Discrete Mathematics and Theoretical Computer Science **6**, 2004, 315–338

Efficient Algorithms on the Moore Family Associated to an Implicational System

Karell Bertet[1] and Mirabelle Nebut[2]

[1]L3I, *Université de La Rochelle, Pôle Sciences et Technologies, av Michel Crépeau, 17042 La Rochelle Cédex 1, France*
`karell.bertet@univ-lr.fr`
[2]LIFL, *Université des Sciences et Technologies de Lille, Bâtiment M3, 59655 Villeneuve d'Ascq Cédex, France*
`mirabelle.nebut@lifl.fr`

received Jun 7, 2002, *revised Jun 13, 2002, Jun 28, 2003*, *accepted Dec 18, 2003*.

An implication system (IS) Σ on a finite set S is a set of rules called Σ-implications of the kind $A \rightarrow_\Sigma B$, with $A, B \subseteq S$. A subset $X \subseteq S$ satisfies $A \rightarrow_\Sigma B$ when "$A \subseteq X$ implies $B \subseteq X$" holds, so ISs can be used to describe constraints on sets of elements, such as dependency or causality. ISs are formally closely linked to the well known notions of closure operators and Moore families. This paper focuses on their algorithmic aspects. A number of problems issued from an IS Σ (e.g. is it minimal, is a given implication entailed by the system) can be reduced to the computation of closures $\varphi_\Sigma(X)$, where φ_Σ is the closure operator associated to Σ. We propose a new approach to compute such closures, based on the characterization of the direct-optimal IS Σ_{do} which has the following properties: 1. it is equivalent to Σ 2. $\varphi_{\Sigma_{do}}(X)$ (thus $\varphi_\Sigma(X)$) can be computed by a single scanning of Σ_{do}-implications 3. it is of minimal size with respect to ISs satisfying 1. and 2. We give algorithms that compute Σ_{do}, and from Σ_{do} closures $\varphi_\Sigma(X)$ and the Moore family associated to φ_Σ.

Keywords: Moore family, implicational system, closure operator, algorithm, lattice.

1 Introduction

As recalled in [CM04], the basic mathematical notion of *closure operator* (an isotone, extensive and idempotent map φ) defined on a poset $(P, \leq)$ is fundamental in a number of fields linked to computer science, in particular when defined on the lattice $(2^S, \subseteq)$ of all subsets of a finite set S. In this case, closure operators are closely linked to the notion of *Moore family*, a family $\mathbb{F} \subseteq 2^S$ which contains S and is closed under intersection (see [CM04] for more details). The notions of closure operator and Moore family both involve the concept of logical or entail implication, used for instance in knowledge systems or relational data-bases (these fields handle systems of implications, called for example functional dependencies in relational data-bases [MR92, Mai83], and association rules in data-mining [PBTL99]). Hence the notion of *Implicational System* (IS for short) defined in [CM04], to which is dedicated this paper.

Formally an IS on S denoted by $\Sigma \subseteq 2^S \times 2^S$ is a set of rules called Σ-*implications* of the kind $A \rightarrow_\Sigma B$, with $A, B \subseteq S$. A subset $X \subseteq S$ satisfies an implication $A \rightarrow_\Sigma B$ when "$A \subseteq X$ implies $B \subseteq X$". So ISs can be used to easily describe constraints between sets of elements, such as dependency or causality. Let us give here an intuitive example which will also be used in the core of the paper (see Ex. 1 in Sect. 3). Assume that $S = \{a,b,c,d,e\}$ is a set of events. The IS $\Sigma = \{a \to b, ac \to d, e \to a\}^\dagger$ can be interpreted as "if a resp. e occurs then so does b resp. a, and if a and c occur then so does d".

Given such a system, several types of questions arise. A common problem is to find a minimum "full" system of implications, from which all implications between elements can be obtained. Another very natural issue is for instance the question "is it possible that a and e occur and not c ?". One can answer using either the *implicational* Moore family *associated to* Σ ($\mathbb{F}_\Sigma$ contains all subsets $X \subseteq S$ that satisfy each Σ-implication) or the *closure operator associated to* $\mathbb{F}_\Sigma$ ($\varphi_{\mathbb{F}_\Sigma}$ maps a subset $X \subseteq S$ on the least element $F \in \mathbb{F}_\Sigma$ s.t. $X \subseteq F$). In our example the answer is "yes" because $abe \in \mathbb{F}_\Sigma$ and $c \notin ae$, or because $c \notin \varphi_{\mathbb{F}_\Sigma}(ae) = abe$. Answering questions about a system using the *closure* $\varphi_{\mathbb{F}_\Sigma}(X)$ has a great advantage: it avoids the construction of the whole Moore family (which contains 14 elements in our example). Moreover $\varphi_{\mathbb{F}_\Sigma}$ can also be used to compute efficiently $\mathbb{F}_\Sigma$, whose direct definition-based generation relies upon an exponential enumeration of all subsets of S. Note that data-mining has to deal with a reverse problem adressing the efficient generation of association rules from a family of closures called *itemsets* [PBTL99].

The properties of implicational Moore families and ISs have been studied in [GD86, Wil94, Wil95, CM04] from a theoretical point of view. This paper focuses on algorithmic issues. Following the intuition given before, it is based on the efficient computation of $\varphi_{\mathbb{F}_\Sigma}(X)$. As detailed in the core of the paper, this computation was addressed in several ways in [Mai83, MR92, Wil95]: $\varphi_{\mathbb{F}_\Sigma}(X)$ is obtained by several enumerations of the implications of Σ. For instance in the previous example the computation of $\varphi_{\mathbb{F}_\Sigma}(ae) = abe$ is performed‡ by scanning once the Σ-implications (first and third implications) but the computation of $\varphi_{\mathbb{F}_\Sigma}(ce) = abcde$ is performed by scanning them twice: The first enumeration brings $ace \in \varphi_{\mathbb{F}_\Sigma}(ce)$ (third implication) and the second one brings $bd \in \varphi_{\mathbb{F}_\Sigma}(ce)$ (first and second implications).

The new approach we propose is based on two fundamental algorithmic observations: 1. the computation of $\varphi_{\mathbb{F}_\Sigma}(X)$ is more efficient when Σ is *optimal*, where optimal means "of minimal size"; 2. the enumeration number of Σ-implications needed to compute $\varphi_{\mathbb{F}_\Sigma}(X)$ can be reduced to 1 when Σ is *direct*. Let us illustrate it on our example. The IS $\Sigma_d = \Sigma \cup \{e \to b, ce \to d\}$ is direct and equivalent to Σ (it is easy to check that $\varphi_{\mathbb{F}_{\Sigma_d}}(ce)$ can be now computed by a single scanning of Σ_d-implications). It is not optimal. The IS $\Sigma_o = \{e \to ab, ac \to d, a \to b, ce \to d\}$ is similarly equivalent to Σ and direct, but also direct-optimal in the sense that there exists no equivalent direct IS of smaller size (though $\Sigma_o \not\subseteq \Sigma_d$). Our approach also consists in computing $\varphi_{\mathbb{F}_\Sigma}(X)$ (hence $\mathbb{F}_\Sigma$) by exploiting the directness and optimality properties: We define the *direct-optimal IS* Σ_{do} *generated from* Σ. First Σ is completed by some implications into the direct IS Σ_d, then Σ_d is modified into the optimal IS Σ_{do} (Σ, Σ_d and Σ_{do} being equivalent).

The paper is organized as follows: Sect. 2 gives notations and standard definitions. Section 3 first gives some preliminaries on the computation of $\varphi_{\mathbb{F}_\Sigma}(X)$ (Sect. 3.1) then defines the notion of direct IS and characterizes the *direct IS* Σ_d *generated from* Σ (Sect. 3.2). In the same way, Sect. 3.3 defines the notion of direct-optimal IS and characterizes the *direct-optimal IS* Σ_o *generated from a direct IS* Σ. By combination of these two definitions, we naturally obtain the *direct-optimal IS* Σ_{do} *generated from* a given IS Σ (Sect. 3.4).

Section 4 deals with algorithmic aspects of the above result. We first describe an efficient data structure

† We abuse notations and write ac for $\{a,c\}$.

‡ At this stage the reader should admit the following recipe: Initialize $\varphi_{\mathbb{F}_\Sigma}(X)$ with X, then iteratively scan Σ-implications until stabilization doing: If $A \to B \in \Sigma$ and $A \subseteq \varphi_{\mathbb{F}_\Sigma}(X)$ then add B to $\varphi_{\mathbb{F}_\Sigma}(X)$.

introduced in [Gan84, HN96, NR99] and called *lexicographic tree*, traditionally used to represent families and extended here to represent ISs (Sect. 4.1). We then give an algorithm to compute the closure $\varphi_{\mathbb{F}_\Sigma}(X)$ from a direct-optimal IS (Sect. 4.2), and an algorithm to compute the *direct-optimal IS* Σ_{do} *generated from* some IS Σ, where Σ and Σ_{do} are represented by a lexicographic tree. We finally propose an algorithm to generate $\mathbb{F}_\Sigma$ (Sect. 4.3), based on properties of the lattice $(\mathbb{F}_\Sigma, \subseteq)$.

2 Definitions and Notations

Let us consider a finite set of elements S. A *family* $\mathcal{F}$ on S is a set of subsets of S: $\mathcal{F} \subseteq 2^S$. A *Moore family* $\mathbb{F}$ on S is a family stable by intersection and which contains S: $S \in \mathbb{F}$ and $F_1, F_2 \in \mathbb{F}$ implies $F_1 \cap F_2 \in \mathbb{F}$. The poset $(\mathbb{F}, \subseteq)$ is a lattice with, for each $F_1, F_2 \in \mathbb{F}$, $F_1 \wedge F_2 = F_1 \cap F_2$ and $F_1 \vee F_2 = \bigcap\{F \in \mathbb{F} \mid F_1 \cup F_2 \subseteq F\}$ (recall that a lattice is an order relation (i.e. reflexive, antisymmetric and transitive) over a set of elements such that any pair x, y of elements has a *join* (i.e. a least upper bound) denoted by $x \vee y$, and a *meet* (i.e. a greatest lower bound) denoted by $x \wedge y$).

Let X, X' be subsets of S. A *closure operator* φ on S is a map on 2^S which is isotone ($X \subseteq X'$ implies $\varphi(X) \subseteq \varphi(X')$), extensive ($X \subseteq \varphi(X)$) and idempotent ($\varphi^2(X) = \varphi(X)$). $\varphi(X)$ is called the *closure* of X by φ. X is said to be *closed* by φ whenever it is a fixed point for φ, i.e. $\varphi(X) = X$.

The set of all Moore families and the set of all closure operators on S are in a one-to-one correspondence. The Moore family $\mathbb{F}_\varphi$ associated to the closure operator φ is the set of all closed elements of φ:

$$\mathbb{F}_\varphi = \{F \subseteq S \mid F = \varphi(F)\} \tag{1}$$

The closure operator $\varphi_\mathbb{F}$ associated to the Moore family $\mathbb{F}$ is such that, for any $X \subseteq S$, $\varphi_\mathbb{F}(X)$ is the least element $F \in \mathbb{F}$ that contains X:

$$\varphi_\mathbb{F}(X) = \bigcap\{F \in \mathbb{F} \mid X \subseteq F\} \tag{2}$$

In particular $\varphi_\mathbb{F}(\emptyset) = \bot_\mathbb{F}$. Note that $\varphi_\mathbb{F}(X) \in \mathbb{F}$ because Moore families are closed by intersection. Moreover for all $F_1, F_2 \in \mathbb{F}$, $F_1 \vee F_2 = \varphi_\mathbb{F}(F_1 \cup F_2)$ and $F_1 \wedge F_2 = \varphi_\mathbb{F}(F_1 \cap F_2) = F_1 \cap F_2$.

Let A, B be subsets of S. An *Implicational System* (IS for short) Σ on S is a binary relation on 2^S: $\Sigma \subseteq 2^S \times 2^S$. A couple $(A, B) \in \Sigma$ is called a Σ-*implication* whose *premise* is A and *conclusion* is B. It is written $A \rightarrow_\Sigma B$ or $A \rightarrow B$ (meaning "A implies B"). The family $\mathbb{F}_\Sigma$ on S associated to Σ is:

$$\mathbb{F}_\Sigma = \{X \subseteq S \mid A \subseteq X \Rightarrow B \subseteq X \text{ for each } A \rightarrow B \in \Sigma\} \tag{3}$$

i.e. it is the set of sets $X \subseteq S$ such that "X contains A implies X contains B". $\mathbb{F}_\Sigma$ is clearly a Moore family called the *implicational Moore family on S associated to* Σ. Several ISs can describe the same Moore family: Σ and Σ' on S are *equivalent* if $\mathbb{F}_\Sigma = \mathbb{F}_{\Sigma'}$. The problem is to find the smallest ones, according to various criteria [Wil94]. Σ is *non-redundant* if $\Sigma \setminus \{X \rightarrow Y\}$ is not equivalent to Σ, for all $X \rightarrow Y$ in Σ. It is *minimum* if $|\Sigma| \leq |\Sigma'|$ for all IS Σ' equivalent to Σ. Σ is *optimal* if $s(\Sigma) \leq s(\Sigma')$ for all IS Σ' equivalent to Σ, where $s(\Sigma)$ is the *size* of Σ defined by:

$$s(\Sigma) = \sum_{A \rightarrow B \in \Sigma} (|A| + |B|) \tag{4}$$

Other definitions not recalled here can be found in the survey of Caspard and Monjardet [CM04].

In the following, S is endowed with a total order $<_\alpha$ or simply α. A subset $X = \{x_1, x_2, \ldots, x_n\}$ is viewed as the word $x_{j_1} x_{j_2} \ldots x_{j_n}$ sorted according to α: $x_{j_1} <_\alpha x_{j_2} <_\alpha \cdots <_\alpha x_{j_n}$. Σ is an IS on S, $\mathbb{F}_\Sigma$ or $\mathbb{F}$ is the Moore family associated to Σ, and $\varphi_{\mathbb{F}_\Sigma}$ or φ_Σ or simply φ is the induced closure operator.

3 Characterization of φ_Σ from Σ

As explained in introduction, a number of problems related to an IS Σ can be answered by computing closures of the kind $\varphi_\Sigma(X)$, for some $X \subseteq S$. Section 3.1 presents important notions used further and introduces our method: The idea is to perform the computation of $\varphi_\Sigma(X)$ not on Σ but on another equivalent IS which makes the computation more efficient. Section 3.2 defines such convenient and equivalent IS, called *direct*. Section 3.3 characterizes the smallest equivalent direct IS inferred from a direct one, called *direct-optimal*. Finally Sect. 3.4 characterizes the direct-optimal IS equivalent to some IS Σ.

3.1 Preliminaries

A direct and naive computation of φ_Σ (or simply φ) follows from equations (2) and (3):

$$\begin{aligned}\varphi(X) = \bigcap\{X' \subseteq S \mid X \subseteq X' \text{ and}\\ A \subseteq X' \text{ implies } B \subseteq X' \text{ for each } A \rightarrow_\Sigma B\}\end{aligned} \tag{5}$$

It requires an enumeration of all subsets X' such that $X \subseteq X' \subseteq S$, plus a test on the premise and conclusion of each implication. Moreover these enumerations must be done for each particular X under consideration.

[Wil94, Wil95] propose a definition of $\varphi(X)$ which induces a more efficient computation:

$$\varphi(X) = X^{\Sigma^{\Sigma^{\cdot^{\cdot^{\cdot^{\Sigma}}}}}} \tag{6}$$

where

$$X^\Sigma = X \cup \bigcup\{B \mid A \subseteq X \text{ and } A \rightarrow_\Sigma B\} \tag{7}$$

According to [Wil95] $\varphi(X)$ is in this way obtained in $O(|S|^2|\Sigma|)$ by iteratively scanning Σ-implications: $\varphi(X)$ is initialized with X then increased with B for each implication $A \rightarrow_\Sigma B$ such that $\varphi(X)$ contains A. The computation cost depends on the number of iterations, in any case bounded by $|S|$. In order to practically limit this number (keeping the same complexity), [Wil95] tunes algorithms using additional data structures.

It is worth noting that for some particular ISs the computation of φ requires only one iteration. Such an IS is called *direct* (one can also find *iteration-free* in [Wil94]):

Definition 1 *An IS Σ is* direct *if, for all $X \subseteq S$:*

$$\varphi(X) = X^\Sigma = X \cup \bigcup\{B \mid A \subseteq X \textit{ and } A \rightarrow_\Sigma B\} \tag{8}$$

Instead of tuning algorithms applied to some IS Σ, a possible approach is to infer from Σ an equivalent and direct IS Σ'. Once it is done, each closure $\varphi(X)$ can be computed by simply enumerating Σ'-implications. As an illustration, let us consider *full* ISs, that are a classical type of direct ISs.

According to [CM04] (Def. 49 p. 20), a *full IS* is a preorder (a reflexive and transitive relation) that contains the preorder $\supseteq$ on $2^S \times 2^S$ and is $\cup$-stable, that is it verifies the property:

$$\text{for all } A, B, C, D \subseteq S, A \rightarrow B \text{ and } C \rightarrow D \text{ imply } A \cup C \rightarrow B \cup D$$

As stated by Prop. 1, a full IS is direct.

Proposition 1 (Corollary 53 in [CM04]) *For Σ a full IS,*

$$\varphi(X) = \bigcup\{B \subseteq S \mid X \rightarrow_\Sigma B\} = X^\Sigma$$

Starting from the notion of full IS, and given some IS Σ, we define the *full IS* Σ_f inferred from Σ, *equivalent* to Σ (Prop. 2), and direct (Prop. 1): it contains all Σ-implications, all implications due to inclusions in $2^S \times 2^S$, and all implications generated by Σ-implications and inclusions.

Definition 2 *The* full *IS Σ_f inferred from Σ is defined as the smallest*[§] *ISs.t.:*

1. $\Sigma \subseteq \Sigma_f$ *and*
2. Σ_f *verifies the three following properties: For all $A,B,C,D \subseteq S$,*

   P1 (inclusion axiom): $B \subseteq A$ *implies* $A \rightarrow_{\Sigma_f} B$

   P2 (transitivity axiom): $A \rightarrow_{\Sigma_f} B$ *and* $B \rightarrow_{\Sigma_f} C$ *implies* $A \rightarrow_{\Sigma_f} C$

   P3 (union axiom): $A \rightarrow_{\Sigma_f} B$ *and* $C \rightarrow_{\Sigma_f} D$ *implies* $A \cup C \rightarrow_{\Sigma_f} B \cup D$

Proposition 2 Σ *and* Σ_f *are equivalent.*

For completeness, we give the proof of this simple result.
Proof: Let us prove that $\mathbb{F}_\Sigma = \mathbb{F}_{\Sigma_f}$.
$\supseteq$. Immediate since $\Sigma \subseteq \Sigma_f$.
$\subseteq$. Consider $F \in \mathbb{F}_\Sigma$. It is easy to check by induction that F satisfies "$A \subseteq F$ implies $B \subseteq F$" for any $A \rightarrow_{\Sigma_f} B$ induced by P_1, P_2 and P_3. □

Using Σ_f, one can compute a closure $\varphi_\Sigma(X)$ in only one iteration. Nevertheless note that the directness of Σ_f is due to the fact that any subset $A \subseteq S$ appears as a premise of a Σ_f-implication: it makes the computation of Σ_f exponential thus impracticable. The idea is then to look for smaller ISs, not necessarily full, but still direct and equivalent to Σ (and Σ_f). The smallest such one is called *direct-optimal*.

Definition 3 *An IS Σ is* direct-optimal *if it is direct, and if $s(\Sigma) \leq s(\Sigma')$ for any direct IS Σ' equivalent to Σ.*

Our approach can be summarized as follows. Given some IS Σ:

- We start from the three axioms that describe a full IS (cf. Def. 2) to define in Sect. 3.2 the *direct* IS Σ_d *inferred from* Σ, whose directness is stated by Th. 1;
- Consider Σ is direct but perhaps not optimal: In this case some Σ-implications can be removed or simplified, while preserving the directness and semantics of Σ. In Sect. 3.3 we first formally characterize *direct-optimal* ISs (Th. 2) then, given a *direct* IS Σ, we define the *direct-optimal* IS Σ_o inferred from Σ.
- By combination of these two results, we obtain the definition of the *direct-optimal IS Σ_{do} inferred from some IS* Σ. Moreover, we state that equivalent ISs define an unique direct-optimal IS (Corollary 1). Closures $\varphi_\Sigma(X)$ can then be computed by only one enumeration of Σ_{do}-implications, at a minimal cost[¶].

§ "Smallest" for the preorder $\subseteq$.

¶ "Minimal" in the sense that using any other equivalent direct IS would be less efficient; Nevertheless in the cases where few closures are needed, or where a small non-direct IS is considered, it may be more efficient to iterate Σ-enumerations instead of computing Σ_d then Σ_{do}.

3.2 Σ_d: a Direct IS Generated from an IS Σ

In this section we define an IS smaller than Σ_f, but still direct and equivalent to Σ. To do so, let us consider again the three axioms that characterize Σ_f (Def. 2), and let us explain what Σ_f-implications can be removed without altering the directness and semantics of the IS, or dually what implications must necessarily be added to Σ. We consider the computation of $\varphi(X)$ indicated by (6), for $X \subseteq S$.

Given a pair of implications (I_1, I_2) present in the IS under construction, the principle is to "summarize" via a third implication the result of the $\varphi(X)$ iterative computation process applied to (I_1, I_2). Axioms P_2 and P_3 do apply this principle. Nevertheless the inferred implications (included these inferred by P_1 are sometimes clearly redundant with properties particular to the closure operator φ. It is the case when no iterative process is needed, because X contains both the implications premises:

1. Assume $A \subseteq X$. The implication $A \rightarrow_{\Sigma_f} B$ stated by P_1 is redundant: it causes the explicit enrichment of $\varphi(X)$ with B while according to Eq. (7) (and due to the φ extensiveness) we have $\varphi(X) \supseteq X$, and $X \supseteq A \supseteq B$.
2. Assume $A, B \subseteq X$. The implication $A \rightarrow_{\Sigma_f} C$ stated by P_2 is redundant with $B \rightarrow_{\Sigma_f} C$, which already states the enrichment of $\varphi(X)$ with C.
3. Assume $A, C \subseteq X$. Similarly the implication $A \cup C \rightarrow_{\Sigma_f} B \cup D$ stated by P_3 is redundant with $A \rightarrow_{\Sigma_f} B$ and $C \rightarrow_{\Sigma_f} D$.

When an iterative process is required to compute $\varphi(X)$, implications inferred by a combination of the three axioms are necessary. For example let us consider the following IS Σ:

$$\{ac \rightarrow_{\Sigma} d \quad , \quad e \rightarrow_{\Sigma} a\}$$

Assume $X = ce$. The computation of $\varphi(X) = acde$ through Σ requires an iterative process: The fact $d \in \varphi(X)$ is known from the first implication only when the intermediate $\varphi(X)$ has been enriched with a (second implication). To be direct, the IS must contain the implication:

$$ce \rightarrow_{\Sigma} d$$

obtained by applying successively:

- P_1 to infer the implication $c \rightarrow c$;
- P_3 applied to $c \rightarrow c$ and $e \rightarrow_{\Sigma} a$ to infer $ce \rightarrow ac$;
- P_2 applied to $ce \rightarrow ac$ and $ac \rightarrow_{\Sigma} d$ to infer $ce \rightarrow_{\Sigma} d$.

Nevertheless implications $c \rightarrow c$ and $ce \rightarrow ac$ are redundant with others, as explained below. To avoid this redundancy, let us consider the pair

$$\{A \rightarrow_{\Sigma} B \ \ , \ \ C \rightarrow_{\Sigma} D\} \tag{9}$$

In the case where the computation of $\varphi(X)$ requires an iteration: $A \subseteq X$ but $C \not\subseteq X$. Because $A \subseteq X$, the first implication adds B to $\varphi(X)$. Now if $C \subseteq X \cup B$, the second implication adds D to $\varphi(X)$. Since $A \subseteq X$ and $C \subseteq X \cup B$ is equivalent to $A \cup (C \setminus B) \subseteq X$, we can summarize this reasoning by the implication (10):

$$A \cup (C \setminus B) \rightarrow_{\Sigma} D \tag{10}$$

In the previous example, $ce \rightarrow d$ is indeed obtained from the pair $\{e \rightarrow_\Sigma a, ac \rightarrow_\Sigma d\}$.

Note that the implication (10) is redundant with the one $C \rightarrow_\Sigma D$ when $B \cap C = \emptyset$, since it yields $A \cup C \rightarrow D$. This case does not happen here due to the condition $C \not\subseteq X$ ($C \not\subseteq X$ and $C \subseteq X \cup B$ imply $C \cap B \neq \emptyset$): We enforce it by imposing $B \cap C \neq \emptyset$ as the application condition of the rule.

The rule that infers implication (10) from implications (9) (called *overlap axiom* in Def. 4) encompasses the combination of axioms P_2 and P_3, but also P_1: The goal of P_1 is mainly to make appear any subset $A \subseteq S$ as a premise of a Σ_f-implication, in order to compute $\varphi(X)$ by Prop. 1. Since we compute $\varphi(X)$ by Equations (6) and (7) instead, we can drop P_1.

The definition of the *direct IS inferred from* Σ now follows directly from what precedes:

Definition 4 *The* direct implicational system Σ_d *generated from* Σ *is defined as the smallest IS s.t.*

1. $\Sigma \subseteq \Sigma_d$ *and*
2. Σ_d *verifies the following property:*

   P4 (overlap axiom) : *for all* $A, B, C, D \subseteq S$*:*

$$A \rightarrow_{\Sigma_d} B,\ C \rightarrow_{\Sigma_d} D \text{ and } B \cap C \neq \emptyset \text{ imply } A \cup (C \backslash B) \rightarrow_{\Sigma_d} D$$

We now adapt Prop. 1 to characterize φ from Σ_d.

Theorem 1 $\varphi(X) = X^{\Sigma_d} = X \cup \bigcup\{B \mid A \subseteq X \text{ and } A \rightarrow_{\Sigma_d} B\}$

Two lemmas are needed to prove this theorem. Lemma 1 states that $\Sigma_d \subseteq \Sigma_f$, therefore that Σ_d is equivalent to Σ since $\Sigma \subseteq \Sigma_d$ and Σ_f is equivalent to Σ. Lemma 2 is the core of the proof: it states that Σ_d contains all "significant" Σ_f-implications. By "significant" we mean an implication $A \rightarrow_{\Sigma_f} B$ s.t. $A \not\subseteq B$, so that it can add $B \backslash A$ to some $\varphi(X)$ and is not trivially redundant like implications inferred by P1 in Def. 2. Lemma 2 states that any such Σ_f-implication $A \rightarrow_{\Sigma_f} B$ is imitated by a set of Σ_d-implications, where a Σ_d-implication is associated to each $y \in B \backslash A$.

Lemma 1 $\Sigma_d \subseteq \Sigma_f$

Proof: Let $\{X_i \rightarrow Y_i\}_{1 \leq i \leq p}$ be the implications successively added to Σ_d in order to complete Σ by application of P4. We define $\Sigma_0 = \Sigma$, $\Sigma_i = \Sigma_{i-1} \cup \{X_i \rightarrow Y_i\}$ and $\Sigma_p = \Sigma_d$. The proof is by induction on i, with $0 \leq i \leq p$. The base case is obtained by definition and Def 2: $\Sigma_0 = \Sigma \subseteq \Sigma_f$.
Inductive step: For $i \geq 1$, let us prove that $\Sigma_{i-1} \subseteq \Sigma_f$ implies $\Sigma_i \subseteq \Sigma_f$, equivalently that $X_i \rightarrow Y_i \in \Sigma_f$. Since $X_i \rightarrow Y_i$ is added to Σ_{i-1} by application of P4, there exist $A \rightarrow_{\Sigma_{i-1}} B$ and $C \rightarrow_{\Sigma_{i-1}} D$ such that $B \cap C \neq \emptyset$, $X_i = A \cup (C \backslash B)$ and $Y_i = D$. By induction hypothesis $A \rightarrow B \in \Sigma_f$ and $C \rightarrow D \in \Sigma_f$. Then

- From $A \rightarrow B \in \Sigma_f$ (by hypothesis) and $B \rightarrow B \cap C \in \Sigma_f$ (by P1) we deduce from P2 that $A \rightarrow B \cap C \in \Sigma_f$.
- From $A \rightarrow B \cap C \in \Sigma_f$ and $C \backslash B \rightarrow C \backslash B \in \Sigma_f$ (by P1) we deduce from P3 that $A \cup (C \backslash B) \rightarrow (B \cap C) \cup (C \backslash B) = C \in \Sigma_f$.
- From $A \cup (C \backslash B) \rightarrow C \in \Sigma_f$ and $C \rightarrow D \in \Sigma_f$ by hypothesis we deduce from P2 that $A \cup (C \backslash B) \rightarrow D \in \Sigma_f$.

Therefore $X_i \rightarrow Y_i \in \Sigma_f$ and the proof is achieved. □

Lemma 2 *For all $X \rightarrow_{\Sigma_f} Y$ and $y \in Y \backslash X$, there exists $X' \rightarrow_{\Sigma_d} Y'$ such that $X' \subseteq X$ and $y \in Y'$.*

Proof: Let $\{X_i \rightarrow Y_i\}_{1 \leq i \leq p}$ be the implications successively added to Σ_f in order to complete Σ by application of P1, P2 or P3. We define $\Sigma_0 = \Sigma$, $\Sigma_i = \Sigma_{i-1} \cup \{X_i \rightarrow Y_i\}$ and $\Sigma_p = \Sigma_f$. The proof is by induction on i, with $0 \leq i \leq p$.

Base case: Since $\Sigma_0 = \Sigma$ and $\Sigma \subseteq \Sigma_d$: for all $X \rightarrow_{\Sigma_0} Y$ and $y \in Y \backslash X$, the implication $X \rightarrow_{\Sigma_d} Y$ verifies $X \subseteq X$ and $y \in Y$.

Inductive step: For $i \geq 1$, assume the property is proved for Σ_{i-1}, i.e. for all $X \rightarrow_{\Sigma_{i-1}} Y$, for all $y \in Y \backslash X$, there exists $X' \rightarrow_{\Sigma_d} Y'$ such that $X' \subseteq X$ and $y \in Y'$. We consider the implication $X_i \rightarrow Y_i$ and some element $y \in Y_i \backslash X_i$ and show that:

$$\text{there exists } X_i' \rightarrow_{\Sigma_d} Y_i' \text{ s.t. } X_i' \subseteq X_i \text{ and } y \in Y_i' \tag{11}$$

If $Y_i \subseteq X_i$ then $Y_i \backslash X_i = \emptyset$ and (11) is trivially satisfied. Assume $Y_i \not\subseteq X_i$ and let us consider that $X_i \rightarrow Y_i$ has been added to Σ_{i-1} by the application of P2 or P3 (since applying P1 implies that $Y_i \subseteq X_i$, which contradicts the hypothesis). Let us consider successively the application of P3 and P2.

$\star$ *Case P3:* There exist $A \rightarrow_{\Sigma_{i-1}} B$ and $C \rightarrow_{\Sigma_{i-1}} D$ such that $X_i = A \cup C$ and $Y_i = B \cup D$, moreover $y \in (B \cup D) \backslash (A \cup C)$. We may assume that $y \in B$, the case $y \in D$ being dual. Then $y \in B \backslash A$ and, since $A \rightarrow B \in \Sigma_{i-1}$ and by induction hypothesis: There exists $A' \rightarrow_{\Sigma_d} B'$ such that $A' \subseteq A$ and $y \in B'$. Since $A' \subseteq A \subseteq A \cup C = X_i$, $A' \rightarrow_{\Sigma_d} B'$ satisfies (11).

$\star$ *Case P2:* There exist $A \rightarrow_{\Sigma_{i-1}} B$ and $B \rightarrow_{\Sigma_{i-1}} C$ such that $X_i = A$, $Y_i = C$ and $y \in C \backslash A$. Let us consider the two sub-cases $y \in B$ and $y \notin B$.

- $y \in B$ implies $y \in B \backslash A$, and since $A \rightarrow B \in \Sigma_{i-1}$: By induction hypothesis there exists $A' \rightarrow_{\Sigma_d} B'$ such that $A' \subseteq A$ and $y \in B'$. Since $A' \subseteq A = X_i$, $A' \rightarrow_{\Sigma_d} B'$ satisfies (11).

- $y \notin B$ implies $y \in C \backslash B$, and since $B \rightarrow C \in \Sigma_{i-1}$: By induction hypothesis there exists $B' \rightarrow_{\Sigma_d} C'$ such that $B' \subseteq B$ and $y \in C'$. If $B' \subseteq A = X_i$ then $B' \rightarrow_{\Sigma_d} C'$ satisfies (11). If $B' \not\subseteq A$, let us write

$$B' \backslash A = \{y_k\}_{1 \leq k \leq q}$$

Since $B' \subseteq B$: $y_k \in B \backslash A$, and since $A \rightarrow B \in \Sigma_{i-1}$: By induction hypothesis there exist q implications $A_k \rightarrow_{\Sigma_d} B_k$ such that $A_k \subseteq A$ and $y_k \in B_k$. Therefore:

$$B' \backslash A \subseteq \bigcup_{1 \leq k \leq q} B_k \text{ and } B' \subseteq A \cup \bigcup_{1 \leq k \leq q} B_k$$

Axiom P_4 is now used to build an implication whose premise is included into A and whose conclusion is C', so it verifies (11) since $X_i = A$ and $y \in C'$. This implication is the last element of a sequence of q implications $A_k' \rightarrow_{\Sigma_k} C'$ obtained by applying iteratively P_4 to implications $A_k \rightarrow_{\Sigma_d} B_k$.

  - initialization: we define $A_1' \rightarrow_{\Sigma_d} C'$ as the result of P_4 applied to $A_1 \rightarrow_{\Sigma_d} B_1$ and $B' \rightarrow_{\Sigma_d} C'$ (note that $y_1 \subseteq B_1 \cap B'$ so $B_1 \cap B' \neq \emptyset$ and P_4 can be applied), so $A_1' = A_1 \cup B' \setminus B_1$.
  - induction: for $1 < k \leq q$, we define $A_k' \rightarrow_{\Sigma_d} C'$ as:

- ∗ the result of P_4 applied to $A_k \rightarrow_{\Sigma_d} B_k$ and $A'_{k-1} \rightarrow_{\Sigma_d} C'$ if $B_k \cap A'_{k-1} \neq \emptyset$, so $A'_k = A_k \cup A'_{k-1} \setminus B_k$
- ∗ $A'_{k-1} \rightarrow_{\Sigma_d} C'$ otherwise, so $A'_k = A'_{k-1}$.

To prove that $A'_q \subseteq A$, let us prove by induction on $k \in [1,q]$, that:

$$A'_k \subseteq A \cup \bigcup_{k<j\leq q} B_j$$

- initialization: For $k=1$, we have $A'_1 = A_1 \cup (B' \backslash B_1)$ so $A'_1 \subseteq A \cup \bigcup_{1<j\leq q} B_j$ directly follows from $B' \subseteq A \cup \bigcup_{1\leq k\leq q} B_k$ and $A_1 \subseteq A$.
- induction step: For $k>1$, the induction hypothesis is

$$A'_{k-1} \subseteq A \cup \bigcup_{k-1<j\leq q} B_j$$

moreover the computation of A'_k depends on the emptiness of $B_k \cap A'_{k-1}$.

  - ∗ If $B_k \cap A'_{k-1} = \emptyset$, then $A'_{k-1} \subseteq (A \cup \bigcup_{k-1<j\leq q} B_j) \setminus B_k$, moreover $A'_k = A'_{k-1}$. So we directly obtain $A'_k \subseteq A \cup \bigcup_{k<j\leq q} B_j$.
  - ∗ If $A'_{k-1} \cap B_k \neq \emptyset$, then $A'_{k-1} \setminus B_k \subseteq A \cup \bigcup_{k<j\leq q} B_j$. Moreover $A'_k = A_k \cup (A'_{k-1} \backslash B_k)$ and since $A_k \subseteq A$, we also obtain $A'_k \subseteq A \cup \bigcup_{k<j\leq q} B_j$.

We finally obtain

$$A'_q \subseteq A \cup \bigcup_{q<j\leq q} B_j \subseteq A$$

and $A'_q \rightarrow_{\Sigma_d} C'$ satisfies (11) (since $A = X_i$ and $y \in C'$). Thus the property is proved.

□

We can now prove Theorem 1.

Proof of Theorem 1: Since Σ and Σ_f are stated equivalent by Prop. 2, proving $\varphi_\Sigma(X) = X^{\Sigma_d}$ for $X \subseteq S$ is equivalent to prove $\varphi_{\Sigma_f}(X) = X^{\Sigma_d}$, where from Prop. 1 and Def. 1:

$$\varphi_{\Sigma_f}(X) = X^{\Sigma_f} = \bigcup \{B \subseteq S \mid X \rightarrow_{\Sigma_f} B\} \tag{12}$$

$$X^{\Sigma_d} = X \cup \bigcup \{B \subseteq S \mid A \rightarrow_{\Sigma_d} B \text{ and } A \subseteq X\} \tag{13}$$

$\supseteq$. Using Eq. (7) $X^{\Sigma_f} = X \cup \{B \subseteq S \mid A \rightarrow_{\Sigma_f} B \text{ and } A \subseteq X\}$. Then $X^{\Sigma_d} \subseteq X^{\Sigma_f}$ directly follows from $\Sigma_d \subseteq \Sigma_f$ stated by Lemma 1.

$\subseteq$. Consider any $b \in X^{\Sigma_f}$. If $b \in X$ then $b \in X^{\Sigma_d}$ by (13). Assume $b \notin X$. Since $b \in X^{\Sigma_f}$, there exists by (12) $X \rightarrow_{\Sigma_f} B$ such that $b \in B$. $b \notin X$ implies $b \in B \backslash X$ and by Lemma 2 there exists $A' \rightarrow_{\Sigma_d} B'$ such that $A' \subseteq X$ and $b \in B'$. So $b \in X^{\Sigma_d}$ and $X^{\Sigma_f} \subseteq X^{\Sigma_d}$. □

3.3 Σ_o : a Direct-Optimal IS Generated from a Direct IS Σ

Let us consider a direct IS Σ. If Σ is not direct-optimal then there exists an equivalent direct IS of smaller size. Like in Sect. 3.2, it means that some premise or conclusion parts of Σ are redundant with some properties particular to closure operators. This redundancy can be suppressed without altering the directness property. Let us consider the computation of $\varphi(X)$ for $X \subseteq S$ and an implication $A \rightarrow_\Sigma B$.

1. Assume $A \subseteq X$ and $A \cap B \neq \emptyset$. $A \rightarrow_\Sigma B$ causes the explicit enrichment of $\varphi(X)$ with $B = (A \cap B) \cup (B \setminus A)$. The $A \cap B$ part is redundant with the isotony and extensiveness of φ from which we have $A \subseteq \varphi(A) \subseteq \varphi(X)$ (moreover $A \cap B \subseteq A$). So only the part $B \setminus A$ of the $A \rightarrow_\Sigma B$ conclusion is useful.

2. Assume $C \rightarrow D \in \Sigma$ with $C \subset A$, $B \cap D \neq \emptyset$ and $A \subseteq X$. Since $C \subset X$, $C \rightarrow_\Sigma D$ causes the explicit enrichment of $\varphi(X)$ with $D = (B \cap D) \cup (D \setminus B)$. The part $B \cap D$ is similarly redundant with $A \rightarrow_\Sigma B$, which already states the enrichment of $\varphi(X)$ with $B = (B \cap D) \cup (B \setminus D)$.

3. Assume $A \rightarrow B' \in \Sigma$, with $B \neq B'$. Then the cardinality $|A|$ is added twice to the size of Σ, while it is only added once if the pair $\{A \rightarrow_\Sigma B, A \rightarrow_\Sigma B'\}$ — in a way redundant — is replaced by the equivalent implication $A \rightarrow B \cup B'$.

4. Assume $A \subseteq X$ and $B = \emptyset$. $A \rightarrow_\Sigma B$ is clearly useless to compute $\varphi(X)$.

Theorem 2 generalizes these remarks: it states that the absence of such redundancies is a necessary and sufficient condition for a direct IS to be direct-optimal.

Theorem 2 *A direct IS Σ is direct-optimal iff:*

P5 (extensiveness axiom): *for all $A \rightarrow_\Sigma B$, $A \cap B = \emptyset$*

P6 (isotony axiom): *for all $A \rightarrow_\Sigma B$ and $C \rightarrow_\Sigma D$, $C \subset A$ implies $B \cap D = \emptyset$*

P7 (premise axiom): *for all $A \rightarrow_\Sigma B$ and $A \rightarrow_\Sigma B'$, $B = B'$*

P8 (not empty conclusion axiom): *for all $A \rightarrow_\Sigma B$, $B \neq \emptyset$.*

Two lemmas are needed to prove this theorem. Lemma 3 states that the deletion of the previously mentioned redundancies preserves the directness property of the considered IS. In Lemma 4 we consider the particular direct ISs whose conclusion parts are singletons. Such an IS Σ does not necessarily verifies P7, but Lemma 4 states that if Σ verify P5 and P6 then Σ is smaller[‖] than any other equivalent such IS(whose conclusions are also singletons).

Lemma 3 *Let Σ be a direct IS.*

1. *If $A \rightarrow B \in \Sigma$ with $A \cap B \neq \emptyset$ then $\Sigma \setminus \{A \rightarrow_\Sigma B\} \cup \{A \rightarrow B \setminus A\}$ is also a direct IS equivalent to Σ of smaller size.*

2. *If $A \rightarrow B \in \Sigma$ and $C \rightarrow D \in \Sigma$ with $C \subset A$ and $B \cap D \neq \emptyset$ then $\Sigma \setminus \{A \rightarrow_\Sigma B\} \cup \{A \rightarrow B \setminus D\}$ is also a direct IS equivalent to Σ of smaller size.*

[‖] In the sense of inclusion.

3. *If $A \to B \in \Sigma$ and $A \to B' \in \Sigma$ with $B \neq B'$ then $\Sigma \setminus \{A \to_\Sigma B, A \to_\Sigma B'\} \cup \{A \to B \cup B'\}$ is also a direct IS equivalent to Σ of smaller size.*

4. *If $A \to B \in \Sigma$ with $B = \emptyset$ then $\Sigma \setminus \{A \to_\Sigma B\}$ is also a direct IS equivalent to Σ of smaller size.*

Proof:

1. Let $A \to_\Sigma B$ be such that $A \cap B \neq \emptyset$. Let us denote by Σ' the IS $\Sigma \setminus \{A \to_\Sigma B\} \cup \{A \to B \setminus A\}$. Let us consider $X \subseteq S$ and prove that Σ' is a direct IS equivalent to Σ by stating $X^{\Sigma'} = X^\Sigma$. When $A \not\subseteq X$, the implications involved in the computation of X^Σ and $X^{\Sigma'}$ are the same, thus $X^{\Sigma'} = X^\Sigma$. When $A \subseteq X$, $X^{\Sigma'}$ is obtained as follows:

$$\begin{aligned}
X^{\Sigma'} &= X \cup \{B' \mid A' \subseteq X, A' \to_{\Sigma'} B'\} \\
&= X \cup \{B' \mid A' \subseteq X, A' \to_{\Sigma'} B' \neq A \to_{\Sigma'} B \backslash A\} \cup B \setminus A \\
&= X \cup \{B' \mid A' \subseteq X, A' \to_{\Sigma'} B' \neq A \to_{\Sigma'} B \backslash A\} \cup B \\
&\qquad \text{since } A \subseteq X \text{ so } X \cup (B \setminus A) = X \cup B \\
&= X \cup \{B' \mid A' \subseteq X, A' \to_{\Sigma} B' \neq A \to_{\Sigma} B\} \cup B \\
&\qquad\qquad \text{by definition of } \Sigma' \\
&= X \cup \{B' \mid A' \subseteq X, A' \to_{\Sigma} B'\} \\
&= X^\Sigma
\end{aligned}$$

2. The proof is the same for $A \to_\Sigma B$ and $C \to_\Sigma D$ such that $C \subset A$ and $B \cap D \neq \emptyset$. Let us denote by Σ' the IS $\Sigma \setminus \{A \to_\Sigma B\} \cup \{A \to B \setminus D\}$. Stating $X^{\Sigma'} = X^\Sigma$ allows to conclude that Σ' is a direct IS equivalent to Σ. In this case, $C \to_\Sigma D \in \Sigma$ implies $D \in X^{\Sigma'}$ when $C \subset A \subseteq X$.

3. The proof is the same for $A \to_\Sigma B$ and $A \to_\Sigma B'$ such that $B \neq B'$.

4. Immediate since the implication $A \to_\Sigma \emptyset$ adds no element to closures.

$\square$

Lemma 4 *Let Σ and Σ' be two equivalent and direct ISs whose conclusions are singletons. If Σ verifies P5 and P6 then $\Sigma \subseteq \Sigma'$.*

Proof: Let $A \to B$ be a Σ-implication. By hypothesis, the conclusion B contains only one element, say b. Since Σ' only owns implications whose conclusion is a singleton, let us prove that $\Sigma \subseteq \Sigma'$ by stating that $A \to b$ is also a Σ'-implication.
Let us consider $\varphi_{\Sigma'}(A)$, the closure of A in Σ', as the union of three subsets:

$$\begin{aligned}
\varphi_{\Sigma'}(A) &= A \cup \{D \mid C \subseteq A, C \to_{\Sigma'} D\} \\
&= A \cup \{B' \mid A \to_{\Sigma'} B'\} \cup \{D \mid C \subset A, C \to_{\Sigma'} D\}
\end{aligned}$$

Similarly $\varphi_\Sigma(A) = A \cup \{B' \mid A \to_\Sigma B'\} \cup \{D \mid C \subset A, C \to_\Sigma D\}$. Since Σ and Σ' are equivalent, $\varphi_\Sigma(X) = \varphi_{\Sigma'}(X)$ for any $X \subseteq S$. In particular since $A \to B \in \Sigma$ and $B = \{b\}$, $b \in \varphi_\Sigma(A)$ and $b \in \varphi_{\Sigma'}(A)$.
Since Σ verifies P5, we deduce from $A \to b \in \Sigma$ that $A \cap \{b\} = \emptyset$ and $b \notin A$. Since Σ verifies P6, $\{b\} \cap D = \emptyset$ for any implication $C \to_\Sigma D$ such that $C \subset A$. So $b \notin \{D \mid C \subset A, C \to_\Sigma D\}$. Since $b \notin A$, we also have $b \notin C$

and $b \notin \varphi_\Sigma(C) = C \cup \{D \mid C \rightarrow_\Sigma D\}$ for each $C \subset A$. Since $\varphi_\Sigma(C) = \varphi_{\Sigma'}(C)$, $b \notin \{D \mid C \subset A, C \rightarrow_{\Sigma'} D\}$. Therefore, the only subset containing b in $\varphi_{\Sigma'}(A)$ is $\{B' \mid A \rightarrow_{\Sigma'} B'\}$ and $A \rightarrow_{\Sigma'} b$ is a Σ'-implication. This achieves the proof. □

We can now prove Theorem 2.

Proof of Theorem 2:

$\Rightarrow$): By Lemma 3, we state that Σ is not direct-optimal when:

1. there exists $A \rightarrow_\Sigma B$ such that $A \cap B \neq \emptyset$ or
2. there exist $A \rightarrow_\Sigma B$ and $C \rightarrow_\Sigma D$ such that $C \subset A$ and $B \cap D \neq \emptyset$ or
3. there exist $A \rightarrow_\Sigma B$ and $A \rightarrow_\Sigma B'$ such that $B \neq B'$ or
4. there exists $A \rightarrow_\Sigma B$ such that $B = \emptyset$.

$\Leftarrow$): Let us introduce $s(\Sigma|A)$ as the size of an IS Σ reduced to its Σ-implications of premise $A \subseteq S$. Note that

$$s(\Sigma) = \sum_{A \subseteq S} s(\Sigma|A) \tag{14}$$

Let Σ be an IS verifying P5, P6, P7 and P8, and let Σ' be a direct IS equivalent to Σ. To prove that Σ is direct-optimal we have to show that $s(\Sigma) \leq s(\Sigma')$. To do so, we use (14) and prove the stronger property:

$$\forall A \subseteq S, s(\Sigma|A) \leq s(\Sigma'|A) \tag{15}$$

Let us consider a set $A \subseteq S$. If there is no Σ-implication of premise A, then we have $s(\Sigma|A) = 0 \leq s(\Sigma'|A)$. If there is a Σ-implication $A \rightarrow B$, where $B = \{b_1, b_2, \ldots, b_n\}$, then it is the only Σ-implication of premise A by P7, and $n > 0$ by P8. Let $A \rightarrow_{\Sigma'} B_1$, …, $A \rightarrow_{\Sigma'} B_m$ be the m Σ'-implications whose premise are A, with** $m \geq 0$, and let p be the total cardinality of their conclusions:

$$\begin{cases} p = 0 & \text{if } m = 0 \\ p = \sum_{1 \leq i \leq m} |B_i| & \text{if } m > 0 \end{cases}$$

Then:

$$\begin{aligned} s(\Sigma|A) &= |A| + n \\ s(\Sigma'|A) &= m|A| + p \end{aligned}$$

In order to compare $s(\Sigma|A)$ and $s(\Sigma'|A)$, let us define from Σ another IS Σ_* whose conclusions are singletons by:

$$\Sigma_* = \bigcup \{C \rightarrow d_1, \ldots, C \rightarrow d_p \mid C \rightarrow \{d_1, \ldots, d_p\} \in \Sigma, C \subseteq S\} \tag{16}$$

Σ_* is direct and equivalent to Σ by Lemma 3(3). It also verifies P5 and P6. Let Σ'_* be defined from Σ' in the same way. Σ_* contains $n > 0$ implications of premise A: $A \rightarrow_{\Sigma_*} b_1, \ldots, A \rightarrow_{\Sigma_*} b_n$; And Σ'_* contains $p \geq 0$ implications of premise A (whose conclusions are also singletons). So we have:

$$\begin{aligned} s(\Sigma_*|A) &= n(|A| + 1) \\ s(\Sigma'_*|A) &= p(|A| + 1) \end{aligned}$$

** Note that $m = 0$ when there is no Σ'-implication of premise A.

Since Σ_* verifies P5 and P6 and the conclusions of Σ_* and Σ'_* are of cardinality 1, Lemma 4 states that $\Sigma_* \subseteq \Sigma'_*$. Then

$$\begin{array}{rcl} s(\Sigma'_*|A) & \geq & s(\Sigma_*|A) \\ p(|A|+1) & \geq & n(|A|+1) \end{array}$$

Therefore $p \geq n > 0$. Remark that $p > 0$ implies $p = \sum_{1\leq i\leq m}|B_i|$ and $m > 0$. We finally obtain $s(\Sigma'|A) \geq s(\Sigma|A)$ by:

$$\begin{array}{rcl} s(\Sigma'|A) = m|A| + p & \geq & m|A| + n \\ & \geq & |A| + n = s(\Sigma|A) \end{array}$$

□

We can now derive from Th. 2 the *direct-optimal IS Σ_o generated from a direct IS Σ*:

Definition 5 *The* direct-optimal IS Σ_o *generated from a* direct *IS Σ is a direct IS s.t.:*

P8 (optimization axiom) *for all $A, B \subseteq S$, $A \to B \in \Sigma_o$ iff $B \neq \emptyset$ and*

$$B = \bigcup\{B' \subseteq S \,|A \to_\Sigma B'\} \backslash \bigcup\{D \subseteq S\,|C \to_\Sigma D \text{ and } C \subset A\} \backslash A \tag{17}$$

3.4 Σ_{do}: a Direct-Optimal IS Generated from an IS Σ

Let us consider an IS Σ. The combination of Def. 4 and Def. 5 describes Σ_{do}, the direct-optimal IS generated from Σ:

Definition 6 *The direct-optimal IS Σ_{do} generated from some IS Σ is defined as the direct-optimal ISobtained by Def. 5 from the direct IS Σ_d which itself is obtained by Def. 4 from Σ.*

Σ_{do} is then an IS of minimal size, equivalent to Σ and such that $\varphi_\Sigma(X)$ can be obtained by scanning only once its implications (see Ex. 1). Moreover equivalent ISs define an unique direct-optimal IS, as stated by the following corollary.

Corollary 1 *Let Σ and Σ' be equivalent ISs. Then $\Sigma_{do} = \Sigma'_{do}$.*

Proof: Let us define Σ_* from Σ_{do} and Σ'_* from Σ'_{do} as indicated by Eq. (16). Remark that Σ_{do} (resp. $\Sigma_{do}{}'$) can dually be defined from Σ_* (resp. Σ_*) since it satisfies axiom P7:

$$\Sigma_{do} = \{C \to \{d_1,\ldots,d_n\} \mid C \to_{\Sigma_*} d_1,\ldots,C \to_{\Sigma_*} d_n, C \subseteq S\} \tag{18}$$

Σ_* (resp. Σ'_*) is direct and equivalent to Σ_{do} (resp. $\Sigma_{do}{}'$) by Lemma 3(3). By construction Σ_* and Σ'_* satisfy $P5$ and $P6$. So by Lemma 4 $\Sigma_* = \Sigma'_*$. We conclude using Eq. (18):

$$\begin{array}{rl} \Sigma_{do} & = \{C \to \{d_1,\ldots,d_n\} \mid C \to_{\Sigma_*} d_1,\ldots,C \to_{\Sigma_*} d_n, C \subseteq S\} \\ & = \{C \to \{d_1,\ldots,d_n\} \mid C \to_{\Sigma'_*} d_1,\ldots,C \to_{\Sigma'_*} d_n, C \subseteq S\} \\ & = \Sigma'_{do} \end{array}$$

□

Example 1 *Consider the following IS Σ on $\{a,b,c,d,e\}$:*

$$\Sigma = \begin{cases} 1: a \to b \\ 2: ac \to d \\ 3: e \to a \end{cases}$$

The full IS Σ_f is not given since it contains more than $3^5 = 243$ implications[††]*. Σ_d and Σ_{do} are given below. Note that Σ_d is not direct-optimal because (3) and (4) do not verify P7.*

$$\Sigma_d = \begin{cases} 1: a \to b \\ 2: ac \to d \\ 3: e \to a \\ 4: \mathbf{e} \to \mathbf{b} \textit{ (P4 on 3 and 1)} \\ 5: \mathbf{ce} \to \mathbf{d} \textit{ (P4 on 3 and 2)} \end{cases} \qquad \Sigma_{do} = \begin{cases} 6: a \to b \\ 7: ac \to d \\ 8: \mathbf{e} \to \mathbf{ab} \\ 9: ce \to d \end{cases}$$

For example, $\varphi(ce) = ce \cup ab \cup d = abcde$ is directly deduced from implications 8 and 9 by Th. 1. Similarly, $\varphi(ae) = ae \cup b \cup ab = abe$ is deduced from implications 6 and 8. It is also easy to check on $\mathbb{F}_\Sigma$ (given on Fig. 1 by its Hasse diagram where the cover relation of the order relation is oriented from bottom to top.) that abcde (resp. abe) is the least set of $\mathbb{F}_\Sigma$ that contains ce (resp. ae).

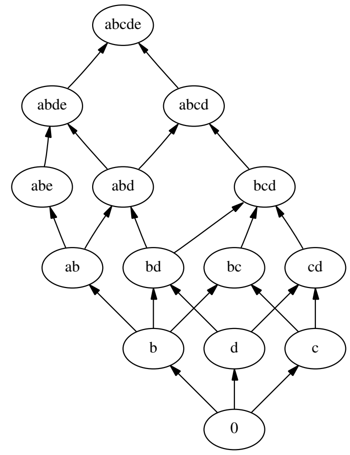

Fig. 1: $\mathbb{F}_\Sigma$ for Σ given in Ex. 1

4 Algorithms

We give in this section algorithms that rely on the results obtained in Sect. 3. We first present in Sect. 4.2 an algorithm which takes as input a *direct-optimal* IS Σ and a subset $X \subseteq S$, and computes the closure $\varphi_\Sigma(X)$. We also give an algorithm which computes from any IS Σ the associated direct-optimal Σ_{do}. In Sect. 4.3 we give an algorithm which takes as input some IS Σ and computes the associated Moore family $\mathbb{F}_\Sigma$, based not on the direct characterization of $\mathbb{F}_\Sigma$ but on properties of the lattice $(\mathbb{F}_\Sigma, \subseteq)$. All algorithms handle ISs and Moore families on S. Both are represented by a data-structure called *lexicographic tree*, presented in Sect. 4.1.

4.1 Lexicographic Tree

A nice and well-known data structure to represent a family $\mathcal{F}$ on S ordered by α, a total order on S, is its *lexicographic tree* of depth $|S|$. Using this tree basic operations on $\mathcal{F}$ (such as deletion, addition and search of a subset) can be efficiently performed in $O(|S|)$. Introduced for a distributive Moore family in [Gan84, MN96], it has been generalized in [NR99] to any family $\mathcal{F}$ by introducing marked nodes. Its principle is intuitively the following. Nodes represent subsets $X \subseteq S$: The tree contains a node for each subset $X \subseteq F$ with $F \in \mathcal{F}$. Conventionally the root represents the empty set. A node that represents an element of $\mathcal{F}$ is marked. Edges are labelled by elements of S so that labels of edges that leave a given

[††] Σ_f exactly contains 275 implications:

- $3^5 = 243$ implications such that the conclusion is a subset of the premise,
- and 32 implications such that the conclusion is not included in the premise.

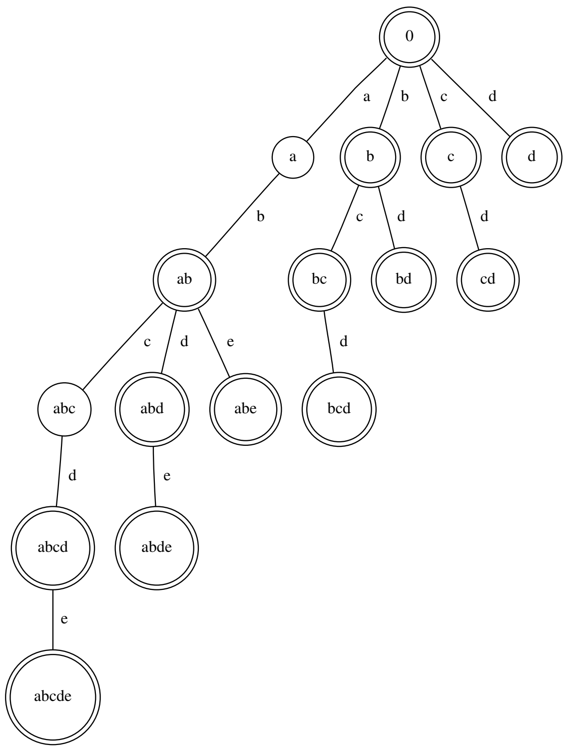

Fig. 2: The lexicographic tree associated to the Moore family $\mathbb{F}$ given on Fig. 1 for the order $\alpha = a < b < c < d < e$.

node are sorted according to α from left to right. Moreover consider a marked node n that represents an element $F \in \mathcal{F}$ sorted according to α. Then (see Prop. 3 below) F can be retrieved from the tree by collecting labels along the path from the root to n (labels along such a path are by construction sorted according to α).

Example 2 *Figure 2 shows the lexicographic tree $T_{\mathbb{F}}$ associated to the Moore family $\mathbb{F}$ given in Fig. 1 for the order $\alpha = a < b < c < d < e$, where each node n is labelled by the set X it represents (0 denotes $\emptyset$) and marked nodes are doubly circled.*

A lexicographic tree is formally defined as follows:

Definition 7 *Let $\mathcal{F}$ be a family on $S = \{s_1, \ldots, s_{|S|}\}$ whose elements are sorted according to $\alpha = s_1 < s_2 < \ldots < s_{|S|}$. The* lexicographic tree *$T_{\mathcal{F}}$ of $\mathcal{F}$ (or simply T) is a 3-uplet $(N, child, mark)$, where:*

- *N is the set of nodes of T, where a node $n_X \in N$ is associated to every subset X of some element $F \in \mathcal{F}$. By convention $n_\emptyset$ is the root of the tree.*

$$N = \{n_X \mid X \subseteq F \text{ and } F \in \mathcal{F}\}$$

- mark *is a boolean function used to distinguish nodes associated to elements of* $\mathcal{F}$. *For* $n_X \in T$*:*

$$mark(n_X) \quad = \quad true \text{ iff } X \in \mathcal{F}$$

- child *associates to each node* n_X *its children: for* $s_i \in S$, $child(n_X, s_i) \in N$ *is either the empty set or the target node of an edge labelled by* s_i *whose source node is* n_X. *If* $X = \{x_1, \ldots, x_m\}$ *is sorted according to* α *and* $m \leq i \leq |S|$*:*

$$\begin{aligned} child(n_X, s_i) &= n_{X \cup \{s_i\}} \text{ if } n_{X \cup \{s_i\}} \in N \\ &= \emptyset \text{ else} \end{aligned}$$

The depth of T *is* $|S|$.

Note that in this definition $n_X \in N$ is seen either as a node of T or as the subset $X \subseteq S$ it represents. As stated by Prop. 3, the subset X can easily be retrieved from the tree.

Proposition 3 *Let* $\mathcal{F}$ *be a family on* S *sorted according to* α *and* T *its lexicographic tree. Then the labels collected along the path from the root* $n_\emptyset$ *to a node* n_X *represent the subset* $X = \{x_1, \ldots, x_m\}$ *sorted according to* α*:*

$$n_\emptyset \xrightarrow{x_1} n_{\{x_1\}} \xrightarrow{x_2} n_{\{x_1,x_2\}} \cdots \xrightarrow{x_m} n_{\{x_1,\ldots,x_m\}} = n_X \tag{19}$$

Consider a family $\mathcal{F}$ on S sorted according to α. Basic operations such as the test if a given $F \subseteq S$ belong to $\mathcal{F}$, the addition or deletion of an element in $\mathcal{F}$ can be done on $T_{\mathcal{F}}$ in $O(|S|)$ (its depth) by a run from the root to a particular node (addition consists in adding or marking a node, deletion consists in deleting or unmarking a node). This complexity is due to the linear order on elements in S, and is lower than the complexity in $O(|\mathcal{F}| \cdot |S|)$ obtained when $\mathcal{F}$ is represented by a list of subsets. The computation of the element $F \in \mathcal{F}$ associated to a marked node $n \in N$ is also done in $O(|S|)$ using Eq.(19). Finally set operations on families such as union, intersection, difference and inclusion test are done in $O(|S|)$, still footnote to the linear order on S.

We extend this lexicographic tree to a *two-level lexicographic tree* to represent a binary relation on 2^S and thus an IS on S.

Definition 8 *Let* Σ *be an IS on* S. *The* two-level lexicographic tree T_Σ *of* Σ *is s.t.*

- *The* initial lexicographic tree *is representing the family* $\{A \subseteq S \mid A \rightarrow_\Sigma B\}$. *Its root is* $n_\emptyset$.
- *Each marked node* n_A *of the initial tree is the root of a* lexicographic subtree *representing the family* $\{B \subseteq S \mid A \rightarrow_\Sigma B\}$.

By construction the depth of a two-level lexicographic tree on S is $2|S|$ and complexities in $O(|S|)$ given for lexicographic trees are still valid. When the considered IS is direct-optimal, each lexicographic subtree encodes only one subset $B \subseteq S$ since a marked node A of the initial subtree is the premise of only one implication, as stated by Th. 2.

Example 3 *The two-level lexicographic tree* $T_{\Sigma_{do}}$ *associated to the IS* Σ_{do} *given in Ex. 1 is shown on Fig. 3, where a double circle indicates a marked node of the initial lexicographic tree, the lexicographic subtrees appear in vertical boxes, and horizontal boxes indicate their marked nodes, labelled by the corresponding implication.*

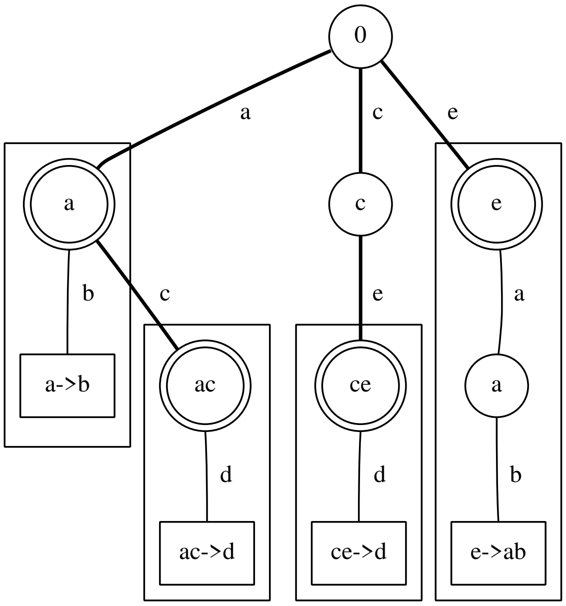

Fig. 3: The two-level lexicographic tree for Σ_{do} of Ex. 1, with the lexicographic order $\alpha = a < b < c < d < e$.

4.2 Computation of $\varphi_\Sigma(X)$, $X \subseteq S$

The functions presented here aim at computing closures $\varphi_\Sigma(X)$ for some IS Σ. The function `closure` in Algorithm 1 is directly derived from the characterization of the closure operator φ_Σ associated to a direct IS (Def. 1). It computes $\varphi_\Sigma(X) = \varphi_{\Sigma_{do}}(X)$ with Σ_{do} as input. The function `complete` in Algorithm 2 first computes Σ_d from Σ using Def. 4, then optimizes Σ_d to obtain Σ_{do} using Def. 5.

Name: `closure`
Input: $X \subseteq S$, sorted according to α
    A direct-optimal IS Σ_{do} on S
Output: $\varphi_\Sigma(X)$
begin
  $\varphi_\Sigma(X) = X$
  foreach $A \rightarrow_{\Sigma_{do}} B$ *such that* $A \subseteq X$ **do**
    add B to $\varphi_\Sigma(X)$
  return $\varphi_\Sigma(X)$
end

Algorithm 1: Computation of $\varphi_\Sigma(X)$

Complexity 1

1. *Function* `closure` *in Algorithm 1 computes* $\varphi_\Sigma(X)$ *from* Σ_{do} *with the following complexities:*

```
Name: complete
Input: An implicational system Σ on S
Output: The direct-optimal IS Σ_do on S
begin
    (Generation of Σ_d by completion of Σ)
    Σ_d = Σ
    foreach A →_{Σ_d} B do
        foreach C →_{Σ_d} D do
            if B ∩ C ≠ ∅ then add A ∪ (C\B) → D to Σ_d
    (Generation of Σ_do by optimization of Σ_d)
    Σ_do = ∅
    foreach A →_{Σ_d} B do
        B' = B
        foreach C →_{Σ_d} D do
            if C = A then B' = B' ∪ D
            if C ⊂ A then B' = B'\D
        B' = B'\A
        add A → B' to Σ_do
    return Σ_do
end
```

Algorithm 2: Computation of Σ_{do} from Σ

- *in function of X and* Σ_{do}*: in* $O(|\Sigma_{do}| \cdot |\varphi_\Sigma(X)|)$*;*
- *in function of* Σ_{do} *only:* $O(s(\Sigma_{do}))$

2. *Function* `complete` *in Algorithm 2 computes* Σ_{do} *from* Σ *in* $O(|\Sigma_d|^2 \cdot |S|)$

Proof:
1. A test for inclusion $Y \subseteq Y'$ can be done in $min(|Y|, |Y'|)$. For the following complexities we shall use either $|Y|$ or $|Y'|$. Similarly for adding Y to Y'. Hence:

- In function of X and Σ_{do}: For each of the $|\Sigma_{do}|$ implications $A \rightarrow_{\Sigma_{do}} B$, the test $A \subseteq X$ is done in $O(|X|) < O(|\varphi_\Sigma(X)|)$, and adding B to $\varphi_\Sigma(X)$ is done in $O(|\varphi_\Sigma(X)|)$. Hence a complexity in $O(|\Sigma_{do}| \cdot |\varphi_\Sigma(X)|)$.

- In function of Σ_{do} only: For each of the $|\Sigma_{do}|$ implications $A \rightarrow_{\Sigma_{do}} B$, the test $A \subseteq X$ is done in $O(|A|)$, and adding B to $\varphi_\Sigma(X)$ is done in $O(|B|)$. Hence a complexity in $\Sigma_{A\rightarrow B}|A| + |B| = s(\Sigma_{do})$.

2. Each of the $|\Sigma_d|^2$ steps of the first nested `for` loop first performs set operations on subsets A, B, C and D that are done in $O(|S|)$. The second `for` loop (of $|\Sigma_d|^2$ steps) also performs set operations and additions and deletions in T_{Σ_d} in $|O(S)|$. Hence a complexity in $O(|\Sigma_d|^2 \cdot |S|)$. □

Since Σ_{do} is direct, the computation of the closure $\varphi_\Sigma(X)$ (function `closure`) is performed in $O(s(\Sigma_{do}))$ with only one enumeration of Σ_{do}-implications. However, a preprocessing (function `complete`) is necessary to compute Σ_{do} from Σ in $O(|\Sigma_d|^2 \cdot |S|)$. When the closure is directly computed from Σ (that can be greater or smaller than Σ_{do}), it is obtained in $O(|\Sigma|\cdot|S|^2)$ [Mai83, Wil95] by several iterations over Σ-implications. So in the cases where few closures are needed, or where a small non-direct IS is considered, it may be more efficient to iterate over Σ-enumerations instead of computing Σ_{do}.

4.3 Generation of $\mathbb{F}_\Sigma$

The definition of $\mathbb{F}_\Sigma$ (or simply $\mathbb{F}$) as the family associated to φ_Σ (Eq. (1)) or as the family generated by Σ (Eq. (3)) cannot be directly used to generate $\mathbb{F}$: it would make the computation exponential since all subsets of S have to be enumerated. We propose another characterization of $\mathbb{F}$ in function of φ_Σ that exploits the fact that $(\mathbb{F},\subseteq)$ is a lattice: it uses lattice properties, in particular properties of its irreducible elements. We first recall some basic definitions.

Consider a lattice L. An element j (resp. m) of L is a *join-irreducible* (resp. *meet-irreducible*) of L if it cannot be obtained as the join (resp. meet) of elements of L all distinct from j (resp. from m). The set of join-irreducible (resp. meet-irreducible) of L is denoted by J_L (resp. M_L). A finite lattice L has a minimal (resp. maximal) element denoted by $\perp$ (resp. $\top$). Conventionally $\perp=\bigvee\emptyset$ and $\top=\bigwedge\emptyset$, therefore $\perp\notin J_L$ and $\top\notin M_L$. A join-irreducible (resp. meet-irreducible) element $j\in J_L$ (resp. $m\in M_L$) covers (resp. is covered by) an unique element in L, which is then denoted by j^- (resp. m^+). If an element $x\in L$ is not a join-irreducible element, then there exists a subset $X\subseteq L$ such that $x=\bigvee X$ and $x\notin X$: Either $x=\perp$ (Remark that when $L=(\mathbb{F}_\Sigma,\subseteq)$, one can have $\perp=\bigvee\emptyset\neq\emptyset$ when $\emptyset\rightarrow_\Sigma A\in\Sigma$ and $A\neq\emptyset$.) and $X=\emptyset$, or it is easy to check that there exists $y,y'\in X$ such that $x=y\vee y'$. For definitions and notations not recalled here, see [Bir67, BM70].

As said before $(\mathbb{F},\subseteq)$ is a lattice, with, for $X\subseteq S$:

$$\bigvee X=\cap\{F\in\mathbb{F}\mid F\subseteq X\}=\varphi_\Sigma(X) \tag{20}$$

$\mathbb{F}$ is characterized in Th. 3 by considering two cases: Either $F\in\mathbb{F}$ is a join-irreducible element of $\mathbb{F}$ or not. It is based on the characterization of join-irreducible elements of $\mathbb{F}$ by Lemma 5 (for $x\in S$ we abuse notations and write $\varphi(x)$ for $\varphi(\{x\})$).

Lemma 5 $J_{\mathbb{F}}\subseteq\{\varphi_\Sigma(x)\mid x\in S\}$

For completeness, we give a proof of this simple result.
Proof: Let us consider the lattice $(\mathbb{F},\subseteq)$. Let $F\in J_{\mathbb{F}}$ be an irreducible element that covers F^-, and $x\in F\setminus F^-$. Let us prove by contradiction that $F=\varphi(x)$, i.e. that F is the least element of $\mathbb{F}$ that contains x. Assume $x\in F'$ for some $F'\in\mathbb{F}$ s.t. $F'\subseteq F$. Then $F'\subseteq F^-$ so $x\in F^-$, which leads to a contradiction. □

Theorem 3 *Let Σ be an implicational system. Then:*

$$\mathbb{F}_\Sigma=\{\varphi_\Sigma(\emptyset)\}\,\cup\,\{\varphi_\Sigma(x)\mid x\in S\}\,\cup\,\{\varphi_\Sigma(F_1\cup F_2)\mid F_1,F_2\in\mathbb{F}_\Sigma\}$$

For completeness, we give a proof of this simple result, folklore in lattice theory.
Proof: Let $\mathcal{F}=\{\varphi_\Sigma(\emptyset)\}\,\cup\,\{\varphi_\Sigma(x)\mid x\in S\}\,\cup\,\{\varphi_\Sigma(F_1\cup F_2)\mid F_1,F_2\in\mathcal{F}\}$. Let us prove that $\mathbb{F}_\Sigma=\mathcal{F}$.
$\supseteq$. Each subset of $\mathcal{F}$ is a closure $\varphi_\Sigma(X)$ for $X\subseteq S$, so belongs to $\mathbb{F}_\Sigma$ (Eq. (1)).
$\subseteq$. Let $F\in\mathbb{F}$. If $F\in J_{\mathbb{F}}$ then $F\in\mathcal{F}$ follows from Lemma 5. Assume $F\notin J_{\mathbb{F}}$. If $F=\perp_{\mathbb{F}}$ then $F=\varphi_\Sigma(\emptyset)$ by Eq. (2) and $F\in\mathcal{F}$. If $F\neq\perp_{\mathbb{F}}$ then it is the join of two subsets $F_1,F_2\in\mathbb{F}$, i.e. $F=F_1\vee F_2=\varphi_\Sigma(F_1\cup F_2)$, thus $F\in\mathcal{F}$. □

A generation of $\mathbb{F}_\Sigma$ on $S = \{x_1, \ldots, x_n\}$ can be derived from Th. 3. Let us define $\mathcal{F}_i$ as the family computed from all $\varphi(x_j)$ with $j \leq i$:

$$\mathcal{F}_i = \varphi_\Sigma(\emptyset) \cup \{\varphi_\Sigma(x_j) \mid x_j \in S \text{ and } j \leq i\} \cup \{\varphi_\Sigma(F_1 \cup F_2) \mid F_1, F_2 \in \mathcal{F}_i\}$$

where $\mathcal{F}_0 = \varphi_\Sigma(\emptyset)$. Clearly $\mathcal{F}_n = \mathbb{F}_\Sigma$ and $\mathcal{F}_i$ can be generated from $\mathcal{F}_{i-1}$ by:

$$\mathcal{F}_i = \mathcal{F}_{i-1} \cup \{\varphi_\Sigma(x_i)\} \cup \{\varphi_\Sigma(F_1 \cup F_2) \mid F_1, F_2 \in \mathcal{F}_i\}, i \geq 1 \tag{21}$$

Using Lemma 6 that defines more precisely the elements in $\mathcal{F}_i \backslash \mathcal{F}_{i-1}$, $\mathcal{F}_i$ can be generated from $\mathcal{F}_{i-1}$ by:

$$\mathcal{F}_i = \mathcal{F}_{i-1} \cup \{\varphi_\Sigma(x_i)\} \cup \{\varphi_\Sigma(F \cup \varphi_\Sigma(x_i)) \mid F \in \mathcal{F}_{i-1}\} \tag{22}$$

Lemma 6 *Let $i \leq n$ and $F \in \mathcal{F}_i \backslash \mathcal{F}_{i-1}$.*
Then there exists $F' \subseteq S$ s.t. $F = \varphi_\Sigma(F' \cup \varphi_\Sigma(x_i))$ with $F' \in \mathcal{F}_{i-1}$ or $F' = \emptyset$.

Proof: We consider two cases:
• Either $\varphi_\Sigma(x_i) \in \mathcal{F}_{i-1}$: In this case it appears from (21) that $\mathcal{F}_i = \mathcal{F}_{i-1}$;
• Or $\varphi_\Sigma(x_i) \notin \mathcal{F}_{i-1}$: In this case let $F^0, F^1, \ldots F^p$ be the closures successively added to $\mathcal{F}_{i-1}$ to obtain $\mathcal{F}_i$ where:

- $F^0 = \varphi_\Sigma(x_i)$
- $\{F^1, \ldots, F^p\} = \{\varphi_\Sigma(F_1 \cup F_2) \mid F_1, F_2 \in \mathcal{F}_i\} \setminus \mathcal{F}_{i-1}$

We prove by induction on j with $0 \leq j \leq p$ that there exists $F' \subseteq S$ s.t. $F^j = \varphi_\Sigma(F' \cup \varphi_\Sigma(x_i))$ with $F' \in \mathcal{F}_{i-1}$ or $F' = \emptyset$.
Base case: For $j = 0$: $F^0 = \varphi_\Sigma(x_i) = \varphi_\Sigma(\emptyset \cup \varphi_\Sigma(x_i))$ (case $F' = \emptyset$).
Inductive step: Let $0 < j \leq p$. Assume the property is proved for $0 \leq k < j$:

$$\text{there exists } F' \subseteq S \text{ s.t. } F^k = \varphi_\Sigma(F' \cup \varphi_\Sigma(x_i)) \text{ with } F' \in \mathcal{F}_{i-1} \text{ or } F' = \emptyset$$

and consider the set F^j. By (21), F^j is the upper bound of two closures F_1 and F_2 in $\mathcal{F}_i$: $F^j = \varphi_\Sigma(F_1 \cup F_2)$. Either $F_1, F_2 \in \mathcal{F}_{i-1}$ or F_1, F_2 in $\mathcal{F}_i \backslash \mathcal{F}_{i-1}$ or $F_1 \in \mathcal{F}_{i-1}$ and $F_2 \in \mathcal{F}_i \backslash \mathcal{F}_{i-1}$, or the converse. The first case implies $F^j \in \mathcal{F}_{i-1}$ by (21), hence a contradiction with $F^j \in \mathcal{F}_i \backslash \mathcal{F}_{i-1}$. We consider only the case $F_1, F_2 \in \mathcal{F}_i \backslash \mathcal{F}_{i-1}$, the other cases being similar.
By the induction hypothesis, there exists $F'_1 \in \mathcal{F}_{i-1}$ (resp. $F'_2 \in \mathcal{F}_{i-1}$) or $F'_1 = \emptyset$ (resp. $F'_2 = \emptyset$) such that $F_1 = \varphi_\Sigma(F'_1 \cup \varphi_\Sigma(x_i))$ (resp. $F_2 = \varphi_\Sigma(F'_2 \cup \varphi_\Sigma(x_i))$). Therefore[‡‡], in the case where $F'_1, F'_2 \in \mathcal{F}_{i-1}$ (the other cases are similar):

$$\begin{aligned} F^j &= \varphi_\Sigma(F_1 \cup F_2) \\ &= \varphi_\Sigma(\varphi_\Sigma(F'_1 \cup \varphi_\Sigma(x_i)) \cup \varphi_\Sigma(F'_2 \cup \varphi_\Sigma(x_i))) \\ &= \varphi_\Sigma(F'_1 \cup F'_2 \cup \varphi_\Sigma(x_i)) \\ &= \varphi_\Sigma(\varphi_\Sigma(F'_1 \cup F'_2) \cup \varphi_\Sigma(x_i)) \end{aligned}$$

$\varphi_\Sigma(F'_1 \cup F'_2)$ belongs to $\mathcal{F}_{i-1}$ by (21), and the proof is achieved.

[‡‡] Note that $\varphi_\Sigma(\varphi_\Sigma(X) \cup \varphi_\Sigma(X')) = \varphi_\Sigma(X \cup X')$ since φ_Σ is idempotent and extensive.

□

The function `Moore family` in Algorithm 3 is based on this characterization: it successively computes $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_n$, where $\mathcal{F}_i$ is computed from $\mathcal{F}_{i-1}$ using (22). It uses the Functions `closure` and `complete`. The use of a lexicographic tree to represent families leads to the following complexity:

Complexity 2 *Function* `Moore family` *in Algorithm 3 computes* $\mathbb{F}_\Sigma$ *in* $O(|\Sigma_d|^2 \cdot |S| + |\mathbb{F}_\Sigma| \cdot |S|(|S| + s(\Sigma_{do})))$.

Proof: The completion of Σ into Σ_{do} is done in $O(|\Sigma_d|^2 \cdot |S|)$ by Function `complete`. The initial computation of $\mathbb{F}_\Sigma = \{\texttt{closure}(\emptyset, \Sigma_{do})\}$ is done in $O(s(\Sigma_{do}))$. For each of the $|S|$ steps of the external `for` loop, a closure is computed by Function `closure` in $O(s(\Sigma_{do}))$ and an addition into $\mathbb{F}_\Sigma$ is done in $O(|S|)$. The same operations occur in the $|\mathbb{F}_\Sigma|$ steps of the internal `for` loop. The complexity $\mathcal{C}$ follows:

$$
\begin{aligned}
\mathcal{C} &= 0(|\Sigma_d|^2 \cdot |S| + s(\Sigma_{do}) + |S| \cdot (s(\Sigma_{do}) + |\mathbb{F}_\Sigma| \cdot (s(\Sigma_{do}) + |S|) + |S|)) \\
&= 0(|\Sigma_d|^2 \cdot |S| + s(\Sigma_{do}) + |S| \cdot s(\Sigma_{do}) + |S| \cdot |\mathbb{F}_\Sigma| \cdot s(\Sigma_{do}) + |S|^2 \cdot |\mathbb{F}_\Sigma| + |S|^2) \\
&\qquad \text{by developing the expression} \\
&= 0(|\Sigma_d|^2 \cdot |S| + |\mathbb{F}_\Sigma| \cdot |S| \cdot s(\Sigma_{do}) + |S|^2 \cdot |\mathbb{F}_\Sigma|) \\
&\qquad \text{by majoration} \\
&= 0(|\Sigma_d|^2 \cdot |S| + |\mathbb{F}_\Sigma| \cdot |S| \cdot (|S| + s(\Sigma_{do})))
\end{aligned}
$$

□

Name: `Moore family`
Input: An implicational system Σ on S
Output: The Moore family $\mathbb{F}_\Sigma$
begin
  Σ_{do} =`complete`(Σ)
  $\mathbb{F}_\Sigma = \{\texttt{closure}(\emptyset, \Sigma_{do})\}$
  foreach $x \in S$ **do**
    C =`closure`(x, Σ_{do})
    foreach $F \in \mathbb{F}_\Sigma$ **do**
      F' =`closure`($F \cup C, \Sigma_{do}$)
      add F' in $\mathbb{F}_\Sigma$
    add C in $\mathbb{F}_\Sigma$
  return $\mathbb{F}_\Sigma$
end

Algorithm 3: computation of $\mathbb{F}_\Sigma$

5 Conclusion and Perspectives

Implicational systems on a finite set S are formally linked to the notions of closure operators and Moore families (see the recent survey [CM04]). The present work addresses algorithmic aspects of implicational systems through the same notions: Given an IS Σ it proposes new algorithms to compute $\varphi_\Sigma(X)$ (the closure of a set $X \subseteq S$ by the operator associated to Σ) and $\mathbb{F}_\Sigma$ (the Moore family associated to Σ).

The computation of $\varphi_\Sigma(X)$ was addressed in several ways in [Mai83, MR92, Wil95]: Algorithms basically rely on a fix-point computation which iterates over Σ-implications. [Wil95] proposes improvements due to sophisticated data structures. Our approach is different: We choose to improve the shape of ISs so that the computation of $\varphi_\Sigma(X)$ can be performed by a single scanning of Σ-implications. Such ISs are said direct, or iteration-free. [CM04] presents the notion of full ISs, that are particular direct ISs whose axiomatic definition is very simple. Nevertheless the computation of the full IS inferred from Σ adds to Σ an exponential number of implications, thus is impracticable. Starting from the remark that some aspects of full ISs are redundant with properties of the closure operator we want to compute, we define a smaller direct IS Σ_d inferred from and equivalent to Σ. Then we optimize Σ_d into the direct-optimal IS Σ_{do} which is the unique IS of minimal size, equivalent to Σ and such that $\varphi_{\Sigma(X)}$ can be obtained by scanning only once its implications. The derived algorithms, based on the representation of ISs by lexicographic trees, compute Σ_{do} in $O(|\Sigma_d|^2 \cdot |S|)$ and $\varphi_\Sigma(X)$ from Σ_{do} in $O(s(\Sigma_{do}))$.

We finally address the computation of $\mathbb{F}_\Sigma$. Though $(\mathbb{F}_\Sigma, \subseteq)$ is a lattice, the construction of $\mathbb{F}_\Sigma$ we propose does not use existing methods that build a lattice using a binary relation between its join and meet-irreducible elements. Instead we characterize $\mathbb{F}_\Sigma$ using the closure operator φ_Σ and properties of the join-irreducible elements of $(\mathbb{F}_\Sigma, \subseteq)$. Due to the use of a lexicographic tree, we obtain an algorithm in $O(|S| \cdot (|\Sigma_d|^2 + |\mathbb{F}| \cdot |S| + |\mathbb{F}| \cdot s(\Sigma_{do}))$.

Potential Applications of the Computation of $\varphi(X)$ As explained in the introduction the algorithms related to ISs and Moore families we propose can be used for example in the field of knowledge systems. Another potential application of the computation of closures concerns the *static analysis* of programs by means of abstract interpretation. In a nutshell the static analysis of a program aims at obtaining as much information as possible on the set of its executions. However, a fully automatic approach has to be avoided. Applications are e.g. proofs of some safety properties on critical systems, aliasing analysis, etc. The approach relies on *non-standard executions* that perform computations using a description of values (*abstract* values) and not concrete ones. *Abstract interpretation* is a theory that expresses static analysis as a correspondence between the concrete semantics of a program and an abstract semantics guided by the property to be proved. It was introduced by Cousot and Cousot [CC77]. Informally the property to be proved induces the choice for a *concrete computation domain* C and an *abstract domain* A, connected by an *abstraction function* $\alpha\colon C \to A$ and a *concretization* one $\gamma\colon C \to A$. (α, γ) is a Galois connection that verifies the following properties: α and γ are monotonous; $\forall x_a \in A$, $x_a = \alpha \circ \gamma(x_a)$; $\forall x_c \in C$, $x_c \sqsubseteq_c \gamma \circ \alpha(x_c)$. Most of the work related to abstract interpretation use this formalism but, as mentioned in the early Cousot works and as extensively used by Giacobazzi (e.g. [GRS00, FGR96]), the theory of abstract interpretation can also be described by means of closure operators between concrete domain and an isomorphism of abstract domains. [GR98] addresses relational program analysis by means of *implications* between pairs of objects. It could be interesting to investigate if this particular relational abstract interpretation framework can benefit from our work. The link between works on ISs and works on systems of boolean implications in classical logic should also be examined.

Equivalent ISs This paper mentions several particular ISs that describe a given Moore family, more or less small with respect to their size: The full IS Σ_f that contains an exponential number of implications, the direct IS Σ_d, the direct-optimal IS Σ_{do}. Some researchers [GD86, CM04] have highlighted other smallest (e.g. in the sense of minimality, non-redundancy) representations of a Moore family by particular

ISs called *bases* (i.e. such a basis is unique and can generate any equivalent IS). The properties of these bases have been well-studied. Some of them provide a nice — though exponential — characterization of a basis from a given Moore family. They also imply that such a basis is not direct, so that a direct-optimal IS is not a basis.

An interesting problem is therefore the characterization of the direct-optimal ISΣ_{do} from a given Moore family $\mathbb{F}$, and its possibly polynomial generation from $\mathbb{F}$. As mentioned in introduction, this problem can be found in data-mining where the family of frequent closed itemsets is used to generate association rules [PBTL99].

Links between ISs and Representations of Lattices ISs are directly linked to lattices since $(\mathbb{F}_\Sigma, \subseteq)$ is a *lattice*, with some particular cases. For instance the Moore family 2^S associated to $\Sigma = \emptyset \times \emptyset$ and ordered by inclusion is a *boolean* lattice. Another case concerns an IS Σ whose premises and conclusions are singletons: They can be represented by a binary relation on S, therefore by an order P. $\mathbb{F}_\Sigma$ is then the set of ideals of P, which is union-stable [Mor64, Bir67]. So $(\mathbb{F}_\Sigma, \subseteq)$ is a *distributive* lattice and can be represented by (i.e. rebuilt from) the sub-order of its join-irreducible elements. Finally in Formal Concept Analysis [GW99] the *Galois* lattice, also called the *concept* lattice, is composed of two Moore families on a set G of objects and a set M of attributes respectively, associated to a binary relation on G and M called a formal concept: The esprit of FCA is to understand the concept lattice as one lattice (of formal concepts). This is the small difference, but which made FCA applicable in real world tasks. (Of course, one can find in the concept lattice the two Moore familes on G and M, resp., but in FCA that is not the primary way to look at it.)

A natural question is then to highlight links between ISs as representations of Moore families (i.e. lattices) and other representations of lattices like the sub-order of join-irreducible elements in the distributive case, the reduced bipartite order, the concepts, the arrows relations [Bir67, BM70, Wil83, BC02], etc.

Acknowledgements

The authors wish to thank anonymous referees for their useful comments on early versions of this paper.

References

[BC02] K. Bertet and N. Caspard. Doubling convex sets in lattices: characterizations and recognition algorithms. *Order*, 19:181–207, 2002.

[Bir67] G. Birkhoff. *Lattice theory*, volume 25. American Mathematical Society, 3rd edition, 1967.

[BM70] M. Barbut and B. Monjardet. *Ordre et classification, Algèbre et combinatoire*. Hachette, Paris, 1970. 2 tomes.

[CC77] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record on the Fourth Annual ACM SIGPLAN-SIGACT Sympsosium on Principles of Programming Languages*, pages 238–252, Los Angelès, California, USA, 1977. ACM Press.

[CM04] N. Caspard and B. Monjardet. Some lattices of closure systems on a finite set. *Discrete Mathematics and Theoretical Computer Sciences*, 6:163–190, 2004.

[FGR96] G. Filé, R. Giacobazzi, and F. Ranzato. A unifying view on abstract domain design. *ACM Computing Surveys*, 28(2):333–336, 1996.

[Gan84] B. Ganter. Two basic algorithms in concept lattices. Technical report, Technische Hochschule Darmstadt, 1984.

[GD86] J.L. Guigues and V. Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaire. *Math. Sci. Hum.*, 95:5–18, 1986.

[GR98] R. Giacobazzi and F. Ranzato. A logical model for relational abstract domains. *ACM Transactions on Programming Languages and Systems*, 20(5):1067–1109, 1998.

[GRS00] R. Giacobazzi, F. Ranzato, and F. Scozzari. Making abstract interpretations complete. *Journal of the ACM*, 47(2):361–416, 2000.

[GW99] B. Ganter and R. Wille. *Formal concept analysis, Mathematical foundations*. Springer Verlag, Berlin, 1999.

[HN96] M. Habib and L. Nourine. Tree structure for distributive lattices and its applications. *TCS*, 165:391–405, Octobre 1996.

[Mai83] D. Maier. *The Theory of Relational Databases*. Computer Sciences Press, 1983.

[MN96] M. Morvan and L. Nourine. Simplicial elimination scheme, extremal lattices and maximal antichains lattices. *Order*, 13:159–173, 1996.

[Mor64] J. Morgado. Note on the distributive closure operators by means of one axiom. *Portugal Maths*, 23:11–25, 1964.

[MR92] H. Mannila and K.J. Räihä. *The design of relational databases*. Addison-Wesley, 1992.

[NR99] L. Nourine and O. Raynaud. A fast algorithm for building lattices. In *Third International Conference on Orders, Algorithms and Applications*, Montpellier, France, august 1999.

[PBTL99] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In LLNCS Springer Verlag, editor, *ICDT'99*, volume 1540, pages 398–416, 1999.

[Wil83] R. Wille. Subdirect decomposition of concepts lattices. *Algebra Universalis*, 17:275–287, 1983.

[Wil94] M. Wild. A theory of finite closure spaces based on implications. *Advances in Mathematics*, 108:118–139, 1994.

[Wil95] M. Wild. Computations with finite closure systems and implications. In *Proceedings of the 1st Annual International Conference on Computing and Combinatorics (COCOON'95)*, volume 959 of *LNCS*, pages 111–120. Springer, 1995.