Knowledge-Based Systems 318 (2025) 113475

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Systems of implications obtained using the Carve decomposition of a formal context

Domingo López-Rodríguez [a], Manuel Ojeda-Hernández [b],*, Tim Pattison [c]

[a] Departamento de Matemática Aplicada, Universidad de Málaga, Andalucía Tech, Málaga, 29071, Spain
[b] Department of Computing Science, Umeå University, Sweden
[c] Defence Science and Technology Group, West Avenue, Edinburgh, SA 5111, Australia

ARTICLE INFO

MSC:
06-04
06A15
03B70

Keywords:
Formal concept analysis
Implicational systems
Attribute implications
Divide-and-conquer algorithms
Knowledge representation

ABSTRACT

The Carve algorithm uses a divide-and-conquer strategy to compute the concept lattice of a formal context. The decomposition phase of the Carve algorithm discovers hierarchical structure in an amenable formal context, which the synthesis phase then exploits to construct the concept lattice from those of the component sub-contexts. In this paper, the problem of computing a sound and complete set of attribute implications via a refinement of the Carve decomposition is studied. Indeed, a set of rules is devised to obtain a set of valid implications which is proved to be complete. The refined decomposition and these rules are implemented in the novel Carve+ algorithm, whose runtime compares favorably with direct computation of the Duquenne–Guigues base of implications via the NextClosure algorithm.

1. Introduction

Formal Concept Analysis (FCA) was introduced by Rudolf Wille in 1982 in his seminal paper [1] to represent and manage the knowledge stored in a relational database — the so-called formal context. The theory and applications of FCA have since advanced significantly. FCA has been applied in such diverse fields as information retrieval [2–5], renewable energy [6], student assessment [7,8], conflict analysis [9–11], malware analysis [12] and sentiment analysis [13]. Recent theoretical developments have largely focused on extending FCA to more versatile frameworks, such as Three-level [14,15], Fuzzy [16], Multi-adjoint [17] and Heterogeneous Formal Concept Analysis [18]. Due to the exponential cost of computing the implicit knowledge in a formal context [19], devising faster and more efficient methods for its computation has also received considerable attention [20–25].

Formal Concept Analysis (FCA) converts a formal context bigraph, represented by its bi-adjacency matrix, into either a concept lattice, represented by a digraph, or a set of attribute implications. The Carve algorithm [24] recursively decomposes the context bigraph into indivisible sub-contexts, invokes *any* conventional FCA algorithm – or combination of algorithms – which constructs the corresponding lattice digraphs, and assembles the overall lattice digraph from these components while returning from the recursion. Carve therefore constitutes a “wrapper” which divides, conquers and parallelizes conventional Formal Concept Analysis of amenable formal contexts. It consists of a preprocessing *analysis* phase which decomposes the context and a post-processing *synthesis* phase which assembles the lattice digraph. In this paper, we characterize the attribute implications derived from each Carve-indivisible sub-context and modify the synthesis phase of the Carve algorithm to produce a sound and complete set of attribute implications for the overall context.

The Carve recursion tree supports not only coordinated visualization of the context bigraph and lattice digraph [26], but also inclusion tree layouts of both graphs [26] and of the context matrix [27]. A formal context whose Carve recursion tree consists of more than a single node is said to be *amenable* to Carve decomposition. However, the more leaf-node sub-contexts it has, and the more similar their sizes, the more successfully will Carve divide, conquer and parallelize the computation. Carve-amenable empirical contexts have been identified in applications as disparate as software re-engineering [28], co-authorship analysis [26] and clique membership in social networks [29].

Each decomposition step removes fully-connected or “universal” objects and attributes, along with any attributes or objects isolated or “orphaned” by their removal, and partitions the resultant sub-context bigraph into its connected components. The identification of these universals and orphans is simplified by both Jointly Reverse Lectic (JRL) ordering of the context bi-adjacency matrix and by its subsequent clarification and reduction [27].

* Corresponding author.
E-mail addresses: dominlopez@uma.es (D. López-Rodríguez), manueloh@cs.umu.se (M. Ojeda-Hernández), tim.pattison@ieee.org (T. Pattison).

https://doi.org/10.1016/j.knosys.2025.113475
Received 20 December 2024; Received in revised form 13 February 2025; Accepted 31 March 2025
Available online 19 April 2025

In this paper we show that a sound and complete set of implications for the global context can be synthesized from these component bases using adaptations of the analysis and synthesis phases of the Carve algorithm which we call AugmentedCarve and CarveImp respectively. These differ from Carve in that: AugmentedCarve produces a finer-grained decomposition of a formal context; and CarveImp constructs a set of implications – vice lattice digraph – for the formal context from those for the Carve-indivisible sub-contexts.

Importantly, the set of implications obtained for a given AugmentedCarve sub-context collates and supplements those for all of its descendant sub-contexts in the AugmentedCarve recursion tree. In the case of AugmentedCarve-indivisible formal contexts, conventional FCA algorithms such as NextClosure [20] or LinCbO [30] can be used to compute an implicational base with the same computational complexity.

This paper is organized as follows. In Section 2 we briefly summarize the derivation of implication systems from formal contexts, and review prior work on dividing, conquering and parallelizing their computation. In Section 3, we introduce the AugmentedCarve decomposition of an amenable formal context, and compare and contrast it with the Carve decomposition. In Section 4 we relate the concept-forming operators of an AugmentedCarve sub-context to those of its children in the AugmentedCarve recursion tree, and thereby relate those of the root and leaf-node sub-contexts. We present a set of rules through which a system of implications for a formal context can be constructed from those of its leaf-node sub-contexts, prove their soundness and completeness, and illustrate them using a running example. We then present the CarveImp algorithm for constructing this system of implications, and trace its execution when applied to the running example. In Section 5 we compare the performance of the Carve+ algorithm, which combines AugmentedCarve and CarveImp, with that of NextClosure on Carve-amenable empirical datasets. Finally, in Section 6, we present the conclusions to this work and propose future research directions in this line.

2. Preliminaries

In this section, we provide a brief introduction to Formal Concept Analysis (Section 2.1), summarize the derivation of implication systems from formal contexts (Section 2.2), and review prior work on dividing, conquering and parallelizing their computation (Section 2.3).

2.1. Formal concept analysis

Formal Concept Analysis (FCA) was introduced by Rudolf Wille in the 1980's [1] to extract knowledge from relational data tables — the so-called formal contexts. Mathematically, a formal context is a triple (G, M, I) (usually abbreviated as $\mathbb{K}$) where G is a set of objects, M is a set of attributes and $I \subseteq G \times M$ is the incidence relation between objects and attributes. Thus, gIm represents that object g has the attribute m, and $g \not I m$ that it does not. There are two distinguished operators in FCA that relate sets of objects and sets of attributes; these are the so-called derivation or concept-forming operators. For $\mathcal{A} \subseteq G$ and $\mathcal{B} \subseteq M$:

$$\mathcal{A}^{\uparrow} := \{m \in M \mid gIm, \text{ for all } g \in \mathcal{A}\}$$

$$\mathcal{B}^{\downarrow} := \{g \in G \mid gIm, \text{ for all } m \in \mathcal{B}\}$$

The pair $({}^{\uparrow}, {}^{\downarrow})$ is a Galois connection, which provides a rich mathematical structure to these operators. A pair $(\mathcal{A}, \mathcal{B}) \in 2^G \times 2^M$ is said to be a formal concept if $\mathcal{A}^{\uparrow} = \mathcal{B}$ and $\mathcal{B}^{\downarrow} = \mathcal{A}$. The set $\mathfrak{B}(\mathbb{K})$ of all formal concepts of the formal context $\mathbb{K}$ is endowed with an order relation given by $(\mathcal{A}_1, \mathcal{B}_1) \leq (\mathcal{A}_2, \mathcal{B}_2)$ if and only if $\mathcal{A}_1 \subseteq \mathcal{A}_2$, or equivalently, $\mathcal{B}_2 \subseteq \mathcal{B}_1$. This order relation gives $\mathfrak{B}(\mathbb{K})$ the structure of a complete lattice, the so-called concept lattice. There are two main knowledge structures in FCA, namely the concept lattice and the base of attribute implications. In this paper we focus on attribute implications, which are explained in the next section.

2.2. Implication systems in formal concept analysis

An attribute implication is an expression of the form $\mathcal{L} \to \mathcal{R}$, where $\mathcal{L}, \mathcal{R} \subseteq M$ are its *antecedent* or left-hand side (LHS) and its *consequent* or right-hand side (RHS) respectively. Throughout this work, the term *implication* will be used as short for *attribute implication*, for the sake of readability. We say that $\mathcal{L} \to \mathcal{R}$ holds in formal context $\mathbb{K}$ (in other words, $\mathcal{L} \to \mathcal{R}$ is valid in $\mathbb{K}$), or that $\mathbb{K}$ is a *model* for $\mathcal{L} \to \mathcal{R}$, whenever $\mathcal{L}^{\downarrow} \subseteq \mathcal{R}^{\downarrow}$ (equivalently, if $\mathcal{R} \subseteq \mathcal{L}^{\downarrow\uparrow}$).

An implication is said to be either *full* if $\mathcal{L} \subsetneq \mathcal{L}^{\downarrow\uparrow} = \mathcal{R}$ or *abbreviated* if $\mathcal{R} = \mathcal{L}^{\downarrow\uparrow} \setminus \mathcal{L}$.

New implications can be derived from existing ones using various combinations of Armstrong's axioms of inference [31]. For $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \subseteq M$, these are:

$$\begin{array}{rrcl} \texttt{Reflexivity:} & \mathcal{Y} \subseteq \mathcal{X} & \Longrightarrow & \mathcal{X} \to \mathcal{Y} \\ \texttt{Transitivity:} & \mathcal{X} \to \mathcal{Y} \text{ and } \mathcal{Y} \to \mathcal{Z} & \Longrightarrow & \mathcal{X} \to \mathcal{Z} \\ \texttt{Augmentation:} & \mathcal{X} \to \mathcal{Y} & \Longrightarrow & \mathcal{X}\mathcal{Z} \to \mathcal{Y}\mathcal{Z} \end{array}$$

Other helpful rules or axioms can be deduced by combining the basic ones. For instance, combining Transitivity and Augmentation yields

$$\texttt{Union:} \quad \mathcal{X} \to \mathcal{Y} \text{ and } \mathcal{X} \to \mathcal{Z} \quad \Longrightarrow \quad \mathcal{X} \to \mathcal{Y}\mathcal{Z}$$

which will prove useful in Section 4.3.

A set or system Σ of attribute implications for a formal context $\mathbb{K}$ is: *sound* if each implication holds in $\mathbb{K}$; *complete* if all other valid implications over $\mathbb{K}$ can be derived from it by means of Armstrong's axioms; *minimal* if it is complete but no proper subset is; and an *implicational base* if it is sound, complete and minimal. The two best-known implicational bases are the Duquenne–Guigues, stem or canonical base [32] and the proper premise base [33].

While efficient algorithms exist for the generation of implicational bases [20,34], the Carve algorithm suggests the possibility of dividing, conquering and parallelizing this computation for amenable formal contexts. An implicational base can be computed for each Carve-indivisible sub-context, and these combined and supplemented using a set of rules which derive the implications of a parent context from those of its children. These rules can be applied incrementally while returning from the recursive procedure whereby the Carve decomposition is computed. Furthermore, since the sets of implications for each child of a parent sub-context relate discrete subsets of attributes, their computation can be parallelized.

2.3. Prior work on the incremental computation of systems of implications

The majority of literature on divide-and-conquer techniques for Formal Concept Analysis (FCA) concentrates on the factorization or decomposition of the concept lattice, or the reconstruction of the lattice from other lattices containing complementary parts of the knowledge represented by the dataset. Consistent with the objective of this paper, however, this section will concentrate on a review of previous work proposing divide-and-conquer strategies for the construction of implication systems in formal contexts.

In [35] the authors propose the construction of the canonical base of implications from the mixture of the bases of two contexts (called *factors*) whose apposition is precisely the original context. In this sense, the union of the bases of the apposed contexts is not sufficient, and they need the definition of hybrid implications that relate the attributes of both factors. However, the construction of these hybrid implications requires the construction of the concept lattice of each of the factor contexts, so it can be a comparatively laborious task. The union of the three implication systems (that of each factor and the hybrid) is sound and complete, although it does not necessarily coincide with the canonical base, which can be reconstructed from this aggregated system using techniques such as the MinimalCover [19].

Computation of the canonical base of implications for a formal context $\mathbb{K}$ involves generating all pseudo-intents and their corresponding closures. Kriegel and Borchmann [36] showed that the canonical base can be computed in increasing order of antecedent cardinality using their NextClosures algorithm, and that all intents and pseudo-intents of a specified cardinality can be computed in parallel. Importantly, the correctness of this approach does not depend on the context $\mathbb{K}$ having specific properties, such as amenability to Carve decomposition.

More recently, the possibility of extracting knowledge from concept lattices using congruence relations has been the subject of study [37]. The lattice associated with a formal context can be reduced to a sublattice using congruences, which condenses the information contained in the whole lattice while retaining the most significant parts. This work demonstrates a strong relationship between the implication systems related to each lattice, the original and the simplified. However, the latter is unable to recover all the knowledge and may omit implications necessary for the completeness of the system.

The last strategy that can be employed for the determination of implication systems is the so-called *Knowledge Cores* [38], which correspond to salient sub-contexts in graph theory and are analogous to bipartite cores. From these cores, different types of implication bases can be computed. However, while this construction is efficient and reveals a large amount of implicit information in the formal context, it is not exhaustive and therefore cannot recover a sound and complete system in all cases.

Rather than competing with these methods, our proposal losslessly decomposes the context into smaller sub-contexts, to which any technique for implication mining can then be applied. It then aggregates and supplements the resultant implication systems – provided they are individually sound and complete[1] – into a sound and complete set.

3. The Carve and AugmentedCarve decompositions

In this section, an extension of the Carve decomposition suitable for computing a complete set of implications is presented. We first introduce the Carve decomposition [24] and illustrate its application to an amenable formal context. We then present the AugmentedCarve decomposition, apply it to the same formal context, and thereby compare and contrast it with Carve. And finally, we motivate the exploitation of the AugmentedCarve recursion tree for construction of a sound and complete set of implications.

3.1. *The* Carve *decomposition*

In this subsection, the main aspects of the ideas followed and procedures performed in the Carve decomposition are summarized. The complete exposition of the methodology and theoretical results of the Carve algorithm can be found in [24].

Let $\mathbb{K} = (G, M, I)$ be a formal context and let $\mathbb{K}_i = (G_i, M_i, I_i)$, $i \in \mathcal{I}$ be the collection of sub-contexts obtained in the Carve decomposition of $\mathbb{K}$. Each sub-context $\mathbb{K}_i$ can be viewed as a bipartite graph with vertex sets G_i and M_i, bi-adjacency matrix I_i, fully-connected or "universal" elements

$$\mathrm{univ}(M_i) := \{\mu \in M_i \mid g I_i \mu \text{ for all } g \in G_i\}$$
$$\mathrm{univ}(G_i) := \{\gamma \in G_i \mid \gamma I_i m \text{ for all } m \in M_i\}$$

and "orphan" elements

$$\mathrm{orph}(M_i) := \{\mu \in M_i \mid g \not{I}_i \mu \text{ for all } g \in G_i \setminus \mathrm{univ}(G_i)\}$$
$$\mathrm{orph}(G_i) := \{\gamma \in G_i \mid \gamma \not{I}_i m \text{ for all } m \in M_i \setminus \mathrm{univ}(M_i)\}$$

which are isolated by removal of the universal elements.

[1] Except where $\varnothing$ is the antecedent.

Table 1
Example formal context with Carve recursion tree superimposed using an inclusion layout.

	A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*	*M*
1	×	×	×	×									
2	×	×	×										
3	×	×			×	×							
4	×	×			×								
5	×	×				×	×						
6	×	×											
7	×							×	×	×	×		
8	×							×	×				
9	×							×					
10	×									×			
11	×											×	×
12	×											×	

Carve identifies and removes the universals and orphans of a sub-context bigraph $\mathbb{K}_i$, and finds the connected components of what remains. If bigraph $\mathbb{K}_i$ is disconnected by the removal of its universal elements, then Carve is applied recursively to any remaining connected components which are not orphans. The sub-contexts $\{\mathbb{K}_i\}_{i\in\mathcal{I}}$ are therefore related by the resultant Carve recursion tree, whose root node represents $\mathbb{K}_0 := \mathbb{K}$ and whose leaf nodes represent sub-contexts which Carve cannot further decompose. We define the following operators on the vertices of this tree:

$$\textbf{Children:} \quad \mathcal{C} : \mathcal{I} \to 2^{\mathcal{I}}, \qquad \mathcal{C}(i) := \{j \in \mathcal{I} \mid \mathbb{K}_j \text{ is a child of } \mathbb{K}_i\}$$
$$\textbf{Parent:} \quad \mathcal{P} : \mathcal{I} \setminus \{0\} \to \mathcal{I}, \qquad \mathcal{P}(i) = j \text{ if, and only if, } i \in \mathcal{C}(j)$$

This naming scheme corresponds to the well-known convention in Computer Science and in Graph Theory of referring to the lower neighbors (with respect to the tree structure) of a node n as its *children*, thus having n the role of *parent*. For instance, in Fig. 1, the children of the node labeled as $\{\{7, 8, 9, 10\}, \{H, I, J, K\}\}$ are $\{\{8, 9\}, \{H, I\}\}$ and $\{\{10\}, \{J\}\}$. Numbering the nodes from top to bottom and left to right, $\{\{7, 8, 9, 10\}, \{H, I, J, K\}\}$ corresponds to the index 2 (the root is numbered 0), hence $\mathcal{C}(2) = \{6, 7\}$ and $\mathcal{P}(6) = \mathcal{P}(7) = 2$, since 6 and 7 are the indices of the aforementioned children.

Note that the root node, which is numbered 0, has no parent, and the term *child* intentionally excludes more distant tree descendants. If $\mathbb{K}_i$ is a descendant of $\mathbb{K}_k$ in this tree, then $G_i \subseteq G_k$, $M_i \subseteq M_k$, and $I_i := I_k \cap (G_i \times M_i)$. For each sub-context $\mathbb{K}_i$, $i \in \mathcal{I}$, its concept-forming operators will be denoted by $\left({}^{\uparrow_i}, {}^{\downarrow_i}\right)$, and defined as usual:

$$A^{\uparrow_i} := \{m \in M_i \mid g I_i m \text{ for all } g \in A\} \quad \text{for } A \subseteq G_i$$
$$B^{\downarrow_i} := \{g \in G_i \mid g I_i m \text{ for all } m \in B\} \quad \text{for } B \subseteq M_i$$

Example 1. An example Carve-amenable formal context is shown in Table 1 with the Carve recursion tree superimposed using an inclusion layout.

The Carve recursion tree is also shown in Fig. 1. Each node lists the objects and attributes of the corresponding sub-context, and universal elements are shown bold.

The lattice digraph for this formal context is shown in Fig. 2. Each concept is shaded according to the depth in the Carve recursion tree at which it is discovered.

3.2. *Extending* Carve*: The* AugmentedCarve *decomposition*

Whereas the Carve decomposition is well suited to dividing and conquering the construction of the lattice digraph, a finer decomposition of the formal context is required for the purposes of systematically and efficiently deriving an implicational system. In the Carve recursion tree, neither universal nor orphan elements persist in descendants of the sub-context in which they are identified. Instead, we consider the

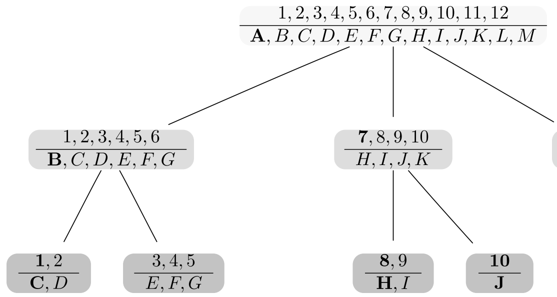

Fig. 1. Carve tree for the example context in Table 1.

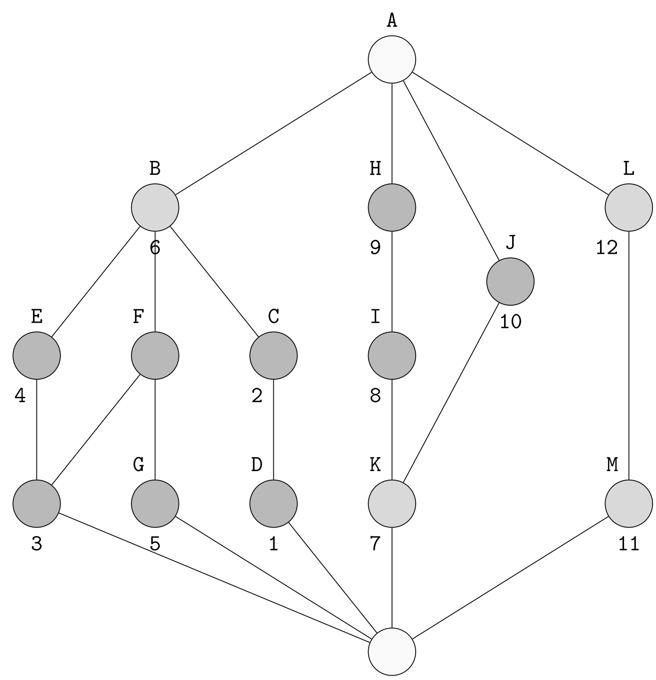

Fig. 2. Lattice digraph for the formal context in Table 1.

sub-context consisting of an orphan and its equivalents, along with the empty set of its non-universal neighbors in the sub-context bigraph, to be a leaf-node child of the sub-context in which it was identified. The orphan is itself universal within this leaf-node sub-context, which also contains no orphans. Furthermore, whereas a sub-context which has universals and a single connected component after removal of the universals – and hence no orphans – corresponds to a leaf node in the Carve decomposition, we here divide it into a parent sub-context, which includes the universals, and a child sub-context which does not. We henceforth refer to the Carve recursion tree augmented with these additional sub-contexts as the *augmented* Carve tree.

Example 2. The augmented Carve tree for the example context in Table 1 is illustrated in Fig. 3.

Only the shaded nodes have counterparts in the Carve recursion tree, with the darkness of the shading indicating tree depth.

The augmented Carve tree is computed recursively using the AugmentedCarve algorithm listed in Algorithm 1 and its helper function in Algorithm 2. Like the Carve decomposition, AugmentedCarve hierarchically partitions a formal context into smaller sub-contexts. Unlike Carve, however, it is also invoked recursively on sub-contexts which remain connected after the removal of universals, and on orphan sub-contexts. Thus a leaf node sub-context in the augmented Carve tree is one which either consists entirely of universals or has none.[2] For example, the AugmentedCarve algorithm further decomposes the leaf node $(\{\mathbf{1}, 2\}, \{\mathbf{C}, D\})$ of the Carve recursion tree in Fig. 1 by removing the universals 1 and C to yield new leaf-nodes, $(\mathbf{2}, \varnothing)$ and $(\varnothing, \mathbf{D})$, as shown in Fig. 3. Furthermore, if the Carve leaf-node sub-context $(\{3, 4, 5\}, \{E, F, G\})$ had included an additional, universal element, then it would have had an intervening parent in the AugmentedCarve – but not Carve – decomposition which included that universal element.

The following information is encapsulated within an AugmentedCarve node: (1) the sets of objects and attributes that define a sub-context of the parent node's context; (2) any universal objects and attributes associated with the node; (3) a list of its child nodes determined by the partitioning scheme of AugmentedCarve; (4) a reference to the parent node. Additionally, each node stores an implicational system Σ initialized to $\varnothing$ in anticipation of its subsequent population during the synthesis phase.

Thus, details about the objects and attributes, as well as their universal elements, and the implications Σ are directly embedded within the node's structure, represented as a triple $(\mathbb{K}, \mathbb{U}, \Sigma)$. Here, $\mathbb{K} = (G, M, I)$ denotes the formal context, and $\mathbb{U} = (\mathrm{univ}(G), \mathrm{univ}(M))$ captures the sets of universal objects and attributes. From a practical standpoint, the nodes are organized in an indexed list $\mathcal{N}$. The parent–child relationships are maintained using the $\mathcal{C}$ and $\mathcal{P}$ operators previously defined.

The main procedure (AugmentedCarve (G, M, I)) in Algorithm 1 initializes the global variables: the list $\mathcal{N}$, the index n of the last element of $\mathcal{N}$, and both operators $\mathcal{C}$ and $\mathcal{P}$. Then, it invokes the recursive procedure AugmentedCarve_helper (Algorithm 2), which grows the augmented tree from the root node, corresponding to the original formal context. AugmentedCarve_helper adds nodes by eliminating universal elements and partitioning the corresponding sub-context into connected components. The latter task is performed by the auxiliary function ConnectedComponents, which identifies connected components of the bipartite graph, and outputs them as formal contexts, to facilitate the recursive calls in lines 11–12.

Algorithm 1: AugmentedCarve (G, M, I)

Input: A formal context (G, M, I)
Output: The AugmentedCarve tree encoded as: the global list of nodes $\mathcal{N}$; the *children* operator, $\mathcal{C}$; and the *parent* operator, $\mathcal{P}$.

```
n := NULL      // Global var.: index of the last node added
N, C, P := []                                // empty lists
AugmentedCarve_helper((G, M, I), i = NULL)
return (N, C, P)
```

AugmentedCarve_helper has two criteria to finish recursion and, therefore, mark a node as a leaf in the tree:

- In line 7, a node with either no objects or no attributes (representing an isolated vertex in the bipartite graph, i.e. an orphan) becomes a leaf.
- Line 10 specifies that if the sole connected component matches the original context (G, M, I), the node is deemed an indivisible leaf.

[2] The one qualification to this characterization is that if the root node of the augmented Carve tree lacks universals, it is only a leaf node if it is also connected.

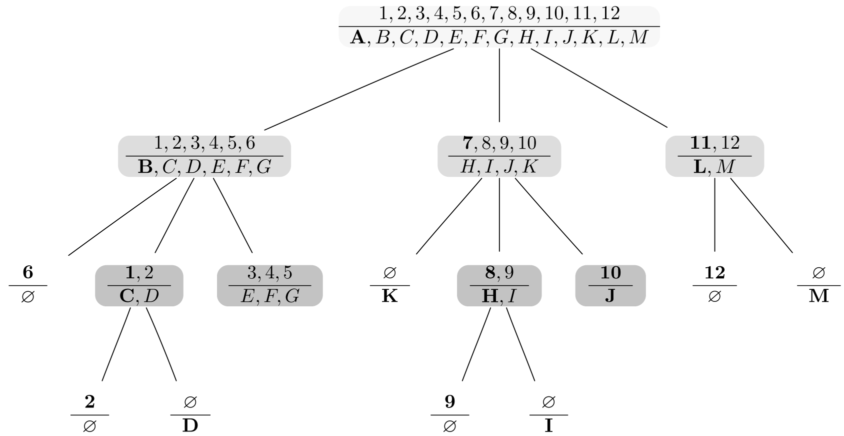

Fig. 3. Augmented Carve tree for the example context in Table 1.

Algorithm 2: AugmentedCarve_helper($(G, M, I), i$)

Input: A formal context (G, M, I); i, the index of the parent node (it may be NULL if this is the root node).

Output: Updates the global list of nodes $\mathcal{N}$ and the operators $\mathcal{C}$ and $\mathcal{P}$. As a side-effect, the construction of the AugmentedCarve tree.

/* Assign an index k to the new node */

if $n =$ NULL **then** // Root node

  $k := 0$

else

  $k := n + 1$ // Insert it after the last indexed node

end

/* Update list of nodes and operators $\mathcal{C}$ and $\mathcal{P}$ */

$\mathcal{N}(k) := \left(\mathbb{K} = (G, M, I),\ \mathbb{U} = (\text{univ}(G), \text{univ}(M)),\ \Sigma = \varnothing\right)$

$n := k$

if $i \neq$ NULL **then** $\mathcal{C}(i) := \mathcal{C}(i) \cup \{k\}$

$\mathcal{P}(k) := i$

// Exit if this is an orphan node

if $G = \varnothing$ ***or*** $M = \varnothing$ **then exit**

/* Remove universal elements and compute connected components (children) of the bipartite graph */

$G' := G \setminus \text{univ}(G);\ M' := M \setminus \text{univ}(M);\ I' := I \cap (G' \times M')$

$\{(G_j, M_j, I_j)\}_{j \in J} :=$ ConnectedComponents(G', M', I')

// Stop if the context is indivisible

if $(G_j, M_j, I_j) = (G, M, I)$ *for some* $j \in J$ **then exit**

/* Recurse on its children */

for $j \in J$ **do**

  AugmentedCarve_helper($(G_j, M_j, I_j), k$)

end

A key distinction from the original Carve procedure lies in AugmentedCarve's behavior after removing universal elements. Even when only one connected component remains, AugmentedCarve generates a new child node if at least one universal element has been removed. This modification, as will become evident in subsequent sections, enables a more granular decomposition that simplifies the computation of implicational systems. Consequently, some leaves in the Carve tree are internal nodes in the AugmentedCarve tree, facilitating the establishment of straightforward rules for constructing implicational systems from the augmented tree. The following scenarios describe the possible forms of leaf nodes:

- If either G or M is empty, the node takes the form $(\mathbb{K} = (\varnothing, M, \varnothing), \mathbb{U} = (\varnothing, M), \Sigma)$ or $(\mathbb{K} = (G, \varnothing, \varnothing), \mathbb{U} = (G, \varnothing), \Sigma)$. In this case, the remaining objects or attributes are treated as universal, corresponding to the existence of *orphans* in the bipartite graph.
- If both G and M consist solely of universal elements, the node is represented as $(\mathbb{K} = (G, M, G \times M), \mathbb{U} = (G, M), \Sigma)$.
- When G and M are non-empty and devoid of universal elements, the node is a proper indivisible leaf with the structure $(\mathbb{K} = (G, M, I), \mathbb{U} = (\varnothing, \varnothing), \Sigma)$.

Example 3. The canonical base of abbreviated implications for the formal context in Table 1 is:

{}	→	{A}
{A, M}	→	{L}
{A, K}	→	{H, I, J}
{A, J, L}	→	{B, C, D, E, F, G, H, I, K, M}
{A, I}	→	{H}
{A, H, L}	→	{B, C, D, E, F, G, I, J, K, M}
{A, H, J}	→	{I, K}
{A, G}	→	{B, F}
{A, F}	→	{B}
{A, E}	→	{B}
{A, D}	→	{B, C}
{A, C}	→	{B}
{A, B, L}	→	{C, D, E, F, G, H, I, J, K, M}
{A, B, J}	→	{C, D, E, F, G, H, I, K, L, M}
{A, B, H}	→	{C, D, E, F, G, I, J, K, L, M}
{A, B, E, F, G}	→	{C, D, H, I, J, K, L, M}
{A, B, C, F}	→	{D, E, G, H, I, J, K, L, M}
{A, B, C, E}	→	{D, F, G, H, I, J, K, L, M}

With reference to the augmented Carve tree in Fig. 3, an attribute clearly implies the universal attributes from each of the sub-contexts to which it belongs, so that for example D → C → B → A. Also, if a sub-context has at least one universal object, then any pair of attributes drawn from its distinct child sub-contexts implies all attributes for that sub-context. For example, attributes I and J belong to peer sub-contexts of the sub-context to which object 7 is universal, so that {I, J} → {H, K} is a valid implication. To see why this is the case, note that the intersection of their attribute extents is the set of objects universal to that sub-context, to which all remaining attributes are adjacent. Similarly, any orphan attribute implies the remaining attributes in its parent sub-context, so that for example {K} → {H, I, J}.

4. Construction of implication systems with AugmentedCarve

In this section, the rules that allow us to obtain a sound and complete set of attribute implications are presented. Each of these rules will be motivated, presented and proved to be sound separately. Later in this section it will be proved that applying all these rules ensures that the implicational system that we obtain is indeed sound and complete.

4.1. Galois connection in relation to AugmentedCarve

Let us explore how the AugmentedCarve decomposition helps rewrite the concept-forming operators in simpler terms, allowing us to navigate the AugmentedCarve recursion tree for their computation.

Lemma 1. *Let $\mathbb{K}_k = (G_k, M_k, I_k)$ with $k \in \mathcal{I}$ be a sub-context resulting from the application of AugmentedCarve to context $\mathbb{K}$, and let $\mathrm{univ}(G_k)$ and $\mathrm{univ}(M_k)$ be the sets of objects and attributes, respectively, which are "universal" within $\mathbb{K}_k$. Then, the derivation operators of $\mathbb{K}_k$ can be expressed in terms of the derivation operators of its AugmentedCarve children $\{\mathbb{K}_i = (G_i, M_i, I_i)\}_{i\in C(k)}$ as follows. For any $O \subseteq G_k$ and $A \subseteq M_k$,*

$$O^{\uparrow_k} = \begin{cases} M_k, & \text{if } O \subseteq \mathrm{univ}(G_k) \\ (O\cap G_i)^{\downarrow_i} \cup \mathrm{univ}(M_k), & \text{if } O\cap G_i \neq \varnothing \text{ for a single } i \in C(k) \\ \mathrm{univ}(M_k), & \text{if there exist } i,j \in C(k),\ i\neq j, \text{ such that } (O\cap G_i)\neq\varnothing \text{ and } (O\cap G_j)\neq\varnothing \end{cases}$$

$$A^{\downarrow_k} = \begin{cases} G_k, & \text{if } A \subseteq \mathrm{univ}(M_k) \\ (A\cap M_i)^{\downarrow_i} \cup \mathrm{univ}(G_k), & \text{if } A\cap M_i \neq \varnothing \text{ for a single } i \in C(k) \\ \mathrm{univ}(G_k), & \text{if there exist } i,j \in C(k),\ i\neq j, \text{ such that } (A\cap M_i)\neq\varnothing \text{ and } (A\cap M_j)\neq\varnothing \end{cases}$$

Proof. Let $\{(G_i, M_i, I_i)\}_{i\in C(k)}$ be the set of children sub-contexts of (G_k, M_k, I_k) after applying the AugmentedCarve algorithm. The AugmentedCarve decomposition partitions $\mathbb{K}_k$ such that $G_i \cap G_j = M_i \cap M_j = \varnothing$ for all $i \neq j$. The proof will be done for a set $A \subseteq M_k$ of attributes; the result for a set of objects is analogous. If $A \subseteq \mathrm{univ}(M_k)$ it is clear that $A^{\downarrow_k} = G_k$.

Let us now suppose that $A\cap M_i \neq \varnothing$ for a single child $i \in C(k)$. By the properties of the AugmentedCarve decomposition, it is $A \subseteq M_i \cup \mathrm{univ}(M_k)$, so $A = (A \cap M_i) \cup (A \cap \mathrm{univ}(M_k))$. Let us call $A_i := A \cap M_i$ and $A_{\mathrm{univ}} := A \cap \mathrm{univ}(M_k)$. Then, $A^{\downarrow_k}$ can be computed as

$$\begin{aligned} A^{\downarrow_k} &= \big((A\cap M_i)\cup(A\cap\mathrm{univ}(M_k))\big)^{\downarrow_k} \\ &= \big(A_i \cup A_{\mathrm{univ}}\big)^{\downarrow_k} \\ &= A_i^{\downarrow_k} \cap A_{\mathrm{univ}}^{\downarrow_k} \end{aligned}$$

by the properties of the Galois connection $({}^{\uparrow_k}, {}^{\downarrow_k})$.

Since $A_{\mathrm{univ}} \subseteq \mathrm{univ}(M_k)$, applying the first case, this yields $A_{\mathrm{univ}}^{\downarrow_k} = G_k$. Now, for A_i, we have

$$A_i^{\downarrow_k} = \{g \in G_k \mid gIm \text{ for all } m \in A_i\}.$$

By AugmentedCarve, the set of objects G_k has been partitioned into $\mathrm{univ}(G_k)$ and $\{G_i\}_{i\in C(k)}$, thus

$$\begin{aligned} A_i^{\downarrow_k} &= \{g \in \mathrm{univ}(G_k) \mid gIm \text{ for all } m\in A_i\} \cup \bigcup_{i\in C(k)} \{g\in G_i \mid gIm \text{ for all } m \in A_i\} \\ &= \mathrm{univ}(G_k) \cup \bigcup_{i\in C(k)} \{g\in G_i \mid gIm \text{ for all } m\in A_i\}. \end{aligned}$$

Since G_j and M_i are in different connected components of the digraph following the AugmentedCarve algorithm, we have that

$$\{g\in G_j \mid gIm \text{ for all } m \in A_i\} = \varnothing.$$

Therefore, we get

$$\mathrm{univ}(G_k) \cup \bigcup_{i\in C(k)} \{g\in G_i \mid gIm \text{ for all } m\in A_i\} = \mathrm{univ}(G_k)\cup A_i^{\downarrow_i}.$$

Adding up, we have that $A_i^{\downarrow_k} = \mathrm{univ}(G_k) \cup A_i^{\downarrow_i}$ and, in consequence,

$$A^{\downarrow_k} = A_i^{\downarrow_k} \cap A_{\mathrm{univ}}^{\downarrow_k} = (\mathrm{univ}(G_k)\cup A_i^{\downarrow_i}) \cap G_k = A_i^{\downarrow_i} \cup \mathrm{univ}(G_k).$$

Finally, assume that for some $i, j \in C(k)$, $i \neq j$ we have that $A\cap M_i \neq \varnothing$ and $A\cap M_j \neq \varnothing$ and consider $m \in A\cap M_i$ and $n \in A \cap M_j$. Then, the previous result can be used as follows: since $\{m\} \subseteq A\cap M_i$ (and i is the only index for which this inclusion holds, by the partition induced by the AugmentedCarve decomposition), we obtain $\{m\}^{\downarrow_k} = \{m\}^{\downarrow_i} \cup \mathrm{univ}(G_k)$. Analogously, $\{n\}^{\downarrow_k} = \{n\}^{\downarrow_j} \cup \mathrm{univ}(G_k)$. Now, as $\{m,n\} \subseteq A$, and using the fact that $G_i \cap G_j = \varnothing$, this implies:

$$\begin{aligned} A^{\downarrow_k} &\subseteq \{m,n\}^{\downarrow_k} = \{m\}^{\downarrow_k} \cap \{n\}^{\downarrow_k} = (\{m\}^{\downarrow_i}\cup\mathrm{univ}(G_k)) \cap (\{n\}^{\downarrow_j}\cup\mathrm{univ}(G_k)) \\ &= (\{m\}^{\downarrow_i} \cap \{n\}^{\downarrow_j}) \cup \mathrm{univ}(G_k) \subseteq (G_i\cap G_j)\cup\mathrm{univ}(G_k) = \mathrm{univ}(G_k). \end{aligned}$$

It only remains to prove $\mathrm{univ}(G_k) \subseteq A^{\downarrow_k}$, but by definition $g \in \mathrm{univ}(G_k)$ if and only if gIm for all $m \in M_k$, thus, in particular, gIm for all $m \in A$ and $\mathrm{univ}(G_k) \subseteq A^{\downarrow_k}$ and we have proved $A^{\downarrow_k} = \mathrm{univ}(G_k)$. □

Remark 1. Observe that since AugmentedCarve partitions the set of attributes M_k into $\mathrm{univ}(M_k)$ and $\{M_i\}_{i\in C(k)}$, the cases considered above cover all the possible cases of subsets of M_k. Any subset $A \subseteq M_k$ is either entirely within $\mathrm{univ}(M_k)$, or overlaps with one unique set M_i for some $i \in C(k)$ or it has some elements in M_i and some in M_j for $i \neq j$.

Corollary 1. *Under the same assumptions as in Lemma 1, let $A \subseteq M_k$ and define $A_i := A \cap M_i$ for all $i \in C(k)$. Then,*

$$A^{\downarrow_k\uparrow_k} = \begin{cases} \mathrm{univ}(M_k) & \text{if } A \subseteq \mathrm{univ}(M_k) \\ A_i^{\downarrow_i\uparrow_i} \cup \mathrm{univ}(M_k) & \text{if } A_i \neq \varnothing \text{ for a single } i \text{ and } A_i^{\downarrow_i} \neq \varnothing \\ M_k & \text{if } A_i \neq \varnothing \text{ for a single } i \text{ and } A_i^{\downarrow_i} = \varnothing \\ M_k & \text{if } A_i \neq \varnothing,\ A_j\neq\varnothing, \text{ with } i\neq j. \end{cases}$$

Proof. Let $A \subseteq M_k$, in order to prove the claim we will consider the four cases separately.

- Assume $A \subseteq \mathrm{univ}(M_k)$, then, by Lemma 1, $A^{\downarrow_k} = G_k$ and applying now the intent operator, in virtue again of Lemma 1, $A^{\downarrow_k\uparrow_k} = G_k^{\uparrow_k} = \mathrm{univ}(M_k)$.
- Assume $A_i \neq \varnothing$ for a single $i \in C(k)$, then $A_i^{\downarrow_k} = A_i^{\downarrow_i} \cup \mathrm{univ}(G_k)$.
  - If $A_i^{\downarrow_i} \neq \varnothing$, then $A_i^{\downarrow_i} \cap G_i \neq 0$, and $i \in C(k)$ is the unique index that satisfies this, therefore, by Lemma 1,

    $$\begin{aligned} A_i^{\downarrow_k\uparrow_k} &= (A_i^{\downarrow_i}\cup\mathrm{univ}(G_k))^{\uparrow_k} = A_i^{\downarrow_i\uparrow_k} \cap \mathrm{univ}(G_k)^{\uparrow_k} \\ &= (A_i^{\downarrow_i\uparrow_i}\cup\mathrm{univ}(M_k))\cap M_k = A_i^{\downarrow_i\uparrow_i}\cup\mathrm{univ}(M_k). \end{aligned}$$
  - If $A_i^{\downarrow_i} = \varnothing$, then $(A_i^{\downarrow_i}\cup\mathrm{univ}(G_k)) \subseteq \mathrm{univ}(G_k)$, which by Lemma 1,

    $$A^{\downarrow_k\uparrow_k} = (A_i^{\downarrow_i}\cup\mathrm{univ}(G_k))^{\uparrow_k} = M_k.$$
- Assume there exist $i, j \in \mathcal{I}$ such that $A_i \neq \varnothing$ and $A_j \neq \varnothing$, then by Lemma 1 we get $A^{\downarrow_k} = \mathrm{univ}(G_k)$, which in particular gives $A^{\downarrow_k} \subseteq \mathrm{univ}(G_k)$ and applying Lemma 1 we get $A^{\downarrow_k\uparrow_k} = \mathrm{univ}(G_k)^{\uparrow_k} = M_k$. □

Note that as a consequence of Corollary 1, $\mathrm{univ}(M_k) \subseteq A^{\downarrow_k\uparrow_k}$ for all $A \subseteq M_k$.

Corollary 2. *Let (G, M, I) be a formal context, then its concept-forming operators $({}^{\uparrow}, {}^{\downarrow})$ can be rewritten as expressions of the derivation operators of each leaf context given by the AugmentedCarve algorithm and the universal objects and attributes of each node in the tree.*

Proof. This is an immediate consequence of applying Lemma 1 to the leaves and going upwards through the tree until the root is reached. □

4.2. Inference rules

This section examines how a system of implications can be inferred from the nodes of the decomposition, with a focus on its typology. This typology is dependent on factors such as the presence or absence of universal elements in a node and the relationship with neighboring nodes (children and parent). The following subsections will focus on different strategies to infer implications, and the corresponding theoretical result proving their validity will be provided. At the end of each subsection, key insights into the significance, computational efficiency, and practical implications of each presented rule are highlighted, ensuring a clearer understanding of its role within the overall framework.

4.2.1. Inheritance

At this point, we present the fundamental pillar on which the proposed method is based, and which has been hinted at by Corollary 2. The construction of the final system of implications will be carried out in accordance with a bottom-up scheme, whereby each node will inherit the set of implications determined by its children, in addition to other implications that are specific to the associated sub-context.

In particular, throughout the subsequent discussion, only implications with non-empty premise will be considered. Note that the only implications of the form $\varnothing \to Y$ valid in the context $\mathbb{K}$ are those where $Y \subseteq \text{univ}(M)$. Thus, we will be interested only in implications with non-empty premises in non-root nodes. Consequently, the initial result to be demonstrated is that this child-to-parent inheritance preserves the validity of implications.

Proposition 1. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_k = (G_k, M_k, I_k)$ a node in its the* Carve *decomposition. Let $\mathbb{K}_i = (G_i, M_i, I_i)$ with $i \in C(k)$ be a child node of $\mathbb{K}_k$, and consider $X, Y \subseteq M_i$, with $X \neq \varnothing$. If $X \to Y$ is a valid implication in $\mathbb{K}_i$, then it is also valid in $\mathbb{K}_k$.*

Proof. Let $\mathbb{K}_k = (G_k, M_k, I_k)$ be a node in the AugmentedCarve decomposition and $i \in C(k)$. Suppose that $X \to Y$ ($X \neq \varnothing$) is a valid implication in $\mathbb{K}_i$ and let us prove its validity in $\mathbb{K}_k$. To this end, we will prove $Y \subseteq X^{\downarrow_k\uparrow_k}$, as this inclusion characterizes the implication validity in $\mathbb{K}_k$ (see Section 2.2). By Corollary 1, since $i \in C(k)$ is the unique index such that $X \subseteq M_i \cup \text{univ}(M_k)$ (by hypothesis, given that $X \subseteq M_i$ and the properties of the AugmentedCarve decomposition) we have that $X^{\downarrow_k\uparrow_k} = X^{\downarrow_i\uparrow_i} \cup \text{univ}(M_k)$ or $X^{\downarrow_k\uparrow_k} = M_k$. Since $X \to Y$ is valid in $\mathbb{K}_i$ we have that $Y \subseteq X^{\downarrow_i\uparrow_i}$ and we get

$$Y \subseteq X^{\downarrow_i\uparrow_i} \subseteq X^{\downarrow_i\uparrow_i} \cup \text{univ}(M_k) \subseteq M_k.$$

Thus, in both cases $Y \subseteq X^{\downarrow_k\uparrow_k}$ and $X \to Y$ is a valid implication in $\mathbb{K}_k$. □

This proposition establishes the fundamental idea that implications remain valid as they propagate from child nodes to their parents in the AugmentedCarve decomposition. The intuition behind this result is that each node in the hierarchy represents a sub-context of the parent node, meaning that any implication derived in a child node must also hold in the broader context of its parent. This guarantees that the knowledge extracted at lower levels of the decomposition is not lost but rather accumulated and extended as we move upwards in the tree.

From a practical point of view, this strategy could be rephrased as the following rule:

Rule 1 (`Inheritance`). *In a node $\mathbb{K}_k = (G_k, M_k, I_k)$, add the implication systems Σ_i with $i \in C(k)$ to Σ_k.*

The Rule 1 (`Inheritance`) serves as a key mechanism for efficiently constructing a complete system of implications. By systematically merging the implication sets of child nodes into their parent, this approach ensures that no valid implications are omitted while avoiding unnecessary recomputation. This bottom-up process simplifies the task of synthesizing the final set of implications at the root node, ultimately leading to a sound and complete representation of the original formal context.

Subsequently, we study different configurations of AugmentedCarve nodes that allow us to define specific strategies to infer valid implications.

4.2.2. Universality

Let us focus on sub-contexts in the AugmentedCarve partition with universal attributes. For instance, continuing from Example 1, let us consider $\mathbb{K}_k = (G_k, M_k, I_k)$, where $G_k = \{1, 2, 3, 4, 5, 6\}$ and $M_k = \{B, C, D, E, F, G\}$ and I_k the induced incidence relation, represented in the following table:

	B	C	D	E	F	G
1	×	×	×			
2	×	×				
3	×			×	×	
4	×			×		
5	×				×	×
6	×					

In this sub-context, it is trivial to see that $m \to B$ is a valid implication for all $m \in M_k$. Recall that, at this point, implications with empty premise are not being considered.

Proposition 2. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_k = (G_k, M_k, I_k)$ a node in the* AugmentedCarve *decomposition of $\mathbb{K}$. Then, for $Y \in \text{univ}(M_k), X \in M_k$, the implication $X \to Y$ is valid in the sub-context $\mathbb{K}_k$.*

Proof. Let $\mathbb{K}_k = (G_k, M_k, I_k)$ be a node in the AugmentedCarve decomposition, $Y \in \text{univ}(M_k)$ and $X \in M_k$, then $X \to Y$ is a valid implication if and only if $Y \in X^{\downarrow_k\uparrow_k}$. This is clear since, by Corollary 1, $\text{univ}(M_k) \subseteq X^{\downarrow_k\uparrow_k}$ and $Y \in \text{univ}(M_k)$. □

Although this result is related only to the validity of this type of implications, some concerns may arise about other valid implications in $\mathbb{K}_k$ besides those of the form $X \to \text{univ}(M_k)$. That is, using only this type of implications does not necessarily ensure the completeness of the implication system in the current node. But, since there are universal attributes in $\mathbb{K}_k$, the only possibilities are:

- $\mathbb{K}_k$ is a leaf node with empty set of objects. In this case, the only valid implications (with non-empty premise) are of the type $X \to \text{univ}(M_k) = M_k$ for all $X \subseteq M_k$. Indeed, we have $X \to Y$ and $Y \to X$ for all $X, Y \subseteq M_k$.
- $\mathbb{K}_k$ is a leaf node with only universal objects. This situation is analogous to the previous one, providing the same set of valid implications.
- $\mathbb{K}_k$ is not a leaf node. In Section 4.3, it will be proved that it suffices to apply the rest of inference rules together with Rule 1 (`Inheritance`), to obtain a complete system of implications.

Corollary 3. *Under the same assumptions as in Proposition 2, let $X \subseteq M_k$. Then, $X \to \text{univ}(M_k)$ is a valid implication in $\mathbb{K}_k$.*

As before, this result can be rewritten as a rule for the construction of the implication system:

Rule 2 (`Universality`). *In a node $\mathbb{K}_k = (G_k, M_k, I_k)$ satisfying $k > 0$ and $\text{univ}(M_k) \neq \varnothing$, for each $X \in M_k$ such that $\text{univ}(M_k) \setminus X \neq \varnothing$, add the implication $X \to \text{univ}(M_k)$ to Σ_k.*

Here the conditions $\text{univ}(M_k) \neq \varnothing$ and $\text{univ}(M_k) \setminus X \neq \varnothing$ for $X \in M_k$ avoid uninformative implications of the form $X \to \varnothing$ and $X \to X$ respectively. In fact, since $\text{univ}(M_k) \setminus X \neq \varnothing$ implies $\text{univ}(M_k) \neq \varnothing$, the first of these conditions serves both purposes. However, since the latter need only be evaluated once per sub-context M_k – rather than once per attribute $X \in M_k$ – we retain both conditions for the purposes of computational efficiency. Also, observe the condition $k > 0$: it states that this rule is not applied in the case of the root node. The reason for this simplification is that, in the root node, if $\text{univ}(M_0) \neq \varnothing$, we will have the implication $\varnothing \to \text{univ}(M_0)$, which is more general than any of the form $X \to \text{univ}(M_0)$, $X \in M_0$. More details will be given at the end of Section 4.3, where a new rule, named `Empty premise` will be presented.

The significance of this rule lies in its ability to simplify the representation of implications in contexts where universal attributes exist. Since universal attributes are present in all object descriptions of the sub-context, they must always appear in the closure of any subset of attributes. The Rule 2 (`Universality`) efficiently encodes this property, ensuring that such attributes are directly inferred from any valid premise without the need for explicit enumeration in every individual case.

Moreover, this rule plays a crucial role in maintaining computational efficiency. By systematically identifying universal attributes and reducing the number of redundant implications, the overall complexity of implication generation is significantly lowered. Without this rule, many implications would need to be derived through multiple inference steps, leading to unnecessary computational overhead. Thus, the Rule 2 (`Universality`) serves as a fundamental mechanism for optimizing the synthesis of a sound and complete implication system.

4.2.3. Contradiction

Let us turn our attention to the node $\mathbb{K}_i = (G_i, M_i, I_i)$ where $G_i = \{3, 4, 5\}$, $M_i = \{E, F, G\}$ and I_i is the corresponding subrelation. Since there are no universal objects in $\mathbb{K}_i$, the set of attributes M_i does not occur at the same time in the formal context, hence the name *contradiction* for this section. In this scenario, the closure of M_i contains $M_k = \{B, C, D, E, F, G\}$, since it is the set of attributes of the parent node, with index $k = \mathcal{P}(i)$. This situation is general, as the next result states.

Proposition 3. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_k = (G_k, M_k, I_k)$ a node of its* AugmentedCarve *decomposition. Let $\mathbb{K}_i = (G_i, M_i, I_i)$ with $i \in \mathcal{C}(k)$ one of $\mathbb{K}_k$'s children, such that $\text{univ}(G_i) = \varnothing$. Then, $M_i \to M_k$ is a valid implication in $\mathbb{K}_k$.*

Proof. To prove the validity of this implication we will use the characterization $M_k \subseteq M_i^{\downarrow_k\uparrow_k}$, which in this case is equivalent to $M_k = M_i^{\downarrow_k\uparrow_k}$. It is clear that $i \in \mathcal{C}(k)$ is the only index such that $M_i \subseteq M_i \cup \text{univ}(M_k)$. In addition, $M_i^{\downarrow_i} = \text{univ}(G_i) = \varnothing$. Therefore, by Corollary 1, $M_k = M_i^{\downarrow_k\uparrow_k}$. □

Let us focus on the case where this *contradiction* scheme is applied to a node, only one of whose children has a non-empty set of attributes, since it will provide a means of producing systems of implications with lower redundancy and more efficiently.

Proposition 4. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_k = (G_k, M_k, I_k)$ a node of its* AugmentedCarve *decomposition. Let us suppose that it has only one child, namely $\mathbb{K}_i = (G_i, M_i, I_i)$, $i \in \mathcal{C}(k)$, with $M_i \neq \varnothing$. Then, the implication $M_i \to M_k$ can be logically inferred using Armstrong's Axioms from either first principles or the implications generated by applying Rule 2 (`Universality`) to $\mathbb{K}_k$.*

Proof. Suppose $\mathbb{K}_i = (G_i, M_i, I_i)$ is the unique child of $\mathbb{K}_k = (G_k, M_k, I_k)$ in the AugmentedCarve decomposition with $M_i \neq \varnothing$. Note that this implies that $\mathbb{K}_i$ has been obtained from $\mathbb{K}_k$ just by removing universal elements. There are two cases:

1. If only universal objects have been removed (i.e., $\text{univ}(M_k) = \varnothing$), then $M_i = M_k$, and the implication $M_i \to M_k$ is trivially deduced using Armstrong's Axiom of `Reflexivity`.
2. If $\text{univ}(M_k) \neq \varnothing$, then $M_k = M_i \cup \text{univ}(M_k)$. By `Reflexivity`, $M_i \to M_i$ is deduced. Since $M_i \subseteq M_k$, by Rule 2 (`Universality`), $M_i \to \text{univ}(M_k)$. Then, by `Union`, we obtain $M_i \to M_k$.

In both cases, $M_i \to M_k$ can be deduced by using the inference schemes mentioned above. □

In other words, the implication $M_i \to M_k$ is not essential if the kth node has only one child with $M_i \neq \varnothing$, that is, if $M_i = M_k \setminus \text{univ}(M_k)$, and it should not be added to Σ_k. These results allow us to state the rule that condenses the idea of this *contradiction* scheme:

Rule 3 (`Contradiction`)**.** *For a node $\mathbb{K}_k = (G_k, M_k, I_k)$, if it has a child of index $i \in \mathcal{C}(k)$, say $\mathbb{K}_i = (G_i, M_i, I_i)$, such that $\text{univ}(G_i) = \varnothing$ and $M_i \neq M_k \setminus \text{univ}(M_k)$, add the implication $M_i \to M_k$ to Σ_k.*

The Rule 3 (`Contradiction`) captures an essential property of the hierarchical decomposition: when a sub-context lacks universal objects, its set of attributes never coexists as a whole in any object of the formal context. Consequently, the closure of these attributes extends beyond the sub-context itself, incorporating attributes from the parent node.

Moreover, this rule prevents redundancy in the implication base by identifying cases where certain implications can be inferred through Armstrong's axioms. In particular, Proposition 4 demonstrates that if a node has only one child containing attributes, then the corresponding implication is automatically deducible from existing rules. By avoiding the explicit addition of such implications to Σ_k, the construction of the implication system remains both minimal and computationally efficient.

4.2.4. Recombination

In our running example, observe that $\mathbb{K}_k = (\{1, \ldots, 6\}, \{B, \ldots, G\}, I_k)$ has two children $\mathbb{K}_i = (\{1, 2\}, \{C, D\}, I_i)$ and $\mathbb{K}_j = (\{3, 4, 5\}, \{E, F, G\}, I_j)$, for some $i, j \in \mathcal{C}(k)$. The reason behind the name *recombination* is that analysis of what can be inferred when two elements of two different children are used as a premise. In our case, it can be observed that, for instance, the closure of $\{D, E\}$ should be, at least, the complete $M_k = \{B, \ldots, G\}$, since, as no object has both attributes, we have $\{D, E\}^{\downarrow_k} = \varnothing$ and therefore $\{D, E\}^{\downarrow_k\uparrow_k} = M_k$. The following result states this fact in general terms:

Proposition 5. *Let $\mathbb{K} = (G, M, I)$ be a formal context, $\mathbb{K}_k = (G_k, M_k, I_k)$ a node of its* Carve *decomposition and $\mathbb{K}_i = (G_i, M_i, I_i)$ and $\mathbb{K}_j = (G_j, M_j, I_j)$ with $i, j \in \mathcal{C}(k)$, $i \neq j$. Then, for $X \in M_i, Y \in M_j$, the implication $XY \to M_k$ is valid in $\mathbb{K}_k$.*

Proof. To prove the validity of the implication $XY \to M_k$, we will use the characterization $M_k \subseteq \{XY\}^{\downarrow_k\uparrow_k}$, which in this case is equivalent to $M_k = \{XY\}^{\downarrow_k\uparrow_k}$. Since $X \in M_i$ and $Y \in M_j$, by Corollary 1, we have that $M_k = \{XY\}^{\downarrow_k\uparrow_k}$. □

As in the rest of sections, this result is re-stated as a construction rule that will be used in the final algorithm to build a set of sound and complete implications:

Rule 4 (`Recombination`)**.** *At a node $\mathbb{K}_k = (G_k, M_k, I_k)$ such that $|\mathcal{C}(k)| \geq 2$, consider its children $\{\mathbb{K}_m\}_{m \in \mathcal{C}(k)}$. Then, for each $X \in M_i$ and $Y \in M_j$, with $i \neq j$, add $XY \to M_k$ to Σ_k.*

Corollary 4. *Let $\mathbb{K}_k$ be a node in the* AugmentedCarve *decomposition and let $i, j \in \mathcal{C}(k)$ such that $\text{univ}(M_i) \neq \varnothing$. Then, for each $X \in \text{univ}(M_i)$ and $Y \in M_j$ with $i \neq j$, infer $XY \to M_k$.*

Corollary 5. *Let $\mathbb{K}_k$ be a node in the* AugmentedCarve *decomposition and let $i, j \in \mathcal{C}(k)$ such that $\text{univ}(M_i) \neq \varnothing$ and $\text{univ}(M_j) \neq \varnothing$. Then, for each $X \in \text{univ}(M_i)$ and $Y \in \text{univ}(M_j)$ with $i \neq j$, infer $XY \to M_k$.*

Both corollaries are immediate consequences of Rule 4 (`Recombination`), but significantly reduce the search space of the rule. Thus, an optimized rule arises.

Rule 4.1 (*Recombination*). *At a node $\mathbb{K}_k = (G_k, M_k, I_k)$ such that $|C(k)| \geq 2$, consider its children $\{\mathbb{K}_m\}_{m\in C(k)}$. Then, if $\mathrm{univ}(M_i) \neq \varnothing$ for each $X \in \mathrm{univ}(M_i)$ and $Y \in M_j$, with $i \neq j$, add $XY \to M_k$ to Σ_k. Otherwise, apply Rule 4 (*`Recombination`*).*

As a matter of fact, in virtue of Rule 2 (`Universality`), the implication system obtained by applying either of these rules are equivalent.

Proposition 6. *Let $\mathbb{K}_k$ be a node in the Carve decomposition and let Σ_1 and Σ_2 be the systems of implications obtained by applying Rules 1–4 and Rules 1–4.1, respectively. Then Σ_1 is logically equivalent to Σ_2.*

Proof. It is clear that every implication in $\Sigma_2 \subseteq \Sigma_1$. If $\mathrm{univ}(M_i) = \varnothing$ for all $i \in C(k)$, then $\Sigma_1 = \Sigma_2$. Assume there exist $i \in C(k)$ such that $\mathrm{univ}(M_i) \neq \varnothing$ and $\Sigma_1 \neq \Sigma_2$. Thus, we need to show that every implication in Σ_1 can be logically entailed from those in Σ_2. Let $X, Y \subseteq M_k$ and let $X \to Y$ be an implication in $\Sigma_1 \setminus \Sigma_2$. Necessarily this implication comes from Rule 4, i.e., there exist $i, j \in C(k)$, $i \neq j$ such that $X = \{mn\}$ where $m \in M_i, n \in M_j$ and $Y = M_k$.

By Rule 4.1 we have that for $m_0 n \to M_k$ where $m_0 \in \mathrm{univ}(M_i)$ and $n \in M_j$. Besides, from Rule 2 (`Universality`), we have that $m \to \mathrm{univ}(M_i)$ and by the axiom of `Reflexivity` we have $\mathrm{univ}(M_i) \to m_0$, which by `Transitivity` gives $m \to m_0$. Applying `Augmentation` we get $mn \to m_0 n$ and by `Transitivity` again we obtain $mn \to M_k$. □

An analogous result can be provided whenever $\mathrm{univ}(M_i)$ and $\mathrm{univ}(M_j)$ are both non-empty, thus a further optimized version of the rule arises as follows.

Rule 4.2 (*Recombination*). *At a node $\mathbb{K}_k = (G_k, M_k, I_k)$ such that $|C(k)| \geq 2$, consider its children $\{\mathbb{K}_m\}_{m\in C(k)}$. Then, if $\mathrm{univ}(M_i) \neq \varnothing$ and $\mathrm{univ}(M_j) \neq \varnothing$, for each $X \in \mathrm{univ}(M_i)$ and $Y \in \mathrm{univ}(M_j)$, with $i \neq j$, add $XY \to M_k$ to Σ_k. Otherwise, apply Rule 4.1 (*`Recombination`*).*

The Rule 4 (`Recombination`) and its derived forms highlight a crucial aspect of how knowledge is synthesized from distinct sub-contexts. Since different children of a node contain separate attribute sets, any implication that involves attributes from two or more children suggests a structural connection at the parent level. This rule effectively captures such dependencies by ensuring that whenever an attribute from one sub-context appears together with an attribute from another sub-context, their closure in the parent context must include all attributes present at that level.

Moreover, the introduction of optimized versions of the this rule significantly reduces the computational complexity of implication generation. By leveraging the presence of universal attributes, the alternative formulations minimize redundant computations and streamline the inference process. This optimization is particularly relevant in large-scale datasets, where reducing the number of generated implications without losing completeness is essential for efficiency.

4.2.5. Indivisibility

In many cases, there will appear leaf nodes in the AugmentedCarve decomposition without universal elements (neither objects nor attributes). In such (heterogeneous) situations, there is not a simple rule that could provide us with a sound and complete implication system. For example, explore the node $\mathbb{K}_k = (\{3, 4, 5\}, \{E, F, G\}, I_k)$:

	E	F	G
3	×	×	
4	×		
5		×	×

Since this node is AugmentedCarve-indivisible, the simplest rule is to resort to a known strategy: the computation of the canonical base of implications. This rule can be stated as follows:

Rule 5 (*Indivisibility*). *At a leaf node $\mathbb{K}_k = (G_k, M_k, I_k)$ with $G_k \neq \varnothing$, $M_k \neq \varnothing$ and $\mathrm{univ}(G_k) = \mathrm{univ}(M_k) = \varnothing$, add its canonical base to its set of implications.*

The Rule 5 (`Indivisibility`) rule addresses the challenge of handling leaf nodes that lack universal elements, making them fundamentally different from other sub-contexts in the AugmentedCarve decomposition. Unlike cases where implications can be inferred through structured rules such as Rule 2 (`Universality`) or Rule 4 (`Recombination`), these indivisible nodes require direct computation of their canonical base. This ensures that all valid implications are extracted without loss of information, preserving the soundness and completeness of the final implication system.

By leveraging the canonical base, this rule guarantees that the minimal and non-redundant set of implications is computed for each indivisible sub-context. While this approach may introduce additional computational overhead compared to other inference rules, it is necessary to handle contexts where no further decomposition is possible. As a result, Rule 5 (`Indivisibility`) provides a crucial mechanism for ensuring that the overall implication system remains both complete and computationally feasible.

4.3. Completeness of the implication system

In the previous section, the soundness of the implication system constructed by the application of Rules 1–5 has been proved. Now, let us analyze its completeness. Since the construction follows a bottom-up approach, we will first prove that all valid implications with non-empty premise can be inferred at the leaf nodes of the AugmentedCarve decomposition, and then we will proceed upwards in the tree.

Lemma 2. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_i = (G_i, M_i, I_i)$ be a leaf node in the AugmentedCarve decomposition. Then, from the set of rules obtained applying Rules 1–5, the implication $A \to A^{\downarrow_i\uparrow_i}$ can be inferred for all $A \subseteq M_i$ with $A \neq \varnothing$.*

Proof. Assume $\mathbb{K} = (G, M, I)$ is a formal context and $\mathbb{K}_i = (G_i, M_i, I_i)$ be a leaf node in the AugmentedCarve decomposition. By the AugmentedCarve algorithm, there are only three options for $\mathbb{K}_i$, either it has no objects and some attributes, denoted by $(\varnothing, M_i)$ in the tree, or both subsets are non-empty with all their elements being universal $(\mathrm{univ}(G_i), \mathrm{univ}(M_i))$ or there are no universal elements in either G_i or M_i. In the first and second cases, let $A \subseteq M_i$, since $\mathrm{univ}(M_i) = M_i$ we get $A^{\downarrow_i\uparrow_i} = M_i$. Applying Rule 2 (`Universality`) to each $Y \in M_i$ and $X \in A$ we have $X \to Y$. By Armstrong's Axiom of `Reflexivity` we get $A \to X$, and by Armstrong's Axiom of `Transitivity` we get $A \to Y$ for all $Y \in M_i$. Thus, applying `Union` we get $A \to M_i = A^{\downarrow_i\uparrow_i}$.

Besides, in the third case, applying Rule 5 (`Indivisibility`) we compute the Duquenne–Guigues base of $\mathbb{K}_i$ via any implication mining algorithm and we get for all $A \neq \varnothing$, $A \subseteq M_i$ the rule $A \to A^{\downarrow_i\uparrow_i}$ can be deduced from the obtained rules. □

Lemma 3. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_i = (G_i, M_i, I_i)$. Then, from the set of rules obtained applying Rules 1–5, the implication $A \to A^{\downarrow\uparrow}$ can be inferred for all $A \subseteq M_i$ with $A \neq \varnothing$.*

Proof. The proof will be done in a bottom-up manner, starting from the base case of the leaves of the decomposition tree and progressing upwards to the root. By Lemma 2, if $\mathbb{K}_i$ is a leaf node, from the set of implications Σ obtained by applying Rules 1–5 we can infer the implications $A \to A^{\downarrow_i\uparrow_i}$.

Let $\mathbb{K}_k$ be a node in the Carve tree and assume that for each child node $i \in C(k)$, the set of implications obtained by applying Rules 1–5 entails the implication $A \to A^{\downarrow_i\uparrow_i}$ for all $A \subseteq M_i, A \neq \varnothing$. Let $A \subseteq M_k$, $A \neq \varnothing$. In virtue of Corollary 1, there are four distinct cases for the closure of A, which will be studied separately. Throughout the proof, the notation $A_i := A \cap M_i$ will be used.

- Assume $A \subseteq \text{univ}(M_k)$, then $A^{\downarrow_k\uparrow_k} = \text{univ}(M_i)$. Let $Y \in \text{univ}(M_k)$ and $X \in A$, applying Rule 2 (`Universality`), we get $X \to Y$. By Armstrong's axiom of `Reflexivity` we have $A \to X$ and, by `Transitivity` we get $A \to Y$, for all $Y \in \text{univ}(M_k)$. By `Union`, we get $A \to \text{univ}(M_k) = A^{\downarrow_k\uparrow_k}$.
- Assume there exists a unique $i \in C(k)$ such that $A \cap M_i \neq \varnothing$ and $A_i^{\downarrow_i} \neq \varnothing$. Then, by Corollary 1 we have $A^{\downarrow_k\uparrow_k} = A^{\downarrow_i\uparrow_i} \cup \text{univ}(M_k)$. By the same argument as in the previous item, we have that $A \to \text{univ}(M_k)$. Also, by hypothesis, the implication $A_i \to A_i^{\downarrow_i\uparrow_i}$ can be obtained and, by Rule 1 (`Inheritance`), it is obtained in $\mathbb{K}_k$. By `Reflexivity` we have $A \to A_i$ and by `Transitivity` we get $A \to A_i^{\downarrow_i\uparrow_i}$. Finally, by `Union` we get $A \to A_i^{\downarrow_i\uparrow_i} \cup \text{univ}(M_k) = A^{\downarrow_k\uparrow_k}$.
- Assume there exists a unique $i \in C(k)$ such that $A \cap M_i \neq \varnothing$ and $A_i^{\downarrow_i} = \varnothing$. Then, by Corollary 1 we have $A^{\downarrow_k\uparrow_k} = M_k$. By hypothesis, we have $A_i \to A_i^{\downarrow_i\uparrow_i}$ and, by Rule 1 (`Inheritance`), it is also obtained in $\mathbb{K}_k$, which, since $A_i^{\downarrow_i} = \varnothing$, is $A_i \to M_i$. There are two cases:

  1. The node i is the only child of node k with $M_i \neq \varnothing$. Then, by Proposition 4, the implication $M_i \to M_k$ can be deduced by using Armstrong's axioms and Rule 2 (`Universality`).
  2. The node i is not the only child of node k with $M_i \neq \varnothing$. In this case, the hypothesis $A_i^{\downarrow_i} = \varnothing$ implies $\text{univ}(G_i) = \varnothing$, hence we can apply Rule 3 (`Contradiction`), to get $M_i \to M_k$.

  In both cases, we can infer $M_i \to M_k$. By the axiom of `Reflexivity` we have $A \to A_i$ and by applying `Transitivity` twice we get $A \to M_k$.
- Assume there exist $i, j \in C(k)$, $i \neq j$ such that $A \cap M_i \neq \varnothing$ and $A \cap M_j \neq \varnothing$. Then, $A^{\downarrow_k\uparrow_k} = M_k$. Let $X \in A \cap M_i$ and $Y \in A \cap M_j$. Then, by Rule 4 (`Recombination`), $XY \to M_k$, by `Reflexivity` we get $A \to XY$ and finally applying `Transitivity` we get $A \to M_k = A^{\downarrow_k\uparrow_k}$.

Since these four cases are the only possible cases we have proved that from the implicational system Σ obtained by the application of Rules 1–5 entails the implication $A \to A^{\downarrow_k\uparrow_k}$, for all $A \subseteq M_k$, $A \neq \varnothing$.

This bottom-up procedure ensures that in $\mathbb{K}$, every implication $A \to A^{\downarrow\uparrow}$, for all $A \subseteq M$, $A \neq \varnothing$ can be entailed. □

In all this discussion, only implications with non-empty premise have been considered. As proved in the next result, it only remains to add a single implication with empty antecedent to ensure the completeness of the implication system.

Theorem 1. *Let $\mathbb{K} = (G, M, I)$ be a formal context and $\mathbb{K}_k = (G_k, M_k, I_k)$ a node in its* AugmentedCarve *decomposition. Let Σ be the set of implications obtained in $\mathbb{K}_k$ by the application of Rules 1–5. Then, the augmented implication system $\Sigma \cup \{\varnothing \to \text{univ}(M_k)\}$ is sound and complete in $\mathbb{K}_k$.*

Proof. The soundness of the implications obtained is given by Propositions 1–3 and 5. The completeness of Σ will be proved by showing that $A \to A^{\downarrow_k\uparrow_k}$ can be derived for all $A \subseteq M_k$. For $A \neq \varnothing$, this is a consequence of Lemma 3. If $A = \varnothing$, it suffices to note that $\text{univ}(M_k) = \varnothing^{\downarrow_k\uparrow_k} = A^{\downarrow_k\uparrow_k}$, hence $A \to A^{\downarrow_k\uparrow_k}$ can be derived by the additional implication $\varnothing \to \text{univ}(M_k)$. □

Since the interest of this approach is to find a sound and complete system of implications for a given formal context $\mathbb{K} = (G, M, I)$, which is the root node in the AugmentedCarve decomposition, a new construction rule is introduced:

Rule 6 (`Empty premise`). *At the root node $\mathbb{K}_0 = (G, M, I)$, add $\varnothing \to \text{univ}(M)$ to the set of implications Σ_0 provided $\text{univ}(M) \neq \varnothing$.*

The proviso $\text{univ}(M) \neq \varnothing$ avoids adding a tautological implication, while not affecting completeness as proven in Theorem 1. As a result, adding Rule 6 to our toolset provides the desired sound and complete implication system:

Corollary 6. *Let $\mathbb{K} = (G, M, I)$ be a formal context. Then, the implication system Σ_0 obtained at the root node by applying Rules 1–6 in its* AugmentedCarve *decomposition is sound and complete.*

4.4. The CarveImp algorithm

The pseudocode for each of Rules 2–4 is listed here as Algorithms 3–5 respectively. These are invoked by the CarveImp algorithm listed as Algorithm 6, which in turn is invoked by the Carve+ algorithm listed as Algorithm 7. Carve+: invokes AugmentedCarve to generate the augmented Carve tree; invokes CarveImp to apply these rules – as well as Rule 5 (`Indivisibility`) at CarveImp lines 2–3, and Rule 1 (`Inheritance`) at line 10 – to each of its nodes; and applies Rule 6 (`Empty premise`) at Carve+ lines 3–4.

Algorithm 3: Universality$(\mathcal{N}, C, \mathcal{P}, k)$

Input: The tree index vector $\mathcal{N}$, the children mapping C, the parent mapping $\mathcal{P}$ and the node index k.
Output: A set Σ of valid implications obtained by the application of Rule 2 (`Universality`).

$\Sigma := \varnothing$
if $k = 0$ **then exit** // Does not apply to root node
if $\text{univ}(M_k) \neq \varnothing$ **then**
  for m **in** M_k **do**
    if $\text{univ}(M_k) \setminus \{m\} \neq \varnothing$ **then**
      $\Sigma := \Sigma \cup \{m \to \text{univ}(M_k)\}$
    end
  end
end
return Σ

Algorithm 4: Contradiction$(\mathcal{N}, C, \mathcal{P}, k)$

Input: The tree index vector $\mathcal{N}$, the children mapping C, the parent mapping $\mathcal{P}$ and the node index k.
Output: A set Σ of valid implications obtained by the application of Rule 3 (`Contradiction`)

$\Sigma := \varnothing$
for i **in** $C(k)$ **do**
  if $\text{univ}(G_i) = \varnothing$ ***and*** $M_i \neq M_k \setminus \text{univ}(M_k)$ **then**
    $\Sigma := \Sigma \cup \{M_i \to M_k\}$
  end
end
return Σ

A formal context $\mathbb{K}$ is first decomposed into its augmented Carve tree and then the implication system computed for the root context $\mathbb{K}_0 = \mathbb{K}$. Since Rules 1, 3 and 4 make use of the implicational systems of the children, the CarveImp algorithm is applied recursively to each child node. Since a leaf node $\mathbb{K}_l$ has by definition no children, Σ_l can be determined without further calls to CarveImp and the recursion bottoms out. While returning from the recursion, CarveImp then applies Rules 1, 3 and 4 to merge and supplement the implicational systems of the children. Once CarveImp returns the resultant implication system Σ_0, Carve+ then applies Rule 6 (`Empty premise`) to supplement it if $\mathbb{K}$ has universal elements.

Rule 5 (`Indivisibility`) is implemented by lines 2–3 of CarveImp, the latter of which invokes the external function ImplicationBase. This serves as a placeholder for any known algorithm which computes a complete set of implications, such as NextClosure [20],

Algorithm 5: RECOMBINATION($\mathcal{N}, \mathcal{C}, \mathcal{P}, k$)

```
Input: The tree index vector 𝒩, the children mapping 𝒞, the
       parent mapping 𝒫 and the node index k.
Output: A set Σ of valid implications obtained by the
        application Rule 4.2 (Recombination)
Σ := ∅
n := length(𝒞(k))
for i from 1 to n − 1 do
    A := M_{𝒞(k)(i)}          // Attribute set of i-th child
    // Use only universal attributes, if present
    if univ(A) = ∅ then M := A else M := univ(A)
    for j from i + 1 to n do
        B := M_{𝒞(k)(j)}  // Attributes in the j-th child.
        if univ(B) = ∅ then N := B else N := univ(B)
        for m in M do
            for n in N do
                Σ := Σ ∪ {mn → M_k}
            end
        end
    end
end
return Σ
```

Algorithm 6: CARVEIMP($\mathcal{N}, \mathcal{C}, \mathcal{P}, k$)

```
Input: The tree index vector 𝒩, the children mapping 𝒞, the
       parent mapping 𝒫 and the node index k.
Output: A set Σ of valid implications
Σ := UNIVERSALITY(𝒩, 𝒞, 𝒫, k)
/* Check if the node is indivisible                    */
if 𝒞(k) = ∅ and univ(G_k) = ∅ and univ(M_k) = ∅ then
    Σ := Σ ∪ IMPLICATIONBASE(G_k, M_k, I_k)
end
else if 𝒞(k) ≠ ∅ then
    // Recurse on its children
    for i in 𝒞(k) do
        Σ_i := CARVEIMP(𝒩, 𝒞, 𝒫, i)
    end
    // Apply the rest of rules
    Σ := Σ ∪ CONTRADICTION(𝒩, 𝒞, 𝒫, k) ∪ RECOMBINATION(𝒩, 𝒞, 𝒫, k)
    Σ := Σ ∪ ⋃_{i∈𝒞(k)} Σ_i   // Apply Rule 1 (Inheritance)
end
return Σ
```

Algorithm 7: CARVE+(G, M, I)

```
Input: A formal context (G, M, I)
Output: A complete set Σ_0 of valid implications in the formal
        context.
𝒩, 𝒞, 𝒫 := AUGMENTEDCARVE(G, M, I)
Σ_0 := CARVEIMP(𝒩, 𝒞, 𝒫, 0)
if univ(M) ≠ ∅ then   // Apply Rule 6 (Empty premise)
    Σ_0 := Σ_0 ∪ {∅ → univ(M)}
end
return Σ_0
```

NEXTCLOSURES [36] or LinCbO [30]. Accordingly its pseudocode is omitted.

Whereas the CARVE algorithm performs the context decomposition and digraph synthesis phases in a single pass through the data, we have – for ease of exposition – invoked AUGMENTEDCARVE and CARVEIMP sequentially, and hence in separate recursions. Since both recursions traverse the same augmented CARVE tree – one to construct it and the other to compute the implication system – and each recursive invocation of CARVEIMP requires access only to a tree node and its children, combining AUGMENTEDCARVE and CARVEIMP within the same recursion should be straightforward.

Example 4. Application of the CARVE+ algorithm to the formal context in Example 1 produces the implications listed in Table 2. With reference to the AUGMENTEDCARVE decomposition in Fig. 3 and its boldface notation for universal elements, we now trace the execution of CARVE+. AUGMENTEDCARVE produces an ordered tree, whose order depends on that in which CONNECTEDCOMPONENTS discovers child sub-contexts; for ease of exposition, assume that they are discovered in – and hence the tree preordered according to – the left–right order shown in Fig. 3. Since nodes of the form $(G_i, \varnothing)$ yield $\Sigma_i = \varnothing$, we omit from the trace those numbered 2,4,10 and 14 in the pre-order. While nodes of the form $(\varnothing, M_i = \{m\})$ also produce $\Sigma_i = \varnothing$, these are included in the trace for reasons which will become apparent at node $\mathbb{K}_7$.

After generating the augmented CARVE tree, CARVE+ invokes CARVEIMP on the root node $\mathbb{K}_0 := \mathbb{K}$, to which Rule 2 (Universality) does not apply. CARVEIMP then recurses on its three subtrees in depth-first manner as detailed in the following paragraphs.

$\mathbb{K}_1 = (\{1, 2, 3, 4, 5, 6\}, \{\mathbf{B}, C, D, E, F, G\})$:

1. Rule 2 (Universality) generates implications 1–5.
2. CARVEIMP is invoked on $\mathbb{K}_3 = (\{\mathbf{1}, 2\}, \{\mathbf{C}, D\})$, where the application of Rule 2 (Universality) yields implication $6 : \{D\} \to \{C\}$. Application of the remaining rules awaits the return of each of the recursive calls to its children.
3. For $\mathbb{K}_5 = (\varnothing, \{\mathbf{D}\})$ no rule applies.
4. Back to $\mathbb{K}_3$, where no additional implications are produced. Rule 3 (Contradiction) does not apply here as $\mathbb{K}_5$ is the only child of $\mathbb{K}_3$ with non-empty attribute set (see Proposition 4).
5. Next, CARVEIMP explores the node $\mathbb{K}_6 = (\{3, 4, 5\}, \{E, F, G\})$. The only applicable rule is Rule 5 (Indivisibility), which generates the Duquenne–Guigues base consisting of implication $7 : \{G\} \to \{F\}$.
6. Returning to $\mathbb{K}_1$, several rules apply:
   - $\mathbb{K}_1$ inherits implications 6–7 by Rule 1 (Inheritance).
   - Rule 3 (Contradiction) can be applied and yields implication $8 : \{E, F, G\} \to \{C, D, B\}$, since $\mathbb{K}_6$ has no universal objects and is not the only child of its parent with non-empty set of attributes.
   - Rule 4 (Recombination) produces implications 9–11.

$\mathbb{K}_7 = (\{\mathbf{7}, 8, 9, 10\}, \{H, I, J, K\})$:

1. Rule 2 (Universality) does not apply, since $\mathbb{K}_7$ has no universal attributes.
2. The first child is leaf node $\mathbb{K}_8 = (\varnothing, \{\mathbf{K}\})$, where no rule applies.
3. The second branch explores $\mathbb{K}_9 = (\{\mathbf{8}, 9\}, \{\mathbf{H}, I\})$ where the application of Rule 2 (Universality) generates implication $12 : \{I\} \to \{H\}$.
4. For its leaf-node child $\mathbb{K}_{11} = (\varnothing, \{\mathbf{I}\})$, no rule applies.
5. Returning to $\mathbb{K}_9$ there are no implications to inherit and the remaining rules do not apply.
6. The third branch in this subtree analyses $\mathbb{K}_{12} = (\{\mathbf{10}\}, \{\mathbf{J}\})$, to which no rule applies.
7. Returning to $\mathbb{K}_7$, the following rules apply:
   - Rule 1 (Inheritance), whereby implication 12 is inherited.

Table 2
Implications generated by Carve+ for the formal context in Table 1. Each row also lists a serial number #, source node index k and rule which, when applied to $\mathbb{K}_k$, generated the implication.

# :	{Antecedent}	→	{Consequent}	k	Rule
1 :	{C}	→	{B}	1	Universality
2 :	{D}	→	{B}	1	Universality
3 :	{E}	→	{B}	1	Universality
4 :	{F}	→	{B}	1	Universality
5 :	{G}	→	{B}	1	Universality
6 :	{D}	→	{C}	3	Universality
7 :	{G}	→	{F}	6	Indivisibility
8 :	{E, F, G}	→	{B, C, D}	1	Contradiction
9 :	{C, E}	→	{B, D, F, G}	1	Recombination
10 :	{C, F}	→	{B, D, E, G}	1	Recombination
11 :	{C, G}	→	{B, D, E, F}	1	Recombination
12 :	{I}	→	{H}	9	Universality
13 :	{K}	→	{H, I, J}	7	Contradiction
14 :	{H, J}	→	{I, K}	7	Recombination
15 :	{M}	→	{L}	13	Universality
16 :	{B, C, D, E, F, G}	→	{A, H, I, J, K, L, M}	0	Contradiction
17 :	{B, H}	→	{A, C, D, E, F, G, I, J, K, L, M}	0	Recombination
18 :	{B, I}	→	{A, C, D, E, F, G, H, J, K, L, M}	0	Recombination
19 :	{B, J}	→	{A, C, D, E, F, G, H, I, K, L, M}	0	Recombination
20 :	{B, K}	→	{A, C, D, E, F, G, H, I, J, L, M}	0	Recombination
21 :	{B, L}	→	{A, C, D, E, F, G, H, I, J, K, M}	0	Recombination
22 :	{H, L}	→	{A, B, C, D, E, F, G, I, J, K, M}	0	Recombination
23 :	{I, L}	→	{A, B, C, D, E, F, G, H, J, K, M}	0	Recombination
24 :	{J, L}	→	{A, B, C, D, E, F, G, H, I, K, M}	0	Recombination
25 :	{K, L}	→	{A, B, C, D, E, F, G, H, I, J, M}	0	Recombination
26 :	{ }	→	{A}	0	Empty premise

- Rule 3 (Contradiction) produces implication 13 : $\{K\} \to \{H, I, J\}$. Observe that even though $\Sigma_8 = \varnothing$, keeping the orphan node $\mathbb{K}_8$ in the AugmentedCarve decomposition has been essential to finding implication 13.
- Rule 4 (Recombination), which gives implication 14 : $\{H, J\} \to \{I, K\}$.

$\mathbb{K}_{13} = (\{\mathbf{11}, 12\}, \{\mathbf{L}, M\})$:

1. Rule 2 (Universality) generates implication 15 : $\{M\} \to \{L\}$.
2. Exploration of child $\mathbb{K}_{15} = (\varnothing, \{\mathbf{M}\})$ finds no applicable rule.
3. Upon returning to $\mathbb{K}_{13}$, no additional rules apply.

$\mathbb{K}_0$: CarveImp finally returns to the root node, to which the following rules are applicable:

1. Rule 1 (Inheritance), whereby implications 1 – 15 are inherited.
2. Rule 3 (Contradiction), producing implication 16.
3. By Rule 4 (Recombination), the implications 17 – 25 are added.
4. CarveImp finally returns to Carve+, where namedrulempty (Empty premise) produces implication 26 : $\varnothing \to \{A\}$.

Note that CarveImp produces 26 implications in contrast with the canonical base presented in Example 1, which has 18 implications. But these two systems of implications are equivalent, and, from a practical point of view, the next section will show that mining implications using Carve+ is more efficient for AugmentedCarve-amenable contexts.

5. Experimental evaluation

In this section, we demonstrate the practical utility of Carve+ through its application to AugmentedCarve-amenable empirical formal contexts chosen from domains as disparate as software engineering, life sciences, text mining, social and collaboration networks. It is important to note that Carve+ is designed to be a *helper*, and not a substitution, for other implication mining methods, hence the aim of this analysis is to determine to what extent the application of Carve+ can help other implication mining techniques. To this end, we test the effect of using our proposal in comparison to the raw application of an implication mining technique. In our experiments, the NextClosure algorithm [20] has been used as the baseline, since it is the *de facto* standard in implication mining in FCA. In addition, NextClosure will play the role of the ImplicationBase function in Algorithm 6.

Table 3
Dataset characteristics and preprocessing statistics. Time is measured in seconds.

Dataset	Size		Time (s)	
	Original	Reduced	JRL order	Reduction
VirusHost	49 × 75	49 × 75	0.002	0.000
Graph Editor	64 × 63	63 × 63	0.001	0.001
20 Newsgroups	37 660 × 239	114 × 119	4.896	0.067
MammalVirus	526 × 504	241 × 252	0.123	0.030
Board-Directors	1422 × 355	413 × 350	0.480	0.084
InfoVis 2004	1036 × 612	559 × 511	0.479	0.241

The datasets are described briefly below, and their key characteristics summarized in Table 3. Note that the formal contexts have been reduced after performing JRL ordering. This reduction serves a two-fold purpose: first, putting all algorithms on the same ground, the subsequent comparison will be fairer; second, without reduction, the computation times of NextClosure for larger datasets would be prohibitive.

VirusHost: Formal context of protein-protein interactions between human DNA tumor virus proteins and host proteins, via yeast two-hybrid (Y2H) experiments [39].

Graph editor: A formal context of classes and methods in a graph editor software project [40]. The relation in this context is "class X *has implemented* method Y".

20 Newsgroups: This dataset contains approximately 20,000 messages distributed across 20 distinct Usenet newsgroups [41], each focusing on specific topics such as technology, politics,

Table 4
Properties of AugmentedCarve and the resultant recursion trees for the datasets used in the experimentation. `Time` is the run-time of the AugmentedCarve algorithm, `Max.` and `Avg.` represent the maximum and mean number of children for each non-leaf node in the tree, `Count` stands for the total number of nodes, and `Largest leaf` stands for the size ($|G_i| \times |M_i|$) of the largest leaf sub-context.

Dataset	Time	Children		Nodes	
		Max.	Avg.	Count	Largest leaf
VirusHost	0.054	11	6	13	38 × 64
Graph Editor	0.057	32	3.1	66	4 × 5
20 Newsgroups	0.062	47	10.2	52	62 × 67
MammalVirus	0.095	34	34	35	206 × 216
Board-Directors	0.217	126	13.1	145	261 × 209
InfoVis 2004	0.468	257	7.5	333	96 × 97

or sports. For our experiments, the document collection was processed to construct a binary formal context resembling a document-term matrix. Relevant terms were selected based on their relative importance and frequency, specifically those appearing in at least 0.1% of the documents. Each term's presence or absence in a message was encoded as a binary relation, capturing the relationship between documents and their terms. This preprocessing resulted in a manageable and structured context suitable for formal concept analysis and implication derivation.

The *20 Newsgroups* dataset serves as a challenging benchmark due to its large size and high dimensionality, making it ideal for evaluating the scalability and effectiveness of implication mining algorithms.

MammalVirus: A formal context of mammalian species and the viruses they are associated with [42].

Board directors: The dataset originates from the work [43] and analyzes the presence, prominence, and social capital of women on corporate boards in Norway. This dataset involves a formal context representing board members and their affiliations, with entities reflecting individuals and attributes indicating organizational connections or roles, highlighting the broader social and structural dynamics influencing board composition.

InfoVis 2004: Bibliographic metadata on 612 publications on information visualization and the standardized names of their 1036 unique authors prepared for use in the InfoVis contest [44] at the 2004 IEEE InfoVis Symposium. The relation in this formal context is "author X *wrote* a paper entitled Y".

All the algorithms have been implemented and all the experiments have been run using the `fcaR` library [45] in the R programming language, with C code for specific time-consuming procedures, using the same data structures in all cases.[3] Also, parallel implementation was made possible by the use of the `parallel` library available in base R [46].

In Table 4, some of the characteristics of the AugmentedCarve decomposition of these datasets are presented. In particular, note the low computation time needed to build the recursion tree.

To assess the efficiency and effectiveness of our method, we compared it with the standard NextClosure algorithm using the following metrics:

- **Computation Time:** The time required to calculate implications. Our approach (Carve+) was evaluated in both parallel and sequential modes. If the execution of either algorithm was not completed within sixty hours, it is recorded in Table 5 as having *timed out*.

[3] Any data relating to formal contexts or code will be made available on request.

Table 5
Comparison of computation time, cardinality, and redundancy between NextClosure and Carve+.

Dataset	NextClosure		Carve+		
	Time (s)	$\|\Sigma\|$	Time (s)		$\|\Sigma\|$
			Paral.	Seq.	
VirusHost	1.327	1 240	0.656	0.656	1491
Graph Editor	0.434	658	0.028	0.028	882
20 Newsgroups	26.563	4 041	1.337	1.338	5282
MammalVirus	1815.640	12 702	360.445	360.446	16 566
Board-Directors	105 532.2	59 537	4471.835	4471.882	59 826
InfoVis 2004	– timed out –		31.702	31.849	121 962

Dataset	Comparison	
	Acc. factor	Redundancy (%)
VirusHost	1.884	20.242
Graph Editor	3.082	34.043
20 Newsgroups	18.667	30.710
MammalVirus	5.035	30.420
Board-Directors	23.598	0.490
InfoVis 2004	>6715	–

- **Implication Count:** The total number of implications generated. NextClosure yields the canonical base, which has no redundancy. To quantify the redundancy introduced by Carve+, we measured the proportion of extra implications generated.

The experimental results are summarized in Table 5. The experimental evaluation highlights several key aspects of Carve+ in comparison with NextClosure. First, the efficiency gains are notable, with Carve+ achieving significant acceleration factors, such as a factor of at least 6715 on the InfoVis 2004 dataset. This improvement is attributed to the efficient decomposition of the formal context which significantly reduces the size of the largest sub-context to which NextClosure is applied. Even for smaller datasets, such as the Graph Editor, the acceleration factor remains impressive, indicating consistent performance across various dataset scales. Notably, however, the additional improvement obtained through parallel implementation was negligible; upon further investigation, it became apparent that almost all of the execution time was spent by NextClosure on the largest leaf node of the AugmentedCarve tree, suggesting that the remaining leaf-node sub-contexts were significantly smaller, and hence that parallelization of Carve+ (vice NextClosure [36]) is not beneficial — at least for the empirical formal contexts investigated here. Thus the principal benefit of Carve+ may be obtained without requiring parallel processing.

A second observation relates to the redundancy in implications produced by Carve+. While NextClosure generates a canonical base without redundancy, Carve+ consistently produces additional implications, resulting in redundancy ratios between 0.49% and 34%. Although this may initially appear as a drawback, the increased redundancy is a trade-off for the substantial reduction in computation time. This makes Carve+ a practical solution for scenarios where computational efficiency is a priority, even if it comes at the cost of some redundancy in the output.

Scalability is another strength of Carve+, as evidenced by its performance on high-dimensional datasets such as the 20 Newsgroups dataset. Despite the dataset's large size and inherent challenges, Carve+ demonstrated reasonable preprocessing and implication generation times, confirming its robustness and scalability for complex data scenarios.

When compared directly with NextClosure, Carve+ addresses the computational bottlenecks associated with larger datasets. While NextClosure remains reliable for generating a canonical base, its performance diminishes significantly as the dataset size increases. In contrast, Carve+ expands the range of practical applications, enabling efficient analysis of datasets like InfoVis 2004, which would otherwise be infeasible with traditional methods.

6. Conclusions and future work

In this paper, the Carve algorithm is adapted to obtain a divide-and-conquer parallelizable algorithm to compute a complete implicational system for any formal context. Just as the Carve algorithm wraps rather than replaces other methods for computing the lattice digraph, the AugmentedCarve decomposition and CarveImp collectively wrap *any* other methods for computing a base of attribute implications. In particular, AugmentedCarve serves as a preprocessing step which takes negligible time to hierarchically partition the formal context. CarveImp then invokes an external implication algorithm such as NextClosure on each of the smaller, leaf-node sub-contexts, and collates and supplements the resultant sets of implications to obtain the overall implicational system. Indeed, a set of rules has been proposed to gather the obtained implications so as to avoid unnecessary redundancy and the theoretical results in Section 4.3 show that the set of implications obtained is complete. Furthermore, the experimental results show that whenever the formal context is amenable, that is, the AugmentedCarve decomposition is not trivial, the combined runtime of AugmentedCarve and CarveImp is much faster than applying the NextClosure algorithm directly. Since Carve+ serves as a wrapper for other algorithms, another algorithm may be substituted for NextClosure, according to the needs of the user or the characteristics of the problem, as long as the implication mining process returns a sound and complete implication system.

The improvements observed with Carve+ have practical implications for fields requiring rapid implication generation, such as biological data analysis and text mining. By striking a balance between computational efficiency and practical usability, Carve+ emerges as a versatile and robust alternative for deriving implications in formal contexts.

Reducing the redundancy of the obtained implicational system, and studying its relationship with the Duquenne–Guigues base [32], are top priorities for future work. The relationship between the largest leaves found by AugmentedCarve and the Knowledge Cores of [38] also warrants investigation. And finally, devising a Carve-like algorithm for fuzzy or multi-adjoint formal contexts, where the complexity problems are much greater than in the classical framework [17,47], promises significant benefit.

CRediT authorship contribution statement

Domingo López-Rodríguez: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Manuel Ojeda-Hernández:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Tim Pattison:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is partially supported by the Kempe foundation, Sweden, 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 JCSMK24-0053 (Kempe), PID2021-127870OB-I00 and (MCIU/AEI/FEDER, UE) and the VALID research project (PID2022-140630NB-I00 funded by MCIN/AEI/10.13039/501100011033).

Data availability

Data will be made available on request.

References

[1] R. Wille, Restructuring lattice theory: An approach based on hierarchies of concepts, Ordered Sets 83 (1982) 445–470, http://dx.doi.org/10.1007/978-94-009-7798-3_15.

[2] J. Poelmans, D.I. Ignatov, S. Viaene, G. Dedene, S.O. Kuznetsov, Text mining scientific papers: A survey on FCA-based information retrieval research, in: Proceedings of the 12th Industrial Conference on Advances in Data Mining: Applications and Theoretical Aspects, ICDM'12, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 273–287, http://dx.doi.org/10.1007/978-3-642-31488-9_22.

[3] U. Priss, Formal concept analysis in information science, Annu. Rev. Inf. Sci. Technol. 40 (2006) 521–543, http://dx.doi.org/10.1002/aris.1440400120.

[4] F.J. Valverde-Albacete, C. Peláez-Moreno, Systems vs. methods: an analysis of the affordances of formal concept analysis for information retrieval, in: C. Carpineto, S.O. Kuznetsov, A. Napoli (Eds.), Proceedings of the Workshop Formal Concept Analysis Meets Information Retrieval (FCAIR 2013), in: CEUR Workshop Proceedings, vol. 977, Moscow, Russia, 2013, pp. 113–126.

[5] V. Codocedo, A. Napoli, Formal concept analysis and information retrieval – a survey, in: J. Baixeries, C. Sacarea, M. Ojeda-Aciego (Eds.), Formal Concept Analysis, in: Lecture Notes in Computer Science, vol. 9113, Springer, Cham, 2015, pp. 61–77, http://dx.doi.org/10.1007/978-3-319-19545-2_4.

[6] M.E. Cornejo, J. Medina, C. Rubio-Manzano, Formal analysis of solar power and weather data, in: Computational Intelligence and Mathematics for Tackling Complex Problems 2, Springer, 2022, pp. 81–89, http://dx.doi.org/10.1007/978-3-030-88817-6_10.

[7] J. Guniš, L. Šnajder, L. Antoni, P. Eliaš, O. Krídlo, S. Krajči, Formal concept analysis of students' solutions on computational thinking game, IEEE Trans. Educ. (2024) http://dx.doi.org/10.1109/te.2024.3442612.

[8] M. Ojeda-Hernández, F. Pérez-Gámez, D. López-Rodríguez, N. Madrid, Á. Mora, Minimal generators from positive and negative attributes: Analysing the knowledge space of a mathematics course, Int. J. Comput. Intell. Syst. 15 (1) (2022) 58, http://dx.doi.org/10.1007/s44196-022-00123-3.

[9] H. Yang, Y. Yao, K. Qin, A lattice-theoretic model of three-way conflict analysis, Knowl.-Based Syst. 288 (2024) 111470, http://dx.doi.org/10.1016/j.knosys.2024.111470.

[10] G. Lang, J. Luo, Y. Yao, Three-way conflict analysis: a unification of models based on rough sets and formal concept analysis, Knowl.-Based Syst. 194 (2020) 105556, http://dx.doi.org/10.1016/j.knosys.2020.105556.

[11] Z. Wang, C. Shi, L. Wei, Y. Yao, Tri-granularity attribute reduction of three-way concept lattices, Knowl.-Based Syst. 276 (2023) 110762, http://dx.doi.org/10.1016/j.knosys.2023.110762.

[12] M. Ojeda-Hernández, D. López-Rodríguez, Á. Mora, A formal concept analysis approach to hierarchical description of malware threats, Forensic Sci. Int.: Digit. Investig. 50 (2024) 301797, http://dx.doi.org/10.1016/j.fsidi.2024.301797.

[13] M. Ojeda-Hernández, D. López-Rodríguez, Á. Mora, Lexicon-based sentiment analysis in texts using formal concept analysis, Internat. J. Approx. Reason. 155 (2023) 104–112, http://dx.doi.org/10.1016/j.ijar.2023.02.001.

[14] F. Pérez-Gámez, P. Cordero, M. Enciso, Á. Mora, Simplification logic for the management of unknown information, Inform. Sci. 634 (2023) 505–519, http://dx.doi.org/10.1016/j.ins.2023.03.015.

[15] J. Liu, J.-S. Mi, D.-Y. Niu, A novel method for generating a canonical basis for decision implications based on object-induced three-way operators, Knowl.-Based Syst. 283 (2024) 111161, http://dx.doi.org/10.1016/j.knosys.2023.111161.

[16] R. Belohlavek, Fuzzy Relational Systems: Foundations and Principles, Vol. 20, Springer Science & Business Media, 2012.

[17] M.E. Cornejo, J. Medina, F.J. Ocaña, Attribute implications in multi-adjoint concept lattices with hedges, Fuzzy Sets and Systems 479 (2024) 108854, http://dx.doi.org/10.1016/j.fss.2023.108854.

[18] L. Antoni, P. Eliaš, J. Guniš, D. Kotlárová, S. Krajči, O. Krídlo, P. Sokol, L. Šnajder, Bimorphisms and attribute implications in heterogeneous formal contexts, Internat. J. Approx. Reason. 172 (2024) 109245, http://dx.doi.org/10.1016/j.ijar.2024.109245.

[19] B. Ganter, S. Obiedkov, S. Rudolph, G. Stumme, Conceptual Exploration, Springer, 2016, http://dx.doi.org/10.1007/978-3-662-49291-8.

[20] B. Ganter, Two basic algorithms in concept analysis, in: L. Kwuida, B. Sertkaya (Eds.), Proceedings of the 8th International Conference on Formal Concept Analysis, in: Lecture Notes in Artificial Intelligence, vol. 5986, Springer Berlin Heidelberg, 2010, pp. 312–340, http://dx.doi.org/10.1007/978-3-642-11928-6_22.

[21] S. Kuznetsov, A fast algorithm for computing all intersections of objects in a finite semi-lattice, Autom. Doc. Math. Linguist. 27 (1993) 11–21.

[22] S. Andrews, In-Close2, a high performance formal concept miner, in: International Conference on Conceptual Structures, ICCS 2011, in: Lecture Notes in Computer Science, vol. 6828 LNAI, Springer, 2011, pp. 50–62, http://dx.doi.org/10.1007/978-3-642-22688-5_4.

[23] J. Konecny, P. Krajča, Systematic categorization and evaluation of CbO-based algorithms in FCA, Inform. Sci. 575 (2021) 265–288, http://dx.doi.org/10.1016/j.ins.2021.06.024.

[24] T. Pattison, A. Ceglar, D. Weber, Efficient formal concept analysis through recursive context partitioning, in: Proceedings of the 14th International Conference on Concept Lattices and their Applications, CEUR Workshop Proceedings, CEUR-WS.org, Olomouc, Czech Republic, 2018, pp. 219–230.

[25] Y. Ke, J. Li, S. Li, Bit-close: a fast incremental concept calculation method, Appl. Intell. 54 (3) (2024) 2582–2593, http://dx.doi.org/10.1007/s10489-024-05272-w.

[26] T. Pattison, D. Weber, A. Ceglar, Enhancing layout and interaction in formal concept analysis, in: Proceedings of the 2014 IEEE Pacific Visualization Symposium, PacificVis, 2014, pp. 248–252, http://dx.doi.org/10.1109/pacificvis.2014.21.

[27] T. Pattison, A. Nataraja, Doubly-lexical order supports standardisation and recursive partitioning of formal context, in: D. Dürrschnabel, D. López Rodríguez (Eds.), Formal Concept Analysis, Springer Nature, Switzerland, Cham, 2023, pp. 17–32, http://dx.doi.org/10.1007/978-3-031-35949-1_2.

[28] M.U. Bhatti, N. Anquetil, M. Huchard, S. Ducasse, A catalog of patterns for concept lattice interpretation in software reengineering, in: D. Zhang, M. Reformat, S. Gokhale, J.C. Maldonado (Eds.), Proceedings of the 24th International Conference on Software Engineering & Knowledge Engineering, SEKE'2012, Knowledge Systems Institute Graduate School, Hotel Sofitel, Redwood City, San Francisco Bay, USA, 2012, pp. 118–124, URL https://inria.hal.science/hal-00700046.

[29] L.C. Freeman, Cliques, galois lattices, and the structure of human social groups, Soc. Netw. 18 (1996) 173–187, http://dx.doi.org/10.1016/0378-8733(95)00271-5.

[30] R. Janostik, J. Konecny, P. Krajča, Lincbo: Fast algorithm for computation of the duquenne-guigues basis, Inform. Sci. 572 (2021) 223–240, http://dx.doi.org/10.1016/j.ins.2021.04.104.

[31] W.W. Armstrong, Dependency structures of database relationships, Inf. Process. 74 (1974) 580–583.

[32] J.-L. Guigues, V. Duquenne, Familles minimales d'implications informatives d'un tableau de données binaires, Math. et Sci. Hum. 95 (1986) 5–18.

[33] B. Ganter, R. Wille, Formal Concept Analysis: Mathematical Foundations, Springer, 1999, http://dx.doi.org/10.1007/978-3-031-63422-2.

[34] J. Poelmans, S.O. Kuznetsov, D.I. Ignatov, G. Dedene, Formal concept analysis in knowledge processing: A survey on models and techniques, Expert Syst. Appl. 40 (2013) 6601–6623, http://dx.doi.org/10.1016/j.eswa.2013.05.007.

[35] P. Valtchev, V. Duquenne, On the merge of factor canonical bases, in: International Conference on Formal Concept Analysis, Springer, 2008, pp. 182–198, http://dx.doi.org/10.1007/978-3-540-78137-0_14.

[36] F. Kriegel, D. Borchmann, NextClosures: Parallel computation of the canonical base with background knowledge, Int. J. Gen. Syst. 46 (5) (2017) 490–510, http://dx.doi.org/10.1080/03081079.2017.1349570.

[37] J.-F. Viaud, K. Bertet, R. Missaoui, C. Demko, Using congruence relations to extract knowledge from concept lattices, Discrete Appl. Math. 249 (2018) 135–150, http://dx.doi.org/10.1016/j.dam.2016.11.021.

[38] T. Hanika, J. Hirth, Knowledge cores in large formal contexts, Ann. Math. Artif. Intell. 90 (6) (2022) 537–567, http://dx.doi.org/10.1007/s10472-022-09790-6.

[39] O. Rozenblatt-Rosen, R.C. Deo, M. Padi, G. Adelmant, M.A. Calderwood, T. Rolland, M. Grace, A. Dricot, M. Askenazi, M. Tavares, et al., Interpreting cancer genomes using systematic host network perturbations by tumour virus proteins, Nature 487 (7408) (2012) 491–495, http://dx.doi.org/10.1038/nature11288.

[40] G. Snelting, F. Tip, Understanding class hierarchies using concept analysis, ACM Trans. Program. Lang. Syst. (TOPLAS) 22 (3) (2000) 540–582, http://dx.doi.org/10.1145/353926.353940.

[41] K. Lang, 20 newsgroups dataset, 1997, http://qwone.com/jason/20Newsgroups/, Online (Accessed 6 December 2024).

[42] K.J. Olival, P.R. Hosseini, C. Zambrana-Torrelio, N. Ross, T.L. Bogich, P. Daszak, Host and viral traits predict zoonotic spillover from mammals, Nature 546 (7660) (2017) 646–650, http://dx.doi.org/10.1038/nature22975.

[43] C. Seierstad, T. Opsahl, For the few not the many? the effects of affirmative action on presence, Prominence, Soc. Cap. Women Dir. Nor. Scand. J. Manag. 27 (1) (2011) 44–54, http://dx.doi.org/10.1016/j.scaman.2010.10.002.

[44] C. Plaisant, J.-D. Fekete, G. Grinstein, Promoting insight-based evaluation of visualizations: From contest to benchmark repository, IEEE Trans. Vis. Comput. Graphics 14 (1) (2008) 120–134, http://dx.doi.org/10.1109/tvcg.2007.70412.

[45] P. Cordero, M. Enciso, D. López-Rodríguez, Á. Mora, Fcar, formal concept analysis with r, R J. 14 (2022) 341–361, http://dx.doi.org/10.32614/RJ-2022-014.

[46] R Core Team, R: A language and environment for statistical computing, in: R Foundation for Statistical Computing, Vienna, Austria, 2023, URL https://www.R-project.org/.

[47] R.G. Aragón, J. Medina, E. Ramírez-Poussa, Reducing concept lattices by means of a weaker notion of congruence, Fuzzy Sets and Systems 418 (2021) 153–169, http://dx.doi.org/10.1016/j.fss.2020.09.013.