A Unified Approach to Biclustering Based on Formal Concept Analysis and Interval Pattern Structure

Nyoman Juniarta, Miguel Couceiro, Amedeo Napoli

A Unified Approach to Biclustering Based on Formal Concept Analysis and Interval Pattern Structure

Nyoman Juniarta[1][0000−0003−4081−2866], Miguel Couceiro[1], and Amedeo Napoli[1]

Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
{nyoman.juniarta, miguel.couceiro, amedeo.napoli}@loria.fr

Abstract. In a matrix representing a numerical dataset, a bicluster is a submatrix whose cells exhibit similar behavior. Biclustering is naturally related to Formal Concept Analysis (FCA) where concepts correspond to maximal and closed biclusters in a binary dataset. In this paper, a unified characterization of biclustering algorithms is proposed using FCA and pattern structures, an extension of FCA for dealing with numbers and other complex data. Several types of biclusters – constant-column, constant-row, additive, and multiplicative – and their relation to interval pattern structures is presented.

Keywords: biclustering, FCA, gene expression, pattern structures

1 Introduction

Given a numerical dataset represented as a table or a matrix with objects in rows and attributes in columns, the objective of clustering is to group a set of objects according to all attributes using a similarity or distance measure. By contrast, biclustering simultaneously operates on the set of objects and the set of attributes, where a subset of objects can be grouped w.r.t. a subset of attributes, based on user-defined constraints such as having constant values, constant values within columns or rows. Then, if a cluster represents object relations at a global scale, a bicluster represents it at a local scale w.r.t. the set of attributes. More generally, biclustering searches in a data matrix for sub-matrices or biclusters composed of a subset of objects (rows) and a subset of attributes (columns) which exhibit a specific behavior w.r.t. some criteria.

Biclustering is an important tool in many domains, e.g. bioinformatics and gene expression data, recommendation and collaborative filtering, text mining, social networks, dimensionality reduction, etc. As surveyed in [17], biclustering received a lot of attention in biology, and especially, for analyzing gene expression data, where biologists are searching for a set of genes whose behavior is consistent across certain experiments/conditions [3, 4, 20]. Biclustering is still actively studied in biology [18, 9, 19]. Biclustering is also actively studied in recommendation systems [12, 13], where the objective is to retrieve a set of users sharing similar interest across a subset of items instead of the set of all possible items.

Table 1: Examples of some bicluster types.

(a) constant-column

4	2	5	3
4	2	5	3
4	2	5	3
4	2	5	3

(b) similar-column

4.0	2.5	5.7	3.1
3.9	2.4	5.6	3.0
4.1	2.4	5.9	2.9
4.1	2.6	5.8	2.9

(c) additive

2	3	5	1
5	6	8	4
1	2	4	0
4	5	7	3

(d) multiplicative

2	6	12	4
1	3	6	2
4	12	24	8
3	9	18	6

Following the lines of [9, 10, 8], in this paper we are interested in biclustering algorithms based on "pattern-mining" techniques [1]. These techniques allow an exhaustive and flexible search with efficient algorithms. Moreover, authors in [9] discuss the benefits of using pattern-based biclustering w.r.t. scalability requirements, and mostly w.r.t. generality and diversity of the types of biclusters which are mined. In addition, they point out the fact that pattern-based biclustering algorithms can naturally take into account overlapping biclusters, and as well, additive, multiplicative and symmetric assumptions concerning biclusters.

In this paper, we revisit all these aspects and propose an alternative framework for pattern-based biclustering based on Formal Concept Analysis (FCA [7]). In [21], authors directly reuse the FCA framework and adapt the algorithms for biclustering. By contrast, in this paper, we go further and we consider the so-called "pattern-structures", an extension of FCA for dealing with complex values such as numbers, sequences, or graphs [6]. We especially reuse "interval pattern structures" – which are detailed in the following – for defining a unique framework for pattern-based biclustering. In this way, we introduce an alternative approach than [9], as we do not need to apply any scaling, discretization, or transformation procedures over the data to discover biclusters.

This paper is organized as follows. First we describe some types of biclustering in Section 2 and basic definitions about FCA in Section 3. We then propose our approach of biclustering based on interval pattern structures in Section 4 and present the empirical experiments in Section 5. Finally, we conclude our work and give some future works in Section 6.

2 Biclustering

In this section, we recall the basic background and discuss illustrative examples of the different types of biclusters [17]. We consider that a dataset is a matrix (G, M) where G is a set of objects and M is a set of attributes. The value of $m \in M$ for object $g \in G$ is written as $m(g)$. In this paper, we work with numerical datasets. In such a dataset, it may be interesting to find which subset of objects have the same values w.r.t. a subset of attributes. Regarding the matrix representation, this is equivalent to the problem of finding a submatrix where

Table 2: A numerical context and an SC bicluster in gray.

G	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	2	1	1	0	6
g_3	2	2	1	7	6
g_4	8	9	2	6	7

all elements have the same value. This task is called *biclustering with constant values*, which is a simultaneous clustering of the rows and columns of a matrix.

Moreover, given a dataset (G, M), a pair (A, B) (where $A \subseteq G$, $B \subseteq M$) is a *constant-column* (CC) bicluster iff $\forall m \in B, \forall g, h \in A, m(g) = m(h)$. An example of CC bicluster is illustrated in Table 1a. CC biclustering have more relaxed variations, namely similar-column (SC) biclustering. With these relaxations, instead of finding biclusters with exactly constant columns, we can obtain biclusters whose columns have similar values as shown in Table 1b. These types of biclusters are widely used in recommendation systems to detect a set of users sharing similar preference over a set of items.

An additive bicluster is illustrated in Table 1c. Here we see that there is a constant difference between any two columns. For example, each value in the second column is two more than the corresponding value in the fourth row. Therefore, given a dataset (G, M), a pair (A, B) (where $A \subseteq G$, $B \subseteq M$) is an *additive* bicluster iff $\forall g, h \in A, \forall m, n \in B, m(g) - n(g) = m(h) - n(h)$; or a *multiplicative* bicluster iff $\forall g, h \in A, \forall m, n \in B, m(g)/n(g) = m(h)/n(h)$. Both additive and multiplicative biclusters were studied in the domain of gene expression dataset [4, 16, 5]. They represent a set of genes having similar expression patterns across a set of experiments.

Bicluster discovery is naturally related to FCA. In this paper, we show that an extension of FCA called partition pattern structures can be used for discovering biclusters. In the following section, we explain some basic theories about FCA and pattern structures.

3 FCA and Pattern Structure

In a binary matrix, FCA tries to find maximal submatrices with a constant value across all of its cells. Therefore, a formal concept is a bicluster with constant value. More precisely, FCA is a mathematical framework based on lattice theory and used for classification, data analysis, and knowledge discovery [7]. From a formal context, FCA detects all formal concepts, and arranges them in a concept lattice. FCA is restricted to specific datasets where each attribute is binary (e.g. has only yes/no value). This limitation prohibits FCA to work in more complex datasets, e.g. a user-rating matrix or a gene expression dataset, which are not binary. Therefore, FCA is then generalized into pattern structures [6].

A *pattern structure* is a triple $(G, (D, \sqcap), \delta)$, where G is a set of objects, $(D, \sqcap)$ is a complete meet-semilattice (of descriptions), and $\delta : G \to D$ maps an object to

Table 3: Example of additive column alignments. (a) Original table and the additive bicluster in gray, (b) alignment on m_1, (c) alignment on m_2.

	m_1	m_2	m_3	m_4
g_1	4	1	3	0
g_2	6	4	6	3
g_3	2	3	5	2
g_4	1	6	1	7

(a)

	m_1	m_2	m_3	m_4
g_1	4	1	3	0
g_2	4	2	4	1
g_3	4	5	7	4
g_4	4	9	4	10

(b)

	m_1	m_2	m_3	m_4
g_1	4	1	3	0
g_2	3	1	3	0
g_3	0	1	3	0
g_4	-4	1	-4	2

(c)

a description. The operator $\sqcap$ is a similarity operation that returns the common elements between any two descriptions. It is verified that $c \sqcap d = c \Leftrightarrow c \sqsubseteq d$. A description can be a number, a set, a sequence, a tree, a graph, or another complex structure. The Galois connection for a pattern structure $(G, (D, \sqcap), \delta)$ is defined as:

$$A^\diamond = \prod_{g \in A} \delta(g), \qquad A \subseteq G, \tag{1}$$

$$d^\diamond = \{g \in G | d \sqsubseteq \delta(g)\}, \qquad d \in D. \tag{2}$$

A pattern concept is a pair (A, d), $A \subseteq G$ and $d \in D$, where $A^\diamond = d$ and $d^\diamond = A$.

FCA can be understood as a particular pattern structure. The description of an object is a set of attributes, and the $\sqcap$ operator between two description is the intersection of two sets of attributes.

4 Biclustering using Interval Pattern Structure

In gene expression data, we often have a numerical matrix. Biclustering in such matrix should find submatrices whose cells present regularities, e.g. each column has similar value in the case of similar-column (SC) biclustering. SC biclustering task is similar to FCA in the sense that FCA also searches consistent submatrix. But since SC biclustering works on a numerical matrix, we need to generalize FCA to a pattern structure. One such generalization is where the description of each object is a set of numerical values and the similarity between any two descriptions is the intervals that encompass those values. This kind of pattern structure is called an *interval pattern structure*.

Interval pattern structures (IPS) was introduced by Kaytoue et al. [14] to analyze gene expression data (GED). A GED is typically represented as a 2-D numerical matrix with genes as rows and conditions as columns, as shown in Table 2. In this matrix, the submatrix $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3, m_5\})$ is an SC bicluster, defined by the parameter $\theta = 1$. It means that the range of values of each column in the submatrix has the length of at most 1.

4.1 Interval Pattern Structure

In IPS, a description is several intervals describing the values of every column. For example, the description of g_1 – denoted by $\delta(g_1)$ – in Table 2 is $\langle[1, 1][2, 2][2, 2]$

Table 4: Some interval pattern concepts with $\theta = 1$ from Table 2.

Extent	Intent
$\{g_1\}$	$\langle[1,1][2,2][2,2][1,1][6,6]\rangle$
$\{g_1, g_3\}$	$\langle[1,2][2,2][1,2] * [6,6]\rangle$
$\{g_1, g_4\}$	$\langle * * [2,2] * [6,7]\rangle$
$\{g_1, g_2, g_3\}$	$\langle[1,2][1,2][1,2] * [6,6]\rangle$
$\{g_1, g_2, g_3, g_4\}$	$\langle * * [1,2] * [6,7]\rangle$

$[1,1][6,6]\rangle$. The similarity operator ($\sqcap$) for IPS is defined as the convex hull of two intervals. Therefore, the similarity of $\delta(g_1)$ and $\delta(g_4)$ – denoted by $\delta(g_1) \sqcap \delta(g_4)$ – is $\langle[1,8][2,9][2,2][1,6][6,7]\rangle$.

Given a subset of objects $A \subseteq G$, Eq. 1 says that $A^\diamond$ is the similarity of the description of all objects in A. Therefore, in IPS the corresponding $A^\diamond$ is the convex hull of the descriptions of all objects in A. For example, with $A = \{g_1, g_2, g_4\}$, $A^\diamond = \langle[1,8][1,9][1,2][0,6][6,7]\rangle$.

Furthermore, given a description $d \in D$, Eq. 2 indicates that $d^\diamond$ is the set of all objects whose description subsumes d. In IPS, a description d_1 is subsumed by another description d_2 – denoted by $d_1 \sqsubseteq d_2$ – if every interval in d_2 is a sub interval in the corresponding interval in d_1. Notice that in IPS, a sub interval subsumes a larger interval. Therefore, if $d_x = \langle[1,8][1,9][1,2][0,6][6,7]\rangle$, then $d_x^\diamond = \{g_1, g_2, g_4\}$. Since $\delta(g_3) = \langle[2,2][2,2][1,1][7,7][6,6]\rangle$, g_3 is not included in $d_x^\diamond$ because the fourth interval ($[7,7]$) is not sub interval of the fourth interval of d_x ($[0,6]$).

Following the definition of a concept of any pattern structure (in Sec. 3), an interval pattern concept is a pair (A, d), for $A \subseteq G$ and $d \in D$, where $A^\diamond = d$ and $d^\diamond = A$. Furthermore, the set of interval pattern concepts are partially ordered, and can be depicted as a lattice. An interval pattern concept (A_1, d_1) is a subconcept of (A_2, d_2) if $A_1 \subseteq A_2$ (equivalently $d_2 \sqsubseteq d_1$).

4.2 Similar-column biclustering

A similar-column (SC) bicluster can be found in an interval pattern concept by introducing a parameter θ. This parameter acts as the maximum difference between any two values to be considered as similar. For example, with $\theta = 1$, the value 1 is similar to 2, but not similar to 3.

In calculating the similarity between any two descriptions, if the length of an interval is larger than θ, then the star sign ($*$) is put as the interval. From Table 2, $\delta(g_2) \sqcap \delta(g_4)$ without θ is $\langle[2,8][1,9][1,2][0,6][6,7]\rangle$, and with $\theta = 1$ is $\langle * * [1,2] * [6,7]\rangle$.

The similarity $\sqcap$ between $*$ and any other interval is $*$. For example, suppose that we have two descriptions $d_x = \langle[1,1][2,3]\rangle$ and $d_y = \langle[2,2]*\rangle$. Then, $d_x \sqcap d_y = \langle[1,2]*\rangle$. This also means that $*$ is subsumed by any other interval. Therefore, the description of each object in Table 2 subsumes $\langle * * [1,2] * [6,7]\rangle$. With $\theta = 1$, $(\{g_1, g_2, g_3, g_4\}, \langle * * [1,2] * [6,7]\rangle)$ is an interval pattern concept. Some interval pattern concepts from Table 2 are listed in Table 4.

Table 5: Example of multiplicative column alignments. (a) Original table and the multiplicative bicluster in gray, (b) alignment on m_2.

	m_1	m_2	m_3	m_4
g_1	3	1	2	3
g_2	1	3	6	9
g_3	2	2	4	6
g_4	1	2	6	8

(a)

	m_1	m_2	m_3	m_4
g_1	3	1	2	3
g_2	0.3	1	2	3
g_3	1	1	2	3
g_4	0.5	1	3	4

(b)

From an interval pattern concept, an SC bicluster can be formed by the concept's extent and the set of columns where the interval is not $*$ in the concept's intent. For example, from the concept $(\{g_1, g_2, g_3\}, \langle[1,2][1,2][1,2]*[6,6]\rangle)$, $(\{g_1, g_2, g_3\}, \{m_2, m_2, m_3, m_5\})$ is an SC bicluster with $\theta = 1$.

By using IPS with parameter θ, constant-column biclustering is a specific case of SC biclustering. It can be noticed that with $\theta = 0$, we obtain intervals with length 0, and that corresponds to constant-column biclusters.

4.3 Additive and multiplicative biclustering

An additive bicluster is a submatrix where there is a constant (or similar) difference between any two columns across all of its rows (see Sec. 2). Constant (or similar) column biclustering is a specific case of additive biclustering. Using this fact, we can obtain additive biclusters by aligning (similar to [9]) each column, and then find interval pattern concepts on the alignments.

Table 3 provides an example of column alignment for additive biclustering. The original matrix is shown in Table 3a, having 4 rows and 4 columns. The submatrix $(\{g_1, g_2, g_3\}, \{m_2, m_3, m_4\})$ is an additive bicluster in the original matrix. This bicluster can be found by applying constant-column or similar-column biclustering to the column alignments. Table 3b shows the first column alignment, can be seen by the consistency of the first column (m_1). In this example, each object value is converted such that its m_1 value is equal to the value of m_1 in g_1. This means that the values 0, -2, 2, and 3 are added to g_1, g_2, g_3, and g_4 respectively. This alignment is repeated for every column. Table 3c is the alignment of m_2, by adding 0, -3, -2, and -5 to g_1, g_2, g_3, and g_4 respectively.

Constant-column (or similar-column) biclustering is applied to every column alignment to find additive biclusters. In the second column alignment (Table 3c), we obtain $(\{g_1, g_2, g_3\}, \{m_2, m_3, m_4\})$ as a constant-column bicluster. This corresponds to the additive bicluster $(\{g_1, g_2, g_3\}, \{m_2, m_3, m_4\})$ in the original matrix (Table 3a).

Multiplicative biclusters can also be obtained using similar column alignment. In multiplicative column alignment, instead of adding values to each row, we multiply each row such that a column has a constant value. Table 5b shows the second column alignment of the original matrix in Table 5a. Here, a constant value is achieved for m_2 by multiplying g_1, g_2, g_3, and g_4 by 1, $\frac{1}{3}$, $\frac{1}{2}$,

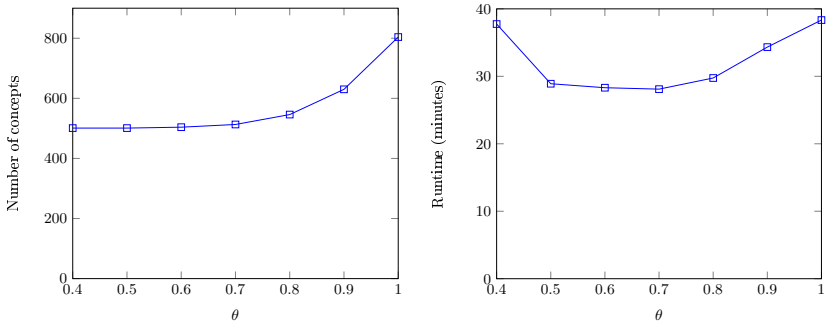

Fig. 1: Effect of θ on a 500×60 dataset with $min\_col = 20$ and $min\_row = 1$.

and $\frac{1}{2}$ respectively. Then, by applying IPS to each alignment, we can obtain the multiplicative biclusters. For example, constant-column biclustering using IPS in Table 5b returns $(\{g_1, g_2, g_3\}, \{m_2, m_3, m_4\})$, which is the corresponding multiplicative bicluster in Table 5a.

4.4 Concept mining

Being a generalization of FCA, the mining of interval pattern concepts can be performed using some existing algorithms that generate a complete list of formal concepts. In this paper, we use CloseByOne (CbO) [15] since it requires us to only define the similarity ($\sqcap$) and subsumption relation ($\sqsubseteq$) of any two descriptions.

In a given numerical matrix, we may obtain an exponential number of interval pattern concepts. To reduce the number of concepts, we should introduce some parameters that can filter out some uninteresting concepts.

The first parameter, θ, is previously mentioned in Sec. 4.2. It limits the length of intervals, and later in Sec. 5 we demonstrate the effect of θ on the runtime and number of concepts.

The second parameter min_col is the minimum number of columns in the retrieved biclusters. The number of columns in a bicluster corresponds to the number of non-star intervals in the concept's intent. For example, the concept with intent $\langle * * [2, 2] * [6, 7] \rangle$ gives us a bicluster with two columns (the third and the fifth). To take into account the min_col parameter, it is necessary to modify the definition of similarity between any two descriptions. In addition to the definition of $\sqcap$ in Sec. 4.1, we verify if the number of non-star intervals in the description. The number of non-star intervals should be more than min_{col}. If not, we "skip" the concept, by converting each interval to $*$. In Table 2 with $\theta = 1$, $g_1 \sqcap g_4$ is $\langle * * [2, 2] * [6, 7] \rangle$. Using $min\_col = 3$ for example, $g_1 \sqcap g_4$ becomes $\langle * * * * * \rangle$.

Related to min_col is min_row, a parameter that put a constraint on the number of rows in a bicluster. It corresponds to the number of objects in a concept's extent. With the inclusion of min_row, the calculation of $Y^{\diamond}$ (all

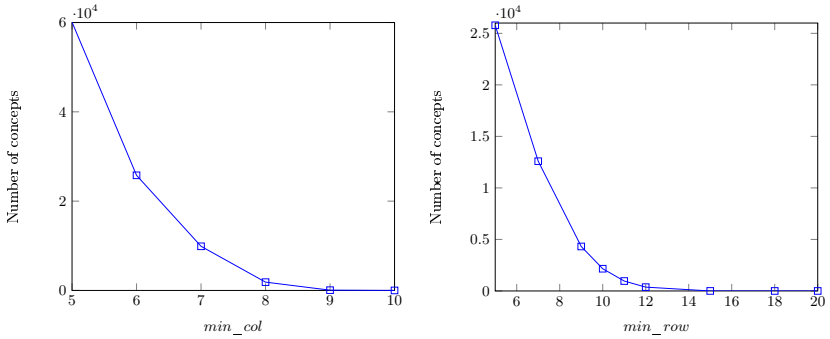

Fig. 2: Effect of min_col (with $\theta = 1$ and $min\_row = 5$) and min_row (with $\theta = 1$ and $min\_col = 6$) on a 500×60 dataset.

objects whose description subsumes Y) is performed only if the number of objects in Z (extent of the candidate concept) is at least min_row.

5 Experiments

In this section, we report some experimental results to show the scalability of IPS in the task of biclustering. By using CbO as concept miner, the space/time complexity of IPS follows CbO (see [15]). We use the synthetic datasets provided by Henriques and Madeira [9]: 500×60 and 1000×100, with hidden SC biclusters.

First, we investigate the effect of θ on the runtime and the number of concepts. The results are illustrated in Fig. 1. The left figure confirms that the larger θ generates more interval pattern concepts, and generally longer runtime as it can be seen in the right figure. The $\theta = 0.4$ requires longer runtime than $\theta = 0.5$ to 0.9. This is normal since for similar number of concepts, the probability of smaller θ obtaining a concept is smaller than the larger θ. Using CbO with smaller θ, a candidate concept will have shorter intervals in its intent, hence smaller number of objects whose description subsumes this interval.

The effect of min_col is shown in Fig. 2 left. Lesser min_col produces more concepts, and therefore longer runtime. Similarly, Fig. 2 right shows that larger min_row generates more concepts.

In the previous experiments, the CbO was terminated until all interval pattern concepts were retrieved. In the following experiment, CbO is terminated until 500 concepts are found. We compare them to BicPAM [9] that uses a discretization parameter (as a number of alphabet/items), while IPS uses the length of intervals as θ. After the mapping step (normalization, discretization, and missing values and noise handling), BicPAM applies a pattern mining method (F2G [11] as default), and the closing step (extension, merging, and filtering) is performed. Results in Table 6 show a similar performance of both methods. It should be noted that the number of biclusters from BicPAM is lower due to the merging and/or filtering.

Table 6: Comparison with BicPAM on 1000×100 dataset. For the IPS, the parameters $min\_row = 10$ and $min\_col = 5$ are used, with varying θ.

Method	Parameter	Runtime (s)	Number of biclusters
BicPAM	alphabet = 20	<15	∼100
	alphabet = 10	<15	<200
	alphabet = 7	<15	<200
	alphabet = 5	<30	∼200
IPS	$\theta = 1$	37	500
	$\theta = 2$	>500	500
	$\theta = 4$	47	500
	$\theta = 8$	39	500

Furthermore, still from Table 6, the runtime of IPS is not exactly correlated with θ (especially with $\theta = 2$), similar to our previous experiment shown in Fig. 1. Overall, with similar runtime, biclustering with IPS can return similar number of biclusters without discretization.

6 Conclusion

In this paper, we propose an alternative method of biclustering in numerical datasets. Discretization is a general preprocessing step while working with numerical values. Here we explore the possibility of working directly on numerical datasets without discretization. This can be achieved using interval pattern structures, where a bicluster can be found from any interval pattern concept. To filter the number of concepts (which can be very large) it is necessary to provide some parameters, like the length of intervals, minimum number of rows and columns, or even minimum number of biclusters. Our experiments show that these parameters can reduce the computation to a reasonable runtime. Another way to reduce the number of biclusters is to develop post-processing techniques similar to BicPAM, which include merging, filtering, and extension.

We use the CbO algorithm, a formal concept generator that can be generalized to interval pattern structures. In-Close 2 [2] in particular is faster than CbO in formal concept mining, but its efficiency in interval pattern concept mining should be studied. Another future research is to extend our FCA-based approach to other types of biclusters, e.g. coherent-evolution, coherent-sign-changes, etc. Furthermore, the existence of missing values and/or outliers should be considered in improving the proposed biclustering method.

References

1. Aggarwal, C.C., Han, J. (eds.): Frequent Pattern Mining. Springer (2014). https://doi.org/10.1007/978-3-319-07821-2, https://doi.org/10.1007/978-3-319-07821-2

2. Andrews, S.: In-Close2, a high performance formal concept miner. In: International Conference on Conceptual Structures. pp. 50–62. Springer (2011)
3. Ben-Dor, A., Chor, B., Karp, R., Yakhini, Z.: Discovering local structure in gene expression data: the order-preserving submatrix problem. Journal of computational biology **10**(3-4), 373–384 (2003)
4. Cheng, Y., Church, G.M.: Biclustering of expression data. In: ISMB. vol. 8, pp. 93–103 (2000)
5. Duarte, R.P., Simões, Á., Henriques, R., Neto, H.C.: FPGA-based OpenCL accelerator for discovering temporal patterns in gene expression data using biclustering. In: Proceedings of the 6th International Workshop on Parallelism in Bioinformatics. pp. 53–62. ACM (2018)
6. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: International Conference on Conceptual Structures. pp. 129–142. Springer (2001)
7. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, 2nd edn. (1999)
8. Henriques, R., Ferreira, F.L., Madeira, S.C.: Bicpams: software for biological data analysis with pattern-based biclustering. BMC bioinformatics **18**(1), 82 (2017)
9. Henriques, R., Madeira, S.C.: BicPAM: Pattern-based biclustering for biomedical data analysis. Algorithms for Molecular Biology **9**(1), 27 (2014)
10. Henriques, R., Madeira, S.C.: Bicspam: flexible biclustering using sequential patterns. BMC bioinformatics **15**(1), 130 (2014)
11. Henriques, R., Madeira, S.C., Antunes, C.: F2g: efficient discovery of full-patterns. ECML/PKDD nfMCP pp. 1–9 (2013)
12. 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)
13. Ignatov, D.I., Poelmans, J., Zaharchuk, V.: Recommender system based on algorithm of bicluster analysis RecBi. arXiv preprint arXiv:1202.2892 (2012)
14. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Information Sciences **181**(10), 1989–2001 (2011)
15. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence **14**(2-3), 189–216 (2002)
16. Li, G., Ma, Q., Tang, H., Paterson, A.H., Xu, Y.: QUBIC: a qualitative biclustering algorithm for analyses of gene expression data. Nucleic Acids Research **37**(15), e101–e101 (2009)
17. 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)
18. Pio, G., Ceci, M., D'Elia, D., Loglisci, C., Malerba, D.: A novel biclustering algorithm for the discovery of meaningful biological correlations between microRNAs and their target genes. BMC bioinformatics **14**(7), S8 (2013)
19. Pontes, B., Giráldez, R., Aguilar-Ruiz, J.S.: Biclustering on expression data: A review. Journal of biomedical informatics **57**, 163–180 (2015)
20. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics **18**(suppl_1), S136–S144 (2002)
21. Veroneze, R., Banerjee, A., Von Zuben, F.J.: Enumerating all maximal biclusters in numerical datasets. Information Sciences **379**, 288–309 (2017)