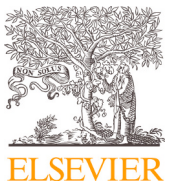
International Journal of Approximate Reasoning 197 (2026) 109747



Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

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

Effective greedy Boolean matrix factorization via the Rice-Siff algorithm

Lubomir Antoni [a,*], Dominika Kotlárová [a], Ondrej Krídlo [a], Domingo López-Rodríguez [b], Manuel Ojeda-Aciego [b]

[a] Institute of Computer Science, Faculty of Science, Pavol Jozef Šafárik University in Košice, Jesenná 5, Košice, 04001, Slovakia
[b] Universidad de Málaga. Departamento de Matemática Aplicada, Blv. Louis Pasteur 35, Málaga, 29071, Spain

ARTICLE INFO

Keywords:
Boolean matrix decomposition
Formal context factorisation
Greedy algorithm

ABSTRACT

This paper introduces a new Boolean Matrix Factorization (BMF) algorithm based on the Rice-Siff agglomerative clustering method. Our approach, Rice-Siff Factorization (RSF), integrates a greedy set-cover framework, where formal concepts serve as interpretable factors, with a dynamic candidate-generation process. By incorporating optimized variants such as RSF with Early Stopping (RSF-ES), we propose a pruning criterion based on order-theoretic properties to detect redundant candidates and significantly enhance factor extraction. Extensive synthetic benchmarks and experiments on real-world datasets demonstrate the effectiveness and robustness of the proposed framework, showing that RSF-ES provides significant scalability gains, yielding speedups on high-dimensional datasets with thousands of attributes while maintaining mathematical exactness. A comprehensive comparison with established factorization algorithms and an analysis of its theoretical properties show that RSF-ES represents a highly efficient and scalable solution for Boolean data analysis.

1. Introduction

Lots of practical applications, from natural language processing to medicine or bioinformatics, need the capacity to extract meaningful structure from high-dimensional observations. A substantial proportion of these can be interpreted as Boolean values, whose analysis poses unique challenges and opportunities.

Formally, we will be dealing with the classical matrix decomposition problem, which considers an $n \times m$ matrix $\mathbf{R}$ and an integer $k < m$, and the objective is to identify an $n \times k$ matrix $\mathbf{U}$ and a $k \times m$ matrix $\mathbf{V}$ such that $\mathbf{R}$ can be approximated as the product of $\mathbf{U}$ and $\mathbf{V}$. If the input matrix $\mathbf{R}$ is binary, it is natural to require that both $\mathbf{U}$ and $\mathbf{V}$ are also binary, and the matrix multiplication is defined in terms of the logical operators for disjunction and conjunction. This is the *Boolean Matrix Factorization* (BMF) problem.

Miettinen *et al.* studied this problem in [1] (called it the *Discrete Basis Problem*, DBP), proved that finding the best approximation is NP-hard, and introduced the ASSO algorithm, which remains one of the most widely used BMF methods. ASSO's efficiency and reasonable empirical performance have made it a benchmark against which newer methods are compared, though it offers no optimality guarantees and can produce redundant or suboptimal factorisations.

Even though finding the best approximation for BMF is NP-hard, in [2,3] it was proved that polynomial-time approximation to arbitrary precision algorithms exist but, unfortunately, running times of those algorithms are hardly practical. Summarizing, algorithms that are efficient in the formal sense of complexity theory are utterly useless in practice.

* Corresponding author.
E-mail addresses: lubomir.antoni@upjs.sk (L. Antoni), dominika.kotlarova@student.upjs.sk (D. Kotlárová), ondrej.kridlo@upjs.sk (O. Krídlo), dominlopez@uma.es (D. López-Rodríguez), aciego@uma.es (M. Ojeda-Aciego).

https://doi.org/10.1016/j.ijar.2026.109747
Received 21 March 2026; Received in revised form 13 June 2026; Accepted 13 June 2026
Available online 15 June 2026

Recent work on this research line has been pursued in several complementary directions: [4] proposes algorithms that perform alternating optimisation of the factor matrices, solving each subproblem via integer programming; the introduction of probabilistic generative models for BMF marks a significant methodological advance, and [5] proposes a generative model in which each observed entry is produced by a Boolean combination of latent factors; another recent direction in BMF research can be found in [6] where a preprocessing method that reorders binary data to reveal banded structure and applies image morphology operations to enhance this structure before factorisation; last but not least, there is a deep connection between BMF and Formal Concept Analysis (FCA). Belohlavek and Vychodil [7] proved that formal concepts provide optimal and universal factors for BMF, and introduced the now so-called *Greedy Concept on Demand* (GreCond) algorithm. Recent work [8] has reformulated the problem using hypergraph theory, not only providing theoretical insight but also enabling new algorithmic approaches based on hypergraph transversal algorithms.

In this paper, we focus on BMF from a hybrid perspective: on the one hand, we rely on the greedy approach exemplified by GreCond and, on the other hand, we consider the approach introduced in [9], where theoretical connections between cluster analysis and concept analysis are investigated, and an algorithm is specified. Specifically, we propose an algorithm, RSF, based on the Rice-Siff algorithm with some "greedy touches": the greedy selection of concepts for the factorization is guided by a distance-based agglomerative approach. As the main drawback in exact (or FCA-based) BMF algorithms is the combinatorial explosion of candidate concepts, this approach is designed to avoid such a computationally expensive exploration. Also, as topological constraints are subtly incorporated into the algorithm, we will show that the factorization provided is more robust and coherent.

The structure of the paper is the following: Section 2 presents the basic notions of formal concept analysis and Boolean matrix factorization; in Section 3, we describe works related to this problem; Sections 4 and 5 present our main algorithm and some optimizations that can be applied to enhance its performance; finally, in Section 6, we perform a thorough comparison of RSF against other algorithms which are representatives of different BMF strategies.

2. Preliminaries

We now recall the basic notions of Formal Concept Analysis [10] and the notions underlying the decomposition of a formal context.

Definition 1. Let B and A be non-empty sets and let $R \subseteq B \times A$ be a binary relation. Then the triple (B, A, R) is called a *formal context*. The elements of B are called *objects*, the elements of A are called *attributes*, and R is called the *incidence relation*.

A formal context may be regarded as a binary object-attribute table specifying which objects are related to which attributes. To capture these relationships formally, we use the following operators.

Definition 2. Let (B, A, R) be a formal context, and let $X \subseteq B$, $Y \subseteq A$ be sets of objects and attributes, respectively. We define mappings $^{\uparrow} : \mathcal{P}(B) \to \mathcal{P}(A)$, $^{\downarrow} : \mathcal{P}(A) \to \mathcal{P}(B)$ between the powersets of objects and attributes by

$$X^{\uparrow} = \{\, y \in A \mid (\forall x \in X)\,(x, y) \in R \,\},$$
$$Y^{\downarrow} = \{\, x \in B \mid (\forall y \in Y)\,(x, y) \in R \,\}.$$

The mappings $^{\uparrow}$ and $^{\downarrow}$ are called the concept-forming operators of the formal context (B, A, R).

The concept-forming operators provide the basis for the definition of a formal concept, as they form a Galois connection and, therefore, their compositions are actually closure operators.

Definition 3. Let (B, A, R) be a formal context with associated concept-forming operators $^{\uparrow}$, $^{\downarrow}$, and let $X \subseteq B$, $Y \subseteq A$ be sets of objects and attributes, respectively. A pair (X, Y) such that $X^{\uparrow} = Y$, $Y^{\downarrow} = X$ is called a *formal concept* of (B, A, R). In the formal concept (X, Y), the set X is called the *extent*, while the set Y is called the *intent*.

We write $\mathfrak{B}(B, A, R)$ to denote the set of all formal concepts of the formal context (B, A, R).

Definition 4. Given $(X_1, Y_1), (X_2, Y_2)$ two formal concepts, we define $(X_1, Y_1) \preceq (X_2, Y_2)$ if and only if $X_1 \subseteq X_2$. Then the partially ordered set $(\mathfrak{B}(B, A, R), \preceq)$ is called the *concept lattice* of the formal context (B, A, R).

The fundamental theorem of FCA [10] states precisely that the concept lattice $\mathfrak{B}(B, A, R)$, equipped with the $\preceq$ order, is actually a complete lattice, whose supremum and infimum operators are given by:

$$\bigvee_{i \in I} (X_i, Y_i) := \left(\left(\bigcup_{i \in I} X_i \right)^{\uparrow\downarrow}, \bigcap_{i \in I} Y_i \right) = \left(\left(\bigcap_{i \in I} Y_i \right)^{\downarrow}, \bigcap_{i \in I} Y_i \right)$$
$$\bigwedge_{i \in I} (X_i, Y_i) := \left(\bigcap X_i, \left(\bigcup_{i \in I} Y_i \right)^{\downarrow\uparrow} \right) = \left(\bigcap_{i \in I} X_i, \left(\bigcap_{i \in I} X_i \right)^{\uparrow} \right)$$

where the second equalities in these expressions are due to the properties of the Galois connection $(^{\uparrow}, {}^{\downarrow})$.

In the setting of Formal Concept Analysis, the decomposition of a formal context can be interpreted as decomposing the incidence relation between objects and attributes. Let (B, A, R) be a formal context with a set of objects B, a set of attributes A, and an incidence relation $R \subseteq B \times A$. The relation R can be represented as a binary matrix $\mathbf{R}$ describing which objects possess which attributes. The goal is to identify a smaller set of latent variables, called factors, that explain the observed relationships between objects and attributes.

Formally, this means finding an integer k and two binary matrices $\mathbf{U} \in \{0,1\}^{|B|\times k}$ and $\mathbf{V} \in \{0,1\}^{k\times|A|}$ describing the relationships between objects and factors, and between factors and attributes, respectively, such that the original incidence matrix $\mathbf{R}$ can be reconstructed as a Boolean product of $\mathbf{U}$ and $\mathbf{V}$. This Boolean product is denoted by $\mathbf{U}\circ\mathbf{V}$ and defined as $(\mathbf{U}\circ\mathbf{V})_{ij} = \bigvee_{l=1}^{k}(\mathbf{U}_{il} \wedge \mathbf{V}_{lj})$ ($i \in \{1,\dots,|B|\}$, $j \in \{1,\dots,|A|\}$), where the operators $\vee$ and $\wedge$ represent the maximum and minimum operators in the Boolean lattice $\{0,1\}$. Thus, a decomposition of a formal context can be viewed as a Boolean matrix factorization of its incidence matrix.

A very important contribution in this field was presented in [7], where Bělohlávek and Vychodil showed that formal concepts provide optimal and universal factors for Boolean matrix decomposition. This connection between formal concepts and matrix factorization is formally established by mapping any set of formal concepts to a pair of binary matrices, as defined below.

Definition 5. Given a set of formal concepts $\mathcal{F} = \{(X_1,Y_1),\dots,(X_k,Y_k)\} \subseteq \mathfrak{B}(B,A,R)$, we define the *object-factor* matrix $\mathbf{U}_{\mathcal{F}} \in \{0,1\}^{|B|\times k}$ and the *factor-attribute* matrix $\mathbf{V}_{\mathcal{F}} \in \{0,1\}^{k\times|A|}$ as follows:

$$\mathbf{U}_{\mathcal{F}}(i,j) = \begin{cases} 1 & \text{if } b_i \in X_j \\ 0 & \text{otherwise} \end{cases}, \quad \mathbf{V}_{\mathcal{F}}(i,j) = \begin{cases} 1 & \text{if } a_j \in Y_i \\ 0 & \text{otherwise} \end{cases},$$

where we consider $B = \{b_1,\dots,b_{|B|}\}$ and $A = \{a_1,\dots,a_{|A|}\}$.

In this framework, the matrix $\mathbf{U}_{\mathcal{F}}$ indicates the membership of objects in the extents of the selected formal concepts in $\mathcal{F}$, whilst the matrix $\mathbf{V}_{\mathcal{F}}$ represents the relationship between those concepts and the attributes via their intents.

The following theorem establishes the existence of a factorization of the incidence matrix using formal concepts as factors (universality).

Theorem 1 ([7]). *Let (B,A,R) be a formal context and let $\mathbf{R} \in \{0,1\}^{|B|\times|A|}$ be its incidence matrix. Then there exists a set $\mathcal{F} \subseteq \mathfrak{B}(B,A,R)$ of formal concepts such that $\mathbf{R} = \mathbf{U}_{\mathcal{F}}\circ\mathbf{V}_{\mathcal{F}}$.*

The following theorem shows that any Boolean factorization can be replaced by a factorization whose factors are formal concepts and whose number does not exceed the number of original factors (optimality):

Theorem 2 ([7]). *Let $\mathbf{R} = \mathbf{U}\circ\mathbf{V}$ be a Boolean factorization of the incidence matrix $\mathbf{R} \in \{0,1\}^{|B|\times|A|}$, where $\mathbf{U} \in \{0,1\}^{|B|\times k}$ and $\mathbf{V} \in \{0,1\}^{k\times|A|}$. Then there exists a set $\mathcal{F} \subseteq \mathfrak{B}(B,A,R)$ of formal concepts of the context (B,A,R) such that $|\mathcal{F}| \leq k$ and for the matrices $\mathbf{U}_{\mathcal{F}}$ and $\mathbf{V}_{\mathcal{F}}$ induced by $\mathcal{F}$ we have $\mathbf{R} = \mathbf{U}_{\mathcal{F}}\circ\mathbf{V}_{\mathcal{F}}$.*

These theoretical foundations establish that the problem of finding a minimal Boolean factorization is equivalent to finding a minimal set of formal concepts that covers all entries of the incidence matrix. This connection effectively bridges the gap between matrix decomposition and the combinatorial problem of set covering within the concept lattice, providing a robust justification for the use of FCA-based heuristics in practical BMF applications.

3. Related work on Boolean matrix factorization

Methods that aim to represent a binary matrix $\mathbf{R}$ as a Boolean product $\mathbf{U}\circ\mathbf{V}$ have been studied for several decades. Early research on such decompositions can be found in several foundational works e.g., [11,12], which also established important complexity results showing that many optimization problems related to binary matrix decomposition are computationally hard. Classical results on Boolean matrices, relations, and their algebraic properties are discussed in the monograph by Kim [13].

Modern algorithmic approaches to Boolean Matrix Factorization can be broadly categorized along two major axes: those rooted in Formal Concept Analysis (FCA) and those emerging from data mining paradigms. Given that attaining an optimal factorization is a computationally hard task (explicitly, the problem of finding a representation with minimal rank is NP-complete [12]), the field has evolved around several foundational heuristic pillars, namely ASSO, GreConD, PaNDa, GreEss, and Hyper+ algorithms, which together define the boundary of available BMF strategies. These methods navigate the critical trade-offs between reconstruction accuracy and model complexity using paradigms ranging from association rule mining to information theory (incorporating metrics such as Minimum Description Length (MDL) [14]), and are typically tailored to either DBP [1] or the *Approximate Factorization Problem* (AFP) [15].

Recent developments have focused on improving the scalability of these foundations. For instance, Trnecka and Trneckova [16] proposed a data reduction technique specifically for FCA-based BMF algorithms, which aims to prune the input relation while preserving the set of potential factors, thus significantly alleviating the computational load of concept extraction. Furthermore, the interest in scalability has led to the exploration of new data structures and optimization techniques. Kolomvakis et al. [4] recently introduced advanced greedy and local-search heuristics supported by highly efficient C++ Boolean vector implementations, emphasizing the importance of bitwise optimizations for handling large-scale sparse datasets.

One of the most foundational algorithms in the data mining community is ASSO, proposed by Miettinen et al. [1]. Primarily designed for the DBP, ASSO utilizes association rule mining to discover rectangular structures $X \times Y$ (inside the input matrix) representing highly associated groups of objects and attributes. It computes a confidence-based matrix where a *1* entry indicates that an association rule between two attributes meets a user-refined threshold. While ASSO is a cornerstone benchmark, its greedy selection mechanism is inherently allowed to *overcover* (reconstructing a 1 where the original matrix had a 0). This overcovering property helps in noisy environments but limits its use in applications requiring exact *from-below* decompositions.

From a purely order-theoretical perspective, the GreConD algorithm (Greedy Concept on Demand) [7] provides a rigorous FCA-based framework. GreConD identifies formal concepts as maximal rectangles by performing a greedy combinatorial sweep of the

attribute space. Unlike ASSO, it strictly ensures a *from-below* approximation, making it ideal for exact decompositions. However, GreCond's performance is intrinsically linked to the closure operator $\cdot^{\downarrow\uparrow}$, which involves repeated vector intersections. For large topologies, these exhaustive closures become the primary computational bottleneck, although efficient implementation strategies to overcome it exist [17]. Furthermore, while efficient, GreCond can converge to suboptimal solutions in specific edge cases, such as contranominal scales [18], where the greedy choice fails to capture the global optimal hierarchy.

To handle noise and structural redundancy more robustly, the PaNDa algorithm [14,19] shifts the focus toward information theory. Leveraging the Minimum Description Length (MDL) principle, PaNDa sequentially extracts rank-1 components (tiles) that most significantly reduce the description complexity of the data. PaNDa and its successor, PaNDa+, are particularly effective at identifying significant patterns in noisy datasets where rigid formal boundaries may be obscured. The greedy addition of components in PaNDa+ terminates naturally when further factors no longer minimize the total error, often resulting in a highly compact basis.

Finally, the GreEss algorithm [15] represents a theoretical refinement of the greedy strategy. By identifying the *essential parts* (or essential elements) of the incidence relation, GreEss focuses the factor search on mandatory rectangles that must be present in any exact decomposition. This insight allows for a deeper geometric understanding of BMF, often yielding more compact factorizations than the standard GreCond by prioritizing these critical topological anchors.

Furthermore, the Hyper and Hyper+ algorithms, introduced by Xiang et al. [20], present a two-phase framework that bridges database tiling and cardinality-based data summarization. The first phase, Hyper, extracts candidate closed patterns (which correspond to formal concepts in FCA terminology) from frequent closed itemsets satisfying a minimum support threshold. In the second phase, Hyper+ selects a subset of these candidates to construct a factorization that minimizes the representation cost (defined as the sum of row and column cardinalities of the factors, $|T| + |I|$) under a given false positive error budget β. While Hyper+ is primarily known as an approximate method that allows noisy reconstruction, it can also function as a logically exact solver when the error tolerance is set to zero ($\beta = 0$), making it a competitive baseline for comparing the structural parsimony of concept-based factorization methodologies.

In the context of this diverse landscape, our proposed Rice-Siff-based algorithms (named RSF and RSF-ES) introduce a hybrid synthesis. While our proposal shares with Hyper+ the core philosophy of merging concepts to discover compact factorizations, their guiding principles are fundamentally distinct. Whereas Hyper+ guides its merging decisions through a purely quantitative optimization of the cardinality-based representation cost (greedily maximizing the reduction in factor sizes $|T| + |I|$), our Rice-Siff-based methods adopt a structural and topological perspective. By replacing the exhaustive combinatorial search tree of GreCond with a semi-metric-driven agglomerative clustering derived from Rice and Siff [9], RSF navigates the search space through structurally-targeted mergers in the concept lattice. This shift from combinatorial *sweeping* to metric *targeting* allows RSF to preserve the mathematical integrity of formal concepts and the underlying order-theoretic structure, while mitigating the high computational cost of exhaustive attribute closures. A topology-driven optimization is formulated in RSF-ES to accelerate this process. By dynamically pruning disjoint and redundant merges, this variant avoids costly closure computations and delivers factorizations significantly faster.

4. Greedy BMF via Rice-Siff

The fundamental bottleneck in exact Boolean Matrix Factorization lies in the exhaustive combinatorial search for an optimal sequence of formal concepts. Methods such as GreCond iteratively compute formal concepts to cover the residual boolean incidence relation. However, generating all possible concepts or searching through the closure space of attribute subsets is computationally prohibitive for highly dimensional or extremely dense matrices.

To circumvent this limitation, we formalize the BMF problem as an approximation of the *Set Cover* problem. Let $R \subseteq B \times A$ be the incidence relation of the input context. The base space to be covered is precisely the set of pairs in R. A set of covering subsets can be constructed from the formal concepts $\mathfrak{B}(B, A, R)$, where each concept (X, Y) covers the rectangle $X \times Y \subseteq R$. Therefore, finding an exact Boolean factorization reduces to finding a minimal family of formal concepts whose union over their incident matrices covers R [7].

It is a well-known result that the Set Cover problem is strongly NP-hard, yet it admits a natural greedy approximation algorithm with a logarithmic approximation ratio [21]. In every step, the greedy approach selects the subset (in our case, represented by a formal concept) that maximizes the coverage of the currently *uncovered* elements of the base set. The Greedy algorithm approximates the optimal covering such that the ratio of the number of selected concepts to the optimal number is bounded by the harmonic number $H(d) = \sum_{i=1}^{d} \frac{1}{i} \leq \ln d + 1$, where d is the maximum coverage of any subset, in our case $d = |R|$. This implies that the greedy algorithm guarantees a number of factors with an approximation ratio of $O(\ln(|R|))$ relative to the optimal number of factors.

While the greedy strategy offers a polynomial-time approximation, selecting the *best* covering formal concept out of a potentially exponential concept lattice remains a severe hurdle. To resolve this, we integrate a modified version of the agglomerative *Rice-Siff* clustering algorithm [9] directly into the greedy selection phase.

Without loss of generality, we formulate the algorithms in this section starting from object concepts, directly inheriting the original setting of Rice and Siff's clustering method [9]. However, due to the complete order-theoretic symmetry of formal contexts and the Boolean matrix factorization problem, the entire procedure is strictly dual. The algorithm is completely agnostic to whether it is initialized with object concepts or attribute concepts. In practice, to minimize the size of the initial candidate pool and the subsequent computational overhead of the agglomerative process, the algorithm should always be initialized using the smaller of the two dimensions of the formal context. That is, starting from object concepts if $|B| \leq |A|$, and from attribute concepts if $|A| < |B|$ (which is mathematically equivalent to executing the algorithm on the transposed context (A, B, R^{-1})).

The classical Rice-Siff algorithm constructs a concept hierarchy by iteratively merging (by computing their supremum) pairs of concepts that exhibit the minimum distance according to a specialized metric, typically starting from the object concepts $C_1 = \{(\{b\}^{\uparrow\downarrow}, \{b\}^{\uparrow}) \mid b \in B\}$. The traditional metric used by the algorithm is defined as $d(O_1, O_2) = 1 - \frac{|X_1 \cap X_2|}{|X_1 \cup X_2|}$ for two candidate concepts $O_1 = (X_1, Y_1)$ and $O_2 = (X_2, Y_2)$ (where $X_1, X_2 \subseteq B$ denote their respective object extents), representing the standard Jaccard distance over their object coverages. This similarity is evaluated over the static set of elements in C_1. However, this standard distance function does not account for the dynamic state of the factorization.

We introduce a state-dependent semi-metric ρ_{un}, parameterized by the continuously shrinking set of uncovered relations $R_{un} \subseteq R$.

Definition 6. Let R_{un} denote the subset of incidence pairs in R strictly not yet covered by any previously selected factor. Given two formal concepts (X_1, Y_1) and (X_2, Y_2), their generalized distance relative to the uncovered relation is defined as:

$$\rho_{un}((X_1, Y_1), (X_2, Y_2)) = 1 - \frac{|(X_1 \times Y_1) \cap (X_2 \times Y_2) \cap R_{un}|}{|(X_1 \times Y_1) \cup (X_2 \times Y_2)|}.$$

The function ρ_{un} serves as a state-aware adaptation of the classical Rice-Siff metric [9]. It should be noted that while ρ_{un} satisfies non-negativity and symmetry, it represents a semi-metric in a measure-theoretic sense as it does not strictly satisfy the triangle inequality. While the original metric captures structural similarity between formal concepts, it remains indifferent to the dynamic progression of the factorization. In the context of BMF, the primary advantage of ρ_{un} lies in its parameterization by R_{un}, which explicitly filters intersections through the residual relation. Furthermore, while the classical metric $d(O_1, O_2)$ compares concepts solely based on their one-dimensional object extents, ρ_{un} evaluates the distance between their full two-dimensional rectangles $X_i \times Y_i \subseteq B \times A$ (i.e., the formal concepts themselves as two-dimensional structures). This ensures that the similarity is computed directly over their coverage areas, perfectly aligning the agglomeration with the discrete rectangle-covering objective of BMF. By penalizing concept unions that overlap in generic regions but bypass critical uncovered topologies, this state-dependent metric guides the agglomerative process to gravitate seamlessly towards concepts that maximize the coverage of the remaining matrix. Consequently, ρ_{un} naturally prioritizes merges that explain new incidence pairs, effectively aligning the clustering logic with the objective of the greedy set-cover approximation.

The mapping ρ_{un} is more than a simple clustering metric; it acts as a topological guide within the bipartite graph $G = (B \dot\cup A, R_{un})$ induced by the currently uncovered incidence cells (where $B \dot\cup A$ denotes the disjoint union of the object and attribute sets). Specifically, the numerator $|(X_1 \times Y_1) \cap (X_2 \times Y_2) \cap R_{un}|$ measures the degree of mutual incidence shared by two concept candidates relative to the residual context. By maximizing this intersection, the algorithm identifies and prioritizes the merging of concepts that share a common core of uncovered cells, while the denominator normalizes the distance to prevent the formation of over-generalized, sparse rectangles. In this way, ρ_{un} evaluates the local density of potential merges, guiding the search toward structurally coherent and tightly-packed Boolean factors.

A critical boundary in this search process occurs when $\rho_{un} = 1.0$, which denotes a total absence of shared coverage in the residual relation. In graph-theoretic terms, this condition signifies that the incidence regions covered by (X_1, Y_1) and (X_2, Y_2) belong to disjoint connected components within the bipartite graph of R_{un}. Merging concepts across such a topological cut provides no structural gain in residual coverage and would necessarily restrict the formal intent to an intersected subset of attributes, potentially bypassing denser local patterns. This fact will allow us to propose, in the next section, some optimizations to the basic algorithm presented here.

Consequently, ρ_{un} not only quantifies similarity but also reflects the connectivity of the matrix residue, ensuring that the greedy search remains anchored to the most cohesive and dense regions of R_{un} at each iteration.

Let us present the idea of coverage for a concept, which will guide the factor selection phase in our proposal.

Definition 7. The *(residual) coverage weight* of a formal concept (X, Y) relative to the residual relation R_{un} is defined as the number of incidence pairs it covers in R_{un}:

$$w(X, Y) = |(X \times Y) \cap R_{un}|.$$

The next result provides a safe mechanism to substantially reduce the computational space of the hierarchical agglomeration. It guarantees that merging concepts with disjoint intents is mathematically sterile.

Lemma 1. *If the intents Y_1, Y_2 of two formal concepts (X_1, Y_1) and (X_2, Y_2) are disjoint ($Y_1 \cap Y_2 = \emptyset$), then their supremum $(X_1, Y_1) \vee (X_2, Y_2)$ covers no elements in the residual relation R_{un}.*

Proof. The supremum of two concepts is defined as $((X_1 \cup X_2)^{\uparrow\downarrow}, Y_1 \cap Y_2)$. If $Y_1 \cap Y_2 = \emptyset$, the resulting intent is empty. Thus, the bounding rectangle of the supremum becomes $(B, \emptyset)$, which has an area of 0 and cannot cover any element in R_{un}. □

Consequently, the agglomeration algorithm can safely filter out pairs where $Y_i \cap Y_j = \emptyset$ directly during the structural evaluation candidates, explicitly skipping them as they contribute no value to the set cover. Similarly, if two candidate concepts have disjoint extents ($X_i \cap X_j = \emptyset$), their coverage intersection is empty, which natively yields a distance of $\rho_{un} = 1.0$. Thus, the computation of ρ_{un} can be bypassed for these orthogonal pairs, directly assigning them the maximum distance. In turn, the agglomeration loop is terminated only when no valid disjoint pairs remain, as any further generalizations would invariably yield the universal empty-intent concept $(B, \emptyset)$. This technical optimization is explicitly incorporated into the design of the Rice-Siff Factorization (RSF) algorithm, presented in Algorithm 1.

Implementation note: While Algorithm 1 (and subsequently Algorithm 2) are described using object-derived concepts for consistency with the theoretical presentation, the implementation automatically detects the optimal orientation. If $|A| < |B|$, a dual version of the algorithm is executed by transposing the input context, thereby starting from attribute-derived concepts $\{(\{a\}^{\downarrow}, \{a\}^{\downarrow\uparrow}) \mid a \in A\}$, which drastically reduces the initial candidate pool size from $|B|$ to $|A|$.

Algorithm 1: Rice-Siff factorization (RSF).

Input: A formal context (B, A, R) with $R \subseteq B \times A$
Output: Factor set $\mathcal{F}$

1 $\mathcal{F} \leftarrow \emptyset$
2 $R_{un} \leftarrow R$
3 **while** $R_{un} \neq \emptyset$ **do**
4 $C_1 \leftarrow \{(\{x\}^{\uparrow\downarrow}, \{x\}^{\uparrow}) \mid x \in B\}$
5 $C_2 \leftarrow C_1$
6 **while** $|C_1| > 1$ **do**
7 $P \leftarrow \{(O_i, O_j) \in C_1 \times C_1 \mid Y_i \cap Y_j \neq \emptyset\}$
8 **if** $P = \emptyset$ **then break**
9 Find $(O_i, O_j) \in P$ minimizing $\rho_{un}(O_i, O_j)$
10 Remove O_i, O_j from C_1
11 $O_{new} \leftarrow O_i \vee O_j$
12 Insert O_{new} into C_1 and C_2
13 Select $(X, Y) \in C_2$ maximizing $w(X, Y)$
14 $\mathcal{F} \leftarrow \mathcal{F} \cup \{(X, Y)\}$
15 $R_{un} \leftarrow R_{un} \setminus (X \times Y)$
16 **return** $\mathcal{F}$

The following results establish the theoretical properties of the RSF algorithm, focusing on its termination, correctness, and computational complexity.

Theorem 3. *Assuming B and A are finite sets, given an input incidence relation $R \subseteq B \times A$, the RSF algorithm (Algorithm 1) terminates in a finite number of steps and computes an exact Boolean factorization, i.e., $\bigcup_{(X,Y)\in F}(X \times Y) = R$.*

Proof. At each global iteration, provided $R_{un} \neq \emptyset$ (line 3), the initialization step (line 4) places object-derived formal concepts into C_1 and C_2. Because the set R is finite, there exists at least one formal concept covering at least one element in R_{un}. The selection step (line 13) chooses a concept (X, Y) from C_2 (the set of all agglomerated candidates) that maximizes residual coverage. Since C_2 includes at least the object concept for any x such that $(x, a) \in R_{un}$, the algorithm strictly chooses an (X, Y) such that $|(X \times Y) \cap R_{un}| \geq 1$. Consequently, the cardinality of R_{un} strictly decreases in every global iteration (line 15). As R is finite, R_{un} becomes the empty set in at most $|R|$ iterations, terminating the loop. □

Theorem 4. *Assuming B and A are finite sets, given an input incidence relation $R \subseteq B \times A$, each iteration of the greedy loop in Algorithm 1 requires at most $O(|B|^3|A|)$ time.*

Proof. Let $n = |B|$ and $m = |A|$. The RSF algorithm consists of a greedy loop (line 3) that repeats until all incidence pairs $|R|$ are covered. In each iteration, C_1 is initialized with n object concepts (line 4). The inner agglomeration loop (line 6) performs $n - 1$ merges. In each merge step, identifying the minimum ρ_{un} requires evaluating the distance between all pairs in C_1. Initially, this involves $O(n^2)$ distance evaluations. The complexity of calculating ρ_{un} involves set intersections and unions over the incidence matrices, which takes $O(nm)$ bitwise operations. Thus, each inner step is $O(n^2 \cdot nm)$. Over n merges, this yields $O(n^3 m)$. Factoring in the selection scan in C_2 (line 11), which is $O(n \cdot nm)$, the total complexity per greedy iteration is $O(n^3 m)$, that is, $O(|B|^3|A|)$. □

By strictly enforcing the dimensional optimization (transposing the context when $|A| < |B|$), the initial candidate pool is always bounded by the smaller dimension, reducing the time complexity per greedy loop to $O(\min(|B|, |A|)^3 \cdot \max(|B|, |A|))$. This optimization guarantees that the cubic bottleneck of the agglomerative clustering is always governed by the smaller dimension of the dataset.

Regarding the approximation quality, RSF selects a formal concept (X, Y) at each step that maximizes residual coverage relative to R_{un} within the structural candidate set C_2. Since C_2 includes (successive agglomerations of) all original object concepts, the algorithm selects a candidate at least as good as the best object concept. This preserves the greedy selection property for the Set Cover problem over the base set R. Following Chvátal's result [21], the number of factors $|\mathcal{F}|$ satisfies $|\mathcal{F}| \leq H(|R|) \cdot k_{OPT} \approx \ln(|R|) \cdot k_{OPT}$, where k_{OPT} is the size of an optimal factorization. Thus, while the selection search is heuristic, the structure of the selection mechanism preserves the greedy footprint, ensuring the $O(\ln(|R|))$ approximation bound established for the discrete set-cover optimization.

Optimality on formal scales

Beyond the logarithmic bound, the RSF algorithm exhibits exact optimality for specific structural classes of matrices commonly found in conceptual scaling. We characterize these context structures following the foundations established by Ganter and Wille [10].

Proposition 1. *For a formal context (B, A, R) of size $|B| = n$ isomorphic to a standard nominal or ordinal scale, the number of factors $|\mathcal{F}|$ extracted by the RSF algorithm is exactly n.*

Table 1
Demonstrative formal context (B, A, R). Crosses (×) denote incidence relation.

	a	b	c	d
1	×	×	×	
2	×	×		
3	×		×	×
4			×	×

Proof. Let R_{un} denote the residual incidence at each step of Algorithm 1, initialized as $R_{un} = R$. The algorithm extracts factors iteratively by selecting $(X, Y) \in C_2$ that maximizes residual coverage $w(X, Y)$. We analyze the trajectory for each scale class:

1. Nominal scales $\mathbb{N}_n = (B, B, =)$.
In this case, $bRa \iff b = a$, i.e., R can be represented as the identity matrix of dimension n.

At the beginning of the first iteration, C_1 and C_2 contain exactly n object concepts of the form $O_b = (\{b\}, \{b\})$, $b \in B$. Since the sets $\{b\}$ are disjoint, for any $b \neq b'$, the intersection of intents is empty ($\{b\} \cap \{b'\} = \emptyset$). Following Lemma 1, the early stopping condition in line 8 is met immediately, so $C_2 = C_1$. In this first iteration, RSF selects one unit concept from C_2, say O_{b_1}, and removes the corresponding crosses from R_{un}.

In each subsequent iteration, the remaining object concepts are still disjoint, so the early stopping condition is met again. Thus, RSF selects one remaining unit concept from C_2 in each iteration. After n steps, $R_{un} = \emptyset$ and $|\mathcal{F}| = n$, since $\mathcal{F}$ is the union of the n disjoint object concepts.

2. (One-dimensional) Ordinal scales $\mathbb{O}_n = (B, B, \leq)$.
For simplicity, let us denote $B = \{1, \ldots, n\}$.

At the beginning of the first iteration, C_1 consists of n nested object concepts $O_b = (\{1 \ldots b\}, \{b \ldots n\})$, $b \in B$. Since C_1 is a chain under the concept ordering, the join $O_i \vee O_j$ for $i < j$ is simply O_j. Thus, the agglomerative pool C_2 is not modified during the execution of the algorithm.

Hence, in each step, the algorithm selects the object concept $O_b \in C_2$ that covers the maximum number of remaining crosses in R_{un}. This iterative process consumes the ladder-like structure of $\mathbb{O}_n$ in exactly n selections, making $\mathcal{F}$ the set of object concepts and therefore $|\mathcal{F}| = n$. □

Corollary 1. *The RSF algorithm is optimal, i.e., the factor set $\mathcal{F}$ satisfies that $|\mathcal{F}|$ coincides with the Boolean rank k_{OPT}, for nominal and ordinal scales.*

Proof. By Proposition 1, $|\mathcal{F}| = n$ for these scales. Since the Boolean rank of these specific structures is known to be exactly n [7,10], RSF achieves minimal factorization. □

Remark 1. For contranominal scales $\mathbb{N}_n^c = (B, B, \neq)$, the Boolean rank is $k_{OPT} = \min\{d \mid n \leq \binom{d}{\lfloor d/2 \rfloor}\}$, which is logarithmic in n, following Sperner's theorem [22]. In these contexts, RSF invariably extracts n factors, as the local coverage of object concepts $O_b = (B \setminus \{b\}, \{b\})$ (size $n - 1$) dominates the greedy selection, masking the more efficient combinatorial overlaps.

Demonstrative trace comparison

To explicitly illustrate the structural advantage of the RSF distance metric agglomeration over classical exact-greedy combinatorial sweeping [7], consider the dense, intersecting formal context (B, A, R) evaluated over a set of objects $B = \{1, 2, 3, 4\}$ and attributes $A = \{a, b, c, d\}$, illustrated in Table 1.

We provide a detailed comparison of the algorithmic traces of GreCond and RSF. This demonstrative case highlights how ρ_{un} guides the factorization process efficiently compared to exhaustive closures.

GreCond trace: The initialization begins with R_{un} containing all 10 incident cells. The algorithm then explores the search tree of attribute combinations to identify maximal coverage. For clarity, we use arrow notation for operators.

Factor 1 Extraction: GreCond evaluates all singleton attribute closures $\{a\}^{\downarrow\uparrow}$, $\{b\}^{\downarrow\uparrow}$, $\{c\}^{\downarrow\uparrow}$, $\{d\}^{\downarrow\uparrow}$, yielding coverage counts of 3, 4, 3, and 4 respectively. Selecting $(\{b\}^{\downarrow}, \{b\}^{\downarrow\uparrow}) = (\{1, 2\}, \{a, b\})$, the algorithm attempts further expansion. However, attempting to expand the concept with $(\{b, c\}^{\downarrow}, \{b, c\}^{\downarrow\uparrow}) = (\{1\}, \{a, b, c\})$ and $(\{b, d\}^{\downarrow}, \{b, d\}^{\downarrow\uparrow}) = (\emptyset, A)$ fails to increase coverage over R_{un}. Consequently, the concept $(\{1, 2\}, \{a, b\})$ is selected, reducing $|R_{un}|$ to 6 cells. The residual relation R_{un} after this step is shown below.

R_{un}	a	b	c	d
1			×	
2				
3	×		×	×
4			×	×

Factor 2 Extraction: A second sweep covers the residual cells. Closures for $\{c\}$ and $\{d\}$ relative to the new residual yield $(\{1, 3, 4\}, \{c\})$ with a weight of 3 and $(\{3, 4\}, \{c, d\})$ with 4. The latter is selected, leaving only two incident fragments: $\{(1, c), (3, a)\}$.

Factor 3 Extraction: Resolving the final residual requires a third sweep. While $\{a\}^{\downarrow\uparrow}$ and $\{c\}^{\downarrow\uparrow}$ each cover a single cell, the expansion $(\{a,c\}^{\downarrow},\{a,c\}^{\downarrow\uparrow}) = (\{1,3\},\{a,c\})$ successfully matches the remaining two cells, completing the factorization.

Completing the factorization, GreCond achieves an exact decomposition with 3 factors: $F = \{(\{1,2\},\{a,b\}),(\{3,4\},\{c,d\}),(\{1,3\},\{a,c\})\}$. In total, the exhaustive greedy search required *22 distinct intent closures* to navigate the attribute hierarchy.

Conversely, RSF avoids object-wise exhaustive closures. By leveraging an object-derived initial population C_1 (Algorithm 1, line 3), it steers the search through directed agglomeration. For clarity, coverage weights in C_1 are left implicit.

Factor 1 Extraction: The algorithm initializes the sets C_1 and C_2 with the object concepts:

$$C_{1,2} = \{(\{1\},\{a,b,c\}),(\{1,2\},\{a,b\}),(\{3\},\{a,c,d\}),(\{3,4\},\{c,d\})\}$$

The metric ρ_{un} evaluates all pairwise distances in C_1. For brevity in the distance matrices, let O_i denote the initial object concept $(\{i\}^{\uparrow\downarrow},\{i\}^{\uparrow})$ above. Because the formal concepts driven by objects 1 and 2, as well as 3 and 4, represent adjacent topologies, their union geometries heavily intersect R_{un}. Conversely, all other disjoint pairwise combinations fail to intersect over R_{un}. The resulting distance matrix is:

ρ_{un}	O_1	O_2	O_3	O_4
O_1	–	0.6	1.0	1.0
O_2		–	1.0	×
O_3			–	0.6
O_4				–

where only elements in the upper-triangular part are shown, due to the symmetry of the metric, and × denotes that the intents are disjoint.

Targeting the first minimum, RSF agglomerates the geometries of objects 1 and 2 into their bounding box $O_{new,1} = (\{1,2\},\{a,b\})$. It removes these two components from C_1 and strictly inserts $O_{new,1} = O_2$ into both pools concurrently:

$$C_1 = \{O_3, O_4, O_2\} \qquad C_2 = \{O_1, O_2, O_3, O_4\}$$

Because $|C_1| > 1$, the agglomeration loop repeats. Re-evaluating the remaining pairs in C_1, the semi-metric registers 0.6 between the initial concepts for 3 and 4, since interpolating either with the newly synthesized $O_{new,1} = O_2$ yields 1.0:

ρ_{un}	O_3	O_4	O_2
O_3	–	0.6	1.0
O_4		–	×
O_2			–

They merge into another node $O_{new,2} = (\{3,4\},\{c,d\}) = O_4$. The algorithm updates the structural sets:

$$C_1 = \{O_2, O_4\} \qquad C_2 = \{O_1, O_2, O_3, O_4\}$$

The final loop iteration attempts to evaluate the last two disjoint macroscopic objects in C_1: O_2 with intent $\{a,b\}$ and O_4 with intent $\{c,d\}$. Because their intents are completely disjoint ($\{a,b\} \cap \{c,d\} = \emptyset$), Lemma 1 is triggered and the agglomeration branch immediately terminates.

The selection scan then evaluates C_2 for the synthesized factor that maximizes total incidence with R_{un}, naturally extracting the dominant structure $(\{1,2\},\{a,b\})$. This establishes the first exact optimal formal concept without searching attribute subsets, successfully shedding its 4 matrix cells from R_{un}.

Factor 2 Extraction: With R_{un} geometrically updated, RSF resets C_1 to the full initial basis. Even though object 2's geometry is entirely depleted ($w(O_2) = 0$), it is not filtered (as algorithmic pruning relies only on intent orthogonality). Thus, the new initial state is:

$$C_1 = \{O_1, O_2, O_3, O_4\}$$

Evaluating the distances over the updated R_{un} topology, the pair O_3 and O_4 still form a robust symmetric intersection covering $(3,c)$ and $(3,d)$, naturally yielding a strict minimum. All other non-disjoint pairs (e.g., (O_1,O_2) and (O_1,O_3)) strictly lack intersecting extensions, causing their purely geometric overlap in R_{un} to be identically zero ($\rho_{un} = 1.0$). The pair (O_2,O_4) is safely filtered by Lemma 1 and skipped:

ρ_{un}	O_1	O_2	O_3	O_4
O_1	–	1.0	1.0	1.0
O_2		–	1.0	×
O_3			–	0.6
O_4				–

RSF structurally targets the minimum and collapses them into the supremum concept $O_{new,3} = O_3 \vee O_4 = (\{3,4\},\{c,d\}) = O_4$. The original elements are removed and O_4 is inserted back:

$$C_1 = \{O_1, O_2, O_4\} \qquad C_2 = \{O_1, O_2, O_3, O_4\}$$

In the next inner iteration, the valid pairs remaining in C_1 are (O_1, O_2) and (O_1, O_4), both evaluating to $\rho_{un} = 1.0$. The algorithm merges one of them, for instance $O_1 \vee O_4 = O_{new,4} = (\{1,3,4\}, \{c\})$, reducing C_1 to $\{O_2, O_{new,4}\}$ and making $C_2 = \{O_1, O_2, O_3, O_4, O_{new,4}\}$. At this point, the intents of O_2 and $O_{new,4}$ are disjoint ($\{a,b\}$ and $\{c\}$) are disjoint, triggering the orthogonality *break* and terminating the loop. The selection step then evaluates all candidates accumulated in C_2. It extracts the optimal structure $(\{3,4\}, \{c,d\})$ as it covers 4 residual elements, significantly outperforming the over-generalized $O_{new,4}$ which only covers 3. The total remaining topological cells in R_{un} is now 2.

Factor 3 Extraction: At the highly fragmented tail of the matrix, the residual R_{un} contains only two disparate incidence cells: $(1,c)$ and $(3,a)$. The algorithm again initializes C_1 with the full object concept basis $\{O_1, O_2, O_3, O_4\}$.

Evaluating the structural metric ρ_{un} across these candidates reveals that $\rho_{un}(O_1, O_3) = 0.4$, while the corresponding to the other pairs equals 2. Because their intents overlap, specifically, O_1 has intent $\{a,b,c\}$ and O_3 has $\{a,c,d\}$, sharing the attributes $\{a,c\}$, they are evaluated and inevitably merged to synthesize the generalized supremum concept $O_{new,5} = O_1 \vee O_3 = (\{1,3\}, \{a,c\})$.

This supremum geometrically bridges the two final residual fragments. It is consequently inserted into C_2 alongside the initial individual object concepts. In the subsequent greedy selection step, while the base concepts O_1 and O_3 each offer a localized residual coverage weight of exactly 1, the bridged structure yields a superior weight: $w(O_{new,5}) = |(\{1,3\} \times \{a,c\}) \cap R_{un}| = 2$. By selecting this synthesized constraint, RSF identically captures both remaining cells in a single factorization operation. Consequently, R_{un} becomes completely empty, the algorithm successfully matches GreCond's optimal 3-factor decomposition, and the Boolean coverage factorization identically resolves.

In total, RSF identifies the same factors as GreCond but minimizes expensive closures. This trace utilized only *15 total closure operations*, consisting of 10 initializations of object concepts and 5 hierarchical merges. This distinction is critical: while object initializations require full-attribute scans, the subsequent merges (simple intent intersections) and the *10 distance evaluations* (ρ_{un}) are computationally efficient, lightweight operations. Because ρ_{un} operates through rapid bitwise algebraic masking between known disjoint target concepts $X_1 \cap X_2$ evaluated exclusively over R_{un}, RSF sidesteps the computationally catastrophic recursive search tree entirely. Hence, RSF tries to solve the NP-hard density covering constraint through directed hierarchical agglomeration (expressed as the structural sets C_1 and C_2 bounded in Algorithm 1) rather than heavily exhaustive bounding guesses.

5. Algorithmic optimizations on RSF

While the RSF algorithm (Algorithm 1) comprehensively explores the hierarchy of conceptual merges to identify the optimal factor mapping the residual matrix, evaluating the exhaustive cross-product of all possible object geometries carries an intense combinatoric burden. Not all agglomerations structurally contribute to the greedy set cover objective. By mathematically analyzing the properties of incidence coverage and intent intersections, we can introduce two critical, heuristic early-stopping conditions that aggressively reduce the computational space of the candidate pools without fundamentally compromising the structural quality of the matrix reconstruction.

First, evaluating an object concept whose generating object shares no incidence with the current residual uncovers a mathematically trivial coverage limit. Rather than propagating this empty base geometry through subsequent hierarchical merges–which structurally dilutes candidate models while contributing exactly zero to the covering solution–we can aggressively prune these components from the initial pool C_1.

The theoretical relationship between the coverage weight and the ρ_{un} metric leads to two critical optimization lemmas that formalize these reductions for the hierarchical agglomeration.

Lemma 2. *If a formal concept (X,Y) has no coverage on the residual relation ($w(X,Y) = 0$), then $\rho_{un}((X,Y),(X',Y')) = 1$ for any other formal concept (X',Y'). Furthermore, if the attributes of the intent Y are empty in R_{un} (i.e., $(\{b\} \times Y) \cap R_{un} = \emptyset$ for all $b \in B$), then $w((X,Y) \vee (X',Y')) = 0$.*

Proof. The first part follows from the fact that $w(X,Y) = 0$ implies $(X \times Y) \cap R_{un} = \emptyset$. Thus, the intersection with any other concept's coverage is also empty, leading to a zero numerator in ρ_{un} and thus $\rho_{un} = 1$. For the second part, since $Y_{sup} = Y \cap Y' \subseteq Y$, if Y satisfies $(\{b\} \times Y) \cap R_{un} = \emptyset$ for all $b \in B$, then $b \times Y_{sup} \cap R_{un} = \emptyset$ for all $b \in B$. Consequently, $w(X_{sup}, Y_{sup}) = \sum_{b \in X_{sup}} |(\{b\} \times Y_{sup}) \cap R_{un}| = 0$. □

Theorem 5. *Let $b \in B$ be an object such that its object concept $O_b = (\{b\}^{\uparrow\downarrow}, \{b\}^{\uparrow}) = (\{b\}^{\uparrow\downarrow}, \{b\}^{\uparrow})$ has zero coverage weight, $w(O_b) = 0$. For any formal concept $O = (X,Y)$ containing b, there exists a corresponding formal concept $O_Z = (X_Z, Y_Z)$ generated entirely from a subset of objects $Z \subseteq X \setminus \{b\}$ with strictly positive initial weights $w((\{z\}^{\uparrow\downarrow}, \{z\}^{\uparrow})) > 0$, such that $w(O_Z) \geq w(O)$.*

Proof. Let $O = (X,Y)$ be a formal concept with $b \in X$. Since $w(O_b) = 0$, the object b has no incident attributes in R_{un}, meaning $(\{b\} \times \{b\}^{\uparrow}) \cap R_{un} = \emptyset$. Because $Y \subseteq \{b\}^{\uparrow}$ (by the properties of $b \in X = Y^{\downarrow}$), it follows that $(\{b\} \times Y) \cap R_{un} = \emptyset$. Thus, b contributes exactly zero to the residual coverage of O.

Let $Z = \{z \in X \mid w((\{z\}^{\uparrow\downarrow}, \{z\}^{\uparrow})) > 0\}$ be the subset of objects in the extent of O that possess at least one uncovered cell in R_{un}. The entire residual coverage of O must strictly originate from the objects in Z, such that:

$$w(O) = |(X \times Y) \cap R_{un}| = |(Z \times Y) \cap R_{un}|.$$

Consider the formal concept $O_Z = \bigvee_{z \in Z} (\{z\}^{\uparrow\downarrow}, \{z\}^{\uparrow}) = (X_Z, Y_Z)$. By definition of the supremum across multiple concepts, its intent is the intersection of their intents: $Y_Z = \bigcap_{z \in Z} \{z\}^{\uparrow}$. Since $Z \subseteq X = Y^{\downarrow}$, it follows that $Y \subseteq \{z\}^{\uparrow}$ for all $z \in Z$, which directly implies $Y \subseteq Y_Z$.

Because the intent of O_Z is a superset of Y, its coverage across the objects in Z is guaranteed to be equal or greater, monotonically expanding the covered incident space:

$$(Z \times Y) \cap R_{un} \subseteq (Z \times Y_Z) \cap R_{un}.$$

Furthermore, the extent $X_Z = Y_Z^{\downarrow}$ contains at least all objects in Z. Any other object $x \in X_Z \setminus Z$ must necessarily have $w((\{x\}^{\uparrow\downarrow}, \{x\}^{\uparrow})) = 0$ (otherwise it would be in Z, given that $X_Z = Y_Z^{\downarrow} \subseteq Y^{\downarrow} = X$ due to antimonotony). Therefore, these extra objects contribute identically zero additional coverage. The total weight of O_Z is exactly its coverage on Z:

$$w(O_Z) = |(X_Z \times Y_Z) \cap R_{un}| = |(Z \times Y_Z) \cap R_{un}|.$$

Combining these facts yields $w(O_Z) \geq |(Z \times Y) \cap R_{un}| = w(O)$. Because O_Z is defined uniquely by the supremum of objects in C_1 (those with $w > 0$), it is accessible in the agglomerative search space without ever initializing or merging $(\{b\}^{\uparrow\downarrow}, \{b\}^{\uparrow})$. □

Let $O_b = (\{b\}^{\uparrow\downarrow}, \{b\}^{\uparrow})$ be an object concept such that $w(O_b) = 0$. By Theorem 5, any candidate concept generated by aggregating O_b is mathematically bounded in residual coverage by another formal concept generated exclusively from objects with strictly positive initial weights. Therefore, O_b is strictly dominated, and its exclusion from the initial candidate pool C_1 perfectly preserves the exactness of the greedy search.

While this guarantees that the maximum residual coverage evaluated by the greedy selection remains mathematically bounded by the positive-weight subsets, the exclusion of zero-weight objects alters the agglomerative trajectory of the algorithm. Retaining O_b introduces sub-optimal intermediary suprema into the candidate pools C_1 and C_2, structurally modifying the available combinations in subsequent iterations. Thus, filtering zero-weight objects acts as a heuristic reduction that preserves the exactness of the greedy upper bound while yielding a distinct candidate pool.

Let us note that even among viable positive-weight topologies, merging concepts with orthogonal geometries or disjoint intents often yields a geometrically sterile supremum. As the structural distance ρ_{un} approaches and ultimately reaches 1.0, the resulting union fails to synthesize any new coverage across the disjoint boundary lines, rendering further evaluation along that specific agglomerative branch computationally redundant.

Proposition 2. *Given two concepts $O_1 = (X_1, Y_1)$ and $O_2 = (X_2, Y_2)$ satisfying that $\rho_{un}(O_1, O_2) = 1.0$, and let $O_{sup} = (X_{sup}, Y_{sup}) = O_1 \vee O_2$ be their supremum. Then,*

$$w(O_{sup}) = \left|(X_{sup} \setminus (X_1 \cap X_2)) \times Y_{sup} \cap R_{un}\right|.$$

Proof. By definition of the metric, $\rho_{un}(O_1, O_2) = 1.0$ implies that the numerator of the distance is zero:

$$((X_1 \times Y_1) \cap (X_2 \times Y_2)) \cap R_{un} = \varnothing$$

Because the intersection of two Cartesian products is the Cartesian product of their intersections, this identically means:

$$((X_1 \cap X_2) \times (Y_1 \cap Y_2)) \cap R_{un} = \varnothing$$

The supremum's intent is by definition $Y_{sup} = Y_1 \cap Y_2$. This establishes that there are strictly no incidence cells in R_{un} that simultaneously belong to both original extensions X_1 and X_2 under the shared intent Y_{sup}.

To determine the weight of the supremum, $w(O_{sup}) = |(X_{sup} \times Y_{sup}) \cap R_{un}|$, we partition its extension X_{sup} into two disjoint subsets: the intersection of the generating extensions: $X_1 \cap X_2$ and the remaining objects in the supremum: $X_{sup} \setminus (X_1 \cap X_2)$.

As proven above, the coverage contributed by the first subset, $((X_1 \cap X_2) \times Y_{sup}) \cap R_{un}$, is strictly empty. Consequently, the entire residual coverage of the supremum must originate structurally from the second subset:

$$\begin{aligned} w(O_{sup}) &= |((X_1 \cap X_2) \times Y_{sup}) \cap R_{un}| + |((X_{sup} \setminus (X_1 \cap X_2)) \times Y_{sup}) \cap R_{un}| \\ &= 0 + \left|(X_{sup} \setminus (X_1 \cap X_2)) \times Y_{sup} \cap R_{un}\right|. \end{aligned}$$

□

This mathematical decomposition exposes the geometric redundancy of agglomerating formal concepts at a structural distance of $\rho_{un} = 1.0$. The analysis strictly proves that no contiguous coverage exists across the intersection $X_1 \cap X_2$ within the residual matrix R_{un}. Therefore, the entire accumulated weight of the supremum, $w(O_{sup})$, must originate uniquely from the remaining disjoint objects $x \in X_{sup} \setminus (X_1 \cap X_2)$.

In graph-theoretic terms, the condition $\rho_{un}(O_1, O_2) = 1.0$ indicates that the incidence regions covered by O_1 and O_2 possess absolutely no geometric overlap in the current residual relation R_{un}. They effectively belong to disjoint connected components within the bipartite subgraph induced by R_{un}. The supremum O_{sup} artificially bridges these independent components merely because their full object closures happen to intersect on a reduced subset of attributes Y_{sup} in the original unconstrained context R.

Because these components are completely disconnected in R_{un}, the coverage extracted by the supremum is simply a disjoint union of isolated regional weights. However, by forcing this topological bridge, the agglomeration severely restricts the formal intent to $Y_{sup} = Y_1 \cap Y_2$. If the greedy search were instead to evaluate those isolated regions independently, it could *organically* identify concepts within those disconnected components possessing strictly larger, component-specific intents, yielding denser and tightly cohesive local covers.

Bridging disconnected components therefore offers no theoretical advantage for the set cover objective. It merely aggregates the absolute weights of isolated regions, artificially inflating the supremum's scalar score while mathematically compromising its true

geometric structure. This inflation can trick the greedy selection into prioritizing a loose, disjoint union over a truly cohesive dense factor, ultimately requiring the extraction of more factors downstream to cover the specific attributes left behind by the generalized intent.

Consequently, interpreting $\rho_{un} = 1.0$ as an early stopping mechanism acts directly as a topological regularizer. By immediately pruning agglomerative branches that attempt to fuse disconnected components, the algorithm mathematically prevents the greedy search from prioritizing artificial, non-cohesive volume over true factorial structure. This optimization significantly accelerates the factorization phase and drastically restricts the unbounded growth of the candidate pool, while structurally safeguarding the decomposition from artificial rank inflation.

Algorithm 2: RSF with early stopping (RSF-ES).

Input: A formal context (B, A, R) with $R \subseteq B \times A$
Output: Factor set $\mathcal{F}$

$\mathcal{F} \leftarrow \emptyset$
$R_{un} \leftarrow R$
while $R_{un} \neq \emptyset$ **do**
  $C_1 \leftarrow \{(\{x\}^{\uparrow\downarrow}, \{x\}^{\uparrow}) \mid x \in B, w((\{x\}^{\uparrow\downarrow}, \{x\}^{\uparrow})) > 0\}$
  $C_2 \leftarrow C_1$
  while $|C_1| > 1$ **do**
    $P \leftarrow \{(O_i, O_j) \in C_1 \times C_1 \mid Y_i \cap Y_j \neq \emptyset\}$
    if $P = \emptyset$ **then break**
    Find $(O_i, O_j) \in P$ minimizing $\rho_{un}(O_i, O_j)$
    if $\rho_{un}(O_i, O_j) = 1.0$ **then break**
    Remove O_i, O_j from C_1
    $O_{new} \leftarrow O_i \vee O_j$
    Insert O_{new} into C_1 and C_2
  Select $(X, Y) \in C_2$ maximizing $w(X, Y)$
  $\mathcal{F} \leftarrow \mathcal{F} \cup \{(X, Y)\}$
  $R_{un} \leftarrow R_{un} \setminus (X \times Y)$
return $\mathcal{F}$

Regarding correctness, the early stopping variant (RSF-ES, Algorithm 2) preserves the fundamental properties of the original RSF algorithm. RSF-ES maintains the guarantee of a perfect reconstruction of the relation R, as the termination and exactness properties remain invariant to the pruning of locally suboptimal candidates, while the reduced search space may lead to significant speedups. This is because the pruning only removes candidates O_{sup} whose residual weight $w(O_{sup})$ is bounded by the existing elements in C_2, the greedy selection step is guaranteed to find a concept maximizing the coverage at each iteration as long as one exists. Consequently, the termination proof and the exactness property remain valid.

Optimized demonstrative trace

To explicitly illustrate the efficiency tradeoff, applying the heuristic dual-pruning of RSF-ES to the previous running example (Table 1) vividly demonstrates how the algorithm cleanly bypasses redundant combinatorics while generating an identical baseline decomposition.

During the extraction of Factor 1, all initial concepts successfully bound non-zero incidence blocks in $R = R_{un}$, so the $w > 0$ filter immediately accepts the full base space $C_1 = \{O_1, O_2, O_3, O_4\}$. The unrestricted distance evaluation organically drives the metric minima to efficiently merge and extract Factor 1 $(\{\{1, 2\}, \{a, b\}\})$ uniformly mirroring the unoptimized trace.

However, the optimizations decisively alter the extraction sequence for Factor 2. When R_{un} is trimmed by the leading block of Factor 1, object 2's coverage falls identically to zero ($w(O_2) = 0$). By applying the first condition (Theorem 5), RSF-ES algorithmically filters O_2 out of the initial pool, structurally restricting C_1 to the true contributing components $\{O_1, O_3, O_4\}$.

The metric evaluates their respective geometries and identifies $\rho_{un}(O_3, O_4) = 0.6$ as a strict minimum, cleanly collapsing them into $O_{new,3} = O_4 = (\{3, 4\}, \{c, d\})$. In the next inner iteration, the updated distance between the two surviving candidates (O_1 and $O_{new,3} = O_4$) natively evaluates to the limit $\rho_{un} = 1.0$. While the unoptimized execution (basic RSF) mindlessly continued merging those concepts because of their shared attribute c, synthesizing the sterile and highly generalized structure $O_{new,4}$ (naming from RSF's trace) before terminating, RSF-ES actively detects the upper bound and immediately triggers the *break* termination. The sequence efficiently halts the agglomeration and extracts the dominant structure $(\{3, 4\}, \{c, d\})$, strictly avoiding the computational overhead required to calculate the useless union.

Finally, in the Factor 3 extraction, the matrix fragments into the completely disjoint tails $(1, c)$ and $(3, a)$. Despite their orthogonal geometric mappings, the overlap of their conceptual intents natively intersects over c, returning $\rho_{un}(O_1, O_3) = 0.4 < 1.0$. Since the metric registers safely under the absolute pruning threshold, the conditional branch allows the agglomerative process to proceed unimpeded, synthesizing the bridging constraint $O_{new,5} = (\{1, 3\}, \{a, c\})$. The selection step correctly maximizes its evaluation

Table 2
Comparison of execution traces for GreCond, RSF, and RSF-ES on the running example.

Property	GreCond	RSF	RSF-ES
Search strategy	Greedy closure exploration	Agglomerative Rice–Siff search	Agglomerative Rice–Siff with pruning
Initial candidate generation	Attribute closures	Object concepts C_1	Filtered object concepts C_1
Candidate pools	No explicit C_1, C_2	Dynamic C_1, C_2	Reduced C_1, C_2
Extracted factors	3	3	3
Closure operator calls	22 closures	15 closures	13 closures
Distance metric ρ_{un}	No	Yes	Yes
Intent orthogonality pruning	No	Yes	Yes
$\rho_{un} = 1.0$ early stopping	No	No	Yes
Zero-weight filtering	No	No	Yes
Redundant intermediate concepts	Possible	Possible	Reduced

matching identically both final residual cells, naturally leaving the matrix empty and securely concluding the exact 3-factor decomposition without ever generating sub-optimal intermediate garbage closures. In aggregate, the RSF-ES process computes 13 total closure operations, consisting of 9 initializations of object concepts and 4 hierarchical merges.

To facilitate a more intuitive comparison of the execution behaviour of GreCond, RSF, and RSF-ES on the running example, Table 2 summarizes the main characteristics of the three approaches, including the number of closure computations, the evolution of the candidate pools, and the impact of the proposed pruning strategies.

6. Experimental evaluation

This section presents a rigorous empirical evaluation of the Rice-Siff-based factorization algorithms. We assess their performance against established Boolean Matrix Factorization methods, demonstrating their efficiency in large-scale scenarios and their ability to discover parsimonious latent structures. All algorithms were integrated into a high-performance bitwise C++ engine within the `fcaR` framework [23] to ensure that execution time comparisons are purely algorithmic. Two different kinds of datasets are used in these experiments, synthetic and real-world.

Evaluated algorithms

To provide a comprehensive assessment of the proposed methodology, we compare RSF against a diverse set of Boolean Matrix Factorization algorithms, covering both exact and heuristic approaches:

- RSF: The complete Rice-Siff Factorization without early stopping. It explores the entire set of candidates generated by the agglomerative process to identify the optimal factors.
- RSF-ES: The Rice-Siff Factorization with Early Stopping. This variant utilizes the state-dependent distance ρ_{un} and applies a pruning threshold of 1.0 to skip redundant combinatoric branches, aiming for maximum efficiency without loss of exactness.
- GreCond: The *Greedy Concept on Demand* algorithm [7]. It is a standard exact BMF algorithm that greedily selects formal concepts as factors to cover the incidence relation.
- GreEss: The greedy algorithm based on essential elements [15]. This method focuses on covering the set of essential entries of the Boolean matrix to achieve a more parsimonious factorization.
- ASSO: The association rule-based heuristic for BMF [1]. Beyond the primary confidence threshold parameter $t \in \{0.5, 1.0\}$, we utilized the default positive and negative correlation penalty weights ($w^+ = 1, w^- = 1$) as recommended in the original paper.
- PaNDa+: A unified heuristic framework for mining approximate top-k binary patterns [19]. We evaluate multiple cost functions proposed by the authors, specifically JA (noise minimization) and JP (noise and pattern complexity minimization) with various penalty thresholds ρ.
- Hyper+ [20]: While the original methodology involves an exact reconstruction phase (Hyper) followed by a summarization phase (Hyper+), we focus on the results of the latter as the primary representative of this family. We enforce a zero-error constraint ($\beta = 0$) during the summarization phase, ensuring that Hyper+ operates as an exact solver. This allows for a fair comparison of structural parsimony with other exact methods like RSF-ES and GreCond. Note that the first phase of Hyper+ consists in a selection of hyperrectangles using as input a sets of attributes with a prescribed minimum support. Following the advice in the original paper, we build the initial pool of hyperrectangles as closed frequent itemsets, which, in our terms, correspond to concepts with the given minimum support. In our experiments, we found that results obtained from Hyper+ were not sensitive to the choice of the minimum support (ranging from 5 to 90% of the total number of objects). Therefore, for the experiments, Hyper+ was set the minimum support to 90% of the total number of objects, as it provides the fastest execution time while maintaining high quality results.

Performance metrics

The quality and efficiency of the factorization are assessed through a multidimensional framework that captures both reconstruction fidelity and structural parsimony:

- *Computational efficiency*: We report the *Execution Time* (in seconds) and the *Peak Memory Consumption* (in MiB).
- *Reconstruction fidelity*: For exact BMF solvers, the reconstruction $\mathbf{R} = \mathbf{U} \circ \mathbf{V}$ is guaranteed to have zero error. For approximate methods, we explicitly report the *undercoverage* (E_u, proportion of missing ones in the binary matrix) and *overcoverage* (E_o, proportion of added ones) to quantify the deviation from the original data.
- *Redundancy ratio (overlap)*: We define the *Overlap* as the proportion of redundant coverage within the decomposition. Formally, $\text{Overlap} = \frac{\sum_{i=1}^{k} \text{Area}(f_i) - \text{Area}(\cup_{i=1}^{k} f_i)}{\sum_{i=1}^{k} \text{Area}(f_i)}$, where $\text{Area}(f_i)$ is the number of cells covered by the i-th factor. This metric quantifies the degree of informational overlap between extracted factors.

Implementation details

To ensure a fair and rigorous comparison, all evaluated algorithms (GreCond, GreEss, ASSO, PaNDa+, Hyper+, and the RSF variants) have been integrated into a unified, high-performance bitwise engine developed in C++. This engine utilizes optimized bitset operations to minimize memory overhead and maximize execution speed. The underlying data structures are carefully designed to process matrix intersections and factor evaluations with minimal memory reallocation, preventing system bottlenecks during large-scale computations.

For the GreCond and GreEss baselines, we started from the original source code provided by the authors[1], to which we integrated state-of-the-art optimizations such as those described in [17]. These include advanced pruning techniques and the exclusive use of the upward operator to speed up formal concept discovery. By sharing the same core C++ data structures across all implementations, we ensure that the observed differences in execution time are purely algorithmic and not biased by implementation-specific efficiencies.

Furthermore, to guarantee a completely fair and optimal comparison, all experiments (both synthetic and real-world) were conducted taking into account the best possible dimensional choice for every algorithm. Since all datasets in our benchmark satisfy $|B| \geq |A|$ (either natively or after transposition), all solvers were executed by operating on the attribute dimension, which represents the smaller of the two dimensions. This configuration minimizes the search space and candidate pool sizes uniformly across all evaluated methods, ensuring that every algorithm runs in its most efficient orientation.

Hardware and Software Environment: All experiments were conducted on an Apple M3 system equipped with 16 GB of RAM, running macOS. The algorithms were executed within R version 4.5.0 using the `Rcpp` [24] package as the interface between the high-level experimental logic and the low-level C++ computational core.

6.1. Synthetic benchmark results

Synthetic datasets were used to evaluate scalability and robustness under controlled conditions. We generated a variety of Boolean matrices varying in dimensions (from 1000×100 to 4000×500). These matrices were constructed by using the random context generation capabilities of the `fcaR` package following a Dirichlet distribution.

The synthetic experiments highlight the trade-offs between exactness, efficiency, and model complexity. Table 3 provides a detailed breakdown of performance across different matrix dimensions for the 4000 objects scenario. These results demonstrate that while traditional greedy solvers may struggle with increasing dimensionality, RSF-ES leverages its topological pruning to maintain high efficiency without sacrificing the exactness of the reconstruction.

The benchmarking on synthetic data reveals two critical insights. First, the RSF-ES variant consistently outperforms other exact solvers in terms of execution time as the number of attributes grows, showing up to a $10\times$ speedup in the 4000×500 scenario compared to RSF. Second, the approximate solvers exhibit a wide range of performance-fidelity trade-offs. Based on these results, we identify PaNDa+ (JP, $\rho = 0.5$) and PaNDa+ (JA) as the most robust representatives of the PaNDa+ family, as they achieve the best balance between factor parsimony (k) and coverage error. Similarly, for the ASSO family, the variant with a higher confidence threshold ($t = 1.0$) demonstrates significantly higher stability and reconstruction fidelity across all synthetic scenarios. Consequently, only these selected variants are included in the subsequent real-world dataset analysis to ensure a focused and meaningful comparison.

6.2. Real-world dataset analysis

Real-world datasets were obtained from the *Factorization of relational data: benchmark datasets* repository from the University Palacký Olomouc, Czech Republic .[2] As summarized in Table 4, these datasets exhibit a wide spectrum of characteristics, ranging from very sparse (*apj*, 0.29 density) to dense (*chess*, 48.68), and cover varying dimensions and underlying formal concept complexities. To ensure methodological consistency and computational efficiency, all experiments were conducted using a *tall* matrix orientation (where $|B| \geq |A|$), transposing the original data when necessary. In this standardized format, as commented earlier, the FCA-based algorithms consistently utilize attribute concepts as the fundamental base for the factorization process. This choice directly implements the dimensional optimization rule discussed in Section 4, exploiting the fact that $|A| < |B|$ in these datasets to operate on the smaller dimension and dramatically accelerate the factorization process. This heterogeneity allows for a comprehensive stress-test of the algorithms' ability to handle different structural patterns.

[1] Available at https://github.com/martin-trnecka/matrix-factorization-algorithms.

[2] All datasets are available at http://datasets.inf.upol.cz/

Table 3
Performance comparison on synthetic datasets (4000 objects). k denotes the number of factors; Time is in seconds; Mem in MiB (= 2^{20} bytes). E_u and E_o represent under- and over-coverage errors, respectively. Results are reported as Mean ± Standard Deviation over 25 runs.

Scenario	Method	k	Time (s)	Mem (MiB)	E_u	E_o	Overlap
4000 × 100	GreConD	119.7±18.2	0.126±0.105	3.4±0.3	0.0±0.0	0.0±0.0	51.3±32.5
	GreEss	97.1±4.5	0.100±0.069	3.0±0.1	0.1±0.1	0.0±0.0	36.5±31.1
	Hyper+	97.1±2.5	0.181±0.216	6.2±0.1	0.0±0.0	0.0±0.0	7.2±9.6
	ASSO ($t = 0.5$)	76.0±14.7	0.006±0.003	2.7±0.2	9.3±4.3	12.8±5.6	41.4±32.3
	ASSO ($t = 1.0$)	86.3±7.9	0.010±0.004	2.9±0.2	0.0±0.0	9.2±5.3	36.9±21.2
	PaNDa+ (JA)	83.8±13.5	0.074±0.021	2.8±0.2	0.0±0.0	13.6±3.8	48.8±35.2
	PaNDa+ (JP)	33.6±29.7	0.035±0.033	2.1±0.5	22.4±27.4	15.5±4.6	7.8±6.6
	PaNDa+ (JP, $\rho = 0.5$)	96.2±11.3	0.082±0.023	3.0±0.2	0.0±0.0	11.4±5.4	4.0±7.4
	PaNDa+ (JP, $\rho = 1.5$)	7.5±6.0	0.010±0.008	1.7±0.1	35.0±24.4	13.5±6.2	1.5±3.3
	RSF	100.2±3.5	0.033±0.019	7.7±0.1	0.0±0.0	0.0±0.0	39.0±30.9
	RSF-ES	99.7±3.6	0.025±0.023	7.7±0.1	0.0±0.0	0.0±0.0	38.0±32.0
4000 × 250	GreConD	288.6±37.7	0.297±0.082	8.5±0.6	0.0±0.0	0.0±0.0	32.3±25.2
	GreEss	245.2±6.2	0.243±0.063	7.8±0.1	0.0±0.0	0.0±0.0	12.4±9.2
	Hyper+	244.4±6.1	0.291±0.251	15.6±0.1	0.0±0.0	0.0±0.0	0.0±0.0
	ASSO ($t = 0.5$)	163.4±74.4	0.012±0.003	6.5±1.2	17.0±3.7	9.7±3.0	14.4±11.5
	ASSO ($t = 1.0$)	232.5±3.1	0.016±0.005	7.6±0.1	0.0±0.0	1.3±0.4	1.0±0.6
	PaNDa+ (JA)	236.8±14.8	0.332±0.020	7.6±0.3	0.0±0.0	13.7±4.2	25.5±21.0
	PaNDa+ (JP)	21.1±16.3	0.032±0.022	4.2±0.3	72.2±8.9	15.9±6.1	12.3±11.1
	PaNDa+ (JP, $\rho = 0.5$)	258.3±7.4	0.363±0.014	8.0±0.1	0.2±0.1	8.5±2.8	0.4±0.5
	PaNDa+ (JP, $\rho = 1.5$)	4.7±3.7	0.010±0.006	3.9±0.1	81.8±7.4	11.0±6.6	0.4±0.6
	RSF	249.2±3.2	0.319±0.147	19.5±0.1	0.0±0.0	0.0±0.0	14.3±9.5
	RSF-ES	245.3±5.7	0.054±0.034	19.4±0.2	0.0±0.0	0.0±0.0	11.9±8.6
4000 × 500	GreConD	508.4±3.3	0.855±0.175	16.4±0.1	0.0±0.0	0.0±0.0	19.0±17.0
	GreEss	503.3±2.3	1.154±0.451	16.3±0.1	0.0±0.0	0.0±0.0	16.5±17.1
	Hyper+	498.7±1.5	0.415±0.350	32.4±0.0	0.0±0.0	0.0±0.0	0.0±0.0
	ASSO ($t = 0.5$)	445.2±14.4	0.061±0.013	15.3±0.2	10.9±8.4	0.8±0.7	30.4±16.7
	ASSO ($t = 1.0$)	497.6±1.5	0.056±0.007	16.2±0.1	0.0±0.0	0.0±0.0	6.4±11.7
	PaNDa+ (JA)	495.5±2.7	1.557±0.035	16.1±0.1	0.0±0.0	0.4±0.2	17.4±19.3
	PaNDa+ (JP)	40.6±17.1	0.134±0.052	8.3±0.3	76.6±9.9	0.5±0.6	27.2±4.1
	PaNDa+ (JP, $\rho = 0.5$)	499.3±1.6	1.575±0.024	16.2±0.0	0.1±0.0	0.2±0.1	0.2±0.3
	PaNDa+ (JP, $\rho = 1.5$)	6.9±6.4	0.029±0.022	7.8±0.1	94.2±4.4	0.0±0.0	0.0±0.2
	RSF	502.5±2.1	5.462±0.778	40.2±0.1	0.0±0.0	0.0±0.0	17.2±17.1
	RSF-ES	500.3±1.8	0.568±0.500	40.0±0.1	0.0±0.0	0.0±0.0	16.4±17.3

The comparative analysis on real-world datasets, detailed in Tables 5 and 6, underscores the significant computational advantage of the RSF-ES variant. This is particularly evident in large-scale scenarios where traditional BMF solvers encounter significant bottlenecks. A primary highlight is the performance on the *americas large* dataset (10127×3485). While GreConD and GreEss require 9.6 and 24.9 seconds respectively, RSF-ES completes the factorization in just 0.407 seconds. This represents an order-of-magnitude speedup, achieved while maintaining absolute reconstruction fidelity.

It can also be observed that, in certain scenarios, GreEss is faster that GreConD. This might seem counter-intuitive as GreConD is used internally by GreEss. The explanation follows from the fact that GreConD, as auxiliary method, is not applied to the entire context matrix. The fundamental advantage of GreEss is that the initial extraction of *essential elements*, although it might be slow, serves as a powerful spatial filter. GreEss generates candidate bounding boxes (intervals) from these essential elements and restricts the execution of the internal GreConD strictly to these reduced submatrices. Since the greedy search space in each iteration is drastically pruned, the computational time saved during the factorization step often vastly outweighs the initial overhead of computing the essential elements, resulting in faster overall execution times on particular datasets.

This efficiency is deeply rooted in the topological pruning enabled by the Rice-Siff distance threshold. By identifying and skipping merges between structurally disconnected components ($\rho_{un} = 1.0$), RSF-ES prevents the uncontrolled growth of the concept candidate pools that typically plague aglomerative search in high-dimensional or dense spaces. Furthermore, RSF-ES often discovers a more parsimonious set of factors than GreConD, approaching the structural economy of GreEss but at a fraction of the computational cost.

Beyond computational speed and parsimony, other quality metrics further contextualize the performance of RSF-ES within the current state of the art. The redundancy ratio, measured by factor overlap (Ov.), indicates that RSF-ES often provides a more independent set of factors than other exact solvers. This behavior is a direct consequence of the $\rho_{un} = 1.0$ pruning mechanism, which prevents the artificial merging of disconnected components. This effect is particularly prominent in datasets such as *americas large* or *dna*, where RSF-ES achieves overlap levels (7.3% and 20.2%, respectively) that are not only significantly lower than those of GreConD, but also surpass the redundancy levels of GreEss (34.8% and 24.2%), an algorithm specifically designed to prioritize essential, non-redundant parts. While MDL-based or heuristic methods like Hyper+ or PaNDa+ naturally exhibit very low overlap due to their approximate nature, RSF-ES manages to approach these levels while maintaining absolute reconstruction fidelity. Re-

Table 4
Summary of real-world datasets used in the experiments. $|B|$ and $|A|$ denote the number of objects and attributes; the density is presented as percentage; $|\gamma(B)|$, $|\mu(A)|$ and $|\mathfrak{B}(B, A, R)|$ represent the number of object concepts, attribute concepts and the total number of formal concepts, respectively.

Dataset	$\|B\|$	$\|A\|$	Density	$\|\gamma(B)\|$	$\|\mu(A)\|$	$\|\mathfrak{B}(B, A, R)\|$
advertisement	3279	1557	0.88	1989	763	9192
americas large	10127	3485	0.53	1354	432	36991
americas small	3477	1587	1.91	259	349	2764
apj	2044	1164	0.29	564	578	798
chess	3196	76	48.68	3196	76	930851337
customer	10961	277	1.50	5656	276	47848
dblp	6980	19	12.95	890	19	2495
dna	4590	392	1.47	1316	371	4483
domino	231	79	4.00	38	23	73
emea	3046	35	6.77	263	34	780
firewall1	709	365	12.35	86	90	317
firewall2	590	325	19.00	11	11	22
mushroom	8124	119	19.33	8124	113	238710
paleo	501	139	5.08	471	139	10225
tic-tac-toe	958	30	33.33	958	30	59505
zoo	101	28	30.48	59	26	379

Table 5
Performance metrics on real-world datasets (Part I). k denotes factors; Ov. is overlap (in %) and Mem represents the maximum RAM memory needed, measured in MiB. An asterisk indicates times lower than 0.001 seconds.

Algorithm	Time	k	Ov.	Mem	Time	k	Ov.	Mem
	advertisement				*chess*			
GreConD	2.496	766	61.4	14.1	0.174	124	80.4	1.5
GreEss	9.614	720	48.7	13.3	0.445	118	78.8	1.4
Hyper+	0.390	763	0.0	14.0	0.106	80	7.9	1.0
PaNDa+ (JP, $\rho = 0.5$)	4.501	569	3.9	10.5	0.032	48	1.2	0.6
PaNDa+ (JA)	4.898	587	52.0	10.8	0.025	35	83.6	0.4
ASSO ($t = 1.0$)	0.600	640	31.8	11.9	0.006	52	38.6	0.6
RSF	14.739	730	56.6	13.4	0.022	75	56.1	1.0
RSF-ES	1.316	732	56.8	13.5	0.022	75	56.1	0.9
	americas large				*customer*			
GreConD	9.634	572	61.3	29.7	0.525	286	8.7	12.3
GreEss	24.899	421	34.8	21.8	1.136	277	2.7	11.9
Hyper+	11.444	432	0.0	22.4	0.101	276	0.0	11.9
PaNDa+ (JP, $\rho = 0.5$)	22.333	427	4.7	22.1	1.317	283	0.9	12.2
PaNDa+ (JA)	20.312	393	52.3	20.4	1.250	272	8.6	11.7
ASSO ($t = 1.0$)	10.124	425	0.1	22.1	0.058	276	2.1	11.8
RSF	3.631	466	40.6	24.3	0.663	276	2.4	11.8
RSF-ES	0.407	431	7.3	22.4	0.126	276	2.4	11.9
	americas small				*dblp*			
GreConD	0.630	197	51.1	3.8	0.006	21	19.2	0.6
GreEss	1.750	189	49.7	3.7	0.090	19	0.0	0.5
Hyper+	0.496	347	0.1	6.7	0.005	19	0.0	0.5
PaNDa+ (JP, $\rho = 0.5$)	1.127	133	0.9	2.6	0.011	22	0.0	0.5
PaNDa+ (JA)	1.226	137	33.4	2.7	0.010	21	32.5	0.6
ASSO ($t = 1.0$)	0.428	176	22.6	3.4	0.002	19	0.0	0.5
RSF	0.313	204	50.0	3.9	*	21	18.9	0.6
RSF-ES	0.184	208	51.0	4.0	*	19	0.0	0.5
	apj				*dna*			
GreConD	0.576	464	41.4	5.7	1.126	511	62.7	9.7
GreEss	1.246	454	32.8	5.6	1.142	371	24.2	7.0
Hyper+	0.029	576	0.2	7.0	0.143	371	0.0	7.1
PaNDa+ (JP, $\rho = 0.5$)	1.160	375	0.3	4.6	1.090	389	1.3	7.4
PaNDa+ (JA)	1.294	425	35.7	5.2	0.950	338	53.4	6.5
ASSO ($t = 1.0$)	0.092	462	22.8	5.7	0.042	355	1.8	6.8
RSF	4.314	461	33.0	5.7	2.036	461	51.3	8.8
RSF-ES	0.312	466	32.7	5.7	0.186	373	20.2	7.1

Table 6
Performance metrics on real-world datasets (Part II). k denotes factors; Ov. is overlap (in %) and Mem represents the maximum RAM memory needed, measured in MiB. An asterisk indicates times lower than 0.001 seconds.

Algorithm	Time	k	Ov.	Mem	Time	k	Ov.	Mem
	domino				*mushroom*			
GreConD	*	21	39.3	0.0	0.311	120	72.4	3.8
GreEss	*	20	8.7	0.0	0.237	100	62.3	3.1
Hyper+	*	23	0.0	0.0	0.123	111	4.5	3.5
PaNDa+ (JP, $\rho = 0.5$)	*	15	0.1	0.0	0.265	110	23.1	3.5
PaNDa+ (JA)	*	17	33.9	0.0	0.231	86	76.2	2.7
ASSO ($t = 1.0$)	*	20	4.1	0.1	0.026	101	54.9	3.2
RSF	*	20	9.1	0.0	0.133	113	72.6	3.5
RSF-ES	*	20	9.1	0.1	0.123	112	72.4	3.5
	emea				*paleo*			
GreConD	0.005	42	37.7	0.5	0.010	151	11.5	0.4
GreEss	0.010	34	0.0	0.4	0.007	143	3.4	0.4
Hyper+	0.002	34	0.0	0.4	0.004	139	0.0	0.4
PaNDa+ (JP, $\rho = 0.5$)	0.007	42	4.9	0.5	0.019	170	1.7	0.4
PaNDa+ (JA)	0.006	36	33.1	0.4	0.017	140	14.3	0.4
ASSO ($t = 1.0$)	*	34	0.0	0.4	0.002	139	0.3	0.3
RSF	*	38	15.8	0.5	0.026	143	4.1	0.4
RSF-ES	*	34	0.0	0.4	0.001	139	0.3	0.3
	firewall1				*tic-tac-toe*			
GreConD	0.016	66	58.3	0.3	0.002	32	11.3	0.2
GreEss	0.025	64	57.8	0.2	0.002	32	11.3	0.2
Hyper+	0.104	90	0.0	0.4	0.003	29	0.0	0.2
PaNDa+ (JP, $\rho = 0.5$)	0.015	40	0.6	0.2	0.004	43	10.3	0.2
PaNDa+ (JA)	0.022	57	45.1	0.2	0.004	27	0.0	0.2
ASSO ($t = 1.0$)	0.006	66	21.2	0.3	*	29	0.0	0.1
RSF	0.007	70	58.7	0.2	*	29	0.0	0.2
RSF-ES	0.004	71	58.7	0.3	*	29	0.0	0.1
	firewall2				*zoo*			
GreConD	0.002	10	41.9	0.0	*	30	52.3	0.1
GreEss	0.003	10	41.5	0.1	*	26	40.0	0.0
Hyper+	0.033	11	0.0	0.1	*	26	5.7	0.0
PaNDa+ (JP, $\rho = 0.5$)	0.003	9	0.0	0.1	*	20	4.7	0.0
PaNDa+ (JA)	0.003	9	39.4	0.1	*	20	52.4	0.0
ASSO ($t = 1.0$)	0.005	10	33.1	0.1	*	24	15.7	0.0
RSF	*	10	41.9	0.0	*	26	40.8	0.0
RSF-ES	*	10	41.9	0.1	*	26	40.8	0.1

garding resource efficiency, all evaluated algorithms maintain a remarkably low memory footprint, with peak consumption staying below 30 MiB across all benchmarks, confirming that the bitwise optimizations ensure high performance regardless of the specific algorithmic strategy employed.

While the exact solvers analyzed above guarantee absolute fidelity ($E_u = E_o = 0$), the approximate methodologies explore different trade-offs between reconstruction accuracy and model complexity. As detailed in Table 7, heuristic methods like PaNDa+ and ASSO exhibit highly variable performance depending on their specific parameterization. For instance, ASSO with a lower confidence threshold ($t = 0.5$) often introduces substantial under-coverage (E_u) and over-coverage (E_o) errors, exceeding 10% in several instances like *mushroom* or *advertisement*. In contrast, PaNDa+ (JA) consistently achieves high fidelity with near-zero error across most datasets, albeit often at the cost of higher factor redundancy compared to its pruning-based counterpart (JP). It is important to note that Hyper+ is omitted from this table as it was executed with a zero-error constraint ($\beta = 0$), thereby operating as an exact solver in our benchmarks. These results underscore that RSF-ES successfully occupies a unique niche: providing the mathematical exactness of traditional FCA methods with a computational efficiency that rivals approximate heuristics.

The high proportion of attribute concepts (AttrC) reported in Table 8, particularly for RSF-ES and RSF, warrants a nuanced interpretation of structural integrity versus greedy efficiency. While purely greedy solvers are designed to maximize local coverage in each step, often by *forcing* combinations of attributes to capture the largest possible area of the residue, the RSF family prioritizes the discovery of the dataset's natural hierarchy. The underlying Rice-Siff agglomerative process only merges components that exhibit genuine global similarity; when the remaining residue no longer presents clear structural affinities, the algorithm naturally reverts to selecting attribute concepts as the most parsimonious way to complete the factorization.

Within our framework, the selection of an AttrC is therefore not a sign of triviality, but rather an indicator of *structural consistency*. It demonstrates that once the cohesive, hierarchical cores of the data are extracted through fusion, the algorithm prefers using base

Table 7
Reconstruction fidelity of approximate BMF algorithms. E_u (under-coverage) and E_o (over-coverage) are reported as percentages (%). Both metrics are identically zero for all exact solvers (RSF-ES, RSF, GreConD, GreEss, and Hyper+).

	PaNDa+ (JP)		PaNDa+ (JA)		ASSO ($t = 1$)	
Dataset	E_u	E_o	E_u	E_o	E_u	E_o
advertisement	0.27	6.30	0.00	10.42	0.00	5.19
americas large	0.04	11.24	0.00	17.71	0.00	0.14
americas small	0.20	7.33	0.00	8.12	0.00	4.13
apj	4.84	1.39	0.00	4.96	0.00	5.43
chess	0.00	17.44	0.00	18.70	0.00	24.31
customer	0.05	3.08	0.00	2.39	0.00	0.02
dblp	0.00	9.81	0.00	9.81	0.00	0.00
dna	0.27	11.25	0.00	15.14	0.00	2.02
domino	1.10	0.96	0.00	10.21	0.00	0.00
emea	0.00	10.05	0.00	12.24	0.00	0.00
firewall1	0.15	2.97	0.00	4.07	0.00	6.34
firewall2	0.00	1.19	0.00	1.19	0.00	0.00
mushroom	0.00	18.26	0.00	23.89	0.00	12.93
paleo	0.03	7.68	0.00	6.03	0.00	0.00
tic-tac-toe	0.00	10.27	0.00	6.26	0.00	0.00
zoo	0.46	21.36	0.00	25.24	0.00	3.15

Table 8
Proportion (%) of attribute concepts (AttrC) in each factorization and mean number of attributes per factor (mean intent size, $\overline{\text{Int}}$) for exact solvers.

	RSF		RSF-ES		GreConD		GreEss		Hyper+	
Dataset	AttrC	$\overline{\text{Int}}$	AttrC	$\overline{\text{Int}}$	AttrC	$\overline{\text{Int}}$	AttrC	$\overline{\text{Int}}$	AttrC	$\overline{\text{Int}}$
advertisement	98.4	6.75	98.6	6.71	85.5	7.35	77.6	7.95	22.0	2.04
americas large	83.0	1.38	99.8	1.05	51.7	1.89	77.2	1.31	89.8	1.00
americas small	95.6	42.89	98.6	41.79	76.1	48.50	70.9	52.88	18.7	4.59
apj	98.7	5.36	99.4	5.29	96.8	5.39	97.4	5.34	44.8	2.03
chess	96.0	5.21	96.0	5.21	12.9	9.29	12.7	8.73	22.5	1.70
customer	100.0	1.92	100.0	1.92	96.5	1.95	98.9	1.96	75.0	1.00
dblp	90.5	1.10	100.0	1.00	90.5	1.10	100.0	1.00	100.0	1.00
dna	74.8	2.15	98.7	1.66	66.3	2.61	90.8	1.80	74.1	1.06
domino	100.0	1.81	100.0	1.81	95.2	1.86	85.0	2.01	34.8	1.00
emea	89.5	1.43	100.0	1.00	81.0	1.77	100.0	1.00	100.0	1.00
firewall1	92.9	5.65	95.8	5.65	84.8	5.79	67.2	6.07	31.1	1.00
firewall2	100.0	2.61	100.0	2.61	100.0	2.61	90.0	2.69	9.1	1.00
mushroom	96.5	8.61	97.3	8.61	60.8	9.99	75.0	11.14	1.8	1.12
paleo	97.2	1.24	100.0	1.01	92.1	1.32	97.2	1.09	99.3	1.00
tic-tac-toe	100.0	1.00	100.0	1.00	81.2	1.19	81.2	1.19	100.0	1.00
zoo	88.5	3.12	88.5	3.12	56.7	3.80	65.4	3.85	46.2	1.15
Weighted Avg.	92.1	6.52	98.9	6.55	75.4	7.00	82.2	7.62	49.6	1.82

concepts as residuals instead of constructing artificial combinations that lack a hierarchical basis. This hybrid strategy allows RSF-ES to achieve exact reconstruction with a total number of factors (k) that remains competitive with state-of-the-art solvers, proving that its effectiveness stems from a balanced decomposition that respects the intrinsic topology of the formal context rather than a simple maximization of step-wise coverage.

6.3. Scalability

A critical aspect of Boolean Matrix Factorization is how algorithms scale with the size and structural complexity of the formal context. While traditional FCA-based methods often exhibit a performance bottleneck linked to the total number of formal concepts $|\mathfrak{B}(B, A, R)|$, RSF-ES is fundamentally driven by the size of the initial conceptual base. In our experimental setup, this base corresponds to the number of attribute concepts $\mu(A)$.

Fig. 1 illustrates the execution time trend for the evaluated exact solvers as a function of $|\mu(A)|$. The results reveal that RSF-ES (blue dashed line) maintains a significantly flatter scalability curve compared to its original variant RSF and traditional greedy solvers like GreEss and GreConD. For instance, in datasets with high conceptual complexity such as *americas large* or *advertisement*, RSF-ES manages to complete the factorization in a fraction of the time required by GreConD, despite both algorithms guaranteeing absolute fidelity.

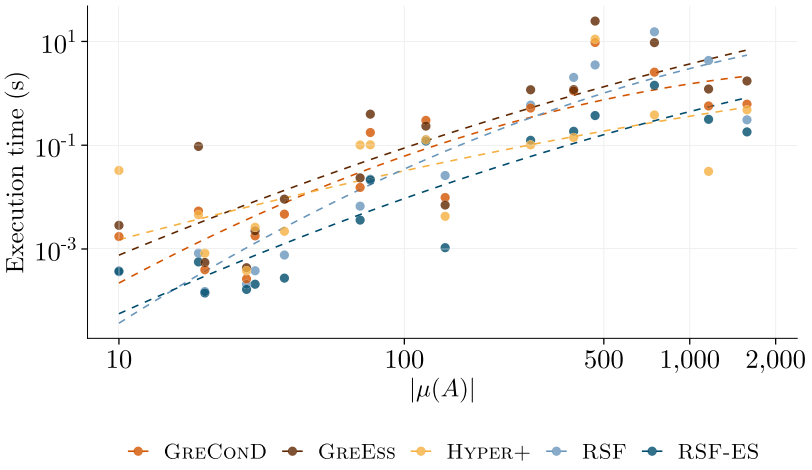


Fig. 1. Execution time scalability of exact BMF solvers, with respect to the number of attribute concepts ($|\mu(A)|$). Both axes are log-scaled.

This efficiency is largely attributed to the synergy between the bitwise-optimized Rice-Siff distance and the topological pruning mechanism. By identifying structurally disjoint components early in the agglomerative process ($\rho_{un} = 1.0$), RSF-ES avoids the quadratic growth of the candidate search space that typically plagues hierarchical clustering in dense contexts. Furthermore, while Hyper+ remains highly competitive, it is worth noting that RSF-ES achieves similar execution times while providing a strictly exact decomposition without the need for the summarization stage. These findings confirm that RSF-ES is not only computationally efficient but also robust to the combinatorial explosion of the concept lattice, making it a viable candidate for large-scale exact summarization.

6.4. Discussion

The experimental results indicate that the integration of the Rice-Siff distance as an early stopping criterion provides a clear computational advantage in exact Boolean Matrix Factorization. By utilizing the topological structure of the data, RSF-ES effectively manages the search space, avoiding the combinatorial explosion that typically affects agglomerative search. This efficiency is achieved while maintaining absolute reconstruction fidelity and extracting a number of factors that is competitive with established greedy methods.

In conclusion, the proposed pruning strategy offers a robust framework for handling high-dimensional datasets where traditional solvers encounter bottlenecks. While algorithms like GreConD and GreEss are efficient for smaller contexts, the scalability demonstrated by RSF-ES in scenarios with thousands of attributes makes it a practical choice for large-scale BMF tasks. These findings suggest that incorporating distance-based pruning is a viable approach to improve the performance of formal concept-based factorization without sacrificing structural parsimony.

7. Conclusions and future work

This work introduced the Rice-Siff Factorization (RSF) and its optimized variant RSF-ES, a new paradigm for Boolean Matrix Factorization that leverages state-dependent agglomerative clustering as a topological regularizer. Our theoretical analysis proves the exact optimality of RSF for nominal and ordinal scales, while demonstrating that the early stopping criterion in RSF-ES effectively prevents factor fragmentation and rank inflation.

In summary, the experimental evidence across both synthetic and real-world benchmarks positions RSF-ES as a robust alternative for exact Boolean Matrix Factorization. By shifting the computational focus from the exhaustive exploration of formal concepts to an optimized attribute-concept base, RSF-ES achieves significant gains in model parsimony, providing up to 27% more compact representations than standard greedy approaches in complex datasets like *dna*. More importantly, these parsimony gains are achieved while maintaining absolute reconstruction fidelity ($E_u = E_o = 0$), unlike approximate heuristics. Finally, our scalability analysis confirms that RSF-ES effectively overcomes the traditional FCA performance bottleneck, remaining computationally efficient in high-dimensional scenarios where traditional solvers may become intractable.

As part of future work, we plan to further investigate the Rice-Siff algorithm with respect to the use of different distance functions and their theoretical approximation aspects. In particular, we intend to study how alternative set-based and vector-based measures influence the merging process, the resulting hierarchy, and the number and quality of the formal concepts produced by the algorithm. Besides the originally used Jaccard distance, potential candidates include distances derived from the Tversky index and the Dice coefficient, as well as vector-based measures obtained from binary representations of sets, such as the Manhattan, Euclidean, and more generally Minkowski distances. In future work, we plan to explore potential applications of the Rice-Siff algorithm in the analysis of social networks, where formal concepts may represent communities of actors sharing common relationships or attributes [25].

CRediT authorship contribution statement

Lubomir Antoni: Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Dominika Kotlárová:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Ondrej Krídlo:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization; **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-Aciego:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, 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.

Acknowledgment

The authors are grateful to Dr. Martin Trnecka, from Palacký University Olomouc, for his generosity in sharing the original source code and granting permission for its porting to R and C. This contribution was pivotal in enabling the high-performance implementation and the subsequent rigorous comparative analysis presented in this work. This research is partially supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under contract VEGA 1/0539/25 (O. Krídlo, D. Kotlárová, L. Antoni); by the State Agency of Research (AEI), the Spanish Ministry of Science, Innovation, and Universities (MCIU) and the European Social Fund (FEDER) through the VALID research project (PID2022-140630NB-I00 funded by MCIN/ AEI/ 10.13039/ 501100011033); by Junta de Andalucía PAIDI group TIC-115 project PPRO-TIC115-G-2023 (PPRO-TIC115-G-FEDER) (D. López-Rodríguez, M. Ojeda-Aciego).

Data availability

Data will be made available on request.

References

[1] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, H. Mannila, The discrete basis problem, IEEE Trans. Knowl. Data Eng. 20 (10) (2008) 1348–1362. https://doi.org/10.1109/TKDE.2008.53
[2] F. Ban, V. Bhattiprolu, K. Bringmann, P. Kolev, E. Lee, D.P. Woodruff, A polynomial-time approximation scheme for p-low rank approximation, in: Proc. of the Symposium on Discrete Algorithms (SODA), SIAM, 2019, pp. 747–766.
[3] F.V. Fomin, P.A. Golovach, F. Panolan, Parameterized low-rank binary matrix approximation, Data Min. Knowl. Discov. 34 (2020) 478–532.
[4] C. Kolomvakis, T. Bobille, A. Vandaele, N. Gillis, Algorithms for Boolean matrix factorization using integer programming and heuristics, 2025, https://arxiv.org/abs/2512.03807.
[5] T. Rukat, C.C. Holmes, M.K. Titsias, C. Yau, Bayesian Boolean matrix factorisation, in: Proc of the Intl Conf on Machine Learning, 70 of *Proceedings of Machine Learning Research*, 2017, pp. 2969–2978. https://proceedings.mlr.press/v70/rukat17a.html.
[6] K. Brazdilova, M. Trnecka, M. Trneckova, Data preprocessing using banded structure and image morphology enhancing Boolean matrix factorization, Knowl. Based Syst. 329 (2025) 114366. https://doi.org/10.1016/j.knosys.2025.114366.
[7] R. Belohlavek, V. Vychodil, Discovery of optimal factors in binary data via a novel method of matrix decomposition, J. Comput. Syst. Sci. 76 (1) (2010) 3–20. https://doi.org/10.1016/j.jcss.2009.05.002
[8] F. Rioult, A. Mouakher, A. Ouali, Size-optimal Boolean matrix factorization, Discrete Appl. Math. 377 (2025) 314–326. https://doi.org/10.1016/j.dam.2025.06.027.
[9] M.D. Rice, M. Siff, Clusters, concepts, and pseudometrics, Electron. Notes Theor. Comput. Sci. 40 (2002) 323–346.
[10] B. Ganter, R. Wille, Formal Concept Analysis: Mathematical Foundations, Springer-Verlag, 2nd edition, 1999.
[11] D.S. Nau, Specificity Covering: Immunological and Other Applications, Computational Complexity and Other Mathematical Properties, and a Computer Program, Master's thesis, Department of Computer Science, Duke University, 1976.
[12] L.J. Stockmeyer, The Set Basis Problem is NP-Complete, Research Report RC5431, IBM Research Center, 1975. https://dominoweb.draco.res.ibm.com/reports/rc5431.pdf.
[13] K.H. Kim, Boolean Matrix Theory and Applications, Marcel Dekker, New York, New York, 1982.
[14] C. Lucchese, S. Orlando, R. Perego, Mining top-k patterns from binary datasets in presence of noise, in: Proceedings of the 2010 SIAM International Conference on Data Mining (SDM), 2010, pp. 165–176. https://doi.org/10.1137/1.9781611972801.15
[15] R. Belohlavek, M. Trnecka, From-below approximations in Boolean matrix factorization: geometry and new algorithm, J. Comput. Syst. Sci. 81 (8) (2015) 1678–1714. https://doi.org/10.1016/j.jcss.2015.06.002
[16] M. Trnecka, M. Trneckova, Data reduction for Boolean matrix factorization algorithms based on formal concept analysis, Knowl. Based Syst. 151 (2018) 63–76. https://doi.org/10.1016/j.knosys.2018.05.035
[17] P. Krajča, M. Trnecka, Parallelization of the GreConD algorithm for Boolean matrix factorization, in: Formal Concept Analysis, Springer International Publishing, Cham, 2019, pp. 208–222.
[18] D.I. Ignatov, A. Yakovleva, On suboptimality of GreConD for Boolean matrix factorisation of contranominal scales, in: Proc. "What can FCA do for Artificial Intelligence?" at IJCAI, 2972 of *CEUR Workshop Proceedings*, 2021, pp. 87–98. https://ceur-ws.org/Vol-2972/paper8.pdf.
[19] C. Lucchese, S. Orlando, R. Perego, A unifying framework for mining approximate top-k binary patterns, IEEE Tr. on Knowledge and Data Engineering 26 (12) (2013) 2900–2913. https://doi.org/10.1109/TKDE.2013.181
[20] Y. Xiang, R. Jin, D. Fuhry, F.F. Dragan, Summarizing transactional databases with overlapped hyperrectangles, Data Min. Knowl. Discov. 23 (2) (2011) 215–251. https://doi.org/10.1007/s10618-010-0203-9

[21] V. Chvátal, A greedy heuristic for the set-covering problem, Math. Oper. Res. 4 (3) (1979) 233–235. https://doi.org/10.1287/moor.4.3.233
[22] D. de Caen, D.A. Gregory, N.J. Pullman, The Boolean rank of zero-one matrices, in: Proc. of the 3rd Caribbean Conf. on Combinatorics and Computing, 1981, pp. 169–173.
[23] P. Cordero, M. Enciso, D. López-Rodríguez, A. Mora, FcaR, formal concept analysis with R, R J. 14 (1) (2022) 341–361. https://doi.org/10.32614/rj-2022-014
[24] D. Eddelbuettel, R. François, Rcpp: seamless R and C++ integration, J. Stat. Softw. 40 (8) (2011) 1–18. https://doi.org/10.18637/jss.v040.i08
[25] S. Krajči, Social network and formal concept analysis, in: W. Pedrycz, S.-M. Chen (Eds.), Social Networks: A Framework of Computational Intelligence, 526 of *Studies in Computational Intelligence*, Springer, Cham, 2014, pp. 37–57. https://doi.org/10.1007/978-3-319-02993-1_3