# The Model of Most Informative Patterns and Its Application to Knowledge Extraction from Graph Databases

Frédéric Pennerath[1,2] and Amedeo Napoli[2]

[1] Supélec, Campus de Metz, 2 rue Édouard Belin 57070 Metz, France
frederic.pennerath@supelec.fr
[2] Orpailleur team, LORIA, BP 239, 54506 Vandoeuvre-lès-Nancy Cedex, France
amedeo.napoli@loria.fr

**Abstract.** This article introduces the class of Most Informative Patterns (MIPs) for characterizing a given dataset. MIPs form a reduced subset of non redundant closed patterns that are extracted from data thanks to a scoring function depending on domain knowledge. Accordingly, MIPs are designed for providing experts good insights on the content of datasets during data analysis. The article presents the model of MIPs and their formal properties wrt other kinds of patterns. Then, two algorithms for extracting MIPs are detailed: the first directly searches for MIPs in a dataset while the second screens MIPs from frequent patterns. The efficiencies of both algorithms are compared when applied to reference datasets. Finally the application of MIPs to labelled graphs, here molecular graphs, is discussed.

## 1 Introduction

Given a dataset describing objects by attributes (or items), a frequent itemset is a subset of attributes such that the number, also called support or frequency, of objects presenting all of these attributes is not less than some threshold. Since the first frequent itemset mining algorithm was proposed [1], frequent itemsets have become a major and prolific model in data-mining that has served many different applications and has been generalized to many different classes of patterns, like sequences, trees, or connected graphs (see for instance the Gaston algorithm [2] later used in Sect. 4.2). However searching frequent patterns is not an ultimate objective. Frequent patterns (of any type, even graphs) are generally considered as the result of an intermediate processing step, usually followed either by the extraction of frequent association rules, or by the extraction of a set of patterns of interest wrt some application specific criteria. In any case, resulting rules or patterns are usually sorted in decreasing order of some score so that only the head of the sorted list, whose members are sometimes called top-k patterns (like area-scored top-k patterns [3] later referred in Sect. 2.1), is considered.

For association rules, many scores are available like confidence or lift. For frequent patterns, scoring often serves supervised classification problems. Scores

W. Buntine et al. (Eds.): ECML PKDD 2009, Part II, LNAI 5782, pp. 205–220, 2009.

like p-value or information gain are then used to assess the discriminative power of patterns relatively to two sets of positive and negative examples. Whereas a direct scoring of patterns may make sense in the framework of machine learning problems, practical relevance of pattern scoring might be discussed in the framework of knowledge discovery, where selected rules or patterns are directly analyzed by experts. In that case two problems occur when providing experts lists of patterns sorted by decreasing order of score.

First finding a good qualitative scoring function is not an easy task in the context of knowledge discovery as scoring must predict interest of experts for patterns. This interest is typically the amount of novel information a pattern brings to experts relatively to their current state of knowledge but this information is obviously hardly assessable. Frequency is an example of a "bad" qualitative scoring function. Because of the anti-monotonic property of frequency, most frequent patterns tend to be the smallest and thus the least informative as well. An extreme example is the empty itemset that carries no information but has the largest possible frequency. However a good scoring function must somehow integrate frequency as the latter reflects likelihood of patterns, from highly improbable to very common. In many applications, the interest of a pattern thus balances between its frequency and the amount of information contained in its structure. Such a balance refers to the notion of data representativeness. The Minimal Description Length principle (MDL) provides a theoretical foundation to assess representativeness. This principle states the better a model helps to encode data with a reversible compression scheme, the more this model is representative of data. This principle has already been used to identify patterns representative of data. Data compression then consists in replacing every occurrence of these representative patterns by new attributes in datasets of attributes [4] or new vertices in datasets of graphs [5]. However MDL-based patterns are limited somehow as they do not take easily into account what experts know and want to know. A better solution is to provide a flexible model that accepts a large family of scoring functions tunable to experts' needs.

The second problem is information redundancy among extracted patterns: Since usual scoring functions are continuous, similar patterns are likely to have similar scores. Consequently top-k patterns gets saturated by patterns similar to the pattern of highest score, especially when patterns like graphs exhibit a high combinatorial power. In practice experts experience difficulties to distinguish patterns providing them new elements of information as they are flooded with redundant copies of already analyzed patterns. One way of reducing the number of useless frequent patterns to consider might consist in introducing additional constraints that patterns have to meet [6]. A common example of pattern constraints is provided by closed patterns: a pattern $P$ is *closed* if the frequency of every pattern containing $P$ is strictly smaller than the frequency of $P$. However, although constraints might reduce the number of patterns, they remain insensitive to pattern redundancy.

In this paper we propose to solve both previous problems by a pattern selection process that outputs a family of patterns we have called *Most Informative*

*Patterns* or MIPs. Intuitively MIPs are defined as local maxima of a scoring function. This function is only required to satisfy few conditions in order to assess pattern representativeness. The objective of MIP model is that every MIP reveals one independent element of interest for experts. In practice MIPs appear in limited number and are not structurally redundant compared to other pattern families so that experts can directly analyse them. The idea underlying the MIP model was initially motivated by a selective extraction of patterns from chemical reaction databases [7]. Contributions of this article are the generalization of this idea into a broad and formal model, the derivation of properties from the model, and the introduction, comparison, and application of two methods to extract frequent MIPs from itemset and graph datasets. To this end, the MIP model and its properties are introduced in Sect. 2, the MIP extraction methods in Sect. 3, and experiments in Sect. 4.

## 2 Introduction of Most Informative Patterns

### 2.1 An Example

In order to illustrate the redundancy problem, let consider the simple example of a dataset containing seven objects described by four attributes from $a$ to $d$ and whose descriptions are respectively $a$, $b$, $ab$, $cd$, $abc$, $abd$, and $abcd$. Let assume experts decide to score itemsets with the product of their length and their frequency (a MDL-related score sometimes called area function [3]). Figure 1 displays resulting frequency and score of every pattern inside the order diagram of itemsets ordered by subset inclusion. The list of itemsets sorted in decreasing order of score is: $ab$ (score of 8); $abc$ and $abd$ (6); $a$ and $b$ (5); $abcd$, $ac$, $bc$, $ad$, $bd$, and $cd$ (4); $acd$, $bcd$, $c$, and $d$ (3); $\emptyset$ (0). When picking patterns from this list in that order, experts might ignore $abc$, $abd$, $a$, and $b$ as these patterns are structurally similar to $ab$ but with a lower score. For the same reason of redundancy, experts might ignore $abcd$, $ac$, $bc$ not as interesting as $abc$, then $ad$ and

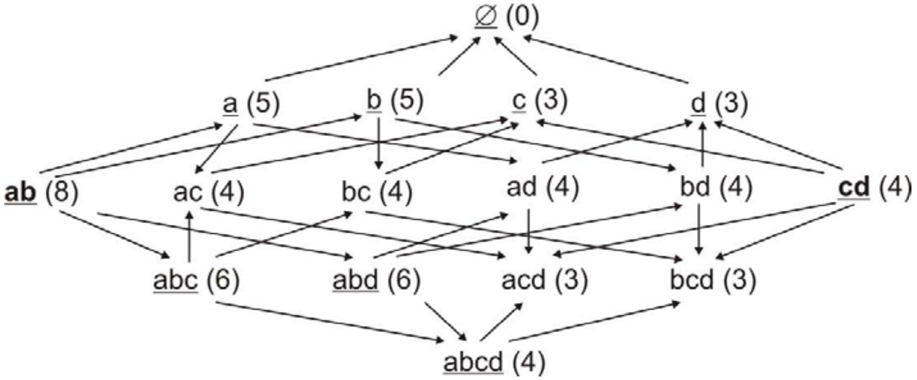


**Fig. 1.** Diagram order of itemsets. Every itemset is labeled with ($s = f \times l$) where $s$, $f$ and $l$ are resp. its score, frequency and length. Closed patterns are underlined.

*bd* not as good as *abd*. However experts might consider next pattern *cd* that has a higher score than those of all similar patterns *acd*, *bcd*, *c* or *d*. Finally all remaining patterns are ignored as they are similar to patterns with better scores. The fact that *cd* is retained whereas its score is lower than those of many ignored patterns illustrates that scoring by itself is a limited approach. Introducing constraints may focus the analysis on a limited number of patterns of a particular type but does not remove pattern redundancy and may discard interesting patterns as well. For instance considering only closed patterns (underlined on Fig. 1) keeps redundant patterns like *a*, *b*, *ab*, *abc*, *abd*, *abcd*, whereas keeping only patterns containing item *a* removes the interesting pattern *cd*. MIPs formalize the screening process described on the previous example.

### 2.2 MIP Definition

Formally let consider a set $\mathcal{P}$ of patterns, ordered by a partial ordering relation $\leq_{\mathcal{P}}$. A *dataset* $\mathcal{D}$ of *objects* is then described by a function $d : \mathcal{D} \rightarrow \mathcal{P}$ mapping every object $o \in \mathcal{D}$ to its *description* $d(o) \in \mathcal{P}$. A pattern $P \in \mathcal{P}$ is said to *describe* an object $o$ if $P \leq_{\mathcal{P}} d(o)$. The *support* or *frequency* of a pattern $M$ is then the number $\sigma(P)$ of objects of $\mathcal{D}$ described by $P$ whereas the *relative frequency* $\sigma_r(P)$ is the fraction of $\sigma(P)$ over the size $|\mathcal{D}|$ of the dataset. Support and relative frequency are non-increasing functions in the *pattern order* $(\mathcal{P}, \leq_{\mathcal{P}})$: the smaller a pattern is, the more objects it describes. In addition, the pattern order is assumed to contain a smallest pattern, called the *empty pattern* and denoted $\emptyset_{\mathcal{P}}$. One of the simplest examples of pattern order is the power set $\mathcal{P} = \mathfrak{P}(\mathcal{A})$ of a set $\mathcal{A}$ of attributes ordered by the subset inclusion relation $\leq_{\mathcal{P}} = \subseteq$, the empty pattern being the empty set. Another example of pattern order is the set of non-isomorphic connected graphs whose vertices and edges are tagged by labels taken from an arbitrary set $\mathcal{L}$. The ordering relation is then the isomorphic subgraph relation and the empty pattern is the empty graph.

As mentioned previously, the model of most informative patterns integrates a scoring function to assess the interest or relevance of a pattern. However only some functions are of interest to score patterns representative of data. The family of those so-called *informative scoring functions* is defined as follows.

**Definition 1.** *Given a dataset* $\mathcal{D}$ *described by patterns from order* $(\mathcal{P}, \leq_{\mathcal{P}})$*, a* scoring function *is a function* $s : \mathcal{P} \times [0; 1] \rightarrow \$$ *mapping a pattern* $P$ *of relative frequency* $\sigma_r(P)$ *in* $\mathcal{D}$ *to a* score $s(P, \sigma_r(P))$ *whose value is taken from a set* $\$$ *ordered by a partial ordering relation* $\leq_{\$}$*. A scoring function* $s$ *is said* informative *if following statements hold for* $s$*:*

1. *For every non-empty pattern* $P$*, partial function* $s^P : f \mapsto s(P, f)$ *is a strictly increasing function of* $f \in [0; 1]$*:*

$$\forall P \in \mathcal{P} \setminus \{\emptyset_{\mathcal{P}}\}, \forall (f_1, f_2) \in [0; 1]^2, f_1 < f_2 \Rightarrow s^P(f_1) <_{\$} s^P(f_2)$$

2. *For every non-null real number* $f \in ]0; 1]$*, partial function* $s^f : P \mapsto s(P, f)$ *is a strictly increasing function of* $P \in \mathcal{P}$*:*

$$\forall f \in ]0; 1], \forall (P_1, P_2) \in \mathcal{P}^2, P_1 <_{\mathcal{P}} P_2 \Rightarrow s^f(P_1) <_{\$} s^f(P_2)$$

3. *A pattern of zero frequency can never get a higher score than a pattern of non-zero frequency:*

$$\forall (P_1, P_2) \in \mathcal{P}^2, \nexists f > 0, s(P_1, f) <_{\$} s(P_2, 0)$$

The already used *area function* $s_a : (P, f) \mapsto |P| \cdot f$ is an example meeting all requirements of an informative function. This function may be interpreted wrt the MDL principle as an estimation of the amount of compressed space when replacing every occurrence of $P$ by a new special symbol (attribute or vertex) [5]. In section 4, we propose to extend this area function by weighting attributes of an itemset or vertex/edge labels of a graph pattern with variable gains of information. The definition of the resulting scoring function is given for graphs (itemsets being equivalent to a graph whose isolated vertices have attributes as labels):

**Definition 2.** *The* information function $s_i$ *is defined as:*

$$s_i : (g, \sigma_r) \mapsto I(g) \cdot \sigma_r$$

*where the factor $I(g)$ of information related to graph pattern $g$ is the sum of information carried by every vertex $v \in V(g)$ of label $l_v(v)$ and every edge $e \in E(g)$ of label $l_e(e)$:*

$$I(g) = \sum_{v \in V(g)} i(l_v(v)) + \sum_{e \in E(g)} i(l_e(e))$$

*Quantity of information associated to a vertex or edge label is in turn:*

$$i(l) = -\log_2 \left( \frac{n(l)}{\sum_{l' \in \mathcal{L}} n(l')} \right)$$

*where $n(l)$ is the number of vertices or edges in $\mathcal{D}$ carrying label $l$.*

However many other informative functions can be considered here. In particular experts can complement or replace the previous factor $I(g)$ by other terms that grow with the pattern: number of vertices, edges and cycles of a given type, number of subgraphs isomorphic to some specific patterns, maximal degree, maximal length of paths or cycles.

Now the definition of MIPs formalizes the selection process described in the introductory example:

**Definition 3.** *Given a pattern order $(\mathcal{P}, \leq_{\mathcal{P}})$, a dataset $\mathcal{D}$ described by the previous set of patterns and an informative scoring function $s$ defined on top of $\mathcal{D}$ and of scoring order $(\$, \leq_{\$})$,*

- *A pattern $P'$ is a* neighbour *of pattern $P$ if $P'$ is an immediate predecessor or successor of $P$ wrt pattern order $(\mathcal{P}, \leq_{\mathcal{P}})$, i.e. $P$ and $P'$ are comparable and no other pattern exists between $P$ and $P'$.*

- *A pattern $P' \in \mathcal{P}$* dominates *pattern $P \in \mathcal{P}$ if $P'$ is a neighbour of $P$ in $(\mathcal{P}, \leq_{\mathcal{P}})$ and scores of $P$ and $P'$ are comparable and verify $s(P', \sigma_r(P')) >_{\$} s(P, \sigma_r(P))$.*
- *A pattern $P$ is a MIP if frequency $\sigma_r(P)$ of $P$ is not null and if no pattern dominates $P$.*

Figure 2 represents diagram of Fig. 1 whose edges have been oriented according to the dominance relation: an arc drawn from $m_1$ to $m_2$ means $m_1$ dominates $m_2$ (rel. to $s_a$). Itemset *abc* is thus dominated by *ab* and dominates *ac*, *bc*, and *abcd*. Most informative patterns, in bold, are those pointed by no arc: they are as expected *ab* of score 8 and *cd* of score 4. The extraction of frequent MIPs consists in finding in dataset $\mathcal{D}$ scores and frequencies of all MIPs whose frequency is not less than some threshold $\sigma_{min}$. It is noticeable that a frequent pattern $P$ may not be dominated by any immediate predecessor and any frequent immediate successor while being dominated by a non-frequent immediate successor. For instance, patterns $c$ and $d$ of Fig. 1 are frequent for $\sigma_{min} = 3$, but are not MIPs as they are dominated by the non-frequent pattern *cd*. Now some properties can be inferred from definitions of informative scoring functions and MIPs.

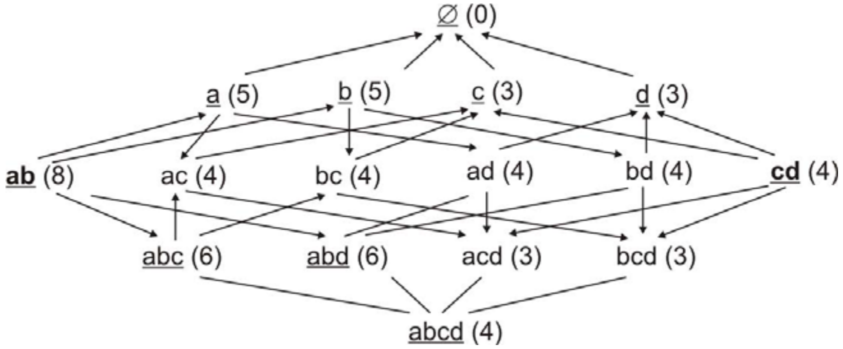


**Fig. 2.** Dominance relation between patterns from example of Fig.1. MIPs are bold.

### 2.3 Properties

Let first assume the considered pattern order verifies the so-called "finiteness hypothesis": for every finite and non-empty dataset, the number of patterns of non-null frequency is finite and non-null. This is true for standard pattern orders like the sets of finite itemsets or finite graphs. This hypothesis allows to prove the following property:

**Property 1.** *The subset of most informative patterns of a finite non-empty dataset is not empty.*

*Proof.* This can be proved by contradiction. If every pattern is dominated by at least one pattern, it is possible to build recursively a sequence of patterns $(P_i)_{i \geq 1}$ from a pattern $P_1$ of positive frequency, such that for every index $i \geq 1$, $P_{i+1}$ dominates $P_i$. Thanks to the third statement of def. 1, all those patterns have a positive frequency and thus, according to the finiteness hypothesis, build a subset of a finite set of patterns. Sequence $(P_i)$ is thus finite and contains a cycle, contradicting the fact $(s(P_i))_{i \geq 1}$ is a strictly increasing sequence of scores.

However the extraction of frequent MIPs may produce no patterns if the threshold $\sigma_{min}$ is too high. Another important property is related to closed patterns:

**Property 2.** *Every most informative pattern is a closed pattern.*

*Proof.* Let $P'$ be a MIP relative to an informative scoring function $s$. If $P'$ is not closed, there exists an immediate successor $P''$ of $P'$ such that $\sigma_r(P'') = \sigma_r(P')$. Since $f = \sigma_r(P') \neq 0$, the second statement of def. 1 applies so that function $s^f : P \mapsto s(P, f)$ is strictly increasing. Since $P' <_{\mathcal{P}} P''$, $s^f(P') <_{\$} s^f(P'')$, and thus $s(P') <_{\$} s^f(P'')$. Because $\sigma_r(P'') = \sigma_r(P')$, $s^f(P'') = s(P'')$ and $s(P') <_{\$} s(P'')$. Domination of $P''$ over $P'$ would contradict the hypothesis $P'$ is a MIP.

On the example of Fig. 2, MIPs *abc* and *cd* appear to be closed. Conversely closed patterns can be seen as a particular case of MIPs:

**Property 3.** *Closed patterns are the most informative patterns relative to the informative scoring function equal to the identity* $Id : (P, f) \mapsto (P, f)$ *and the scoring order equal to the product order* $(\mathcal{P}, \leq_{\mathcal{P}}) \times ([0; 1], \leq)$[1].

*Proof.* Given a MIP $P$, let assume $P$ is not closed. At least one immediate successor $P'$ would have the same frequency as $P$ and since by definition $P <_{\mathcal{P}} P'$, $P'$ would dominate $P$ according to the definition of product order, contradicting the initial hypothesis. Conversely a closed pattern has a higher frequency than every immediate successor and is larger (rel. to $<_{\mathcal{P}}$) than every immediate predecessor, so that it cannot be dominated and thus is a MIP.

Both properties 2 and 3 prove together that closed patterns build the least restrictive family of most informative patterns (and thus the largest as well) among every possible choice of informative scoring functions.

## 3 Extraction of Frequent Most Informative Patterns

We propose two distinct approaches to extract frequent MIPs. The first one is a one-step extraction of MIPs from datasets, while the second is a two-step process that screens frequent MIPs from frequent patterns.

[1] The product order $(E_1 \times E_2, \leq_{12})$ of two orders $(E_1, \leq_1)$ and $(E_2, \leq_2)$ is defined by $(x_1, x_2) \leq_{12} (y_1, y_2)$ iff $x_1 \leq_1 y_1$ and $x_2 \leq_2 y_2$.

### 3.1 Direct Extraction Method

As seen in the previous section, every arc $P_1 \rightarrow P_2$ of the diagram order of $(\mathcal{P}, \leq_{\mathcal{P}})$ connects a pattern $P_1$ to an immediate successor $P_2$ of $P_1$. Since every arc defines a possible relation of dominance, an algorithm extracting frequent MIPs must potentially look at every arc whose origin $P_1$ is frequent. Consequently the direct extraction method explores the pattern order in a DFS manner and when crossing an arc $P_1 \rightarrow P_2$, compares scores of $P_1$ and $P_2$ and if these scores are comparable and different, withdraw one of the two patterns from the set of valid MIP candidates. In order to remember which patterns are still valid candidates, it is required to maintain a *mip* flag for every frequent pattern, initialized to true. To this end, a pattern dictionary $\mathcal{T}$ is used to map a pattern $P$ to an entry $\mathcal{T}(P)$ containing the *mip* flag along with frequency and score of $P$. This dictionary uses a trie structure for storing canonical encoding of patterns. In case of itemsets, this encoding is simply the list of attributes sorted in some arbitrary order. In case of labeled connected graphs, encoding first assumes to compute a canonical ordering of vertices of this graph thanks to some state-of-the-art algorithm like Nauty [8], and then encodes the resulting canonical graph as a sequence of symbols for accessing the trie. The DFS exploration is performed thanks to a recursive function detailed on Fig. 3. This function `develop` takes a current pattern $P$ and its entry $e$ in $\mathcal{T}$ as arguments. Line 1 then computes in one single pass over $\mathcal{D}$, frequencies of the set $S$ of all immediate successors of $P$ occurring in $\mathcal{D}$ (i.e. of non-null frequency). This operation can be done efficiently by storing in memory all embeddings of the current pattern in dataset

```
Function develop(pattern P, entry e)
Data: Dataset $\mathcal{D}$, threshold $\sigma_{min}$, scoring function $s$ and order $(\$, \leq_{\$})$
Result: List of frequent MIPs with their scores and frequencies
1 Extract set $S = \{(P', \sigma_r(P'))\}$ of all imm. succ. $P'$ of $P$ occur. in $\mathcal{D}$ ;
  foreach $(P', \sigma'_r) \in S$ do
      if $\sigma'_r \geq \sigma_{min}$ then
          Search for entry $e'$ mapped to $P'$ in $\mathcal{T}$ ;
          if $e'$ does not exist then
              Create entry $e'$ such that $e'.score \leftarrow s(P', \sigma'_r)$, $e'.freq \leftarrow \sigma'_r$, and
              $e'.mpi \leftarrow$ true and map $P'$ to $e'$ in $\mathcal{T}$ ;
2             Call develop $(P', e')$
3         if $e.score <_{\$} e'.score$ then
              $e.mpi \leftarrow$ false
4         else if $e.score >_{\$} e'.score$ then
              $e'.mpi \leftarrow$ false
5     else if $e.score <_{\$} s(P', \sigma'_r)$ then
          $e.mpi \leftarrow$ false
```

**Fig. 3.** Recursive procedure for a direct extraction of frequent MIPs

(using data structures like tid-lists [9] for itemsets or occurrence lists [10] for connected graphs). Then line 2 calls recursively the function in order to further develop every frequent immediate successor $P'$ of $P$ that has not been explored yet (i.e. that has not already been inserted in $\mathcal{T}$). In any case, scores of $P$ and $P'$ are compared (lines 3, 4, and 5) to discard dominated patterns from the set of MIP candidates. At the end of recursion started with the empty pattern as argument, the algorithm outputs frequent MIPs as patterns contained in $\mathcal{T}$ with a true flag, along with their scores and frequencies.

### 3.2 Frequent Pattern Screening Method

Another solution is to screen frequent MIPs from frequent patterns produced by an existing frequent pattern mining algorithm. This screening processes frequent patterns level by level as a level-wise algorithm like `Apriori` [1]: level of order $n$ is the set of frequent patterns with the same length equal to $n$ (i.e. number of attributes for itemsets and number of edges for graphs). More exactly the algorithm compares the score of every frequent pattern of level $n$ with scores of their immediate predecessors of level $n-1$ for every non-empty level $n$. Comparison of scores allows to rule out i) MIP candidates of level $n$ that are dominated by at least one immediate predecessor and ii) MIP candidates of level $n-1$ that are dominated by at least one **frequent** immediate successor. This processus is called the *primary screening* as it does not exactly produce the set of frequent MIPs but only the superset of frequent MIP candidates that are not dominated by any of their immediate predecessors and frequent successors. A *secondary screening* is required to rule out MIP candidates that are dominated by at least one non-frequent immediate successor. The method is summarized on Fig. 4. It takes as input the set $\mathcal{F}$ of frequent patterns wrt threshold $\sigma_{min}$ and returns the list $\mathcal{I}$ of frequent MIPs. The idea is that lists $L_{-1}$, $L'_{-1}$, $L''_{-1}$, and $L'_0$ contain successive copies of level $l-1$ (for the three first lists) and of level $l$ (for $L'_0$), where each pattern is tagged by its *mip* flag, score and frequency. The *mip* tag

```
Data: Dataset $\mathcal{D}$, threshold $\sigma_{min}$, scoring function $s$, order $(\$, \leq_{\$})$, and list $\mathcal{F}$
      of frequent patterns with their frequencies
Result: List $\mathcal{I}$ of frequent MIPs with scores and frequencies

Partition $\mathcal{F}$ into levels $(\mathcal{F}_l)_{0 \leq l \leq k}$ of the same length $l$ ; Load $\mathcal{F}_0$ into list $L_{-1}$ ;
for $l$ from 1 to $k+1$ do
    (Clear lists $L'_{-1}$, $L'_0$, and $L''_{-1}$) ;
    if $l \leq k$ then
        Primary screening between $L_{-1}$ (lev. $l-1$) and $\mathcal{F}_l$ (lev. $l$) producing
        resp. MIP candidates in lists $L'_{-1}$ (lev. $l-1$) and $L'_0$ (lev. $l$) ;
        Rename $L'_0$ in $L_{-1}$
    Secondary screening of $L'_{-1}$ producing MIPs in list $L''_{-1}$ ;
    Append $L''_{-1}$ to $\mathcal{I}$
```

**Fig. 4.** Algorithm computing frequent MIPs by screening frequent patterns

initialized to true, may get false during primary filtering at iteration $l$ (from $\mathcal{F}_l$ to $L'_0$), during primary filtering at iter. $l+1$ (from $L'_0 = L_{-1}$ to $L'_{-1}$) or finally during secondary filtering at iter. $l+1$ (from $L'_{-1}$ to $L''_{-1}$). At that stage members of $L''_{-1}$ are necessarily MIPs and are added to $\mathcal{I}$.

Primary filtering consists i) first in loading $L_{-1}$ into a pattern dictionary $\mathcal{T}$ identical to the one used by the first algorithm (i.e. a pattern is mapped to its *mip* flag, score and frequency) ii) then for every pattern $P$ of $\mathcal{F}_l$ in computing every immediate predecessor $P'$ of $P$ and retrieving the entry of $P'$ from $\mathcal{T}$ (that necessarily exists as $P'$ is necessarily frequent) iii) in comparing scores and updating accordingly *mip* flags of $P$ and $P'$. In case of itemsets, computing immediate predecessors of $P$ consists in withdrawing any attribute of $P$ but in case of connected graphs, this requires not only to withdraw any edge of $P$ but also to ensure the resulting graph is still connected. In other words, only edges that are not bridges may be withdrawn. Bridge edges can be identified thanks to a DFS algorithm [11] of complexity linear in the number of edges of $P$.

Finally secondary filtering consists given any MIP candidate $P$ (i.e. any pattern of $L'_{-1}$ with a true *mip* flag), in computing in one pass over the dataset $\mathcal{D}$ the frequencies of all immediate successors of $P$ occurring at least once in $\mathcal{D}$. Scores of these successors are then computed one by one until one of these scores is larger than score of $P$, otherwise $P$ is output as a MIP.

## 4 Experiments

Experiments aim at answering two issues. The first is the comparison of algorithm performances on reference itemset datasets, while the second is a practical and qualitative assessment of MIP relevance on a reference graph dataset.

### 4.1 Performance Comparison

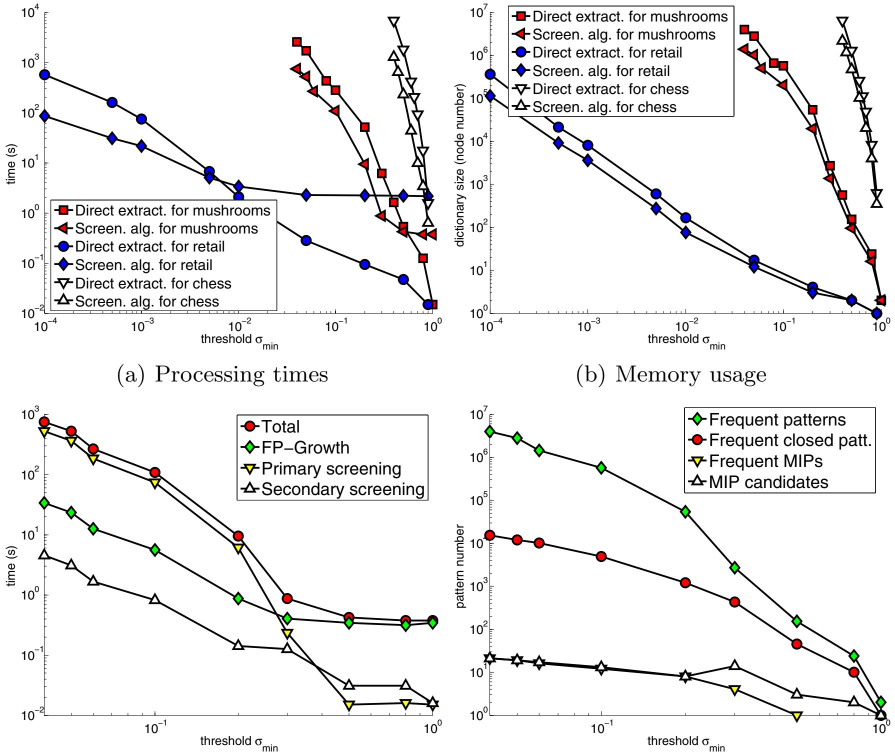
Both algorithms can be proved to be sound and complete so that they can be distinguished only by their performance and scalability. A theoretical comparison of algorithm complexity does not allow to draw conclusions as theoretical bounds mostly rely on non-assessable measures specific to datasets (like distribution of frequent patterns over levels, number of MIP candidates that have to be processed by the secondary filtering...). For this reason, this section proposes an experimental comparison of both algorithms. In order to ease comparison, tests have been performed on reference itemset datasets. Compared to other more complex pattern families like graphs, itemsets have the advantage to be simple to process so that the risk is reduced to bias performance measurement by differences of implementation quality.

For a fair comparison, time spent for searching frequent itemset has been included in the processing time of the screening process. To this end, a version[2] of

[2] The used implementation has been written by Bart Goethals and can be downloaded from http://www.adrem.ua.ac.be/g̃oethals/software/

| Dataset | Mushroom | Vote | Retail | Chess |
|---|---|---|---|---|
| Object number | 8124 | 435 | 88162 | 3196 |
| Attribute number | 119 | 17 | 16470 | 75 |
| Rel./abs. threshold $\sigma_{min}$ | 4 % / 325 | 0.2 % / 1 | 0.01 % / 9 | 40 % / 1279 |
| Total time for direct extraction (s) | 901 | 1.5 | 576 | 6810 |
| Total time for screening MIPs (s) | 747 | 6.8 | 86 | 1304 |
| includ. time used by FP-Growth (s) | 34 | 0.5 | 21 | 84 |
| N. of frequent patterns | 3.957.084 | 44,073 | 322,924 | 6,472,981 |
| N. of freq. closed patterns | 15.463 | 6478 | 229,303 | 1,366,834 |
| N. of freq. MIPs | 21 | 7 | 1045 | 2 |

**Fig. 5.** Test results for various datasets using the area function $s_a$



(a) Processing times

(b) Memory usage

(c) Proc. times of screening proc. steps

(d) Pattern numbers

**Fig. 6.** Details of the test results on log-log scales. Figures (c) and (d) focus on the Mushroom dataset.

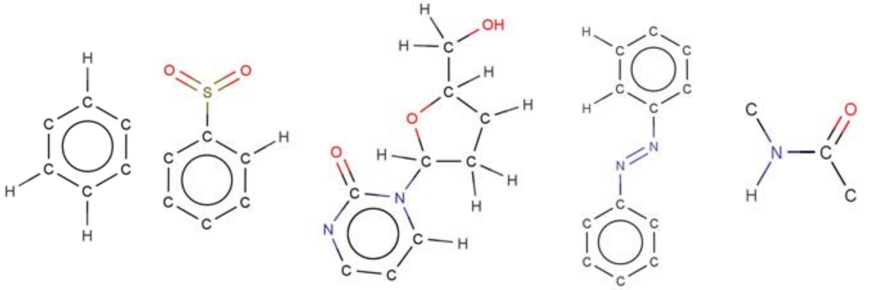
FP-growth [12] has been used as one of the most efficient itemset mining algorithms. Tests have been run on a standard laptop (single thread on Intel Core 2, 1.8 GHz) on four datasets contrasting with each other by their size, density and purpose. All datasets are from the UCI repository, except Retail that is provided by Tom Brijs [13]. Table of Fig. 5 compares processing times of both algorithms and summarizes numbers of patterns when using the area function $s_a$. The table shows frequent MIPs are very few compared to frequent patterns (ratio from 2 to 5 decades) and closed frequent patterns (2 to 3 decades) even for sparse datasets like Mushroom. In case of Chess, almost no MIPs are found as the dataset describes uncorrelated objects (i.e. winning or loosing chessboard configurations) that do not share enough common patterns to make some MIPs emerge. In comparison, datasets Mushroom, Vote or Retail describe set of objects (resp. mushrooms, senators and customers) that are likely to build families sharing common attributes and thus to provide MIPs. Concerning efficiency, the screening process appears always faster than the direct extraction, expect for small datasets like Vote. In the latter case, the larger time overhead of the screening process makes it slower for short processing time (i.e. for small datasets or large $\sigma_{min}$). This overhead can been observed on Fig. 6(a). The figure displays the evolution of processing times wrt threshold $\sigma_{min}$. It shows the screening process is always faster than the direct extraction algorithm for low values of $\sigma_{min}$, even if the performance ratio is rather small. Figure 6(b) shows the screening process requires less memory as this process only requires to store one level of frequent patterns at a time, while the direct extraction requires to store all frequent patterns. However the ratio is less than a decade, as the screening process stores levels of frequent patterns by wasting many unused intermediate nodes in the trie structure, whereas the direct extraction uses every node of the trie to store a frequent pattern. Distribution of processing time between steps of the screening process is detailed on Fig. 6(c). For small values of $\sigma_{min}$, most of the time appears to be spent on primary screening. It is interesting to observe on Fig. 6(d) that the number of MIP candidates does not necessarily increase when threshold $\sigma_{min}$ decreases, as other pattern families do. The reason is that a MIP candidate dominated by a non-frequent successor $P$ gets discarded by the primary screening as soon as $\sigma_{min}$ gets less than $\sigma_r(P)$.

### 4.2 MIP Relevance

MIPs have been used by authors to extract most informative reaction patterns from chemical reaction databases [7]. Those families of chemical reactions have shown to be characteristic of independent large families of reactions. However chemical reaction processing requires to describe too many details so that we propose a somewhat simpler application that consists in extracting MIPs from 1408 molecular graphs contained in NCI DTP AIDS antiviral active and moderately active datasets (cf dtp.nci.nih.gov/docs/aids/aids_data.html) without taking into account negative examples (i.e. the inactive dataset). MIPs are extracted in two-steps, first by mining frequent subgraphs by Gaston [2], one of the most efficient algorithms to perform this task, and then by applying the

| $L_{mips}$ | $L_{fcps}$ | $L_{fps}$ | Score | Freq. | |
|---|---|---|---|---|---|
| rank | rank | rank | $s_i(P)$ | $\sigma(P)$ | Comment |
| 1 | 1 | 1 | 76.4 | 888 | Phenyl group |
| 2 | 217 | 237 | 32.3 | 298 | Sulfonyl + phenyl groups |
| 3 | 224 | 244 | 31.8 | 401 | First fragment of carbon skeleton |
| 7 | 314 | 365 | 28 | 101 | Signif. fragm. of AIDS active mol. |
| 15 | 632 | 1344 | 24.2 | 106 | Other significant fragment |
| 53 | 1615 | 6765 | 22.3 | 116 | Azo benzene group |
| 74 | 2681 | 11528 | 21.7 | 216 | Polycyclic aromatic hydrocarbon |
| 80 | 3775 | 15046 | 21.4 | 107 | Double aromatic amine |
| 82 | 3837 | 15778 | 21.3 | 161 | Sulfonic acid + phenyl groups |
| 95 | 11799 | 38918 | 20.1 | 174 | Diol group |
| 111 | 37083 | 123812 | 17.5 | 249 | Ether group |
| 142 | 45806 | 211961 | 13.2 | 786 | Carbonyl group |
| 145 | 45950 | 213207 | 13.1 | 167 | Phenyl + amide groups |
| 152 | 47109 | 221915 | 12.1 | 270 | Amide group |
| 169 | 50985 | 237114 | 8.72 | 271 | Alkene group |
| 176 | 53288 | 241210 | 3.53 | 107 | Sulfide group |
| 177 | 53329 | 241261 | 2.2 | 211 | Imine group |
| 178 | 53333 | 241269 | 1.34 | 116 | Sodium |
| 179 | 53334 | 241270 | 1.27 | 117 | Ammonium group |

**Fig. 7.** The 19 frequent interesting MIPs and their ranks in $L_{mips}$, $L_{fcps}$, and $L_{fps}$



(a) $1^{st}$ MIP (b) $2^{nd}$ MIP (c) $7^{th}$ MIP (d) $53^{rd}$ MIP (e) $152^{nd}$ MIP

**Fig. 8.** Some of the 19 frequent interesting MIPs

screening process to frequent subgraphs wrt to function $s_i$. For a threshold $\sigma_{min} = 100$ (7 %), the number of frequent patterns, frequent closed and MIPs are respectively 262728, 53335, and 179. Test has then consisted in reproducing the same visual analysis for the three pattern families. To this end, the three set of frequent patterns, frequent closed patterns and frequent MIPs have been sorted in decreasing order of scores in three lists $L_{fps}$, $L_{fcps}$ and $L_{mips}$. For each list, the 179 first patterns (i.e. $L_{mips}$ length) have been visually analyzed: a pattern has been considered as interesting only if it brings some new obvious pieces of chemical information (mostly defined in term of functional groups and cycle configurations) compared to previous patterns of higher scores. Whereas 19 MIPs have been identified as interesting in $L_{mips}$, all patterns from rank 2 to 179 in $L_{fcps}$ and $L_{fps}$ appear to be structural variations of the pattern of rank 1 (that is also the $1^{st}$ MIP). This pattern is the phenyl ring shown on Fig. 8(a) that is common in molecules and is unrelated with the AIDS antiviral application. This shows how much frequent and even closed patterns are structurally redundant and not adapted to experts' visual analysis. In comparison the 179 MIPs provide 19 non-redundant interesting patterns described on Fig. 7. The increasing gap between the ranks of successive MIPs in $L_{fcps}$ or $L_{fps}$ gives an idea of the number of redudant patterns in those lists. Conversely a further analysis shows that each of the 1000 first closed patterns appears similar to one of the interesting MIPs. In other words, no important information appears to be lost when considering only the 19 MIPs. Some of these 19 interesting MIPs are represented on Fig. 8. The $7^{\text{th}}$ MIP (cf Fig. 8(c)) is particularly interesting as it includes very specific chemical information but still appears in 71 active molecules and 30 moderately active molecules. Other MIPs appear to have various size, frequencies, and types of atoms or bonds. In particular some MIPs appear to represent well-known functional groups, e.g. amide group on Fig. 8(e).

## 5 Related Work

Since the advent of frequent itemsets and the Apriori algorithm [1], many methods have been proposed to reduce the number of frequent patterns to a restricted subset. Their approaches vary depending on applications these methods serve, like data compression, data summarization, or supervised classification, patterns being then used as classification features. The oldest works have proposed condensed representations like closed [14] or free [15] patterns in order to reduce number of patterns. These approaches consist in replacing the set of frequent patterns along with their frequencies into an equivalent and reduced subset of patterns. Since then, this approach has been generalized to other functions than frequency [16]. However in many practical applications, the compression gain appears insufficient to allow a direct interpretation of condensed representations by experts, especially when datasets are dense. As their direct analysis is impossible, methods have proposed to summarize set of frequent patterns by clustering frequent itemsets [17] or even graphs [18]. Other approaches have recently proposed to link pattern mining to constraint programming so that user-defined constraints can easily be injected into the mining process [6]. Whereas experts may this way focus on patterns with specific structures, pattern constraints are generally insensitive to pattern redundancy.

Recent works have been proposed to address specifically the problem of reducing pattern redundancy [4,19,20,21]. Most of these approaches aim at find a reduced set of patterns that covers (i.e. subsumes) the whole dataset: for instance Siebes et al. [4] use the MDL principle to encode transactions as unions of itemsets, whereas Bringmann et al. [21] find a basis of patterns, possibly graphs, whose the various combinations (as conjunctions of patterns) may match every transaction, one by one. Similarly Hasan et al. [20] proposes to extract from a graph dataset a basis of orthogonal (i.e. non-redundant) graph patterns with a large covering of data. In order to achieve coverage of transactions, all those methods produce patterns that are not defined on an individual basis but all together as a set of interdependent patterns. This set is generally defined as an optimum relative to some global scoring function. Optimizing such a global criterion requires a large amount of processing as the search space (i.e. the power set of the set of patterns!) is huge. For this reason, a greedy heuristic algorithm is generally used to select the next best pattern to add to the set under construction. In comparison, the MIP model contrasts on several points: first the purpose of MIPs is not covering all transactions but finding significant patterns relatively to user expectation (through a scoring function). Second the MIP model addresses the redundancy problem with considerations purely based on pattern space, not on transaction coverage. Third MIPs are defined on an individual basis, and for this reason, a complete extraction without heuristics is possible for reasonable frequency thresholds.

## 6 Conclusion

MIPs provide experts a very reduced set of patterns that are representative of a dataset and are not redundant compared to other families of patterns like closed patterns. In addition the model accepts a large choice of scoring functions in order to reflect representativeness wrt to expert knowledge. The method consisting in screening frequent MIPs from frequent patterns appears more efficient and more scalable than a direct extraction even if the gain varies from significant to slight levels, depending on datasets. The model has been tested on datasets made of itemsets but also of molecular graphs, and chemical reactions. In the two latter cases, MIPs have shown to provide significant patterns (i.e. molecule fragments or reaction patterns) characteristic of distinct families of objects (i.e. families of molecules or chemical reactions). However some MIPs still appear redundant for low level of scores because of noisy variations in the scoring function. Therefore we plan as a perspective, to screen more severely patterns according to their score in order to remove these artefacts.

## References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., pp. 207–216. ACM Press, New York (1993)
2. Nijssen, S., Kok, J.N.: A quickstart in frequent structure mining can make a difference. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, pp. 647–652. ACM Press, New York (2004)
3. Soulet, A., Crémilleux, B.: Extraction des top-k motifs par approximer-et-pousser. In: Noirhomme-Fraiture, M., Venturini, G. (eds.) Extraction et gestion des connaissances (EGC 2007), Actes des cinquièmes journées Extraction et Gestion des Connaissances, Namur, Belgique. Volume RNTI-E-9 of Revue des Nouvelles Technologies de l'Information, Cépaduès-Éditions, January 23-26, vol. 2, pp. 271–282 (2007)
4. Siebes, A., Vreeken, J., van Leeuwen, M.: Item sets that compress. In: Ghosh, J., Lambert, D., Skillicorn, D.B., Srivastava, J. (eds.) SDM. SIAM, Philadelphia (2006)

5. Cook, D.J., Holder, L.B.: Substructure discovery using minimum description length and background knowledge. J. of Art. Intell. Res. 1, 231–255 (1994)
6. Raedt, L.D., Guns, T., Nijssen, S.: Constraint programming for itemset mining. In: Li, Y., Liu, B., Sarawagi, S. (eds.) KDD, pp. 204–212. ACM, New York (2008)
7. Pennerath, F., Napoli, A.: La famille des motifs les plus informatifs. application à l'extraction de graphes en chimie organique. Revue I3 8(2), 252 (2008)
8. McKay, B.D.: Practical graph isomorphism. Congr. Numer. 30, 45–87 (1981)
9. Zaki, M.J.: Scalable algorithms for association mining. IEEE T. Knowl. Data. En. 12(3), 372–390 (2000)
10. Borgelt, C., Berthold, M.R.: Mining molecular fragments: Finding relevant substructures of molecules. In: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), Maebashi City, Japan, vol. 51. IEEE Computer Society, Los Alamitos (2002)
11. Tarjan, R.E.: Depth-first search and linear graph algorithms. SIAM J. Comput. 1(2), 146–160 (1972)
12. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Min. Knowl. Discov. 8(1), 53–87 (2004)
13. Brijs, T., Swinnen, G., Vanhoof, K., Wets, G.: Using association rules for product assortment decisions: A case study. In: KDD, pp. 254–260 (1999)
14. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattices. International Journal of Information Systems 24(1), 25–46 (1999)
15. Boulicaut, J.F., Bykowski, A., Rigotti, C.: Free-sets: A condensed representation of boolean data for the approximation of frequency queries. Data Min. Knowl. Discov. 7(1), 5–22 (2003)
16. Soulet, A., Crémilleux, B.: Adequate condensed representations of patterns. Data Min. Knowl. Discov. 17(1), 94–110 (2008)
17. Yan, X., Cheng, H., Han, J., Xin, D.: Summarizing itemset patterns: a profile-based approach. In: Grossman, R., Bayardo, R.J., Bennett, K.P. (eds.) KDD, pp. 314–323. ACM, New York (2005)
18. Chen, C., Lin, C.X., Yan, X., Han, J.: On effective presentation of graph patterns: a structural representative approach. In: Shanahan, J.G., Amer-Yahia, S., Manolescu, I., Zhang, Y., Evans, D.A., Kolcz, A., Choi, K.S., Chowdhury, A. (eds.) CIKM, pp. 299–308. ACM, New York (2008)
19. Gallo, A., Bie, T.D., Cristianini, N.: Mini: Mining informative non-redundant itemsets. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 438–445. Springer, Heidelberg (2007)
20. Hasan, M.A., Chaoji, V., Salem, S., Besson, J., Zaki, M.J.: Origami: Mining representative orthogonal graph patterns. In: Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), pp. 153–162. IEEE Computer Society, Los Alamitos (2007)
21. Bringmann, B., Zimmermann, A.: One in a million: picking the right patterns. Knowledge and Information Systems (2008)