Journal of Computer and System Sciences 81 (2015) 1678–1697

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

From-below approximations in Boolean matrix factorization: Geometry and new algorithm

Radim Belohlavek *, Martin Trnecka

Data Analysis and Modeling Lab, Dept. Computer Science, Palacký University, Czech Republic

ARTICLE INFO

Article history:
Received 10 March 2014
Received in revised form 31 January 2015
Accepted 31 March 2015
Available online 11 June 2015

Keywords:
Boolean matrix
Matrix decomposition
Closure structures
Concept lattice
Approximation algorithm

ABSTRACT

We present new results on Boolean matrix factorization and a new algorithm based on these results. The results emphasize the significance of factorizations that provide from-below approximations of the input matrix. While the previously proposed algorithms do not consider the possibly different significance of different matrix entries, our results help measure such significance and suggest where to focus when computing factors. An experimental evaluation of the new algorithm on both synthetic and real data demonstrates its good performance in terms of good coverage by the first few factors as well as a small number of factors needed for an almost exact decomposition and indicates that the algorithm outperforms the available ones in these terms. We also propose future research topics.

1. Introduction

Boolean matrix factorization (BMF, called also Boolean matrix decomposition) is becoming an established method for analysis and preprocessing of data. The existing BMF methods are based on various types of heuristics because the main computational problems involved are known to be provably hard. The heuristics employed, however, use only a limited theoretical insight regarding BMF. This paper attempts to show that a better understanding of the geometry of Boolean data results in a better understanding of BMF, theoretically justified heuristics, and better algorithms.

In particular, we present new results in BMF derived from examining the closure and order-theoretic structures related to Boolean data, namely the lattice of all fixpoints of the Galois connections associated with the input matrix (so-called concept lattice). This viewpoint makes explicit the essence of BMF as a covering problem and emphasizes one type of factorizations we call from-below factorizations. Such factorizations and related notions were examined in some previous papers. While all the existing BMF methods consider the entries containing 1s in the input matrix essentially equally important, we propose to differentiate their role. In particular, we examine the entries that are essential for BMF in that their coverage by factors guarantees exact decomposition of the input matrix I by these factors. Crucial in our approach are intervals in the concept lattice associated with I. We show that every such interval contains just the factors covering a certain block full of 1s in I and that the intervals form reasonable subspaces for the search of factors. We present a new BMF algorithm which is based on these results and computes from-below factorizations. It turns out from experimental evaluation on both synthetic and real data that the new algorithm outperforms the existing BMF algorithms. Moreover, we clarify some connections between the existing approaches to BMF and argue that the closure and order-theoretic structures utilized in this paper provide a natural geometric view and represent a useful framework for theoretical analysis of the various BMF problems.

* Corresponding author.
E-mail addresses: radim.belohlavek@acm.org (R. Belohlavek), martin.trnecka@gmail.com (M. Trnecka).

http://dx.doi.org/10.1016/j.jcss.2015.06.002
0022-0000/

2. Preliminaries and related work

2.1. Notation and basic notions

Throughout this paper, we denote by I an $n \times m$ Boolean matrix, interpreted primarily as an object–attribute incidence (hence the symbol I) matrix, i.e. the entry I_{ij} corresponding to the row i and the column j is either 1 or 0, indicating that the object i does or does not have the attribute j. The set of all $n \times m$ Boolean matrices is denoted by $\{0,1\}^{n\times m}$. The ith row and jth column vector of I is denoted by $I_{i\_}$ and $I_{\_j}$, respectively. A general aim in BMF is to find for a given $I \in \{0,1\}^{n\times m}$ (and possibly other given parameters) matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ for which

$$I \text{ (approximately) equals } A \circ B, \tag{1}$$

where $\circ$ is the Boolean matrix product, i.e. $(A \circ B)_{ij} = \max_{l=1}^{k} \min(A_{il}, B_{lj})$. A decomposition of I into $A \circ B$ may be interpreted as a discovery of k factors that exactly or approximately explain the data: interpreting I, A, and B as the object–attribute, object–factor, and factor–attribute matrices, the model (1) reads: the object i has the attribute j if and only if there exists factor l such that l applies to i and j is one of the particular manifestations of l. The least k for which an exact decomposition $I = A \circ B$ exists is called the *Boolean rank* (Schein rank) of k and is denoted by $\mathrm{rank}_{\mathrm{B}}(I)$.

Recall that the L_1-norm (Hamming weight in case of Boolean matrices) $||\cdot||$ and the corresponding metric $E(\cdot,\cdot)$ are defined for $C, D \in \{0,1\}^{n\times m}$ by

$$||C|| = \sum_{i,j=1}^{m,n} |C_{ij}| \quad \text{and} \quad E(C,D) = ||C - D|| = \sum_{i,j=1}^{m,n} |C_{ij} - D_{ij}|. \tag{2}$$

The following variants of the BMF problem, relevant to this paper, are considered in the literature.

- *Discrete Basis Problem* (DBP, [20]):
  Given $I \in \{0,1\}^{n\times m}$ and a positive integer k, find $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ that minimize $||I - A \circ B||$.
- *Approximate Factorization Problem* (AFP, [4]):
  Given I and prescribed error $\varepsilon \geq 0$, find $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ with k as small as possible such that $||I - A \circ B|| \leq \varepsilon$.

These two problems reflect two important views on BMF. The first one emphasizes the importance of the first k (presumably most important) factors. The second one emphasizes the need to account for (and thus to explain) a prescribed portion of data, which is specified by ε.

2.2. Related work

Matrix decompositions represent an extensive subject whose coverage is beyond the scope of this paper. A good overview from BMF viewpoint is found e.g. in [20]. Except for the area of Boolean matrix theory itself, see e.g. [12], relevant results are traditionally presented in the literature on binary relations and graph theory, see e.g. [6,27]. These results may be translated to the results on Boolean matrices due to various correspondences, such as those connecting Boolean matrices, bipartite graphs, and binary relations, and pertain mostly to combinatorial and computational complexity questions. An important related area is formal concept analysis (FCA) [10], in which Boolean matrices are represented by so-called formal contexts, i.e. binary relations between objects and attributes. FCA provides solid lattice-theoretical foundations which we utilize.

Decompositions of Boolean matrices using methods designed originally for real-valued data and various modifications of these methods appear in a number of papers. [29] compares several approaches to assessment of dimensionality of Boolean data and concludes that a major problem with applying to Boolean data the methods designed originally for real-valued data is the lack of interpretability. Similar observations were presented by other authors. Among the first works on applications of BMF in data analysis are [24,25], in which the authors have already been aware of the provable computational difficulty (NP-hardness) of the decomposition problem due to NP-hardness of the set basis problem [28]. The interest in BMF in data mining is primarily due to the work of Miettinen et al. In particular, the DBP, the corresponding complexity results, and the Asso algorithm discussed below appeared in [20]. In [11], they authors examine "tiling" of Boolean data and various related problems, their complexity, and algorithms. Tiling is closely related to BMF as it corresponds to the from-below factorizations we investigate in this paper and is discussed in more detail in Section 5.1. In [4], we showed that formal concepts (i.e. fixpoints of Galois connections) are natural factors of Boolean matrices, proved their optimality for exact factorizations, described transformations between attribute and factor spaces, and proposed two BMF algorithms discussed below. In [31], the authors investigate the problem of summarizing transactional databases by so-called hyperrectangles, examine the computational complexity of the problems involved, provide the Hyper algorithm and discuss related problems. The summarizations involved may be rephrased as Boolean matrix decompositions and this approach is discussed in more detail in Sections 3 and 5. Directly relevant to our paper is also [16], where the authors propose the PaNDa algorithm discussed in Section 5. In particular, we use the algorithms proposed in [4,11,16,20,31] in the experimental evaluation of our new algorithm. Further work

relevant to BMF includes other Miettinen's papers, such as [17] on Boolean CX and CUR decompositions, [18] which investigates sparsity in BMF, and [21] employing the minimum description length principle in BMF. [22] presents a useful survey containing several results on complexity and various ranks for Boolean matrices. Regarding ranks, the reader is also referred to [20]; for complexity issues see the above papers and [30]. Interesting applications of BMF and its various extensions in role mining are found in [8,14,15,30]. In particular, [8] proposes factorization algorithms to which we refer in Section 5. Applications in reducing dimensionality in classification of Boolean data are found in [26].

3. From-below approximations and geometry of BMF

3.1. Factorizations as coverings and from-below approximations

We first make explicit the following view of decompositions, present implicitly in [4]. For matrices J_1 and J_2, we put

$$J_1 \leq J_2 \,(J_1 \text{ is contained in } J_2) \quad \text{iff} \quad (J_1)_{ij} \leq (J_2)_{ij} \text{ for every } i, j. \tag{3}$$

A matrix $J \in \{0,1\}^{n\times m}$ is called *rectangular* (a rectangle, for short) if $J = C \circ D$ for some $C \in \{0,1\}^{n\times 1}$ (column) and $D \in \{0,1\}^{1\times m}$ (row). Clearly, this implies that upon suitable permutations of columns and rows the 1s in J form a rectangular area. We say that J (or, the pair $\langle C, D\rangle$ for which $J = C \circ D$) *covers* $\langle i, j\rangle$ if $J_{ij} = 1$.

Observation 1. *The following conditions are equivalent for any $I \in \{0,1\}^{n\times m}$.*

(a) *$I = A \circ B$ for some $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$.*
(b) *There exist rectangles $J_1, \ldots, J_k \in \{0,1\}^{n\times m}$ such that $I = J_1 \vee \cdots \vee J_k$, i.e. $I_{ij} = \max_{l=1}^{k} (J_l)_{ij}$.*
(c) *There exist rectangles $J_1, \ldots, J_k \in \{0,1\}^{n\times m}$ contained in I such that $I_{ij} = 1$ if and only if $\langle i, j\rangle$ is covered by some J_l.*

In particular, if A and B are the matrices from Observation 1(a) then one may put $J_l = A_{\_l} \circ B_{l\_}$ ($l = 1, \ldots, k$), i.e. J_l is the product of the lth column of A and the lth row of B, to obtain the rectangles in (b) and (c). Conversely, if $J_1 = C_1 \circ D_1, \ldots, J_k = C_k \circ D_k$ are the rectangles in (b) or (c) then the matrices A and B in which the lth column and lth row are C_l and D_l, respectively, satisfy (a). Hence, if A and B form the output of any BMF method for an input matrix I, one may identify the factors $l = 1, \ldots, k$ with pairs consisting of the column $A_{\_l}$ and row $B_{l\_}$ or, equivalently, with rectangles $A_{\_l} \circ B_{l\_}$. Furthermore, the objective to compute A and B with no/small error $E(I, A \circ B)$ may be rephrased as the goal to compute from I a set of rectangles that exactly/approximately cover I.

Clearly, E as defined by (2) may be seen as being a sum of two components, E_u ("uncovered") corresponding to 1s in I that are 0s in $A \circ B$ and E_o ("overcovered") corresponding to 0s in I that are 1s in $A \circ B$:

$$\begin{aligned} E(I, A \circ B) &= E_u(I, A \circ B) + E_o(I, A \circ B), \text{ where} \\ E_u(I, A \circ B) &= |\{\langle i, j\rangle \,;\, I_{ij} = 1, (A \circ B)_{ij} = 0\}|, \\ E_o(I, A \circ B) &= |\{\langle i, j\rangle \,;\, I_{ij} = 0, (A \circ B)_{ij} = 1\}|. \end{aligned}$$

Even though E_u and E_o look symmetric, they have a highly non-symmetric role in BMF. These two components are implicitly used in Asso algorithm [20] and are treated non-symmetrically by function *cover* using two different weights. The non-symmetry is seen from the following observation which says that as we add new factors to the already established ones (i.e., add columns and rows to A and B, respectively), E_u may only decrease while E_o may only increase. This property is easy to see using Observation 1.

Observation 2. *Let $A' \in \{0,1\}^{n\times(k+1)}$ and $B' \in \{0,1\}^{(k+1)\times m}$ result by adding to A and B a single column and row, respectively. Then*

$$E_u(I, A' \circ B') \leq E_u(I, A \circ B) \text{ and } E_o(I, A' \circ B') \geq E_o(I, A \circ B).$$

The importance of Observation 2 derives from the following consideration. Due to the provable hardness of the BMF related problems, such as DBP or AFP, it seems reasonable to assume that conceivable algorithms follow the logic of Observation 2 in that they output one factor after another. This is indeed the case of the main existing algorithms discussed below. With such algorithms, Observation 2 provides a warning. Namely, we should be careful with committing E_o error because E_o never decreases by adding further factors.

The most extreme strategy is not to commit E_o error at all, i.e. add the constraint $E_o(I, A \circ B) = 0$. As the requirement $E_o(I, A \circ B) = 0$ is equivalent to $A \circ B \leq I$, we call a BMF algorithm producing results with zero E_o a *from-below factorization algorithm* and say that A and B provide a *from-below approximation* of I.

The restriction to from-below factorization may naturally be required due to the nature of the data. A point in case is role mining [14,15,30] in which the data represents users (rows) and their access permissions (columns). In this case, committing E_o, which corresponds to asserting new access permissions to users, is not desirable, much more than not asserting

a permission, see [14,15,30], implying the need of from-below factorizations. In addition, the restriction to from-below factorizations means that we exploit only a restricted class of factorizations. Surprisingly however, we show that such restriction leads to very good BMF algorithms which outperform the available algorithms producing the general factorizations, i.e. algorithms committing E_o error. A further advantageous feature of the from-below approximations is the fact that they are amenable to theoretical analysis in terms of closure and order-theoretic structures, as demonstrated below.

To every Boolean matrix $I \in \{0,1\}^{n\times m}$, one might associate the pair $\langle {}^{\uparrow_I}, {}^{\downarrow_I}\rangle$ (denoted also $\langle {}^{\uparrow}, {}^{\downarrow}\rangle$) of operators assigning to sets

$$C \subseteq X = \{1,\ldots,n\} \quad \text{and} \quad D \subseteq Y = \{1,\ldots,m\}$$

the sets

$$C^{\uparrow_I} = \{j \in Y \mid \forall i \in C : I_{ij} = 1\} \text{ and } D^{\downarrow_I} = \{i \in X \mid \forall j \in D : I_{ij} = 1\}.$$

That is, $C^{\uparrow}$ is the set of all attributes (columns) shared by all objects (rows) in C and $D^{\downarrow}$ is the set of all objects sharing all attributes in D. The set

$$\mathcal{B}(I) = \{\langle C, D\rangle \mid C \subseteq X, D \subseteq Y, C^{\uparrow} = D, D^{\downarrow} = C\}$$

is called the *concept lattice* of I. It consists of all $\langle {}^{\uparrow}, {}^{\downarrow}\rangle$-closed pairs $\langle C, D\rangle$, called the *formal concepts* of I, with C and D called the *extent* and the *intent*. The set $\mathcal{B}(I)$ equipped with the partial order $\leq$ (modeling the subconcept–superconcept hierarchy) defined by $\langle C_1, D_1\rangle \leq \langle C_2, D_2\rangle$ iff $C_1 \subseteq C_2$ iff $D_1 \supseteq D_2$ indeed forms a complete lattice. Concept lattices are the basic structures in formal concept analysis (FCA); we refer to [7,10] for details. The pair $\langle {}^{\uparrow}, {}^{\downarrow}\rangle$ forms a Galois connection between X and Y and the compound mappings ${}^{\uparrow\downarrow}$ and ${}^{\downarrow\uparrow}$ form closure operators in X and Y, respectively [10]. A concept lattice may be visualized using a particularly labeled line diagram and carries useful information about the data I which we utilize in our paper. Note also that several polynomial-time delay algorithms are available for computing $\mathcal{B}(I)$ [13].

An important link between BMF and formal concepts consists in the following facts. First, in view of Observation 1, rectangles contained in I are the building blocks of decompositions of I. Clearly, most efficient are the rectangles that are maximal w.r.t. containment $\leq$ defined by (3). As is well known, maximal rectangles contained in I correspond to formal concepts in $\mathcal{B}(I)$ in that J is a maximal rectangle in I if and only if there exists a formal concept $\langle C, D\rangle \in \mathcal{B}(I)$ such that $J_{ij} = 1$ is equivalent to $i \in C$ and $j \in D$. This link is utilized in [4], in particular in two BMF algorithms which we use in our experimental comparison below. Next, we generalize a theorem from [4] regarding exact decompositions to from-below approximations. Given a set $\mathcal{F} = \{\langle C_1, D_1\rangle, \ldots, \langle C_k, D_k\rangle\} \subseteq \mathcal{B}(I)$ (with a fixed indexing of the formal concepts $\langle C_l, D_l\rangle$), define the $n \times k$ and $k \times m$ Boolean matrices $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ by

$$(A_{\mathcal{F}})_{il} = \begin{cases} 1 & \text{if } i \in C_l, \\ 0 & \text{if } i \notin C_l, \end{cases} \quad \text{and} \quad (B_{\mathcal{F}})_{lj} = \begin{cases} 1 & \text{if } j \in D_l, \\ 0 & \text{if } j \notin D_l, \end{cases} \tag{4}$$

for $l = 1,\ldots,k$. That is, the lth column and lth row of A and B are the characteristic vectors of C_l and D_l, respectively.

Remark 1. The preceding paragraph and Observation 1 make it easy to see that the Minimum Tiling Problem (MTP) considered in [11] is equivalent to the problem of finding an exact decomposition of a Boolean matrix. Namely, a database of n objects and m items considered in [11] may be identified with a Boolean matrix $I \in \{0,1\}^{n\times m}$; a tile in I is a pair $\langle C, D\rangle$ where $C \subseteq \{1,\ldots,n\}$ and $D \subseteq \{1,\ldots,m\}$ such that every object in C has every item in D. Hence, tiles in I may be identified with rectangles contained in I. Moreover, maximal tiles in I are just formal concepts of I. MTP consists in finding a smallest set of tiles that cover the whole database. It is now clear that every set $\mathcal{F}$ of tiles of I may be identified with matrices $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ as in (4), that $A_{\mathcal{F}} \circ B_{\mathcal{F}} \leq I$ (i.e. $\mathcal{F}$ provides a from-below approximation of I), and that $\mathcal{F}$ is a solution to MTP iff $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ present a solution to the AFP problem from Section 2.1 for $\varepsilon = 0$. [11] proposed an algorithm for the MTP which we examine below. Note also that the connection of tiling to BMF is not mentioned in [11].

The next theorem asserts that decompositions utilizing formal concepts as factors are the best as far as the from-below approximations are concerned.

Theorem 1. *Let $A \circ B \leq I$ for $n \times k$ and $k \times m$ Boolean matrices A and B. Then there exists a set $\mathcal{F} \subseteq \mathcal{B}(I)$ of formal concepts of I with $|\mathcal{F}| \leq k$ such that for the $n \times |\mathcal{F}|$ and $|\mathcal{F}| \times m$ Boolean matrices $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ we have*

$$A_{\mathcal{F}} \circ B_{\mathcal{F}} \leq I \text{ and } E(I, A_{\mathcal{F}} \circ B_{\mathcal{F}}) \leq E(I, A \circ B).$$

Proof. The proof follows the same logic as the one of [4, Theorem 2] and we include it for reader's convenience. An informal argument: each of the k rectangles corresponding to $A \circ B$ is a rectangle in I and is contained in a maximal rectangle (formal concept) in I. The set $\mathcal{F}$ of these formal concepts has at most k elements and covers at least as many entries of I as those

covered by $A \circ B$. Formally, every rectangle $J_l = A_{\_l} \circ B_{l\_}$ is contained in I. According to Observation 1, $A \circ B = \max_{l=1}^{k} J_l$. Now consider the sets $C_l = \{i \mid A_{il} = 1\}$ and $D_l = \{j \mid B_{lj} = 1\}$. Every $\langle C_l^{\uparrow\downarrow}, C_l^{\uparrow}\rangle$ is a formal concept in $\mathcal{B}(I)$ (a well-known fact in FCA). Moreover $C_l \subseteq C_l^{\uparrow\downarrow}$, since $^{\uparrow\downarrow}$ is a closure operator. As $I_{il} = 1$ for every $i \in C_l$ and $j \in D_l$, it follows that $D_l \subseteq C_l^{\uparrow}$. Now consider the set

$$\mathcal{F} = \{\langle C_1^{\uparrow\downarrow}, C_1^{\uparrow}\rangle, \ldots, \langle C_k^{\uparrow\downarrow}, C_k^{\uparrow}\rangle\} \subseteq \mathcal{B}(I)$$

and the matrices $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$. Clearly $\mathcal{F}$ contains at most k elements. It is easy to check that the rectangle corresponding to $\langle C_l^{\uparrow\downarrow}, C_l^{\uparrow}\rangle$, i.e. the cross-product $(A_{\mathcal{F}})_{\_l} \circ (B_{\mathcal{F}})_{l\_}$, is contained in I and, due to the above observation, contains J_l. Hence,

$$A \circ B = \max_{l=1}^{k} J_l \leq \max_{l=1}^{k} (A_{\mathcal{F}})_{\_l} \circ (B_{\mathcal{F}})_{l\_} = A_{\mathcal{F}} \circ B_{\mathcal{F}} \leq I.$$

It follows that $E(I, A_{\mathcal{F}} \circ B_{\mathcal{F}}) \leq E(I, A \circ B)$, finishing the proof. □

3.2. Intervals in $\mathcal{B}(I)$, role of entries in I, and the essential part of I

As we show in this section, formal concepts and other structures associated with the Boolean matrices help us understand the geometry of BMF. In particular, we show that the concept lattice $\mathcal{B}(I)$ may help us differentiate the role of entries of I in decompositions—an issue not addressed in the existing literature—and that intervals in $\mathcal{B}(I)$ play a crucial role in this regard.

For formal concepts $\langle C_1, D_1\rangle, \langle C_2, D_2\rangle \in \mathcal{B}(I)$, the subset

$$[\langle C_1, D_1\rangle, \langle C_2, D_2\rangle] = \{\langle E, F\rangle \in \mathcal{B}(I) \mid \langle C_1, D_1\rangle \leq \langle E, F\rangle \leq \langle C_2, D_2\rangle\} \tag{5}$$

of $\mathcal{B}(I)$ is called the *interval in $\mathcal{B}(I)$ bounded by* $\langle C_1, D_1\rangle$ and $\langle C_2, D_2\rangle$. Furthermore, for $C \subseteq X$ and $D \subseteq Y$, let $\gamma(C) = \langle C^{\uparrow\downarrow}, C^{\uparrow}\rangle$ and $\mu(D) = \langle D^{\downarrow}, D^{\downarrow\uparrow}\rangle$, i.e. $\gamma(C)$ and $\mu(D)$ are the least formal concept in $\mathcal{B}(I)$ whose extent includes C and the greatest one whose intent includes D. $\gamma(\{i\})$ and $\mu(\{j\})$, denoted simply by $\gamma(i)$ and $\mu(j)$, are called the object and attribute concept determined by $i \in X$ and $j \in Y$, respectively. Denote

$$\mathcal{I}_{C,D} = [\gamma(C), \mu(D)]. \tag{6}$$

Clearly, every interval in $\mathcal{B}(I)$ is of the form (6). Of particular importance are the intervals of the form

$$\mathcal{I}_{ij} = [\gamma(i), \mu(j)].$$

The following lemma describes crucial properties for understanding the role of intervals in from-below decompositions which are utilized below.

Lemma 1.

(a) *$\mathcal{I}_{C,D}$ is non-empty iff $C \times D \subseteq I$, i.e. if $I_{ij} = 1$ for every $i \in C$ and $j \in D$. In particular, $\mathcal{I}_{ij}$ is non-empty iff $I_{ij} = 1$.*
(b) *$\mathcal{I}_{C,D} = \{\langle E, F\rangle \in \mathcal{B}(I) \mid C \subseteq E, D \subseteq F\} = \{\langle E, F\rangle \in \mathcal{B}(I) \mid C^{\uparrow\downarrow} \subseteq E, D^{\downarrow\uparrow} \subseteq F\}$. In particular, $\mathcal{I}_{ij}$ is the set of all concepts that cover $\langle i, j\rangle$.*
(c) *If $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 1$ then $\mathcal{F}$ contains at least one concept in $\mathcal{I}_{ij}$.*

Proof. (a) As $\mathcal{I}_{C,D} \neq \emptyset$ iff $\gamma(C) \leq \mu(D)$, we need to check that $\gamma(C) \leq \mu(D)$ is equivalent to $C \times D \subseteq I$. Using basic properties of Galois connections [10], we get $\gamma(C) \leq \mu(D)$ iff $C^{\uparrow\downarrow} \subseteq D^{\downarrow}$ iff $D \subseteq C^{\uparrow}$ iff for every $y \in D$ we have $I_{ij} = 1$ for every $i \in C$, i.e. iff $C \times D \subseteq I$.

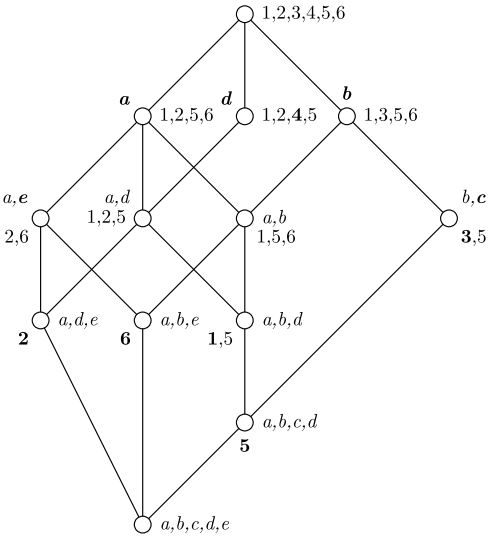
(b) We have $\langle E, F\rangle \in \mathcal{I}_{C,D}$ iff $\gamma(C) \leq \langle E, F\rangle \leq \mu(D)$ iff $C^{\uparrow\downarrow} \subseteq E$ and $D^{\downarrow\uparrow} \subseteq F$. Now, since $C^{\uparrow\downarrow}$ is the least extent ($^{\uparrow\downarrow}$-closed set of objects) containing C, $C^{\uparrow\downarrow} \subseteq E$ is equivalent to $C \subseteq E$; dually, $D^{\downarrow\uparrow} \subseteq F$ is equivalent to $D \subseteq F$, proving (b).

(c) Due to Observation 1 and (4), $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 1$ means that there exists $\langle C, D\rangle \in \mathcal{F}$ covering $\langle i, j\rangle$, whence (b) implies $\langle C, D\rangle \in \mathcal{I}_{ij}$. □

Remark 2. Interestingly, our problem may be reformulated as a certain graph-marking problem. Consider for a given matrix I and $\varepsilon \geq 0$ the line diagram of the concept lattice $\mathcal{B}(I)$ [7,10], i.e. a labeled Hasse diagram in which the nodes represent formal concepts of $\mathcal{B}(I)$ and the nodes representing concepts $\gamma(i)$ and $\mu(j)$ are labeled by "i" and "j" (the diagram is explained in Example 1). Due to Lemma 1(b) and (c), the problem to find a smallest set $\mathcal{F}$ of formal concepts for which $E(I, A_{\mathcal{F}} \circ B_{\mathcal{F}}) \leq \varepsilon$ (hence in case $\varepsilon = 0$ the problem to find a decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$ with a smallest possible $\mathcal{F}$) may be reformulated as the following graph-marking problem: In the line diagram of $\mathcal{B}(I)$, mark the smallest number of nodes such that with a possible exception of ε cases, every non-empty interval $\mathcal{I}_{ij}$ (object i, attribute j) contains at least one marked node. This geometric perspective is illustrated in Example 1 and is used in what follows.

Example 1. Consider the following Boolean matrix I, representing objects $1, \ldots, 6$ (rows) and attributes $a, \ldots, e$ (columns).

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$



The right part shows the line diagram of the concept lattice $\mathcal{B}(I)$ [10]. That is, the nodes and lines represent the concepts and the partial order $\leq$ of $\mathcal{B}(I)$. Every node represents the formal concept whose extent and intent consist of the objects and attributes attached to the node. Thus, the middle of the three nodes below the top node represents the formal concept $\langle\{1, 2, 4, 5\}, \{d\}\rangle$, while first node of the three below the top node represents $\langle\{1, 2, 5, 6\}, \{a\}\rangle$. There is a line from the node representing $\langle\{5\}, \{a, b, c, d\}\rangle$ up to the one representing $\langle\{1, 5\}, \{a, b, d\}\rangle$ because the first node is a direct predecessor of the second one. In general, $\langle C_1, D_1\rangle \leq \langle C_2, D_2\rangle$ iff there is a path going up from the node representing $\langle C_1, D_1\rangle$ to the one representing $\langle C_2, D_2\rangle$. Bold object and attribute names indicate object and attribute concepts. For instance, $\langle\{1, 5\}, \{a, b, d\}\rangle$ is the object concept $\gamma(1)$ because 1 appears in bold as a label at the corresponding node. Similarly, $\mu(a) = \langle\{1, 2, 5, 6\}, \{a\}\rangle$ is an attribute concept. Observe what is true in general, namely that the objects (attributes) attached to every node are just those that appear in bold on some downward (upward) path leading from the node. Hence, one can remove all the object and attribute labels except for the bold ones without any loss of information and obtain the so-called reduced labeling.

We can easily see from the diagram that the interval $\mathcal{I}_{4a}$ is empty, corresponding to $I_{4a} = 0$, while $\mathcal{I}_{1a}$ is non-empty,

$$\mathcal{I}_{1a} = \{\langle\{1, 5\}, \{a, b, d\}\rangle, \langle\{1, 2, 5\}, \{a, d\}\rangle, \langle\{1, 5, 6\}, \{a, b\}\rangle, \langle\{1, 2, 5, 6\}, \{a\}\rangle\},$$

corresponding to $I_{1a} = 1$. In view of Lemma 1 and Remark 2, the set

$$\mathcal{F} = \{\langle\{1, 5\}, \{a, b, d\}\rangle, \langle\{1, 2, 4, 5\}, \{d\}\rangle, \langle\{2, 6\}, \{a, e\}\rangle, \langle\{3, 5\}, \{b, c\}\rangle, \langle\{6\}, \{a, b, e\}\rangle\}$$

is a set of factor concepts of I, i.e. $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$, because if we mark the nodes representing the concepts in $\mathcal{F}$, every non-empty interval $\mathcal{I}_{ij}$ contains a marked node.

Clearly, a basic distinction may be made between the entries $\langle i, j\rangle$ of I containing 0 and those containing 1, i.e. between $I_{ij} = 0$ and $I_{ij} = 1$. Namely, the entries with 1 are those that need to be covered by factors to obtain an exact decomposition of I. In this sense, the entries of I containing 1 are sufficient. Some of these sufficient entries may, however, still be omitted and yet, the coverage of the remaining entries still guarantees a decomposition of I, resulting in further differentiation of the entries. In general, with a prescribed precision $\varepsilon \geq 0$ of decomposition, the differentiation may be based on the following question. Is it possible to identify a matrix $J \leq I$ with a small number of 1s with the property?:

$$\text{for any } \mathcal{F} \subseteq \mathcal{B}(I), \text{ if } J \leq A_{\mathcal{F}} \circ B_{\mathcal{F}} \text{ then } E(I, A_{\mathcal{F}} \circ B_{\mathcal{F}}) \leq \varepsilon \tag{7}$$

Note that (3) says that the coverage of all 1s in J guarantees the coverage of all 1s in I with a possible exception of ε cases. A Boolean matrix $J \leq I$ satisfying (7) that is minimal w.r.t. $\leq$ (i.e. the partial order defined by (3)) is called *ε-essential* for I. In what follows, we restrict to $\varepsilon = 0$ and call 0-essential matrices simply *essential*, or essential parts of I. We now describe the essential parts and utilize them in a new decomposition algorithm.

For $I \in \{0, 1\}^{n \times m}$ denote by $\mathcal{E}(I)$ the $n \times m$ Boolean matrix given by

$$(\mathcal{E}(I))_{ij} = 1 \text{ iff } \mathcal{I}_{ij} \text{ is non-empty and minimal w.r.t. } \subseteq,$$

where $\subseteq$ denotes set inclusion. Note that

$$\mathcal{I}_{ij} \subseteq \mathcal{I}_{i'j'} \text{ iff } \gamma(i') \leq \gamma(i) \text{ and } \mu(j) \leq \mu(j') \text{ iff } \{i\}^{\uparrow} \subseteq \{i'\}^{\uparrow} \text{ and } \{j\}^{\downarrow} \subseteq \{j'\}^{\downarrow} \tag{8}$$

and that a non-empty $\mathcal{I}_{ij}$ is minimal w.r.t. $\subseteq$ if it does not contain any other $\mathcal{I}_{i'j'}$, i.e. $\mathcal{I}_{ij} = \mathcal{I}_{i'j'}$ whenever $\mathcal{I}_{i'j'} \subseteq \mathcal{I}_{ij}$ for every i', j'. The next theorem asserts that $\mathcal{E}(I)$ is a unique essential part of I. The theorem concerns *clarified* matrices by which we mean matrices with no identical rows and columns. Clarification, i.e. removal of duplicate rows and columns, is a simple and useful preprocessing because it removes redundant information. Moreover, it is easy to see that the decompositions of I and its clarified I' are in one-to-one correspondence. In fact, as is readily seen from the proof, the sufficiency of $\mathcal{E}(I)$ holds for general matrices; and the assumption of clarification is used to prove uniqueness of $\mathcal{E}(I)$.

Theorem 2. *$\mathcal{E}(I)$ is a unique essential part of I, for every clarified I.*

Proof. We need to show (a) $\mathcal{E}(I) \leq I$ and that for any $\mathcal{F} \subseteq \mathcal{B}(I)$, if $\mathcal{E}(I) \leq A_{\mathcal{F}} \circ B_{\mathcal{F}}$ then $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$; and (b) if J satisfies (a) then $\mathcal{E}(I) \leq J$.

(a) $\mathcal{E}(I) \leq I$ follows from the definition of $\mathcal{E}(I)$ and Lemma 1(a). Let $\mathcal{E}(I) \leq A_{\mathcal{F}} \circ B_{\mathcal{F}}$ and assume by contradiction that there exists $\langle i, j \rangle$ for which $I_{ij} = 1$ and $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 0$. Consider any minimal interval $\mathcal{I}_{i'j'} \subseteq \mathcal{I}_{ij}$ (at least one exists). By definition of $\mathcal{E}(I)$, $(\mathcal{E}(I))_{i'j'} = 1$, whence also $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{i'j'} = 1$, from which it follows that there exists $\langle C, D \rangle \in \mathcal{F}$ which covers $\langle i', j' \rangle$. Due to Lemma 1(b), $\langle C, D \rangle \in \mathcal{I}_{i'j'}$ whence Lemma 1(b) and $\mathcal{I}_{i'j'} \subseteq \mathcal{I}_{ij}$ yield that $\langle C, D \rangle$ covers $\langle i, j \rangle$ and thus $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 1$, contradicting the assumption.

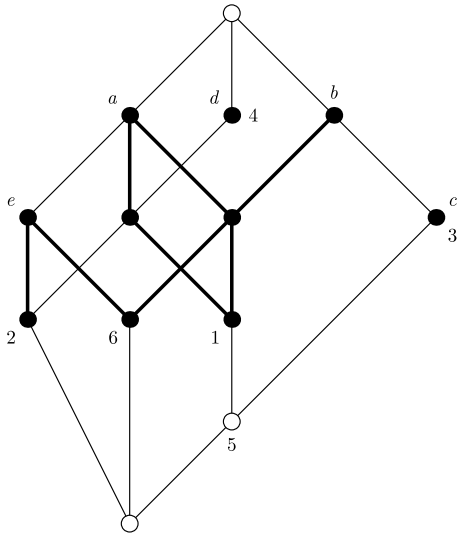
(b) By contradiction, assume that there exists $\langle i, j \rangle$ for which $\mathcal{E}(I)_{ij} = 1$ and $J_{ij} = 0$. Since $\mathcal{E}(I) \leq I$, we have $I_{ij} = 1$. To prove the assertion, it hence suffices to show that there exists a set $\mathcal{F} \subseteq \mathcal{B}(I)$ for which $J \leq A_{\mathcal{F}} \circ B_{\mathcal{F}}$ and yet $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 0$. For this purpose, consider an arbitrary $\mathcal{G} \subseteq \mathcal{B}(I)$ for which $J \leq A_{\mathcal{G}} \circ B_{\mathcal{G}}$ (clearly, such $\mathcal{G}$ exists). If $\mathcal{G}$ does not contain any concept from $\mathcal{I}_{ij}$, we are done by taking $\mathcal{F} = \mathcal{G}$, since then $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 0$ by Lemma 1(c). Otherwise, do the following for every $\langle C, D \rangle \in \mathcal{G} \cap \mathcal{I}_{ij}$: Remove $\langle C, D \rangle$ from $\mathcal{G}$ and add instead for every $\langle i', j' \rangle$ for which $J_{i'j'} = 1$ some formal concept $\langle C_{i'}, D_{j'} \rangle \in \mathcal{I}_{i'j'} - \mathcal{I}_{ij}$ and denote the resulting set of concepts by $\mathcal{F}$. Observe that such $\langle C_{i'}, D_{j'} \rangle$ always exists. Namely, since $J \leq I$, $J_{i'j'} = 1$ implies $I_{i'j'} = 1$ and hence due to Lemma 1(a), $\mathcal{I}_{i'j'}$ is non-empty. Now, $\mathcal{I}_{i'j'} \neq \mathcal{I}_{ij}$, since otherwise we have $\gamma(i) = \gamma(i')$ and $\mu(j) = \mu(j')$, and since I is clarified, this yields $i = i'$ and $j = j'$ which is impossible since we assumed $J_{ij} = 0$. Since $\mathcal{I}_{ij}$ is minimal w.r.t $\subseteq$ and different from $\mathcal{I}_{i'j'}$, we get the existence of $\langle C_{i'}, D_{j'} \rangle \in \mathcal{I}_{i'j'} - \mathcal{I}_{ij}$. Now, Lemma 1(c) implies that $J \leq A_{\mathcal{F}} \circ B_{\mathcal{F}}$ and yet $(A_{\mathcal{F}} \circ B_{\mathcal{F}})_{ij} = 0$, finishing the proof. □

Note that the part of the previous theorem claiming the sufficiency of covering the entries $\langle i, j \rangle$ with $\mathcal{E}(I)_{ij} = 1$ to obtain exact decomposition of I is noted in [9, p. 130] where the authors call such entries tight. Note also that an extension of the theorem to general, non-clarified matrices is simple but we omit it. Let us only note that for non-clarified matrices, essential matrices are not unique (they are easy to describe in terms of $\mathcal{E}(I)$ and the duplicate rows and columns). Denote the union of intervals $\mathcal{I}_{ij}$ corresponding to 1s in $\mathcal{E}(I)$ by $\mathcal{B}_{\mathcal{E}}(I)$, i.e.

$$\mathcal{B}_{\mathcal{E}}(I) = \bigcup \{\mathcal{I}_{ij} \mid (\mathcal{E}(I))_{ij} = 1\}. \tag{9}$$

Example 2. Consider again the matrix I of Example 1 and its concept lattice $\mathcal{B}(I)$, this time with a reduced labeling. The underlined entries in I are just the essential 1s, i.e. those with $\mathcal{E}(I)_{ij} = 1$.

$$\begin{pmatrix} \underline{1} & \underline{1} & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & \underline{1} \\ 0 & 1 & \underline{1} & 0 & 0 \\ 0 & 0 & 0 & \underline{1} & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & \underline{1} & 0 & 0 & \underline{1} \end{pmatrix}$$



For instance, while $I_{2a} = 1$, we have $\mathcal{E}(I)_{2a} = 0$ since the interval $\mathcal{I}_{2a}$ contains a different, smaller interval, namely $\mathcal{I}_{2e}$, whence $\mathcal{I}_{2a}$ is not minimal.

The bold part of the diagram corresponds to $\mathcal{B}_{\mathcal{E}}(I)$, i.e. the union of the six intervals $\mathcal{I}_{ij}$ for which $\mathcal{E}(I)_{ij} = 1$, cf. (9). One can now easily see that the set

$$\mathcal{F} = \{\langle\{1,5,6\},\{a,b\}\rangle, \langle\{1,2,4,5\},\{d\}\rangle, \langle\{2,6\},\{a,e\}\rangle, \langle\{3,5\},\{b,c\}\rangle\}$$

covers all the $\langle i, j\rangle$ with $\mathcal{E}(I)_{ij} = 1$. Namely, in the diagram of $\mathcal{B}(I)$, this is equivalent to the fact that each of the six bold intervals $\mathcal{I}_{ij}$ contains some formal concept in $\mathcal{F}$ (see also Remark 2 and Example 1). Due to Theorem 2, $\mathcal{F}$ covers all entries of I containing 1, whence $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$, i.e. $\mathcal{F}$ is a set of factor concepts of I. In particular, we have

$$A_{\mathcal{F}} = \begin{pmatrix} 1&1&0&0\\ 0&1&1&0\\ 0&0&0&1\\ 0&1&0&0\\ 1&1&0&1\\ 1&0&1&0 \end{pmatrix} \quad \text{and} \quad B_{\mathcal{F}} = \begin{pmatrix} 1&1&0&0&0\\ 0&0&0&1&0\\ 1&0&0&0&1\\ 0&1&1&0&0 \end{pmatrix}$$

An interesting property of $\mathcal{E}(I)$ whose further elaboration we utilize in the new decomposition algorithm in Section 4 is contained in the following theorem showing how factorizations of I may be obtained from factorizations of $\mathcal{E}(I)$.

Theorem 3. *Let $\mathcal{G} \subseteq \mathcal{B}(\mathcal{E}(I))$ be a set of factor concepts of $\mathcal{E}(I)$, i.e. $\mathcal{E}(I) = A_{\mathcal{G}} \circ B_{\mathcal{G}}$. Then every set $\mathcal{F} \subseteq \mathcal{B}(I)$ containing for each $\langle C, D\rangle \in \mathcal{G}$ at least one concept from $\mathcal{I}_{C,D}$ is a set of factor concepts of I, i.e. $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.*

Proof. Let for $\langle C, D\rangle \in \mathcal{G}$ denote by $\langle E, F\rangle_{\langle C,D\rangle}$ a concept in $\mathcal{F} \cap \mathcal{I}_{C,D}$ which exists according to the assumption. Due to Lemma 1(a), $C \subseteq E$ and $D \subseteq F$. Since this is true for every $\langle C, D\rangle \in \mathcal{G}$, we readily obtain $A_{\mathcal{G}} \circ B_{\mathcal{G}} \leq A_{\mathcal{F}} \circ B_{\mathcal{F}}$. The assumption $\mathcal{E}(I) = A_{\mathcal{G}} \circ B_{\mathcal{G}}$ now yields $\mathcal{E}(I) \leq A_{\mathcal{F}} \circ B_{\mathcal{F}}$. As $\mathcal{E}(I)$ is an essential part of I, we get $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$, finishing the proof. □

Remark 3. Clearly, Theorem 3 may be generalized to arbitrary factorizations of $\mathcal{E}(I)$. Namely, suppose $\mathcal{E}(I) = A \circ B$ for some $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ and let $C_l = \{i \mid A_{il} = 1\}$, $D_l = \{j \mid B_{lj} = 1\}$ for each $l = 1, \ldots, k$. Then every set $\mathcal{F} \subseteq \mathcal{B}(I)$ containing at least one concept from $\mathcal{I}_{C_l,D_l}$ for each $l = 1, \ldots, k$, is a set of factor concepts of I. Namely, each $\langle C_l, D_l\rangle$ may be extended to a formal concept of $\mathcal{E}(I)$, the collection $\mathcal{G}$ of all such formal concepts satisfies $\mathcal{E}(I) = A_{\mathcal{G}} \circ B_{\mathcal{G}}$ and then Theorem 3 applies.

Theorem 3 implies that the rank of $\mathcal{E}(I)$ provides an upper bound on the rank of I:

Theorem 4. *For every Boolean matrix I we have* $\mathrm{rank}_{\mathrm{B}}(I) \leq \mathrm{rank}_{\mathrm{B}}(\mathcal{E}(I))$.

Proof. Let $k = \mathrm{rank}_{\mathrm{B}}(\mathcal{E}(I))$. Due to [4], we may safely assume the existence of $\mathcal{G} \subseteq \mathcal{B}(\mathcal{E}(I))$ and $A_{\mathcal{G}} \in \{0,1\}^{n\times k}$ and $B_{\mathcal{G}} \in \{0,1\}^{k\times m}$ such that $\mathcal{E}(I) = A_{\mathcal{G}} \circ B_{\mathcal{G}}$. Consider any $\mathcal{F} \subseteq \mathcal{B}(I)$ containing for each $\langle C, D\rangle \in \mathcal{G}$ exactly one formal concept in the interval $\mathcal{I}_{C,D}$ of $\mathcal{B}(I)$. Note that such $\mathcal{F}$ exists since for every $\langle C, D\rangle \in \mathcal{G}$ we have $C \times D \subseteq \mathcal{E}(I)$ because $\langle C, D\rangle$ is a formal concept of $\mathcal{E}(I)$, and hence $\mathcal{E}(I) \leq I$ implies $C \times D \subseteq I$. Due to Lemma 1(a), $\mathcal{I}_{C,D}$ is a non-empty interval in $\mathcal{B}(I)$. According to Theorem 3, $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$. Now, clearly, $\mathrm{rank}_{\mathrm{B}}(I) \leq |\mathcal{F}| \leq |\mathcal{G}| = \mathrm{rank}_{\mathrm{B}}(\mathcal{E}(I))$, finishing the proof. □

Remark 4. The estimation by Theorem 4 is not tight. Namely, as one may check,

$$\text{for } I = \begin{pmatrix} 10111\\ 01101\\ 01001\\ 10110 \end{pmatrix} = \begin{pmatrix} 110\\ 011\\ 001\\ 100 \end{pmatrix} \circ \begin{pmatrix} 10110\\ 00101\\ 01001 \end{pmatrix}, \text{ we have } \mathcal{E}(I) = \begin{pmatrix} 00001\\ 00100\\ 01000\\ 10010 \end{pmatrix}.$$

While we see that $\mathrm{rank}_{\mathrm{B}}(I) \leq 3$ (in fact, the rank equals 3), one may easily check that $\mathrm{rank}_{\mathrm{B}}(\mathcal{E}(I)) = 4$.

Another issue connected to $\mathcal{E}(I)$ is the following. Due to Theorem 2 of [4], optimal exact decompositions may be obtained by using as factors the formal concepts of $\mathcal{B}(I)$. The next theorem shows that even more restricted formal concepts are sufficient, namely those in $\mathcal{B}_{\mathcal{E}}(I)$.

Theorem 5. *For every I there exists $\mathcal{F} \subseteq \mathcal{B}_{\mathcal{E}}(I)$ with $|\mathcal{F}| = \mathrm{rank}_{\mathrm{B}}(I)$ for which $A_{\mathcal{F}} \circ B_{\mathcal{F}} = I$.*

Proof. Due to Theorem 2 of [4], there exists $\mathcal{F} \subseteq \mathcal{B}(I)$ with $|\mathcal{F}| = \mathrm{rank}_B(I)$ for which $A_{\mathcal{F}} \circ B_{\mathcal{F}} = I$. It is sufficient to show that $\mathcal{F} \subseteq \mathcal{B}_{\mathcal{E}}(I)$. Suppose by contradiction that there exists $\langle C, D\rangle \in \mathcal{F} - \mathcal{B}_{\mathcal{E}}(I)$, i.e. $\langle C, D\rangle$ does not belong to any minimal $\mathcal{I}_{i,j}$. Since $\mathcal{F}$ covers I, and hence also $Ess(I)$, for every $\langle i', j'\rangle$ with $\mathcal{E}(I)_{i'j'} = 1$, there exists a formal concept $\langle C', D'\rangle \in \mathcal{F}$ which covers $\langle i', j'\rangle$. By Lemma 1(b), $\langle C', D'\rangle \in \mathcal{I}_{i',j'}$ and hence $\langle C', D'\rangle \neq \langle C, D\rangle$. For $\mathcal{F}' = \mathcal{F} - \{\langle C, D\rangle\}$ we thus have $A_{\mathcal{F}'} \circ B_{\mathcal{F}'} \geq \mathcal{E}(I)$ hence $A_{\mathcal{F}'} \circ B_{\mathcal{F}'} = I$ by Theorem 2. We obtained a set $\mathcal{F}'$ of factors of I with $|\mathcal{F}'| = |\mathcal{F}| - 1 < \mathrm{rank}_B(I)$, a contradiction. □

Remark 5. Theorem 1, dealing with approximate from-below factorizations, and Theorem 5, dealing with a stronger restriction of the search space for optimal factorizations, provide two different improvements of Theorem 2 of [4]. The following example shows that the natural common generalization of Theorems 1 and 5 does not hold. Namely, the generalization would result from Theorem 1 by replacing the condition $\mathcal{F} \subseteq \mathcal{B}(I)$ by $\mathcal{F} \subseteq \mathcal{B}_{\mathcal{E}}(I)$. Consider the following matrix I and the concept $\langle C, D\rangle \in \mathcal{B}(I)$:

$$I = \begin{pmatrix} 1 & \underline{1} & 0 & 0 \\ 1 & 0 & \underline{1} & 0 \\ 1 & 0 & 0 & \underline{1} \end{pmatrix}, \qquad \langle C, D\rangle = \{\{1,2,3\}, \{1\}\}.$$

For $\mathcal{G} = \{\langle C, D\rangle\}$ and $A = A_{\mathcal{G}}$ and $B = B_{\mathcal{G}}$ we have $A \circ B \leq I$ and $E(I, A \circ B) = 3$. Now, $\langle C, D\rangle \notin \mathcal{B}_{\mathcal{E}}(I)$ and there is no one-element subset $\mathcal{F} \subseteq \mathcal{B}_{\mathcal{E}}(I)$ for which $E(I, A_{\mathcal{F}} \circ B_{\mathcal{F}}) \leq E(A \circ B, I)$, i.e. the generalization does not hold.

The next lemma is easy to see and shows that $\mathcal{E}(I)$ can be computed easily:

Lemma 2. *$\mathcal{E}(I)_{ij} = 1$ if an only if the following conditions are fulfilled:*

(a) $I_{ij} = 1$;
(b) *for every i' with $\{i'\}^{\uparrow} \subset \{i\}^{\uparrow}$ we have $I_{i'j} = 0$ (i.e., no i' whose row is contained in the row of i contains j);*
(c) *for every j' with $\{j'\}^{\downarrow} \subset \{j\}^{\downarrow}$ we have $I_{i'j} = 0$ (i.e., no j' whose column is contained in the column of j contains i).*

4. GreEss: a new BMF algorithm

The new BMF algorithm described in this section is primarily designed for AFP but can also be used for DBP (see Section 2). The algorithm is based on the properties of essential parts $\mathcal{E}(I)$ of Boolean matrices I established above. Two important features of essential parts we start with are the following. First, since $\mathcal{E}(I)$ represents the entries of I with 1 whose coverage by arbitrary factors guarantees an exact decomposition of I by these factors, $\mathcal{E}(I)$ provides us with information about where to focus in the search for factors of I. Second, since the number $||\mathcal{E}(I)||$ of 1s in $\mathcal{E}(I)$ tends to be significantly smaller than the number $||I||$ of 1s in I (see Section 5.1), covering the 1s in $\mathcal{E}(I)$ tends to be simpler than covering the 1s in the original matrix I. Note also that due to Lemma 2, $\mathcal{E}(I)$ is computed easily.

In particular, we build upon the following idea. We intend to form a collection $\mathcal{G}$ of possibly overlapping groups of essential 1s, i.e. 1s in $\mathcal{E}(I)$. These are considered as "seeds" for finding factors of I in that each group $g \in \mathcal{G}$ is to be covered by a factor—a formal concept $\langle C, D\rangle \in \mathcal{B}(I)$ (taking formal concepts of I as factors is optimal for AFP due to Theorem 1). Since each concept $\langle C, D\rangle \in \mathcal{B}(I)$ corresponds to a maximal rectangle in I, every group g covered by $\langle C, D\rangle$ may be extended to a maximal rectangle in $\mathcal{E}(I)$, i.e. to a formal concept in $\mathcal{B}(\mathcal{E}(I))$, which will still be covered by $\langle C, D\rangle$. Therefore, reasonable candidates for the groupings $\mathcal{G}$ are sets of formal concepts of $\mathcal{E}(I)$, i.e. $\mathcal{G} \subseteq \mathcal{B}(\mathcal{E}(I))$.

Considering $\mathcal{G} \subseteq \mathcal{B}(\mathcal{E}(I))$ is a reasonable strategy in view of Theorem 3, which suggests to compute a factor set $\mathcal{G}$ of $\mathcal{E}(I)$ and then compute from $\mathcal{G}$ a factor set $\mathcal{F}$ of I. Our algorithm utilizes an improvement of this idea. Namely, we compute a set $\mathcal{G} \subseteq \mathcal{B}(\mathcal{E}(I))$ which need not be a factorization of $\mathcal{E}(I)$, i.e. may be smaller and satisfy $A_{\mathcal{G}} \circ B_{\mathcal{G}} < \mathcal{E}(I)$. Nevertheless, $\mathcal{G}$ still has the following important property: If for each $\langle C, D\rangle \in \mathcal{G}$ we pick exactly one concept $c_{\langle C,D\rangle}$ in the interval $\mathcal{I}_{C,D}$ of $\mathcal{B}(I)$, then no matter how we pick, the resulting $\mathcal{F} = \{c_{\langle C,D\rangle} \mid \langle C, D\rangle \in \mathcal{G}\}$ provides a factorization of I, i.e. $A_{\mathcal{F}} \circ B_{\mathcal{F}} = I$. The pseudocode of our algorithm, called GreEss, is described in Algorithms 1 and 2. GreEss is designed for AFP, i.e. it takes as its input a matrix I and $\varepsilon \geq 0$ and produces a set $\mathcal{F}$ of formal concepts of I for which $||I - A_{\mathcal{F}} \circ B_{\mathcal{F}}|| \leq \varepsilon$. Hence, for $\varepsilon = 0$ the algorithm produces an exact decomposition of I.

We now provide a detailed description of this algorithm and justify its correctness. We shall need the following lemma.

Lemma 3. *Let $C \times D \subseteq I$, i.e. $I_{ij} = 1$ for every $i \in C$ and $j \in D$, let $J = I \cap (D^{\downarrow_I} \times C^{\uparrow_I})$. Then $\mathcal{I}_{C,D} = \mathcal{B}(D^{\downarrow_I}, C^{\uparrow_I}, J)$.*

Proof. Let $\langle E, F\rangle \in \mathcal{I}_{C,D}$. Since the least and the greatest concepts in $\mathcal{I}_{C,D}$ are $\langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle$ and $\langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle$, respectively, we have $E \subseteq D^{\downarrow_I}$ and $F \subseteq C^{\uparrow_I}$. Since $E^{\uparrow_I} = F$ and $F^{\downarrow_I} = E$ and since J is the restriction of I to $D^{\downarrow_I} \times C^{\uparrow_I}$, we clearly have $E^{\uparrow_J} = F$ and $F^{\downarrow_J} = E$, establishing $\langle E, F\rangle \in \mathcal{B}(D^{\downarrow_I}, C^{\uparrow_I}, J)$.

Conversely, let $\langle E, F\rangle \in \mathcal{B}(D^{\downarrow_I}, C^{\uparrow_I}, J)$. Clearly, $C \times C^{\uparrow_I} \subseteq I$ and $D^{\downarrow_I} \times D \subseteq I$. Since $C \times D \subseteq I$, we have $C \subseteq D^{\downarrow_I}$ and $D \subseteq C^{\uparrow_I}$. Therefore, since J is a restriction of I, $C \times C^{\uparrow_I} \subseteq J$ and $D^{\downarrow_I} \times D \subseteq J$. Since $F \subseteq C^{\uparrow_I}$ and $E \subseteq D^{\downarrow_I}$, we obtain $C \subseteq C^{\uparrow_I\downarrow_J} \subseteq F^{\downarrow_J} = E$ and $D \subseteq D^{\downarrow_I\uparrow_J} \subseteq E^{\uparrow_J} = F$. It remains to verify $E^{\uparrow_I} = F$ and $F^{\downarrow_I} = E$. $E^{\uparrow_I} \supseteq F$ follows directly from

Algorithm 1: GreEss.

Input: Boolean matrix I and $\varepsilon \geq 0$
Output: set $\mathcal{F}$ of factors for which $||I - A_{\mathcal{F}} \circ B_{\mathcal{F}}|| \leq \varepsilon$

1 $\mathcal{G} \leftarrow$ ComputeIntervals(I)
2 $U \leftarrow \{\langle i, j\rangle \,|\, I_{ij} = 1\}$; $\mathcal{F} \leftarrow \emptyset$
3 **while** $|U| > \varepsilon$ **do**
4 $\quad s \leftarrow 0$
5 $\quad$ **foreach** $\langle C, D\rangle \in \mathcal{G}$ **do**
6 $\quad\quad J \leftarrow I \cap (D^{\downarrow_I} \times C^{\uparrow_I})$
7 $\quad\quad F \leftarrow \emptyset$; $s_{\langle C,D\rangle} \leftarrow 0$
8 $\quad\quad$ **while** *exists* $j \in C^{\uparrow_I} - F$ *s.t.* $|(F \cup \{j\})^{\downarrow_J} \times (F \cup \{j\})^{\downarrow_J\uparrow_J} \cap U| > s_{\langle C,D\rangle}$ **do**
9 $\quad\quad\quad$ select j maximizing $|(F \cup \{j\})^{\downarrow_J} \times (F \cup \{j\})^{\downarrow_J\uparrow_J} \cap U|$
10 $\quad\quad\quad F \leftarrow (F \cup \{j\})^{\downarrow_J\uparrow_J}$; $E \leftarrow (F \cup \{j\})^{\downarrow_J}$
11 $\quad\quad\quad s_{\langle C,D\rangle} \leftarrow |E \times F \cap U|$
12 $\quad\quad$ **end**
13 $\quad\quad$ **if** $s_{\langle C,D\rangle} > s$ **then**
14 $\quad\quad\quad \langle E', F'\rangle \leftarrow \langle E, F\rangle$
15 $\quad\quad\quad \langle C', D'\rangle \leftarrow \langle C, D\rangle$
16 $\quad\quad\quad s \leftarrow s_{\langle C,D\rangle}$
17 $\quad\quad$ **end**
18 $\quad$ **end**
19 $\quad$ **add** $\langle E', F'\rangle$ **to** $\mathcal{F}$
20 $\quad$ **remove** $\langle C', D'\rangle$ **from** $\mathcal{G}$
21 $\quad U \leftarrow U - E' \times F'$
22 **end**
23 **return** $\mathcal{F}$

Algorithm 2: ComputeIntervals.

Input: Boolean matrix I
Output: Set $\mathcal{G} \subseteq \mathcal{B}(\mathcal{E}(I))$
1 $\mathcal{E} \leftarrow \mathcal{E}(I)$; $U \leftarrow \{\langle i, j\rangle \,|\, \mathcal{E}_{ij} = 1\}$; $\mathcal{G} \leftarrow \emptyset$ **while** $U \neq \emptyset$ **do**
2 $\quad D \leftarrow \emptyset$; $s \leftarrow 0$
3 $\quad$ **while** *exists* $j \notin D$ *with* $|((D \cup \{j\})^{\downarrow\mathcal{E}})^{\uparrow_I\downarrow_I} \times ((D \cup \{j\})^{\downarrow\mathcal{E}\uparrow\mathcal{E}})^{\downarrow_I\uparrow_I} \cap U| > s$ **do**
4 $\quad\quad$ select j which maximizes $|((D \cup \{j\})^{\downarrow\mathcal{E}})^{\uparrow_I\downarrow_I} \times ((D \cup \{j\})^{\downarrow\mathcal{E}\uparrow\mathcal{E}})^{\downarrow_I\uparrow_I} \cap U|$
5 $\quad\quad D \leftarrow (D \cup \{j\})^{\downarrow\mathcal{E}\uparrow\mathcal{E}}$; $C \leftarrow (D \cup \{j\})^{\downarrow\mathcal{E}}$
6 $\quad\quad s \leftarrow |C^{\uparrow_I\downarrow_I} \times D^{\downarrow_I\uparrow_I} \cap U|$
7 $\quad$ **end**
8 $\quad$ **add** $\langle C, D\rangle$ **to** $\mathcal{G}$
9 $\quad U \leftarrow U - C^{\uparrow_I\downarrow_I} \times D^{\downarrow_I\uparrow_I}$
10 **end**
11 **return** $\mathcal{G}$

$E^{\uparrow_J} = F$ and $J \leq I$. On the other hand, if $j \in E^{\uparrow_I}$ then since $C \subseteq E$ and hence $E^{\uparrow_I} \subseteq C^{\uparrow_I}$, we have $j \in C^{\uparrow_I}$. This means that j is an attribute of the context $\langle D^{\downarrow_I}, C^{\uparrow_I}, J\rangle$ and since J is a restriction of I, $j \in E^{\uparrow_I}$ implies $j \in E^{\uparrow_J} = F$, proving $E^{\uparrow_I} \subseteq F$. $F^{\downarrow_I} = E$ can be verified in a similar manner. □

GreEss first calls ComputeIntervals (described in detail below) which computes the aforementioned $\mathcal{G}$. In the loop 3–22, GreEss is picking concepts $\langle E', F'\rangle$ from intervals $\mathcal{I}_{C,D}$, for $\langle C, D\rangle \in \mathcal{G}$, in such a way that at most one concept is selected from every interval, until the size of the collection U of uncovered entries of I is less than ε. In 5–18, the yet unused intervals $\mathcal{I}_{\langle C,D\rangle}$ are searched in a greedy manner as follows. We start by the attribute concepts $\gamma(j) \in \mathcal{I}_{C,D}$ and try to extend them greedily by adding attributes (loop 8–12). Due to Lemma 3, restricting to J and using $^{\uparrow_J}$ and $^{\downarrow_J}$ guarantees that we do not leave $\mathcal{I}_{C,D}$. Note that the greedy extension by attributes is inspired by [4]. A possible extension of the so far best found $\langle E, F\rangle$ is accepted (l. 8) if the extended concept $\langle (F \cup \{j\})^{\downarrow_J}, (F \cup \{j\})^{\downarrow_J\uparrow_J}\rangle$ covers a larger number of entries of U than the number $s_{\langle C,D\rangle}$ covered by $\langle E, F\rangle$. The best such concept of all the unused intervals, denoted $\langle E', F'\rangle$, is then added to $\mathcal{F}$, the interval is marked as used by removing $\langle C', D'\rangle$ from $\mathcal{G}$ (l. 20), and the entries covered by $\langle E', F'\rangle$ are removed from U (l. 21). ComputeIntervals computes $\mathcal{E}(I)$, computes and adds to $\mathcal{G}$ the concepts of $\mathcal{B}(\mathcal{E}(I))$ computed in a greedy manner by starting from attribute concepts and adding attributes similarly as above. The difference, however, is that the entries of U that are considered as covered by a candidate $\langle C, D\rangle$ are those in $C^{\uparrow_I\downarrow_I} \times D^{\downarrow_I\uparrow_I}$, i.e. not only those of $C \times D$. This is possible because the factors for I are selected from the intervals $\mathcal{I}_{C,D}$ in GreEss and due to Lemma 1(b), every such factor covers $C^{\uparrow_I\downarrow_I} \times D^{\downarrow_I\uparrow_I}$. This is why $\mathcal{G}$ need not be a factorization of $\mathcal{E}(I)$, i.e. may be smaller. These considerations tell us:

Theorem 6. GreEss *is correct and provides a from-below approximation of* I.

Example 3. We now demonstrate GreEss. Suppose the input data consists of the matrix I from Examples 1 and 2 and that $\varepsilon = 0$. GreEss first calls ComputeIntervals, which computes the matrix $\mathcal{E}(I)$ (see Example 2) and initializes U by marking as uncovered all entries with $\mathcal{E}(I)_{ij} = 1$. The factors of $\mathcal{G}$ are computed in the loop 2–11 by the extension explained above, i.e. by selecting consecutively attribute j whose addition to the so-far computed $\langle C, D\rangle$ maximizes the overlap with U, until such extension is possible. The extension starts with $D = \emptyset$. In 4–8, one first selects $j = a$. Namely, we have $((D \cup \{a\})^{\downarrow\varepsilon})^{\uparrow_I\downarrow_I} = \{1, 5\}$ and $((D \cup \{a\})^{\downarrow\varepsilon\uparrow\varepsilon})^{\downarrow_I\uparrow_I} = \{a, b, d\}$, hence the size of the overlap $\{\langle 1, a\rangle, \langle 1, b\rangle\}$ with U is 2. Since for $j = b, c, d, e$, the overlap is not larger, one selects the attribute a and puts $C = (D \cup \{a\})^{\downarrow\varepsilon} = \{1\}$ and $D = (D \cup \{a\})^{\downarrow\varepsilon\uparrow\varepsilon}) = \{a, b\}$. With the updated $\langle C, D\rangle$, i.e. attempt to extend it in the next run of 4–8 by the best attribute $j \notin D$ provided the extension enlarges the overlap with U, until such extension is possible. In our case, no further extension is possible, one exits the loop 4–8, adds $\langle C, D\rangle = \langle\{1\}, \{a, b\}\rangle$ to $\mathcal{G}$ (line 9) and removes from U the set $\{1, 5\} \times \{a, b, d\}$ (line 10). One then goes back to line 3 and repeats the above procedure until U is empty, i.e. all entries in $\mathcal{E}(I)$ are covered. $\mathcal{G}$ then contains $\langle\{1\}, \{a, b\}\rangle$, $\langle\{2, 6\}, \{e\}\rangle$, $\langle\{1, 6\}, \{b\}\rangle$, $\langle\{3\}, \{c\}\rangle$ and $\langle\{4\}, \{d\}\rangle$.

GreEss then continues by initializing U and $\mathcal{F}$ in line 2. While U is non-empty, one selects for every $\langle C, D\rangle \in \mathcal{G}$ the best formal concept $\langle E, F\rangle$ in the interval $\mathcal{I}_{C,D}$. The selection of $\langle E, F\rangle$ is made in loop 8–12 and follows the same logic as the above extension in ComputeIntervals. In particular, the first $\langle C, D\rangle$ in $\mathcal{G}$ is $\langle\{1\}, \{a, b\}\rangle$ and with initially setting $F = \emptyset$ in line 7, the extension ends up with $\langle E, F\rangle = \langle\{1, 5, 6\}, \{a, b\}\rangle$. We compute the $\langle E, F\rangle$s in the same manner for all the $\langle C, D\rangle \in \mathcal{G}$ and obtain as $\langle E, F\rangle$ the rectangles $\langle\{1, 5, 6\}, \{a, b\}\rangle$, $\langle\{2, 6\}, \{a, e\}\rangle$, $\langle\{1, 5, 6\}, \{a, b\}\rangle$, $\langle\{3, 4\}, \{b, c\}\rangle$, $\langle\{1, 2, 4, 5\}, \{d\}\rangle$. Of these, $\langle\{1, 5, 6\}, \{a, b\}\rangle$ has the largest overlap with U, is added to $\mathcal{F}$ (line 19), its corresponding $\langle C, D\rangle = \langle\{1\}, \{a, b\}\rangle$ is removed from $\mathcal{G}$ (line 20) to ensure that the interval $\mathcal{I}_{C,D}$ is not searched again, and the pairs in $\{1, 5, 6\} \times \{a, b\}$ are removed from U. The above process is then repeated until U is empty, producing eventually the set $\mathcal{F}$ consisting of $\langle\{1, 5, 6\}, \{a, b\}\rangle$,$\langle\{1, 2, 4, 5\}, \{d\}\rangle$, $\langle\{2, 6\}, \{a, e\}\rangle$ and $\langle\{3, 5\}, \{b, c\}\rangle$.

We now derive an upper bound of the worst case time complexity of GreEss. Let us first note that n, m, and $||I||$ denote the number of rows, columns, and entries with 1 of the input matrix I. Moreover, we shall assume $\max(n, m) \leq ||I||$ which is a reasonable condition somewhat simplifying the analysis. Furthermore, observe that computing $^{\uparrow_J}$ or $^{\downarrow_J}$ for $J \in \{0, 1\}^{p\times q}$ is done in time $O(pq)$ for a general argument, while $\{i\}^{\uparrow_J}$ and $\{j\}^{\downarrow_J}$ takes time $O(q)$ and $O(p)$, respectively. Consider first ComputeIntervals. It first computes $\mathcal{E}(I)$, which may be done in time $O(n^2m^2)$, since it suffices to repeat for every of the nm entries of I the test given by Lemma 2 and since the test may be done in time $O(nm)$. The loop 2–11 repeats at most $||\mathcal{E}(I)|| = O(||I||)$ times. Inside this loop the most critical is the number of executions of the innermost cycle in the while loop 4–8. Most expensive in that cycle is computing $((D \cup \{j\})^{\downarrow\varepsilon\uparrow\varepsilon})^{\downarrow_I\uparrow_I}$, which takes time $O(nm)$. The while loop 4–8 proceeds in two cycles. The outer one proceeds at most m times since no more than m attributes may eventually be added when extending the rectangle under construction. Within the jth execution of the outer cycle, the inner cycle is executed at most $m + 1 - j$ times since this is the number of remaining candidate attributes for extending the so far computed rectangle $\langle C, D\rangle$. Hence the innermost cycle is executed $\sum_{j=1}^{m}(m + 1 - j) = O(m^2)$ times which along with the at most $O(nm)$ steps within each execution of the innermost cycle yields that the body of loop 2–11 requires time $O(nm^3)$. Altogether, loop 2–11 takes time $O(||I||nm^3)$. Since $\max(n, m) \leq ||I||$, the time for ComputeIntervals itself is $O(n^2m^2) + O(||I||nm^3) = O(||I||nm^3)$.

After ComputeIntervals, GreEss executes at most $||I|| - \varepsilon = O(||I||)$ times the loop 3–22 within which it executes at most $|\mathcal{G}| = O(||I||)$ times the loop 5–17. Within the loop 5–17, a construction of rectangle $\langle E, F\rangle$ by extension proceeds analogously as in 2–11 of ComputeIntervals and (for the same reasons as above) takes time $O(nm^3)$. The loop 3–22 thus runs in time $O(||I||^2nm^3)$. To sum up, GreEss has a polynomial upper bound of time complexity, namely $O(||I||nm^3) + O(||I||^2nm^3) = O(||I||^2nm^3)$. An interesting problem, presumably combinatorially rather involved, is to obtain a tighter upper bound since in several steps, our estimate is loose.

5. Experimental evaluation

In this section, we provide an experimental evaluation of GreEss and its comparison with five main existing BMF algorithms, described in Section 5.1. We use synthetic and real datasets in a scenario similar to that used in [20] and other papers on BMF.

5.1. Algorithms and datasets

Algorithms We now describe the algorithms used in our comparison.

Tiling is proposed in [11] for MTP (Remark 1). To find a small tiling, the algorithm iteratively computes a tile that covers the largest number of still uncovered 1s of all the tiles in I, until all 1s in I are covered, implementing thus the well-known greedy set cover algorithm. It follows from Remark 1 that Tiling provides a from-below factorization of I.

Asso [20], probably the most discussed BMF algorithm in the data mining literature, first computes an $m \times m$ Boolean matrix A in which $A_{ij} = 1$ if the confidence of rule $\{i\} \Rightarrow \{j\}$ exceeds a parameter τ. The rows of A are then used as candidates for the rows of the factor–attribute matrix. The actual rows are selected using a greedy approach using function *cover* that rewards with weight w^+ the decrease of error E_u and penalizes with weight w^- the increase of E_o that is due to a given row of A. Asso is designed to solve the DBP (Section 2) and commits both types of errors, E_u and E_o. It thus provides general factorizations which need not be the from-below ones.

Table 1
Synthetic data.

Dataset	k	Dens A	Dens B	Dens I	Avg $\|\|\mathcal{E}(I)\|\|$	Avg $\|\|\mathcal{E}(I)\|\|/\|\|I\|\|$
Set A1	20	0.11	0.05	0.1	2268±188	0.0434±0.0043
Set A2	20	0.11	0.10	0.2	1623±156	0.0164±0.0019
Set A3	20	0.15	0.12	0.3	736±89	0.0048±0.0006
Set A4	20	0.17	0.15	0.4	417±103	0.0021±0.0005
Set B1	30	0.07	0.05	0.1	1450±128	0.0291±0.0031
Set B2	30	0.11	0.07	0.2	601±152	0.0058±0.0015
Set B3	30	0.12	0.01	0.3	613±285	0.0040±0.0019
Set B4	30	0.14	0.12	0.4	1811±990	0.0091±0.0049
Set C1	40	0.07	0.04	0.1	791±124	0.0150±0.0024
Set C2	40	0.11	0.05	0.2	2714±749	0.0275±0.0076
Set C3	40	0.13	0.07	0.3	7018±2054	0.0457±0.0132
Set C4	40	0.13	0.10	0.4	17783±4667	0.0872±0.0223

Hyper [31] produces from a Boolean matrix I a set $\mathcal{F}$ of rectangles (called hyperrectangles by the authors) in I that provide an exact decomposition of I. Hyper attempts to find $\mathcal{F}$ with the smallest minimal cost for which Hyper employs the minimal description length principle (MDLP) in that for the cost of a rectangle $\langle C, D\rangle$ (C and D being sets of rows and columns) $|C| + |D|$ and the cost of $\mathcal{F}$ is the sum of the costs of the rectangles in $\mathcal{F}$. Instead of the number of factors themselves, the primary concern for Hyper is the description length of the factors. This makes the problem different from DBP and AFP. However, the fact that a small number of rectangles (factors) tends to have small description length and the claims in [31] that Hyper provides a good summarization of the data makes Hyper a relevant decomposition algorithm. In particular, Hyper generates the set $\mathcal{F}$ of factor rectangles by first computing for a given support α supplied by a user the set F_α of all α-frequent itemsets from I. Then, it generates the factor rectangles from the set $\mathcal{C}_\alpha$ of rectangles corresponding to the itemsets in F_α enriched by all the singleton itemsets. For each rectangle in $\mathcal{C}_\alpha$, Hyper finds the minimum cost rectangle by adding rows from a list of rows sorted by their coverage of the yet uncovered data. The rectangle with minimum cost over all the itemsets is then added to $\mathcal{F}$. Even though, as emphasized by the authors, Hyper runs in time polynomial w.r.t. $\mathcal{C}_\alpha$, its time is not polynomial w.r.t. to the input size because the size of $\mathcal{C}_\alpha$ may be exponential w.r.t. to the input size—an important fact not mentioned by the authors.

GreConD [4, Algorithm 2] performs a particular greedy search "on demand" for formal concepts of the input matrix I and utilizes these concepts to produce matrices $A_\mathcal{F}$ and $B_\mathcal{F}$. This search improves the idea of the basic set cover algorithm in that it avoids the necessity to compute all formal concepts of I. This results in orders of magnitude time saving while retaining the quality of decomposition. Note also that [4, Algorithm 1] implements the basic set cover algorithm, thus proceeds essentially as the Tiling algorithm but is considerably more time efficient because the maximal rectangles are computed in advance. GreConD is a from-below decomposition algorithm, designed to compute exact decompositions. When stopped after computing the first k factors or after the error E does not exceed ε, GreConD provides approximate solutions to both DBP and AFP (Section 2).

PaNDa (Patterns in Noisy Datasets) [16] is designed to solve a modification of DBP which consists in employing the MDLP. In particular, for a given I and k, the problem is to extract from I a set $\mathcal{F}$ of k patterns (pairs $\langle C, D\rangle$ of sets of rows and columns) minimizing the cost of $\mathcal{F}$. The cost of $\mathcal{F}$ is the sum of description complexity of $\mathcal{F}$, defined the same way as for Hyper, and the error $E(I, A_\mathcal{F} \circ B_\mathcal{F})$. Every $\langle C, D\rangle$ in $\mathcal{F}$ is computed by first computing its core and then extending the core to $\langle C, D\rangle$. A core is, in fact, a rectangle contained in I and is computed by adding columns from a sorted list (sorting by several criteria is proposed). Extension to $\langle C, D\rangle$ is performed by adding further columns and rows to a core until such addition does not help in minimizing the cost. The authors also propose simple randomization to overcome the drawback of selecting the columns from a fixed, sorted list.

Synthetic data We use the datasets described in Table 1. Every Set Xi consists of 1000 matrices I of size 1000×500, obtained as Boolean products of matrices A and B that are generated randomly with prescribed densities dens A and dens B and the corresponding dimensions. The column dens I contains the average density of I.

Real data. We used the datasets Mushroom [1], DBLP,[1] Paleo,[2] Chess [1], DNA [23] Tic-tac-toe [1], Americas-small, Apj, and Firewall 1 [8], see Table 2, most of which are well known and used in the literature on BMF.

Note at this point that the datasets Americas-small, Apj, and Firewall 1 plus six additional ones used in [8] are all real-world data from the role minimization area (see the text below Observation 2). The authors in [8] proposed an exponential time complexity algorithm for computing the Boolean rank and obtained with it exact ranks of all of their nine datasets. We observed that the feasibility of computing exact ranks is due to very high redundancy of these datasets, which removal

[1] http://www.informatik.uni-trier.de/~ley/db/.
[2] NOW public release 030717, available from http://www.helsinki.fi/science/now/.

Table 2
Real data.

Dataset	Size	$\|\|I\|\|$	$\|\|\mathcal{E}(I)\|\|$	$\|\|\mathcal{E}(I)\|\|/\|\|I\|\|$
Mushroom	8124×119	186852	82965	0.444
DBLP	19×6980	40637	1601	0.039
Paleo	501×139	3537	1906	0.539
Chess	3196×76	118252	71296	0.603
DNA	4590×392	26527	1685	0.064
Tic-tac-toe	958×29	9580	9580	1.000
Americas-small	3477×1587	105205	51258	0.487
Apj	2044×1164	6841	2981	0.436
Firewall 1	365×709	31951	2787	0.087

makes it easy to compute the rank. This is also congruent with the author's claim that for seven of the nine datasets the problem was solved solely by the first, reduction step of their algorithm. When testing, we were able to compute the ranks reported by the authors but had to terminate the computations of all the other above datasets except DBLP after a month of computation. The algorithm nevertheless provides us with the information about ranks of the feasible datasets: rank(DBLP) = 19, rank(Americas-small) = 178, rank(Apj) = 453, rank(Firewall 1) = 64, which we utilize below. [8] also proposed a heuristic factorization algorithm which makes use of rows and columns as factors. We do not include it in our comparison since it yields a flat covering function (see below). Note also that resorting to rows in a similar manner is a part of Hyper.

Reduction in number of entries with 1*: I vs. $\mathcal{E}(I)$* In view of the properties of essential parts and the strategy of GreEss, it is clearly significant to observe the ratio $||\mathcal{E}(I)||/||I||$ of the number of entries containing 1 in $\mathcal{E}(I)$ to the corresponding number for I. These characteristics are provided in Table 1 and Table 2. As one can see, the reduction in the number of 1s in the essential parts is significant.

5.2. Performance of algorithms

Arguably, the most important aspect in evaluating the performance of BMF algorithms is the quality of decompositions delivered by the algorithms. The existing literature provides numerous evidence demonstrating that the factors in Boolean data are meaningful and interesting, hence we focus on comparison in terms of quantitative criteria described below. In addition to the quality of decompositions, another issue is the time complexity.

Time complexity Time complexity is not as critical a constraint as the quality of delivered decompositions, though clearly, time complexity should not be prohibitive. An upper bound of time complexity of GreEss is derived above; for information regarding Tiling, Asso, Hyper, GreConD, and PaNDa we refer the above-mentioned papers. In what follows, we provide a comparison of running times of the six algorithms. We implemented all algorithms in MATLAB with critical parts written C and compiled to binary MEX files. We employed about the same level of optimization to make the time demands of the algorithms comparable. Tiling, GreConD, and GreEss run without parameters to be set. For the other algorithms, we followed the recommendations by the authors. We, however, experimented with setting the parameters and chose them individually, with the best performance for every given dataset. Note that the dependence on parameters may be considered both advantage (adaptivity) and disadvantage (burden on user's side). In particular, Asso requires us to set τ, and (one of) w^+ and w^- (see above). In most cases, the best choice was $0.8 \leq \tau < 1$ and $w^+, w^- \in \{1, 2, 3\}$. For Hyper, we set the support parameter $\alpha \geq 0.3$ and used closed α-frequent itemsets (see above). For PaNDa, we used attribute sorting by frequency and the randomization described in [16]. These settings are used in the evaluation below.

The overall fastest algorithm is GreConD. This algorithm does not perform any data preprocessing and utilizes a very fast heuristic for computing the factors. Second to GreConD is GreEss which was about 2× slower. Third to GreConD is Asso which was about 3–4× slower than GreConD. Fourth and fifth in terms of time demand are Hyper and PaNDa which are about 5× slower than GreConD. However, the time consumed by Hyper depends on the size of the set $\mathcal{C}_\alpha$ of frequent itemsets, and hence depends on α (see above). As is well-known, the number of frequent as well as closed frequent itemsets may be exponential in the number of items. As a result, the worst case time complexity of Hyper is exponential in the number of attributes, as mentioned above. Tiling is the slowest of all the compared algorithms. On average, it was about 400× slower than GreConD. This is because in selecting each tile, Tiling browses the set of all maximal tiles which is usually very large and may be exponential in terms of the minimum of the number of objects and attributes. Note also that according to [4] and our experience, GreConD implemented in C factorizes Mushroom dataset in the order of seconds on an ordinary PC.

Quality of decompositions Recall that GreEss is designed for the AFP problem. However, we take into account both views reflecting the goals of AFP and DBP (Section 2.1) and require that a good factorization algorithm computes a decomposition (or approximate decomposition) of the input matrix I using a reasonably small number of factors in such a way that the first

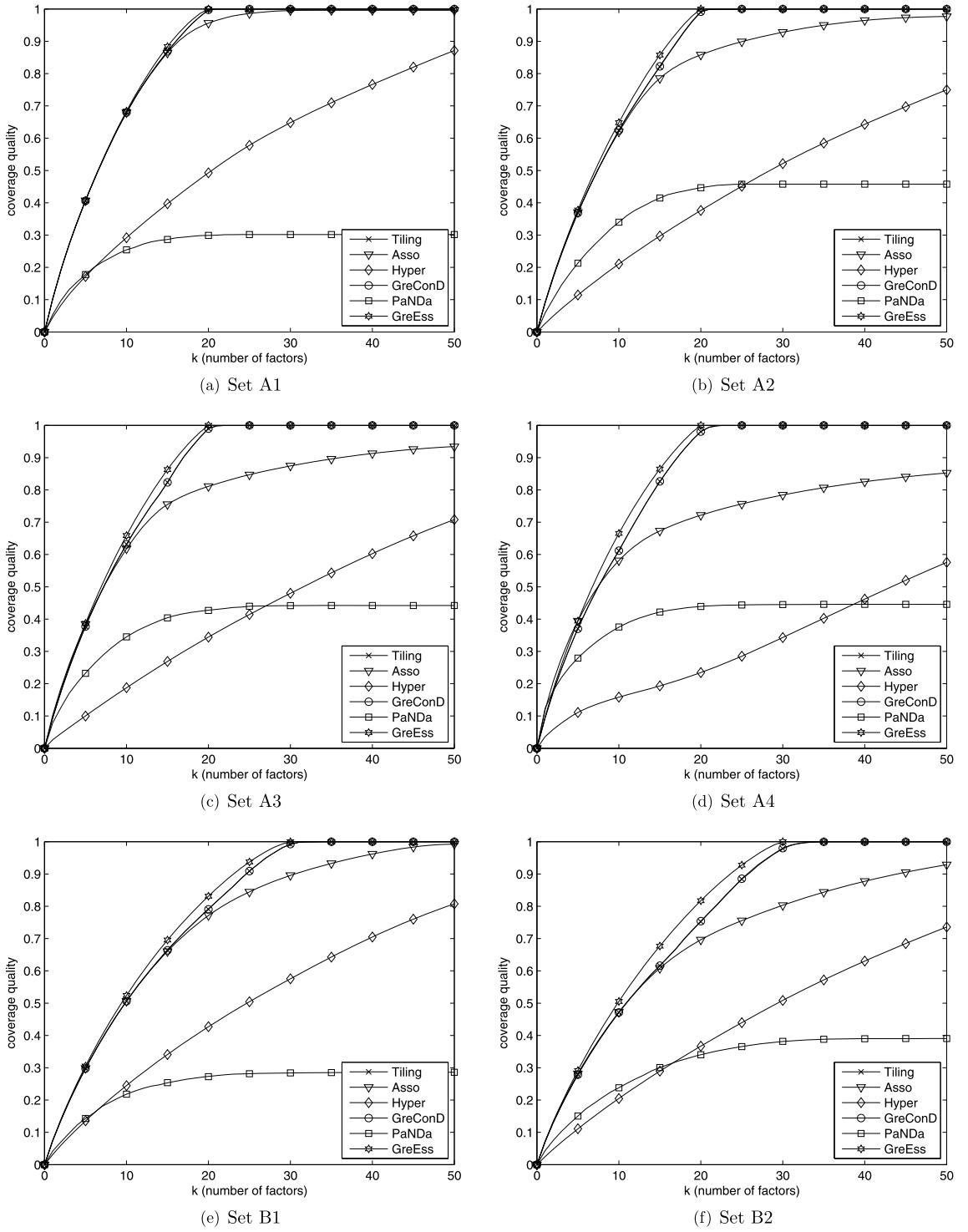


Fig. 1. Coverage quality of the first k factors (synthetic data).

factors have a reasonably good coverage, i.e. explain a large portion of data. For this purpose we compare the factorization algorithms as follows. We employ the following function of $A \in \{0,1\}^{n\times l}$ and $B \in \{0,1\}^{l\times m}$ representing the *coverage quality* of the first l factors delivered by the particular algorithm:

$$c = 1 - E(I, A \circ B)/||I||. \tag{10}$$

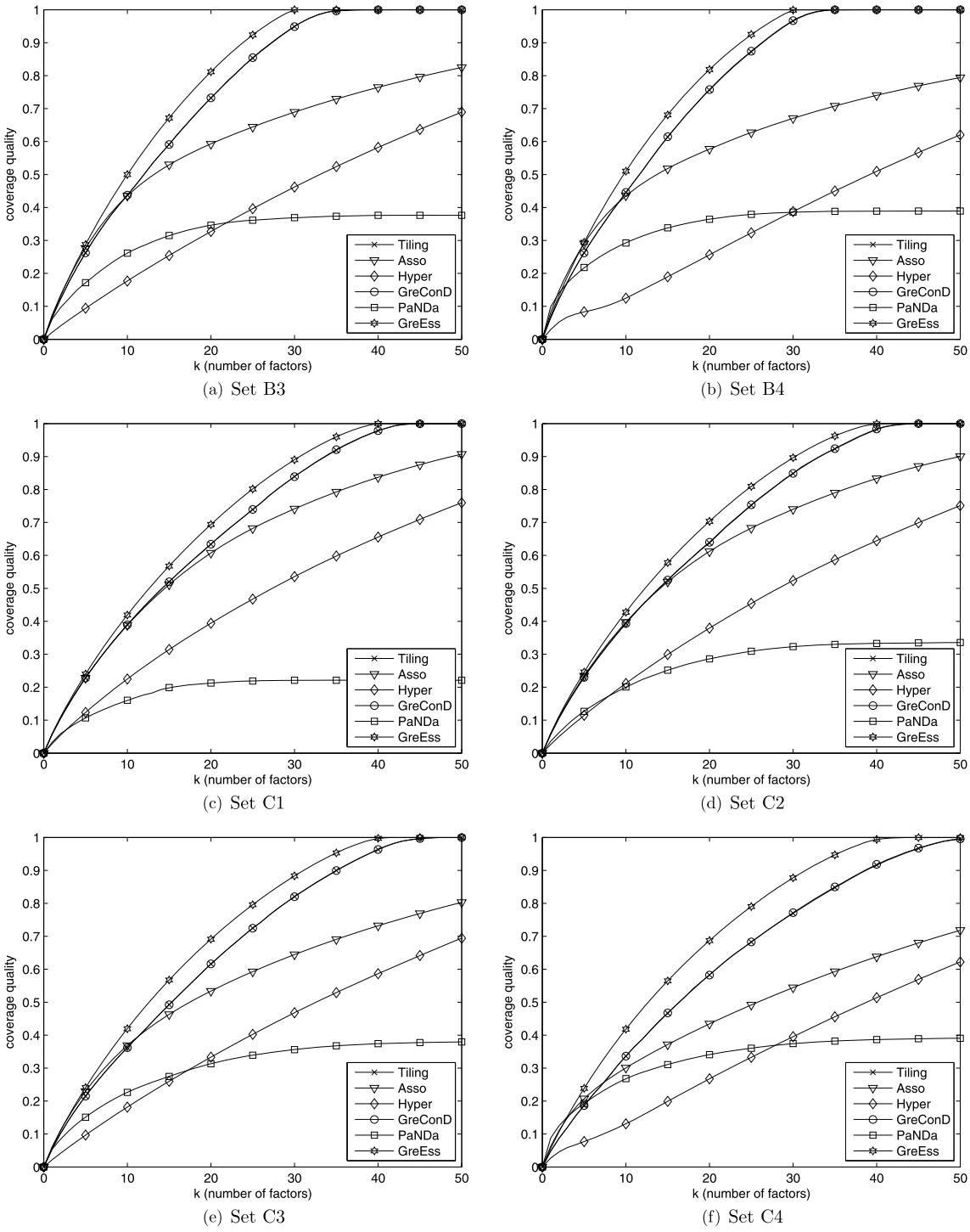


Fig. 2. Coverage quality of the first k factors (synthetic data) continued.

Similar functions are used in [4,11]. We observe the values of c for $l = 0, \ldots, k$, where k is the number of factors delivered by a particular algorithm. Clearly, for $l = 0$ (no factors added, A and B are "empty") we have $c = 0$. In accordance with the above requirements, for a good factorization algorithm c should be increasing in l, should have relatively large values even for small l, and it is desirable that for $l = k$ we have $I = A \circ B$, i.e. the data is fully explained by all the k factors computed, in which case $c = 1$.

The results for synthetic and real data are shown in Figs. 1, 2 and Tables 3, 4. The results for synthetic data are obtained as averages over the 1000 datasets comprised by each Set Xi. In Fig. 1,2, we display the curves of the coverage quality as a function of k. In Table 3, we display coverage quality and its standard deviation of the sets of the first k factors computed by the algorithms for selected k. In Table 4, we display the number of factors needed to cover 25%, 50%, 75%, 95%, and 100% of the data for the six real datasets.

All the algorithms compute the factors for a given I one after another. In case of Tiling, Hyper, GreConD and GreEss, this process is guaranteed to stop when an exact decomposition $I = A \circ B$ is found. With Asso and PaNDa, it mostly happens that an exact decomposition is not found, which is seen in the tables and graphs and is indicated by NA in Table 4. This feature is a consequence of committing the error E_o non-repairable by further factors (cf. Observation 2 and the discussion below) and is a clear disadvantage of Asso and PaNDa when a large coverage is required. This is in particular true of PaNDa which, even though having a good coverage by the first few factors particularly on dense datasets, is often not capable of reaching coverage higher than 0.4. Asso tends to perform better than PaNDa, has a good coverage by the first couple of factors and is able to achieve significantly higher coverage than PaNDa. Asso tends to perform slightly better on datasets which are sparse. The experiments show that both Asso and PaNDa are suitable for DBP with a small prescribed number k of factors but are not suitable for AFP with a small prescribed error ε (i.e. high coverage), the latter being particularly true of PaNDa.

On the other hand, Tiling, Hyper, GreConD and GreEss are all primarily designed for AFP. Let us first observe that except for Hyper, they nevertheless have a good performance in obtaining a good coverage by the first few factors, i.e. from the DBP view point, as well. In fact, GreEss slightly outperforms Asso on synthetic datasets and is slightly outperformed by it on real datasets—the two algorithms appear to comparable in this respect. When large coverage, and hence AFP with small ε is concerned, GreEss has clearly the best performance in comparison with Tiling and GreConD on the synthetic datasets and the best performance on most of the real datasets. In fact, it is only outperformed by Tiling and GreConD on DNA for coverages 25–75% (but significantly wins on when close to 100%).

For Hyper, we observe two of its features which we discuss below. First, a generally rather slow increase of coverage with a growing number of factors on all the synthetic data as well as on Mushroom, Chess, DNA and Apj. Second, with exception of Mushroom and Chess, the number of factors needed for exact decomposition of I does not exceed $\min(m, n)$—the smaller of the number of rows and columns. The reason for this behavior is the following. Hyper (see its description above) includes in the set $\mathcal{C}_\alpha$ not only the rectangles corresponding to α-frequent itemsets but also those corresponding to all the singleton itemsets, i.e. the rectangles corresponding to the columns of the input matrix. Including the singleton itemsets guarantees that an exact decomposition of the input matrix is found. It turns out from a closer examination of the results, however, that the factors corresponding to the singleton items are used very often. This causes a very low coverage by the sets of the first k factors of Hyper compared to the other algorithms, because the factors corresponding to the singleton items are very thin. Note in this connection that a trivial factorization algorithm that outputs for an input matrix $I \in \{0, 1\}^{n \times m}$ the set $\mathcal{F}$ containing the m rectangles corresponding to the columns of I, resulting in a trivial factorization of I into the product of I and the $m \times m$ identity matrix, will have a similar behavior as Hyper, namely a slowly-growing curve of coverage quality which, nevertheless, reaches full coverage (exact decomposition) with $k \leq m$. We consider the poor coverage by the first k factors even for a relatively large k a significant drawback of Hyper as well as the trivial factorization algorithm. Note that due to its logic described above, the heuristic algorithm of [8] also shares this drawback.

From the data analysis viewpoint, computing a factorization with high accuracy, such as 95% coverage, is sufficient and GreEss typically achieves it with much less than $\min(m, n)$ factors (Table 4). We may also observe on the synthetic datasets that with growing density, the difference in performance of GreEss compared to the other algorithms tends to increase. When exact factorization is desirable and the rank of I is significantly smaller than $\min(m, n)$, GreEss clearly outperforms Hyper (and the trivial factorization algorithm). This is the case of all the synthetic data, the Americas-small, Apj, Firewall 1 (and in fact all the other datasets from [8]), and possibly also Mushroom for which we nevertheless do not know the rank. For instance, ranks of Americas-small, Apj, Firewall 1 are known (see above) an equal to 178, 453 and 64. While Hyper provides the poor approximations 1571, 1163, and 105, GreEss delivers rather tight ones, namely 190, 454, and 64, respectively. On the other hand, if the rank is $\min(m, n)$ or slightly smaller, GreEss, as well as Tiling and GreConD, often end up with more than $\min(m, n)$, are trivially outperformed by the trivial factorization and typically also by Hyper. This is the case of Paleo, Chess, DNA, and Tic-tac-toe. For DBLP, for which the rank is known and equal to $\min(m, n) = 19$, GreEss and Hyper both compute the rank. From this perspective, an interesting problem is to develop an algorithm which computes factors one-by-one, shares the above mentioned good features of GreEss plus the property of never exceeding $\min(m, n)$ factors for exact decomposition.

To sum up, GreEss appears the best algorithm overall. For all datasets, it has the highest coverage by the first few factors (along with Asso), provides the best almost exact factorizations, and for datasets with rank considerably smaller than $\min(m, n)$ it delivers the best rank approximations which tend to be tight. In this respect, Tiling and GreConD have also a good performance. Both perform very similarly, confirming the evaluation results of Algorithm 1 and GreConD in [4] (cf. the description of GreConD above), but are outperformed by GreEss. For datasets with rank close to $\min(m, n)$, Hyper delivers better exact decompositions than GreEss but a clear drawback of Hyper is its very slowly increasing coverage. Asso, on the other hand, has a rapidly increasing coverage in the beginning, comparable to that of GreEss. On the other hand,

Table 3
Quality of decompositions (synthetic data).

Dataset	k	Coverage c of the first k factors					
		Tiling	Asso	Hyper	GreConD	PaNDa	GreEss
Set A1	5	0.406±0.067	0.407±0.067	0.172±0.091	0.406±0.067	0.178±0.101	0.407±0.067
	10	0.680±0.043	0.679±0.044	0.292±0.085	0.680±0.043	0.254±0.128	0.684±0.041
	15	0.869±0.029	0.864±0.031	0.397±0.081	0.868±0.028	0.286±0.155	0.884±0.021
	20	0.997±0.007	0.956±0.031	0.493±0.077	0.997±0.007	0.299±0.167	1.000±0.000
Set A2	5	0.368±0.045	0.370±0.045	0.114±0.049	0.368±0.045	0.213±0.063	0.377±0.040
	10	0.625±0.041	0.619±0.041	0.211±0.047	0.625±0.041	0.340±0.084	0.648±0.031
	15	0.822±0.031	0.785±0.048	0.297±0.045	0.821±0.030	0.415±0.115	0.858±0.018
	20	0.992±0.016	0.858±0.053	0.376±0.044	0.990±0.015	0.446±0.128	1.000±0.000
Set A3	5	0.377±0.045	0.384±0.041	0.100±0.052	0.377±0.045	0.232±0.052	0.387±0.037
	10	0.632±0.040	0.618±0.041	0.188±0.050	0.632±0.040	0.345±0.058	0.659±0.024
	15	0.823±0.028	0.755±0.054	0.269±0.048	0.823±0.028	0.404±0.076	0.863±0.015
	20	0.990±0.017	0.811±0.052	0.344±0.045	0.990±0.017	0.427±0.091	1.000±0.000
Set A4	5	0.370±0.039	0.394±0.035	0.111±0.041	0.370±0.039	0.279±0.036	0.394±0.030
	10	0.612±0.037	0.581±0.044	0.159±0.039	0.612±0.037	0.376±0.049	0.665±0.021
	15	0.826±0.025	0.673±0.051	0.193±0.047	0.826±0.025	0.422±0.065	0.864±0.012
	20	0.981±0.020	0.722±0.047	0.234±0.063	0.981±0.020	0.439±0.076	1.000±0.000
Set B1	5	0.298±0.070	0.298±0.070	0.136±0.080	0.298±0.070	0.143±0.093	0.305±0.069
	10	0.506±0.061	0.506±0.061	0.245±0.076	0.506±0.061	0.218±0.104	0.524±0.055
	15	0.664±0.052	0.660±0.052	0.341±0.072	0.664±0.052	0.254±0.125	0.695±0.041
	20	0.791±0.044	0.772±0.046	0.427±0.068	0.791±0.044	0.272±0.139	0.831±0.027
	25	0.909±0.028	0.844±0.040	0.504±0.065	0.909±0.028	0.281±0.149	0.937±0.014
	30	0.993±0.012	0.896±0.032	0.576±0.062	0.992±0.012	0.284±0.151	1.000±0.001
Set B2	5	0.279±0.046	0.282±0.045	0.111±0.059	0.279±0.046	0.151±0.059	0.291±0.047
	10	0.471±0.038	0.472±0.037	0.205±0.056	0.471±0.038	0.238±0.064	0.505±0.036
	15	0.616±0.036	0.608±0.040	0.290±0.053	0.616±0.036	0.301±0.061	0.677±0.026
	20	0.754±0.032	0.696±0.046	0.367±0.049	0.752±0.033	0.340±0.070	0.817±0.019
	25	0.885±0.027	0.756±0.047	0.439±0.047	0.884±0.027	0.366±0.080	0.927±0.012
	30	0.980±0.021	0.803±0.044	0.508±0.044	0.978±0.021	0.382±0.088	1.000±0.001
Set B3	5	0.262±0.044	0.276±0.041	0.094±0.044	0.262±0.044	0.172±0.034	0.287±0.037
	10	0.438±0.040	0.435±0.047	0.177±0.041	0.438±0.040	0.261±0.036	0.500±0.030
	15	0.591±0.034	0.530±0.053	0.254±0.039	0.591±0.034	0.315±0.046	0.672±0.024
	20	0.732±0.034	0.592±0.052	0.327±0.037	0.732±0.032	0.346±0.061	0.812±0.018
	25	0.855±0.032	0.644±0.049	0.396±0.035	0.853±0.031	0.361±0.070	0.923±0.010
	30	0.949±0.029	0.689±0.045	0.462±0.033	0.949±0.029	0.369±0.077	1.000±0.000
Set B4	5	0.262±0.044	0.290±0.042	0.083±0.042	0.262±0.044	0.218±0.039	0.295±0.038
	10	0.446±0.042	0.436±0.055	0.125±0.055	0.446±0.042	0.292±0.041	0.510±0.029
	15	0.615±0.036	0.518±0.059	0.190±0.062	0.614±0.034	0.338±0.048	0.681±0.022
	20	0.758±0.032	0.577±0.056	0.256±0.066	0.756±0.030	0.364±0.056	0.818±0.015
	25	0.874±0.030	0.627±0.051	0.323±0.067	0.872±0.027	0.379±0.064	0.925±0.010
	30	0.967±0.025	0.670±0.046	0.388±0.065	0.966±0.023	0.386±0.069	0.999±0.004
Set C1	5	0.228±0.056	0.228±0.057	0.124±0.061	0.228±0.056	0.107±0.068	0.240±0.052
	10	0.390±0.055	0.388±0.057	0.225±0.056	0.389±0.055	0.161±0.076	0.419±0.046
	15	0.520±0.051	0.511±0.057	0.314±0.052	0.520±0.051	0.199±0.099	0.568±0.041
	20	0.634±0.049	0.607±0.058	0.394±0.049	0.634±0.049	0.213±0.110	0.694±0.033
	25	0.740±0.041	0.682±0.056	0.467±0.045	0.739±0.041	0.219±0.116	0.802±0.025
	30	0.839±0.033	0.741±0.053	0.535±0.041	0.840±0.033	0.221±0.119	0.890±0.017
	35	0.921±0.024	0.793±0.049	0.598±0.037	0.919±0.023	0.221±0.119	0.960±0.009
	40	0.979±0.016	0.837±0.044	0.656±0.033	0.978±0.015	0.221±0.119	1.000±0.001
Set C2	5	0.230±0.052	0.234±0.052	0.115±0.060	0.230±0.052	0.127±0.062	0.246±0.048
	10	0.394±0.047	0.397±0.048	0.212±0.057	0.394±0.047	0.201±0.059	0.428±0.039
	15	0.525±0.044	0.520±0.048	0.299±0.053	0.524±0.043	0.252±0.060	0.578±0.032
	20	0.640±0.039	0.612±0.048	0.379±0.050	0.638±0.037	0.286±0.064	0.703±0.027
	25	0.754±0.035	0.683±0.047	0.454±0.047	0.752±0.032	0.309±0.075	0.809±0.022
	30	0.849±0.032	0.740±0.045	0.524±0.044	0.848±0.030	0.323±0.084	0.896±0.015
	35	0.924±0.024	0.790±0.042	0.587±0.039	0.923±0.023	0.330±0.089	0.962±0.010
	40	0.983±0.018	0.833±0.037	0.644±0.037	0.982±0.016	0.333±0.091	0.999±0.002
Set C3	5	0.215±0.046	0.229±0.044	0.097±0.044	0.215±0.046	0.151±0.039	0.241±0.040
	10	0.362±0.045	0.369±0.049	0.182±0.041	0.362±0.045	0.226±0.043	0.420±0.035
	15	0.492±0.038	0.463±0.054	0.260±0.039	0.493±0.039	0.275±0.049	0.567±0.029
	20	0.616±0.035	0.534±0.053	0.333±0.037	0.616±0.035	0.313±0.059	0.691±0.024
	25	0.725±0.032	0.593±0.050	0.402±0.036	0.725±0.033	0.339±0.065	0.796±0.019

Table 3 (*continued*)

Dataset	k	Coverage c of the first k factors					
		Tiling	Asso	Hyper	GreConD	PaNDa	GreEss
	30	0.820±0.031	0.645±0.046	0.468±0.034	0.820±0.030	0.356±0.070	0.883±0.013
	35	0.899±0.024	0.691±0.041	0.529±0.032	0.900±0.024	0.368±0.075	0.953±0.009
	40	0.964±0.022	0.732±0.037	0.586±0.031	0.963±0.021	0.374±0.079	0.997±0.004
Set C4	5	0.186±0.034	0.207±0.041	0.077±0.035	0.186±0.034	0.193±0.029	0.238±0.028
	10	0.337±0.031	0.301±0.048	0.131±0.046	0.336±0.031	0.268±0.033	0.418±0.025
	15	0.468±0.030	0.371±0.047	0.199±0.051	0.467±0.030	0.311±0.036	0.565±0.021
	20	0.582±0.031	0.434±0.045	0.267±0.049	0.583±0.032	0.341±0.039	0.687±0.018
	25	0.683±0.032	0.492±0.042	0.333±0.047	0.683±0.032	0.360±0.045	0.790±0.015
	30	0.772±0.033	0.544±0.038	0.395±0.046	0.770±0.031	0.375±0.051	0.878±0.012
	35	0.850±0.030	0.593±0.034	0.456±0.044	0.848±0.029	0.382±0.055	0.947±0.009
	40	0.918±0.028	0.638±0.030	0.514±0.041	0.916±0.027	0.387±0.059	0.994±0.007

Asso is typically neither able of computing exact not highly accurate decompositions. The same is true of PaNDa which is nevertheless outperformed by Asso.

From the point of view of the new strategy of GreEss, which is based on the results regarding essential part of I, the following conclusions may be drawn. Contrary to Tiling, Asso, Hyper, GreConD and PaNDa, which all use different strategies of greedy coverage, but all aim at covering the most of the uncovered 1s in I, GreEss proceeds differently. In its greedy coverage, GreEss focuses on the essential 1s in I and considers them as "seeds" of good factors. Such strategy is theoretically well justified, is fast, and leads to improvement in quality of Boolean matrix factorization.

6. Conclusions

We presented new results on BMF that are based on examining the closure and order-theoretic structures related to Boolean data. The results let us differentiate the role of entries of the input matrix and suggest where to focus in computing decompositions. We proposed a new BMF algorithm, GreEss, based on these results and provided its experimental evaluation. It turns out that the algorithm performs well both in terms of coverage of the input data by the first k factors (i.e. by a small number of the most important factors) and in terms of the number of factors needed for an almost exact decomposition of the input matrix (i.e. factors that almost fully explain the input data) and that GreEss outperforms the existing algorithms from this viewpoint. The presented results, both theoretical and experimental, emphasize the role of from-below factorization algorithms in BMF, of which GreEss is an example.

An important topic for future research is to utilize further the present results regarding essential parts of Boolean matrices and to further investigate the role of entries of Boolean matrices for BMF. In particular, it seems promising to explore the possibility to still reduce $\mathcal{E}(I)$ to $\mathcal{E}(\mathcal{E}(I))$, and in general, to $\mathcal{E}^p(I)$. Furthermore, the notion of ε-essential part shall be investigated for $\varepsilon > 0$. Another topic is to utilize as heuristics other information that may be obtained from the intervals $\mathcal{I}_{ij}$, in particular the number $|\mathcal{I}_{ij}|$ of concepts covering $\langle i, j\rangle$. This number is difficult to compute but our preliminary results indicate that it may be approximated quickly. Note that the case $|\mathcal{I}_{ij}| = 1$ corresponds to so-called mandatory factors considered in [4], i.e. factors that need to be present in every exact decomposition of I. The techniques employed in [8], in particular the interesting relationship to graph problems—a reduction and coloring, also need closer examination from the perspective of computing approximate decompositions. A particular problem suggested by the experiments is to develop an algorithm sharing the good features of GreEss that additionally never produces a number of factors larger than the smaller of the number of rows and columns for exact decompositions.

An important topic is to extend the theoretical framework to general factorizations involving rectangles containing possibly 0s, which are sometimes called fault-tolerant concepts or noisy tiles, see e.g. [5]. An interesting goal is to extend the present results beyond Boolean data, namely to ordinal and semiring-valued data, see e.g. [2] for general results on closure structures and decompositions of such data. Let us mention that three- and multi-way data received a considerable attention recently. [3,19] present approaches to factorization of three-way Boolean data. An extension of the present results to multi-way data seems another important research topic. Last but not least, the problem of noise in Boolean data, often discussed in the data mining literature, needs careful attention. This includes its proper conceptual treatment derived from real-world examples and the subsequent evaluation of the present factorization strategies.

Acknowledgments

We thank the reviewers for their suggestions, particularly for bringing [8] to our attention and suggesting improvements in the experimental part. The first results were obtained by support of grant No. P202/10/0262, the substantial revision then by grant No. GA15-17899S of the Czech Science Foundation. M. Trnecka also acknowledges partial support by the grant No. PrF 2014034, IGA 2014 of Palacký University.

Table 4
Quality of decompositions (real data).

Dataset	Coverage (100c%)	Number of factors needed for the prescribed coverage					
		Tiling	Asso	Hyper	GreConD	PaNDa	GreEss
Mushroom	25%	3	2	8	3	1	2
	50%	7	6	19	7	NA	8
	75%	24	36	37	24	NA	26
	95%	63	NA	70	63	NA	62
	100%	119	NA	122	120	NA	105
DBLP	25%	2	2	2	2	NA	2
	50%	5	5	5	5	NA	5
	75%	10	10	10	11	NA	10
	95%	19	17	17	19	NA	17
	100%	21	19	19	20	NA	19
Paleo	25%	16	16	14	16	NA	15
	50%	39	40	38	39	NA	38
	75%	75	76	73	76	NA	73
	95%	129	NA	121	127	NA	122
	100%	151	NA	139	152	NA	145
Chess	25%	2	1	9	1	1	1
	50%	5	2	26	4	NA	6
	75%	16	15	39	15	NA	17
	95%	47	NA	59	46	NA	46
	100%	124	NA	90	124	NA	113
DNA	25%	8	6	24	8	NA	13
	50%	32	27	67	33	NA	41
	75%	94	80	155	96	NA	105
	95%	157	NA	239	154	NA	156
	100%	489	NA	392	496	NA	408
Tic-tac-toe	25%	5	6	5	5	NA	5
	50%	12	12	11	12	NA	12
	75%	19	19	18	19	NA	19
	95%	28	27	26	28	NA	28
	100%	31	29	29	32	NA	32
Americas small	25%	1	1	1	1	1	1
	50%	1	1	1	1	1	1
	75%	4	3	92	4	NA	4
	95%	30	63	483	30	NA	33
	100%	197	NA	1571	197	NA	190
Apj	25%	1	1	8	1	1	1
	50%	34	36	66	34	NA	35
	75%	138	146	277	138	NA	136
	95%	332	371	822	332	NA	328
	100%	463	NA	1163	463	NA	454
Firewall 1	25%	1	1	1	1	1	1
	50%	1	1	1	1	1	1
	75%	2	2	2	2	2	2
	95%	6	NA	8	6	NA	6
	100%	66	NA	105	66	NA	64

References

[1] K. Bache, M. Lichman, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml, 2013, University of California, School of Information and Computer Science, Irvine, CA.
[2] R. Belohlavek, Optimal decompositions of matrices with entries from residuated lattices, J. Log. Comput. 22 (6) (2012) 1405–1425.
[3] R. Belohlavek, C. Glodeanu, V. Vychodil, Optimal factorization of three-way binary data using triadic concepts, Order 30 (2) (2013) 437–454 (preliminary version in Proc. GrC 2010).
[4] 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 (preliminary version in Proc. SCIS & ISCIS 2006).
[5] J. Besson, R.G. Pensa, C. Robardet, J.F. Boulicaut, Constraint-based mining of fault-tolerant patterns from Boolean data, in: Proc. KDID, 2006, pp. 55–71.
[6] R.A. Brualdi, H.J. Ryser, Combinatorial Matrix Theory, Cambridge University Press, 1991.
[7] B.A. Davey, H.A. Priestley, Introduction to Lattices and Order, 2nd ed., Cambridge University Press, 2002.
[8] A. Ene, et al., Fast exact and heuristic methods for role minimization problems, in: Proc. SACMAT, 2008, pp. 1–10.
[9] B. Ganter, C.V. Glodeanu, Ordinal Factor Analysis, Lecture Notes in Computer Science, vol. 7278, 2012, pp. 128–139.
[10] B. Ganter, R. Wille, Formal Concept Analysis: Mathematical Foundations, Springer, Berlin, 1999.
[11] F. Geerts, B. Goethals, T. Mielikäinen, Tiling databases, in: Proc. Discovery Science, 2004, pp. 278–289.

[12] K.H. Kim, Boolean Matrix Theory and Applications, M. Dekker, NY, 1982.
[13] S.O. Kuznetsov, S. Obiedkov, Comparing performance of algorithms for generating concept lattices, J. Exp. Theor. Artif. Intell. 14 (2002) 189–216.
[14] H. Lu, J. Vaidya, V. Atluri, Optimal Boolean matrix decomposition: application to role engineering, in: Proc. IEEE ICDE, 2008, pp. 297–306.
[15] H. Lu, J. Vaidya, V. Atluri, Y. Hong, Constraint-aware role mining via extended Boolean matrix decomposition, IEEE Trans. Dependable Secure Comput. 9 (5) (2012) 655–669.
[16] C. Lucchese, S. Orlando, R. Perego, Mining top-K patterns from binary datasets in presence of noise, in: SIAM DM, 2010, pp. 165–176.
[17] P. Miettinen, The Boolean column and column–row matrix decompositions, Data Min. Knowl. Discov. 17 (2008) 39–56.
[18] P. Miettinen, Sparse Boolean matrix factorizations, in: Proc. IEEE ICDM, 2010, pp. 935–940.
[19] P. Miettinen, Boolean tensor factorizations, in: Proc. IEEE ICDM, 2011, pp. 447–456.
[20] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, H. Mannila, The discrete basis problem, IEEE Trans. Knowl. Data Eng. 20 (10) (2008) 1348–1362, (preliminary version in Proc. PKDD 2006).
[21] P. Miettinen, J. Vreeken, Model order selection for Boolean matrix factorization, in: Proc. ACM SIGKDD, 2011, pp. 51–59.
[22] S.D. Monson, S. Pullman, R. Rees, A survey of clique and biclique coverings and factorizations of (0, 1)-matrices, Bull. Inst. Comb. Appl. 14 (1995) 17–86.
[23] S. Myllykangas, et al., DNA copy number amplification profiling of human neoplasms, Oncogene 25 (55) (2006) 7324–7332.
[24] D.S. Nau, Specificity covering, Tech. Rep. CS-1976-7, Duke University, 1976.
[25] D.S. Nau, G. Markowsky, M.A. Woodbury, D.B. Amos, A mathematical analysis of human leukocyte antigen serology, Math. Biosci. 40 (1978) 243–270.
[26] J. Outrata, Boolean factor analysis for data preprocessing in machine learning, in: Proc. ICMLA, 2010, pp. 899–902.
[27] G. Schmidt, Relational Mathematics, Cambridge University Press, 2011.
[28] L. Stockmeyer, The set basis problem is NP-complete, Tech. Rep. RC5431, IBM, Yorktown Heights, NY, USA, 1975.
[29] N. Tatti, T. Mielikäinen, A. Gionis, H. Mannila, What is the dimension of your binary data?, in: Proc. IEEE ICDM, 2006, pp. 603–612.
[30] J. Vaidya, V. Atluri, Q. Guo, The role mining problem: finding a minimal descriptive set of roles, in: Proc. SACMAT, 2007, pp. 175–184.
[31] Y. Xiang, R. Jin, D. Fuhry, F.F. Dragan, Summarizing transactional databases with overlapped hyperrectangles, Data Min. Knowl. Discov. 23 (2011) 215–251 (preliminary version in Proc. ACM KDD 2008).