Mining concepts from large SAGE gene expression matrices

François Rioult[1], Céline Robardet[2,3], Sylvain Blachon[2], Bruno Crémilleux[1], Olivier Gandrillon[2], and Jean-François Boulicaut[3]

[1] GREYC CNRS UMR 6072, F-14032 Caen, France
`Francois.Rioult@info.univ-caen.fr`
[2] CGMC CNRS UMR 5534, F-69622 Villeurbanne cedex, France
`gandrillon@cgmc.univ-lyon1.fr`
[3] LIRIS CNRS FRE 2672, F-69621 Villeurbanne cedex, France
`Jean-Francois.Boulicaut@insa-lyon.fr`

Abstract. One of the crucial needs in post-genomic research is to analyze expression matrices (e.g., SAGE and microarray data) to identify a priori interesting sets of genes, e.g., sets of genes that are frequently co-regulated. Such matrices provide expression values for given biological situations (the lines) and given genes (columns). The inductive database framework enables to support knowledge discovery processes by means of sequences of queries that concerns both data processing and pattern querying (extraction, post-processing). We provide a simple formalization of a relevant pattern domain (language of patterns, evaluation functions and primitive constraints) that has been proved useful for specifying various analysis tasks. Recent algorithmic results w.r.t. the efficient evaluation (constraint-based mining) of the so-called inductive queries are emphasized and illustrated on a $90 \times 12\ 636$ human SAGE expression matrix.

1 Introduction

We are now entering the post-genome era and it seems obvious that, in a near future, the critical need will not be to generate data, but to derive knowledge from huge data sets generated at very high throughput. This has been a challenge for quite some time in genomic research, and is now extending to the domain of transcriptome research, i.e., the analysis of gene expression data. Different techniques (including microarray [12] and SAGE [23]) allow to study the simultaneous expression of (tens of) thousands of genes in various biological situations. The data generated by those experiments can then be seen as expression matrices in which the expression level of genes (the columns) are recorded in various biological situations (the lines). What is presently required is to try to find out groups of co-regulated genes, also known as synexpression groups [19], which, based on the guilt by association approach, are assumed to participate in a common function, or module, within the cell. Indeed, biologist often use clustering techniques to identify sets of genes that have similar expression profiles (see, e.g., [13]). Recently, association rule mining has been studied as a complementarity approach for the identification of a priori interesting set of gene [2]. An added-value is to provide a symbolic description that quantify the "association" between sets of genes w.r.t. to boolean expression properties (e.g., over-expression, under-expression, strong variation). Mining large gene expression matrices gives rise to new problems w.r.t. the standard application of association rule mining (e.g., for basket analysis). However, thanks to the properties of the so-called *condensed representations* and Galois connections, [22] shows that it is possible to mine *concepts* [24] in microarray data. Concept post-processing enables to perform various tasks like conceptual clustering or frequent association rule mining (over genes and/or biological situations).

The contribution of this paper is threefold. First, it describes gene expression data analysis within an inductive database approach [14, 6, 10]. For that purpose, we provide a formalization of the *pattern domain* `RNA` and discuss few evaluation functions and primitive constraints that have been proved useful. Next, recent algorithmic results w.r.t. the efficient evaluation of the so-called *inductive queries* are introduced. Finally, we discuss an original research on human SAGE data that leads to a $90 \times 27\ 679$ expression matrix analysis, i.e., a quite large expression matrix w.r.t. previous work. Even though this paper does not present new biological results, the overall approach in biological terms has been already validated on a reduced set of genes [2]. We are pretty confident that given the breakthrough into extraction feasibility, biological meaning will now be extracted almost at will.

In Section 2, we define the `RNA` pattern domain and thus an inductive database approach on gene expression data analysis. In Section 3, we consider inductive query optimization issues. In Section 4 we describe the experimental validation of our approach on two matrices built from public human SAGE data. Section 5 concludes.

2 The `RNA` pattern domain

Mannila and Toivonen have formalized useful data mining tasks as follows [16]. Given, a language of patterns or models $\mathcal{L}$ to be considered, a database $\mathbf{r}$ and a selection predicate q, the aim is then to find the theory $Th(\mathcal{L}, q, \mathbf{r}) = \{\phi \in \mathcal{L} \mid q(\phi, \mathbf{r}) \text{ is true}\}$. Furthermore, it is clear that, in many situations, users are interested in *extended theories*, i.e., not only elements from $\mathcal{L}$ but also the results of some evaluation functions for these a priori interesting patterns or models (e.g., frequency, accuracy). Computing theories can be embedded into the general framework of *inductive databases* as it has been formalized in [6].

The *schema of an inductive database* is a pair $(\mathbf{R}, (\mathcal{Q}_\mathbf{R}, \mathcal{E}))$, where $\mathbf{R}$ is a database schema, $\mathcal{Q}_\mathbf{R}$ is a collection of patterns or models, $\mathcal{E}$ is a collection of evaluation functions that define pattern or model properties in the data. An instance of the schema, an *inductive database* $(\mathbf{r}, s)$ consists of a database $\mathbf{r}$ over the schema $\mathbf{R}$ and a subset $s \subseteq \mathcal{Q}_\mathbf{R}$. A typical KDD process operates on both of the components of an inductive

database. The user can select data from $\mathbf{r}$ and s remains the same. The user can also select subsets s, and $\mathbf{r}$ is not modified. Let us just consider simple typical queries, i.e., selections[1]. A data selection example is $\sigma_C(\mathbf{r}_0, s_0) = (\mathbf{r}_1, s_1)$ where $\mathbf{r}_1 = \sigma_C(\mathbf{r}_0)$ and $s_1 = s_0$. For instance, we will use this kind of query to remove some biological situations that do not verify criterion C. Any data manipulation can be performed there. A pattern selection example is $\tau_{C'}(\mathbf{r}_0, s_0) = (\mathbf{r}_2, s_2)$ where $\mathbf{r}_2 = \mathbf{r}_0$ and s_2 contains only the patterns or models that satisfy criterion C'. For instance, this kind of query can be used to select sets of genes that have some desired properties, e.g., co-regulation in at least p biological situations. Queries on inductive databases satisfy a *closure property*: queries that return data, queries that return patterns or models (data mining and post-processing queries) and queries that cross over the data and the patterns or models (post-processing queries) are all queries on inductive databases and return an inductive database. For instance, we can compute $\tau_{C'}(\sigma_C(\mathbf{r}_0, s_0)) = (\mathbf{r}_3, s_3)$. Any query that has to compute patterns or models is called an *inductive query*. Notice that when the user needs the evaluation functions (e.g., frequencies), their values are computed from the current data part of the inductive database instance.
This approach is studied in depth in the cInQ consortium[2] and has lead to interesting results for local pattern discovery (item sets, association rules, linear graphs, sequences, strings, see [4, 10] for survey papers and an introduction to the terminology). In this paper, we consider a pattern domain related to item sets since biologists consider that useful knowledge about the transcriptome can be expressed as sets of genes and/or sets of biological situations that have some properties. As a methodological guideline, specifying a pattern domain leads to the definition of pattern languages, evaluation functions and primitive constraints. A query language must enable to define standard queries on the data component but also inductive queries. As a first approximation, we can consider that inductive queries (i.e., selection predicates) are built from boolean combinations of primitive constraints.

Let us now consider the languages we have for the `RNA` pattern domain. Let $\mathcal{S}$ denote a set of biological situations and $\mathcal{A}$ denote a set of genes. An expression matrix (ED) associates each couple of $\mathcal{S} \times \mathcal{A}$ a real number, i.e., an expression value. It is out of the scope of this paper to discuss its semantics (exact number of sequenced tags[3] - or absolute frequency- in the case of SAGE data [23], variation between two studied experimental conditions in the case of microarray data [12]).
Raw expression data can be stored in, e.g., relational databases, and be queried by any standard query language (selection of situations, projection on sets of genes). Also, aggregates can be used to derive summarization of raw data, e.g., the expression mean value or the standard variation for each gene. It makes sense to abstract a raw expression matrix into a

[1] Selection of data and patterns are respectively denoted by σ and τ. As it is clear from the context, the operation can also be applied on inductive database instances.
[2] European Contract IST-2000-26469, **c**onsortium on discovering knowledge using **In**ductive **Q**ueries.
[3] For SAGE, tags correspond to genes and the biological situations are called libraries.

boolean matrix $\mathbf{r}$ that records expression properties. In the example from Figure 1, $\mathcal{S} = \{s_1, \ldots s_5\}$ and $\mathcal{A} = \{a_1, a_2, \ldots a_{10}\}$. Each attribute a_j denotes a property about the expression of gene j. The expression data is thus represented by the matrix $\mathbf{r}$ of the binary relation $R \subset \mathcal{S} \times \mathcal{A}$ defined for each situation and each attribute. $(s_i, a_j) \in \mathbf{r}$ denotes that situation i has the property j, e.g., that gene j is over-expressed (under-expressed, has a strong variation) in situation i. In the following, we assume that expression properties encode over-expressions.
The boolean data to be mined is thus a 3-tuple $\mathbf{r} = (\mathcal{S}, \mathcal{A}, R)$.
The `RNA` pattern language is the collection of couples from $\mathcal{L}_\mathcal{A} \times \mathcal{L}_\mathcal{S}$ where $\mathcal{L}_\mathcal{A} = 2^\mathcal{A}$ (sets of genes) and $\mathcal{L}_\mathcal{S} = 2^\mathcal{S}$ (sets of situations). For instance, a typically interesting pattern for a biologist can be (X, T) where X is a set of genes that include at least one transcription factor and which are consistently up-regulated in all the normal lung tissues, i.e., set T.

	Attributes									
Situations	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
s_1	1	1	1	1	0	1	1	0	0	0
s_2	1	1	1	1	0	0	0	0	1	1
s_3	1	1	1	1	0	0	0	0	1	1
s_4	0	0	0	0	1	1	1	1	1	1
s_5	1	0	1	0	1	1	1	1	0	0

Fig. 1. Example of a boolean matrix $\mathbf{r}_1$

To obtain boolean matrices from a raw expression matrix ED, we use one discretization operator. Typically, such an operator assigns the true value when the expression value is above some threshold value. Let us denote by β_1 and β_2 two operators, $\beta_1(ED) = \mathbf{r}_1$ and $\beta_2(ED) = \mathbf{r}_2$ are two boolean matrices that are generally different. Important properties like containment can be proved or studied (see Section 4 for examples).
Among the simple boolean data transformations, *transposition* can be used. If $\mathbf{r} = (\mathcal{S}, \mathcal{A}, R)$ is a boolean expression matrix, the transposed matrix is ${}^t\mathbf{r} = (\mathcal{A}, \mathcal{S}, {}^tR)$ where $(a, s) \in {}^tR \iff (s, a) \in R$.

Let us now consider evaluation functions for the `RNA` pattern domain. We do not claim that this is an exhaustive list of useful primitives on situations $\times$ genes expression matrices. We mainly consider Galois operators (see, e.g., [24]) that have been proved extremely useful.

Definition 1 (Galois connection [24]). *If $T \subseteq \mathcal{S}$ and $X \subseteq \mathcal{A}$, assume $f(T, \mathbf{r}) = \{a \in \mathcal{A} \mid \forall s \in T, (s, a) \in R\}$ and $g(X, \mathbf{r}) = \{s \in \mathcal{S} \mid \forall a \in X, (s, a) \in R\}$. f provides the set of over-expressed genes that are common to a set of situations and g provides the set of situations that share a given set of attributes (expression properties). (f, g) is the so-called Galois connection between $\mathcal{S}$ and $\mathcal{A}$. We use the classical notations $h = f \circ g$ and $h' = g \circ f$ to denote the Galois closure operators.*

Definition 2 (Frequency). *The* frequency *of a set of genes $X \subseteq \mathcal{A}$ denoted $\mathcal{F}(X, \mathbf{r})$ is the size of $g(X, \mathbf{r})$. The frequency of a set of situations $T \subseteq \mathcal{S}$ is the size of $f(T, \mathbf{r})$.*

Given Figure 1 (parameter $\mathbf{r}_1$ is omitted), let us consider the pattern (X, T) where $X = \{a_1, a_3\}$ and $Y = \{s_1, s_2, s_3, s_5\}$. We have $\mathcal{F}(X) = 4$ and $T = g(X)$.

Let us now introduce some primitive constraints for the `RNA` pattern domain. Our primitive constraints are defined on sets of genes and sets of situations.

Definition 3 (Constraints on frequencies). *Given a set of genes $X \subseteq \mathcal{A}$ and an absolute frequency threshold γ, $\mathcal{C}_{\text{minfreq}}(X, \mathbf{r}) \equiv \mathcal{F}(X, \mathbf{r}) \geq \gamma$. Sets that satisfy $\mathcal{C}_{\text{minfreq}}$ are said* γ-frequent *in $\mathbf{r}$. The maximal frequency is defined as $\mathcal{C}_{\text{maxfreq}}(X, \mathbf{r}) \equiv \mathcal{F}(X, \mathbf{r}) \leq \gamma$. These constraints can be defined on sets of situations as well.*

Definition 4 (Closed set and $\mathcal{C}_{Close}$ constraint). *A set of genes $X \subseteq \mathcal{A}$ is closed (it satisfies the $\mathcal{C}_{Close}$ constraint in $\mathbf{r}$) iff $h(X, \mathbf{r}) = X$. A set of situations $T \subseteq \mathcal{S}$ is closed iff $h'(T, \mathbf{r}) = T$.*

Given Figure 1, assume the `RNA` pattern $(\{a_1, a_2\}, \{s_1, s_2, s_3\})$. if $\gamma = 3$, $\{a_1, a_2\}$ satisfies $\mathcal{C}_{\text{minfreq}}$ in $\mathbf{r}_1$. Furthermore, $\{s_1, s_2, s_3\} = g(\{a_1, a_2\})$. $\{s_1, s_2, s_3\}$ is a closed set of situations (i.e., $h'(\{s_1, s_2, s_3\}) = \{s_1, s_2, s_3\}$) but $\{a_1, a_2\}$ is not a closed set on genes: $h(\{a_1, a_2\}) = f(g(\{a_1, a_2\})) = \{a_1, a_2, a_3, a_4\}$.
The closure of a set of genes X, $h(X, \mathbf{r})$, is the maximal (w.r.t. set inclusion) superset of X which has the same frequency than X in $\mathbf{r}$. A closed set of genes is thus a maximal set of genes whose expression properties (true values) are shared by a set of situations. For instance, the closed set $\{a_1, a_3\}$ in $\mathbf{r}_1$ (see Figure 1) is the largest set of genes that are over-expressed simultaneously in situations s_1, s_2, s_3 and s_5. The Galois connection gives rise to *concepts* [24] that associate closed sets of genes with closed sets of situations.

Definition 5 (Concept). *If $X \in \mathcal{L}_{\mathcal{A}}$ and $T \in \mathcal{L}_{\mathcal{S}}$, we say that (X, T) is a concept in $\mathbf{r}$ when $T = g(X, \mathbf{r})$ and $X = f(T, \mathbf{r})$. By construction, concepts are built on closed sets and each closed set of genes (resp. situations) is linked to a closed set of situations (resp. genes) [24].*

Seven `RNA` patterns are concepts in $\mathbf{r}_1$ (see Figure 1). Examples of concepts are $(\{a_1, a_3\}, \{s_1, s_2, s_3, s_5\})$ and $(\{a_1, a_2, a_3, a_4, a_9, a_{10}\}, \{s_2, s_3\})$. $(\{a_1, a_2\}, \{s_1, s_2, s_3\})$ is not a concept.
Concept are interesting for the biologist: they can suggest the so-called transcription modules.

An important kind of primitive constraint concerns the *syntactical restrictions* that can be checked without any access to the data. [17] contains a systematic study of syntactical constraints for sets.

Definition 6 (Syntactic constraints). *A syntactic constraint enforces that a set $Y \in \mathcal{L}_C$, where $\mathcal{L}_C \subseteq \mathcal{L}_{\mathcal{A}}$ or $\mathcal{L}_C \subseteq \mathcal{L}_{\mathcal{S}}$. Various means can be used to specify $\mathcal{L}_C$, e.g., regular expressions.*

Some other interesting constraints can use additional information about the genes or the situations. For instance, given a set of genes X, it is possible to use biological knowledge about gene functions and enforce constraints on gene functions for the genes in X.

Many data mining processes on gene expression matrices can be formalized as the computation of `RNA` patterns whose set components satisfy combinations of primitive constraints.

Mining the frequent sets of genes is specified as the computation of $\{X \in \mathcal{L}_{\mathcal{A}} \mid \mathcal{C}_{\mathrm{minfreq}}(X, \mathbf{r})$ satisfied$\}$. We can then provide each `RNA` pattern of the form $(X, g(X, \mathbf{r}))$ where X is frequent. This collection can suggest synexpression groups. Adding syntactical constraints (e.g., enforcing the presence or the absence of some genes) is also often used by biologists. Frequent sets of situations can be desired as well.

Mining the closed sets of genes is specified as the computation of $\{X \in \mathcal{L}_{\mathcal{A}} \mid \mathcal{C}_{Close}(X, \mathbf{r})$ satisfied$\}$. These sets provide a valuable information to biologists thanks to closeness property, e.g., closed sets of genes are maximal sets of genes that are co-regulated in a set of situations. In that context, each `RNA` pattern of the form $(X, g(X, \mathbf{r}))$ is a concept. Dually, it is possible to compute closed sets on situations T and associate the closed set of genes $f(T, \mathbf{r})$. A typically useful task is to look for every concept (X, T) such that X is frequent. Syntactical restrictions can be used as well.
Many other examples could be given, e.g., feature construction by looking for sets of genes that satisfy $\mathcal{C}_{\mathrm{minfreq}}$ in one data set and $\mathcal{C}_{\mathrm{maxfreq}}$ in another one [11]. As a typical interesting finding for a biologist, one might look for each `RNA` pattern (X, T) such that X is a set of genes that are frequently up-regulated in all the medulloblastomas ($T = g(X)$) and that are infrequently found up-regulated in corresponding normal regions of the brain.
Post-processing queries can be understood as queries on materialized collections of sets: the user selects the sets that fulfill some new criteria. Notice however that, from the specification point of view, they are not different from data mining queries even though the evaluation does not need an extraction phase.

3 Inductive query evaluation

In this section, our goal is to emphasize that the current algorithmic know-how can tackle the evaluation of the kind of inductive query we need thanks to a clever use of the Galois connection. We chose to emphasize constraint-based extraction of sets of genes for which it is then possible to associate sets of situations (using the g operator).
Mining frequent sets has been extensively studied the last 10 years. One major recent progress comes from the various algorithms that compute efficiently the sets that satisfy a conjunction of anti-monotonic and monotonic constraints, e.g., [17, 11, 15, 7]. Indeed, most of the primitive constraints we have considered are monotonic or anti-monotonic. Some of them, the succinct ones [17], are in fact syntactical constraints

that can be put transformed into a conjunction of monotonic and anti-monotonic constraints. We assume the reader knows well the background in constraint-based mining and the efficient use of monotonicity.

Expression matrices have generally a few tens of lines (biological situations) and thousands or even tens of thousands of columns (genes). Thus, the computation of sets of genes that satisfy a given constraint $\mathcal{C}$ is extremely hard. Indeed, as soon as we have more than a few tens of columns, only a quite small subset of the search space can be explored. Then, the size of the solution, i.e., the collection of the sets that satisfy $\mathcal{C}$ can be so huge that no algorithm can compute them all. When a constraint like $\mathcal{C}_{\text{minfreq}}$ is used, it is possible to take a greater frequency threshold to decrease a priori the size of the solution. The used threshold can however be disappointing for the biologist: extracted patterns are so frequent that they are already known (e.g., they are the so-called house-keeping genes). Furthermore, in the expression matrices we have to analyze, the number of the frequent sets can be huge, whatever is the frequency threshold. It comes from the rather low number of lines and thus the small number of possible frequencies. Clearly, Apriori-like algorithms that have to compute the frequency of at least every frequent set can not be used here. Any Apriori-based strategy for pushing the other constraints might fail too.

When the minimal frequency constraint is used, one of the key idea for inductive query optimization in `RNA` can come from the condensed representation of the frequent sets. They contain a much smaller number of sets with their frequencies even though it is straightforward and efficient to regenerate all the frequent sets and their frequencies. Various condensed representations have been studied, see, e.g., [8, 9]. Indeed, it is easy to derive the whole collection of the frequent sets of genes from $\{X \in \mathcal{L}_{\mathcal{A}} \mid \mathcal{C}_{\text{minfreq}}(X, \mathbf{r}) \wedge \mathcal{C}_{Close}(X, \mathbf{r}) \text{ satisfied}\}$. This compact representation can be computed efficiently, see, e.g., [20, 5, 21, 25, 1].

The algorithm we use is based on free set extraction [5]. Freeness characterizes the closed set generators (i.e., the closures of the free sets are the closed sets).

Definition 7 (Freeness and $\mathcal{C}_{\text{free}}$ constraint). *A set of genes $X \subseteq \mathcal{A}$ is free iff the frequency of X in $\mathbf{r}$ is strictly lower than the frequency of every strict subset of X. We say that X satisfies constraint $\mathcal{C}_{\text{free}}$ in $\mathbf{r}$. Interestingly, freeness is an anti-monotonic property while closeness is not an anti-monotonic one.*

Given Figure 1, $\{a_1, a_6\}$ satisfies $\mathcal{C}_{\text{free}}$ in $\mathbf{r}_1$ but $\{a_1, a_2, a_3\}$ does not.

In other terms, we compute the collection of the closed sets of genes as $\{h(X, \mathbf{r}) \in \mathcal{L}_{\mathcal{A}} \mid \mathcal{C}_{\text{free}}(X, \mathbf{r}) \text{ satisfied}\}$. Minimal frequency constraint can be added as well.

Even though these approaches have given excellent results on large matrices for transactional data (e.g., highly correlated and rather dense data in WWW usage mining applications), they often fail on expression matrices because of the their "pathological" dimensions. Furthermore, we want to enable the use of various discretization operators and thus the analysis of more or less dense matrices. It appeared crucial to us that we can achieve a breakthrough w.r.t. extraction feasibility.

We have studied the extraction from the transposed matrices using the Galois connection to infer the results that would have been extracted from the initial matrices. [22] provides a general framework for transposed extractions given a constraint $\mathcal{C}_{\text{minfreq}}$ with the frequency threshold greater than 1. In a context where the number of columns is quite large w.r.t. the number of lines, i.e., the case for gene expression matrices, it is possible to compute every concept (absolute frequency threshold set to 1) based on the following observation:

– The direct extraction computes the closed sets of genes and for each closed set X ($h(X, \mathbf{r}) = X$), computing $g(X, \mathbf{r}) = T$ enables to provide the concept (X, T). This computation can be intractable due to the number of genes.
– The transposed extraction computes the closed sets of situations and for each closed set T ($h(T, {}^t\mathbf{r}) = h'(T, \mathbf{r}) = T$), computing $g(T, {}^t\mathbf{r}) = f(T, \mathbf{r}) = X$ enables to provide the concept (X, T).
– Computing $g(X, \mathbf{r})$ during the direct extraction or $g(T, {}^t\mathbf{r})$ during the transposed extraction can be performed at almost no cost during the computations of the associated free sets and their closures.

Thus, it is possible to obtain the same collection of concepts when extracting them from a matrix or its transposed. The choice between one or the other method can be guided by the dimension of the matrix: for expression matrices, our experience is that transposition is often needed. It is important to know that when concepts are obtained, it is straightforward to provide frequent sets (for genes and situations) and more generally, many constraint-based extractions of `RNA` patterns can be performed by filtering techniques and, eventually, partial regeneration phases.

4 Applications to SAGE data

We are working with the publicly available SAGE data produced from human cells[4] and it leads to difficult data mining contexts.
Analyzing human SAGE data is relevant since this data source has been largely under-exploited today. The only available on line approach consists in comparing the existing libraries 2 by 2 to extract differential information. To the best of our knowledge, [18] is the unique study on the complete human SAGE data mining. One obvious reason for such a poor exploitation lies in the structure of the data, including a high error rate for low frequency tags (and especially tags appearing only once in a library). The use of discretization operators provides a solution to the problem of low frequency tags. It is our conviction that some essential biological information might be derived from the mass of the SAGE data.
We designed a relational database for storing the data available on the NCBI site. From such a database, we have built two gene expression matrices, the so-called 74×822 and 90×12636 matrices.
The construction of the 74×822 matrix is described in [2]. It records the expression level, as of June 2001, for 822 genes belonging to the minimal transcriptome set [23]. After having extracted biologically relevant information from the 74×822 matrix for which the direct extraction worked

[4] www.ncbi.nlm.nih.gov/SAGE/index.cgi

quite well, we decided to build the most exhaustive SAGE expression matrix,i.e., to the best of our knowledge, an original contribution to human SAGE data analysis. From the human libraries available in December 2002. We selected those with more than 20 000 tag sequences and we eliminated the tag sequences for which the identification was ambiguous, based on the SAGE map file [5]. Numerous tags are present only once in a library. Nowadays, it is difficult to evaluate whether these tags represent some real genes or correspond to sequencing errors. We therefore only kept the tags appearing at least twice in at least one library. It as provided a matrix of 90 libraries and 27 679 tags. Using simple statistics (see [3]), we have been able to remove 15 043 tags. In the end, we produced a matrix recording the expression level of 90 libraries and 12 636 tags.

We have chosen to extract the information concerning the over expression of genes: the true value (1) for one given library and one given gene indicates the over expression of this gene in this library. On the contrary, a false value (0) indicates that this gene is not over expressed in this library. We have studied four discretization techniques:

- "ENE" for "Expressed or not". We assign the value 1 when the tag is present (whatever is its value) in the library, 0 otherwise.
- "Mid-Ranged". The highest and lowest expression values in the library are identified for each tag and the mid-range value is defined as being equidistant from these two numbers (arithmetic mean). Then, for a given tag, all expression values that are strictly above the mid-range value give rise to value 1, 0 otherwise.
- "Max - X% Max". The cut off is fixed w.r.t. the maximal expression value observed for each tag. From this value, we deduce a percentage X of this value, 25% in our experiments to decide for over expression.
- "X% Max". For each tag, we consider libraries in which its level of expression is in X% of the highest values (5% in our experiments). These tags are assigned to value 1, 0 for the others.

These different discretization procedures give rise to boolean matrices with varying densities (number of true values on the number of values, see Columns 2 in Table 1). It estimates the difficulty of the extractions. From a qualitative point of view, there is no good discretization method. The impact of the discretization on the validity/interestingness of the extracted regularities must be studied in each particular context. A typical mining task is then to look at $\tau_C(\beta_i(ED)) \cap \tau_C(\beta_j(ED))$, i.e., looking at the similar sets of genes that have been found by two different discretization operators. Indeed, many useful tasks will also involve other set operations between the pattern collections.

We have used the `mv-miner` prototype developed by F. Rioult with an absolute frequency threshold of 1. In that context, it provides each free set on the columns, its frequency, its closure (i.e., a closed set on the columns) and its associated closed sets w.r.t. the lines (Pentium 800MHz with RAM 4GB and 3GB for swap, linux operating system). We have compared the extraction performances not only between the large and the smaller matrices but also across the different discretizations. The

[5] ftp://ftp.ncbi.nih.gov/pub/sage/map/Hs/NlaIII/

	Discretization	Density	Nb free sets	Nb closed sets
M1	ENE	82.8	intractable	intractable
^{t}M1	ENE	82.8	intractable	intractable
M1	Mid-Ranged	12.2	13 580 544	80 068
^{t}M1	Mid-Ranged	12.2	209 829	80 068
M1	Max - 25% Max	3.8	35 934	1 386
^{t}M1	Max - 25% Max	3.8	3 211	1 386
M1	5% Max	4.8	72 630	1 808
^{t}M1	5% Max	4.8	3 362	1 808

	Discretization	Density	Nb free sets	Nb closed sets
M2	ENE	34.5	intractable	intractable
^{t}M2	ENE	34.5	intractable	intractable
M2	Mid-Ranged	4.8	intractable	intractable
^{t}M2	Mid-Ranged	4.8	324 565	196 130
M2	Max - 25% Max	2.2	intractable	intractable
^{t}M2	Max - 25% Max	2.2	21 603	9 150
M2	5% Max	4.7	intractable	intractable
^{t}M2	5% Max	4.7	54 762	31 766

Table 1. Results for $M1 = 74 \times 822$ and $M2 = 90 \times 12\ 636$

results on the boolean matrices derived from 74×822 (resp. $90 \times 12\ 636$) are in Table 1.
In these contexts, one database scan does not cost too much and extraction time is clearly related to the number of generated candidates. Only the numbers of free sets (with frequency $\neq 0$) and closed sets in both a boolean matrix and its transposed matrix are given here. Results are very interesting. Intractability in very high density matrices is understandable. In every case for the larger matrices, extraction becomes feasible by working on the transposed matrix. For instance, using the Mid-Ranged discretization on the large $90 \times 12\ 636$ matrix, the process has failed after more than 17 hours of computations while it has taken less than 1 minute on its transposed version. Others preliminary experiments on microarray data [22] have confirmed the added-value of the approach.

5 Conclusion

We are studying an inductive database approach to gene expression data analysis. Among others, it enforces us to think in terms of primitive constraints and efficient constraint-based mining techniques. Efficiency is needed not only for tractability but also for supporting the dynamic aspects of knowledge discovery (interactivity with the biologists).
We have identified a small set of primitives that are quite useful for real gene expression data analysis. Expressing analysis process by means of a sequence of queries is also important for optimizing the computations

when, e.g., new expression data is available (a new evaluation of a potentially complex process from a new initial expression matrix).
Surprisingly, the pathological dimensions of the expression matrices that were for us a bottleneck a few months ago, enable now to extract every concept in real data. It comes from the combination of an efficient algorithm for computing the closed sets from the free sets and the use of the Galois connection properties.
We are currently carrying the experimentation on the large SAGE matrix to extract biologically meaningful groups of co-regulated genes and their associated sets of biological situations. This should result in more biologically interesting findings than in [2] since the large matrix records the expression level of far more genes, and therefore of far more particular genes of special interest to a given biologist.

Acknowledgements. This work has been partially funded by the EU contract cInQ IST-2000-26469 (FET arm of the IST programme) and a French inter-EPST Bioinformatic program (2002-2003) through which Céline Robardet is supported. François Rioult is supported by the IRM department of the Caen CHU, the comité de la Manche de la Ligue contre le Cancer and the Conseil Régional de Basse-Normandie. Sylvain Blachon is supported by the comité de Saône et Loire de la Ligue contre le Cancer.

References

1. Y. Bastide, R. Taouil, N. Pasquier, G. Stumme, and L. Lakhal. Mining frequent patterns with counting inference. *SIGKDD Explorations*, 2(2):66 – 75, Dec. 2000.
2. C. Becquet, S. Blachon, B. Jeudy, J.-F. Boulicaut, and O. Gandrillon. Strong association rule mining for large gene expression data analysis: a case study on human SAGE data. *Genome Biology*, 12, 2002.
3. S. Blachon, C. Robardet, J-F. Boulicaut, and O. Gandrillon. Extraction de régularités dans des données d'expression SAGE humaines. In *Proceedings Informatique et analyse du transcriptome JPGD'03*, Lyon, F, May 2003. In French.
4. J.-F. Boulicaut. Inductive databases and multiple uses of frequent itemsets: the cInQ approach. In *Database Support for Data Mining Application*, Rosa Meo et al. Eds. Springer-Verlag LNCS 2682, In Press. To appear.
5. J.-F. Boulicaut, A. Bykowski, and C. Rigotti. Approximation of frequency queries by mean of free-sets. In *Proceedings PKDD'00*, volume 1910 of *LNAI*, pages 75–85, Lyon, F, Sept. 2000. Springer-Verlag.
6. J.-F. Boulicaut, M. Klemettinen, and H. Mannila. Modeling KDD processes within the inductive database framework. In *Proceedings DaWaK'99*, volume 1676 of *LNCS*, pages 293–302, Florence, I, Sept. 1999. Springer-Verlag.
7. C. Bucila, J. Gehrke, D. Kifer, and W. White. DualMiner: A dual pruning algorithm for itemsets with constraints. *Data Mining and Knowledge Discovery* 7(3):241–272, 2003.

8. A. Bykowski. *Condensed representations of frequent sets: application to descriptive pattern discovery.* PhD thesis, INSA Lyon, F-69621 Villeurbanne cedex, France, Oct. 2002.
9. T. Calders and B. Goethals. Mining all non derivable frequent itemsets. In *Proceedings PKDD'02*, volume 2431 of *LNAI*, pages 74–83, Helsinki, FIN, Aug. 2002. Springer-Verlag.
10. L. De Raedt. A perspective on inductive databases. *SIGKDD Explorations*, 4(2):69–77, January 2003.
11. L. De Raedt and S. Kramer. The levelwise version space algorithm and its application to molecular fragment finding. In *Proceedings IJCAI'01*, pages 853 – 862, Seattle, USA, Aug. 2001. Morgan Kaufmann.
12. J. L. DeRisi, V. R. Iyer, and P. O. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278, 1997.
13. M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings National Academy of Science USA*, 95:14863–14868, 1998.
14. T. Imielinski and H. Mannila. A database perspective on knowledge discovery. *CACM*, 39(11):58–64, Nov. 1996.
15. B. Jeudy and J.-F. Boulicaut. Optimization of association rule mining queries. *Intelligent Data Analysis*, 6(4):341 – 357, 2002.
16. H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery journal*, 1(3):241–258, 1997.
17. R. Ng, L. V. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proceedings of ACM SIGMOD'98*, pages 13–24, Seattle, USA, May 1998. ACM Press.
18. R. Ng, J. Sander, and M. Sleumer. Hierarchical cluster analysis of sage data for cancer profiling. In *Proceedings BIOKDD'01 co-located with ACM SIGKDD'01*, San Francisco, USA, Aug. 2001.
19. C. Niehrs and N. Pollet. Synexpression groups in eukaryotes. *Nature*, 402:483–487, 1999.
20. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1):25–46, Jan. 1999.
21. J. Pei, J. Han, and R. Mao. CLOSET an efficient algorithm for mining frequent closed itemsets. In *Proceedings SIGMOD Workshop DMKD'00*, Dallas, USA, May 2000.
22. F. Rioult, J.-F. Boulicaut, B. Crémilleux, and J. Besson. Using transposition for pattern discovery from microarray data. In *Proceedings SIGMOD Workshop DMKD'03*, pages 73–79, San Diego, USA, June 2003.
23. V. Velculescu, L. Zhang, B. Vogelstein, and K. Kinzler. Serial analysis of gene expression. *Science*, 270:484–487, 1995.
24. R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In *Ordered sets*, pages 445–470. Reidel, 1982.
25. M. J. Zaki. Generating non-redundant association rules. In *Proceedings SIGKDD'00*, pages 34 – 43, Boston, USA, Aug. 2000. ACM Press.