# Biclustering Numerical Data in Formal Concept Analysis

Mehdi Kaytoue[1], Sergei O. Kuznetsov[2], and Amedeo Napoli[1]

[1] Laboratoire Lorrain de Recherche en Informatique et ses Applications (LORIA)
Campus Scientifique, B.P. 70239 – Vandœuvre-lès-Nancy – France
kaytouem@loria.fr, napoli@loria.fr
[2] State University Higher School of Economics
Pokrovskiy Bd. 11 – 109028 Moscow – Russia
skuznetsov@hse.ru

**Abstract.** A numerical dataset is usually represented by a table where each entry denotes the value taken by an object in line for an attribute in column. A bicluster in a numerical data table is a subtable with close values different from values outside the subtable. Traditionally, largest biclusters were found by means of methods based on linear algebra. We propose an alternative approach based on concept lattices and lattices of interval pattern structures. In other words, this paper shows how formal concept analysis originally tackles the problem of biclustering and provides interesting perspectives of research.

**keywords:** biclustering, numerical data, formal concept analysis, pattern structures, conceptual scaling.

## 1 Introduction

We consider the problem of biclustering numerical data [7,4,16] using techniques of Formal Concept Analysis (FCA) [5,6]. A numerical dataset is given by sets of objects, attributes, and attribute values for objects (many-valued contexts in terms of FCA). The description of an object is a tuple of values, each component corresponding to an attribute value. An example of numerical dataset is given in Table 1 where lines denote objects, while columns denote attributes.

To analyze such a dataset, a major data-mining task is clustering, a data analysis technique used in several domains, e.g. gene expression data analysis. It allows one to group objects into clusters according to some similarity criteria between their description, the similarity being defined according to an adequate distance, following given characteristics [9]. However, clusters are *global* patterns since similarity between objects is computed w.r.t. all attributes simultaneously (possibly weighted). In many applications, and especially in gene expression data analysis, *local* patterns are preferred [3,16] and consist in pairs $(A, B)$ where $A$ is a subset of objects related to a subset of attributes $B$. Indeed, it is known that a set of genes is activated (e.g. translated into proteins for enabling a biological process) under some conditions only, i.e. only for some attributes. Accordingly, a bicluster is generally represented by a rectangle of values in a numerical data table, see e.g. a bicluster in Table 2. In Table 1, one can see that both biclusters $(\{g_1, g_2\}, \{m_1, m_2, m_3, m_4\})$ and $(\{g_1, g_2\}, \{m_5\})$ give more meaningful information than cluster $\{g_1, g_2\}$ being described by all attributes, since the values taken by objects in $A$ for attributes in $B$ are more similar.

There are many definitions of a bicluster, depending on the relation between subsets of objects and subsets of attributes, as discussed in [16]. In this paper, we consider two types of biclusters: firstly, constant biclusters that can be represented as rectangle of equal values (see Table 3), and secondly, biclusters of similar values, that can be represented by rectangle of similar values (see Table 4). In general case, extracting all biclusters is an intractable problem [16], so in practice heuristics are used. Obviously, even best heuristics may result in the loss of “interesting” biclusters.

The purpose of this paper is to show that an approach based on Formal Concept Analysis (FCA [5]) can be used for biclustering numerical data, leading to a complete, correct and non-redundant enumeration of all maximal biclusters (either of constant or similar values). Such non-heuristic based enumeration has not been deeply considered in the literature due to the very important number of possible biclusters. Whereas a first study is given in [2], we propose here two equivalent FCA-based methods, whose underlying closure operator enables a natural enumeration of maximal biclusters. The first one relies on conceptual scaling (discretization) of numerical data giving rise to several binary tables from which biclusters can be extracted as formal concepts. A second method avoids *a priori* scaling and is based on interval pattern structures [6,12], an FCA formalism that allows one to build concept lattices directly from numerical data from which biclusters of interest can be extracted.

The paper is organized as follows. We first give a brief introduction to FCA, before formally stating the problem of extracting biclusters from numerical data. Then, Section 2 presents the first method based on scaling while Section 3 details the method based on pattern structures. Finally, a discussion compares both approaches w.r.t. their scalability and usage, and highlights several perspectives of research.

### 1.1 Preliminaries on FCA

We use standard notations of [5]. Let $G$ and $M$ be arbitrary sets and $I \subseteq G \times M$ be an arbitrary binary relation between $G$ and $M$. The triple $(G, M, I)$ is called a formal context. Each $g \in G$ is interpreted as an object, each $m \in M$ is interpreted as an attribute. The fact $(g, m) \in I$ is interpreted as “$g$ has attribute $m$”. The two following derivation operators $(\cdot)'$ are considered:

$$\begin{aligned} A' &= \{m \in M \mid \forall g \in A : gIm\} \qquad && \textit{for } A \subseteq G, \\ B' &= \{g \in G \mid \forall m \in B : gIm\} \qquad && \textit{for } B \subseteq M \end{aligned}$$

which define a Galois connection between the powersets of $G$ and $M$. For $A \subseteq G$, $B \subseteq M$, a pair $(A, B)$ such that $A' = B$ and $B' = A$, is called a *(formal) concept.*

Concepts are partially ordered by $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ ($\Leftrightarrow B_2 \subseteq B_1$). With respect to this partial order, the set of all formal concepts forms a complete lattice called the *concept lattice* of the formal context $(G, M, I)$. For a concept $(A, B)$ the set $A$ is called the *extent* and the set $B$ the *intent* of the concept. Certain data are not given directly by binary relations, e.g. numerical data. Such data is usually represented by a many-valued context $(G, M, W, I)$, a 4-tuple constituted of a set of objects $G$, a set of attributes $M$, a set of attribute values $W$ and a ternary relation $I$ defined on the Cartesian product $G \times M \times W$. $(g, m, w) \in I$, also written $g(m) = w$, means that "the value of attribute $m$ taken by object $g$ is $w$". The relation $I$ verifies that $g(m) = w$ and $g(m) = v$ always implies $w = v$. For applying the FCA machinery, a many-valued context needs to be transformed into a formal context with so-called conceptual scaling. The choice of a scale should be wisely done w.r.t. data and goals since affecting the size, the interpretation, and the computation of the resulting concept lattice.

## 1.2 Problem Setting

Here a numerical dataset is realized by a many-valued context $(G, M, W, I)$ where $W$ is a set of values that objects $g \in G$ can take for attributes $m \in M$. Such many-valued contexts are usually represented by a numerical table where a table-entry gives the value $m(g) \in W$, i.e. the value taken by attribute $m$ in column for object $g$ in line. The Table 1 gives an example (taken from [2]) that we consider throughout this paper, with objects $G = \{g_1, ..., g_4\}$, attributes $M = \{m_1, ..., m_5\}$, and e.g. $m_2(g_4) = 9$.

A bicluster is given by a pair $(A, B)$ with $A \subseteq G$ and $B \subseteq M$. Intuitively, a bicluster is represented by a rectangle of values, or sub-table (modulo line and column permutations), see e.g. the bicluster $(\{g_2, g_3, g_4\}, \{m_3, m_4\})$ highlighted grey in Table 2.

**Definition 1 (Bicluster).** *Given a numerical dataset $(G, M, W, I)$, a bicluster is a pair $(A, B)$ with $A \subseteq G$ and $B \subseteq M$.*

In [16], several types of biclusters are introduced. The type of a bicluster $(A, B)$ depends on the relation between the values taken by attributes in $B$ for objects in $A$. In this paper, we consider constant biclusters (equality relation) and biclusters of similar values (similarity relation) as defined in the next paragraphs.

A constant bicluster can be interpreted as a rectangle of identical values, and is defined as follows.

**Table 1.** A numerical dataset

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ |
|---|---|---|---|---|---|
| $g_1$ | 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 |

**Table 2.** $(\{g_2, g_3, g_4\}, \{m_3, m_4\})$

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ |
|---|---|---|---|---|---|
| $g_1$ | 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 |

**Table 3.** A constant bicluster

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ |
|---|---|---|---|---|---|
| $g_1$ | 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 |

**Table 4.** A bicluster of similar values

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ |
|---|---|---|---|---|---|
| $g_1$ | **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 |

**Definition 2 (Constant bicluster).** *Given a numerical dataset* $(G, M, W, I)$*, a constant bicluster is a bicluster* $(A, B)$ *such that* $m_i(g_j) = m_k(g_l), \forall g_j, g_l \in A, \forall m_i, m_k \in B$*.*

Since the number of possible biclusters in a numerical dataset can be very large, the notion of maximality gives naturally rise to maximal constant biclusters, i.e. “largest rectangles of identical values”.

**Definition 3 (Maximal constant biclusters).** *Given a numerical dataset* $(G, M, W, I)$*, a constant bicluster* $(A, B)$ *is maximal if it does not exist a constant bicluster* $(E, F)$ *with either* $A \subset E$ *or* $B \subset F$*.*
*In other terms,* $(A, B)$ *is a maximal constant bicluster iff*

– $(A \cup \{g\}, B)$ *is not a constant bicluster* $\forall g \in G \backslash A$
– $(A, B \cup \{m\})$ *is not a constant bicluster* $\forall m \in M \backslash B$

Table 3 shows an example of maximal constant bicluster $(\{g_1, g_2, g_3\}, \{m_5\})$. One should remark that $(\{g_1, g_2\}, \{m_5\})$ is constant but not maximal. Note that maximal constant biclusters taking values 1 in a 1/0 table are formal concepts.

The fact that constant biclusters correspond to sets of objects taking equal values for same attributes is a too strong condition in real-world data. This may lead to the well-known problem of pattern overwhelming. Instead of considering equality, one may relax this condition and consider a similarity relation between values. This idea was introduced in [2] for handling noise in a numerical dataset. Two values $w_1, w_2 \in W$ are said to be similar if their difference does not exceed a user-defined parameter $\theta$. A similarity relation denoted by $\simeq_\theta$ is formally defined by: $w_1 \simeq_\theta w_2 \iff |w_1 - w_2| \leq \theta$. According to this formalization of similarity, a bicluster of similar values can be defined as a “generalization” of constant biclusters.

**Definition 4 (Bicluster of similar values).** *A bicluster* $(A, B)$ *is a bicluster of similar values if* $m_i(g_j) \simeq_\theta m_k(g_l), \forall g_j, g_l \in A, \forall m_i, m_k \in B$*.*

**Definition 5 (Maximal biclusters of similar values).** *A bicluster of similar values* $(A, B)$ *is maximal if there does not exist a bicluster of similar values* $(E, F)$ *with either* $A \subset E$ *or* $B \subset F$*.*

Table 4 shows an example of maximal bicluster of similar values $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ with $\theta = 1$. Note that bicluster $(\{g_1, g_2\}, \{m_1, m_2\})$ fulfils the conditions of similarity but is not maximal. Obviously, constant biclusters are biclusters of similar values when $\theta = 0$.

In this paper we consider the problem of mining all maximal (i) constant biclusters and (ii) biclusters of similar values from a numerical dataset. The novelty here lies in the use of Formal Concept Analysis for a correct, complete and non-redundant enumeration (without heuristics). Indeed, we show in the following sections how to define a scaling to build formal contexts whose concepts exactly correspond to the two types of biclusters. However, this leads to the definition of several contexts whose preparation and mining may be inefficient. Then, based on so-called interval pattern structures, we show how binarization can be avoided, which results in reducing practical computational complexity.

## 2 Mining Biclusters by Means of Conceptual Scaling

In this section, we present two scaling procedures allowing to build formal contexts from which (i) constant biclusters and (ii) biclusters of similar values, can be extracted within the existing FCA framework. Intuitively, scaling allows to express bicluster searchspace under the form of binary tables, while the Galois connection allows to extract maximal biclusters represented as concepts.

### 2.1 Constant Biclusters

A maximal constant bicluster can be interpreted as a maximal rectangle of identical values. Recall that formal concepts correspond to maximal rectangles of 1 values in binary tables. Accordingly, a maximal constant bicluster containing values $w \in W$ from a numerical dataset $(G, M, W, I)$ corresponds to a concept in a context $\mathbb{K}_w = (G, M, I_w)$ where $(g, m) \in I_w \iff m(g) = w$. One should naturally consider one formal context for each value $w \in W$, which results in a context family $\mathbb{K}_W$ defined as follows:

$$\mathbb{K}_W = \{\mathbb{K}_w = (G, M, I_w) \mid w \in W \ (m, g) \in I_w \iff m(g) = w\}$$

The procedure building the family $\mathbb{K}_W$ from $(G, M, W, I)$ involves one conceptual scaling for each $w \in W$ (actually nominal scalings related to each value $w$ [5]). Figure 1 gives $\mathbb{K}_w = (G, M, I_w)$ for $w = 1$ and $w = 6$. The collection of concepts of each context $\mathbb{K}_w = (G, M, I_w)$ is denoted by $\mathfrak{B}(G, M, I_w)$, or simply $\mathfrak{B}_w$. Examples are given in Figure 1.

The two obvious propositions hold.

**Proposition 1.** *Given a set of objects $A \subseteq G$ and a set of attributes $B \subseteq M$, a concept $(A, B)$ of $\mathbb{K}_w$ corresponds to a maximal constant bicluster $(A, B)$ of values $w$ from numerical dataset $(G, M, W, I)$.*

**Proposition 2.** *There is a one-to-one correspondence between the set of concepts $\bigcup_{w \in W} \mathfrak{B}_w$ and the set of all maximal biclusters.*

| $w \in W$ | $\mathbb{K}_w$ | $\mathfrak{B}_w$ | Bicluster corresponding to first concept on left list |
|---|---|---|---|
| ... | ... | ... | ... |
| 1 | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ g_1 & \times & & & \times & \\ g_2 & & \times & \times & & \\ g_3 & & & \times & & \\ g_4 & & & & & \end{array}$ | $(\{g_2, g_3\}, \{m_3\})$ $(\{g_2\}, \{m_2, m_3\})$ $(\{g_1\}, \{m_1, m_4\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ g_1 & 1 & 2 & 2 & 1 & 6 \\ g_2 & 2 & 1 & \mathbf{1} & 0 & 6 \\ g_3 & 2 & 2 & \mathbf{1} & 7 & 6 \\ g_4 & 8 & 9 & 2 & 6 & 7 \end{array}$ |
| ... | ... | ... | ... |
| 6 | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ g_1 & & & & & \times \\ g_2 & & & & & \times \\ g_3 & & & & & \times \\ g_4 & & & & \times & \end{array}$ | $(\{g_1, g_2, g_3\}, \{m_5\})$ $(\{g_4\}, \{m_4\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ g_1 & 1 & 2 & 2 & 1 & \mathbf{6} \\ g_2 & 2 & 1 & 1 & 0 & \mathbf{6} \\ g_3 & 2 & 2 & 1 & 7 & \mathbf{6} \\ g_4 & 8 & 9 & 2 & 6 & 7 \end{array}$ |
| ... | ... | ... | ... |

**Fig. 1.** Extracting constant biclusters from the dataset of Table 1

Hence, an algorithm that constructs the set of concepts $\bigcup_{w \in W} \mathfrak{B}_w$ gives a correct, complete and non redundant enumeration of all maximal constant biclusters.

Figure 1 gives two examples of concepts and their corresponding bicluster representation in the original numerical table.

## 2.2 Biclusters of Similar Values

The number of constant biclusters can be very large in real-world data, where numerical attribute domains contain many different values. Moreover, it leads to a huge number of artifacts, e.g. the maximal constant bicluster $(A, B) = (\{g_4\}, \{m_4\})$ is a rectangle of area 1, i.e. the product $|A| \times |B|$. One should therefore relax the equality constraint on numerical values when performing scaling with similarity relation $\simeq_\theta$ defined in the introduction. Intuitively, with $\theta = 1$, the previous example is not maximal anymore, whereas $(\{g_3, g_4\}, \{m_4, m_5\})$ is maximal with area equal to 4. For that matter, one should extract rectangles with pairwise similar values w.r.t $\simeq_\theta$. However, this relation is reflexive and symmetric but not transitive, hence a *tolerance relation.*

As related in [14], a tolerance relation $T$ over an arbitrary set $G$, i.e. $T \subseteq G \times G$, can be represented by a formal context $(G, G, T)$. A formal concept of $(G, G, T)$

**Table 5.** Formal context of relation $\simeq_\theta$ over $W = \{0, 1, 2, 5, 6, 7, 8, 9\}$ with $\theta = 1$ (left). Corresponding tolerance classes (middle). Renaming classes as the convex hull of their elements (right).

| $\simeq_1$ | 0 | 1 | 2 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|
| 0 | × | × | | | | | |
| 1 | × | × | × | | | | |
| 2 | | × | × | | | | |
| 6 | | | | × | × | | |
| 7 | | | | × | × | × | |
| 8 | | | | | × | × | × |
| 9 | | | | | | × | × |

| Classes of tolerance |
|---|
| $\{0, 1\}$ |
| $\{1, 2\}$ |
| $\{6, 7\}$ |
| $\{7, 8\}$ |
| $\{8, 9\}$ |

| Renamed classes |
|---|
| $[0, 1]$ |
| $[1, 2]$ |
| $[6, 7]$ |
| $[7, 8]$ |
| $[8, 9]$ |

where intent is equal to extent corresponds to a *class of tolerance*, i.e., a maximal subset of $G$ such that all pairs of its elements are in relation $T$.

Going back to the tolerance relation $\simeq_\theta$ on a set of values $W$, tolerance classes are maximal sets of pairwise similar values, corresponding to concepts $(A, B)$ of $(W, W, \simeq_\theta)$ such that $A = B$ [11]. This is exactly what we need to characterize maximal biclusters of similar values. More details on this process are given in [11], while Table 5 shows initial context $(W, W, \simeq_\theta)$ and corresponding classes of tolerance from the numerical dataset of Table 1.

Now that classes of tolerance, or maximal sets of pairwise similar values, are characterized and computed, we can rename them for sake of readability and use them for scaling the initial dataset from which maximal biclusters of similar values can be extracted.

We choose to rename a class $K \subseteq W$ as the convex hull of its elements, i.e. the interval $[k_i, k_j]$ s.t. $k_i$ and $k_j$ are respectively smallest and largest values of $K$ w.r.t. natural order $\leq$ on numbers. Indeed, when $|K|$ becomes large for certain data, this new name is more concise. Moreover, any $k \in [k_i, k_j]$ respects $k \simeq_\theta k_i \simeq_\theta k_j$.

Biclusters of similar values are a generalization of constant ones, i.e. with all values included in interval $[k_i, k_j]$ for a given class of tolerance. We should now also consider one formal context for each class of tolerance, hence a family of contexts. Consider a numerical dataset $(G, M, W, I)$, and a class of tolerance from $W$ which corresponds to the interval $[k_i, k_j]$. The associated formal context is given by:

$$\begin{aligned}(G, M, I_{[k_i,k_j]})\ s.t.\ (g, m) \in I \Leftrightarrow m(g) \in [k_i, k_j]\ and \\ (\exists h_1, h_2 \in m'\ s.t.\ m(h_1) = k_i\ and\ m(h_2) = k_j \\ or\ \exists n_1, n_2 \in g'\ s.t.\ n_1(g) = k_i\ and\ n_2(g) = k_j)\end{aligned}$$

First condition $m(g) \in [k_i, k_j]$ means that $m(g)$ should be similar with all elements of the current class of tolerance. The two other conditions come from the fact that classes of tolerance are computed from the set $W$: since a bicluster is represented by a rectangle in the numerical table, we should consider only

| Class of tolerance | Formal context[a] | Concepts | Bicluster corresponding to first concept on left list |
|---|---|---|---|
| $[0,1]$ | $\begin{array}{cccc} & m_2 & m_3 & m_4 \\ \hline g_1 & & & \times \\ g_2 & \times & \times & \times \end{array}$ | $(\{g_1,g_2\},\{m_4\})$ $(\{g_2\},\{m_2,m_3,m_4\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ \hline g_1 & 1 & 2 & 2 & \mathbf{1} & 6 \\ g_2 & 2 & 1 & 1 & \mathbf{0} & 6 \\ g_3 & 2 & 2 & 1 & 7 & 6 \\ g_4 & 8 & 9 & 2 & 6 & 7 \end{array}$ |
| $[1,2]$ | $\begin{array}{ccccc} & m_1 & m_2 & m_3 & m_4 \\ \hline g_1 & \times & \times & \times & \times \\ g_2 & \times & \times & \times & \\ g_3 & \times & \times & \times & \\ g_4 & & & \times & \end{array}$ | $(\{g_1,g_2,g_3\},\{m_1,m_2,m_3\})$ $(\{g_1\},\{m_1,m_2,m_3,m_4\})$ $(\{g_1,g_2,g_3,g_4\},\{m_3\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ \hline g_1 & \mathbf{1} & \mathbf{2} & \mathbf{2} & 1 & 6 \\ g_2 & \mathbf{2} & \mathbf{1} & \mathbf{1} & 0 & 6 \\ g_3 & \mathbf{2} & \mathbf{2} & \mathbf{1} & 7 & 6 \\ g_4 & 8 & 9 & 2 & 6 & 7 \end{array}$ |
| $[6,7]$ | $\begin{array}{ccc} & m_4 & m_5 \\ \hline g_1 & & \times \\ g_2 & & \times \\ g_3 & \times & \times \\ g_4 & \times & \times \end{array}$ | $(\{g_3,g_4\},\{m_4,m_5\})$ $(\{g_1,g_2,g_3,g_4\},\{m_5\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ \hline g_1 & 1 & 2 & 2 & 1 & 6 \\ g_2 & 2 & 1 & 1 & 0 & 6 \\ g_3 & 2 & 2 & 1 & \mathbf{7} & \mathbf{6} \\ g_4 & 8 & 9 & 2 & \mathbf{6} & \mathbf{7} \end{array}$ |
| $[7,8]$ | $\begin{array}{ccc} & m_1 & m_5 \\ \hline g_4 & \times & \times \end{array}$ | $(\{g_4\},\{m_1,m_5\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ \hline g_1 & 1 & 2 & 2 & 1 & 6 \\ g_2 & 2 & 1 & 1 & 0 & 6 \\ g_3 & 2 & 2 & 1 & 7 & 6 \\ g_4 & \mathbf{8} & 9 & 2 & 6 & \mathbf{7} \end{array}$ |
| $[8,9]$ | $\begin{array}{ccc} & m_1 & m_2 \\ \hline g_4 & \times & \times \end{array}$ | $(\{g_4\},\{m_1,m_2\})$ | $\begin{array}{cccccc} & m_1 & m_2 & m_3 & m_4 & m_5 \\ \hline g_1 & 1 & 2 & 2 & 1 & 6 \\ g_2 & 2 & 1 & 1 & 0 & 6 \\ g_3 & 2 & 2 & 1 & 7 & 6 \\ g_4 & \mathbf{8} & \mathbf{9} & 2 & 6 & 7 \end{array}$ |

[a] Empty lines and columns are omitted.

**Fig. 2.** Extracting all maximal biclusters of similar values from Table 1

similar values in column (second condition) or dually lines (third condition) to test whether a value belongs to a class of tolerance.

Consider the formal context $\mathbb{K}_{[k_i,k_j]}$ which corresponds to the class of tolerance $[k_i,k_j]$ and a concept $(A,B)$ from this context. The following propositions hold.

**Proposition 3.** *$(A,B)$ is a maximal bicluster of similar values.*

**Proposition 4.** *There is a one-to-one correspondence between the set of concepts from all formal contexts* $\mathbb{K}_{[k_i,k_j]}$ *and the set of all maximal biclusters of similar values.*

Thus, an algorithm computing the set of concepts from all formal contexts $\mathbb{K}_{[k_i,k_j]}$ gives a correct, complete and non redundant enumeration of maximal biclusters of similar values.

Figure 2 gives the formal context $\mathbb{K}_{[k_i,k_j]}$ for each class of tolerance $[k_i, k_j]$, their respective concepts and bicluster representation in the initial numerical Table 1.

## 3 Mining Biclusters from Pattern Concept Lattice

Until now, we presented how (constant) biclusters (of similar) values can be extracted using standard FCA tools such as scaling and concept extraction algorithms. Since resulting binary tables may be numerous and large (i.e. one for each class of tolerance), we present in this section an approach based on pattern structures. Pattern structures are introduced in [6] and can be thought as a "generalization" of formal contexts to complex data from which a concept lattice can be built without *a priori* scaling. We consider in this section only biclusters of similar values, since being more general than constant ones and more useful for real-world applications.

### 3.1 Pattern Structures

Formally, let $G$ be a set (interpreted as a set of objects), let $(D, \sqcap)$ be a meet-semilattice (of potential object descriptions) and let $\delta : G \longrightarrow D$ be a mapping. Then $(G, \underline{D}, \delta)$ with $\underline{D} = (D, \sqcap)$ is called a *pattern structure*, and the set $\delta(G) := \{\delta(g) \mid g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$, of $(D, \sqcap)$. Thus each $X \subseteq \delta(G)$ has an infimum $\sqcap X$ in $(D, \sqcap)$ and $(D_\delta, \sqcap)$ is the set of these infima. Each $(D_\delta, \sqcap)$ has both lower and upper bounds, resp. 0 and 1. Elements of $D$ are called *patterns* and are ordered by subsumption relation $\sqsubseteq$: given $c, d \in D$ one has $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$.
A pattern structure $(G, \underline{D}, \delta)$ gives rise to the following derivation operators $(\cdot)^\square$:

$$A^\square = \bigsqcap_{g \in A} \delta(g) \qquad for\ A \subseteq G,$$

$$d^\square = \{g \in G | d \sqsubseteq \delta(g)\} \qquad for\ d \in D.$$

These operators form a Galois connection between the powerset of $G$ and $(D, \sqsubseteq)$. *Pattern concepts* of $(G, \underline{D}, \delta)$ are pairs of the form $(A, d)$, $A \subseteq G$, $d \in D$, such that $A^\square = d$ and $A = d^\square$. For a pattern concept $(A, d)$ the component $d$ is called a *pattern intent* and is a description of all objects in $A$, called *pattern extent.* Intuitively, $(A, d)$ is a pattern concept if adding any element to $A$ changes $d$ through $(\cdot)^\square$ operator and equivalently taking $e \sqsupset d$ changes $A$. Like in case of formal contexts, for a pattern structure $(G, \underline{D}, \delta)$ a pattern $d \in D$ is called *closed* if $d^{\square\square} = d$ and a set of objects $A \subseteq G$ is called *closed* if $A^{\square\square} = A$. Obviously, pattern extents and intents are closed.

## 3.2 Interval Pattern Structures

In [12], a numerical dataset $(G, M, W, I)$ is represented by a so-called interval pattern structure $(G, (D, \sqcap), \delta)$ where $D$ is a set of interval vectors, the $i^{th}$ dimension giving an interval of values from $W$ for attribute $m_i \in M$. We denote such vectors as *interval patterns.* In Table 1, the description of object $g_1$ is the interval pattern $\delta(g_1) = \langle [1,1], [2,2], [2,2], [1,1], [6,6] \rangle$. Interval patterns can be represented as $|M|$-hyperrectangles in Euclidean space $\mathbb{R}^{|M|}$, whose sides are parallel to the coordinate axes.

Now we detail how interval patterns are ordered. Consider firstly a single attribute $m \in M$, with value domain $W_m \subseteq W$. Elements of $W_m$ can be ordered within a meet-semi-lattice making them potential object descriptions. Recalling that any $w \in W_m$ can be written as interval $[w, w]$, the infimum $\sqcap$ of two intervals $[a_1, b_1]$ and $[a_2, b_2]$, with $a_1, b_1, a_2, b_2 \in \mathbb{R}$ is: $[a_1, b_1] \sqcap [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$, i.e. the largest interval containing them. Indeed, when $c$ and $d$ are intervals, $c \sqsubseteq d \Leftrightarrow c \sqcap d = c$ holds:

$$\begin{aligned}
[a_1, b_1] \sqsubseteq [a_2, b_2] \Leftrightarrow & \quad [a_1, b_1] \sqcap [a_2, b_2] = [a_1, b_1] \\
\Leftrightarrow & \quad [min(a_1, a_2), max(b_1, b_2)] = [a_1, b_1] \\
\Leftrightarrow & \quad a_1 \leq a_2 \text{ and } b_1 \geq b_2 \\
\Leftrightarrow & \quad [a_1, b_1] \supseteq [a_2, b_2].
\end{aligned}$$

As objects are described by several intervals, each one standing for a given attribute, *interval patterns* have been introduced as $p$-dimensional vector of intervals, with $p = |M|$. Given two interval patterns $e = \langle [a_i, b_i] \rangle_{i \in [1,p]}$ and $f = \langle [c_i, d_i] \rangle_{i \in [1,p]}$ their infimum $\sqcap$ and induced ordering relation $\sqsubseteq$ are given by:

$$\begin{aligned}
e \sqcap f &= \langle [a_i, b_i] \rangle_{i \in [1,p]} \sqcap \langle [c_i, d_i] \rangle_{i \in [1,p]} \\
&= \langle [a_i, b_i] \sqcap [c_i, d_i] \rangle_{i \in [1,p]}
\end{aligned}
\qquad
\begin{aligned}
e \sqsubseteq f &\Leftrightarrow \langle [a_i, b_i] \rangle_{i \in [1,p]} \sqsubseteq \langle [c_i, d_i] \rangle_{i \in [1,p]} \\
&\Leftrightarrow [a_i, b_i] \sqsubseteq [c_i, d_i], \ \forall i \in [1,p]
\end{aligned}$$

This means that patterns with larger intervals are subsumed by patterns with smaller ones. Hence, one can define a pattern structure $(G, (D, \sqcap), \delta)$ from a numerical dataset $(G, M, W, I)$, where $(D, \sqcap)$ is a meet-semi-lattice of interval patterns. This is deeply detailed in [12]. We illustrate here the Galois connection.

$$\begin{aligned}
\{g_2, g_3\}^{\square} &= \delta(g_2) \sqcap \delta(g_3) \\
&= \langle [2,2], [1,2], [1,1], [0,7], [6,6] \rangle
\end{aligned}$$

$$\begin{aligned}
\langle [2,2], [1,2], [1,1], [0,7], [6,6] \rangle^{\square} &= \{g \in G | \langle [2,2], [1,2], [1,1], [0,7], [6,6] \rangle \sqsubseteq \delta(g)\} \\
&= \{g_2, g_3\}
\end{aligned}$$

Hence $(\{g_2, g_3\}, \langle [2,2], [1,2], [1,1], [0,7], [6,6] \rangle)$ is a pattern concept. The set of all pattern concepts gives rise to a pattern concept lattice, see Figure 3 for our example. Intuitively, $(A_1, d_1) \leq (A_2, d_2)$ means that corresponding hyperrectangle of $(A_1, d_1)$ is included in corresponding hyperrectangle of $(A_2, d_2)$.

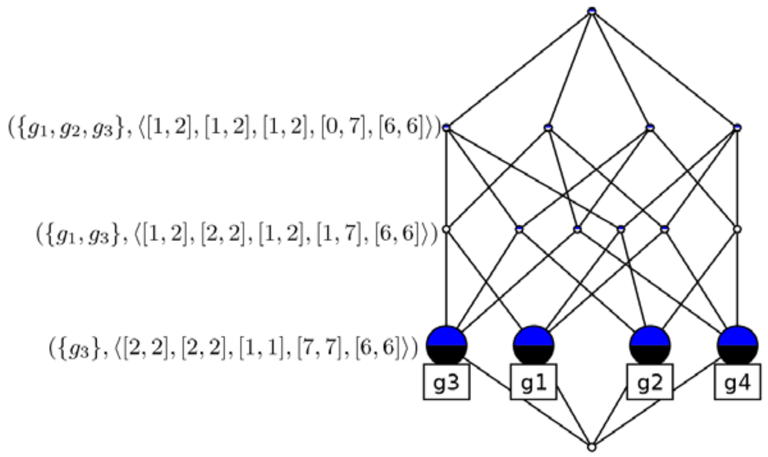
$(\{g_1, g_2, g_3\}, \langle[1,2],[1,2],[1,2],[0,7],[6,6]\rangle)$
$(\{g_1, g_3\}, \langle[1,2],[2,2],[1,2],[1,7],[6,6]\rangle)$
$(\{g_3\}, \langle[2,2],[2,2],[1,1],[7,7],[6,6]\rangle)$
g3
g1
g2
g4

**Fig. 3.** Pattern concept lattice of pattern structure from Table 1. 3 concepts are fully described with respective pattern extent and intent.

## 3.3 Biclusters of Similar Values in Pattern Concepts

A pattern concept $(A, d)$ of a numerical dataset $(G, W, M, I)$ can be seen as a bicluster $(A, M)$ since it gives a range of value for each attribute $m \in M$. Bicluster representation of $(\{g_2, g_3\}, \langle[2,2],[1,2],[1,1],[0,7],[6,6]\rangle)$ is given in Table 6.

However, a pattern concept $(A, d)$ is not necessarily a bicluster of similar values, for three reasons. First, $d$ may contain intervals larger than $\theta$, i.e. all values in columns are not necessarily similar. Secondly, $d$ may contain different intervals whose values are not similar, i.e. all values in lines may not be similar. Finally, if those conditions are respected, it is not sure that maximality of biclusters holds. We show how to control these statements to extract maximal biclusters of similar values from the pattern concept lattice.

**First statement.** Avoiding intervals of size larger than $\theta$ in a pattern intent $d$ means that a pattern concept will correspond to a rectangle for which each column has similar values. For that matter, consider a modification $(G, (D*, \sqcap), \delta)$ of the interval pattern structure defined in the previous subsection: the set $D*$ consists of tuples, whose components are either intervals or the null element $*$.

**Table 6.** Interval pattern as bicluster

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ |
|---|---|---|---|---|---|
| $g_1$ | 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 |

**Table 7.** Introducing $\theta = 1$

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ |
|---|---|---|---|---|---|
| $g_1$ | 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 |

For two intervals $[a_1, b_1]$ and $[a_2, b_2]$, with $a_1, b_1, a_2, b_2 \in \mathbb{R}$ their infimum $\sqcap$ is defined as follows: $[a_1, b_1] \sqcap [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$ if $|max(b_1, b_2) - min(a_1, a_2)| \leq \theta$ and $*$ otherwise. Moreover, $* \sqcap [a, b] = *$ for any $a, b \in \mathbb{R}$. Consider that for $d \in D$, $d_m$ denotes the interval given for attribute $m \in M$. Now, given two interval vectors $c = \langle c_i \rangle$ and $d = \langle d_i \rangle$ their infimum is computed componentwise: $c \sqcap d = \langle c_i \sqcap d_i \rangle$. Applying operators of the Galois connection on set $\{g_2, g_3\}$ derives the concept $(\{g_2, g_3\}, \langle [2, 2], [1, 2], [1, 1], *, [6, 6] \rangle)$, while starting with set $\{g_1, g_4\}$ allows to derive concept $(\{g_1, g_2, g_3, g_4\}, \langle *, *, [1, 2], *, [6, 6] \rangle)$. The resulting pattern concept lattice is given in Figure 4 and contains only 11 concepts compared to 16 when the operation $\sqcap$ is not constrained with $\theta$. Table 7 shows the bicluster representation of $(\{g_2, g_3\}, \langle [2, 2], [1, 2], [1, 1], *, [6, 6] \rangle)$, i.e. a rectangle for which values in each column are similar w.r.t. $\theta = 1$. Note that one should ignore attributes that take the value $*$ in pattern intent.

**Second statement.** From a pattern structure $(G, (D*, \sqcap), \delta)$, we are able to build a pattern concept lattice whose concepts corresponds to rectangles having similar values in columns. We should therefore also consider similar values in lines. Going back to concept $(\{g_2, g_3\}, \langle [2, 2], [1, 2], [1, 1], *, [6, 6] \rangle)$, we remark that $(\{g_2, g_3\}, \{m_1, m_2, m_3\})$ and $(\{g_2, g_3\}, \{m_5\})$ are biclusters of similar values that can be built from the initial pattern concept. Indeed, the intervals describing attributes $m_1$, $m_2$, and $m_3$ and pairwise similar ($[2, 2] \simeq_\theta [1, 2] \simeq_\theta [2, 2]$ with $\theta = 1$), while interval describing attribute $m_5$ is similar with no others. We should accordingly consider classes of tolerance between attribute descriptions to extract biclusters of similar values. The similarity relation $\simeq_\theta$ is adapted for intervals as follows: $[a_1, b_1] \simeq_\theta [a_2, b_2] \iff max(b_1, b_2) - min(a_1, a_2) \leq \theta$.

**Proposition 5.** *Given a pattern concept $(A, d)$, any pair $(A, B)$ with $B \subseteq M$ is a bicluster of similar values iff $\{d_m\}_{\forall m \in B}$ is a class of tolerance w.r.t. relation $\simeq_\theta$ over the set $\{d_m\}_{\forall m \in M}$.*

*Proof.* Consider that $(A, B)$ is not a bicluster of similar values: $\exists g_1, g_2 \in A$, and $\exists m_1, m_2 \in B$ such that $m_1(g_1) \not\simeq_\theta m_2(g_2)$, a contradiction.

**Third statement.** By controlling the two first statements, we are able to extract biclusters of similar values from the pattern concept lattice of $(G, (D*, \sqcap), \delta)$. By the properties of classes of tolerance making a class a maximal set of similar values, we know that biclusters are maximal in colums, i.e. no columns can be added without violating the similarity relation. However, we are not sure that biclusters are maximal in lines. Going back to previous example, i.e. $(\{g_2, g_3\}, \langle [2, 2], [1, 2], [1, 1], *, [6, 6] \rangle)$, the extracted biclusters $(\{g_2, g_3\}, \{m_1, m_2, m_3\})$ and $(\{g_2, g_3\}, \{m_5\})$ are not maximal. Indeed, we have $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ and $(\{g_1, g_2, g_3\}, \{m_5\})$ that are also biclusters of similar values. If such biclusters are not maximal, this means that objects can be added in the extent $A$ while $B$ remains the same set. Due to the generalization/specialization property of concept lattices, such larger bicluster can be found in the direct upper neighbours of concept $(\{g_2, g_3\}, \langle [2, 2], [1, 2], [1, 1], *, [6, 6] \rangle)$, i.e. concept $(\{g_1, g_2, g_3\}, \langle [1, 2], [1, 2], [1, 2], *, [6, 6] \rangle)$.

**Example.** The Figure 4 gives the pattern concept lattice of $(G, (D*, \sqcap), \delta)$ with $\theta = 1$. For each pattern intent, elements of each class of tolerance are either underlined, crossed-off, or in bold. For a pattern concept $(A, d)$, when a class is underlined, or in bold, it means that $(A, B)$, $B$ being the set of attribute corresponding to this class, is a maximal bicluster of similar values. If element of the class are crossed-off, this means that $(A, B)$ is not maximal, i.e $(C, B)$ with $A \subset C$ can be characterized also in a direct upper concept. For example, take concept $(\{g_1, g_2\}, \langle [1, 2], [1, 2], [1, 2], [0, 1], [6, 6] \rangle)$. From this concept, according to classes of tolerance, one can characterize the following biclusters of similar values $(\{g_1, g_2\}, \{m_1, m_2, m_3\})$, $(\{g_1, g_2\}, \{m_4\})$ and $(\{g_1, g_2\}, \{m_5\})$. However, $(\{g_1, g_2\}, \{m_4\})$ is the one only that is maximal, i.e. that cannot be characterized from upper pattern concepts with larger extents.

Hence, all biclusters of similar values can be computed from pattern concepts by standard algorithms. These considerations lead to two dual ways of constructing maximal biclusters of similar values as pattern concepts: bottom-up and top-down.

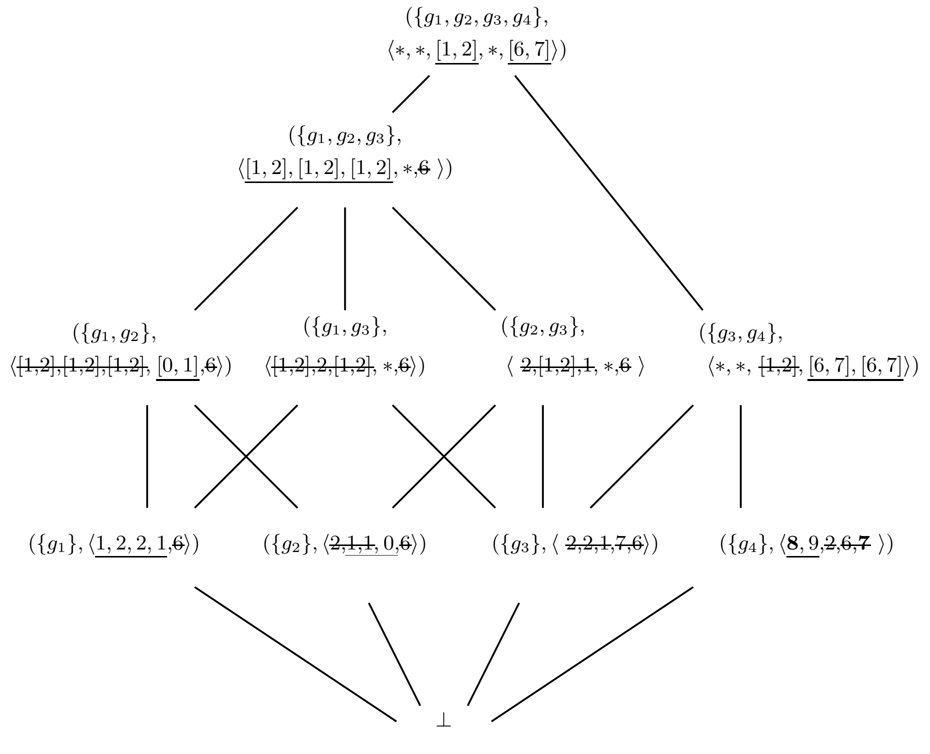
$(\{g_1, g_2, g_3, g_4\}, \langle *, *, \underline{[1, 2]}, *, \underline{[6, 7]} \rangle)$
$(\{g_1, g_2, g_3\}, \langle \underline{[1, 2], [1, 2], [1, 2]}, *, \cancel{6} \rangle)$
$(\{g_1, g_2\}, \langle \cancel{[1,2],[1,2],[1,2]}, \underline{[0, 1]}, \cancel{6} \rangle)$
$(\{g_1, g_3\}, \langle \cancel{[1,2],2,[1,2]}, *, \cancel{6} \rangle)$
$(\{g_2, g_3\}, \langle \cancel{2,[1,2],1}, *, \cancel{6} \rangle$
$(\{g_3, g_4\}, \langle *, *, \cancel{[1,2]}, \underline{[6, 7], [6, 7]} \rangle)$
$(\{g_1\}, \langle \underline{1, 2, 2, 1}, \cancel{6} \rangle)$
$(\{g_2\}, \langle \cancel{2,1,1}, \underline{0}, \cancel{6} \rangle)$
$(\{g_3\}, \langle \cancel{2,2,1,7,6} \rangle)$
$(\{g_4\}, \langle \underline{\mathbf{8}}, 9, \cancel{2,6,7} \rangle)$
$\perp$

**Fig. 4.** Pattern concept lattice of pattern structure from Table 1 with $\theta = 1$. When an interval from a pattern intent has same left and right borders, a value is given instead for sake of readability.

## 4 Discussion and Conclusion

This paper focused on the problem of biclustering numerical data with formal concept analysis. The goal was not to propose a new kind of bicluster, but rather to argue that two existing types of biclusters can be extracted using FCA techniques. For that matter, we proposed two methods producing equivalent results. The first is based on conceptual scaling, while the second on interval pattern structures. It is now expected to experiment these approaches, compare them with other biclustering algorithms (e.g. from [2]) and investigate how to handle other types of biclusters defined in [16]. We should also study the impact of the variation of $\theta$ on the concept lattice granularity, or dually on the number of formal contexts/concepts. Finally, we should examine how formal concept analysis in fuzzy seetings can contribute to biclustering problems. Indeed, similarity and tolerance relations are widely studied in such settings [1].

We discuss now our both methods.

Consider the method based on scaling. The strength of such approach is to produce binary tables. Any FCA algorithm (discussed and compared in [15]), or closed itemset algorithm (e.g. Charm [8]) can be used for extracting biclusters. Moreover, since each context of the produced family is independent from the others, a distributed computation is naturally possible: one core can be assigned for each formal context. It also allows to mine other kinds of binary patterns. For example, one can mine fault-tolerant patterns that would correspond to quasi biclusters of similar values, i.e. accepting some exceptions, see e.g. [17]. Meanwhile, searching for frequent biclusters (i.e. involving a number of objects higher than a user-defined threshold [18]) is straightforward. It rises also interesting questions: what is the meaning of an association rule? of a minimal generator?

The second method proposes to extract biclusters from a concept lattice, providing an interesting ordered hierarchy of biclusters. Computing the pattern concept lattice by adapting standard FCA algorithms such as CloseByOne is efficient as experimented in [12], while this algorithm can be parallelized [13]. In [10], CloseByOne was adapted to mine frequent closed interval patterns and their minimal generators. How this algorithm can be adapted for mining frequent biclusters is an interesting perspective of research. The fact that biclusters can be extracted from an ordered hierarchy of concepts make the pattern concept lattice a good structure for user queries. For example, a biologist may be interested in a particular set of genes for a given study. Accordingly, navigating in the concept lattice helps him discovering the different biclusters in which those genes occurs with other good candidates. We can describe such query as extensional since it starts by given a set of objects. On another hand, the approach based on scaling is more useful for so called intentional queries: the biologist is interested in all biclusters with values in a given interval (or class of tolerance) and accordingly only selects the formal context associated to this class.

## Acknowledgments

The second author was supported by the project of the Russian Foundation for Basic Research, grant no. 08-07-92497-NTsNIL_a. Other authors were supported by the "Contrat de Plan Etat Région – Modélisation des biomolécules et leurs intéractions" (Lorraine, France, 2007–2013). Authors would like to thank Sébastien Vaillant for implementing the scaling procedure.

## References

1. Belohlávek, R., Funioková, T.: Similarity and fuzzy tolerance spaces. J. Log. Comput. 14(6), 827–855 (2004)
2. Besson, J., Robardet, C., Raedt, L.D., Boulicaut, J.F.: Mining bi-sets in numerical data. In: Džeroski, S., Struyf, J. (eds.) KDID 2006. LNCS, vol. 4747, pp. 11–23. Springer, Heidelberg (2007)
3. Boulicaut, J.F., Besson, J.: Actionability and formal concepts: A data mining perspective. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 14–31. Springer, Heidelberg (2008)
4. Cheng, Y., Church, G.: Biclustering of expression data. In: Proc. 8th International Conference on Intelligent Systems for Molecular Biology (ISBM), pp. 93–103 (2000)
5. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Heidelberg (1999)
6. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS 2001. LNCS (LNAI), vol. 2120, pp. 129–142. Springer, Heidelberg (2001)
7. Hartigan, J.A.: Direct clustering of a data matrix. J. Am. Statistical Assoc. 67(337), 123–129 (1972)
8. Hsiao, C.J., Zaki, M.J.: Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. IEEE Trans. on Knowl. and Data Eng. 17(4), 462–478 (2005)
9. Jiang, D., Tang, C., Zhang, A.: Cluster analysis for gene expression data: a survey. IEEE Transactions on Knowledge and Data Engineering 16(11), 1370–1386 (2004)
10. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Pattern Mining in Numerical Data: Extracting Closed Patterns and their Generators. Research Report RR-7416, INRIA (2010)
11. Kaytoue, M., Assaghir, Z., Napoli, A., Kuznetsov, S.O.: Embedding tolerance relations in formal concept analysis: an application in information fusion. In: Huang, J., Koudas, N., Jones, G., Wu, X., Collins-Thompson, K., An, A. (eds.) CIKM, pp. 1689–1692. ACM, New York (2010)
12. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Information Sciences (2010) (in Press, Corrected Proof)
13. Krajca, P., Outrata, J., Vychodil, V.: Advances in algorithms based on cbo. In: Kryszkiewicz, M., Obiedkov, S. (eds.) International Conference on Concept Lattices and Their Applications (2010)
14. Kuznetsov, S.O.: Galois connections in data analysis: Contributions from the soviet era and modern russian research. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis. LNCS (LNAI), vol. 3626, pp. 196–225. Springer, Heidelberg (2005)

15. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Intell. 14(2-3), 189–216 (2002)
16. Madeira, S., Oliveira, A.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
17. Pensa, R.G., Boulicaut, J.F.: Towards fault-tolerant formal concept analysis. In: AI*IA. pp. 212–223 (2005)
18. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing iceberg concept lattices with titanic. Data Knowl. Eng. 42(2), 189–222 (2002)