Ganter and Wille 1999) that are used in this document. The basic structure is a *formal context* $\mathbb{K} := (G, M, I)$ consisting of a set G of *objects*, a set M of *attributes*, and an *incidence relation* $I \subseteq G \times M$. For a pair $(g, m) \in I$, we also use the infix notation $g\ I\ m$, and say that the object g *has* the attribute m. Furthermore, $\mathbb{K}$ induces the *derivation operators* $\cdot^I \colon \wp(G) \to \wp(M)$ and $\cdot^I \colon \wp(M) \to \wp(G)$ that are defined as follows for object sets $A \subseteq G$ and attribute sets $B \subseteq M$:

$$A^I := \{\, m \in M \mid \forall g \in A \colon g\ I\ m \,\} \quad \text{and} \quad B^I := \{\, g \in G \mid \forall m \in B \colon g\ I\ m \,\}.$$

In other words, A^I is the set of all attributes that all objects from A have in common, and dually B^I is the set of all objects which have all attributes from B. It is well-known that both derivation operators form a so-called *Galois connection* between the powersets $\wp(G)$ and $\wp(M)$, i.e., the following statements hold true for all subsets $A, A_1, A_2 \subseteq G$ and $B, B_1, B_2 \subseteq M$:

(1) $A \subseteq B^I \Leftrightarrow B \subseteq A^I$.
(2) $A \subseteq A^{II}$.
(3) $A^I = A^{III}$.
(4) $A_1 \subseteq A_2 \Rightarrow A_2^I \subseteq A_1^I$.
(5) $B \subseteq B^{II}$.
(6) $B^I = B^{III}$.
(7) $B_1 \subseteq B_2 \Rightarrow B_2^I \subseteq B_1^I$.

A *formal concept* of $\mathbb{K}$ is a pair (A, B) such that $A^I = B$ as well as $B = A^I$. We refer to A as the *extent*, and to B as the *intent* of (A, B). The set of all formal concepts is denoted as $\mathfrak{B}(\mathbb{K})$, and $\mathsf{Int}(\mathbb{K})$ symbolizes the set of all intents of $\mathbb{K}$. Furthermore, $\mathfrak{B}(\mathbb{K})$ can be ordered by defining $(A, B) \leq (C, D)$ if $A \subseteq C$ (or dually if $B \supseteq D$), and indeed then $(\mathfrak{B}(\mathbb{K}), \leq)$ is a complete lattice where infima and suprema are given as follows:

$$\bigwedge_{t\in T}(A_t, B_t) = (\bigcap_{t\in T} A_t, (\bigcup_{t\in T} B_t)^{II}) \quad \text{and} \quad \bigvee_{t\in T}(A_t, B_t) = ((\bigcup_{t\in T} A_t)^{II}, \bigcap_{t\in T} B_t).$$

One of the earliest results of Formal Concept Analysis was the proof showing that every complete lattice can be represented as the *concept lattice* $(\mathfrak{B}(\mathbb{K}), \leq)$ of a suitably chosen formal context $\mathbb{K}$ (Wille 1982, Theorem on Page 5).

An *implication* over the set M is an expression of the form $X \to Y$ where $X, Y \subseteq M$. An implication $X \to Y$ is *valid* in $\mathbb{K}$ if $X^I \subseteq Y^I$, and this is denoted as $\mathbb{K} \models X \to Y$. A set $\mathcal{L}$ of implications is *valid* in $\mathbb{K}$, and we write $\mathbb{K} \models \mathcal{L}$, if each implication in $\mathcal{L}$ is valid in $\mathbb{K}$. The set of all implications over M is symbolized by $\mathsf{Imp}(M)$, and $\mathsf{Imp}(\mathbb{K})$ is the set of all implications that are valid in $\mathbb{K}$. An implication $X \to Y$ *follows* from the set $\mathcal{L}$ if $X \to Y$ is valid in every context with attribute set M in which $\mathcal{L}$ is valid, and we shall denote this by $\mathcal{L} \models X \to Y$. Furthermore, a *model* of $X \to Y$ is a set $T \subseteq M$ such that $X \subseteq T$ implies $Y \subseteq T$, and we may then write $T \models X \to Y$. It is easy to prove the validity of the following lemma.

Lemma 1. *Let $\mathbb{K}$ be a formal context, and $X \to Y$ an implication. Then the following statements are equivalent:*

(1) $X \to Y$ is valid in $\mathbb{K}$.
(2) Each object intent $\{g\}^I$ for $g \in G$ is a model of $X \to Y$.
(3) Each intent A^I for $A \subseteq G$ is a model of $X \to Y$.
(4) $Y \subseteq X^{II}$.

A *model* of $\mathcal{L}$ is a model of all implications in $\mathcal{L}$, and $X^{\mathcal{L}}$ is the smallest superset of X that is a model of $\mathcal{L}$. The set $X^{\mathcal{L}}$ can be computed as follows.

$$X^{\mathcal{L}} = \bigcup\{\, X^{\mathcal{L}(n)} \mid n \in \mathbb{N}_+ \,\}$$
$$\text{where} \quad X^{\mathcal{L}(1)} := X \cup \bigcup\{\, B \mid A \to B \in \mathcal{L} \text{ and } A \subseteq X \,\}$$
$$\text{and} \quad X^{\mathcal{L}(n+1)} := (X^{\mathcal{L}(n)})^{\mathcal{L}(1)} \quad \text{for all } n \in \mathbb{N}_+.$$

We shall denote the set of all models of an implication set $\mathcal{L}$ as $\mathsf{Mod}(\mathcal{L})$.

The following lemma shows some well-known equivalent statements for entailment of implications from implication sets. We will not prove them here.

Lemma 2. *Let $\mathcal{L} \cup \{X \to Y\}$ be a set of implications over M. Then, the following statements are equivalent:*

(1) $X \to Y$ follows from $\mathcal{L}$.
(2) If $\mathbb{K}$ is a formal context with attribute set M such that $\mathcal{L}$ is valid in $\mathbb{K}$, then $X \to Y$ is also valid in $\mathbb{K}$.
(3) If $T \subseteq M$ and T is a model of $\mathcal{L}$, then T is a model of $X \to Y$, too.
(4) $Y \subseteq X^{\mathcal{L}}$.

From the above definition of a formal concept it follows that an attribute set $B \subseteq M$ is an intent of $\mathbb{K}$ if $B = B^{II}$. An attribute set $P \subseteq M$ is called a *pseudo-intent* of $\mathbb{K}$ if it is no intent, i.e., $P \neq P^{II}$, and for each pseudo-intent $Q \subsetneq P$ the set inclusion $Q^{II} \subseteq P$ is satisfied. We denote the set of all pseudo-intents of $\mathbb{K}$ by $\mathsf{PsInt}(\mathbb{K})$. Then, the *canonical base* of $\mathbb{K}$ is defined as the following implication set, cf. Guigues and Duquenne (1986); Ganter (1984):

$$\mathsf{Can}(\mathbb{K}) := \{\, P \to P^{II} \mid P \in \mathsf{PsInt}(\mathbb{K}) \,\}.$$

The canonical base has the property that it is a *minimal implicational base* of $\mathbb{K}$, i.e., it is an *implicational base* of $\mathbb{K}$, meaning that it is a set of valid implications of $\mathbb{K}$ (*soundness*) such that every valid implication of $\mathbb{K}$ is entailed by it (*completeness*). Furthermore, its cardinality is minimal among all implicational bases of $\mathbb{K}$.

For an implication set $\mathcal{L}$, we define the closure operator $\mathcal{L}^*$ as follows:

$$\begin{aligned} X^{\mathcal{L}^*} &:= \bigcup \{\, X^{\mathcal{L}(n)^*} \mid n \in \mathbb{N}_+ \,\} \\ \text{where} \quad X^{\mathcal{L}(1)^*} &:= X \cup \bigcup \{\, B \mid A \to B \in \mathcal{L} \text{ and } A \subsetneq X \,\} \\ \text{and} \quad X^{\mathcal{L}(n+1)^*} &:= (X^{\mathcal{L}(n)^*})^{\mathcal{L}(1)^*} \quad \text{for all } n \in \mathbb{N}_+. \end{aligned}$$

In particular, let $\mathbb{K}^* := \mathsf{Can}(\mathbb{K})^*$. It is then easy to verify that a subset $X \subseteq M$ is an intent or a pseudo-intent of $\mathbb{K}$ if, and only if, X is a closure of $\mathbb{K}^*$, cf. Lemma 4.

Note that there is a strong correspondence between Formal Concept Analysis and Propositional Logic. A formal context is just another notion for a set of propositional models (where the attributes in M are considered as propositional variables). In particular, for a formal context $\mathbb{K} := (G, M, I)$ the set $\mathcal{P}_{\mathbb{K}} := \{\, \chi^M_{g^I} \mid g \in G \,\}$ is a set of propositional models such that for each implication $X \to Y$ over M, $X \to Y$ is valid in $\mathbb{K}$ if, and only if, $\bigwedge X \to \bigwedge Y$ is valid in $\mathcal{P}_{\mathbb{K}}$. Note that $\chi^M_B \colon M \to \{0,1\}$ is the *characteristic function* of B in M, i.e., $\chi^M_B(m) := 1$ if $m \in B$, and $\chi^M_B(m) := 0$ otherwise.

Conversely, if $\mathcal{P} \subseteq \{0,1\}^M$ is a set of propositional models over a set M of propositional variables, then the formal context $\mathbb{K}_{\mathcal{P}} := (\{\, p^{-1}(1) \mid p \in \mathcal{P} \,\}, M, \ni)$ satisfies $\mathcal{P} \models \bigwedge X \to \bigwedge Y$ if, and only if, $\mathbb{K}_{\mathcal{P}} \models X \to Y$, for all implications $X \to Y$ over M. Here, p^{-1} denotes the *pre-image* of a propositional model $p \colon M \to \{0,1\}$, i.e., $p^{-1} \colon \{0,1\} \to \wp(M)$ where $p^{-1}(i) := \{\, m \in M \mid p(m) = i \,\}$.

3. Background Knowledge

In a former paper (Kriegel and Borchmann 2015) we have presented a parallel algorithm for computing canonical bases of formal contexts. However, the presence of background knowledge in form of an implication set was not investigated. Henceforth, we will provide an extension that allows for the computation of canonical bases with respect to background knowledge.

First, some notions have to be introduced. Let $\mathbb{K} := (G, M, I)$ be a formal context, and let $\mathcal{B}$ be a set of implications over M. We may distinguish two cases: Either $\mathcal{B}$ is valid in $\mathbb{K}$, or $\mathcal{B}$ is not valid in $\mathbb{K}$, i.e., $\mathcal{B}$ contains at least one implication that is not valid in $\mathbb{K}$.

First assume that all background implications in $\mathcal{B}$ are valid in $\mathbb{K}$. In (Stumme 1996; Ganter 1999) the following notions have been introduced. A *$\mathcal{B}$-pseudo-intent* of $\mathbb{K}$ is an attribute set $P \subseteq M$ that is no intent of $\mathbb{K}$, but is a model of $\mathcal{B}$, and for all $\mathcal{B}$-pseudo-intents $Q \subsetneq P$ it is true that $Q^{II} \subseteq P$. The set of all these pseudo-intents is denoted by $\mathsf{PsInt}(\mathbb{K}, \mathcal{B})$, and then the *canonical base* of $\mathbb{K}$ *relative* to $\mathcal{B}$ is given as

$$\mathsf{Can}(\mathbb{K}, \mathcal{B}) := \{\, P \to P^{II} \mid P \in \mathsf{PsInt}(\mathbb{K}, \mathcal{B}) \,\}$$

It is an *implicational base* of $\mathbb{K}$ *relative* to $\mathcal{B}$, i.e., its union with $\mathcal{B}$ is an implicational base of $\mathbb{K}$, or equivalently, for all implications $X \to Y$, it is true that

$$\mathbb{K} \models X \to Y \Leftrightarrow \mathsf{Can}(\mathbb{K}, \mathcal{B}) \cup \mathcal{B} \models X \to Y.$$

Furthermore, it is *minimal* (in the sense of minimal cardinality) among all implicational bases of $\mathbb{K}$ relative to $\mathcal{B}$.

Second, in the more general case assume that there is at least one background implication in $\mathcal{B}$ that is not valid in $\mathbb{K}$. However, the set of background implications is considered to be correct, as it has been created, adjusted, or checked, manually by a human expert. Then this would imply that the observed dataset, described by the formal context $\mathbb{K}$, is faulty. The goal is to axiomatize those implications that do not already follow from the background knowledge $\mathcal{B}$, but have all those intents of $\mathbb{K}$ as models which are also models of $\mathcal{B}$. In particular, this is similar to the notion of a C-implication where C is a closure operator on M, as introduced by Belohlávek and Vychodil (2006, 2013). Of course, there is a one-to-one correspondence between closure operators on M and implication sets on M, and hence it does not matter whether we describe the external constraints as a closure operator C or as an implication set $\mathcal{B}$.

A *$\mathcal{B}$-implication* is an implication where both premise and conclusion are models of $\mathcal{B}$. Furthermore, a $\mathcal{B}$-implication $X \to Y$ is *valid* in $(\mathbb{K}, \mathcal{B})$ if each intent of $\mathbb{K}$ that is a model of $\mathcal{B}$ also is a model of $X \to Y$. For an implication set $\mathcal{L} \cup \{X \to Y\}$, we say that $\mathcal{L}$ *$\mathcal{B}$-entails* $X \to Y$ if each model of $\mathcal{L}$ that is a model of $\mathcal{B}$ also is a model of $X \to Y$. It is readily verified that this is equivalent to $\mathcal{L} \cup \mathcal{B} \models X \to Y$. An *implicational base* of $(\mathbb{K}, \mathcal{B})$ is a set $\mathcal{L}$ of $\mathcal{B}$-implications that are valid in $(\mathbb{K}, \mathcal{B})$ (*soundness*), and furthermore $\mathcal{B}$-entails all $\mathcal{B}$-implications that are valid in $(\mathbb{K}, \mathcal{B})$ (*completeness*), i.e., for each $\mathcal{B}$-implication $X \to Y$, it holds true that $(\mathbb{K}, \mathcal{B}) \models X \to Y$ if, and only if, $\mathcal{B} \cup \mathcal{L} \models X \to Y$.

$$X^{II \curlyvee \mathcal{B}} := \bigcap \{ Y \mid X \subseteq Y \subseteq M \text{ and } Y = Y^{II} \text{ and } Y = Y^{\mathcal{B}} \}$$

is the smallest superset of X that is both an intent of $\mathbb{K}$ and a model of $\mathcal{B}$. Attribute sets X that are closed, i.e., which satisfy $X = X^{II \curlyvee \mathcal{B}}$, are called *intents* of $(\mathbb{K}, \mathcal{B})$. We shall denote the set of all intents of $(\mathbb{K}, \mathcal{B})$ by $\mathsf{Int}(\mathbb{K}, \mathcal{B})$. It is easy to verify that $\mathsf{Int}(\mathbb{K}, \mathcal{B}) = \mathsf{Int}(\mathbb{K}) \cap \mathsf{Mod}(\mathcal{B})$. Note that in the lattice of closure operators, $\cdot^{II \curlyvee \mathcal{B}}$ is the supremum of the closure operator $\cdot^{II} \colon \wp(M) \to \wp(M)$ which is induced by $\mathbb{K}$, and the closure operator $\cdot^{\mathcal{B}} \colon \wp(M) \to \wp(M)$ which is induced by $\mathcal{B}$.

Lemma 3. *Let $\mathbb{K}$ be a formal context, and $\mathcal{B} \cup \{X \to Y\}$ be an implication set. Then the following statements are equivalent:*

(1) $X \to Y$ is valid in $(\mathbb{K}, \mathcal{B})$.
(2) Each intent of $(\mathbb{K}, \mathcal{B})$ is a model of $X \to Y$.
(3) $Y \subseteq X^{II \curlyvee \mathcal{B}}$.

A pair (A, B) where $A \subseteq G$ and $B \subseteq M$ is a *formal concept* of $(\mathbb{K}, \mathcal{B})$ if B is an intent of $(\mathbb{K}, \mathcal{B})$ and $A = B^I$. The set of all formal concepts of $(\mathbb{K}, \mathcal{B})$ is denoted as $\mathfrak{B}(\mathbb{K}, \mathcal{B})$. Belohlávek and Vychodil (2006, 2013) prove that this set of formal concepts constitutes a complete lattice. An attribute set $P \subseteq M$ is called *pseudo-intent* of $(\mathbb{K}, \mathcal{B})$ if P is no intent of $\mathbb{K}$, but is a model of $\mathcal{B}$, and contains all closures $Q^{II \curlyvee \mathcal{B}}$ for pseudo-intents $Q \subsetneq P$. The set of all pseudo-intents is $\mathsf{PsInt}(\mathbb{K}, \mathcal{B})$, and the *canonical base* is defined as

$$\mathsf{Can}(\mathbb{K}, \mathcal{B}) := \{ P \to P^{II \curlyvee \mathcal{B}} \mid P \in \mathsf{PsInt}(\mathbb{K}, \mathcal{B}) \}.$$

It can be shown that this base is a minimal implicational base of the implications valid in $(\mathbb{K}, \mathcal{B})$. It is easy to show that an attribute set $X \subseteq M$ is an intent or a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ if, and only if, it is a model of $\mathcal{B}$ as well as a closure of $(\mathbb{K}, \mathcal{B})^* := \mathsf{Can}(\mathbb{K}, \mathcal{B})^*$.

Finally, note that this latter case is really a generalization of the former, since if $\mathbb{K} \models \mathcal{B}$, all intents of $\mathbb{K}$ are models of $\mathcal{B}$, i.e., $Y = Y^{II}$ implies $Y = Y^{\mathcal{B}}$. Thus, the closure operators $\cdot^{II \curlyvee \mathcal{B}}$ and $\cdot^{II}$ coincide.

4. Parallel Computation of the Canonical Base with Background Knowledge

The well-known `NextClosure` algorithm developed by Ganter (2010) can be used to enumerate the implications of the canonical base. The mathematical idea behind this algorithm is to compute all intents and pseudo-intents of our formal context $\mathbb{K}$ in a certain linear order, namely the *lectic order*. As an advantage the next (pseudo-)intent is uniquely determined, but we potentially have to compute several candidate closures in order to find it. As we have seen in Section 2, those sets form a complete lattice, and the `NextClosure` algorithm uses the closure operator $\mathbb{K}^*$ of this lattice to enumerate the pseudo-intents (and the intents as a by-product) of $\mathbb{K}$ in the lectic order. Furthermore, this algorithm is inherently sequential, i.e., it is not possible to parallelize it.

In our approach we shall not make use of the lectic order. Indeed, our algorithm will enumerate all intents and pseudo-intents of $(\mathbb{K}, \mathcal{B})$ in the subset-order, with no further restrictions. As a benefit we get a very easy and obvious way to parallelize this enumeration. Moreover, in multi-threaded implementations no communication between different threads is necessary. However, as it is the case with all other known algorithms for computing the canonical base, we also have to compute all intents in addition to all pseudo-intents of the given formal context $\mathbb{K}$ and the background knowledge $\mathcal{B}$.

The main idea is very simple and works as follows. From the definition of pseudo-intents we see that in order to decide whether an attribute set $P \subseteq M$ is a pseudo-intent we only need all pseudo-intents $Q \subsetneq P$, i.e., it suffices to know all pseudo-intents with a smaller cardinality than P. This allows for the level-wise computation of the canonical base with respect to the subset inclusion order, i.e., we can enumerate the (pseudo-)intents with respect to increasing cardinality.

An algorithm that implements this idea works as follows. First we start by considering the $\mathcal{B}$-closure of the empty set, as it is the smallest model of $\mathcal{B}$. Of course, it must either be an intent or a pseudo-intent, and the distinction can be made by checking whether $\emptyset^{\mathcal{B}} = \emptyset^{II \curlyvee \mathcal{B}}$. Then assuming inductively that all pseudo-intents with cardinality $< k$ have been determined, we can correctly decide whether a subset $P \subseteq M$ with $|P| = k$ is a pseudo-intent or not.

To compute the set of all intents and pseudo-intents of $(\mathbb{K}, \mathcal{B})$, the algorithm manages a set of *candidates* that contains the (pseudo-)intents on the current level. Then, whenever a pseudo-intent P has been found, the $\subseteq$-next closure is uniquely determined by its closure $P^{II \curlyvee \mathcal{B}}$. If an intent B has been found, then the $\subseteq$-next closures must be of the form $(B \cup \{m\})^{(\mathbb{K},\mathcal{B})^*}$ for an attribute $m \notin B$. However, as we are not aware of the full implicational base of $(\mathbb{K}, \mathcal{B})$ yet, but only of an *approximation* $\mathcal{L}$ of it, the operators $(\mathbb{K}, \mathcal{B})^*$ and $\mathcal{L}^*$ do not coincide on all models of $\mathcal{B}$. We will show that they yield the same closure for models B of $\mathcal{B}$ with a cardinality $|B| \leq k$ if $\mathcal{L}$ contains all implications $P \to P^{II \curlyvee \mathcal{B}}$ where P is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ with a cardinality $|P| < k$. Consequently, the $\mathcal{L}^*$-closure of a set $B \cup \{m\}$ may not be an intent or a pseudo-intent of $(\mathbb{K}, \mathcal{B})$. Instead, they are added to the candidate list, and are processed when all pseudo-intents with smaller cardinality have been determined. We will formally prove that this technique is correct. Furthermore, the computation of all pseudo-intents and intents of cardinality k can be done in parallel, since they are independent of each other.

In summary, we can describe the inductive structure of the algorithm as follows: Let $\mathbb{K}$ be a finite formal context, and $\mathcal{B}$ be a set of background implication. We use four variables: k denotes the current cardinality of candidates, $\mathbf{C}$ is the set of candidates, $\mathfrak{B}$ is a set of formal concepts, and $\mathcal{L}$ is an implication set. Then the algorithm works as follows.

(1) Set $k := |\emptyset^{\mathcal{B}}|$, $\mathbf{C} := \{\emptyset^{\mathcal{B}}\}$, $\mathfrak{B} := \emptyset$, and $\mathcal{L} := \emptyset$.
(2) In parallel: For each candidate set $C \in \mathbf{C}$ with cardinality $|C| = k$, determine

whether it is both $\mathcal{L}^*$-closed and $\mathcal{B}$-closed. If not, then add its closure $C^{\mathcal{L}^* \curlyvee \mathcal{B}}$ to the candidate set $\mathbf{C}$, and go to Step 5.

(3) If C is an intent of $(\mathbb{K}, \mathcal{B})$, then add the formal concept (C^I, C) to $\mathfrak{B}$. Otherwise, C must be a pseudo-intent, and thus we add the implication $C \to C^{II \curlyvee \mathcal{B}}$ to the set $\mathcal{L}$, and add the formal concept $((C^{II \curlyvee \mathcal{B}})^I, C^{II \curlyvee \mathcal{B}})$ to the set $\mathfrak{B}$.

(4) For each observed intent $C^{II \curlyvee \mathcal{B}}$, add all its upper neighbors $C^{II \curlyvee \mathcal{B}} \cup \{m\}$ where $m \notin C^{II \curlyvee \mathcal{B}}$ to the candidate set $\mathbf{C}$.

(5) Wait until all candidates of cardinality k have been processed. If $k < |M|$, then increase the candidate cardinality k by 1, and go to Step 2. Otherwise return $\mathfrak{B}$ and $\mathcal{L}$.

In order to approximate the operator $\mathcal{L}^*$ we furthermore introduce the following notion: If $\mathcal{L}$ is a set of implications, then $\mathcal{L}{\restriction_k}$ denotes the subset of $\mathcal{L}$ that consists of all implications the premises of which have a cardinality not exceeding k.

Lemma 4. *Let $\mathbb{K} := (G, M, I)$ be a formal context, $\mathcal{B} \subseteq \mathsf{Imp}(M)$ an implication set over M, and $\mathcal{L} \subseteq \mathsf{Imp}(M)$ the canonical base of $(\mathbb{K}, \mathcal{B})$. Then, for all attribute sets $X \subseteq M$ that are models of $\mathcal{B}$, the following statements are equivalent:*

(1) X is either an intent or a pseudo-intent of $(\mathbb{K}, \mathcal{B})$.
(2) X is $(\mathbb{K}, \mathcal{B})^*$-closed.
(3) X is $\mathcal{L}^*$-closed.
(4) X is $(\mathcal{L}{\restriction_{|X|-1}})^*$-closed.
(5) There is a $k \geq |X| - 1$ such that X is $(\mathcal{L}{\restriction_k})^*$-closed.
(6) For all $k \geq |X| - 1$, it holds that X is $(\mathcal{L}{\restriction_k})^*$-closed.

Proof. (1⇒2) If $X \in \mathsf{Int}(\mathbb{K}, \mathcal{B})$, then it immediately follows that X contains all $(\mathbb{K}, \mathcal{B})$-closures of its subsets (since $\cdot^{II \curlyvee \mathcal{B}}$ is a closure operator), and hence must be closed under $(\mathbb{K}, \mathcal{B})(1)^*$. Consequently, X is a closure of $(\mathbb{K}, \mathcal{B})^*$. If $X \in \mathsf{PsInt}(\mathbb{K}, \mathcal{B})$, then it is $(\mathbb{K}, \mathcal{B})(1)^*$-closed by definition of a pseudo-intent. (2⇒1) Vice versa, assume that X is a closure of $(\mathbb{K}, \mathcal{B})^*$ which is no intent of $(\mathbb{K}, \mathcal{B})$. Since X is a model of $\mathcal{B}$, it follows that it cannot be an intent of $\mathbb{K}$. Hence, X must be a pseudo-intent of $(\mathbb{K}, \mathcal{B})$. (2⇔3) is trivial. (3⇔4) follows directly from the fact that $P \subsetneq X$ implies $|P| < |X|$. (4⇔5) The only-if direction is trivial. Consider $k \geq |X| - 1$ such that X is $(\mathcal{L}{\restriction_k})^*$-closed. Then X contains all conclusions B where $A \to B \in \mathcal{L}$ is an implication with premise $A \subsetneq X$ such that $|A| \leq k$. Of course, $A \subsetneq X$ implies $|A| < |X|$, and thus X is $(\mathcal{L}{\restriction_{|X|-1}})^*$-closed as well. (4⇔6) The only-if-direction is trivial. Finally, assume that $k \geq |X| - 1$ and X is $(\mathcal{L}{\restriction_{|X|-1}})^*$-closed. Obviously, there are no subsets $A \subsetneq X$ with $|X| \leq |A| \leq k$, and so X must be $(\mathcal{L}{\restriction_k})^*$-closed, too. □

As an immediate consequence of Lemma 4 we infer that in order to decide the $(\mathbb{K}, \mathcal{B})^*$-closedness of an attribute set X it suffices to know all implications in the canonical base the premises of which have a lower cardinality than X.

Corollary 5. *If $\mathcal{L}$ contains all implications $P \to P^{II \curlyvee \mathcal{B}}$ where P is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ with $|P| < k$, and otherwise only implications with premise cardinality k, then for all attribute sets $X \subseteq M$ with $|X| \leq k$, the following statements are equivalent:*

(1) X is either an intent or a pseudo-intent of $(\mathbb{K}, \mathcal{B})$.
(2) X is a closure of $\mathcal{L}^*$ and a model of $\mathcal{B}$.

In a certain sense, this corollary allows us to approximate the set of all $(\mathbb{K}, \mathcal{B})^*$-closures in the order of increasing cardinality, and thus also permits the approximation of the closure operator $\mathcal{L}^*$ where $\mathcal{L}$ is the canonical base of $(\mathbb{K}, \mathcal{B})$. In the following Lemma 6 we will characterize the structure of the lattice of all $(\mathbb{K}, \mathcal{B})^*$-closures, and also give a method to compute upper neighbors. It is true that between comparable pseudo-intents there must always be an intent. In particular, the unique upper $(\mathbb{K}, \mathcal{B})^*$-closed neighbor of a pseudo-intent must be an intent.

Algorithm 1. `NextClosuresWithBackgroundKnowledge`

`Input:` a formal context $\mathbb{K} \coloneqq (G, M, I)$
`Input:` an implication set $\mathcal{B} \subseteq \mathsf{Imp}(M)$
`Initialize:` a candidate set $\mathbf{C} \coloneqq \{\emptyset^{\mathcal{B}}\}$
`Initialize:` a formal concept set $\mathfrak{B} \coloneqq \emptyset$
`Initialize:` an implication set $\mathcal{L} \coloneqq \emptyset$
`for all` $k = |\emptyset^{\mathcal{B}}|, \ldots, |M|$ `do`
  `for all` $C \in \mathbf{C}$ with $|C| = k$ `do in parallel`
    `if` $C = C^{\mathcal{L}^*}$ and $C = C^{\mathcal{B}}$ `then`
      `if` $C \neq C^{II}$ `then`
        $\mathcal{L} \coloneqq \mathcal{L} \cup \{C \to C^{II\curlyvee\mathcal{B}}\}$
      $\mathfrak{B} \coloneqq \mathfrak{B} \cup \{((C^{II\curlyvee\mathcal{B}})^I, C^{II\curlyvee\mathcal{B}})\}$
      $\mathbf{C} \coloneqq \mathbf{C} \cup \{\, C^{II\curlyvee\mathcal{B}} \cup \{m\} \mid m \notin C^{II\curlyvee\mathcal{B}} \,\}$
    `else`
      $\mathbf{C} \coloneqq \mathbf{C} \cup \{C^{\mathcal{L}^*\curlyvee\mathcal{B}}\}$
  Wait for termination of all parallel processes.
`Output:` the set $\mathfrak{B}$ of all formal concepts of $(\mathbb{K}, \mathcal{B})$
`Output:` the canonical base $\mathcal{L}$ of $(\mathbb{K}, \mathcal{B})$

Lemma 6. *Let $\mathbb{K} \coloneqq (G, M, I)$ be a formal context, and $\mathcal{B} \subseteq \mathsf{Imp}(M)$ be an implication set over M. Then the following statements hold true:*

(1) If $P \subseteq M$ is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$, then there is no intent or pseudo-intent strictly between P and $P^{II\curlyvee\mathcal{B}}$.
(2) If $B \subseteq M$ is an intent, then the next intents or pseudo-intents are of the form $(B \cup \{m\})^{(\mathbb{K},\mathcal{B})^*}$ for attributes $m \notin B$.
(3) If $X \subsetneq Y \subseteq M$ are neighboring $(\mathbb{K}, \mathcal{B})^*$-closures, then $Y = (X \cup \{m\})^{(\mathbb{K},\mathcal{B})^*}$ for all attributes $m \in Y \setminus X$.

Proof. (1) Assume that $P \in \mathsf{PsInt}(\mathbb{K}, \mathcal{B})$. Then for every intent B between P and $P^{II\curlyvee\mathcal{B}}$, i.e., $P \subseteq B \subseteq P^{II\curlyvee\mathcal{B}}$, we have that $B = B^{II\curlyvee\mathcal{B}} = P^{II\curlyvee\mathcal{B}}$. Thus, there cannot be an intent strictly between P and $P^{II\curlyvee\mathcal{B}}$. Furthermore, if Q were a pseudo-intent such that $P \subsetneq Q \subseteq P^{II\curlyvee\mathcal{B}}$, then by definition of a pseudo-intent it follows that $P^{II\curlyvee\mathcal{B}} \subseteq Q$, which is an obvious contradiction.

(2) Let $B \subseteq M$ be an intent of $(\mathbb{K}, \mathcal{B})$. Consider a subset $X \supseteq B$ that is an intent or a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ such that there is no other intent or pseudo-intent between them. Of course, then $B \subseteq B \cup \{m\} \subseteq X$ for all $m \in X \setminus B$. Consequently, $B = B^{(\mathbb{K},\mathcal{B})^*} \subsetneq (B \cup \{m\})^{(\mathbb{K},\mathcal{B})^*} \subseteq X^{(\mathbb{K},\mathcal{B})^*} = X$. Then $(B \cup \{m\})^{(\mathbb{K},\mathcal{B})^*}$ is an intent or a pseudo-intent between B and X which strictly contains B, and thus $(B \cup \{m\})^{(\mathbb{K},\mathcal{B})^*} = X$.

(3) Let $m \in Y \setminus X$. Then $X \cup \{m\} \subseteq Y$ implies that $X \subsetneq (X \cup \{m\})^{(\mathbb{K},\mathcal{B})^*} \subseteq Y$, since Y is already closed. Consequently, $(X \cup \{m\})^{(\mathbb{K},\mathcal{B})^*} = Y$. □

We are now ready to formulate our algorithm `NextClosuresWithBackgroundKnowledge` in pseudo-code, see Algorithm 1. In the remainder of this section we shall show that this algorithm always terminates for finite formal contexts $\mathbb{K}$ and sets $\mathcal{B}$ of background implications, and that it returns the canonical base as well as the set of all formal concepts of $(\mathbb{K}, \mathcal{B})$. Beforehand, let us introduce the following notation:

(1) Algorithm 1 is *in state* k (where $-1 \leq k \leq |M|$) if it has processed all candidate sets with a cardinality $\leq k$, but none of cardinality $> k$.
(2) $\mathbf{C}_k$ denotes the set of candidates in state k.
(3) $\mathcal{L}_k$ denotes the set of implications in state k.
(4) $\mathfrak{B}_k$ denotes the set of formal concepts in state k.

Proposition 7. *Let $\mathbb{K} \coloneqq (G, M, I)$ be a formal context, let $\mathcal{B} \subseteq \mathsf{Imp}(M)$ be an implication set over M, and assume that Algorithm 1 has been started on $(\mathbb{K}, \mathcal{B})$ and is* *in state k. Then, the following statements are true:*

(1) $\mathbf{C}_k$ contains all pseudo-intents of $(\mathbb{K}, \mathcal{B})$ with cardinality $k+1$, and all intents of $(\mathbb{K}, \mathcal{B})$ with cardinality $k+1$ the corresponding formal concept of which is not yet in $\mathfrak{B}_k$.
(2) $\mathfrak{B}_k$ contains all formal concepts of $(\mathbb{K}, \mathcal{B})$ the intent of which has cardinality $\leq k$.
(3) $\mathcal{L}_k$ contains all implications $P \to P^{II \curlyvee \mathcal{B}}$ where the premise P is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ with cardinality $\leq k$.
(4) Between the states k and $k+1$ an attribute set with cardinality $k+1$ is an intent or pseudo-intent of $(\mathbb{K}, \mathcal{B})$ if, and only if, it is a closure of both $\mathcal{L}^*$ and $\mathcal{B}$.

Proof. We prove the statements by induction on k. The base case handles the initial state $k = |\emptyset^{\mathcal{B}}| - 1$. Of course, $\emptyset^{\mathcal{B}}$ is always an intent or a pseudo-intent of $(\mathbb{K}, \mathcal{B})$. Furthermore, it is contained in the candidate set $\mathbf{C}$. As $\emptyset^{\mathcal{B}}$ is the smallest model of $\mathcal{B}$, there are no models of $\mathcal{B}$ with cardinality $\leq |\emptyset^{\mathcal{B}}| - 1$, and so $\mathfrak{B}_{|\emptyset^{\mathcal{B}}|-1}$ and $\mathcal{L}_{|\emptyset^{\mathcal{B}}|-1}$ trivially satisfy Statements 2 and 3. Finally, we have that $\mathcal{L}_{|\emptyset^{\mathcal{B}}|-1} = \emptyset$, and hence every attribute set is $\mathcal{L}^*_{|\emptyset^{\mathcal{B}}|-1}$-closed, in particular $\emptyset^{\mathcal{B}}$.

We now assume that the induction hypothesis is true for k. For every implication set $\mathcal{L}$ between states k and $k+1$, i.e., $\mathcal{L}_k \subseteq \mathcal{L} \subseteq \mathcal{L}_{k+1}$, the induction hypothesis yields that $\mathcal{L}$ contains all implications $P \to P^{II \curlyvee \mathcal{B}}$ where P is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ with cardinality $\leq k$, and furthermore only implications the premises of which have cardinality $k+1$ (by definition of Algorithm 1). Additionally, we know that the candidate set $\mathbf{C}$ contains all pseudo-intents P of $(\mathbb{K}, \mathcal{B})$ where $|P| = k+1$, and all intents B of $(\mathbb{K}, \mathcal{B})$ such that $|B| = k+1$ and $(B^I, B) \notin \mathfrak{B}$. Corollary 5 immediately yields the validity of Statements 2 and 3 for $k+1$, as those $(\mathbb{K}, \mathcal{B})^*$-closures are recognized correctly in Line 3. Then $\mathcal{L}_{k+1}$ contains all implications $P \to P^{II \curlyvee \mathcal{B}}$ where P is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$ with $|P| \leq k+1$, and hence each implication set $\mathcal{L}$ with $\mathcal{L}_{k+1} \subseteq \mathcal{L} \subseteq \mathcal{L}_{k+2}$ contains all those implications, too, and furthermore only implications with a premise cardinality $k+2$. By another application of Corollary 5 we conclude that also Statement 4 is satisfied for $k+1$.

Finally, we show Statement 1 for $k+1$. Consider any $(\mathbb{K}, \mathcal{B})^*$-closed set X where $|X| = k+2$. Then Lemma 6 states that for all lower $(\mathbb{K}, \mathcal{B})^*$-neighbors Y and all $m \in X \setminus Y$ it is true that $(Y \cup \{m\})^{(\mathbb{K}, \mathcal{B})^*} = X$. We proceed with a case distinction.

If there is a lower $(\mathbb{K}, \mathcal{B})^*$-neighbor Y which is a pseudo-intent, then Lemma 6 yields that the (unique) next $(\mathbb{K}, \mathcal{B})^*$-neighbor is obtained as $Y^{II \curlyvee \mathcal{B}}$, and the formal concept $((Y^{II \curlyvee \mathcal{B}})^I, Y^{II \curlyvee \mathcal{B}})$ is added to the set $\mathfrak{B}$ in Line 6. Of course, it is true that $X = Y^{II \curlyvee \mathcal{B}}$.

Otherwise all lower $(\mathbb{K}, \mathcal{B})^*$-neighbors Y are intents, and in particular this is the case for X being a pseudo-intent, cf. Lemma 6. Then for all these Y we have $(Y \cup \{m\})^{(\mathbb{K}, \mathcal{B})^*} = X$ for all $m \in X \setminus Y$. Furthermore, all sets Z with $Y \cup \{m\} \subsetneq Z \subsetneq X$ are not $(\mathbb{K}, \mathcal{B})^*$-closed. Since $X \setminus Y$ is finite, the following sequence must also be finite:

$$C_0 \coloneqq Y \cup \{m\} \quad \text{and} \quad C_{i+1} \coloneqq C_i^{\mathcal{L}^* \curlyvee \mathcal{B}} \quad \text{where} \quad \mathcal{L}_{|C_i|-1} \subseteq \mathcal{L} \subseteq \mathcal{L}_{|C_i|}.$$

The sequence is well-defined, since implications from $\mathcal{L}_{|C_i|} \setminus \mathcal{L}_{|C_i|-1}$ have no influence on the closure of C_i. Furthermore, the sequence obviously ends with the set X, and contains no further $(\mathbb{K}, \mathcal{B})^*$-closed sets, and each of the sets $C_0, C_1, \ldots$ appears as a candidate during the run of the algorithm, cf. Lines 7 and 9. □

From the previous result we can infer that in the last state $|M|$ the set $\mathfrak{B}$ contains all formal concepts of $(\mathbb{K}, \mathcal{B})$, and that $\mathcal{L}$ is the canonical base of $(\mathbb{K}, \mathcal{B})$. Both sets are returned from Algorithm 1, and hence we can conclude that `NextClosuresWithBackgroundKnowledge` is sound and complete. The following corollary summarizes our results obtained so far, and also shows termination.

Corollary 8. *If Algorithm 1 is started on a finite formal context $\mathbb{K}$ and a finite implication set $\mathcal{B}$ as input, then it terminates, and returns both the set of all formal concepts as well as the canonical base of $(\mathbb{K}, \mathcal{B})$ as output.*

Algorithm 2. `NextClosures`

`Input:` a formal context $\mathbb{K} \coloneqq (G, M, I)$
`Initialize:` a candidate set $\mathbf{C} \coloneqq \{\emptyset\}$
`Initialize:` a formal concept set $\mathfrak{B} \coloneqq \emptyset$
`Initialize:` an implication set $\mathcal{L} \coloneqq \emptyset$
`for all` $k = 0, \ldots, |M|$ `do`
  `for all` $C \in \mathbf{C}$ with $|C| = k$ `do in parallel`
    `if` $C = C^{\mathcal{L}^*}$ `then`
      `if` $C \neq C^{II}$ `then`
        $\mathcal{L} \coloneqq \mathcal{L} \cup \{C \to C^{II}\}$
      $\mathfrak{B} \coloneqq \mathfrak{B} \cup \{(C^I, C^{II})\}$
      $\mathbf{C} \coloneqq \mathbf{C} \cup \{\, C^{II} \cup \{m\} \mid m \notin C^{II} \,\}$
    `else`
      $\mathbf{C} \coloneqq \mathbf{C} \cup \{C^{\mathcal{L}^*}\}$
  Wait for termination of all parallel processes.
`Output:` the set $\mathfrak{B}$ of all formal concepts of $\mathbb{K}$
`Output:` the canonical base $\mathcal{L}$ of $\mathbb{K}$

Proof. The second part of the statement is a direct consequence of Proposition 7. In the final state $|M|$ the set $\mathcal{L}$ contains all implications $P \to P^{II \curlyvee \mathcal{B}}$ where P is a pseudo-intent of $(\mathbb{K}, \mathcal{B})$. In particular, $\mathcal{L}$ is the canonical base. Furthermore, the set $\mathfrak{B}$ contains all formal concepts of $(\mathbb{K}, \mathcal{B})$.

Finally, the computation time between states k and $k+1$ is finite, because there are only finitely many candidates of cardinality $k+1$, and the computation of closures of the operators $\cdot^{\mathcal{L}^*}$, $\cdot^{\mathcal{B}}$, $\cdot^{II}$, and $\cdot^{II \curlyvee \mathcal{B}}$, can be done in finite time. As there are exactly $|M|$ states for a finite formal context, the algorithm must terminate. □

One could ask whether there are formal contexts that do not allow for a speed-up in the enumeration of all intents and pseudo-intents on parallel execution. This would happen for formal contexts the intents and pseudo-intents of which are linearly ordered, i.e., form a chain with respect to the subset inclusion order. However, the next Lemma 9 shows that this is impossible.

Remark that a formal context $\mathbb{K} \coloneqq (G, M, I)$ is *clarified* if $\{g\}^I = \{h\}^I$ implies $g = h$ for all objects $g, h \in G$, and dually $\{m\}^I = \{n\}^I$ implies $m = n$ for all attributes $m, n \in M$.

Lemma 9. *For each non-empty clarified formal context, the set of its intents and pseudo-intents is not linearly ordered with respect to the subset inclusion order.*

Proof. Assume that $\mathbb{K} \coloneqq (G, M, I)$ with $G \coloneqq \{g_1, \ldots, g_n\}$, $n > 0$, were a clarified formal context with (pseudo-)intents $P_1 \subsetneq P_2 \subsetneq \ldots \subsetneq P_\ell$. In particular, then all object intents form a chain $g_1^I \subsetneq g_2^I \subsetneq \ldots \subsetneq g_n^I$ where $n \leq \ell$. Since $\mathbb{K}$ is clarified, it follows $|g_{j+1}^I \setminus g_j^I| = 1$ for all j, and hence w.l.o.g. $M = \{m_1, \ldots, m_n\}$, and $g_i \; I \; m_j$ if, and only if, $i \geq j$. Hence, $\mathbb{K}$ is isomorphic to the ordinal scale $\mathbb{K}_n \coloneqq (\{1, \ldots, n\}, \{1, \ldots, n\}, \leq)$. It is easy to verify that the pseudo-intents of $\mathbb{K}_n$ are either $\emptyset$, or of the form $\{m, n\}$ where $m < n-1$, a contradiction. □

Consequently, there is no formal context with a linearly ordered set of intents and pseudo-intents. Hence, a parallel enumeration of the intents and pseudo-intents will always result in a speed-up compared to a sequential enumeration.

In the case where no background knowledge is available, i.e., $\mathcal{B} = \emptyset$, we can easily see that Algorithm 1 may be simplified to Algorithm 2 which computes the canonical base of a formal context $\mathbb{K}$, as it has been described by Kriegel (2015); Kriegel and Borchmann (2015).

5. Benchmarks

The purpose of this section is to show that our parallel Algorithms 1 and 2 for computing the canonical base indeed yield a speed-up, both qualitatively and quantitatively, compared to

	Formal Context	Objects	Attributes	Density
ⓐ	car.cxt	1728	25	28 %
ⓑ	mushroom.cxt	8124	119	19 %
ⓒ	tic-tac-toe.cxt	958	29	34 %
ⓓ	wine.cxt	178	68	20 %
ⓔ	algorithms.cxt	2688	54	22 %
ⓕ	o1000a10d10.cxt	1000	10	10 %
ⓖ	o1000a20d10.cxt	1000	20	10 %
ⓗ	o1000a36d17.cxt	1000	36	16 %
ⓘ	o1000a49d14.cxt	1000	49	14 %
ⓙ	o1000a50d10.cxt	1000	50	10 %
ⓚ	o1000a64d12.cxt	1000	64	12 %
ⓛ	o1000a81d11.cxt	1000	81	11 %
ⓜ	o1000a100d10-001.cxt	1000	100	11 %
ⓝ	o1000a100d10-002.cxt	1000	100	11 %
ⓞ	o1000a100d10.cxt	1000	100	11 %
ⓟ	o2000a81d11.cxt	2000	81	11 %
ⓠ	24.cxt	17	26	51 %
ⓡ	35.cxt	18	24	43 %
ⓢ	51.cxt	26	17	76 %
ⓣ	54.cxt	20	20	48 %
ⓤ	79.cxt	25	26	68 %

Figure 1. Formal Contexts in Benchmarks

the classical algorithm based on `NextClosure` (Ganter 2010). To this end, we shall present the running times of our algorithm `NextClosures` when applied to selected data sets and with a varying number of available CPU cores. We shall see that, up to a certain limit, the running time of our algorithm decreases proportional to the number of available CPU cores. Furthermore, we shall also show that this speed-up is not only qualitative, but indeed yields a real speed-up compared to the original sequential algorithm for computing the canonical base.

The presented algorithms `NextClosures` and `NextClosuresWithBackgroundKnowledge` have been integrated into *Concept Explorer FX* (Kriegel 2010–2017). The implementations are straight-forward adaptions of Algorithms 1 and 2 to the programming language Java 8, and heavily use the new Stream API and thread-safe concurrent collection classes (like `ConcurrentHashMap`). As we have described before, the processing of all candidates on the current cardinality level can be done in parallel, i.e., for each of them a separate thread is started that executes the necessary operations for Lines 3 to 9 in Algorithms 1 and 2, respectively. Furthermore, as the candidates on the same level cannot affect each other, no communication between the threads is needed. More specifically, we have seen that the decision whether a candidate is an intent or a pseudo-intent is independent of all other sets with the same or a higher cardinality.

The formal contexts used for the benchmarks[1] are listed in Figure 1, and are either obtained from the FCA Data Repository (ⓐ to ⓓ, and ⓕ to ⓟ), randomly created (ⓠ to ⓤ), or created from experimental results (ⓔ). For each of them we executed the implementation at least three times, and recorded the average computation times. The experiments were performed on the following two systems:

Taurus (1 Node of Bull HPC-Cluster)
CPU: 2x Intel Xeon E5-2690 with eight cores @ 2.9 GHz, RAM: 32 GB
Atlas (1 Node of Megware PC-Farm)
CPU: 4x AMD Opteron 6274 with sixteen cores @ 2.2 GHz, RAM: 64 GB

Please note that the experiments were only conducted for the implementation of the

[1] The test contexts used for the experiments can be obtained from the authors via email.

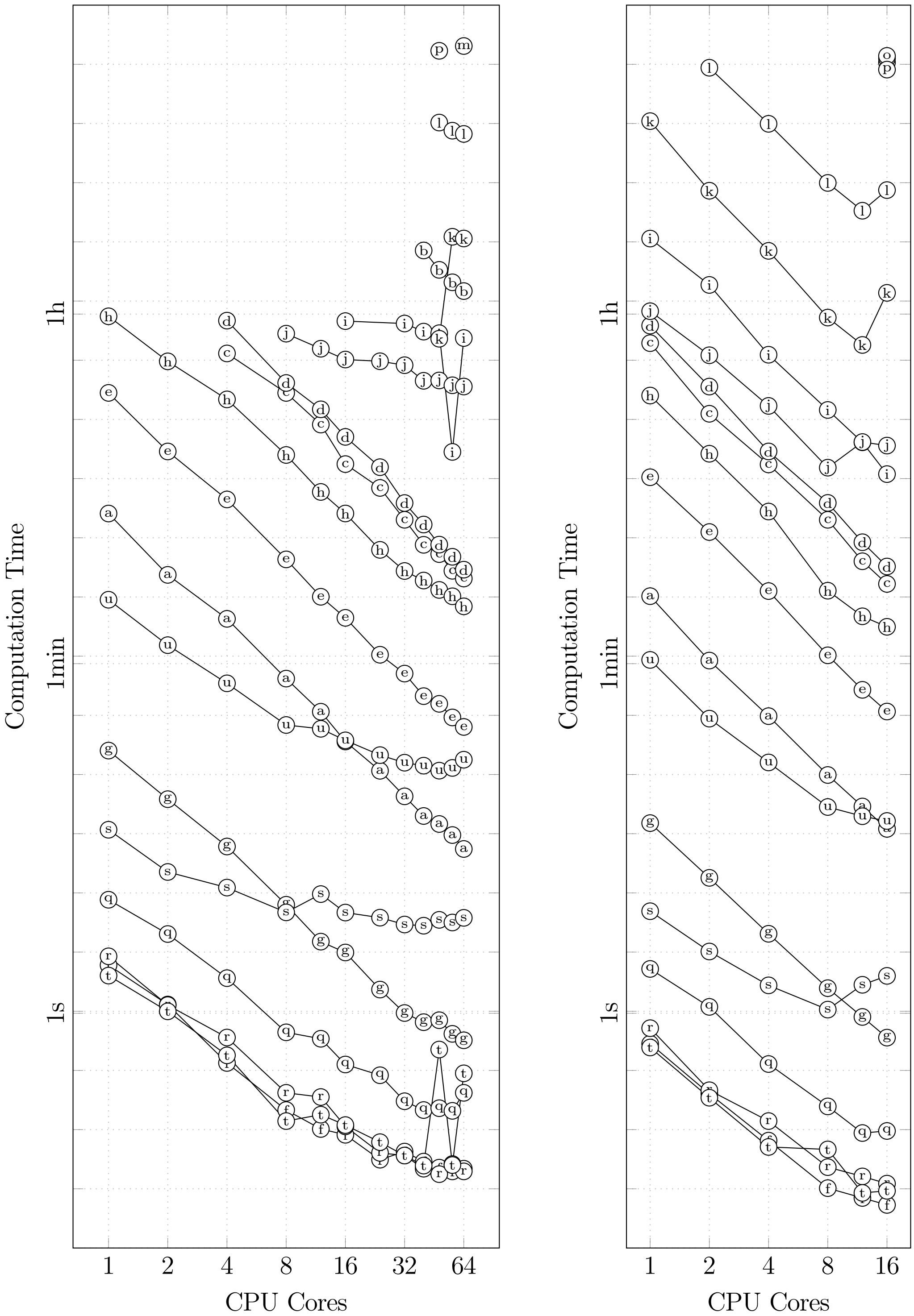

Figure 2. Benchmark Results (left: Atlas, right: Taurus)

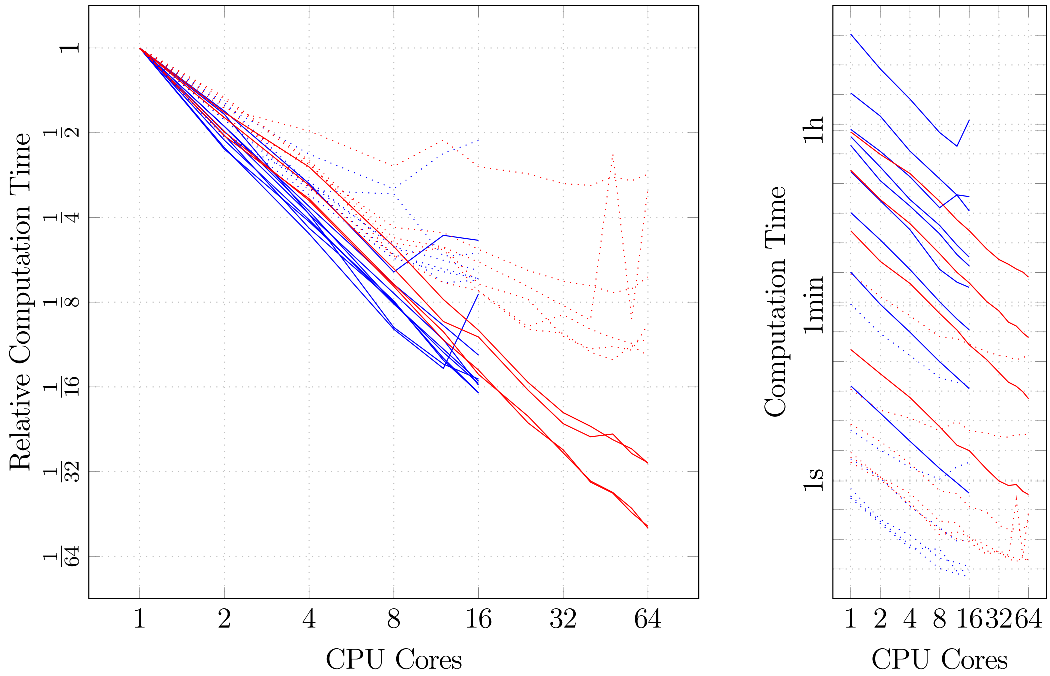

Figure 3. Benchmark Results (red: Atlas, blue: Taurus)

simpler Algorithm 2 without any background knowledge. The execution of Algorithm 1 would surely be slower, and the concrete slow-down depends on the size of the background knowledge $\mathcal{B}$, due to the additional costs for computing closures with respect to the induced closure operator $\cdot^{\mathcal{B}}$, and for computing closures of the supremum of $\cdot^{II}$ and $\cdot^{\mathcal{B}}$, respectively. However, the processing of all candidates with the same cardinality is still independent, i.e., the same scaling behaviour is to be expected when more CPU cores are available and utilized, and of course if the input formal context is large enough.

The benchmark results are displayed in Figures 2 and 3. While in Figure 2 the individual results for the test contexts are tagged by their label as defined in Figure 1, no individual labeling is done in Figure 3. However, solid lines represent large formal contexts with more than 20 attributes and more than 100 objects, and dotted lines denote smaller formal contexts. The charts have both axes logarithmically scaled, to emphasize the correlation between the execution times and the number of available CPU cores. We can see that the computation time is almost inverse linear proportional to the number of available CPU cores, provided that the context is large enough, meaning there are enough candidates on each cardinality level for the computation to be done in parallel. However, we note that there are some cases where the computation times increase when utilizing all available CPU cores. We are currently not aware of an explanation for this exception, but we conjecture that this is due to some technical details of the platforms or the operation systems, e.g., some background tasks that are executed during the benchmark, or overhead caused by thread maintenance. Note that we did not have full system access during the experiments, but could only execute tasks by scheduling them in a batch system. Additionally, for some of the test contexts only benchmarks for a large number of CPU cores could be performed, due to the time limitations on the test systems.

Furthermore, we have performed the same benchmark with small-sized contexts having at most 15 attributes. The computation times were far below one second. We have noticed that there is a certain number of available CPU cores for which there is no further increase in speed of the algorithm. This happens when the number of candidates is smaller than that of the available CPU cores.

Finally, we compared our two implementations of `NextClosure` and `NextClosures` when only one CPU core is utilized. The comparison was performed on a notebook with

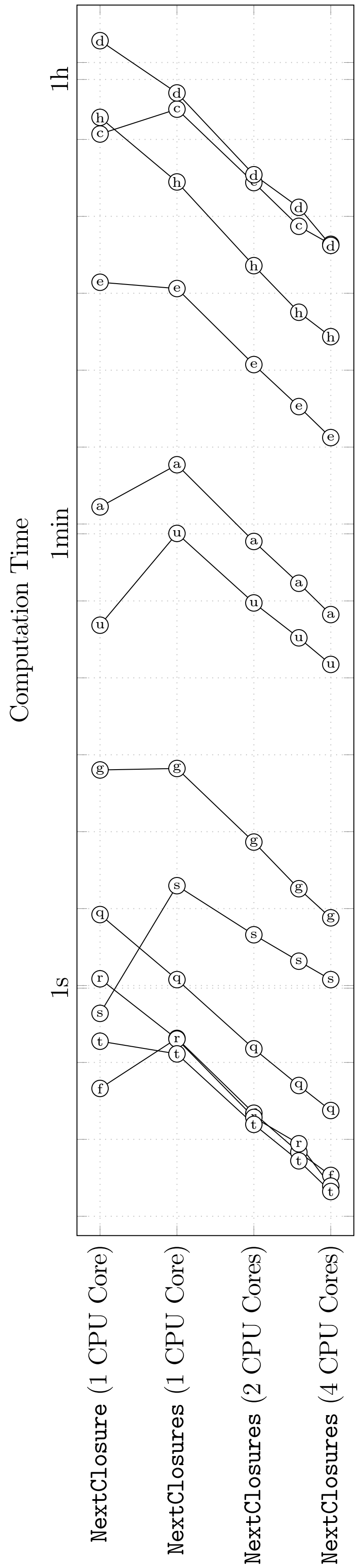

Figure 4. Performance Comparison

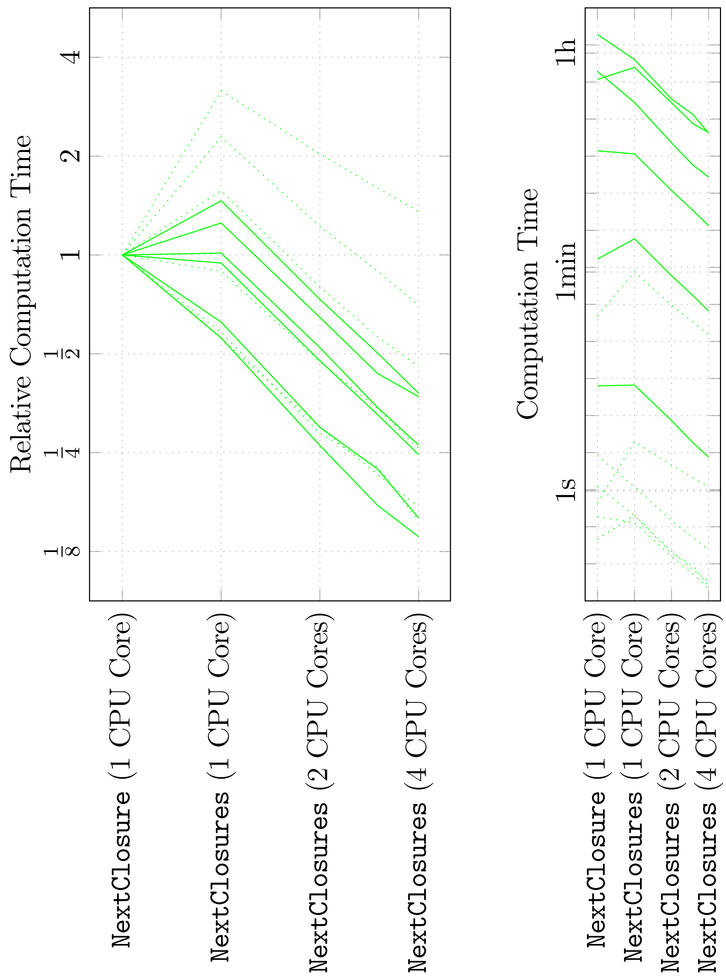

Figure 5. Performance Comparison

Intel Core i7-3720QM CPU with four cores @ 2.6 GHz and 8 GB RAM. The results are shown in Figures 4 and 5. We conclude that our proposed algorithm is on average as fast as `NextClosure` on the test contexts. The computation time ratio is between $\frac{1}{3}$ and 3, depending on the specific context. Low or no speed-ups are expected for formal contexts where `NextClosure` does not have to compute candidate closures in order to find the next, but where it can find the next intent or pseudo-intent immediately. Those formal contexts exist and some of them have been used in our benchmarks.

Please do not take the absolute computation times too seriously, as they can certainly be lowered by utilizing other more efficient data structures, or faster programming languages. For example, `NextClosures` was reimplemented in *Concept Explorer FX* (Kriegel 2010–2017), and the new version essentially operates on `java.util.BitSet`s. Due to its smaller memory footprint and the faster execution of its methods (compared to `java.util.HashSet`), the absolute computation times were reduced by a factor of approximately 10.

6. Conclusion

In this paper we have introduced the parallel algorithm `NextClosuresWithBackground-Knowledge` for the computation of the canonical base of a formal context with respect to a set of background implications. It constructs the set of all intents and pseudo-intents of a given formal context from bottom to top in a level-wise order and increasing cardinality. As the elements in a certain level of the corresponding lattice can be computed independently, they can also be enumerated in parallel, thus yielding a parallel algorithm for computing the canonical base. Indeed, first benchmarks show that `NextClosures` allows for a speed-up that is proportional to the number of available CPU cores, up to a certain natural limit. Furthermore, we have compared its performance to the well-known algorithm `NextClosure` when utilizing only one CPU core. It could be observed that on average our algorithm (on one CPU core) has the same performance as `NextClosure`, at least for the test contexts.

A famous algorithm from Formal Concept Analysis, which is based on the classical `NextClosure` algorithm, is *attribute exploration* (Ganter and Obiedkov 2016). This algorithm allows a domain expert to examine a data set for *completeness*, i.e., to assess whether for each object in a certain *target domain* a representative is present in the data set in question. The algorithm achieves this by trying to find differences between the implicational theory of the current data set and the target domain. If some such implication is found, the domain expert is asked whether it is indeed valid in the target domain. If not, then the data set is incomplete, and the expert is required to present some counterexample that is added to it.

Attribute exploration has a number of nice applications, but those applications are mostly concerned with rather small or purely mathematical domains. This is because the algorithm itself does not scale very well for large data sets, as it is based on the sequential `NextClosure` algorithm to compute the next implication. However, Kriegel (2016b) shows how our parallel `NextClosures` algorithm can be extended to a parallel attribute exploration algorithm. This takes away one important obstacle for the practicability of attribute exploration. Hence, as a logical next step in our line of research, real-world applications that were previously inaccessible to attribute exploration shall be sought and investigated.

Acknowledgements

The authors gratefully thank Bernhard Ganter for his helpful hints on optimal formal contexts for his `NextClosure` algorithm. Furthermore, the authors greatly appreciate the constructive hints and helpful remarks of the anonymous reviewers.

The benchmarks were performed on servers at the Institute of Theoretical Computer Science, and the Centre for Information Services and High Performance Computing (ZIH) at TU Dresden. We thank them both for their generous allocations of computer time.

References

Belohlávek, Radim, and Vilém Vychodil. 2006. "Formal Concept Analysis with Constraints by Closure Operators." In *Conceptual Structures: Inspiration and Application, 14th International Conference on Conceptual Structures, ICCS 2006, Aalborg, Denmark, July 16-21, 2006, Proceedings,* Vol. 4068 of *Lecture Notes in Computer Science* edited by Henrik Schärfe, Pascal Hitzler, and Peter Øhrstrøm. 131–143. Springer.

Belohlávek, Radim, and Vilém Vychodil. 2013. "Closure-Based Constraints in Formal Concept Analysis." *Discrete Applied Mathematics* 161 (13-14): 1894–1911.

Borchmann, Daniel, Felix Distel, and Francesco Kriegel. 2016. "Axiomatisation of General Concept Inclusions from Finite Interpretations." *Journal of Applied Non-Classical Logics* 26 (1): 1–46.

Distel, Felix. 2010. "Hardness of Enumerating Pseudo-Intents in the Lectic Order." In *Formal Concept Analysis, 8th International Conference, ICFCA 2010, Agadir, Morocco, March 15-18, 2010. Proceedings,* Vol. 5986 of *Lecture Notes in Computer Science* edited by Léonard Kwuida and Barış Sertkaya. 124–137. Springer.

Distel, Felix. 2011. "Learning Description Logic Knowledge Bases from Data using Methods from Formal Concept Analysis." Ph.D. thesis. Technische Universität Dresden.

Distel, Felix, and Barış Sertkaya. 2011. "On the Complexity of Enumerating Pseudo-Intents." *Discrete Applied Mathematics* 159 (6): 450–466.

FCA Data Repository. offline since 2013. `http://www.fcarepository.com`.

Fu, Huaiguo, and Engelbert Mephu Nguifo. 2004. "A Parallel Algorithm to Generate Formal Concepts for Large Data." In *Concept Lattices, Second International Conference on Formal Concept Analysis, ICFCA 2004, Sydney, Australia, February 23-26, 2004, Proceedings,* Vol. 2961 of *Lecture Notes in Computer Science* edited by Peter W. Eklund. 394–401. Springer.

Ganter, Bernhard. 1984. *Two Basic Algorithms in Concept Analysis.* FB4-Preprint 831. Darmstadt, Germany: Technische Hochschule Darmstadt.

Ganter, Bernhard. 1999. "Attribute Exploration with Background Knowledge." *Theoretical Computer Science* 217 (2): 215–233.

Ganter, Bernhard. 2010. "Two Basic Algorithms in Concept Analysis." In *Formal Concept Analysis, 8th International Conference, ICFCA 2010, Agadir, Morocco, March 15-18, 2010. Proceedings,* Vol. 5986 of *Lecture Notes in Computer Science* edited by Léonard Kwuida and Barış Sertkaya. 312–340. Springer.

Ganter, Bernhard, and Sergei A. Obiedkov. 2016. *Conceptual Exploration.* Springer.

Ganter, Bernhard, and Rudolf Wille. 1999. *Formal Concept Analysis - Mathematical Foundations.* Springer.

Guigues, Jean-Luc, and Vincent Duquenne. 1986. "Famille minimale d'implications informatives résultant d'un tableau de données binaires." *Mathématiques et Sciences Humaines* 95: 5–18.

Kriegel, Francesco. 2010–2017. "Concept Explorer FX." Software for Formal Concept Analysis with Description Logic Extensions. `https://github.com/francesco-kriegel/conexp-fx`.

Kriegel, Francesco. 2015. *NextClosures – Parallel Exploration of Constrained Closure Operators.* LTCS-Report 15-01. Chair for Automata Theory, Technische Universität Dresden.

Kriegel, Francesco. 2016a. "Axiomatization of General Concept Inclusions from Streams of Interpretations with Optional Error Tolerance." In *Proceedings of the 5th International Workshop "What can FCA do for Artificial Intelligence"? co-located with the European Conference on Artificial Intelligence, FCA4AI@ECAI 2016, The Hague, the Netherlands, August 30, 2016.*, Vol. 1703 of *CEUR Workshop Proceedings* edited by Sergei O. Kuznetsov, Amedeo Napoli, and Sebastian Rudolph. 9–16. CEUR-WS.org.

Kriegel, Francesco. 2016b. "Parallel Attribute Exploration." In *Graph-Based Representation and Reasoning - 22nd International Conference on Conceptual Structures, ICCS 2016, Annecy, France, July 5-7, 2016, Proceedings,* Vol. 9717 of *Lecture Notes in Computer Science* edited by Ollivier Haemmerlé, Gem Stapleton, and Catherine Faron-Zucker. 91–106. Springer.

Kriegel, Francesco, and Daniel Borchmann. 2015. "NextClosures: Parallel Computation of the Canonical Base." In *Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications, Clermont-Ferrand, France, October 13-16, 2015.*, Vol. 1466 of *CEUR Workshop Proceedings* edited by Sadok Ben Yahia and Jan Konecny. 181–192. CEUR-WS.org.

Kuznetsov, Sergei O. 2004. "On the Intractability of Computing the Duquenne-Guigues Base." *Journal of Universal Computer Science* 10 (8): 927–933.

Lindig, Christian. 2000. "Fast Concept Analysis." In *Working with Conceptual Structures – Contributions to ICCS 2000,* edited by Gerhard Stumme. 152–161. Aachen, Germany: Shaker Verlag. Supplemental Proceedings to Conceptual Structures: Logical, Linguistic, and Computational Issues, 8th International Conference on Conceptual Structures, ICCS 2000, Darmstadt, Germany, August 14-18, 2000.

Obiedkov, Sergei A., and Vincent Duquenne. 2007. "Attribute-Incremental Construction of the Canonical Implication Basis." *Annals of Mathematics and Artificial Intelligence* 49 (1-4): 77–99.

Stumme, Gerd. 1996. *Attribute Exploration with Background Implications and Exceptions.* 457–469. Studies in Classification, Data Analysis, and Knowledge Organization. Berlin, Heidelberg: Springer.

Vychodil, Vilém, Petr Krajča, and Jan Outrata. 2008. "Parallel Recursive Algorithm for FCA." In *Proceedings of the 6th International Conference on Concept Lattices and Their Applications,* edited by Radim Bělohlávek and Sergei O. Kuznetsov. 71–82. Palacký University, Olomouc.

Wille, Rudolf. 1982. *Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts.* 445–470. Dordrecht: Springer Netherlands.