Computational and Applied Mathematics (2026) 45:433
https://doi.org/10.1007/s40314-026-03823-0

Direct-optimal systems of attribute implications in fuzzy formal concept analysis

Manuel Ojeda-Hernández[1] · **Domingo López-Rodríguez**[2]

Received: 14 October 2025 / Revised: 9 February 2026 / Accepted: 19 May 2026

Abstract
Efficient closure computation is a key challenge in fuzzy Formal Concept Analysis. Direct implicational systems, which allow for one-step closure calculation, offer a powerful solution, but their formalization and practical computation in the fuzzy setting have remained largely open problems. This paper provides a comprehensive contribution to fill this gap. We first extend the concept of a direct system to fuzzy attribute implications and establish a complete theoretical characterization via a novel Fuzzy Exchange Condition. Building on this, we define the notion of a direct-optimal system and develop the `DirectOptimal` algorithm, a provably correct, interleaved strategy for its computation. A rigorous experimental evaluation demonstrates that our proposed algorithm is orders of magnitude more efficient than naive sequential approaches, and we diagnose the distinct computational bottlenecks that cause these simpler methods to fail. This work thus delivers a complete and empirically validated pipeline, from theory to practice, for the efficient computation of direct-optimal bases.

Keywords Fuzzy Formal Concept Analysis · Systems of implications · Direct-optimal basis · Simplification Logic

Mathematics Subject Classification 03B52 · 68T30 · 06B15

1 Introduction

Functional dependencies, Horn clauses, and implications play the roles of if-then rules in different fields of mathematics, namely Relational Databases, Logic Programming, and Formal Concept Analysis.

Manuel Ojeda-Hernández and Domingo López-Rodríguez contributed equally to this work.

✉ Domingo López-Rodríguez
dominlopez@uma.es

Manuel Ojeda-Hernández
manuojeda@uma.es

[1] Department of Algebra, Geometry and Topology, Universidad de Málaga, Bvd. Louis Pasteur, 29071 Málaga, Spain

[2] Department of Applied Mathematics, Universidad de Málaga, Bvd. Louis Pasteur, 29071 Málaga, Spain

Published online: 12 June 2026

Formal Concept Analysis was introduced by Wille (1982) in the eighties as a mathematical tool to store, manage, and represent knowledge stored in relational data tables. Nevertheless, it is natural that some attributes can be neither true nor false but allow for some graduality. For instance, in a medical database, a patient can have a low, medium, high, or very high fever. Thus, fuzzy Formal Concept Analysis (FFCA) was introduced, first by Burusco and Fuentes-González (1994) where the set of truth-values was a lattice, and then by Pollandt (1997) and Bělohlávek (1998, 2002), whose works focus on the use of a complete residuated lattice as the set of truth-values. This paradigm was later extended to the multi-adjoint framework (Medina et al. 2009; Aragón et al. 2024, 2025).

Computing the closure of an element via an implicational system is an iterative procedure that runs until a fixed point is reached (Ganter and Obiedkov 2016). Due to this iteration, two lines of research arise naturally: one being the study of which properties ensure that bases have as few implications as possible, these studies involve research on pseudointents and quasi-closed elements (Guigues and Duquenne 1986; Ojeda-Hernández et al. 2022; Vychodil and Bělohlávek 2005); the other studies implicational systems that need only one iteration to compute the closure, the so-called direct systems (Bertet and Monjardet 2010; Rodríguez-Lorenzo et al. 2018). This paper follows the latter.

Consider, for instance, a logic-based recommender system, the database is fixed and each user has to be offered a recommendation, which is computed as a closure of a certain input. Thus, the basis of implications is computed only once, but several thousands of closures will be computed; therefore, having a larger basis of implications that ensures an almost immediate computation of the recommendation is worthwhile. Moreover, most feedback surveys allow for a graded punctuation, e.g. five-star range, 0–10 graded mark or graded linguistic labels such as {did not like it, liked it, loved it}; which emphasizes the need for a many-valued or fuzzy theoretical framework.

Despite the increase of computational complexity in the fuzzy setting and the amount of work done on the first line of research (Bělohlávek and Vychodil 2005, 2006; Ojeda-Hernández et al. 2022, 2023; Vychodil and Bělohlávek 2005; Cornejo et al. 2026), there are few contributions on direct systems in the fuzzy setting. In this line we highlight the work of Cordero et al. (2018), which opened the field. Intuitively, even if direct systems are larger than the stem, or Duquenne–Guigues basis, the implicational system computation is done all at once and from that point on closure computation is almost immediate. In cases where several closures have to be computed, direct systems arise as a suitable option. A major research goal would be the combination of both research lines mentioned above, that is, building a direct system with the fewest number of implications. This topic is known in the classical case as the direct-optimal basis of implications, where the computation of closures is much faster due to computing it with only one iteration and, at the same time, passing through a smaller number of implications in said iteration. This research line was studied by Rodríguez-Lorenzo et al. in the binary case (Rodríguez-Lorenzo et al. 2018).

In this paper, we present the definition of direct system of implications in the fuzzy setting. This property is characterized in algebraic terms and an algorithm to compute a direct system of implications from a complete system of implications is provided. The main contributions of this work are threefold. First, we provide a complete theoretical characterization of directness for fuzzy implicational systems via a novel Fuzzy Exchange Condition, bridging the gap between algebraic structure and computational behaviour. Second, we formalize the notion of a *direct-optimal* system and propose the `DirectOptimal` algorithm, an efficient, interleaved strategy for its computation, for which we prove correctness and termination. Finally, through a rigorous experimental evaluation, we demonstrate the overwhelming practical superiority of our interleaved algorithm compared to more naive, sequential approaches.

The paper is structured as follows. Section 2 recalls the fundamental concepts of fuzzy Formal Concept Analysis, Fuzzy Attribute Simplification Logic, and crisp direct bases. In Sect. 3, we formally introduce the notion of a direct system in the fuzzy setting and present our main theoretical result, which characterizes this property. Section 4 is dedicated to the pursuit of optimality, defining redundancy and direct-optimal systems, and Sect. 5 presents a novel algorithm for their computation. Section 6 provides a theoretical analysis of the computational complexity of the proposed algorithms. In Sect. 7, we conduct a comprehensive experimental evaluation to validate our proposals and analyse their scalability. Finally, Sect. 8 summarizes our findings and outlines several avenues for future research.

2 Preliminaries

In this section we will recall the notions needed to follow the results of the paper, which encompass FFCA, Fuzzy Attribute Simplification Logic and direct implicational systems.

Throughout this paper, let $\mathbb{L} = (L, \wedge, \vee, \otimes, \rightarrow, 0, 1)$ be a complete residuated lattice, which is an algebra where

- $(L, \wedge, \vee, 0, 1)$ is a complete lattice with 0 and 1 being the least and the greatest elements of L, respectively,
- $(L, \otimes, 1)$ is a commutative monoid (i.e., $\otimes$ is commutative, associative, and 1 is neutral with respect to $\otimes$), and
- $\otimes$ and $\rightarrow$ satisfy the so-called *adjointness property*: for all $a, b, c \in L$, we have that $a \otimes b \leq c$ iff $a \leq b \rightarrow c$.

This structure is utilized in mathematical fuzzy logics and their applications as structures of truth degrees with $\otimes$ and $\rightarrow$ used as truth functions of *fuzzy conjunction* and *fuzzy implication*, respectively (Hájek 2013). The unit interval with the Łukasiewicz, Gödel and Goguen pairs of t-norms and implications are examples of residuated complete lattices.

Let L^U be the set of $\mathbb{L}$-sets, or fuzzy sets, on the universe U. Operations with $\mathbb{L}$-sets are defined element-wise. For instance, $A \cup B \in L^U$ is defined as $(A \cup B)(u) = A(u) \vee B(u)$ for all $u \in U$. Binary $\mathbb{L}$-relations (binary fuzzy relations) on a set U can be thought of as $\mathbb{L}$-sets on the universe $U \times U$. That is, a binary $\mathbb{L}$-relation on U is a mapping $\rho \in L^{U \times U}$ assigning to each $x, y \in U$ a truth degree $\rho(x, y) \in L$ (a degree to which x and y are related by ρ). A well-known binary fuzzy relation is the subsethood degree relation $S\colon L^U \times L^U \rightarrow L$ that extends the idea of subsethood to the graded setting. For two fuzzy sets $A, B \in L^U$, the degree to which A is a subset of B is defined by $S(A, B) = \bigwedge_{x \in U}(A(x) \rightarrow B(x))$.

2.1 Fuzzy formal concept analysis

A fuzzy formal context is a tuple $\mathbb{K} = (G, M, I)$ where G and M are non-empty sets of objects and attributes, respectively and $I\colon G \times M \rightarrow L$ is a fuzzy relation between objects and attributes.

In this setting, $I(g, m)$ represents the degree to which object g satisfies attribute m. For fuzzy sets of objects and attributes $X \in L^G$, $Y \in L^M$ we define the concept-forming operators

$^{\uparrow}: L^G \to L^M$, $^{\downarrow}: L^M \to L^G$ by

$$X^{\uparrow}(m) = \bigwedge_{g \in G} X(g) \to I(g, m),$$
$$Y^{\downarrow}(g) = \bigwedge_{m \in M} Y(m) \to I(g, m).$$

These mappings form a fuzzy Galois connection and, as a consequence, both compositions $^{\uparrow\downarrow}$ and $^{\downarrow\uparrow}$ are closure operators. A pair $(A, B) \in L^G \times L^M$ is said to be a formal concept if $A^{\uparrow} = B$ and $B^{\downarrow} = A$. The set of all formal concepts is in fact a complete lattice, the so-called concept lattice. In FFCA there are two main knowledge structures, one is the already mentioned concept lattice and the other is the set of valid attribute implications in the formal context.

An attribute implication is an expression of the form $A \Rightarrow B$, read "A implies B", where $A, B \in L^M$. An implication $A \Rightarrow B$ is said to be valid in $\mathbb{K}$ if $B \subseteq A^{\downarrow\uparrow}$.

2.2 Fuzzy attribute simplification logic

A fuzzy set $X \in L^M$ is said to be a model of an implication $A \Rightarrow B$ if

$$\|A \Rightarrow B\|_X = S(A, X) \to S(B, X) = 1.$$

The set of all models is denoted by $Mod(A \Rightarrow B)$. A set of implications Σ is called an implicational system. The models of Σ, denoted by $Mod(\Sigma)$ is a system of fuzzy sets defined by

$$Mod(\Sigma) = \{X \in L^M : \|A \Rightarrow B\|_X = 1 \text{ for all } A \Rightarrow B \in \Sigma\}.$$

The validity of an implication $A \Rightarrow B$ with respect to an implicational system Σ is defined by

$$\|A \Rightarrow B\|_{\Sigma} = \bigwedge_{X \in Mod(\Sigma)} \|A \Rightarrow B\|_X.$$

Simplification Logic (Mora et al. 2012; Cordero et al. 2020) arose as a means to obtain smaller, more readable attribute implications in FCA, with no loss of knowledge. This approach is deeply linked to the optimality of systems, as studied in the crisp case by Rodríguez-Lorenzo et al. (2018). As a matter of fact, Bertet et al. gave a characterization result on direct-optimal systems, Theorem 2 in Bertet and Nebut (2004), that bears impressive similarities with the derivation rules of Simplification Logic.

Simplification Logic was extended to the fuzzy case in Bělohlávek et al. (2016), where the set of truth-values was a complete residuated lattice $\mathbb{L} = (L, \vee, \wedge, 0, 1, \otimes, \to)$. The axioms and derivation rules of Fuzzy Attribute Simplification Logic (FASL) are introduced below, for all $A, B, C, D \in L^M$, $c \in L$,

`[Ax]`	infer $AB \Rightarrow A$,	(Axiom)
`[Mul]`	from $A \Rightarrow B$ infer $c \otimes A \Rightarrow c \otimes B$,	(Multiplication)
`[Sim]`	from $A \Rightarrow B, C \Rightarrow D$ infer $AC \smallsetminus B \Rightarrow D$,	(Simplification)

where $\smallsetminus: L \to L$ is an operation that satisfies for all $a, b, c \in L$,

$$a \smallsetminus b \leq c \text{ if and only if } a \leq b \vee c.$$

Note that, if $a \leq b$, then $a \smallsetminus b = 0$. In this paper, the juxtaposition of two sets X and Y is used to simplify the notation for the union of those sets, i.e., XY stands for $X \cup Y$.

The operation $\setminus$ is extended to fuzzy sets in a pointwise manner, that is, for any $A, B \in L^M, m \in M$

$$(A \setminus B)(m) = A(m) \setminus B(m).$$

The provability of FASL is denoted by $\vdash$, i.e., $T \vdash A \Rightarrow B$ means that $A \Rightarrow B$ is provable from T using `[Ax]`, `[Mul]`, and `[Sim]`.

Theorem 1 (Bělohlávek et al. 2016) *Let L and M be finite, let T be a set of formulas. For the axiomatic system of FASL we have $T \vdash A \Rightarrow B$ if and only if $\|A \Rightarrow B\|_T = 1$.*

In FASL the following logical equivalences hold.

Theorem 2 (Bělohlávek et al. 2016) *For any $A, B, C, D \in L^M$, the following equivalences hold true*:

`[DeEq]` $\{A \Rightarrow B\} \equiv \{A \Rightarrow B \setminus A\}$;
`[UnEq]` $\{A \Rightarrow B, A \Rightarrow C\} \equiv \{A \Rightarrow BC\}$;
`[SiEq]` *If* $A \subseteq C$ *then* $\{A \Rightarrow B, C \Rightarrow D\} \equiv \{A \Rightarrow B, A(C \setminus B) \Rightarrow D \setminus B\}$.

2.3 Crisp direct bases

Given an implicational system $\Sigma = \{A \Rightarrow B \mid A, B \subseteq M\}$, consider the operator $\pi_\Sigma \colon 2^M \to 2^M$ defined as,

$$\pi_\Sigma(X) = X \cup \bigcup \{B \mid A \subseteq X, A \Rightarrow B \in \Sigma\}.$$

This operator is inflationary and isotone, which makes it a preclosure operator in the terms of Čech (1966). It is not a closure operator in general, since it is not necessarily idempotent. However, a closure operator can be obtained by iterating π_Σ. Thus, we obtain $\varphi_\Sigma \colon 2^M \to 2^M$ defined by

$$\varphi_\Sigma(X) = \pi_\Sigma(X) \cup \pi_\Sigma^2(X) \cup \pi_\Sigma^3(X) \cup \cdots.$$

Definition 1 An implicational system Σ is said to be direct if $\varphi_\Sigma(X) = \pi_\Sigma(X)$, for all $X \subseteq M$, that is, if

$$\varphi_\Sigma(X) = X \cup \bigcup \{B \mid A \subseteq X, A \Rightarrow B \in \Sigma\}.$$

Direct systems of implications are of interest since they allow a fast computation of the closure operator. Whereas general systems of implications need several runs through all the implications to get the closure, direct system need only one. Therefore, there is a trade-off between the cost of computing the direct implicational system, which is in general higher than others, and the cost of computing closures, which is much faster with direct systems of implications.

The following result characterizes the directness of an implicational system in terms of a single property.

Lemma 1 (Rodríguez-Lorenzo et al. 2018) *An implicational system Σ is direct if and only if it satisfies the following condition*: *for all $A \Rightarrow B, C \Rightarrow D \in \Sigma$, $b \in B \setminus A$ and $d \in D \setminus (A \cup C)$, if $b \in C$ then there exists $G \Rightarrow H \in \Sigma$ such that $G \subseteq A \cup C$ and $d \in H$.*

3 Direct systems of implications in the fuzzy setting

In this section, the problem of defining and characterizing the directness of an implicational system in the fuzzy setting is studied. Notice that this line of research is different from the one in Cordero et al. (2018). In the cited work, the authors consider implications of the form $A \overset{\vartheta}{\Rightarrow} B$ where $A, B \subseteq M$, that is, both A and B are crisp sets and the truth-value ϑ is the degree of validity of the implication, also called confidence in the setting of association rules. In this paper, an implication is, as defined in Sect. 2.1, an expression $A \Rightarrow B$, where $A, B \in L^M$.

The notation Σ for an implicational system is commonly used in papers that deal with implications in general. Nevertheless, papers concerning logic use the notation T from "theory". From this point on, we will follow the notation of Rodríguez-Lorenzo et al. (2018), and denote implicational systems by Σ.

Throughout the rest of the paper, unless otherwise stated, the complete residuated lattice L will be a discretization of the unit interval, hence L will be a finite chain.

Given an implicational system Σ, the operator $\pi_\Sigma \colon L^M \to L^M$ is defined as (see Bělohlávek and Vychodil 2017),

$$\pi_\Sigma(X)(m) = X(m) \vee \bigvee \{S(A, X) \otimes B(m) \mid A \Rightarrow B \in \Sigma\}.$$

Again, π_Σ is not idempotent and the closure operator is obtained by iteration. Thus, we obtain $\varphi_\Sigma \colon L^M \to L^M$ defined by

$$\varphi_\Sigma(X) = \pi_\Sigma(X) \cup \pi_\Sigma^2(X) \cup \pi_\Sigma^3(X) \cup \cdots .$$

Definition 2 An implicational system Σ is said to be direct if $\varphi_\Sigma(X) = \pi_\Sigma(X)$, for all $X \in L^M$, that is, if for all $m \in M$,

$$\varphi_\Sigma(X)(m) = X(m) \vee \bigvee \{S(A, X) \otimes B(m) \mid A \Rightarrow B \in \Sigma\}.$$

Lemma 2 *Let $X \in L^M$, Σ an implicational system and $m \in M$ such that $\pi_\Sigma(X)(m) > X(m)$. Then there exists an implication $G \Rightarrow H \in \Sigma$ such that $\pi_\Sigma(X)(m) = S(G, X) \otimes H(m)$.*

Proof By definition of implicational closure,

$$\pi_\Sigma(X)(m) = X(m) \vee \bigvee \{S(A, X) \otimes B(m) \mid A \Rightarrow B \in \Sigma\}.$$

Since L is linear and finite, the supremum is reached and is indeed a maximum. Thus, since $\pi_\Sigma(X)(m) > X(m)$, there exists an implication $G \Rightarrow H \in \Sigma$ such that $S(G, X) \otimes H(m) = \pi_\Sigma(X)(m)$. □

Notice that the use of a finite chain as the set of truth-values allows us to find one implication that fulfils the condition, rather that a collection of implications whose supremum reaches the desired value. This will be of paramount importance throughout the rest of the section.

Definition 3 Let L be a lattice and $x \in L$. The *height* of x is defined as the length of the longest chain from the bottom element of L to x, and will be denoted by $h(x)$.

Remark 1 It is easy to see that if $x \leq y$ for $x, y \in L$, then $h(x) \leq h(y)$. If the order relation between x and y is strict, then the same inequality holds for their heights.

Definition 4 Let us consider the following metric $p\colon \Sigma \times L^M \times L^M \to \mathbb{R}$ defined as

$$p(A \Rightarrow B, Y, X) = \sum_{\substack{x \in M \\ X(x) < S(A,Y) \otimes A(x)}} h\left(S(A, Y) \otimes A(x)\right).$$

Remark 2 Observe that the sum is well-defined since L is finite. Also notice that $p(A \Rightarrow B, Y, X) = 0$ if and only if $S(A, Y) \otimes A \subseteq X$.

In the crisp case, the directness of an implicational system is characterized by the so-called exchange condition, which is seen in Lemma 1. This condition is related to Simplification Logic, the use of `[Sim]` can be hinted by the nature of the formula. The following is an extension of the exchange condition to the fuzzy setting.

Definition 5 An implicational system Σ is said to satisfy the fuzzy exchange condition if for every pair of implications $A \Rightarrow B, C \Rightarrow D \in \Sigma$, for all $m, n \in M$ and all $c, d \in L$, if $c \otimes B(m) > c \otimes A(m)$, $d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n)$ and $d \otimes C(m) < c \otimes B(m)$ then there exists an implication $G \Rightarrow H \in \Sigma$ such that

$$S(G, c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)) \otimes H(n) \geq d \otimes D(n).$$

Lemma 3 *Let Σ be an implicational system, $X, Y \in L^M$, $Y \subseteq \pi_\Sigma(X)$ and $C \Rightarrow D \in \Sigma$ such that $S(C, Y) \otimes D(m) > \pi_\Sigma(X)(m)$ for some $m \in M$. If Σ satisfies the fuzzy exchange condition and $p(C \Rightarrow D, Y, X) > 0$, then there exist an implication $G \Rightarrow H \in \Sigma$ and a fuzzy set $Z \in L^M$ such that $p(G \Rightarrow H, Z, X) < p(C \Rightarrow D, Y, X)$. Moreover, the following inequality holds $S(G, Z) \otimes H(m) \geq S(C, Y) \otimes D(m)$.*

Proof Let Σ satisfy the fuzzy exchange condition, $C \Rightarrow D \in \Sigma$, $X, Y \in L^M$ such that $Y \subseteq \pi_\Sigma(X)$, $S(C, Y) \otimes D(m) > \pi_\Sigma(X)(m)$ and $p(C \Rightarrow D, Y, X) > 0$. Then, there exists an attribute $n \in M$ such that, $S(C, Y) \otimes C(n) > X(n)$.

Since $Y \subseteq \pi_\Sigma(X)$, by Lemma 2 there exists an implication $A \Rightarrow B \in \Sigma$ such that $S(A, X) \otimes B(n) \geq S(C, Y) \otimes C(n)$. Hence, we have two implications $A \Rightarrow B, C \Rightarrow D \in \Sigma$ and two truth-values $S(A, X), S(C, Y) \in L$ such that

$$S(A, X) \otimes B(n) \geq S(C, Y) \otimes C(n) > X(n) \geq S(A, X) \otimes A(n), \text{ as stated,}$$

$$S(C, Y) \otimes D(m) \overset{(i)}{>} \pi_\Sigma(X)(m) \geq Y(m) \overset{(ii)}{\geq} S(C, Y) \otimes C(m) \text{ and}$$

$$S(C, Y) \otimes D(m) \overset{(iii)}{>} \pi_\Sigma(X)(m) \geq X(m) \overset{(iv)}{\geq} S(A, X) \otimes A(m),$$

where (i) and (iii) hold by hypothesis, (ii) and(iv) are due to modus ponens. Therefore,

$$S(C, Y) \otimes D(m) > S(A, X) \otimes A(m) \vee S(C, Y) \otimes C(m).$$

Thus, applying the fuzzy exchange condition there exists $G \Rightarrow H \in \Sigma$ such that

$$S(G, S(A, X) \otimes A \cup S(C, Y) \otimes C \smallsetminus S(A, X) \otimes B) \otimes H(m) \geq S(C, Y) \otimes D(m).$$

Hence, we have found an implication $G \Rightarrow H$ and a set

$$Z = S(A, X) \otimes A \cup (S(C, Y) \otimes C \smallsetminus S(A, X) \otimes B)$$

that satisfy the required properties.

Assume $o \in M$ satisfies $X(o) < S(G, Z) \otimes G(o)$, then

$$\begin{aligned} S(G, Z) \otimes G(o) &\leq Z(o) \\ &\overset{\text{(i)}}{=} S(A, X) \otimes A(o) \vee S(C, Y) \otimes C(o) \smallsetminus S(A, X) \otimes B(o) \\ &\overset{\text{(ii)}}{\leq} X(o) \vee S(C, Y) \otimes C(o) \smallsetminus S(A, X) \otimes B(o) \\ &\overset{\text{(iii)}}{\leq} S(C, Y) \otimes C(o) \smallsetminus S(A, X) \otimes B(o) \\ &\overset{\text{(iv)}}{\leq} S(C, Y) \otimes C(o), \end{aligned}$$

where (i) is the definition of Z, (ii) is by *modus ponens*, (iii) is since $X(o) < S(G, Z) \otimes G(o)$, thus the supremum is greater than $S(G, Z) \otimes G(o)$ if and only if $S(G, Z) \otimes G(o) < S(C, Y) \otimes C(o) \smallsetminus S(A, X) \otimes B(o)$, finally (iv) is due to the properties of the difference operator.

In particular for $n \in M$, either $S(G, Z) \otimes G(n) \leq X(n)$ and $S(G, Z) \otimes G(n)$ is not summed, or $X(n) < S(G, Z) \otimes G(n)$ is satisfied and we have

$$\begin{aligned} S(G, Z) \otimes G(n) &\leq Z(n) \\ &\leq S(C, Y) \otimes C(n) \smallsetminus S(A, X) \otimes B(n) \\ &\overset{(i)}{<} S(C, Y) \otimes C(n), \end{aligned}$$

where (i) holds since $S(A, X) \otimes B(n) > S(C, Y) \otimes C(n)$ and the difference operator ensures $S(C, Y) \otimes C(n) \smallsetminus S(A, X) \otimes B(n) = 0$.

That is, for every attribute $j \in M$, it is $S(G, Z) \otimes G(j) \leq S(C, Y) \otimes C(j)$, where the inequality is strict for $j = n$. Therefore,

$$\begin{aligned} p(G \Rightarrow H, Z, X) &= \sum_{\substack{x \in M \\ X(x) < S(G,Z) \otimes G(x)}} h(S(G, Z) \otimes G(x)) \\ &= \sum_{\substack{x \in M \smallsetminus \{n\} \\ X(x) < S(G,Z) \otimes G(x)}} h(S(G, Z) \otimes G(x)) + h(S(G, Z) \otimes G(n)) \cdot \mathbf{1}_{X(n) < S(G,Z) \otimes G(n)} \\ &< \sum_{\substack{x \in M \smallsetminus \{n\} \\ X(x) < S(C,Y) \otimes C(x)}} h(S(C, Y) \otimes C(x)) + h(S(C, Y) \otimes C(n)) \\ &= \sum_{\substack{x \in M \\ X(x) < S(C,Y) \otimes C(x)}} h(S(C, Y) \otimes C(x)) = p(C \Rightarrow D, Y, X). \end{aligned}$$

□

Theorem 3 *Let $\mathbb{K}$ be a formal context. An implicational system Σ is direct if and only if it satisfies the fuzzy exchange condition.*

Proof For the forward implication, let $\Sigma = \{A_i \Rightarrow B_i \mid i \in I\}$ be a direct implicational system (i.e. $\varphi_\Sigma = \pi_\Sigma$), $A \Rightarrow B, C \Rightarrow D \in \Sigma$ and $c, d \in L$. Then, by `[Mul]`, $c \otimes A \Rightarrow c \otimes B$ and $d \otimes C \Rightarrow d \otimes D$ are valid implications in $\mathbb{K}$. In addition, applying `[Sim]`, we get $(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)) \Rightarrow d \otimes D$ is a valid implication in $\mathbb{K}$, that is,

$$d \otimes D \subseteq \varphi_\Sigma((c \otimes A \cup d \otimes C \smallsetminus c \otimes B)) = \pi_\Sigma((c \otimes A \cup (d \otimes C \smallsetminus c \otimes B))).$$

Assume there are $m, n \in M$ such that $c \otimes B(m) > c \otimes A(m)$, $d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n)$ and $d \otimes C(m) < c \otimes B(m)$. Then,

$$d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n) \geq (c \otimes A \cup (d \otimes C \smallsetminus c \otimes B))(n),$$

and since Σ is direct by Lemma 2 there exists an implication $G \Rightarrow H \in \Sigma$ such that

$$S(G, c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)) \otimes H(n) \geq d \otimes D(n).$$

For the converse implication, let Σ be an implicational system satisfying the fuzzy exchange condition. One must show that $\varphi_\Sigma(X) = \pi_\Sigma(X)$, or equivalently that $\pi_\Sigma(X) = \pi_\Sigma^2(X)$, or still equivalently (since π_Σ is inflationary) that $\pi_\Sigma^2(X) \subseteq \pi_\Sigma(X)$. Assume that there exists X with $\pi_\Sigma(X) \subsetneq \pi_\Sigma^2(X)$, i.e., there exists $m \in M$ such that

$$\pi_\Sigma(X)(m) < \pi_\Sigma^2(X)(m).$$

Then, by Lemma 2, there exists $A \Rightarrow B \in \Sigma$ with

$$\pi_\Sigma(X)(m) < \pi_\Sigma^2(X)(m) = S(A, \pi_\Sigma(X)) \otimes B(m).$$

Let us consider the metric in Definition 4 applied particularly on the implication $A \Rightarrow B$, $\pi_\Sigma(X)$ and X. Recall that $p(A \Rightarrow B, \pi_\Sigma(X), X) = 0$ if and only if $S(A, \pi_\Sigma(X)) \otimes A \subseteq X$, which gives $S(A, \pi_\Sigma(X)) \otimes B \subseteq \pi_\Sigma(X)$, a contradiction. Thus, one has $p(A \Rightarrow B, \pi_\Sigma(X), X) > 0$ and we are in the conditions to apply Lemma 3 to get an implication $A_1 \Rightarrow B_1 \in \Sigma$ and a set $Z_1 \in L^M$ such that $p(A_1 \Rightarrow B_1, A_1, Z_1) < p(A \Rightarrow B, A, \pi_\Sigma(X))$ and

$$S(A_1, Z_1) \otimes B_1(m) \geq S(A, \pi_\Sigma(X)) \otimes B(m) > \pi_\Sigma(X)(m).$$

At this point, Lemma 3 can be applied iteratively until $p(A_k, Z_k) = 0$, convergence is ensured since both L and M are finite. In addition,

$$S(A_k, Z_k) \otimes B_k(m) \geq \cdots \geq S(A, \pi_\Sigma(X)) \otimes B(m) > \pi_\Sigma(X)(m).$$

Notice that this is a contradiction since $p(A_k, Z_k) = 0$ implies $S(A_k, Z_k) \otimes A_k \subseteq X$, therefore $S(A_k, Z_k) \otimes B_k(m) \leq \pi_\Sigma(X)(m)$. Hence, $\pi_\Sigma(X) = \pi_\Sigma^2(X)$ and Σ is direct. □

The characterization of directness provided in Theorem 3 is not merely descriptive; it offers a constructive method to transform any implicational system into an equivalent direct one. The fuzzy exchange condition must hold for every pair of implications within the system, which suggests an iterative, fixed-point approach. Starting with an initial system Σ, one can repeatedly add the implications required to satisfy the condition for all pairs until the system becomes closed under this operation.

The core of this procedure is a function that, for any given pair of implications $A \Rightarrow B$ and $C \Rightarrow D$, generates the set of new implications needed to fulfill the fuzzy exchange condition. This function, `AddDerived`, introduced in Ojeda-Hernández and López-Rodríguez (2025), is presented in Algorithm 1. Its logic is grounded in Fuzzy Attribute Simplification

Logic, systematically generating the consequences $G \Rightarrow H$ that arise from applying the simplification rule ([Sim]) to scaled versions of the input implications.

Algorithm 1: AddDerived($A \Rightarrow B, C \Rightarrow D, L$)

```
Σ := ∅
foreach c, d ∈ L do
    if d ⊗ C ⟍ c ⊗ B ≠ ∅ and d ⊗ D ⟍ c ⊗ A ≠ ∅ then
        G := c ⊗ A ∪ (d ⊗ C ⟍ c ⊗ B)
        H := d ⊗ D ⟍ c ⊗ A
        Σ := Σ ∪ {G ⇒ H}
return Σ
```

In Ojeda-Hernández and López-Rodríguez (2025), this function was wrapped inside a procedure BuildDirectSystem which iteratively called Algorithm 1 to saturate the implication system and reach a fixpoint, a direct system equivalent to the one used as its input.

Example 1 To illustrate the outcome of this procedure, let us recall the example from Ojeda-Hernández and López-Rodríguez (2025). Consider the initial system

$$\Sigma = \{a_{0.5}\, b_{0.5}\, d \Rightarrow a\, b\, c\, e, \quad a_{0.5}\, d\, e \Rightarrow a\, b\, c_{0.5}\}.$$

After the iterative application of the algorithm until a fixed point is reached, the resulting direct system is

$$\begin{aligned}\Sigma_d = \big\{&a_{0.5}\, b_{0.5}\, d \Rightarrow a\, b\, c\, e, \quad a_{0.5}\, d\, e \Rightarrow a\, b\, c, \quad d_{0.5}\, e_{0.5} \Rightarrow a_{0.5}\, b_{0.5}\, c_{0.5},\\ &d_{0.5} \Rightarrow a_{0.5}\, b_{0.5}\, c_{0.5}\, e_{0.5}, \quad d\, e \Rightarrow a\, b\, c, \quad d\, e_{0.5} \Rightarrow a\, b\, c\, e, \quad d \Rightarrow a\, b\, c\, e\big\}.\end{aligned}$$

As the example demonstrates, this constructive process yields a direct system, but at the cost of a notable increase in its cardinality, suggesting the presence of redundancy. This observation motivates the need for more concise systems, a topic to be addressed in Sect. 4. In fact, the following result will allow us to refine the AddDerived function itself.

Theorem 4 *Let Σ be an implicational system, then the following are equivalent*:

1. Σ *satisfies the fuzzy exchange condition*
2. Σ *is a direct system*
3. *For each pair $A \Rightarrow B, C \Rightarrow D \in \Sigma$, for all $m, n \in M$ and all $c, d \in L$, if $c \otimes B(m) > c \otimes A(m)$, $d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n)$ and $d \otimes C(m) < c \otimes B(m)$ then*

$$\pi_\Sigma\left(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)\right)(n) \geq d \otimes D(n).$$

Proof 1 implies 2 is a consequence of Ojeda-Hernández and López-Rodríguez (2025, Theorem 3).

For 2 implies 3, assume Σ is a direct basis of implications, that is, $\varphi_\Sigma = \pi_\Sigma$ and let $A \Rightarrow B, C \Rightarrow D \in \Sigma$ and $c, d \in L$ such that $c \otimes B(m) > c \otimes A(m)$, $d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n)$ and $d \otimes C(m) < c \otimes B(m)$. Since $A \Rightarrow B$ is a valid implication in $\mathbb{K}$, $c \otimes A \Rightarrow c \otimes B$ is also a valid implication by [Mul]. Similarly, $C \Rightarrow D$ is a valid implication in $\mathbb{K}$ and [Mul] yields $d \otimes C \Rightarrow d \otimes D$ is a valid implication in $\mathbb{K}$. Applying [Sim] to these two new implications we get that $c \otimes A \cup (d \otimes C \smallsetminus c \otimes B) \Rightarrow d \otimes D$ is a valid implication in $\mathbb{K}$, or equivalently, since Σ is direct

$$d \otimes D \subseteq \varphi_\Sigma(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)) = \pi_\Sigma(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)).$$

Recall now that subsethood is equivalent to an inequality of the truth degrees for each attribute, in particular for attribute $n \in M$ we get,

$$d \otimes D(n) \leq \pi_\Sigma(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B))(n).$$

Lastly, for 3 implies 1, assume condition 3 holds and let $A \Rightarrow B, C \Rightarrow D \in \Sigma$, $m, n \in M$ and $c, d \in L$, such that $c \otimes B(m) > c \otimes A(m)$, $d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n)$ and $d \otimes C(m) < c \otimes B(m)$, then condition 3 ensures

$$\pi_\Sigma(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B))(n) \geq d \otimes D(n).$$

Applying the definition of π_Σ directly we get,

$$\begin{aligned} d \otimes D(n) &\leq \pi_\Sigma(c \otimes A \cup (d \otimes C \smallsetminus c \otimes B))(n) \\ &= c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)(n) \\ &\quad \vee \bigvee \{S(X, c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)) \otimes Y(n) \mid X \Rightarrow Y \in \Sigma\}. \end{aligned}$$

By hypothesis,

$$d \otimes D(n) > c \otimes A(n) \vee d \otimes C(n) \geq c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)(n),$$

therefore, there exists some $G \Rightarrow H \in \Sigma$ such that

$$S(G, c \otimes A \cup (d \otimes C \smallsetminus c \otimes B)) \otimes H(n) \geq d \otimes D(n).$$

□

With the insight provided by Theorem 4, we can introduce a refinement to the `AddDerived` function. The key idea is to prune the generated implications by removing any information from the consequent that is already derivable in a single step using only the parent implications from which it was derived. This ensures that the newly added implications are more concise, contributing only the knowledge that is not immediately obtainable. The refined procedure is detailed in Algorithm 2.

Algorithm 2: `AddDerivedWithPruning`$(A \Rightarrow B, C \Rightarrow D, L)$

```
Σ := ∅
foreach c, d ∈ L do
    if d ⊗ C ∖ c ⊗ B ≠ ∅ and d ⊗ D ∖ c ⊗ A ≠ ∅ then
        G := c ⊗ A ∪ (d ⊗ C ∖ c ⊗ B)
        H := d ⊗ D ∖ c ⊗ A
        H_red := H ∖ π_{A⇒B, C⇒D}(G)
        if H_red ≠ ∅ then
            Σ := Σ ∪ {G ⇒ H_red}
return Σ
```

In the computation of H_{red}, we simplify the consequent by removing the knowledge that is already provided by a single application of the π operator with the parent implications. Consequently, the implicational system added to complete the direct system is more succinct.

Example 2 Revisiting the example from Ojeda-Hernández and López-Rodríguez (2025) and applying the construction algorithm with the pruning strategy yields the following result:

$$\begin{aligned} \Sigma_d = \{&d\, e_{0.50} \Rightarrow a\, b\, c\, e, d\, e \Rightarrow a\, b\, c, d \Rightarrow a\, b\, c\, e, \\ &a_{0.50}\, d\, e \Rightarrow a\, b\, c, a_{0.50}\, b_{0.50}\, d \Rightarrow a\, b\, c\, e\}. \end{aligned}$$

As can be seen by comparing the results, the pruning strategy incorporated into `AddDerivedWithPruning` guarantees the generation of a direct system with a smaller number of implications than the original approach. The study of direct systems of minimum cardinality, however, remains to be addressed and will be the focus of the following section.

4 In pursue of optimality

This section delves into the search for an appropriate notion of non-redundancy in the framework of fuzzy implication systems. Throughout this section, we will suppose that the studied system, Σ, is *reduced*, that is, in the right-hand side of the implications, no attribute appears with a degree equal to or lower than its degree in the corresponding left-hand side. Any implication $A \Rightarrow B$ can be transformed (maintaining the logical equivalence) into reduced form via substitution by $A \Rightarrow B \smallsetminus A$ (using `[DeEq]`). In particular, if $B(m) \neq 0$, we have $A(m) < B(m)$ for all $m \in M$.

The study of redundancy will be done in a nested manner, first the redundancy of attributes will be considered within an implication and then the redundancy of an implication within an implicational system. Intuitively, redundancy will be considered as not providing new information to the closure operator, therefore the attribute is not needed in this particular implication. This is formalized below.

Definition 6 Let Σ be an implicational system and $A \Rightarrow B \in \Sigma$, an attribute $m \in M$ is said to be non-redundant in $A \Rightarrow B$ with respect to Σ if

$$\bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} < B(m).$$

Proposition 5 *Let Σ be an implicational system, $A \Rightarrow B \in \Sigma$, $m \in M$ and let $\Sigma' = (\Sigma \smallsetminus \{A \Rightarrow B\}) \cup \{A \Rightarrow (B \smallsetminus \{m\})\}$. Then, m is a redundant attribute in $A \Rightarrow B$ if and only if $\pi_\Sigma = \pi_{\Sigma'}$.*

Proof We will first prove the forward implication.

Assume $A \Rightarrow B \in \Sigma$ and $m \in M$ be a redundant attribute in $A \Rightarrow B$ wrt Σ and let $X \in L^M$. Since m is redundant,

$$\bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \geq B(m).$$

Then,

$$\begin{aligned}
\pi_{\Sigma'}(X)(m) &= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma'\} \\
&= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A,X) \otimes (B \smallsetminus \{m\})(m) \\
&\overset{\text{(i)}}{=} X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\overset{\text{(ii)}}{\geq} X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\quad \vee \bigvee\{S(C,A) \otimes S(A,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\overset{\text{(iii)}}{=} X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\quad \vee S(A,X) \otimes \Big(\bigvee\{S(C,A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\}\Big) \\
&\overset{\text{(iv)}}{\geq} X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A,X) \otimes B(m) \\
&= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma\} = \pi_{\Sigma}(X),
\end{aligned}$$

where (i) is due to $(B \smallsetminus \{m\})(m) = 0$, (ii) is due to the transitivity of the fuzzy order S, (iii) is due to the distributivity of the product $\otimes$ over the supremum $\vee$ and (iv) is due to m being redundant.

In addition,

$$\begin{aligned}
\pi_{\Sigma'}(X)(m) &= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma'\} \\
&= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A,X) \otimes (B \smallsetminus \{m\})(m) \\
&\overset{\text{(i)}}{\leq} X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A,X) \otimes B(m) \\
&= X(m) \vee \bigvee\{S(C,X) \otimes D(m) \mid C \Rightarrow D \in \Sigma\} = \pi_{\Sigma}(X),
\end{aligned}$$

where (i) is due to $B \smallsetminus \{m\} \subseteq B$.

Let now $m' \in M$ be an attribute $m \neq m'$, then

$$\begin{aligned}
\pi_{\Sigma'}(X)(m') &= X(m') \vee \bigvee\{S(C,X) \otimes D(m') \mid C \Rightarrow D \in \Sigma'\} \\
&= X(m') \vee \bigvee\{S(C,X) \otimes D(m') \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A,X) \otimes (B \smallsetminus \{m\})(m') \\
&\overset{\text{(i)}}{=} X(m') \vee \bigvee\{S(C,X) \otimes D(m') \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A,X) \otimes B(m')
\end{aligned}$$

$$= X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma\} = \pi_\Sigma(X),$$

where (i) is due to $a \smallsetminus 0 = a$ for all $a \in L$. Therefore, $\pi_\Sigma = \pi_{\Sigma'}$.

Conversely, assume $\pi_\Sigma = \pi_{\Sigma'}$, then

$$\begin{aligned}
\pi_{\Sigma'}(A)(m) &= A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma'\} \\
&= A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\quad \vee S(A, A) \otimes (B \smallsetminus \{m\})(m) \\
&= A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\}. \\
\pi_\Sigma(A)(m) &= A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma\} \\
&= A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\quad \vee S(A, A) \otimes B(m) \\
&= A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\quad \vee B(m).
\end{aligned}$$

Therefore,

$$B(m) \leq A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\},$$

or equivalently, $B(m) \leq \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\}$, since either $B(m) = 0$ and it is smaller than anything, or $B(m) \neq 0$ and we have $A(m) < B(m)$ by our initial assumption. Hence, by definition, m is redundant in $A \Rightarrow B$. □

The last result highlights the meaning of redundancy in this setting, that is, the attribute $m \in M$ is redundant in $A \Rightarrow B$ wrt Σ if and only if it does not provide additional information. Moreover, removing redundant attributes does not affect the directness of an implicational system.

Proposition 6 *Let Σ be an implicational system, $A \Rightarrow B \in \Sigma$ and $m \in M$ be a redundant attribute in $A \Rightarrow B$ wrt Σ. Let $\Sigma' = \Sigma \smallsetminus \{A \Rightarrow B\} \cup \{A \Rightarrow (B \smallsetminus \{m\})\}$, then Σ is direct if and only if Σ' is direct.*

Proof Let Σ be an implicational system, $A \Rightarrow B \in \Sigma$ and $m \in M$ be a redundant attribute in $A \Rightarrow B$ wrt Σ. Then, by Proposition 5, $\pi_\Sigma = \pi_{\Sigma'}$. Assume Σ is a direct system of implications, then $\varphi_\Sigma = \pi_\Sigma = \pi_{\Sigma'}$. We want to prove $\pi_{\Sigma'} = \varphi_{\Sigma'}$. Notice that φ_Σ is a closure operator, in particular it is idempotent. Therefore, $\pi_{\Sigma'}$ is idempotent and $\pi_{\Sigma'} = \varphi_{\Sigma'}$.

The converse implication is analogous and is then omitted. □

Similarly, this notion can be extended to implications as follows.

Definition 7 An implication $A \Rightarrow B \in \Sigma$ is said to be redundant if every attribute $m \in M$ is redundant in $A \Rightarrow B$ wrt Σ.

Proposition 7 *The following statements are equivalent*:

1. *$A \Rightarrow B$ is non-redundant*
2. *for some $m \in M$,*

$$A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} < B(m).$$

3. *for some* $X \in L^M$ *and* $m \in M$,

$$X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} < S(A, X) \otimes B(m).$$

Proof We will prove this in a circular manner.

1. implies 2. Assume $A \Rightarrow B$ is non-redundant, then there exists an attribute $m \in M$ that is non-redundant. By Definition 6,

$$A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} < B(m).$$

2. implies 3. Assume 2 holds, then there exists $m \in M$ such that

$$A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} < B(m).$$

In particular, for $X = A \in L^M$, we have

$$\begin{aligned} X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} \\ = A(m) \vee \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} \\ < B(m) = S(A, A) \otimes B(m) = S(A, X) \otimes B(m). \end{aligned}$$

Thus, statement 3. holds.

Lastly, 3. implies 1. Assume there exists a set $X \in L^M$ and an attribute $m \in M$ such that

$$\begin{aligned} S(A, X) \otimes B(m) &> X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} \\ &\geq \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} \\ &\geq \bigvee \{S(C, A) \otimes S(A, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} \\ &\geq S(A, X) \otimes \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} \end{aligned}$$

Therefore,

$$S(A, X) \otimes \bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} < S(A, X) \otimes B(m),$$

which implies, as L is a chain,[1]

$$\bigvee \{S(C, A) \otimes D(m) \mid C \Rightarrow D \in \Sigma \setminus \{A \Rightarrow B\}\} < B(m),$$

which, by Definition 6, implies that m is non-redundant in $A \Rightarrow B$ with respect to Σ. □

Again, the meaning behind the non-redundancy of an implication is that it provides some extra information to the closure operator. Conversely, a redundant implication does not provide information to the closure operator and can be discarded. Moreover, discarding redundant implications does not mess with the directness of the implicational system.

Proposition 8 *The following statements are equivalent*:

1. $A \Rightarrow B \in \Sigma$ *is redundant.*
2. $\pi_\Sigma = \pi_{\Sigma \setminus \{A \Rightarrow B\}}$.

[1] If $x \leq y$ then $z \otimes x \leq z \otimes y$. Since $z \otimes y < z \otimes x$ we have that $z \otimes x \not\leq z \otimes y$, which implies $x \not\leq y$ which, in a chain, gives $y < x$.

Proof Assume $A \Rightarrow B \in \Sigma$ is redundant, then by Proposition 7, for all $X \in L^M$ and $m \in M$,

$$X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \geq S(A, X) \otimes B(m).$$

Therefore, let $X \in L^M$ and $m \in M$,

$$\begin{aligned}
\pi_\Sigma(X)(m) &= X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma\} \\
&= X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&\qquad \vee S(A, X) \otimes B(m) \\
&= X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \\
&= \pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}(X),
\end{aligned}$$

which proves $\pi_\Sigma = \pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$.

Conversely, assume $\pi_\Sigma = \pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$ and let $X \in L^M$ and $m \in M$, then,

$$\begin{aligned}
&X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \vee S(A, X) \otimes B(m) \\
&\quad = X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma\} \\
&\quad = \pi_\Sigma(X)(m) = \pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}(X)(m) \\
&\quad = X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\}.
\end{aligned}$$

Therefore,

$$X(m) \vee \bigvee \{S(C, X) \otimes D(m) \mid C \Rightarrow D \in \Sigma \smallsetminus \{A \Rightarrow B\}\} \geq S(A, X) \otimes B(m).$$

This is valid for all $X \in L^M$ and $m \in M$, thus by Proposition 7 $A \Rightarrow B$ is redundant. □

Proposition 9 *Let Σ be an implicational system and $A \Rightarrow B \in \Sigma$ be a redundant implication. Then Σ is direct if and only if $\Sigma \smallsetminus \{A \Rightarrow B\}$ is direct.*

Proof Let Σ be an implicational system and $A \Rightarrow B \in \Sigma$ be a redundant implication in Σ. Then, by Proposition 7, $\pi_\Sigma = \pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$. Assume Σ is a direct system of implications, then $\varphi_\Sigma = \pi_\Sigma = \pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$. We want to prove $\pi_{\Sigma \smallsetminus \{A \Rightarrow B\}} = \varphi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$. Notice that φ_Σ is a closure operator, in particular it is idempotent. Therefore, $\pi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$ is idempotent and $\pi_{\Sigma \smallsetminus \{A \Rightarrow B\}} = \varphi_{\Sigma \smallsetminus \{A \Rightarrow B\}}$.

The converse implication is analogous and is then omitted. □

This discourse motivates the following definition.

Definition 8 An implicational system Σ is said to be *direct-optimal* if it is direct and every implication and every attribute in Σ is non-redundant.

Intuitively, a system is direct-optimal if neither its implications or attributes could be reduced without losing the logical equivalence, or the directness of the system.

With these results in mind, we can propose an algorithmic approach to the removal of redundancy in an implicational system. Note that if the input system Σ is direct, by the previous results, the output will be a direct-optimal system, as we will prove afterwards.

Algorithm 3: `RemoveRedundancyAux(`Σ`)`

```
change ← false
Σ_f ← ∅
foreach A ⇒ B ∈ Σ do
    if |Σ| = 1 then
        Σ_f ← Σ                          // Only one implication, return
        break
    Σ' ← Σ ∖ {A ⇒ B}
    D ← π_Σ'(A)                          // Closure of A using Σ'
    B_new ← B ∖ D                        // Part of B not derivable from the rest
    if B_new ≠ ∅ then
        // Add the (potentially reduced) implication to Σ_f
        Σ ← Σ ∖ {A ⇒ B} ∪ {A ⇒ B_new}
        Σ_f ← Σ_f ∪ {A ⇒ B_new}
        if B_new ≠ B then
            change ← true                // An implication was changed
    else
        // The original A ⇒ B was fully redundant wrt Σ'
        Σ ← Σ ∖ {A ⇒ B}
        change ← true                    // An implication was removed
                                         // Σ_f now contains the reduced implications
return change, Σ_f
```

Lemma 4 *Let Σ be an implicational system and let Σ_r be the output of* `RemoveRedundancyAux(`Σ`)`, *i.e., the system resulting from a single pass of the auxiliary reduction algorithm* (*Algorithm 3*). *Then, the one-step closure operator is preserved by this transformation, that is, $\pi_\Sigma = \pi_{\Sigma_r}$.*

Proof The algorithm `RemoveRedundancyAux` transforms the input system Σ into the output system Σ_r through a finite sequence of two types of operations on the implications $A \Rightarrow B \in \Sigma$: (1) the removal of a redundant attribute from the consequent B, or (2) the removal of the entire implication $A \Rightarrow B$. We will show that both operations preserve the π operator.

1. *Attribute removal*: An attribute m is removed from the consequent B of an implication $A \Rightarrow B$ if $B(m) \leq \pi_{\Sigma \setminus \{A \Rightarrow B\}}(A)(m)$. By Definition 6, this is precisely the condition for m to be a redundant attribute in $A \Rightarrow B$ with respect to Σ. According to Proposition 5, removing a redundant attribute from an implication is an operation that preserves the one-step closure operator.
2. *Implication removal*: An implication $A \Rightarrow B$ is removed entirely if its consequent is reduced to the empty set, which occurs if and only if $B \subseteq \pi_{\Sigma \setminus \{A \Rightarrow B\}}(A)$. As a direct consequence of Propositions 7 and 8, this condition is equivalent to the implication $A \Rightarrow B$ being redundant in Σ. These propositions further establish that an implication is redundant if and only if its removal preserves the one-step closure operator, i.e., $\pi_\Sigma = \pi_{\Sigma \setminus \{A \Rightarrow B\}}$.

Since the transformation from Σ to Σ_r is composed entirely of a sequence of operations that individually preserve the π operator, the final system Σ_r must be equivalent to the initial system Σ with respect to their one-step closure operators. Thus, $\pi_\Sigma = \pi_{\Sigma_r}$. □

Let us consider the iterative application of `RemoveRedundancyAux` until the output variable `change` is false, that is, until a fixed point is obtained. We will refer to this iterative procedure as `RemoveRedundancy` (whose input is an implicational system), and omit its pseudocode due to its simplicity and for the sake of readability. Thus, `RemoveRedundancy` is a fixed-point wrapper that iteratively invokes the auxiliary reduction logic defined in Algorithm 3 until the system stabilizes.

Theorem 10 *Let Σ be an implicational system and let Σ_r be the result of applying* `RemoveRedundancy`(Σ). *The following properties hold*:

1. Σ_r *is equivalent to* Σ.
2. Σ_r *is irredundant*; *it contains no redundant implications or attributes.*
3. *If* Σ *is a direct system*, *then* Σ_r *is a direct-optimal system.*

Proof The proof is structured by addressing each of the three claims.
1. Equivalence The `RemoveRedundancy` algorithm reaches a fixed point by iteratively applying the transformations in `RemoveRedundancyAux`, which has been proven to preserve the π operator (see Lemma 4). Thus, $\pi_{\Sigma_k} = \pi_{\Sigma_{k+1}}$ for all k, therefore $\pi_\Sigma = \pi_{\Sigma_r}$ and both systems are equivalent.
2. Irredundancy The algorithm terminates when a fixed point is reached, meaning a full pass of `RemoveRedundancyAux` produces no changes. Then Σ_r is this fixed point. We proceed by contradiction.

- Assume Σ_r contains a redundant implication $A \Rightarrow B$. By definition, $B \subseteq \pi_{\Sigma_r \setminus \{A \Rightarrow B\}}(A)$. An application of `RemoveRedundancyAux` to Σ_r would compute $B_{\text{new}} = \varnothing$ and remove the implication, contradicting that Σ_r is a fixed point.
- Assume Σ_r contains an implication $A \Rightarrow B$ with a redundant attribute m. By Definition 6, $B(m) \leq \pi_{\Sigma_r \setminus \{A \Rightarrow B\}}(A)(m)$. An application of `RemoveRedundancyAux` would yield $B_{\text{new}}(m) < B(m)$, thus modifying the implication. This again contradicts that Σ_r is a fixed point.

Therefore, Σ_r contains no redundant implications or attributes.
3. Direct-Optimality Assume the input system Σ is direct. As commented before, $\pi_{\Sigma_r} = \pi_\Sigma$ hence if Σ is direct, then Σ_r is direct as well. A system that is both direct and irredundant is, by definition, direct-optimal. □

The procedure to build a direct-optimal system from an arbitrary system Σ, according to what has been proved, consists of the following two stages:

1. Construction of a direct system: $\Sigma_d \leftarrow$ `BuildDirectSystem`(Σ, L); and
2. Removal of the redundant items: $\Sigma_{do} \leftarrow$ `RemoveRedundancy`(Σ_d).

Example 3 Let us illustrate the behaviour of the `RemoveRedundancy` algorithm on the direct implicational system Σ_d obtained in the previous section, which consists of the following seven implications:

1. $a_{0.5}\, b_{0.5}\, d \Rightarrow a\, b\, c\, e$
2. $a_{0.5}\, d\, e \Rightarrow a\, b\, c$
3. $d_{0.5}\, e_{0.5} \Rightarrow a_{0.5}\, b_{0.5}\, c_{0.5}$

4. $d_{0.5} \Rightarrow a_{0.5}\, b_{0.5}\, c_{0.5}\, e_{0.5}$
5. $d\, e \Rightarrow a\, b\, c$
6. $d\, e_{0.5} \Rightarrow a\, b\, c\, e$
7. $d \Rightarrow a\, b\, c\, e$

The algorithm proceeds in passes, iterating until the system stabilises.

First Pass The algorithm checks each implication for redundancy against the rest of the system. Let us examine the first implication, $A \Rightarrow B$, where $A = \{a_{0.5}, b_{0.5}, d\}$ and $B = \{a, b, c, e\}$. To do this, we compute $D = \pi_{\Sigma'}(A)$, where Σ' is the system formed by the other six implications. Note that Σ' contains the very general rule (7), $d \Rightarrow a\, b\, c\, e$. Since A contains the attribute d with degree 1, the premise of rule (7) is fully satisfied ($S(\{d\}, A) = 1$). The application of this single rule during the one-pass computation of the π operator yields:

$$\pi_{\Sigma'}(A) \supseteq A \cup (S(\{d\}, A) \otimes \{a, b, c, e\}) = A \cup \{a, b, c, e\} = \{a, b, c, d, e\}.$$

The resulting set D clearly covers the original consequent B. Thus, $B_{\text{new}} = B \setminus D = \varnothing$, and the first implication is proven to be redundant and is removed.

A similar line of reasoning demonstrates that implications (2) through (6) are also consequences of the most general rule (7). For instance, to check rule (5), $d\, e \Rightarrow a\, b\, c$, we note that its premise contains d, which again triggers rule (7) in Σ' and makes the original consequent fully derivable.

The only implication that survives this process is (7) itself. When checking $d \Rightarrow a\, b\, c\, e$, the reduced system Σ' (which now contains rules 1-6) does not contain any rule powerful enough to derive the entire consequent $\{a, b, c, e\}$ from the premise $\{d\}$. Therefore, the implication is kept.

At the end of the first pass, the system has been drastically reduced to a single implication:

$$\Sigma_1 = \{d \Rightarrow a\, b\, c\, e\}.$$

Since the system has changed, the algorithm proceeds to a second pass.

Second Pass The algorithm now runs on the system Σ_1. Since it contains only one implication, no further reductions are possible. The algorithm detects no changes, the stability condition is met, and the process terminates.

The final direct-optimal system returned by the algorithm is:

$$\Sigma_{do} = \{d \Rightarrow a\, b\, c\, e\}.$$

This example illustrates how the `RemoveRedundancy` procedure effectively prunes a direct system, eliminating subsumed rules to converge on a more compact and optimal, yet logically equivalent, representation.

5 A combined algorithm for direct-optimality

While the construction of a direct system and the removal of redundancy can be seen as two distinct sequential stages, a more powerful approach is to combine them into a single, iterative algorithm. Interleaving saturation and reduction allows the process to converge more effectively. For instance, a reduction step may simplify the system, making the subsequent saturation phase more efficient. Conversely, the saturation step may uncover a new, more general implication that allows the next reduction phase to prune a significant amount of previously held information. This interplay motivates the combined fixed-point algorithm, `DirectOptimal`, presented in Algorithm 4.

The `DirectOptimal` procedure iteratively applies a saturation stage, to enforce directness, followed by a reduction stage, to remove redundancy. The loop continues until the system stabilises, meaning a full pass of both stages results in no changes. The saturation logic is encapsulated in the `SaturationPass` subroutine, detailed in Algorithm 5, which exhaustively applies the `AddDerivedWithPruning` function (Algorithm 2), making the saturation stage more efficient.

Algorithm 4: `DirectOptimal`(Σ_{in}, L)

Input: An implicational system Σ_{in}, a lattice L
Output: A direct-optimal implicational system Σ_{out}

```
// Initialize the system
Σ ← Σ_in
// Iterate until a fixed point is reached
repeat
    // Stage 1: Single pass of saturation
    change_1, Σ_temp ← SaturationPass(Σ)
    if not change_1 then                                   // System already stable
        Σ := Σ_temp
        break
    // Stage 2: Single pass of redundancy removal
    change_2, Σ ← RemoveRedundancyAux(Σ_temp)
until not change_1 and not change_2
Σ_out ← Σ
return Σ_out
```

Algorithm 5: `SaturationPass`(Σ_{in}, L)

Input: An implicational system Σ_{in}, a lattice L
Output: The system after one pass of saturation Σ_{out}, and a boolean flag `change`

```
Σ ← Σ_in
change ← false
Σ_derived ← ∅
// Combine every pair of implications exactly once
foreach A ⇒ B ∈ Σ and C ⇒ D ∈ Σ do
    Σ_new ← AddDerived(A ⇒ B, C ⇒ D, L)
    Σ_derived ← Σ_derived ∪ Σ_new
// Merge the newly derived rules into the system
foreach G ⇒ H ∈ Σ_derived do
    if there is no implication in Σ with premise G then
        Σ ← Σ ∪ {G ⇒ H}
        change ← true                                      // A change was made
    else
        Let H_old be the consequent of G in Σ
        H_merged ← H_old ∪ H
        if H_merged ≠ H_old then
            Update the consequent of G to H_merged in Σ
            change ← true                                  // A change was made
Σ_out ← Σ
return change, Σ_out
```

The following technical result establishes the key to the correction and convergence of `DirectOptimal`.

Lemma 5 *Let $(\Sigma_k)_{k\in\mathbb{N}}$ be the sequence of implicational systems at the beginning of each iteration of the* `DirectOptimal` *algorithm (Algorithm 4). The sequence of their corresponding one-step closure operators, $(\pi_{\Sigma_k})_{k\in\mathbb{N}}$, is monotonic non-decreasing. That is, for any L-fuzzy set $X \in L^M$ and for all $k \geq 0$, the following inclusion holds*:

$$\pi_{\Sigma_k}(X) \subseteq \pi_{\Sigma_{k+1}}(X).$$

Proof Let us consider the transformation from Σ_k to Σ_{k+1} within a single iteration of the algorithm's main loop. This transformation consists of two phases.

1. Saturation phase: The algorithm first computes an intermediate system $\Sigma'_k = \text{SaturationPass}(\Sigma_k)$. This procedure is monotonic with respect to the implicational system; it only adds new rules or enlarges the consequents of existing ones. Therefore, it holds that $\Sigma_k \subseteq \Sigma'_k$. The π operator is defined as a supremum over the implications in the system. As the supremum over a larger set is always greater than or equal to the supremum over a subset, it follows directly that:
$$\pi_{\Sigma_k}(X) \subseteq \pi_{\Sigma'_k}(X) \quad \forall X \in L^M$$
2. Reduction phase: The algorithm then computes the next state $\Sigma_{k+1} = \texttt{RemoveRedundancyAux}(\Sigma'_k)$. As shown in Lemma 4, this transformation preserves the π operator, that is, $\pi_{\Sigma_{k+1}} = \pi_{\Sigma'_k}$.

Combining the results from both phases, we can construct the following chain of relations for any $X \in L^M$:
$$\pi_{\Sigma_k}(X) \subseteq \pi_{\Sigma'_k}(X) = \pi_{\Sigma_{k+1}}(X)$$
This proves that the sequence $(\pi_{\Sigma_k})_{k\in\mathbb{N}}$ is monotonic non-decreasing, which completes the proof. □

The correctness and behaviour of the `DirectOptimal` algorithm are formally stated in the following theorem.

Theorem 11 *The* `DirectOptimal` *algorithm* (*Algorithm 4*), *given an initial system of implications* Σ_{in}, *terminates and returns a direct-optimal system* Σ_{do} *that is equivalent to* Σ_{in}.

Proof The proof is structured in three parts, addressing each of the theorem's claims: termination, equivalence, and the direct-optimal nature of the output.
1. Termination The algorithm's state is determined by the implicational system Σ. The set of attributes M and the lattice of truth-values L are finite, which implies that the universe of all possible implicational systems is also finite.

According to Lemma 5, the sequence of one-step closure operators $(\pi_{\Sigma_k})_{k\in\mathbb{N}}$ is monotonic non-decreasing. A non-decreasing sequence of functions over a finite domain must converge in a finite number of steps. That is, there exists an iteration N such that for all $k \geq N$, $\pi_{\Sigma_k} = \pi_{\Sigma_N}$.

Once the π operator stabilizes, the `SaturationPass` will no longer add any new information that alters the one-step closure. Subsequently, the `RemoveRedundancyAux` pass will also produce a system with the same π operator. As the state transitions of the algorithm are determined entirely by these operations, the system Σ_k itself must also reach a fixed point, causing the algorithm to terminate.
2. Equivalence We need to prove that the output Σ_{do} is equivalent to the input Σ_{in}, meaning $\varphi_{\Sigma_{do}} = \varphi_{\Sigma_{in}}$. We show that the closure operator φ is an invariant of the main loop. Let Σ_k be the system at the start of an iteration.

- Saturation phase: The intermediate system $\Sigma'_k = \text{SaturationPass}(\Sigma_k)$ is created by adding only semantic consequences of Σ_k. This operation ensures that the resulting system Σ'_k is logically equivalent to Σ_k, so the closure operator remains unchanged: $\varphi_{\Sigma'_k} = \varphi_{\Sigma_k}$.

- Reduction phase: The next system is $\Sigma_{k+1} = \texttt{RemoveRedundancyAux}(\Sigma'_k)$. According to Lemma 4, this operation preserves the one-step operator, $\pi_{\Sigma_{k+1}} = \pi_{\Sigma'_k}$. Since the closure operator φ is the idempotent extension of π, this equality implies the preservation of the closure operator as well: $\varphi_{\Sigma_{k+1}} = \varphi_{\Sigma'_k}$.

By chaining these equalities, we get $\varphi_{\Sigma_{k+1}} = \varphi_{\Sigma'_k} = \varphi_{\Sigma_k}$. Thus, the closure operator is an invariant of the loop. By induction, the final system Σ_{do} must be equivalent to the initial system Σ_{in}.

3. Direct-Optimality Let Σ_{do} be the system at the fixed point where the algorithm terminates.

- Directness: If Σ_{do} were not direct, its one-step operator $\pi_{\Sigma_{do}}$ would not be idempotent. The `SaturationPass` would then necessarily add implications to enforce the fuzzy exchange condition, causing a change in the system. This contradicts the algorithm's termination condition. Therefore, Σ_{do} must be direct.
- Irredundancy: If Σ_{do} contained any redundant implication or attribute, the `RemoveRedundancyAux` phase would modify the system. This would also contradict the termination condition. Therefore, Σ_{do} must be irredundant.

Since the output system Σ_{do} is direct, irredundant, and has been shown to be equivalent to Σ_{in}, it is, by definition, a direct-optimal system. □

Example 4 (Execution Trace of `DirectOptimal`) To illustrate the algorithm's behaviour, we trace its execution on our running example. The process unfolds in iterative passes, each comprising a saturation and a pruning stage, until the system stabilises.

Pass 1 The algorithm begins with the initial two-implication system.

- Saturation stage: The process starts by combining the initial rules. This generates a set of new candidate implications, and the system is expanded to a direct, but larger, set of five rules:

$$\{d\, e_{0.50} \Rightarrow a\, b\, c\, e,\ d\, e \Rightarrow a\, b\, c_{0.50}, \qquad d \Rightarrow a\, b\, c\, e,$$
$$a_{0.50}\, d\, e \Rightarrow a\, b\, c,\ a_{0.50}\, b_{0.50}\, d \Rightarrow a\, b\, c\, e\}$$

  The most significant discovery during this phase is the powerful and general implication $d \Rightarrow a\, b\, c\, e$.
- Pruning stage: The algorithm then analyzes this expanded five-implication system. The newly discovered general rule proves to be decisive. When the other four implications are evaluated, the algorithm finds that their consequents are now fully derivable in a single step from their premises, thanks to the presence of $d \Rightarrow a\, b\, c\, e$. Consequently, these four rules are identified as redundant and are eliminated. The pass concludes with a dramatically simplified system, Σ_1, containing only the single most general rule:

$$\Sigma_1 = \{d \Rightarrow a\, b\, c\, e\}$$

Pass 2 The second pass begins with the highly compact system Σ_1 from the previous step.

- Saturation stage: The algorithm attempts to combine the single rule with itself. While candidates are generated, none of them provide new information, so the system remains unchanged.
- Pruning stage: With only one rule, no reductions are possible, and the system is again unchanged.

Termination Since the system at the end of Pass 2 is identical to the system at the end of Pass 1, the algorithm has reached a fixed point and terminates. The final output is the direct-optimal system $\Sigma_{do} = \{d \Rightarrow a\, b\, c\, e\}$.

This toy example may give the false impression that all the knowledge of the implicational system can be condensed into a single pass to obtain a direct-optimal system. However, in the general case, this is not the case, as demonstrated in the following example.

Example 5 We explicitly trace the execution of the `DirectOptimal` algorithm on a small example to understand the benefits of this procedure. The process iteratively alternates between a saturation phase (generating new knowledge) and a pruning phase (removing redundancy) until convergence.

Initialization The algorithm starts with the input system Σ_0:

$$\Sigma_0 = \left\{ \begin{array}{l} b_{0.5}\, c\, e_{0.5} \Rightarrow a\, b\, d_{0.5}, \\ b\, c_{0.5}\, d_{0.5}\, e \Rightarrow a_{0.5}, \\ a_{0.5}\, c_{0.5}\, d_{0.5}\, e_{0.5} \Rightarrow b_{0.5}\, d \end{array} \right\}.$$

Pass 1

- Saturation phase: The algorithm combines the rules in Σ_0. This single pass generates derived implications, expanding the system to a set of 8 intermediate rules. Note the appearance of rules like $c_{0.5}\, d_{0.5}\, e_{0.5} \Rightarrow d$:

$$\Sigma_1^{sat} = \left\{ \begin{array}{ll} c_{0.5}\, d_{0.5}\, e_{0.5} \Rightarrow d, & c\, e_{0.5} \Rightarrow a\, b\, d_{0.5}, \\ b_{0.5}\, c\, e_{0.5} \Rightarrow a\, b\, d, & b\, c_{0.5}\, e \Rightarrow a_{0.5}, \\ b\, c_{0.5}\, d_{0.5}\, e \Rightarrow a_{0.5}\, d, & a_{0.5}\, c_{0.5}\, e_{0.5} \Rightarrow b_{0.5}\, d, \\ a_{0.5}\, c_{0.5}\, d_{0.5}\, e_{0.5} \Rightarrow b_{0.5}\, d, & a_{0.5}\, c\, d_{0.5}\, e_{0.5} \Rightarrow a\, b \end{array} \right\}.$$

- Pruning phase: The algorithm checks Σ_1^{sat} for redundancy. Several complex rules are found to be subsumed by simpler ones (e.g., $b\, c_{0.5}\, e \Rightarrow a_{0.5}$ is removed as it becomes redundant in the presence of the other rules). The system is reduced to 4 rules:

$$\Sigma_1 = \left\{ \begin{array}{l} c_{0.5}\, d_{0.5}\, e_{0.5} \Rightarrow d, \\ c\, e_{0.5} \Rightarrow a\, b, \\ b_{0.5}\, c\, e_{0.5} \Rightarrow d, \\ a_{0.5}\, c_{0.5}\, e_{0.5} \Rightarrow d \end{array} \right\}.$$

Pass 2

- Saturation phase: The algorithm performs a new saturation pass on Σ_1. It discovers a new, highly general rule: $c_{0.5}\, e_{0.5} \Rightarrow d$. This is added to the candidate set:

$$\Sigma_2^{sat} = \left\{ \begin{array}{ll} c_{0.5}\, e_{0.5} \Rightarrow d, & c_{0.5}\, d_{0.5}\, e_{0.5} \Rightarrow d, \\ c\, e_{0.5} \Rightarrow a\, b\, d, & b_{0.5}\, c\, e_{0.5} \Rightarrow d, \\ a_{0.5}\, c_{0.5}\, e_{0.5} \Rightarrow d & \end{array} \right\}.$$

- Pruning phase: The discovery of $c_{0.5}\, e_{0.5} \Rightarrow d$ is decisive. The algorithm detects that the rules requiring extra premises (like $d_{0.5}$, $b_{0.5}$, or $a_{0.5}$) to imply d are now redundant. They are pruned, leaving the minimal system:

$$\Sigma_2 = \left\{ \begin{array}{l} c_{0.5}\, e_{0.5} \Rightarrow d, \\ c\, e_{0.5} \Rightarrow a\, b \end{array} \right\}.$$

Pass 3 (Termination) Saturating Σ_2 produces candidates that are identical to or subsumed by Σ_2. As no new implications are added, the algorithm terminates. The final direct-optimal system is:

$$\Sigma_{do} = \{c_{0.5}\, e_{0.5} \Rightarrow d, \quad c\, e_{0.5} \Rightarrow a\, b\}.$$

This trace exemplifies the algorithm's effective strategy: it first expands the system to discover latent and powerful general knowledge, and then leverages this new knowledge to prune all subsumed information, thereby converging efficiently on a compact and optimal basis.

For contrast, the *two-stage* method has been applied to the same initial Σ_0.

- *Full saturation*: Unlike `DirectOptimal`, this method pushes the saturation to the limit before attempting any reduction. It performs 7 internal saturation passes, exploding the system to a set of 27 implications (omitted for the sake of readability). This set includes many highly redundant and complex rules (e.g., $a_{0.5}\, b\, c\, e \Rightarrow a$) alongside the optimal ones.
- *Full pruning*: Finally, it prunes this massive set. While it successfully identifies the same final basis Σ_{do}, the computational cost is significantly higher: the efficiency gap is evident in the operation counts. While `DirectOptimal` required only 174 calculations of the π operator, the two-stage method required 6473 calculations to reach the exact same result. The iterative pruning strategy of `DirectOptimal` prevents the combinatorial explosion observed in the two-stage approach.

6 Computational complexity analysis

While the preceding sections have established the correctness of the proposed algorithms, a formal analysis of their computational complexity is essential to understand their practical feasibility and performance trade-offs. The generation of a direct-optimal basis is known to be a computationally hard problem. In this section, we analyze the complexity of our algorithms with respect to the primary parameters of the problem: the number of attributes, denoted by $|M|$; the size of the finite lattice of truth-values, $|L|$; and the size of the implicational system, $|\Sigma|$.

6.1 Primitive operations

The overall complexity of the algorithms is determined by a few core operations that are used repeatedly. A precise analysis requires understanding their individual costs.

- Fuzzy set operations: Basic operations on fuzzy sets in L^M, such as union ($\cup$), fuzzy subtraction ($\smallsetminus$), and scalar multiplication ($\otimes$), require iterating through all attributes. Their complexity is therefore linear in the number of attributes, i.e., $O(|M|)$.
- One-step closure ($\pi_\Sigma(X)$): The computation of $\pi_\Sigma(X)$ for a fuzzy set X involves iterating through each of the $|\Sigma|$ implications. For each implication $A \Rightarrow B$, computing the similarity $S(A, X)$ takes $O(|M|)$ time. Subsequent operations are dominated by the final union, which is also $O(|M|)$. Thus, the complexity of a single $\pi_\Sigma(X)$ computation is $O(|\Sigma| \cdot |M|)$.
- Derivation (`AddDerived`): The `AddDerived` function (Algorithm 1) iterates through all $|L|^2$ pairs of scalars from the lattice. The body of this loop consists of fuzzy set operations of complexity $O(|M|)$. Its total complexity is therefore $O(|L|^2 \cdot |M|)$.

6.2 Analysis of algorithms

We now analyze the algorithmic strategies that rely on global passes over the entire implicational system.

The* two-stage *approach This is the most straightforward strategy, consisting of two distinct and sequential fixed-point computations: a full saturation followed by a full reduction.

- Stage 1: Saturation. A direct system is constructed by repeatedly calling a saturation pass (such as `SaturationPass`) until no new implications are generated. A single pass combines every pair of implications, so its complexity is $O(|\Sigma|^2 \cdot |L|^2 \cdot |M|)$. This process can cause an explosive growth in the number of implications. If k_{sat} is the number of passes and $|\Sigma_{\max}|$ is the maximum size of the system during this stage, the total complexity is dominated by the final passes, approaching $k_{\text{sat}} \cdot O(|\Sigma_{\max}|^2 \cdot |L|^2 \cdot |M|)$. The potentially exponential size of $|\Sigma_{\max}|$ makes this stage the primary bottleneck.
- Stage 2: Reduction. The large, direct, but highly redundant system $\Sigma_{\max}$ is then pruned. This involves repeatedly calling a reduction pass (such as `RemoveRedundancyAux`) until no further changes occur. A single reduction pass has a complexity of $O(|\Sigma|^2 \cdot |M|)$, as it checks each implication against the rest of the system.

The main drawback of this approach is its *brute-force* nature. It first generates a massive amount of information, much of it redundant, only to expend significant effort afterwards to clean it up.

The* `DirectOptimal` *Algorithm This algorithm (Algorithm 4) represents a more refined strategy. Instead of completing the saturation and reduction phases entirely, it interleaves single passes of each until a global fixed point is reached: a single iteration of the main `while` loop consists of one call to the procedure `SaturationPass` (Algorithm 5) followed by one call to the function `RemoveRedundancyAux` (Algorithm 3).

The complexity of an iteration k with a system of size $|\Sigma_k|$ is the sum of the costs of each pass: $O(|\Sigma_k|^2 \cdot |L|^2 \cdot |M|) + O(|\Sigma_k|^2 \cdot |M|)$, which is dominated by the saturation cost.

While the worst-case complexity remains high, this interleaved approach is practically superior to the two-stage method. The reduction pass inside the loop actively prunes the system, keeping the size of $|\Sigma_k|$ more manageable. By removing redundant implications as they are discovered, the algorithm prevents them from participating in the costly quadratic pairing of the next saturation pass. This strategy avoids the full combinatorial explosion of the naive saturation, representing a more intelligent search for the direct-optimal basis.

7 Experimental evaluation

The theoretical complexity analysis established high worst-case computational bounds for global-pass algorithms. This section presents a focused empirical evaluation designed to quantify the practical performance differences between the sequential two-stage strategy (TS) and the more refined, interleaved approach of the `DirectOptimal` algorithm (DO). The objectives are: (1) to empirically validate the performance hierarchy between these algorithms, and (2) to analyse the operational costs that explain the observed differences in execution time. Following this comparative analysis, we will outline the methodology for a deeper scalability study focused exclusively on the superior `DO` algorithm.

7.1 Experimental setup

To provide a rigorous empirical validation of our theoretical findings, this section details the experimental methodology. We outline the specific algorithmic configurations under review, the process for generating controlled synthetic datasets, and the performance metrics captured to assess their efficiency and scalability. This detailed description is intended to ensure our results are both transparent and reproducible.

Algorithms compared Our evaluation focuses on three algorithmic configurations:

- `TS-NoPruning`: The baseline Two-Stage approach using the basic `AddDerived` function.
- `TS-Pruning`: The Two-Stage approach enhanced with the pruning strategy in `AddDerivedWithPruning`.
- `DO`: The `DirectOptimal` algorithm, which interleaves single global passes of saturation (using effectively `AddDerivedWithPruning`) and reduction.

All the algorithms were implemented in `R` and `C`, and all experiments were run on `R` version 4.3.

Synthetic dataset generation To systematically assess performance, we used synthetically generated implicational systems, controlling for the number of attributes ($|M| \in \{4, \ldots, 20\}$), initial implications ($|\Sigma_{in}| \in \{5, \ldots, 20\}$), and lattice size ($|L| \in \{3 \ldots, 9\}$). For each parameter combination, multiple (50) random instances were generated. Random sets of attributes were chosen for the left-hand and right-hand sides, imposing that the resulting implication $A \Rightarrow B$ be reduced, that is, $B = B \setminus A$. This choice in generating test implicational systems is due to the fact that they represent the most general and unfavourable scenario for algorithms, since these implicational systems naturally contain redundancy at different levels. In the event that random formal contexts had been generated and the implication base calculated, the systems would not contain such levels of redundancy, and the algorithms would not have to work as hard as with the systems used in the experiment.

Performance metrics We captured two categories of metrics to build a comprehensive performance profile for each algorithm:

- Execution time: The primary metric, measuring the total wall-clock time in seconds.
- Operational cost: The total number of calls to the computationally expensive π-operator and to the `AddDerived` functions.

7.2 Results and discussion

The experimental results, summarized in Table 1 and visualized in Fig. 1, unequivocally demonstrate the overwhelming superiority of the interleaved `DO` algorithm in terms of practical performance.

Across all tested configurations, the data reveals a dramatic difference in execution time. The `DO` algorithm is consistently faster than both Two-Stage (TS) approaches by several orders of magnitude as the problem size increases. For instance, for instances with $|M| = 12$ attributes, `DO` is approximately **800 times faster** than `TS-Pruning` and nearly **1,200 times faster** than `TS-NoPruning`. While the pruning strategy in `TS-Pruning` aims to be an optimization, the data shows its overhead can sometimes lead to worse performance than the unoptimized version. Ultimately, the results confirm that the algorithmic strategy of interleaving saturation and reduction is profoundly more effective than a sequential approach, rendering the TS methods impractical for non-trivial problems. The column labelled as $|\Sigma_{\text{final}}|$

Table 1 Comparative performance metrics of Two-Stage (TS) and DirectOptimal (DO) algorithms, over some representative test configurations

Algorithm	$\|M\|$	Execution time (s)	π-calculations	`AddDerived` calls	$\|\Sigma_{\text{final}}\|$
TS-NoPruning	4	0.0420	37.80	1820.10	3.1
TS-Pruning	4	0.0170	3097.80	578.90	3.1
DO	4	0.0020	317.40	58.60	3.1
TS-NoPruning	5	0.1546	56.70	5148.60	2.7
TS-Pruning	5	0.0828	12,690.10	2253.90	2.7
DO	5	0.0021	273.50	49.40	2.7
TS-NoPruning	6	0.7877	97.40	20,625.40	2.9
TS-Pruning	6	1.0903	130,505.50	23,042.00	2.9
DO	6	0.0050	524.20	93.90	2.9
TS-NoPruning	7	1.4869	132.80	32,863.40	6.2
TS-Pruning	7	0.8897	96,719.10	18,194.20	6.2
DO	7	0.0126	1044.60	179.60	6.2
TS-NoPruning	8	5.9426	246.40	110,894.20	5.2
TS-Pruning	8	7.1173	626,591.60	107,529.20	5.2
DO	8	0.0149	1091.30	185.30	5.2
TS-NoPruning	9	19.6704	341.80	291,718.60	8.8
TS-Pruning	9	16.0393	1,228,074.00	224,381.10	8.8
DO	9	0.0303	1663.30	292.00	8.8
TS-NoPruning	10	69.4697	384.00	635,418.90	6.0
TS-Pruning	10	44.9555	2,998,076.20	542,356.50	6.0
DO	10	0.0178	1109.80	199.80	6.0
TS-NoPruning	11	45.9408	515.90	537,624.00	11.5
TS-Pruning	11	42.0920	2,579,744.60	471,738.60	11.5
DO	11	0.0993	4209.70	782.50	11.5
TS-NoPruning	12	186.3007	979.71	1,827,648.57	13
TS-Pruning	12	126.8008	7,337,209.83	1,297,409.83	13
DO	12	0.1570	5789.83	1027.17	13

presents the average cardinality of the resulting implicational system. Note that all algorithms arrived at the same final system.

The source of this dramatic performance gap is revealed by analysing the different computational bottlenecks for each algorithm. The data in Table 1 shows that the two TS methods fail for fundamentally different reasons, a fact that is strikingly visualized in the efficiency plot shown in Fig. 1.

The inefficiency of `TS-NoPruning` stems from the massive number of `AddDerived` calls. This is visible in Fig. 1 by the position of its cluster in the *top-left*: despite performing very few π-calculations, its execution time is extremely high. Its bottleneck is not the cost of pruning, but the sheer, brute-force computational effort of generating an enormous volume of candidate implications.

Conversely, the bottleneck for `TS-Pruning` is an astronomical number of π-calculations. Its cluster extends along the diagonal up to the *top-right* of the plot, showing a clear correlation between its millions of calls to the expensive π-operator and its poor per-

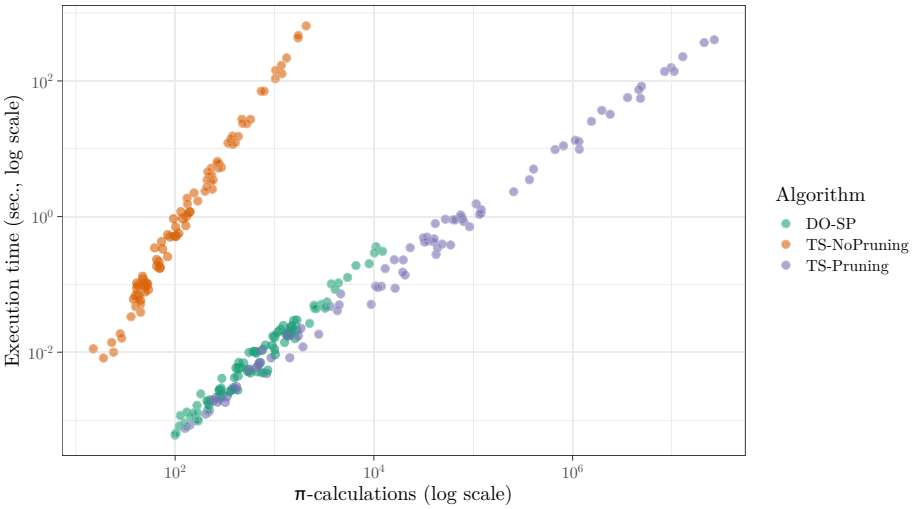

Fig. 1 Efficiency plot correlating execution time with the number of π-calculations for every experimental run. Note the logarithmic scales on both axes

formance. This demonstrates that its strategy of pruning every single candidate implication is computationally prohibitive.

The efficiency of `DO` lies in its balanced approach, with its results clustered in the *bottom-left* (the high-efficiency quadrant). It drastically reduces the number of `AddDerived` calls compared to `TS-NoPruning`, and it performs orders of magnitude fewer π-calculations than `TS-Pruning`. This is because its interleaved pruning acts on the system in passes, not on every candidate rule.

In conclusion, the `DO` algorithm's strategy is superior because it avoids the twin pitfalls of the Two-Stage methods: the brute-force generation cost of `TS-NoPruning` and the excessive pruning overhead of `TS-Pruning`. It is the only tested algorithm that effectively controls both the number of generated rules and the number of expensive pruning checks—a balance that proves to be the key to its efficiency.

Scalability analysis of the `DirectOptimal` algorithm

Having established the clear superiority of the `DO` algorithm, this section presents a focused analysis of its scalability. The Two-Stage approaches are not considered further due to their prohibitive computational cost. The objective is to characterize how the performance of `DO` degrades as the key dimensions of the problem increase. The results of this analysis are presented in Fig. 2.

Dependence on initial implications ***($|\Sigma_{in}|$)*** The analysis reveals a clear positive correlation between the number of initial implications and the execution time. As shown in the left panel of Fig. 2, the growth in this experimental range appears to be nearly linear. While the theoretical worst-case complexity of the saturation pass is quadratic ($O(|\Sigma_k|^2)$), the observed data does not exhibit a strong super-linear trend. This suggests that for the tested configurations, the interleaved reduction phase of the `DO` algorithm is highly effective at controlling the growth of the intermediate system size, $|\Sigma_k|$. By pruning redundancies in each pass, the algorithm prevents the combinatorial explosion that would lead to a more pronounced quadratic

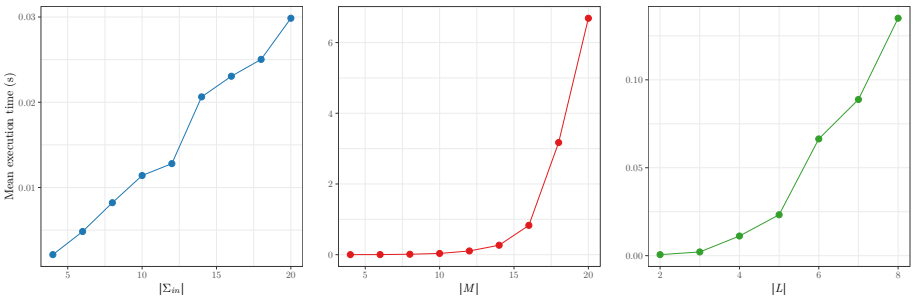

Fig. 2 Scalability of the DO algorithm. Each panel shows the growth in mean execution time (linear scale) as one parameter is varied while others are held constant

behaviour, resulting in a much more graceful and manageable scalability with respect to the initial system size.

Dependence on attributes ***(|M|)*** The number of attributes is confirmed to be the most critical factor for scalability. The middle panel of Fig. 2 shows a dramatic, explosive growth in execution time. This steep, upward-curving trend is the classic visual signature of *exponential growth*. It is crucial to note that this exponential behaviour does not arise from the cost of primitive operations (which scale linearly with $|M|$), but from the combinatorial explosion of the problem space itself. The number of distinct fuzzy sets is $|L|^{|M|}$, which causes the number of potential implications that the algorithm must consider and generate to grow exponentially. This exponential increase in the overall volume of work, rather than the cost of any single operation, is what drives the observed performance and confirms the profound impact of the problem's dimensionality.

Dependence on lattice size ***(|L|)*** The granularity of the truth-value lattice also has a significant impact on performance. The right panel of the figure demonstrates that the execution time grows faster than linearly as the lattice size increases. The data is consistent with the hypothesized quadratic trend, which empirically validates the impact of the $O(|L|^2)$ complexity of the `AddDerived*` functions, a fundamental component of the saturation process.

8 Conclusions and future work

In this paper, we have addressed the challenge of efficient closure computation in Fuzzy Formal Concept Analysis (FFCA) by extending the concept of direct implicational systems to the fuzzy setting. Our primary theoretical contribution is a complete characterization of the directness property via a novel Fuzzy Exchange Condition (FEC), which provided the foundation for defining redundancy and, ultimately, the notion of a *direct-optimal* system. Building upon this, we developed and proved the correctness of the `DirectOptimal` (DO) algorithm, an efficient, interleaved strategy for computing these bases that contrasts with simpler, naive approaches.

A comprehensive experimental evaluation has rigorously validated our proposals, unequivocally demonstrating the overwhelming superiority of the `DirectOptimal` algorithm over sequential Two-Stage (TS) methods. The results show that DO is faster by orders of magnitude. Crucially, our analysis diagnosed the distinct computational bottlenecks that

cause the naive methods to fail-either an explosion in rule generation or a prohibitive pruning overhead.

In conclusion, this paper delivers a complete theory-to-practice pipeline for direct-optimal bases in FFCA. We have provided the necessary theoretical characterizations, developed a provably correct and practically efficient algorithm, and empirically demonstrated its vast superiority over simpler alternatives. This work establishes direct-optimal systems as an achievable and computationally viable tool for knowledge representation and processing in fuzzy contexts.

This research opens several promising avenues for future work. An immediate and natural extension would be the investigation of incremental, worklist-based algorithms. Such methods promise even greater efficiency by propagating changes locally rather than performing global passes, and their comparison with `DO` would be highly valuable. Furthermore, a more extensive empirical study on a wider range of real-world and synthetic datasets, perhaps using different complete residuated lattices, would help to further solidify the practical understanding of these algorithms.

Looking towards broader horizons, the application of these fast closure computation methods in real-world domains, such as recommender systems or intelligent data analysis, is a crucial next step to demonstrate their practical impact. Another interesting direction would be to extend the theoretical framework to more general logical settings, such as interval-valued fuzzy sets or possibility theory. Finally, the challenge of making the resulting direct-optimal fuzzy bases interpretable and easily visualizable for domain experts remains a significant and important area for future research in human-centered knowledge discovery.

Acknowledgements The authors would like to thank Dr. Kira Adaricheva who inspired us to study this topic.

Author Contributions All authors contributed equally to this work.

Funding Funding for open access publishing: Universidad de Málaga/CBUA This research is partially supported by the State Agency of Research (AEI), the Spanish Ministry of Science, Innovation, and Universities (MCIU) and the European Social Fund (FEDER) through the research projects with reference PID2021-127870OB-I00 and (MCIU/AEI/FEDER, EU) and the VALID research project (PID2022-140630NB-I00 funded by MCIN/AEI/10.13039/501100011033). Funding for open access publishing: Universidad de Málaga/CBUA.

Data availability The data is available to everyone.

Materials availability Not applicable.

Code availability All the code for the algorithms will be made available within the `fcaR` package after acceptance.

Declarations

Conflict of interest Authors declare to have no competing interest.

Ethics approval and consent to participate All authors consent to the submission of this work to the journal.

Consent for publication All authors consent to the submission of this work to the journal.

References

Aragón RG, Medina J, Ramírez-Poussa E (2024) Factorizing formal contexts from closures of necessity operators. Comput Appl Math 43:1–24. https://doi.org/10.1007/s40314-024-02590-0

Aragón RG, Medina J, Ramírez-Poussa E (2025) Decomposition of contexts into independent subcontexts based on thresholds. Comput Appl Math 44:1–22. https://doi.org/10.1007/s40314-025-03302-y

Bělohlávek R (1998) Fuzzy concepts and conceptual structures: induced similarities. Joint conference on information sciences proceedings, vol 1. Assoc. Intel. Machinery, pp 179–182

Bělohlávek R (2002) Fuzzy relational systems. Springer. https://doi.org/10.1007/978-1-4615-0633-1

Bělohlávek R, Vychodil V (2005) Fuzzy attribute implications: computing non-redundant bases using maximal independent sets. Australasian joint conference on artificial intelligence. LNAI 3809, Springer-Verlag Berlin Heidelberg, pp 1126–1129. https://doi.org/10.1007/11589990153

Bělohlávek R, Vychodil V (2006) Attribute implications in a fuzzy setting. In: Formal concept analysis: 4th international conference, ICFCA 2006, Dresden, Germany, February 13–17. Proceedings, pp 45–60 (2006). Springer. https://doi.org/10.1007/116714043

Bělohlávek R, Vychodil V (2017) Attribute dependencies for data with grades II. Int J Gen Syst 46(1):66–92. https://doi.org/10.1080/03081079.2016.1205712

Bělohlávek R, Cordero P, Enciso M, Mora A, Vychodil V (2016) Automated prover for attribute dependencies in data with grades. Int J Approx Reason 70:51–67. https://doi.org/10.1016/j.ijar.2015.12.007

Bertet K, Monjardet B (2010) The multiple facets of the canonical direct unit implicational basis. Theor Comput Sci 411(22–24):2155–2166. https://doi.org/10.1016/J.TCS.2009.12.021

Bertet K, Nebut M (2004) Efficient algorithms on the family associated to an implicational system. Discret Math Theor Comput Sci 6:315–338. https://doi.org/10.46298/dmtcs.330

Burusco A, Fuentes-González R (1994) The study of the L-fuzzy concept lattice. Mathw Soft Comput 3:209–218

Čech E (1966) Topological spaces. Interscience Publishers,(Revised edition prepared by Z. Frolík and M. Katětov, based on the original manuscripts)

Cordero P, Enciso M, Mora A (2018) Directness in fuzzy formal concept analysis. International conference on information processing and management of uncertainty in knowledge-based systems. Springer, pp 585–595. https://doi.org/10.1007/978-3-319-91473-250

Cordero P, Enciso M, Mora A, Vychodil V (2020) Parameterized simplification logic I: reasoning with implications and classes of closure operators. Int J Gen Syst 49(7):724–746. https://doi.org/10.1080/03081079.2020.1831484

Cornejo ME, Medina J, Ocaña FJ (2026) Theories, models and bases of attribute implications in multi-adjoint concept lattices with hedges. Comput Appl Math 45(30):1–33. https://doi.org/10.1007/s40314-025-03391-9

Ganter B, Obiedkov S (2016) Conceptual exploration. Springer Berlin. https://doi.org/10.1007/978-3-662-49291-8

Guigues JL, Duquenne V (1986) Familles minimales d'implications informatives résultant d'une tables de données binaires. Mathématiques et Sciences Humaines 95:5–18

Hájek P (2013) Metamathematics of fuzzy logic. Trends in logic. Springer

Medina J, Ojeda-Aciego M, Ruiz-Calviño J (2009) Formal concept analysis via multi-adjoint concept lattices. Fuzzy Sets Syst 160(2):130–144. https://doi.org/10.1016/j.fss.2008.05.004

Mora A, Enciso M, Cordero P, Fortes I (2012) Closure via functional dependence simplification. Int J Comput Math 89:510–526. https://doi.org/10.1080/00207160.2011.644275

Ojeda-Hernández M, López-Rodríguez D (2025) On direct systems of implications with graded attributes. The 14th conference of the European Society for Fuzzy Logic and Technology (EUSFLAT). https://doi.org/10.1007/978-3-031-97228-713

Ojeda-Hernández M, Cabrera IP, Cordero P (2022) Quasi-closed elements in fuzzy posets. J Comput Appl Math 404:113390. https://doi.org/10.1016/j.cam.2021.113390

Ojeda-Hernández M, Cabrera IP, Cordero P, Muñoz-Velasco E (2023) On pseudointents in fuzzy formal concept analysis. International conference on conceptual structures. Springer, pp 36–40. https://doi.org/10.1007/978-3-031-40960-84

Pollandt S (1997) Fuzzy Begriffe: Formale Begriffsanalyse Von Unscharfen Daten. Springer, Berlin

Rodríguez-Lorenzo E, Bertet K, Cordero P, Enciso M, Mora A (2018) Direct-optimal basis computation by means of the fusion of simplification rules. Discret Appl Math 249:106–119. https://doi.org/10.1016/j.dam.2017.12.031

Vychodil V, Bělohlávek R (2005) Fuzzy attribute logic: attribute implications, their validity, entailment, and non-redundant basis. In: Proceedings of the eleventh international fuzzy systems association world congress, vol I

Wille R (1982) Restructuring lattice theory: an approach based on hierarchies of concepts. Springer, pp 445–470. https://doi.org/10.1007/978-94-009-7798-315