Mining Bi-sets in Numerical Data

Jérémy Besson[1,2], Céline Robardet[1],
Luc De Raedt[3], and Jean-François Boulicaut[1]

[1] LIRIS UMR 5205 CNRS/INSA Lyon
Bâtiment Blaise Pascal, F-69621 Villeurbanne, France
[2] UMR INRA/INSERM 1235
F-69372 Lyon cedex 08, France
[3] Albert-Ludwigs-Universitat Freiburg
Georges-Kohler-Allee, Gebaude 079 D-79110 Freiburg, Germany
celine.robardet@insa-lyon.fr

Abstract. Thanks to an important research effort the last few years, inductive queries on set patterns and complete solvers which can evaluate them on large 0/1 data sets have been proved extremely useful. However, for many application domains, the raw data is numerical (matrices of real numbers whose dimensions denote objects and properties). Therefore, using efficient 0/1 mining techniques needs for tedious Boolean property encoding phases. This is, e.g., the case, when considering microarray data mining and its impact for knowledge discovery in molecular biology. We consider the possibility to mine directly numerical data to extract collections of relevant bi-sets, i.e., couples of associated sets of objects and attributes which satisfy some user-defined constraints. Not only we propose a new pattern domain but also we introduce a complete solver for computing the so-called numerical bi-sets. Preliminary experimental validation is given.

1 Introduction

Popular data mining techniques concern 0/1 data analysis by means of set patterns (i.e., frequent sets, association rules, closed sets, formal concepts). The huge research effort of the last 10 years has given rise to efficient complete solvers, i.e., algorithms which can compute complete collections of the set patterns which satisfy user-defined constraints (e.g., minimal frequency, minimal confidence, closeness or maximality). It is however common that the considered raw data is available as matrices where we get numerical values for a collection of attributes describing a collection of objects. Therefore, using the efficient techniques in 0/1 data has to start by Boolean property encoding, i.e., the computation of Boolean values for new sets of attributes. For instance, raw microarray data can be considered as a matrix whose rows denote biological samples and columns denote genes. In that context, each cell of the matrix is a quantitative measure of the activity of a given gene in a given biological sample. Several researchers have considered how to encode Boolean gene expression properties like, e.g., gene over-expression [1,7,12,11]. In such papers, the computed Boolean matrix has the same number of attributes

S. Džeroski and J. Struyf (Eds.): KDID 2006, LNCS 4747, pp. 11–23, 2007.

than the raw data but it encodes only one specific property. Given such datasets, efficient techniques like association rule mining (see, e.g., [1,7]) or formal concept discovery (see, e.g., [4]) have been considered.

Such a Boolean encoding phase is however tedious. For instance, we still lack a consensus on how the over-expression property of a gene can be specified or assessed. As a result, different views on over-expression will lead to different Boolean encoding and thus potentially quite different collections of patterns. To overcome these problems, we investigate the possibility to mine directly the numerical data to find interesting local patterns. Global pattern mining from numerical data, e.g., clustering and bi-clustering, has been extensively studied (see [10] for a survey). Heuristic search for local patterns has been studied as well (see, e.g., [2]). However, very few researchers have investigated the non heuristic, say complete, search of well-specified local patterns from numerical data. In this paper, we introduce the Numerical Bi-Sets as a new pattern domain (NBS). Intuitively, we specify collections of bi-sets, i.e., associated sets of rows and columns such that the specified cells (for each row-column pair) of the matrix contain similar values. This property is formalized in terms of constraints, and we provide a complete solver for computing NBS patterns. We start from a recent formalization of constraint-based bi-set mining from 0/1 data (extension of formal concepts towards fault-tolerance introduced in [3]) both for the design of the pattern domain and its associated solver. The next section concerns the formalization of the NBS pattern domain and its properties. Section 3 sketches our algorithm and Section 4 provides preliminary experimental results. Section 5 discusses related work and, finally, Section 6 concludes.

2 A New Pattern Domain for Numerical Data Analysis

Let us consider a set of objects $\mathcal{O}$ and a set of properties $\mathcal{P}$ such that $|\mathcal{O}| = n$ and $|\mathcal{P}| = m$. Let us denote by $\mathcal{M}$ a real valued matrix of dimension $n \times m$ such that $\mathcal{M}(i,j)$ denotes the value of property $j \in \mathcal{P}$ for the object $i \in \mathcal{O}$ (see Table 1 for an example). Our language of patterns is the language of bi-sets, i.e., couples made of a set of rows (objects) and a set of columns (properties). Intuitively, a bi-set (X,Y) with $X \in 2^{\mathcal{O}}$ and $Y \in 2^{\mathcal{P}}$ can be considered as a rectangle or sub-matrix within $\mathcal{M}$ modulo row and column permutations.

Definition 1 (NBS). *Numerical Bi-Sets (or* NBS *patterns) in a matrix are the bi-sets* (X,Y) *such that* $|X| \geq 1$ *and* $|Y| \geq 1$ *(*$X \subseteq \mathcal{O}$, $Y \subseteq \mathcal{P}$*) which satisfy the constraint* $\mathcal{C}_{in} \wedge \mathcal{C}_{out}$*:*

$$\mathcal{C}_{in}(X,Y) \equiv |\max_{i \in X,\, j \in Y} \mathcal{M}(i,j) - \min_{i \in X,\, j \in Y} \mathcal{M}(i,j)| \leq \epsilon$$

$$\mathcal{C}_{out}(X,Y) \equiv \forall y \in \mathcal{P} \setminus Y,\ |\max_{i \in X,\, j \in Y \cup \{y\}} \mathcal{M}(i,j) - \min_{i \in X,\, j \in Y \cup \{y\}} \mathcal{M}(i,j)| > \epsilon$$

$$\forall x \in \mathcal{O} \setminus X,\ |\max_{i \in X \cup \{x\},\, j \in Y} \mathcal{M}(i,j) - \min_{i \in X \cup \{x\},\, j \in Y} \mathcal{M}(i,j)| > \epsilon$$

where ϵ *is a user-defined parameter.*

Each NBS pattern defines a sub-matrix $\mathcal{S}$ of $\mathcal{M}$ such that the absolute value of the difference between the maximum value and the minimum value on $\mathcal{S}$ is less

Table 1. A real valued matrix; the bold rectangles indicate two NBS patterns

	p_1	p_2	p_3	p_4	p_5
o_1	1	2	2	1	6
o_2	2	1	1	0	6
o_3	2	2	1	7	6
o_4	8	9	2	6	7

or equal to ϵ (see $\mathcal{C}_{in}$). Furthermore, no object or property can be added to the bi-set without violating this constraint (see $\mathcal{C}_{out}$). This ensures the maximality of the specified bi-sets.

In Figure 1 (left), we can find the complete collection of NBS patterns which hold in the data from Table 1 when we have $\epsilon = 1$. In Table 1, the two bold rectangles are two examples of such NBS patterns (i.e., the underlined patterns of Figure 1 (left)). Figure 1 (right) is an alternative representation for them: each cross in the 3D-diagram corresponds to an element in the matrix from Table 1.

The search space for bi-sets can be ordered thanks to a specialization relation.

Definition 2 (Specialization and monotonicity). *Our specialization relation on bi-sets denoted $\preceq$ is defined as follows: $(\bot_O, \bot_P) \preceq (\top_O, \top_P)$ iff $\bot_O \subseteq \top_O$ and $\bot_P \subseteq \top_P$. We say that $(\top_O, \top_P)$ extends or is an extension of $(\bot_O, \bot_P)$. A constraint $\mathcal{C}$ is anti-monotonic w.r.t. $\preceq$ iff $\forall B$ and $D \in 2^{\mathcal{O}} \times 2^{\mathcal{P}}$ s.t. $B \preceq D$, $\mathcal{C}(D) \Rightarrow \mathcal{C}(B)$. Dually, $\mathcal{C}$ is monotonic w.r.t. $\preceq$ iff $\mathcal{C}(B) \Rightarrow \mathcal{C}(D)$.*

Assume $\mathcal{W}_\epsilon$ denotes the whole collection of NBS patterns for a given threshold ϵ. Let us now discuss some interesting properties of this new pattern domain:

- $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$ are respectively anti-monotonic and monotonic w.r.t. $\preceq$ (see Property 1).
- Each NBS pattern (X, Y) from $\mathcal{W}_\epsilon$ is maximal w.r.t. $\preceq$ (see Property 2).
- If there exists a bi-set (X, Y) with similar values (belonging to an interval of size ϵ), then there exists a NBS (X', Y') from $\mathcal{W}_\epsilon$ such that $(X, Y) \preceq (X', Y\prime)$ (see Property 3).
- When ϵ increases, the size of NBS pattern increases too, whereas some new NBS patterns which are not extensions of previous one can appear (see Property 4).
- The collection of numerical bi-sets is paving the dataset (see Corollary 1), i.e., any data item belongs to at least one NBS pattern.

Property 1 (Monotonicity). The constraint $\mathcal{C}_{in}$ is anti-monotonic and the constraint $\mathcal{C}_{out}$ is monotonic.

Proof. Let (X, Y) a bi-set s.t. $\mathcal{C}_{in}(X, Y)$ is true, and let (X', Y') be a bi-set s.t. $(X', Y') \preceq (X, Y)$. This implies that $\mathcal{C}_{in}(X', Y')$ is also true:

$$|\max_{i \in X',\, j \in Y'} \mathcal{M}(i,j) - \min_{i \in X',\, j \in Y'} \mathcal{M}(i,j)|$$
$$\leq |\max_{i \in X,\, j \in Y} \mathcal{M}(i,j) - \min_{i \in X,\, j \in Y} \mathcal{M}(i,j)| \leq \epsilon$$

$$((o_1, o_2, o_3, o_4), (p_5))$$
$$\underline{((o_3, o_4), (p_4, p_5))}$$
$$((o_4), (p_1, p_5))$$
$$((o_1, o_2, o_3, o_4), (p_3))$$
$$((o_4), (p_1, p_2))$$
$$((o_2), (p_2, p_3, p_4))$$
$$((o_1, o_2), (p_4))$$
$$((o_1), (p_1, p_2, p_3, p_4))$$
$$\underline{((o_1, o_2, o_3), (p_1, p_2, p_3))}$$

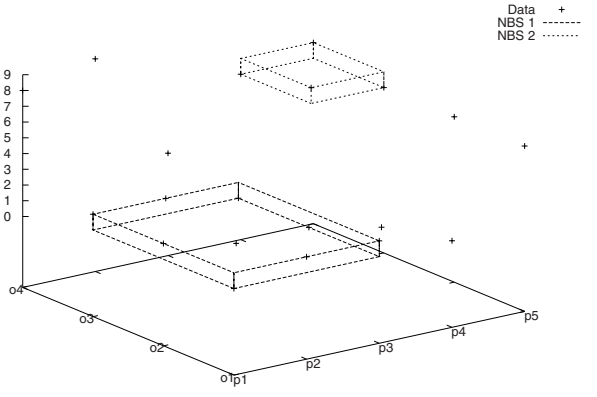

Fig. 1. Examples of NBS

If (X, Y) satisfies $\mathcal{C}_{out}$ and $(X, Y) \preceq (X', Y')$, then $\mathcal{C}_{out}(X', Y')$ is also true:

$$\forall y \in \mathcal{P} \setminus Y, \; | \max_{i \in X,\, j \in Y \cup \{y\}} \mathcal{M}(i,j) - \min_{i \in X,\, j \in Y \cup \{y\}} \mathcal{M}(i,j)|$$
$$> \forall y \in \mathcal{P} \setminus Y', \; | \max_{i \in X',\, j \in Y' \cup \{y\}} \mathcal{M}(i,j) - \min_{i \in X',\, j \in Y' \cup \{y\}} \mathcal{M}(i,j)| > \epsilon$$

Property 2 (Maximality). The NBS patterns are maximal bi-sets w.r.t. our specialization relation $\preceq$, i.e., if $(X, \perp_P)$ and $(X, \top_P)$ are two NBS patterns from $\mathcal{W}_\epsilon$, then $\perp_P \not\subseteq \top_P$ and $\top_P \not\subseteq \perp_P$.

Proof. Assume $\perp_P \subseteq \top_P$. $(X, \perp_P)$ does not satisfy Equation 2, because for $y \in \top_P \setminus \perp_P$, $|\max_{i \in X} \mathcal{M}(i, y) - \min_{i \in X} \mathcal{M}(i, y)| \leq \epsilon$.

Property 3 (NBS patterns extending bi-sets of close values). Let $I_1, I_2 \in \mathbb{R}$, $I_1 \leq I_2$, and (X, Y) be a bi-set such that $\forall i \in X$, $\forall j \in Y$, $\mathcal{M}(i, j) \in [I_1, I_2]$. Then, there exists a NBS (U, V) with $\epsilon = |I_1 - I_2|$ such that $X \subseteq U$ and $Y \subseteq V$.

Thus, if there are bi-sets of which all values are within a small range, there exists at least one NBS pattern which extends it.

Proof. V can be recursively constructed from $Y' = Y$ by adding a property y s.t. $y \in \mathcal{P} \setminus Y'$ to Y' if $|\max_{i \in X,\, j \in Y' \cup \{y\}} \mathcal{M}(i,j) - \min_{i \in X,\, j \in Y' \cup \{y\}} \mathcal{M}(i,j)| \leq \epsilon$, and then continue until no further property can be added. At the end, $Y' = V$. After that, we extend in a similar way the set X towards U. By construction, (U, V) is a NBS pattern with $\epsilon = |I_1 - I_2|$. Notice that we can have several (U, V) which extend (X, Y).

When $\epsilon = 0$, the NBS pattern collection contains all maximal bi-sets of identical values. As a result, we get a paving (with overlapping) of the whole dataset.

Property 4 (NBS pattern size is growing with ϵ). Let (X, Y) be a NBS pattern from $\mathcal{W}_\epsilon$. There exists $(X', Y') \in \mathcal{W}_{\epsilon'}$ with $\epsilon' > \epsilon$ such that $X \subseteq X'$ and $Y \subseteq Y'$.

Proof. Proof is trivial given Property 3.

Corollary 1. *As $\mathcal{W}_0$ is paving the data, then $\mathcal{W}_\epsilon$ is paving the data as well.*

3 Algorithm

The whole collection of bi-sets ordered by $\preceq$ forms a lattice whose bottom is $(\bot_O, \bot_P) = (\emptyset, \emptyset)$ and top is $(\top_O, \top_P) = (\mathcal{O}, \mathcal{P})$. Let us denote by $\mathcal{B}$ the set of sublattices[1] of $((\emptyset, \emptyset), (\mathcal{O}, \mathcal{P}))$: $\mathcal{B} = \{((\bot_O, \bot_P), (\top_O, \top_P))$ s.t. $\bot_O, \top_O \in 2^{\mathcal{O}}, \bot_P, \top_P \in 2^{\mathcal{P}}$ and $\bot_O \subseteq \top_O$, $\bot_P \subseteq \top_P\}$ where the first (resp. the second) bi-set is the bottom (resp. the top) element.

Property 5. Let $NBS_F = ((\bot_O, \bot_P), (\top_O, \top_P)) \in \mathcal{B}$, for all $(X, Y) \in NBS_F$ we have the following properties:

- $e \in \bot_O \Rightarrow e \in X$
- $e \in \bot_P \Rightarrow e \in Y$
- $e \notin \top_O \Rightarrow e \notin X$
- $e \notin \top_P \Rightarrow e \notin Y$

$NBS_F = ((\bot_O, \bot_P), (\top_O, \top_P))$ is the set of all the bi-sets (X, Y) s.t. $\bot_O \subseteq X \subseteq \top_O$ and $\bot_P \subseteq Y \subseteq \top_P$. A sublattice represents explicitly a search space for bi-sets.

Our algorithm NBS-Miner explores some of the sublattices of $\mathcal{B}$ built by means of three mechanisms: enumeration, pruning and propagation. It starts with the sublattice $((\emptyset, \emptyset), (\mathcal{O}, \mathcal{P}))$, i.e., the lattice containing all the possible bi-sets. Table 2 introduces the algorithm NBS-Miner. We now provide details about the three mecanisms.

3.1 Candidate Enumeration

The enumeration function splits recursively the current sublattice (the candidate), say NBS_F, in two new sublattices containing all the bi-sets of NBS_F.

[1] X is a sublattice of Y if Y is a lattice, X is a subset of Y and X is a lattice with the same join and meet operations than Y.

Property 6. Let $NBS_F = ((\bot_O, \bot_P), (\top_O, \top_P)) \in \mathcal{B}$ and $e \in \top_O \setminus \bot_O$, then $NBS_1 = ((\bot_O \cup \{e\}, \bot_P), (\top_O, \top_P))$ and $NBS_2 = ((\bot_O, \bot_P), (\top_O \setminus \{e\}, \top_P))$ is a partition of NBS_F. NBS_1 contains all the bi-sets of NBS_F which contain e and NBS_2 contains all the bi-sets of NBS_F which do not contain e. If $e \in \top_P \setminus \bot_P$, $NBS_1 = ((\bot_O, \bot_P \cup \{e\}), (\top_O, \top_P))$ and $NBS_2 = ((\bot_O, \bot_P), (\top_O, \top_P \setminus e))$ is a partition of NBS_F as well.

The enumeration function selects an element of the set $e \in \top_O \setminus \bot_P \cup \top_P \setminus \bot_P$ and its generates two new sublattices. More formally, we use the following functions *Enum* and *Choose*.

Let $Enum : \mathcal{B} \times \mathcal{O} \cup \mathcal{P} \rightarrow \mathcal{B}^2$ such that

$$Enum(((\bot_O, \bot_P), (\top_O, \top_P)), e) = \begin{cases} (((\bot_O \cup \{e\}, \bot_P), (\top_O, \top_P)), ((\bot_O, \bot_P), (\top_O \setminus \{e\}, \top_P))) \text{ if } e \in \mathcal{O} \\ (((\bot_O, \bot_P \cup \{e\}), (\top_O, \top_P)),\ ((\bot_O, \bot_P), (\top_O, \top_P \setminus \{e\}))) \text{ if } e \in \mathcal{P} \end{cases}$$

where $e \in \top_O \setminus \bot_O$ or $e \in \top_P \setminus \bot_P$. *Enum* generates two new sublattices which are a partition of its input parameter.

Let $Choose : \mathcal{B} \rightarrow \mathcal{O} \cup \mathcal{P}$ be a function which returns one of the element $e \in \top_O \setminus \bot_O \cup \top_P \setminus \bot_P$.

3.2 Candidate Pruning

Obviously, we do not want to explore all the bi-sets. We want either to stop the enumeration when one can ensure that none bi-set of NBS_F is a NBS (Pruning) or to reduce the search space when a part of NBS_F can be removed witout loosing any NBS pattern (Propagation). The sublattice allows to compute bounds of any (anti-)monotonic constraints w.r.t. $\preceq$. For instance, $\mathcal{C}_{min\_area}(X, Y) \equiv \#X \times \#Y > 20$ is a monotonic constraint and $\mathcal{C}_{max\_area}(X, Y) \equiv \#X \times \#Y < 3$ is an anti-monotonic constraint, when #E denotes the size of the set E. If $NBS_F = ((\{o_1, o_3\}, \{p_1, p_2\}), (\{o_1, o_2, o_3, o_4\}, \{p_1, p_2, p_3, p_4\}))$ then none of the bi-sets of NBS_F satisfy $\mathcal{C}_{min\_area}$ and $\mathcal{C}_{max\_area}$. Actually, we have $\#\{o_1, o_3\} \times \#\{p_1, p_2\} > 3$ and $\#\{o_1, o_2, o_3, o_4\} \times \#\{p_1, p_2, p_3, p_4\} < 20$. None bi-set satisfies $\mathcal{C}_{min\_area}$ and $\mathcal{C}_{max\_area}$, whatsoever the enumeration. Intuitively, the monotonic constraints use the top of the sublattice to compute a bound whereas the anti-monotonic constraints use its bottom.

For the pruning, we use the following function:

Let $Prune^m_{\mathcal{C}} : \mathcal{B} \rightarrow \{\text{TRUE,FALSE}\}$ be a function which returns TRUE iff the monotonic constraint $\mathcal{C}^m$ (w.r.t. $\preceq$) is satisfied by the top of the sublattice.

$$Prune^m_{\mathcal{C}}((\bot_O, \bot_P), (\top_O, \top_P)) \equiv \mathcal{C}^m(\top_O, \top_P)$$

If $Prune^m_{\mathcal{C}}((\bot_O, \bot_P), (\top_O, \top_P))$ is false then none of the bi-sets contained in the sublattice satisfies $\mathcal{C}^m$.

Let $Prune^{am}_{\mathcal{C}} : \mathcal{B} \rightarrow \{\text{TRUE,FALSE}\}$ be a function which returns TRUE iff the anti-monotonic constraint $\mathcal{C}^{am}$ (w.r.t $\preceq$) is satisfied by te bottom of the sublattice:

$$Prune^{am}_{\mathcal{C}}((\bot_G, \bot_M), (\top_G, \top_M)) \equiv \mathcal{C}^{am}(\bot_G, \bot_M)$$

If $Prune^{am}_{\mathcal{C}}((\bot_O, \bot_P), (\top_O, \top_P))$ is false then none of the bi-sets contained in the sublattice satisfies $\mathcal{C}^{am}$.

Let $Prune_{\mathcal{C}_{NBS}} : \mathcal{B} \rightarrow \{\text{TRUE,FALSE}\}$ be the pruning function. Due to Property 1, we have

$$Prune_{\mathcal{C}_{NBS}}((\bot_O, \bot_P), (\top_O, \top_P)) \equiv \mathcal{C}_{in}(\bot_O, \bot_P) \wedge \mathcal{C}_{out}(\top_O, \top_P)$$

When $Prune_{\mathcal{C}_{NBS}}((\bot_O, \bot_P), (\top_O, \top_P))$ is false then no NBS pattern is contained in the sublattice $((\bot_O, \bot_P), (\top_O, \top_P))$.

3.3 Propagation

The propagation plays another role. It enables to reduce the size of the search space, i.e., it does not consider the entire current sublattice NBS_F but a smaller sublattice $NBS_P \in \mathcal{B}$ such that $NBS_P \subset NBS_F$. For instance, if $((\bot_O \cup \{e_1\}, \bot_P), (\top_O, \top_P))$ and $((\bot_O, \bot_P), (\top_O, \top_P \setminus \{e_2\}))$ do not contain any NBS pattern, then we can keep going the enumeration process with $((\bot_O, \bot_P \cup \{e_2\}), (\top_O \setminus e_1, \top_P))$ instead of NBS_F. $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$ can be used to reduce the size of the sublattices by moving objects of $\top_O \setminus \bot_O$ into $\bot_O$ or outside $\top_O$, and similarly on attributes. The following function is used to reduce the size of the sublattice:

The function $Prop_{in}\, \mathcal{B} \rightarrow \mathcal{B}$ and $Prop_{out}\, \mathcal{B} \rightarrow \mathcal{B}$ are used to do it as follow:

$$\begin{aligned}
&Prop_{in}((\bot_O, \bot_P), (\top_O, \top_P)) = \{((\bot^1_O, \bot^1_P), (\top_O, \top_P)) \in \mathcal{B} \mid \\
&\bot^1_O = \bot_O \cup \{x \in \top_O \setminus \bot_O \mid \mathcal{C}_{out}((\bot_O, \bot_P), (\top_O \setminus \{x\}, \top_P)) \text{ is false}\} \\
&\bot^1_P = \bot_P \cup \{x \in \top_P \setminus \bot_P \mid \mathcal{C}_{out}((\bot_O, \bot_P), (\top_O, \top_P \setminus \{x\})) \text{ is false}\}\}
\end{aligned}$$

$$\begin{aligned}
&Prop_{out}((\bot_O, \bot_P), (\top_O, \top_P)) = \{((\bot_O, \bot_P), (\top^1_O, \top^1_P)) \in \mathcal{B} \mid \\
&\top^1_O = \top_O \setminus \{x \in \top_O \setminus \bot_O \mid \mathcal{C}_{in}((\bot_O \cup \{x\}, \bot_P), (\top_O, \top_P)) \text{ is false}\} \\
&\top^1_P = \top_P \setminus \{x \in \top_P \setminus \bot_P \mid \mathcal{C}_{in}((\bot_O, \bot_P), (\top_O, \top_P \cup \{x\})) \text{ is false}\}\}
\end{aligned}$$

Let $Prop\, \mathcal{B} \rightarrow \mathcal{B}$ s.t. $Prop_{in}(Prop_{out}(\mathcal{L}))$ is recursively applied as long as its result changes.

We call a leaf a sublattice $\mathcal{L} = ((\bot_O, \bot_P), (\top_O, \top_P))$ which contains only one bi-set i.e., $(\bot_O, \bot_P) = (\top_O, \top_P)$. NBS are these leaves.

Example 1. Here are examples of the function $Prop$ with the data of Table 1.

- $((\bot_O, \bot_P), (\top_O, \top_P)) = ((\{o_1\}, \{p_1\}), (\{o_1, o_2, o_3, o_4\}, \{p_1, p_2, p_3, p_4, p_5\}))$
  $Prop((\bot_O, \bot_P), (\top_O, \top_P)) = ((\bot_O, \bot_P), (\top_O \setminus \{o_4\}, \top_P \setminus \{p_5\}))$
- $((\bot_O, \bot_P), (\top_O, \top_P)) = ((\{o_1, o_2\}, \{p_1\}), (\{o_1, o_2, o_3\}, \{p_1, p_2, p_3, p_4\}))$
  $Prop_{out}((\bot_O, \bot_P), (\top_O, \top_P)) = ((\bot_O, \bot_P), (\top_O, \top_P \setminus \{p_4\}))$
  $Prop_{in}((\bot_O, \bot_P), (\top_O, \top_P \setminus \{p_4\})) =$
  $((\{o_1, o_2, o_3\}, \{p_1, p_2, p_3\}), (\{o_1, o_2, o_3\}, \{p_1, p_2, p_3\}))$

Table 2. NBS-MINER pseudo-code

$\mathcal{M}$ is a real valued matrix, $\mathcal{C}$ a conjunction of monotonic and anti-monotonic constraints on $2^{\mathcal{O}} \times 2^{\mathcal{P}}$ and ϵ is a positive value.

NBS-Miner

Generate$((\emptyset, \emptyset), (\mathcal{O}, \mathcal{P}))$

End NBS-Miner
Generate$(\mathcal{L})$

Let $\mathcal{L} = ((\bot_O, \bot_P), (\top_O, \top_P))$
$\mathcal{L} \leftarrow Prop(\mathcal{L})$
If $Prune(\mathcal{L})$ then
  If $(\bot_O, \bot_P) \neq (\top_O, \top_P)$ then
    $(\mathcal{L}_1, \mathcal{L}_2) \leftarrow Enum(\mathcal{L}, Choose(\mathcal{L}))$
    Generate$(\mathcal{L}_1)$
    Generate$(\mathcal{L}_2)$
  Else Store $\mathcal{L}$
  End if
End if

End Generate

4 Experiments

We report a preliminary experimental evaluation of the NBS pattern domain and its implemented solver. We have been considering the "peaks" matrix of matlab (30*30 matrix with values ranging between -10 and +9). We used $\epsilon = 4.5$ and we obtained 1700 NBS patterns. On Figure 2, we plot in white one extracted NBS. The two axes ranged from 0 to 30 correspond to the two matrix dimensions and the third one indicates their corresponding values (row-column pairs).

In a second experiment, we enforced that the values inside the extracted patterns to be greater than 1.95 (minimal value constraint). Figure 3 shows the 228 extracted NBS patterns when $\epsilon = 0.1$. Indeed, the white area corresponds to the union of 228 extracted patterns.

To study the impact of ϵ parameter, we used the malaria dataset [5]. It records the numerical gene expression value of 3 719 genes of Plasmodium falciparum during its complete lifecycle (a time series of 46 biological situations). We used a minimal size constraint on both dimension, i.e., looking for the NBS patterns (X, Y) s.t. $|X| > 4$ and $|Y| > 4$. Furthermore, we have been adding a minimal value constraint. Figure 4 provides the mean and standard deviation of the area of the NBS patterns from this dataset w.r.t. the ϵ value.

As it was expected owed to Property 4, the mean area increases with ϵ.

Figure 5 reports on the number of NBS patterns in the malaria dataset. From $\epsilon = 75$ to $\epsilon = 300$, this number decreases. It shows that the size of the NBS

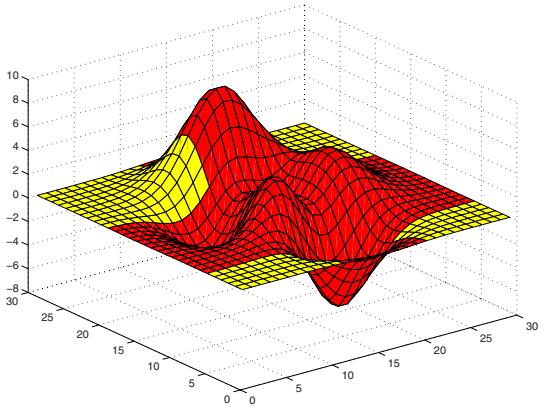

Fig. 2. An example of a NBS pattern

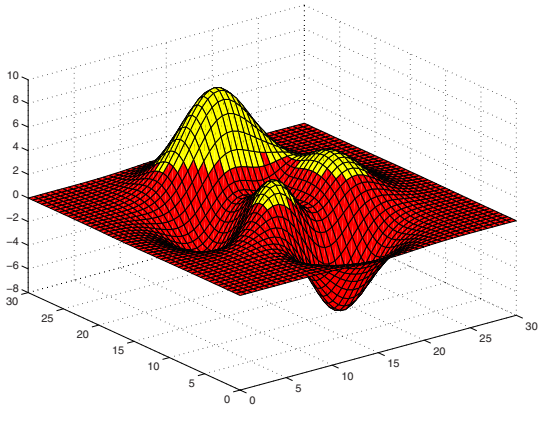

Fig. 3. Examples of extracted NBS

pattern collection tends to decrease when ϵ increases. Intuitively, many patterns are gathered when ϵ increases whereas few patterns are extended by generating more than one new pattern. Moreover, the minimal size constraint can explain the increase of the collection size. Finally, when the pattern size increases with ϵ, new NBS patterns can appear in the collection.

5 Related Work

[14,6,13] propose to extend classical frequent itemset and association rule definitions for numerical data. In [14], the authors generalize the classical notion of itemset support in 0/1 data when considering other data types, e.g., numerical ones. Support computation requires data normalization, first translating the

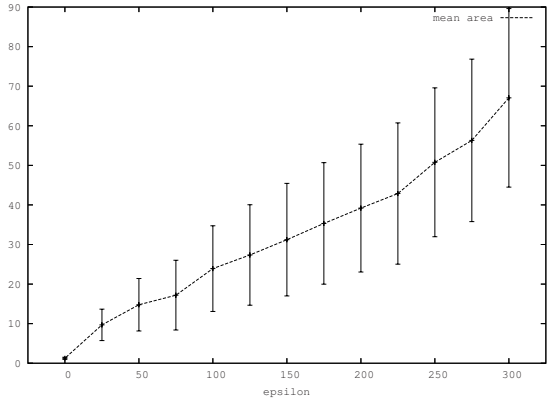

Fig. 4. Mean area of the NBS w.r.t. ϵ

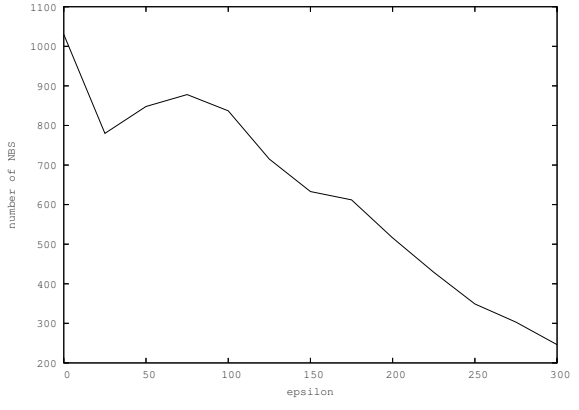

Fig. 5. Collection sizes w.r.t. ϵ

values to be positive, and then dividing each column entry by the sum of the column entries. After such a treatment, each entry is between 0 and 1, and the sum of the values for a column is equal to 1. The support of an itemset is then computed as the sum on each row of the minimum of the entries of this itemset. If the items have identical values on all the rows, then the support is equal to 1, and the more the items are different, the more the support value decreases toward 0. This support function is anti-monotonic, and thus the authors propose to adapt an Apriori algorithm to compute the frequent itemsets according to this new support definition. [6] proposes new methods to measure the support of itemsets in numerical data and categorical data. They adapt three well-known correlation measures: Kendall's τ, Spearman's ρ and Spearman's Footrule F. These measures are based on the rank of the values of objects for each attribute, not the

values themselves. They extend these measures to sets of attributes (instead of 2 variables). Efficient algorithms are proposed. [13] uses an optimization setting for finding association rules in numerical data. The type of extracted association rules is: "if the weighted sum of some variables is greater than a threshold then a different weighted sum of variables is with high probability greater than a second threshold". They propose to use hyperplanes to represent the left-hand and the right-hand sides of such rules. Confidence and coverage measures are used. It is unclear wether it is possible to extend these approaches to bi-set computation.

Hartigan proposes a bi-clustering algorithm that can be considered as a specific collection of bi-sets [8]. He introduced a partition-based algorithm called "Block Clustering". It splits the original data matrix into bi-sets and it uses the variance of the values inside the bi-sets to evaluate the quality of each bi-set. Then, a so-called ideal constant cluster has a variance equal to zero. To avoid the partitioning of the dataset into bi-sets with only one row and one column (i.e., leading to ideal clusters), the algorithm searches for K bi-sets within the data. The quality of a collection of K bi-sets is considered as the sum of the variance of the K bi-sets. Unfortunately, this approach uses a local optimization procedure which can lead to unstable results.

In [15], the authors propose a method to isolate subspace clusters (bi-sets) containing objects varying similarly on subset of columns. They propose to compute bi-sets (X, Y) such that given $a, b \in X$ and $c, d \in Y$ the 2×2 sub-matrix entries $((a, b), (c, d))$ included in (X, Y) satisfies $|\mathcal{M}(a, c) + \mathcal{M}(b, d) - (\mathcal{M}(a, d) + \mathcal{M}(b, c))| \leq \delta$. Intuitively, this constraint enforces that the change of value on the two attributes between the two objects is confined by δ. Thus, inside the bi-sets, the values have the same profile. The algorithm first considers all pairs of objects and all pairs of attributes, and then combines them to compute all the bi-sets satisfying the anti-monotonic constraint.

Liu and Wang [9] have proposed an exhaustive bi-cluster enumeration algorithm. They are looking for order-preserving bi-sets with a minimum number of rows and a minimum number of columns. This means that for each extracted bi-set (X, Y), there exists an order on Y such that according to this order and for each element of X the values are increasing. They want to provide all the bi-clusters that, after column reordering, represent coherent evolutions of the symbols in the matrix. This is achieved by using a pattern discovery algorithm heavily inspired in sequential pattern mining algorithms. These two local pattern types are well defined and efficient solvers are proposed. Notice however that these patterns are not symmetrical: they capture similar variations on one dimension and not similar values.

Except for the bi-clustering method of [8], all these methods focus on one of the two dimensions. We have proposed to compute bi-sets with a symmetrical definition which is one of the main difficulties in bi-set mining. This is indeed one of the lessons from all the previous work on bi-set mining from 0/1 data, and, among others, the several attempts to mine fault-tolerant extensions to formal concepts instead of fault-tolerant itemsets [3].

6 Conclusion

Efficient data mining techniques tackle 0/1 data analysis by means of set patterns. It is however common, for instance in the context of gene expression data analysis, that the considered raw data is available as a collection of real numbers. Therefore, using the available algorithms needs for a beforehand Boolean property encoding. To overcome such a tedious task, we started to investigate the possibility to mine set patterns directly from the numerical data. We introduced the Numerical Bi-Sets as a new pattern domain. Some nice properties of NBS patterns have been considered. We have described our implemented solver NBS-Miner in quite generic terms, i.e., emphasizing the fundamental operations for the complete computation of NBS patterns. Notice also that other monotonic or anti-monotonic constraints can be used in conjunction with $\mathcal{C}_{in} \wedge \mathcal{C}_{out}$, i.e., the constraint which specifies the pattern domain. It means that search space pruning can be enhanced for mining real-life datasets provided that further user-defined constraints are given. The perspectives are obviously related to further experimental validation, especially the study of scalability issues. Furthermore, we still need for an in-depth understanding of the complementarity between NBS pattern mining and bi-set mining from 0/1 data.

Acknowledgments. This research is partially funded by the EU contract IQ FP6-516169 (FET arm of the IST programme). J. Besson is paid by INRA (ASC post-doc).

References

1. Becquet, C., Blachon, S., Jeudy, B., Boulicaut, J.-F., Gandrillon, O.: Strong-association-rule mining for large-scale gene-expression data analysis: a case study on human sage data. Genome Biology, 12 (November 2002)
2. Bergmann, S., Ihmels, J., Barkai, N.: Iterative signature algorithm for the analysis of large-scale gene expression data. Physical Review 67 (March 2003)
3. Besson, J., Pensa, R., Robardet, C., Boulicaut, J.-F.: Constraint-based mining of fault-tolerant patterns from boolean data. In: Bonchi, F., Boulicaut, J.-F. (eds.) KDID 2005. LNCS, vol. 3933, pp. 55–71. Springer, Heidelberg (2006)
4. Besson, J., Robardet, C., Boulicaut, J.-F., Rome, S.: Constraint-based concept mining and its application to microarray data analysis. Intelligent Data Analysis 9(1), 59–82 (2005)
5. Bozdech, Z., Llinás, M., Pulliam, B., Wong, E., Zhu, J., DeRisi, J.: The transcriptome of the intraerythrocytic developmental cycle of plasmodium falciparum. PLoS Biology 1(1), 1–16 (2003)
6. Calders, T., Goethals, B., Jaroszewicz, S.: Mining rank correlated sets of numerical attributes. In: Proceedings ACM SIGKDD 2006, Philadelphia, USA, August 2006, pp. 96–105 (2006)
7. Creighton, C., Hanash, S.: Mining gene expression databases for association rules. Bioinformatics 19(1), 79–86 (2002)
8. Hartigan, J.: Direct clustering of data matrix. Journal of the American Statistical Association 67(337), 123–129 (1972)

9. Liu, J., Wang, W.: Op-cluster: Clustering by tendency in high dimensional space. In: Proceedings IEEE ICDM'03, Melbourne, USA, December 2003, pp. 187–194 (2003)
10. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. ACM/IEEE Trans. on computational biology and bioinformatics 1(1), 24–45 (2004)
11. Pensa, R., Boulicaut, J.-F.: Boolean property encoding for local set pattern discovery: an application to gene expression data analysis. In: Morik, K., Boulicaut, J.-F., Siebes, A. (eds.) Local Pattern Detection. LNCS (LNAI), vol. 3539, pp. 114–134. Springer, Heidelberg (2005)
12. Pensa, R.G., Leschi, C., Besson, J., Boulicaut, J.-F.: Assessment of discretization techniques for relevant pattern discovery from gene expression data. In: Proceedings ACM BIOKDD 2004, Seattle, USA, August 2004, pp. 24–30 (2004)
13. Ruckert, U., Richter, L., Kramer, S.: Quantitative association rules based on half-spaces: An optimization approach. In: Proceedings IEEE ICDM 2004, November 2004, pp. 507–510, Brighton, UK (2004)
14. Steinbach, M., Tan, P.-N., Xiong, H., Kumar, V.: Generalizing the notion of support. In: Proceedings ACM SIGKDD 2004, Seatle, USA, pp. 689–694 (2004)
15. Wang, H., Wang, W., Yang, J., Yu, P.S.: Clustering by pattern similarity in large data sets. In: Proceedings ACM SIGMOD 2002, Madison, USA, June 2002, pp. 394–405 (2002)