Pattern Structures for Identifying Biclusters with Coherent Sign Changes

Nyoman Juniarta, Victor Codocedo, Miguel Couceiro, Mehdi Kaytoue, Amedeo Napoli

Pattern Structures for Identifying Biclusters with Coherent Sign Changes

Nyoman Juniarta[1], Victor Codocedo[2], Miguel Couceiro[1], Mehdi Kaytoue[3], and Amedeo Napoli[1]

[1] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
{nyoman.juniarta, miguel.couceiro, amedeo.napoli}@loria.fr
[2] Universidad Técnica Federico Santa María, Chile
victor.codocedo@inf.utfsm.cl
[3] Univ Lyon, INSA Lyon, CNRS, LIRIS UMR 5205, F-69621 Lyon, France
mehdi.kaytoue@insa-lyon.fr

Abstract. In this paper we are studying the task of finding coherent-sign-changes biclusters in a binary matrix. This task can be applied to the interpretation of gene expression data, where such a bicluster represents a set of experiments that affect a set of genes in a consistent way. We start with a binary table and study biclustering methods based on FCA and partition pattern structures. Pattern concepts provide biclusters and their hierarchical relation, which can be used to analyze the profile of genes in the given expression data. Our approach is purely symbolic, so we can detect larger biclusters and work with rather complex data.

Keywords: biclustering, FCA, gene expression, pattern structures

1 Introduction

Gene expression data can be represented as a matrix, where rows and columns represent genes and experiments respectively. Each cell contains the numeric expression level of a given gene under a given experiment. In such data, we can say that an experiment affect a gene by either lowering or raising its expression, according to the gene's normal level. One may be interested in finding a subset of genes and a subset of experiments, such that the experiments affect the genes in a consistent way. In other words, any two experiments in the subset have always either the same effect or the opposite effect on every gene in the subset. This task corresponds to the mining of coherent-sign-changes (CSC) biclusters.

Biclustering is an important technique aimed at discovering patterns in a matrix representing a dataset. It is related to standard clustering whose main objective is to group the rows based on their similarity. On the other hand, biclustering refers to the problem of discovering submatrices whose cells exhibit similar behavior. This problem is also called co-clustering [9], where rows and columns are clustered simultaneously.

In this paper, we present a method based on FCA and pattern structures for discovering a specific type of bicluster: coherent-sign-changes bicluster. An

existing approach in [20] can mine this bicluster type, but it is statistical, since its discovery of CSC biclusters is based on the magnitude of the expression changes. Our approach is more symbolic, by taking into account only the direction of the changes, with expectation of detecting larger biclusters. Our FCA-based method also gives us the hierarchical structure of all biclusters, allowing an easier interpretation of the results by experts. Furthermore, pattern structures and AddIntent algorithm allows us to define a threshold of bicluster size, so that we can limit the amount of retrieved biclusters.

This paper is organized as follows. First, we discuss some related work about the discovery of biclusters with coherent sign changes in Section 2. Then, some basic definitions of biclustering are presented in Section 3. We explain our approach in Section 4, and discuss the experiments in Section 5. Finally, we concludes our article and give some research perspectives in Section 6.

2 Related Work

The row–column clustering was introduced in [10], and Cheng and Church [3] were the firsts to used the term biclustering while working on gene expression data. A bicluster in [3] is a subset of genes and a subset of conditions with a high similarity score, statistically measured by calculating variances over all values in the submatrix.

Still in the domain of gene expression data, the algorithm called SAMBA was proposed in [20] to discover a submatrix where the expressions of a subset of genes significantly changes across a subset of conditions. The first model of SAMBA searches a submatrix where there is a *joint* change across all genes, without looking whether it is an increase or a decrease. The second model takes into account the direction of the change, such that any two conditions in the submatrix have either always the same effect or always the opposite effect. We call this type of submatrix a coherent-sign-changes bicluster, as denoted in [18].

Regarding bicluster discovery based on FCA, several methods were proposed. In a binary matrix, dense approximate bicluster discovery was studied in [8, 11] based on standard FCA. This is similar to mining formal concept, but instead of "exact" concepts, the authors relax the problem such that the "approximate" concepts (having a certain amount of empty cells) can also be detected. For biclustering with similar values in a numerical matrix, Kaytoue et al. in [17] proposed standard FCA with scaling and interval pattern structures. Triadic Concept Analysis was also studied in [15] to extract this bicluster type. Furthermore, a partition pattern structure was presented in [5, 13] for mining bicluster with constant columns.

3 Biclustering

In this section, we recall the basic background and discuss illustrative examples of the different types of biclusters as described in [18]. We consider that a dataset is composed of a set of objects G, each of which has values over a set

of attributes M. This dataset can be represented as a numerical matrix where each cell indicates the value of an object w.r.t. an attribute, and can be formally written in Def. 1.

Definition 1 (Dataset)**.** A dataset is a numerical context (G, M, I) where G is a set of objects, M is a set of attributes, and I corresponds to $m(g)$, which is the value of $m \in M$ for object $g \in G$.

One may be interested in finding which subset of objects possesses the same values w.r.t. a subset of attributes. Regarding the matrix representation, this is equivalent to the problem of finding a submatrix that has a constant value over all of its elements (example in Table 1a). This task is called biclustering with constant values, which is a simultaneous clustering of the rows and columns of a matrix.

2	2	2	2	+	+	−
2	2	2	2	+	+	−
2	2	2	2	−	−	+
2	2	2	2	−	−	+
(a)				(b)		

Table 1. Examples of two bicluster types: (a) constant-values and (b) coherent-sign-changes (`CSC`).

In coherent-sign-changes (`CSC`) bicluster, the matrix is binary. In this bicluster, each row is correlated (either entirely identical or entirely opposite) to all other rows. In the example in Table 1c, the first row is identical to the second and opposite to the third and fourth. We can also see this bicluster by comparing the columns. In the example, the first column is identical to the second and opposite to the third. Before formalizing the definition of `CSC` bicluster, first we introduce the notation of column submatrix and its similarity.

Definition 2 (Column submatrix)**.** In a binary dataset (G, M, I), given a set of objects $A \subseteq G$ and an attribute $m \in M$, $m(A)$ is the column submatrix formed by the attribute m over A.

The submatrix $m_j(A)$ is equal to $m_k(A)$, denoted as $m_j(A) \simeq m_k(A)$, if all rows in $m_j(A)$ are either entirely identical or entirely opposite to the corresponding rows in $m_k(A)$. This can be formally written as:

$$m_j(A) \simeq m_k(A) \Longleftrightarrow \forall g \in A : m_j(g) = m_k(g) \text{ or } \forall g \in A : m_j(g) = \neg m_k(g).$$

With the previous notation, the definition of a `CSC` bicluster is given as follows.

Definition 3 (Coherent-sign-changes bicluster)**.** Given a binary dataset (G, M, I), a pair (A, B) (where $A \subseteq G$, $B \subseteq M$) is a coherent-sign-changes bicluster if $\forall m_j, m_k \in B : m_j(A) \simeq m_k(A)$.

In the bicluster discovery, a bicluster can be found entirely within another larger bicluster. We then say that this small bicluster is not maximal. The notion of maximal bicluster is *the same for any type of bicluster*, and given in the following definition.

Definition 4. (Maximal bicluster) Given a dataset (G, M, I), a bicluster (A, B) is a maximal bicluster if adding an object $g \in G \backslash A$ to A or an attribute $m \in M \backslash B$ to B does not result in a bicluster.

4 The Pattern Structures of Signed Partition

Table 2. Running example.

$\mathcal{M}$	m_1	m_2	m_3	m_4
g_1	$+$	$+$	$-$	$-$
g_2	$+$	$+$	$-$	$-$
g_3	$-$	$-$	$+$	$-$
g_4	$+$	$+$	$+$	$+$
g_5	$-$	$-$	$-$	$-$

4.1 Formalization

In the task of CSC bicluster discovery in a formal context (G, M, I), we propose an approach based on partition pattern structures. Instead of partition of objects in G as described in [2, 5], here we use *partition of attributes* in M. It is still similar to an object partition since an attribute partition covers every attribute in M and there is no overlapping between any two partition components.

To formally define our signed partition, first we define the notion of signed attribute and signed partition component as follows.

Definition 5 (Signed attribute)**.** Let M be a set of attributes, $m \in M$ be an attribute, and $* \in \{-, +\}$ be a sign. A *signed attribute* m^* is an attribute m having a sign $*$.

Definition 6 (Signed partition component)**.** A *signed partition component* (or *sp-component*) c is a subset of M, where each attribute in c is associated to their corresponding sign $*$. Therefore, $c = \{m_1^*, \cdots, m_n^*\}$.

For example, m_1^+ is a signed attribute where the sign $+$ is given to m_1, and $\{m_1^+, m_2^-, m_4^+\}$ is a signed partition component. Since an sp-component contains not only attributes but also their associated sign, we define the equality of two sp-components according to these two aspects as follows.

Definition 7 (SP-component equality)**.** Any two sp-components are equal iff both contain the same set of attributes, and they have either entirely same sign or entirely opposite sign.

Therefore, if we have $c_1 = \{m_1^+, m_2^-, m_4^+\}$, $c_2 = \{m_1^+, m_2^-, m_4^+\}$, and $c_3 = \{m_1^-, m_2^+, m_4^-\}$, then $c_1 = c_2 = c_3$.

Definition 8 (Signed partition)**.** A *signed partition* (or *s-partition*) d is a collection of sp-components, written as $d = \{c_1, \cdots, c_n\}$, such that every attribute in M is present in exactly one sp-component.

For example, given $M = \{m_1, \cdots, m_4\}$, then $\{\{m_1^+, m_2^-, m_4^+\}, \{m_3^+\}\}$ is a valid signed partition of M. The set of all possible s-partitions is denoted as D. This allows us to create an s-partition mapping $\delta : G \to D$ which assigns an object to an s-partition over M. For an object m, $\delta(m)$ is an s-partition containing only one sp-component. This sp-component contains all attributes in M with the corresponding sign according to the object g. Example from Table 2:

$$\begin{aligned}
\delta(g_1) &= \delta(g_2) = \{\{m_1^+, m_2^+, m_3^-, m_4^-\}\} \\
\delta(g_3) &= \{\{m_1^-, m_2^-, m_3^+, m_4^-\}\} \\
\delta(g_4) &= \{\{m_1^+, m_2^+, m_3^+, m_4^+\}\} \\
\delta(g_5) &= \{\{m_1^-, m_2^-, m_3^-, m_4^-\}\}.
\end{aligned}$$

Notice that since the sp-components in $\delta(g_4)$ and $\delta(g_5)$ contain the same attributes with entirely opposite sign, according to Def. 7 we have $\delta(g_4) = \delta(g_5)$. This mapping is formulated as follows:

$$\begin{aligned}
\delta(g) &= \{\{m_j^{*_j} | m_j \in M\}\} \\
&\text{where } *_j = m_j(g).
\end{aligned} \tag{1}$$

4.2 Signed Partition Space

For the task of CSC bicluster discovery, here we define relations between any two s-partitions. The set of all possible s-partitions D is a meet-semilattice where we can define the meet of any two s-partitions.

First, we define the notation $m(c)$ as the sign of an attribute m in an sp-component c. For example, if $c = \{m_1^+, m_2^-, m_3^-\}$, then $m_1(c) = +$. With this notation, we define the similarity ($\cap^{\pm}$) between any two sp-components as:

$$\begin{aligned}
c_1 \cap^{\pm} c_2 = \{&\{m_j^* \in c_1 | m_j(c_1) = m_j(c_2)\}, \\
&\{m_j^* \in c_1 | m_j(c_1) = \neg m_j(c_2)\}\},
\end{aligned} \tag{2}$$

where $*$ corresponds to the sign of m_j in c_1, i.e. $m_j(c_1)$.

In other words, the operator $\cap^{\pm}$ between c_1 and c_2 gives $\{c_{12}, c_{1|2}\}$. The c_{12} represents all attributes who are present in c_1 and c_2 with the same sign, while $c_{1|2}$ represents all attributes who are present in c_1 and c_2, but with opposite

sign. The signs in the resulting sp-component are the same as those in the first sp-component. Example:

$$\begin{aligned} \text{if } c_x &= \{m_1^+, m_2^-, m_3^-, m_4^-\} \\ \text{and } c_y &= \{m_1^+, m_2^-, m_3^+, m_4^+, m_5^-\}, \\ \text{then } c_x \cap^{\pm} c_y &= \{\{m_1^+, m_2^-\}, \{m_3^-, m_4^-\}\}. \end{aligned}$$

Since the signs in $c_{1|2}$ follow the first sp-component, the result of $c_1 \cap^{\pm} c_2$ could be different to $c_2 \cap^{\pm} c_1$. This can be resolved by Def. 7 that ensures the commutativity of $\cap^{\pm}$. For example:

$$\begin{aligned} c_x \cap^{\pm} c_y &= \{\{m_1^+, m_2^-\}, \{m_3^-, m_4^-\}\}, \\ c_y \cap^{\pm} c_x &= \{\{m_1^+, m_2^-\}, \{m_3^+, m_4^+\}\}, \\ c_x \cap^{\pm} c_y &= c_y \cap^{\pm} c_x. \end{aligned}$$

Having defined the similarity of any two sp-components, we can now define the similarity of any two s-partitions. The similarity (or the meet) of two s-partitions $d_1 = \{c_1 \cdots c_k\}$ and $d_2 = \{c_1 \cdots c_n\}$, with $k = |d_1|$ and $n = |d_2|$, is defined as:

$$d_1 \sqcap d_2 = \{c_i \cap^{\pm} c_j | \forall c_i \in d_1, c_j \in d_2\}, \tag{3}$$

and the order between two s-partitions is given by:

$$d_1 \sqsubseteq d_2 \iff d_1 \sqcap d_2 = d_1. \tag{4}$$

Let C the set of all sp-components in M, and D is the set of all s-partitions in M. We have $\cap^{\pm} : C^2 \to D$ and $\sqcap : D^2 \to D$. Example from Table 2:

$$\begin{aligned} \delta(g_1) \sqcap \delta(g_3) &= \{\{m_1^+, m_2^+, m_3^-, m_4^-\}\} \sqcap \{\{m_1^-, m_2^-, m_3^+, m_4^-\}\} \\ &= \{\{m_4^-\}, \{m_1^+, m_2^+, m_3^-\}\}. \end{aligned}$$

Suppose that $d_1 = \{\{m_4^-\}, \{m_1^+, m_2^+, m_3^-\}\}$. Then $d_1 \sqsubseteq \delta(g_1)$, $d_1 \sqsubseteq \delta(g_2)$, and $d_1 \sqsubseteq \delta(g_3)$.

In order to define a partial order among $d \in D$, the $\sqcap$ operator has to be commutative, idempotent, and associative. These properties are shown in the following propositions.

Proposition 1. *The operator $\sqcap$ is commutative, i.e. $d_1 \sqcap d_2 = d_2 \sqcap d_1$.*

Proof. Consider $d_1 = \{c_1 \cdots c_n\}$ with $n = |d_1|$ and $d_2 = \{c_1 \cdots c_k\}$ with $k = |d_2|$.

$$\begin{aligned} d_1 \sqcap d_2 &= d_2 \sqcap d_1 \\ \{c_i \cap^{\pm} c_j | \forall c_i \in d_1, c_j \in d_2\} &= \{c_j \cap^{\pm} c_i | \forall c_i \in d_1, c_j \in d_2\} \end{aligned}$$

It is previously stated that $\cap^{\pm}$ is also commutative. Therefore, both sides of the equation above are equal. □

Proposition 2. *The operator $\sqcap$ is idempotent, i.e. $d_1 \sqcap d_1 = d_1$.*

Proof. Consider $d_1 = \{c_1 \cdots c_n\}$ with $n = |d_1|$.

$$d_1 \sqcap d_1 = \{c_i \cap^{\pm} c_j | \forall c_i \in d_1, c_j \in d_1\}$$

Since there is no overlap among $c_i \in d_1$, then $c_i \cap^{\pm} c_j$ is an empty set for $i \neq j$. Therefore :

$$\begin{aligned} d_1 \sqcap d_1 &= \{c_i \cap^{\pm} c_j | \forall c_i, c_j \in d_1 \text{ and } i = j\} \\ &= \{c_i \cap^{\pm} c_i | \forall c_i \in d_1\} \\ &= \{c_i | c_i \in d_1\} \\ &= d_1 \end{aligned}$$

□

Proposition 3. *The operator $\sqcap$ is associative, i.e. $(d_1 \sqcap d_2) \sqcap d_3 = d_1 \sqcap (d_2 \sqcap d_3)$.*

Proof. Consider $d_1 = \{c_1 \cdots c_n\}$ with $n = |d_1|$, $d_2 = \{c_1 \cdots c_p\}$ with $p = |d_2|$, and $d_3 = \{c_1 \cdots c_q\}$ with $q = |d_3|$.

$$\begin{aligned} & (d_1 \sqcap d_2) \sqcap d_3 \\ =& \{c_i \cap^{\pm} c_j | \forall c_i \in d_1, c_j \in d_2\} \sqcap d_3 \\ =& \{c_i \cap^{\pm} c_j \cap^{\pm} c_k | \forall c_i \in d_1, c_j \in d_2, c_k \in d_3\} \\ =& d_1 \sqcap \{c_j \cap^{\pm} c_k | \forall c_j \in d_2, c_k \in d_3\} \\ =& d_1 \sqcap (d_2 \sqcap d_3) \end{aligned}$$

□

With the definition of similarity ($\sqcap$) and the associated partial ordering ($\sqsubseteq$) between two s-partitions, we then define the notion of signed partition pattern concept in the following subsection.

4.3 Signed Partition Pattern Structures

Let G a set of objects, M a set of attributes. The lattice of s-partitions of M is $(D, \sqcap)$, where $\delta : G \to D$ maps an object to an s-partition as defined in Eq. 1.

A *signed partition pattern structure* is determined by the triple $(G, (D, \sqcap), \delta)$, where the derivation operators for $A \subseteq G$ and $d \in D$ are defined as:

$$A^{\square} = \bigsqcap_{g \in A} \delta(g), \tag{5}$$

$$d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\}. \tag{6}$$

(A, d) is a *signed partition pattern concept* (or *spp-concept*) when $A^{\square} = d$ and $d^{\square} = A$. From an spp-concept (A, d), a `CSC` bicluster is any pair (A, c) where $c \in d$ (we can ignore the attribute signs in c). All spp-concepts from Table 2 are listed in Table 3. In the concept $(\{g_1, g_2, g_3\}, \{\{m_1^+, m_2^+, m_3^-\}, \{m_4^-\}\})$ for

Table 3. All sign partition pattern concepts from Table 2 and their corresponding CSC biclusters.

Concept		CSC bicluster	
Extent	Intent	Objects	Attributes
$\{g_1, g_2\}$	$\{\{m_1^+, m_2^+, m_3^-, m_4^-\}\}$	$\{g_1, g_2\}$	$\{m_1, m_2, m_3, m_4\}$
$\{g_3\}$	$\{\{m_1^-, m_2^-, m_3^+, m_4^-\}\}$	$\{g_3\}$	$\{m_1, m_2, m_3, m_4\}$
$\{g_4, g_5\}$	$\{\{m_1^+, m_2^+, m_3^+, m_4^+\}\}$	$\{g_4, g_5\}$	$\{m_1, m_2, m_3, m_4\}$
$\{g_1, g_2, g_3\}$	$\{\{m_1^+, m_2^+, m_3^-\}, \{m_4^-\}\}$	$\{g_1, g_2, g_3\}$	$\{m_1, m_2, m_3\}$
		$\{g_1, g_2, g_3\}$	$\{m_4\}$
$\{g_1, g_2, g_4, g_5\}$	$\{\{m_1^+, m_2^+\}, \{m_3^+, m_4^+\}\}$	$\{g_1, g_2, g_4, g_5\}$	$\{m_1, m_2\}$
		$\{g_1, g_2, g_4, g_5\}$	$\{m_3, m_4\}$
$\{g_3, g_4, g_5\}$	$\{\{m_1^+, m_2^+, m_4^+\}, \{m_3^+\}\}$	$\{g_3, g_4, g_5\}$	$\{m_1, m_2, m_4\}$
		$\{g_3, g_4, g_5\}$	$\{m_3\}$
$\{g_1, g_2, g_3, g_4, g_5\}$	$\{\{m_1^+, m_2^+\}, \{m_3^+\}, \{m_4^+\}\}$	$\{g_1, g_2, g_3, g_4, g_5\}$	$\{m_1, m_2\}$
		$\{g_1, g_2, g_3, g_4, g_5\}$	$\{m_3\}$
		$\{g_1, g_2, g_3, g_4, g_5\}$	$\{m_4\}$

example, we can find the CSC bicluster $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$. Looking back to the original table, this CSC bicluster means that in $A = \{g_1, g_2, g_3\}$, we have $m_1(A) \simeq m_2(A) \simeq m_3(A)$ (recall the definition of $\simeq$ in Section 3).

The order between any two spp-concepts is given by $(A_1, d_1) \leq (A_2, d_2) \iff A_1 \subseteq A_2$ or $d_2 \sqsubseteq d_1$. Using this order, the lattice of all spp-concepts from Table 2 can be constructed and is shown in Figure 1. It should be noticed that the lattice is readable and interpretable only if its size is small. This lattice is useful not only for understanding the hierarchical structure among all biclusters, but also for detecting maximal biclusters.

By Def. 4, the bicluster $(\{g_1, g_2, g_4, g_5\}, \{m_1, m_2\})$ from Table 2 is not maximal, since we can add g_3 that constructs another bicluster $(\{g_1, g_2, g_3, g_4, g_5\}, \{m_1, m_2\})$. The non-maximal biclusters can be detected from the concept lattice [16]. Consider two concepts (A_1, d_1) and (A_2, d_2) such that $(A_1, d_1) \leq (A_2, d_2)$, and an sp-component c. The bicluster (A_1, c) is maximal iff $c \in d_1$ and $c \notin d_2$.

For example, consider the concept $p_1 = (\{g_1, g_2, g_4, g_5\}, \{\{m_1^+, m_2^+\}, \{m_3^+, m_4^+\}\})$ and $p_2 = (\{g_1, g_2, g_3, g_4, g_5\}, \{\{m_1^+, m_2^+\}, \{m_3^+\}, \{m_4^+\}\})$, where $p_1 \leq p_2$. We see that the sp-component $\{m_1^+, m_2^+\}$ is in the intent of both p_1 and p_2. Therefore, the bicluster $(\{g_1, g_2, g_4, g_5\}, \{m_1, m_2\})$ from p_1 is not maximal.

4.4 Mining SPP Concepts

As described in Section 3, CSC bicluster is a submatrix in a binary matrix. Therefore, given a numerical matrix, it is required to transform it into binary matrix. This can be done by scaling, for example by introducing a threshold, and each numerical value can be transformed to $+$ or $-$ based on whether it is above or below the threshold. In a gene expression data for example, a threshold can be the normal expression level for each gene. An expression that is above (or below) this normal level should be transformed to $+$ (or $-$ respectively).

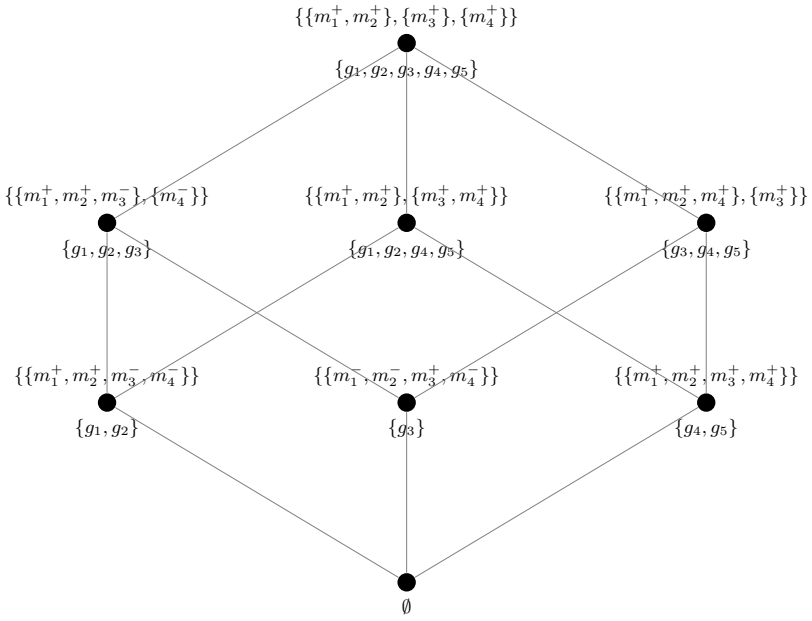

Fig. 1. Sign partition pattern lattice for pattern structure in Table 2.

In the task of mining formal concepts, the algorithm called AddIntent was proposed in [19]. This algorithm can be used for any pattern structures by defining the meet ($\sqcap$) and the order ($\sqsubseteq$) between any two descriptions. Having defined the meet in Eq. 3 and the order in Eq. 4, we then use AddIntent to mine spp-concepts in a binary matrix. Furthermore, this algorithm is also effective for building a concept lattice, which is needed in our case to detect the maximality of any CSC bicluster.

5 Experiments

As previously explained in Section 4, CSC biclusters can be found in any spp-concept. Therefore, from a binary matrix, we should retrieve all spp-concepts. To do that, we reuse the AddIntent source code in [4] by modifying the definition $\sqcap$ and $\sqsubseteq$ operators. This algorithm also allows us to build the lattice of all concepts. We can reduce the lattice by choosing a threshold θ that applies to the intent of a concept. This threshold defines the minimal size of an sp-component that an intent should have. Since the lattice construction is performed by a bottom-up approach, this threshold allows to "prune" the lattice. For example, with $\theta = 3$, the lattice for Table 2 does not contain the concept $(\{g_1, g_2, g_4, g_5\}, \{\{m_1^+, m_2^+\}, \{m_3^+, m_4^+\}\})$–since none of its sp-components has ≥ 3 attributes–as shown in Figure 2.

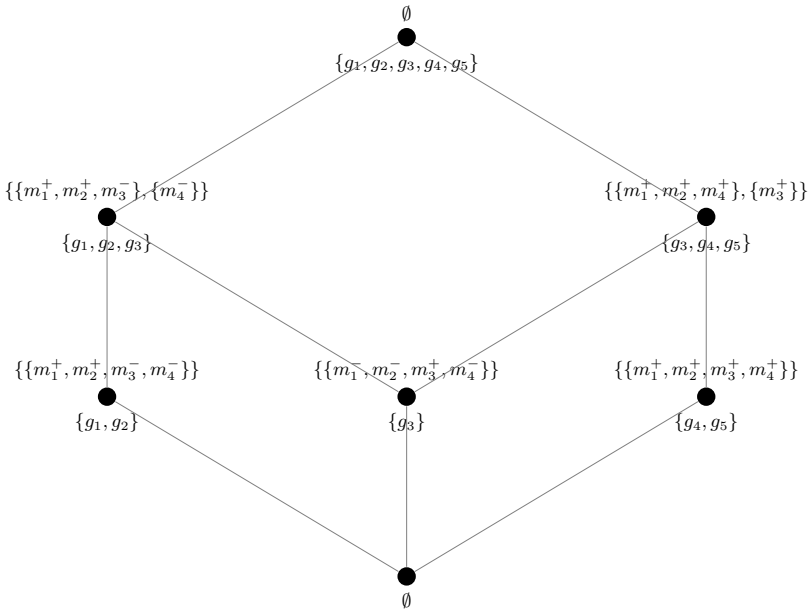

Fig. 2. Sign partition pattern lattice for pattern structure in Table 2 with $\theta = 3$.

We tested our method to lymphoma dataset provided in [1]. It contains the numerical expression levels of 4026 genes over 96 tissues. The objective of CSC bicluster discovery in this dataset is to find a subset of genes that behave in a consistent way over a subset of tissues. For this task, we convert this numerical dataset to binary by assigning $-$ and $+$ for the values < 0 and ≥ 0 respectively.

The number of concepts and runtime for different thresholds are listed in Table 4. We tested three thresholds: 70, 80, and 90. As shown here, higher θ can reduce the number of concepts, and consequently reduce the runtime.

For $\theta = 70$, around 157K concepts are obtained. Among them, only 153K have extent size larger than 1. This means that there are 153K CSC biclusters having at least 70 columns and at least 2 rows. Furthermore, still with $\theta = 70$, the largest extent size is 8, meaning that among the biclusters with ≥ 70 columns, there are no bicluster with > 8 rows.

Higher θ corresponds to higher number of columns in the biclusters and thus lower number of rows. With $\theta = 90$, we see that among the biclusters with ≥ 90 rows, there are no bicluster with > 3 rows.

6 Conclusion

In this paper we have presented an approach to mine biclusters with coherent sign changes in a binary matrix. We formulated our method based on partition pattern structures. The generated lattice allows us to examine the maximality

Table 4. Experiments on lymphoma dataset.

θ	Runtime (minutes)	Number of concepts All	Number of concepts Extent size > 1	Largest extent size
70	229.4	157K	153K	8
80	62.9	7K	2K	7
90	62.1	4K	83	3

of discovered biclusters. Another advantage is that we can choose a threshold that defines a minimum number of columns of the biclusters. This is also useful to reduce the number of concepts and the computational time.

The computational time can be further reduced by taking into account the maximality of a bicluster. If the intent of a concept p contains an sp-component c that is present in the intent of p's superconcept, then this c indicates a non-maximal bicluster. In this case, c should be removed from the intent of p. However, this may change the definition of $\sqcap$.

Another aspect that should be studied is the possibility of a matrix that has another sign in addition to $+$ and $-$. This new sign can represent a missing value, or in the case of threshold-based transformation, a value that is equal to the threshold. It can be resolved using tolerance relation introduced in [14], such that a value equal to the threshold should be regarded as similar to both $+$ and $-$. In the case of missing value, it can be resolved by modifying the definition of attribute partition in Def. 8 which permits an attribute to be not present in any sp-component. This modification may consequently require modifications on the definition of meet and order between s-partitions.

Eventually, the `CSC` bicluster discovery can be applied in a domain besides gene expression data. Frequent gradual itemset mining was studied in [7] to extract gradual rules from a numerical table, e.g. a hotel price table with 3 attributes: m_p for city population, m_d for distance from city center, and m_r for room price. We may find an sp-component $\{m_p^+, m_d^-, m_r^+\}$. It is related to the rule saying "the more/less m_p, the less/more m_d, then the more/less m_r".

Moreover, some studies ([6, 12]) show the benefits of biclustering in the recommendation systems. In a user–movie rating matrix for example, a constant-column bicluster represents a set of users having the same interest across a set of movies. On the other hand, a `CSC` bicluster in this matrix represents a set of users having either the same or the opposite interest. This is useful for a new user u: we can recommend movies liked by users similar to u and movies disliked by users opposite to u.

References

1. Alizadeh, A.A., Eisen, M.B., Davis, R.E., Ma, C., Lossos, I.S., Rosenwald, A., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., et al.: Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature **403**(6769), 503 (2000)

2. Baixeries, J., Kaytoue, M., Napoli, A.: Characterizing functional dependencies in formal concept analysis with pattern structures. Annals of Mathematics and Artificial Intelligence **72**, 129–149 (2014)
3. Cheng, Y., Church, G.M.: Biclustering of expression data. In: ISMB. vol. 8, pp. 93–103 (2000)
4. Codocedo, V., Bosc, G., Kaytoue, M., Boulicaut, J.F., Napoli, A.: A proposition for sequence mining using pattern structures. In: International Conference on Formal Concept Analysis. pp. 106–121. Springer (2017)
5. Codocedo, V., Napoli, A.: Lattice-based biclustering using partition pattern structures. In: Proceedings of the Twenty-first European Conference on Artificial Intelligence. pp. 213–218. IOS Press (2014)
6. Codocedo-Henríquez, V.: Contributions à l'indexation et à la récupération d'information utilisant l'analyse formelle de concepts. Ph.D. thesis, Université de Lorraine (2015)
7. Di-Jorio, L., Laurent, A., Teisseire, M.: Mining frequent gradual itemsets from large databases. In: International Symposium on Intelligent Data Analysis. pp. 297–308. Springer (2009)
8. Gnatyshak, D., Ignatov, D.I., Semenov, A., Poelmans, J.: Analysing online social network data with biclustering and triclustering. In: Proceedings of the "Concept Discovery in Unstructured Data" conference. vol. 871, pp. 30–39. Citeseer (2012)
9. Govaert, G., Nadif, M.: Co-clustering. Wiley-IEEE Press (2013)
10. Hartigan, J.A.: Direct clustering of a data matrix. Journal of the american statistical association **67**(337), 123–129 (1972)
11. Ignatov, D.I., Kuznetsov, S.O., Poelmans, J.: Concept-based biclustering for internet advertisement. In: Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on. pp. 123–130. IEEE (2012)
12. Ignatov, D.I., Poelmans, J., Zaharchuk, V.: Recommender system based on algorithm of bicluster analysis RecBi. arXiv preprint arXiv:1202.2892 (2012)
13. Kaytoue, M.: Traitement de données numériques pas analyse formelle de concepts et structures de patrons. Ph.D. thesis, Université Henri Poincare – Nancy 1 (2011)
14. Kaytoue, M., Assaghir, Z., Napoli, A., Kuznetsov, S.O.: Embedding tolerance relations in formal concept analysis: an application in information fusion. In: Proceedings of the 19th ACM international conference on Information and knowledge management. pp. 1689–1692. ACM (2010)
15. Kaytoue, M., Kuznetsov, S.O., Macko, J., Napoli, A.: Biclustering meets triadic concept analysis. Annals of Mathematics and Artificial Intelligence **70**(1-2), 55–79 (2014)
16. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Biclustering numerical data in formal concept analysis. In: International Conference on Formal Concept Analysis. pp. 135–150. Springer (2011)
17. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining Gene Expression Data with Pattern Structures in Formal Concept Analysis. Information Science **181**(10), 1989–2001 (2011)
18. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) **1**(1), 24–45 (2004)
19. van der Merwe, D., Obiedkov, S., Kourie, D.: AddIntent: A new incremental algorithm for constructing concept lattices. In: International Conference on Formal Concept Analysis. pp. 372–385. Springer (2004)
20. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics **18**(suppl_1), S136–S144 (2002)