Modeling and Mining Optimal Patterns using Dynamic CSP

Willy Ugarte, Patrice Boizumault, Bruno Crémilleux, Samir Loudni

Modeling and Mining Optimal Patterns using Dynamic CSP

Willy Ugarte* Patrice Boizumault*, Bruno Crémilleux* and Samir Loudni*
*GREYC Laboratory (CNRS UMR 6072)
Université de Caen Basse-Normandie, 14032 CAEN,
{firstname.lastname}@unicaen.fr

***Abstract*—We introduce the notion of Optimal Patterns (OPs), defined as the best patterns according to a given user preference, and show that OPs encompass many data mining problems. Then, we propose a generic method based on a Dynamic Constraint Satisfaction Problem to mine OPs, and we show that any OP is characterized by a basic constraint and a set of constraints to be dynamically added. Finally, we perform an experimental study comparing our approach vs ad hoc methods on several types of OPs.**

***Keywords*-Pattern Mining, Optimisation, Dynamic CSP.**

I. Introduction

Relationships between constraint programming (CP) and data mining have recently received considerable attention [1], [2], [3]. This success likely comes from the declarative and flexible model provided by this new framework. Hypotheses and patterns that analysts seek to discover are specified in terms of constraints. Then, powerful CP solvers ensure to produce the complete set of patterns satisfying the constraints. Constraints (e.g. must-link and cannot-link in clustering, coverage in many data mining tasks) and new constraints defining a problem at hand can easily be integrated without developing specialized methods. Through its declarative and flexible nature, this CP-based approach for data mining clearly helps pattern selection strategies such as minimizing the redundancy or combining patterns on the basis of their usefulness in a given context. This approach falls into the general trend to produce sets of patterns satisfying properties on the whole set of patterns [4] which is a promising road to discover useful patterns.

There are several propositions in the literature to extract sets of patterns defined by constraints based on several patterns such as pattern teams [5], minimum tiling [6], top-k [7], conceptual clustering [8], redescription mining [2], to name a few. Even if these methods share the idea of evaluating the interest of a pattern w.r.t. a set of patterns, there are only very few attempts to design generic approaches. Even if these methods share the idea of comparing patterns between them to obtain more relevant set of patterns, there are only very few attempts to design generic approaches in this area. In this line of research, [4] provides a general definition of a two step procedure to mine constraint-based pattern sets by exploiting the analogies with local pattern mining. In the CP-based paradigm, k-pattern sets [3], [2] look for sets containing a fixed number of patterns satisfying many constraints that can be expressed in one framework. Other methods based on CP are devoted to specific sets of patterns such as top-k [9] or skypatterns [10].

This non-exhaustive list of data mining methods to extract sets of patterns shows the importance of this research activity. Many other methods for mining sets of patterns can be designed. But what kinds of sets of patterns can be produced? What about these sets of patterns w.r.t. usual pattern mining methods? This paper addresses these issues.

The key idea is to model sets of patterns thanks to the notion of preference. Then we define the optimal patterns (OPs) which are the best patterns w.r.t. the preference (an OP is a pattern for which there are no preferred patterns). A major result is that numerous data mining problems can be modeled in this framework including well-known tasks (e.g. condensed representations of patterns [11], [12], relevant subgroups [13]) but, more interestingly, the problems indicated above and many others as we will see in the rest of the paper. Taken as a whole, we think that the OP notion provides a better understanding of the different families of sets of patterns.

Second, we propose a generic method to mine OPs which is based on a dynamic CSP (Constraint Satisfaction Problems) framework. The main principle is as follows: when a solution is found, a constraint is dynamically added to the current CSP to find a better solution and then successively reduce the search space. The process stops when no better solution can be found. Finally, experiments show that our approach competes well with ad hoc methods despite its generic modeling.

Section II introduces the notion of OPs, defined as the best patterns according to a given user preference. Section III shows that OPs encompass many data mining problems. Section IV presents our generic approach for mining OPs. Section V synthesizes the related work. Section VI describes the experimental study of several OPs and compares the performance obtained over ad hoc methods.

II. Definitions

A. Preliminaries

We sketch a few well-known notions from the pattern mining area. Let $\mathcal{I}$ be a set of binary *items*.

- An *itemset* x is a non-empty subset of $\mathcal{I}$, i.e. $x \in \mathcal{L}_{\mathcal{I}}=2^{\mathcal{I}}\setminus\emptyset$. A k-itemset is an itemset of cardinality k.

- A *dataset* $\mathcal{T}$ is a multiset of itemsets of $\mathcal{L}_\mathcal{I}$. An entry of $\mathcal{T}$ is called a *transaction*. Let d be the 0/1 matrix where, for each transaction $t \in \mathcal{T}$ and each item $i \in \mathcal{I}$, $(d_{t,i} = 1)$ iff $(i \in t)$.
- The *coverage* of an itemset x is the set of transactions covering x: $\mathtt{T}(x) = \{t \in \mathcal{T} \mid x \subseteq t\}$.
- There are usual measures for an itemset x:
  - *Frequency*: $\mathtt{freq}(x) = |\mathtt{T}(x)|$.
  - *Frequency w.r.t. a class* c_i: Let $\mathcal{T}_\mathrm{i}$ be the subset of transactions associated to class c_i. $\mathtt{freq}_\mathrm{i}(x) = |\{t \in \mathcal{T}_\mathrm{i} \mid x \subseteq t\}|$
  - $\mathtt{size}(x)$ is the number of items that x contains.
  - $\mathtt{area}(x) = \mathtt{freq}(x) \times \mathtt{size}(x)$.
  - $\mathtt{max}(x.val)$ (resp. $\mathtt{min}$) is the highest (resp. lowest) value of the item values of x for attribute val.
  - $\mathtt{mean}(x) = (\mathtt{min}(x.val) + \mathtt{max}(x.val))/2$.
- An itemset x is *closed for a measure* $\mathtt{m}$ iff x has no superset with the same value for $\mathtt{m}$.
- The *growth-rate* of an itemset x w.r.t. a class c_i is:
$$\mathtt{gr}_\mathrm{i}(x) = \begin{cases} 0 & \text{if } \mathtt{freq}_\mathrm{i}(x) = 0 \\ \infty & \text{if } \mathtt{freq}(x) = \mathtt{freq}_\mathrm{i}(x) \\ \frac{|\mathcal{T} \setminus \mathcal{T}_\mathrm{i}| \times \mathtt{freq}_\mathrm{i}(x)}{|\mathcal{T}_\mathrm{i}| \times (\mathtt{freq}(x) - \mathtt{freq}_\mathrm{i}(x))} & \text{otherwise} \end{cases}$$
- An itemset x is *emergent* for a class c_i iff $\mathtt{gr}_\mathrm{i}(x) \geq \psi_{gr}$, where ψ_{gr} is a threshold. An itemset x is a *Jumping Emerging Pattern* (JEP) for c_i iff $\mathtt{freq}(x) = \mathtt{freq}_\mathrm{i}(x)$.
- A *pattern set* x is a non-empty subset of $\mathcal{L}_\mathcal{I}$, i.e. $x \in 2^{\mathcal{L}_\mathcal{I}} \setminus \emptyset$.
- A *k-pattern set* x is a k-tuple of itemsets, i.e. $x \in \mathcal{L}_\mathcal{I}^k$.

B. Search Space and Local Patterns

The search space is a key feature to get a sound understanding of the different problems of sets of patterns.

Definition 1 (SEARCH SPACE).
A search space $\mathbb{S}$ is the feasible set of all possible solutions for a given problem.

Example 1. For pattern mining, $\mathbb{S}{=}\mathcal{L}_\mathcal{I}$. For pattern set mining, $\mathbb{S}{=}2^{\mathcal{L}_\mathcal{I}} \setminus \emptyset$. For k-pattern set mining, $\mathbb{S}{=}\mathcal{L}_\mathcal{I}^k$.

Definition 2 (CONSTRAINT ON A SEARCH SPACE).
A predicate $\mathtt{c} : \mathbb{S} \rightarrow \{true, false\}$ is called a constraint on a search space $\mathbb{S}$.

Definition 3 (THEORY W.R.T. A CONSTRAINT ON A SEARCH SPACE $\mathbb{S}$ [14]).
Given a search space $\mathbb{S}$, a dataset $\mathcal{T}$ and a constraint $\mathtt{c}$, the theory $\mathrm{T}h(\mathbb{S}, \mathcal{T}, \mathtt{c})$ is the set of elements of $\mathbb{S}$ satisfying $\mathtt{c}$ in $\mathcal{T}$: $\mathrm{T}h(\mathbb{S}, \mathcal{T}, \mathtt{c}) = \{x \in \mathbb{S} \mid \mathtt{c}(x)\}$

We call *local constraint* a constraint which does not require comparisons between elements of $\mathbb{S}$ to be checked.

Definition 4 (LOCAL CONSTRAINT).
Let x be an element of a search space $\mathbb{S}$. A constraint $\mathtt{c}$ on $\mathbb{S}$ is a *local constraint* iff $\mathtt{c}$ only requires x to be checked (comparisons between elements of $\mathbb{S}$ are not necessary).

C. Preference and Optimal Patterns

The key idea is to model sets of patterns thanks to the notion of preference, and to define OPs as the best patterns w.r.t. this preference. A major result is that numerous data mining problems can be modeled in this framework (see Section III).

Definition 5 (PREFERENCE ON A SEARCH SPACE $\mathbb{S}$).
A preference $\rhd$ is a strict (i.e. irreflexive) partial[1] order relation on $\mathbb{S}$. Let x and y be two elements in $\mathbb{S}$, $x \rhd y$ indicates that x is preferred to y.

Example 2. Let $\mathbb{S}{=}\mathcal{L}_\mathcal{I}$ and $x, y \in \mathbb{S}$.
a. Let $\mathtt{m} : \mathbb{S} \rightarrow \mathbb{R}$ a measure, $x \rhd y$ iff $\mathtt{m}(x) > \mathtt{m}(y)$.
b. Let M a set of measures, x (Pareto-)dominates y w.r.t. M (denoted by $x \succ_M y$) iff $\forall m \in M, m(x) \geq m(y)$ and $\exists m' \in M, m'(x) > m'(y)$. Thus, $x \rhd y$ iff $x \succ_M y$.

Definition 6 (OPTIMALITY CONSTRAINT).
Let $\mathbb{S}$ be a search space, $\rhd$ a preference on $\mathbb{S}$, $\mathtt{basic}(x)$ a basic constraint on $\mathbb{S}$, and $\mathsf{E} = \mathrm{T}h(\mathbb{S}, \mathcal{T}, \mathtt{basic})$. An optimality constraint on $\mathbb{S}$ is defined as:
$$\mathtt{c}(x) \equiv \mathtt{basic}(x) \wedge \nexists\, y_1, \ldots, y_p \in \mathsf{E} : \bigwedge_{1 \leq j \leq p} y_j \rhd x$$
$\mathtt{basic}(x)$ states that x must satisfy a basic property (e.g. to be frequent, to be closed w.r.t. a measure or a set of measures). The second part of the definition means there are no p other distinct elements $y_1, \ldots, y_p \in \mathsf{E}$ that are preferred to x w.r.t. $\rhd$.

Example 3. Let $\mathbb{S} = \mathcal{L}_\mathcal{I}$. A skypattern [15] is a non (Pareto-)dominated pattern w.r.t. a set of measures M: $\mathtt{c}(x) \equiv \mathtt{closed}_M(x) \wedge \nexists\, y \in \mathsf{E} : y \succ_M x$. The basic constraint $\mathtt{closed}_M(x)$ states that x is closed w.r.t. M to avoid redundant skypatterns (see Section III-A2).

In other words, Definition 6 means that OPs are the best patterns according to a preference. This modeling may appear simple but it is powerful for a twofold reason. First, as we will see in the next section, many pattern mining problems can be modeled in this framework. The mining of set of patterns falls into this framework due to the multiple comparisons between x and the y_j (see Def. 6). Second, this modeling is conducive to effective mining methods based on Dynamic CSP (see section IV). From now on, a pattern x is called *local* (resp. *optimal*) if x is defined w.r.t. a local (resp. optimality) constraint over $\mathbb{S}$.

III. LOCAL PATTERNS AND OPTIMAL PATTERNS

This section provides a wide range of pattern mining problems (itemsets, pattern sets and k-pattern sets) and classifies them as local ones or optimal ones.

A. Itemset Mining

This section reviews different itemset mining problems ($\mathbb{S} = \mathcal{L}_\mathcal{I}$) and classifies them as local ones or optimal ones.

1) Local Itemsets: We give two well-known examples of local itemsets ($\psi_{\mathtt{freq}}$ and $\psi_{\mathtt{area}}$ are thresholds).
a. *Frequent Itemsets*: $\mathtt{c}(x) \equiv \mathtt{freq}(x) \geq \psi_{\mathtt{freq}}$.
b. *Large Tile Mining [6]*: $\mathtt{c}(x) \equiv \mathtt{freq}(x) \times \mathtt{size}(x) \geq \psi_{\mathtt{area}}$.

[1] There are possibly two elements $x, y \in \mathbb{S}$ ($x \neq y$) such that $x \not\rhd y$ and $y \not\rhd x$.

2) Optimal Itemsets: Many itemset mining problems can be modeled as optimal itemsets even if they are not initially defined according to an optimization criterion. It illustrates the broad scope of the OPs. Due to the space limitation, these problems are only outlined.

a. *Closed Itemsets [12], Free Itemsets [11] and Maximal Itemsets.* Closed itemsets are itemsets with no specialization having the same frequency. Let $\psi_{\texttt{freq}}$ be a frequency threshold. Closed Itemsets are defined by the optimality constraint:
$\texttt{c}(x) \equiv \texttt{freq}(x) \geq \psi_{\texttt{freq}} \wedge \nexists\, y \in \mathsf{E} : y \supset x \wedge \texttt{freq}(y) = \texttt{freq}(x)$
Free and maximal itemsets are expressed by simple variants of the previous definition:
- Free ones: $\texttt{c}(x) \equiv \texttt{freq}(x) \geq \psi_{\texttt{freq}} \wedge \nexists\, y \in \mathsf{E} : y \subset x \wedge \texttt{freq}(y) = \texttt{freq}(x)$
- Maximal ones: $\texttt{c}(x) \equiv \texttt{freq}(x) \geq \psi_{\texttt{freq}} \wedge \nexists\, y \in \mathsf{E} : y \supset x$

b. *Peak Itemsets [16].* Let $d(x,y)$=$|x \backslash y| + |y \backslash x|$ be a distance, $\texttt{m}$ a measure, ϵ an integer and δ a real. A peak itemset has a high value according to $\texttt{m}$ w.r.t. those of its neighbors:
$\texttt{c}(x) \equiv \texttt{freq}(x) \geq \psi_{\texttt{freq}} \wedge \nexists\, y \in \mathsf{E} : d(x,y) \leq \delta \wedge \epsilon \times \texttt{m}(y) > \texttt{m}(x)$

c. top-k *itemsets [7].* Let $\texttt{m}$ be a measure and k an integer. The top-k itemsets is the set of the k best itemsets according to $\texttt{m}$:
$\texttt{c}(x) \equiv \texttt{freq}(x) \geq \psi_{\texttt{freq}} \wedge \nexists\, y_1, \ldots, y_k \in \mathsf{E} : \bigwedge_{1 \leq j \leq k} \texttt{m}(y_j) > \texttt{m}(x)$

d. *The N-most interesting k-itemsets [17].* Let N be an integer. The set of the N-most frequent k-itemsets is defined by the optimality constraint:
$\texttt{c}(x) \equiv \texttt{size}(x) = k \wedge \nexists\, y_1, \ldots, y_N \in \mathsf{E} : \bigwedge_{1 \leq j \leq N} \texttt{freq}(y_j) > \texttt{freq}(x)$

e. *Relevant Subgroup Discovery [13].* A relevant subgroup consists of itemsets that discriminate a class $\mathcal{T}_1$ against a class $\mathcal{T}_2$, where $\mathcal{T}_1$ and $\mathcal{T}_2$ form a partition of $\mathcal{T}$:
$$\texttt{c}(x) \equiv \quad \texttt{freq}_1(x) \geq \psi_{\texttt{freq}} \wedge \quad \nexists\, y \in \mathsf{E} : \quad (\texttt{T}_1(y) \supset \texttt{T}_1(x)) \wedge (\texttt{T}_2(y) \subset \texttt{T}_2(x)) \wedge (\texttt{T}(y) = \texttt{T}(x) \Rightarrow y \subset x)$$

f. *Pattern compression problem [18].* Let $d(x,y) = 1 - \frac{|\texttt{T}(x) \cap \texttt{T}(y)|}{|\texttt{T}(x) \cup \texttt{T}(y)|}$ be a distance, and δ a threshold. An itemset x is *representative* iff no other itemset y in the neighborhood of x ($d(x,y) \leq \delta$) is included in x. The compression problem consists in finding all representative itemsets:
$\texttt{c}(x) \equiv \texttt{closed}(x) \wedge \nexists\, y \in \mathsf{E} : d(x,y) \leq \delta \wedge y \not\subset x$

g. *Maximally informative k-itemset [19].* Let x be a k-itemset. The joint entropy of x is defined by: $H(x)$= $-\sum_{B \in \{0,1\}^k} p(x{=}B) \log p(x{=}B)$ where $p(x{=}B)$ is the joint probability of $(x{=}B)$. x is a maximally informative k-itemset (*miki*) iff any k-itemset y has a greater joint entropy ($H(y) > H(x)$). A miki is an itemset of specified size, which maximizes the joint entropy:
$\texttt{c}(x) \equiv \texttt{size}(x) = k \wedge \nexists\, y \in \mathsf{E} : H(y) > H(x)$

h. *Essential Jumping Emerging Patterns [20].* Let $\psi_{\texttt{freq}}$ be a frequency threshold. A JEP is said to be Essential iff any frequent itemset $y \subset x$ is not a JEP:
$\texttt{c}(x) \equiv (\texttt{freq}(x) = \texttt{freq}_1(x) \wedge \texttt{freq}(x) \geq \psi_{\texttt{freq}}) \wedge \nexists\, y \in \mathsf{E} : y \subset x$

i. *Skypatterns* (see Example 3, Section II-C).

B. k-Pattern Set Mining

This section reviews several k-pattern set mining problems ($\mathbb{S}$=$\mathcal{L}^k_{\mathcal{I}}$) and classifies them as local ones or optimal ones.

1) Local k-Pattern Sets: We sketch two local k-pattern sets mining problems, [3], [2] give other problem examples.

a. *Exception Rules [21].* An exception rule is defined within the context of a pair of rules as follows (where $\mathfrak{i}$ is an item such as a class value, k=2, x_1 and x_2 are patterns):
$$\texttt{c}((x_1,x_2)) \equiv \begin{cases} \texttt{freq}(x_1 \setminus x_2 \rightarrow \mathfrak{i}) \geq \psi_{\texttt{freq}} \wedge \\ \texttt{freq}(x_1 \setminus x_2) - \texttt{freq}(x_1 \setminus x_2 \rightarrow \mathfrak{i}) \leq \delta_1 \wedge \\ \texttt{freq}(x_1 \rightarrow \neg\mathfrak{i}) \leq \Psi_{\texttt{freq}} \wedge \\ \texttt{freq}(x_1) - \texttt{freq}(x_1 \rightarrow \neg\mathfrak{i}) \leq \delta_2 \end{cases}$$
Such a pair of rules is composed of a common sense rule $x_1 \backslash x_2 \rightarrow \mathfrak{i}$ and an exception rule $x_1 \rightarrow \neg\mathfrak{i}$ since usually if $x_1 \backslash x_2$ then $\mathfrak{i}$, isolating unexpected information. This definition assumes that the common sense rule has a high frequency and a rather high confidence and the exception rule has a low frequency and a very high confidence; the confidence of a rule $x_1 \rightarrow x_2$ is defined as $\texttt{freq}(x_1 \cup x_2)/\texttt{freq}(x_1)$.

b. *Conceptual Clustering [8].* The closed patterns are well-designed for clustering based on associations because a closed pattern gathers the maximum amount of similarity between a set of transactions. The standard conceptual clustering problem can then be formalized as to find k closed patterns $x_1, x_2, ..., x_k$ (i.e., clusters) covering all transactions without any overlap on these transactions.
$$\texttt{clus}(x) \equiv \bigwedge_{i \in [1..k]} \texttt{closed}(x_i) \wedge \bigcup_{i \in [1..k]} \texttt{T}(x_i) = \mathcal{T} \wedge \bigwedge_{i,j \in [1..k]} \texttt{T}(x_i) \cap \texttt{T}(x_j) = \emptyset$$
Finally, for other examples like unexpected rules, classification conflicts, synexpression groups, ..., see [2], [3].

2) Optimal k-Pattern Sets: k-pattern set mining problems can be modeled as OPs. We briefly present two of them. Other examples are for instance Redescription Mining, k-term DNF Learning and Concept Learning [2].

a. *Maximum k-Tiling [6].* The task consists of finding k tiles (i.e., blocks of 1 in a binary matrix) having the largest possible area, where the area is the total number of 1 which are part of the tile. We consider tiles associated to closed patterns.
$$\texttt{c}(x) \equiv \bigwedge_{i \in [1..k]} \texttt{closed}(x_i) \wedge \nexists\, y \in \mathsf{E}, \texttt{area}(y) > \texttt{area}(x)$$
$$\text{where } \texttt{area}(x) \equiv |\bigcup_{i \in [1..k]} \{(t,j) \mid t \in \texttt{T}(x_i) \wedge j \in x_i\}|$$

b. *Conceptual Clustering with optimization [2].* As it may exist a large number of clusterings x satisfying the constraint $\texttt{clus}(x)$, preferences can be used to discriminate between them. For example, the user may prefer clusterings with a minimum size:
$\texttt{c}(x) \equiv \texttt{clus}(x) \wedge \nexists\, y \in \mathsf{E}, \min_{j \in [1..k]} \{\texttt{freq}(y_j)\} > \min_{i \in [1..k]} \{\texttt{freq}(x_i)\}$
Other preferences (e.g. balanced clusterings) can be expressed in the same way.

C. Pattern Set Mining

This section reviews several pattern set mining problems ($\mathbb{S}$=$2^{\mathcal{L}_\mathcal{I}}\backslash\emptyset$) and classifies them as local ones or optimal ones.

1) Local Pattern Sets: As examples, we give two pattern set problems presented in [4].

a. *Frequent Pattern Sets*: $\mathtt{c}(x) \equiv \mathtt{support}(x) \geq \psi_{\mathtt{support}}$

b. *Large Pattern Sets*: $\mathtt{c}(x) \equiv \mathtt{size}(x) \geq \psi_{\mathtt{size}}$

2) Optimal Pattern Sets: The scope of the OPs encompasses the pattern set tasks. Here are two examples:

a. *Pattern Teams* [5]: Let $\Phi : 2^{\mathcal{L}_\mathcal{I}} \rightarrow \mathbb{R}$ be a quality measure. A pattern team is a set of k-itemsets s.t. there is no other pattern set y of size k having a greater quality w.r.t. Φ:

$\mathtt{c}(x) \equiv \mathtt{size}(x) = k \ \wedge \nexists\, y \in \mathsf{E}, \Phi(y) > \Phi(x)$

b. *Minimum Tiling* [6]: It consists of finding a tiling (i.e., collection of tiles) of which the area equals to the total number of ones in the dataset and consists of the minimum number of tiles:

$\mathtt{c}(x) \equiv (\mathtt{area}(x) = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}} d_{t,i}) \wedge \nexists\, y \in \mathsf{E}, \mathtt{size}(y) < \mathtt{size}(x)$

where $\mathtt{area}(x) = |\bigcup_{x_i\in x} \{(t,j) \mid t \in \mathtt{T}(x_i) \wedge j \in x_i\}|$

IV. Mining OPs using CP

This section shows how mining OPs can be modeled and solved using DynCSP [22]. The main idea is that at each step, once a solution is found, a new constraint is dynamically added to find a better solution (in the sense of $\rhd$) and reduce the search space. The process stops when no better solution can be found. Our approach is generic as it can be applied to the mining of any kind of OP. Its completeness is ensured by the completeness of the constraint solver.

Section IV-A briefly recalls the notion of DynCSP. Section IV-B shows how the mining of OPs can be modeled and solved using a DynCSP. Section IV-C is devoted to the modeling of OP examples presented in Section II. Section IV-D provides the boolean encoding of patterns.

A. Dynamic CSP

A CSP P=(X, D, C) is defined by a finite set of variables X, a set of domains D which maps every variable $x \in X$ to a finite set of values $D(x)$ and a finite set of constraints C. A *DynCSP* (Dynamic CSP) [22] is a sequence $P_1, P_2, ..., P_n$ of CSPs, where each P_{i+1} is the result of provided changes to the previous P_i. These changes may affect the variables, domains and constraints. For our approach, changes between CSP P_i and CSP P_{i+1} are only performed by adding new constraints without any removal of constraints. Additions are handled in a straightforward way with the help of filtering. Solving such a DynCSP involves solving a single CSP with additional constraints posted during search. These constraints will survive backtracking and state that next solutions should verify both the current set of constraints as well as the added ones.

B. Mining OPs using DynCSP

This section shows how mining OPs according to a preference $\rhd$ can be modeled and solved using a DynCSP. Let $\mathbb{S}$ be a search space. Consider the sequence $P_1, P_2, ..., P_n$ of CSPs where each $P_i = (\{x\}, \mathbb{S}, q_i(x))$ and

- $q_1(x) = \mathtt{basic}(x)$
- $q_{i+1}(x) = q_i(x) \wedge \phi(s_i, x)$ where s_i is the first solution of the query $q_i(x)$.

Constraints $\phi(s_i, x)$ require successively that all obtained solutions s_i are not better (w.r.t. the preference $\rhd$) than the unknown element x we are looking for (if s_i was preferred to x, x would not be an OP). Thus, at step $(i + 1)$, the constraint $\phi(s_i, x)$ will be added: $\phi(s_i, x) \equiv s_i \not\rhd x$.

Therefore, any obtained solutions s_1, s_2, ..., s_i cannot be better than x (immediate proof by induction). New dynamically added constraints $\phi(s_i, x)$ allow to reduce the search space. The extraction process stops when no better solution can be obtained, i.e there exists n such that $q_{n+1}(x)$ has no solution.

However, all mined solutions s_1, s_2, ..., s_n are not necessarily OPs according to $\rhd$. Some of them may be *intermediate* solutions that are only useful to improve the pruning of the search space. The latter (i.e. solutions s_i for which there exists s_j $(1 \leq i < j \leq n)$ such that $s_j \rhd s_i$) are removed in post-processing. Thus, the extraction is performed in two steps: (1) compute the set of solutions $\{s_1, s_2, \ldots, s_n\}$ using a DynCSP, and (2) remove all s_i that are *intermediate* solutions (not OPs).

C. Examples

We give four examples of OP mining using DynCSP: closed itemsets, skypatterns, top-k itemsets and conceptual clustering. Table I provides a summary of the other examples of OPs given in Section II. Each example is modeled using a basic constraint $\mathtt{basic}(x)$ and dynamically added constraints $\phi(s_i, x)$ (see Section II-C). For an OP defined by a preference $\rhd$, $\phi(s_i, x) \equiv s_i \not\rhd x$ states that s_i cannot be preferred to the unknown pattern x we are looking for.

1. Closed Itemsets. In this example, the closed itemsets are defined w.r.t. the measure `freq`. For closed itemsets w.r.t. any measure `m`, just replace `freq` by `m`:

- $\mathtt{basic}(x) \equiv \mathtt{freq}(x) \geq \psi_{\mathtt{freq}}$
- $\phi(s_i, x) \equiv s_i \not\rhd x \equiv (s_i \not\supset x) \vee (\mathtt{freq}(s_i) \neq \mathtt{freq}(x))$

2. Skypatterns. Let M be a set of measures. Preference $\rhd$ is the Pareto-Dominance. To avoid redundant skypatterns, the basic constraint states that x must be closed w.r.t. M:

- $\mathtt{basic}(x) \equiv \mathtt{closed}_M(x)$

$$\begin{aligned}\phi(s_i, x) &\equiv s_i \not\rhd x \ \equiv \ s_i \not\succ_M x \\ &\equiv \neg(\forall m \in M, m(s_i) \geq m(x) \wedge \exists m' \in M, m(s_i) > m(x)) \\ &\equiv \neg(\textstyle\bigwedge_{m\in M} m(s_i) \geq m(x) \wedge \bigvee_{m'\in M} m'(s_i) > m'(x)) \\ &\equiv (\textstyle\bigvee_{m\in M} m(s_i) < m(x)) \vee (\bigwedge_{m'\in M} m'(s_i) \leq m'(x)) \\ &\equiv (\bigvee_{m\in M} m(s_i)<m(x)) \vee (\bigwedge_{m'\in M} m'(s_i)=m'(x))\end{aligned}$$

3. Top-k Itemsets for a measure `m`. Each solution s_i is stored in a list S. While k itemsets are not yet mined (i.e. i<k),

Table I: Basic constraint and dynamically added constraints for each OP

	Optimal Problem	$\texttt{basic}(x)$	$\phi(s_i, x)$
$\mathcal{L}_\mathcal{I}$	Maximal Itemsets	$\texttt{freq}(x) \geq \psi_{\texttt{freq}}$	$s_i \not\supset x$
	Free Itemsets	$\texttt{freq}(x) \geq \psi_{\texttt{freq}}$	$(s_i \not\subset x) \vee (\texttt{freq}(s_i) \neq \texttt{freq}(x))$
	Peak Itemsets	$\texttt{freq}(x) \geq \psi_{\texttt{freq}}$	$(d(x, s_i) > \delta) \vee (\epsilon \times \texttt{m}(s_i) \leq \texttt{m}(x))$
	The N-most interesting k-Itemsets	$\texttt{size}(x) = k$	$\begin{cases} \texttt{freq}(x) \geq \min\limits_{s_j \in S \cup \{s_i\}} \texttt{freq}(s_j) & \text{if} \quad i \geq N \\ true & \text{otherwise} \end{cases}$
	Relevant Subgroup Discovery	$\texttt{freq}(x) \geq \psi_{\texttt{freq}}$	$\texttt{T}_1(x) \not\subset \texttt{T}_1(s_i) \vee \texttt{T}_2(x) \not\supset \texttt{T}_2(s_i) \vee$ $(\texttt{T}(s_i) = \texttt{T}(x) \wedge s_i \not\subset x)$
	Pattern compression problem	$\texttt{closed}(x)$	$(d(x, s_i) > \delta) \vee (s_i \subset x)$
	Maximally informative k-Itemsets	$\texttt{size}(x) = k$	$H(s_i) \leq H(x)$
	Essential Jumping Emerging Patterns	$(\texttt{freq}(x) = \texttt{freq}_1(x)) \quad \wedge$ $(\texttt{freq}_1(x) \geq \psi_{\texttt{freq}})$	$s_i \not\subset x$
$\mathcal{L}^k_\mathcal{I}$	Maximum k-Tiling	$\bigwedge\limits_{i \in [1..k]} \texttt{closed}(x_i)$	$\texttt{area}(s_i) \leq \texttt{area}(x)$
$2^{\mathcal{L}_\mathcal{I}}$	Pattern Teams	$\texttt{size}(x) = k$	$\Phi(s_i) \leq \Phi(x)$
	Minimum Tiling	$(\texttt{area}(x) = \sum\limits_{t \in \mathcal{T}} \sum\limits_{i \in \mathcal{I}} d_{i,t})$	$\texttt{size}(s_i) < \texttt{size}(x)$

it's too early to constrain the search i.e. $\phi(s_i, x) \equiv true$. As soon as the k-th solution s is obtained, we impose $\phi(s_i, x)$ and remove, from S, the solution with the lowest value.

- $\texttt{basic}(x) \equiv \texttt{freq}(x) \geq \psi_{\texttt{freq}}$
- $\phi(s_i, x) \equiv \begin{cases} \texttt{m}(x) \geq \min\limits_{s_j \in S \cup \{s_i\}} \{\texttt{m}(s_j)\} & \text{if} \quad i \geq k \\ true & \text{otherwise} \end{cases}$

So, $\phi(s_i, x) \equiv \neg(\bigwedge\limits_{1 \leq j \leq k} \texttt{m}(y_j) > \texttt{m}(x)) \equiv \texttt{m}(x) \geq \min\limits_{s_j \in S \cup \{s_i\}} \{\texttt{m}(s_j)\}$

4. Conceptual Clustering. Each k-pattern set x must verify the $\texttt{clus}(x)$ constraint (see Section III-B2), and the preference enforces a small difference between cluster coverings.

- $\texttt{basic}(x) \equiv \texttt{clus}(x)$

$$\begin{aligned} \phi(s_i, x) &\equiv s_i \not\triangleright x \\ &\equiv \neg(\min_{j \in [1..k]} \{\texttt{freq}((s_i)_j)\} > \min_{m \in [1..k]} \{\texttt{freq}(x_m)\}) \\ &\equiv \min_{j \in [1..k]} \{\texttt{freq}((s_i)_j)\} \leq \min_{m \in [1..k]} \{\texttt{freq}(x_m)\} \end{aligned}$$

D. Pattern encoding

We now introduce the boolean encoding of a local pattern. Let $\mathcal{I}$ be a set of items, $\mathcal{T}$ a dataset, and d the 0/1 matrix associated to $\mathcal{T}$ (for each transaction t and each item i, $(d_{t,i} = 1)$ iff $(i \in t)$). Pattern variables are set variables represented by their characteristic function with Boolean variables. [1] models an unknown pattern x and its associated dataset $\mathcal{T}$ by introducing two sets of Boolean variables:

- item variables $\{X_i \mid i \in \mathcal{I}\}$ where $(X_i{=}1)$ iff $(i \in x)$,
- transaction variables $\{T_t \mid t \in \mathcal{T}\}$ where $(T_t{=}1)$ iff $(x \subseteq t)$.

Each set of Boolean variables aims to represent the characteristic function of the unknown pattern. The relationship between x and $\mathcal{T}$ is modeled by reified constraints stating that, for each transaction t, $(T_t = 1)$ iff x is a subset of t:

$$\forall t \in \mathcal{T}, (T_t = 1) \Leftrightarrow \sum_{i \in \mathcal{I}} X_i \times (1 - d_{t,i}) = 0 \qquad (1)$$

Using this Boolean encoding, it is worth noting that some measures are easy to encode: $\texttt{freq}(x) = \sum_{t \in \mathcal{T}} T_t$ and $\texttt{size}(x) = \sum_{i \in \mathcal{I}} X_i$. So, the minimal frequency constraint $\texttt{freq}(x) \geq \theta$ (where θ is a threshold) is encoded by the constraint $\sum_{t \in \mathcal{T}} T_t \geq \theta$. In the same way, the maximal size constraint $\texttt{size}(x) \leq \alpha$ (where α is a threshold) is encoded by the constraint $\sum_{i \in \mathcal{I}} X_i \leq \alpha$.

In order to model several pattern variables, it is only needed to make for each of them a different set of boolean variables (see [2] and [3]). If we have $x_1, \ldots, x_n$ a set of pattern variables, we represent the unknown pattern x_j by its set of boolean item variables $\{X_{i,j} \mid i \in \mathcal{I}\}$ and $\{T_{t,j} \mid t \in \mathcal{T}\}$. Each unknown pattern x_j is associated to the dataset $\mathcal{T}$ in the same way as a local pattern is:

$$\forall t \in \mathcal{T}, (T_{t,j} = 1) \Leftrightarrow \sum_{i \in \mathcal{I}} X_{i,j} \times (1 - d_{t,i}) = 0 \qquad (2)$$

V. Related Work

As patterns represent "fragmented knowledge" and often there is no clear view of how these knowledge fragments interact and can be combined to give a global picture, many works in the literature propose to discover sets of patterns defined by constraints based on several patterns [5], [6], [7], [8]. In the area of pattern sets, [4] exploits the analogies with local pattern mining to design a general definition of two step constraint-based pattern set mining, nevertheless the computational side remains limited. The CP-paradigm is at the core of generic approaches to deal with k-pattern set mining [3], [2]. These methods enable to model sets of patterns in a declarative and flexible way, but they look for sets containing a fixed number of patterns. Finally, the authors of [23] have proposed a novel algebra extending relational algebras towards pattern mining. A key idea is that many data mining tasks can be more easily described with combinations of constraints and dominance relations than only constraints. Many problems of itemsets are addressed by this algebra, but some problems such as the top-k do not come under this algebra.

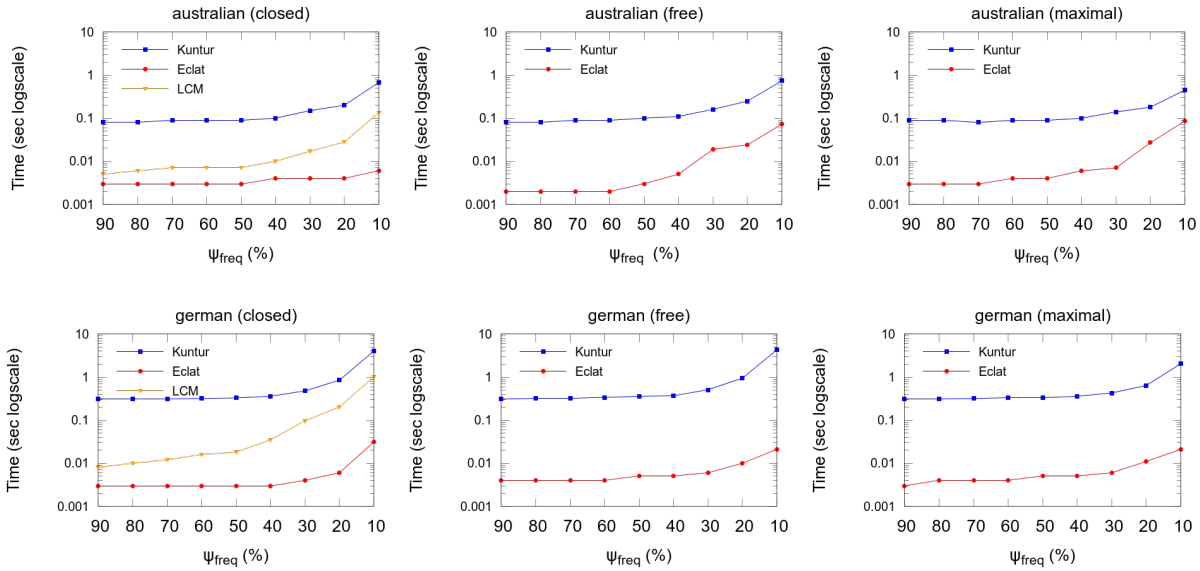

Figure 1: Comparing CPU-times (closed, free, maximal itemsets).

VI. Experiments

Our experiments focus on five types of OPs: closed/free/maximal itemsets, skypatterns, Essential JEPs, peak itemsets and conceptual clustering. For each of them, we compare our generic approach KUNTUR with one or more ad hoc methods on UCI[2] datasets. All experiments were performed under Linux with an Intel Core i3 2.13 GHz RAM 4 GB. KUNTUR was developed in GECODE[3].

Section III has provided a wide range of optimal pattern mining problems over itemsets, pattern sets and k-pattern sets. Section IV has proposed a generic method for mining OPs using DynCSP. Obviously, our approach will be less effective for "simple" patterns whose extraction has been deeply studied. But, for mining more elaborated patterns, our approach will be as competitive or more efficient than existing dedicated methods. The remainder of this section illustrates these two issues.

A. Closed, free and maximal itemsets

We compare KUNTUR with the *top* miners ECLAT [24] and LCM [25] (only for closed itemsets). The experiments were conducted by varying the frequency threshold $\psi_{\texttt{freq}}$. Figure 1 shows that ad hoc methods for these itemsets are more efficient than our generic approach. This result was expected because the research community provided 15 years of efforts on the extraction of condensed representations.

Closedness, freeness and maximality constraints are often used for modeling. Fortunately, it is possible to directly model such constraints using the boolean encoding without any loss of efficiency. In the following, we handle the example of the closedness for the freq measure. The closedness constraint ensures that a pattern has no superset with the same frequency. If item i belongs to x, it is obvious that $\texttt{freq}(x \cup \{i\}) = \texttt{freq}(x)$. Conversely, if $\texttt{freq}(x \cup \{i\}) = \texttt{freq}(x)$, then i must belong to x (if not, x would not be maximal for inclusion). $\texttt{freq}(x)$ is encoded as $\sum_{t \in \mathcal{T}} T_t$ and $\texttt{freq}(x \cup \{i\})$ is encoded as $\sum_{t \in \mathcal{T}} T_t \times d_{t,i}$. Finally, the constraint $closed(x)$ is encoded using Equation (3).

$$\forall i \in \mathcal{I}, (X_i = 1) \Leftrightarrow \sum_{t \in \mathcal{T}} T_t \times (1 - d_{t,i}) = 0 \qquad (3)$$

B. Skypatterns

We compare KUNTUR with AETHERIS [15] which is the only ad hoc approach for mining skypatterns. AETHERIS computes a condensed representation made of closed itemsets w.r.t. a given set of measures, then filter them to get the skypatterns. [10] reports experiments on 23 UCI datasets for which we considered the measures $M = \{\texttt{freq}, \texttt{max}, \texttt{area}, \texttt{mean}, \texttt{gr}_1\}$ (see Section II-A). In this paper, we only report CPU-times for the 6 datasets requiring more than 30 seconds, either for KUNTUR or AETHERIS. For the remaining 17 datasets, CPU-times are very small and quite similar for both approaches (For more details, see [10]).

The skypatterns are mined for six different sets of measures: the five subsets of 4 measures from M denoted $M_1 \ldots M_5$ and $M_6 = M$. For mean, attribute values were randomly generated in the interval $[0..1]$. Figure 2 depicts a scatter plot of CPU-times for KUNTUR and AETHERIS. Each point represents a skypattern query for one of the datasets: its x-value (log-scale) is the CPU-time for KUNTUR, its y-value (log scale) the CPU-time for AETHERIS. KUNTUR outperforms AETHERIS on many

[2] www.ics.uci.edu/~mlearn/MLRepository.html
[3] www.gecode.org/

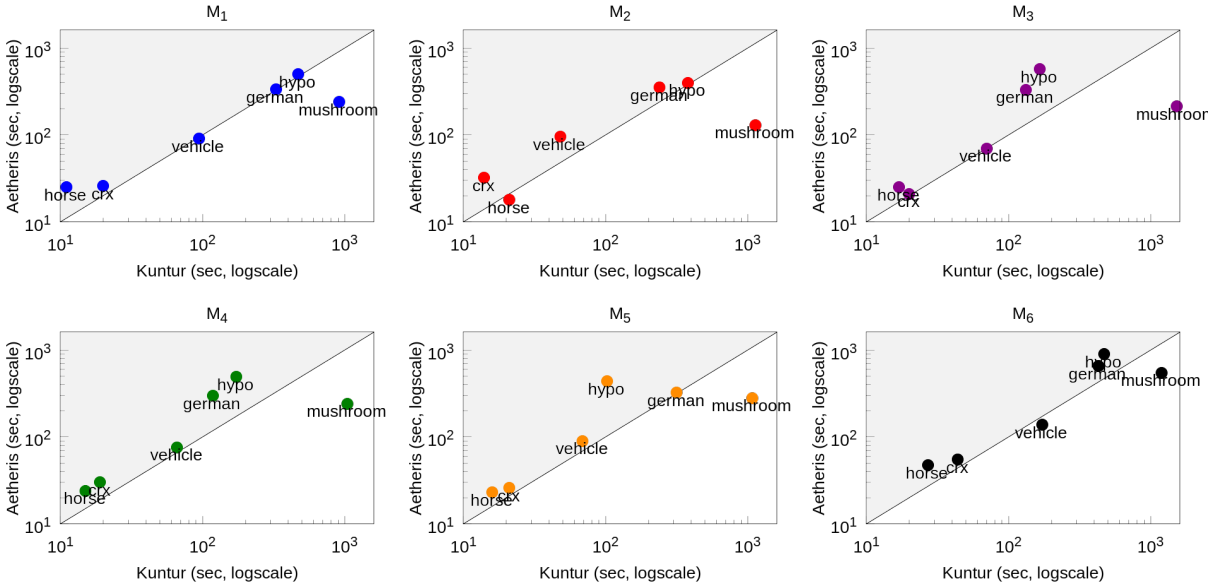

Figure 2: Comparing CPU-times (skypatterns for $M_i, i \in [1..6]$).

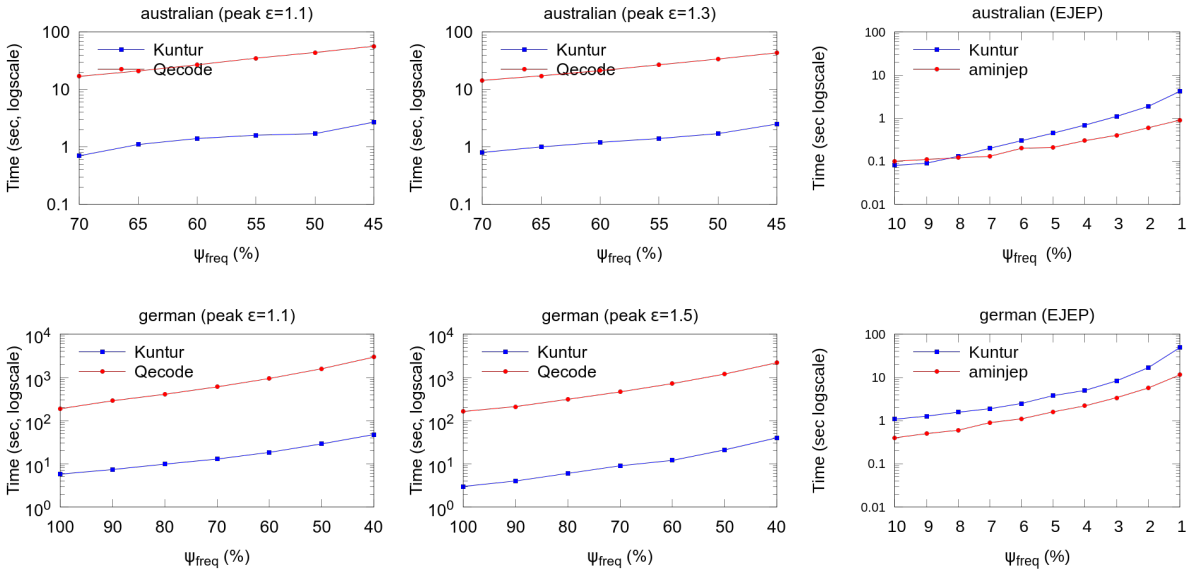

Figure 3: Comparing CPU-times (peak itemsets and Essential JEPs).

datasets (e.g. almost all of the points are in the left part of the plot field). The only exception is the mushroom dataset. This dataset, which is the largest one (both in terms of transactions and items) and with the lowest density (around 18%), leads to the extraction of a relatively small number of closed patterns. This greatly promotes `Aetheris`.

C. Essential JEPs

We compare KUNTUR with the ad-hoc approach AMINJEP [26] which is the only available implementation[4] to extract the Essential JEPs. Experiments were conducted by varying the frequency threshold $\psi_{\texttt{freq}}$. Figure 3 (right side) shows that AMINJEP is slightly better than KUNTUR (average speed-up value 2.4), but our generic approach remains competitive.

D. Peak patterns

We compare KUNTUR with [27]. This approach, developed in QECODE, is the only known approach to extract the peak itemsets. The experiments were conducted by varying the frequency threshold $\psi_{\texttt{freq}}$ for different values of ϵ. Figure 3 shows that KUNTUR significantly outperforms

[4]We have contacted the authors of [20] to get the original code; unfortunately it is not available.

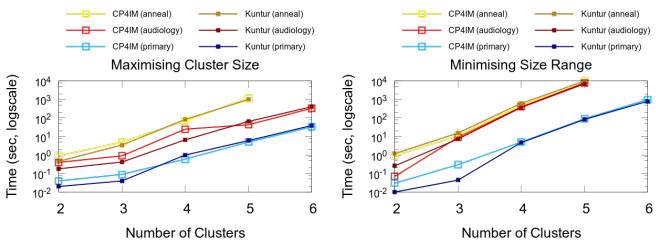

Figure 4: Comparing CPU-times (Conceptual clustering).

[27], as the evolution of CPU-times seems quasi-linear for KUNTUR while it is clearly exponential for [27].

E. Conceptual clustering with Optimisation

We compare KUNTUR with CP4IM [2], which is the only available approach for conceptual clustering with optimisation. Figure 4 (left) reports CPU-times for maximising cluster size, while Figure 4 (right) reports CPU-times for minimising size range. Performances are very close since both approaches share the same boolean encoding. The only difference is that [2] uses a branch and bound algorithm while KUNTUR relies on DynCSPs.

F. Synthesis

Obviously, our generic method is less effective for "simple" patterns whose extraction has been deeply studied (e.g. closed, free, maximal itemsets). However, for mining more elaborated patterns, our approach is as competitive (e.g. skypatterns, Essential JEPs and conceptual clustering), or more efficient (e.g. peak itemsets) than existing methods.

VII. Conclusion

In this paper, we have introduced the notion of Optimal Patterns and we have shown that OPs encompass many data mining problems. A key idea of our framework is to model patterns thanks to the notion of preference. The solving step is performed by a generic method based on dynamic CSPs. Experiments compare the OPs approach with ad hoc methods and show that our approach competes well despite its generic side. The declarative and flexible modeling of our approach paves the way for the definition and discovery of new sets of patterns.

Acknowledgments. This work was supported by the National Agency of Research Hybride ANR-11-BS02-002 project.

References

[1] L. De Raedt, T. Guns, and S. Nijssen, "Constraint programming for itemset mining," in *KDD*, 2008, pp. 204–212.

[2] T. Guns, S. Nijssen, and L. De Raedt, "k-pattern set mining under constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 2, pp. 402–418, 2013.

[3] M. Khiari, P. Boizumault, and B. Crémilleux, "Constraint programming for mining n-ary patterns," in *16th CP*, LNCS, vol. 6308, 2010, pp. 552–567.

[4] L. De Raedt and A. Zimmermann, "Constraint-based pattern set mining," in *7th SIAM SDM*. SIAM, 2007, pp. 237–248.

[5] A. J. Knobbe and E. K. Y. Ho, "Pattern teams," in *10th PKDD*, LNCS, vol. 4213, 2006, pp. 577–584.

[6] F. Geerts, B. Goethals, and T. Mielikäinen, "Tiling databases," in *DS*, LNCS, vol. 3245, 2004, pp. 278–289.

[7] J. Han, J. Wang, Y. Lu, and P. Tzvetkov, "Mining top-k frequent closed patterns without minimum support," in *ICDM*, 2002, pp. 211–218.

[8] D. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Machine Learning*, 2(2) 139-172, 1987.

[9] S. Jabbour, L. Sais, and Y. Salhi, "The top-k frequent closed itemset mining using top-k SAT problem," in *ECML PKDD*, LNCS, vol. 8190, 2013, pp. 403–418.

[10] W. Ugarte, P. Boizumault, S. Loudni, B. Crémilleux, and A. Lepailleur, "Mining (soft-) skypatterns using dynamic CSP," in *CPAIOR*, LNCS, vol. 8451, 2014, pp. 71–87.

[11] J. Boulicaut, A. Bykowski, and C. Rigotti, "Free-sets: A condensed representation of boolean data for the approximation of frequency queries," *Data Min. Knowl. Discov.*, vol. 7, no. 1, pp. 5–22, 2003.

[12] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Discovering frequent closed itemsets for association rules," in *ICDT*, LNCS, vol. 1540, 1999, pp. 398–416.

[13] P. K. Novak, N. Lavrac, and G. I. Webb, "Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining," *Journal of Machine Learning Research*, vol. 10, pp. 377–403, 2009.

[14] H. Mannila and H. Toivonen, "Levelwise search and borders of theories in knowledge discovery," *Data Min. Knowl. Discov.*, vol. 1, no. 3, pp. 241–258, 1997.

[15] A. Soulet, C. Raïssi, M. Plantevit, and B. Crémilleux, "Mining dominant patterns in the sky," in *ICDM*, 2011, pp. 655–664.

[16] B. Crémilleux and A. Soulet, "Discovering knowledge from local patterns with global constraints," in *ICCSA*, LNCS, vol. 5073, 2008, pp. 1242–1257.

[17] A. Fu, R. Kwong, and J. Tang, "Mining N-most interesting itemsets," in *ISMIS*, LNCS 1932, 2000, pp. 59–67.

[18] D. Xin, J. Han, X. Yan, and H. Cheng, "Mining compressed frequent-pattern sets," in *VLDB*. ACM, 2005, pp. 709–720.

[19] A. J. Knobbe and E. K. Y. Ho, "Maximally informative k-itemsets and their efficient discovery," in *12th SIGKDD*. ACM, 2006, pp. 237–244.

[20] H. Fan and K. Ramamohanarao, "An efficient single-scan algorithm for mining essential jumping emerging patterns for classification," *PAKDD*, LNCS 2336, 2002, pp. 456–462.

[21] E. Suzuki, "Undirected discovery of interesting exception rules," *IJPRAI*, vol. 16, no. 8, pp. 1065–1086, 2002.

[22] G. Verfaillie and N. Jussien, "Constraint solving in uncertain and dynamic environments: A survey," *Constraints*, vol. 10, no. 3, pp. 253–281, 2005.

[23] B. Négrevergne, A. Dries, T. Guns, and S. Nijssen, "Dominance programming for itemset mining," in *ICDM*. IEEE Computer Society, 2013, pp. 557–566.

[24] C. Borgelt, "Efficient Implementations of Apriori and Eclat," in *ICDM Workshop FIMI*, 2003.

[25] T. Uno, T. Asai, Y. Uchida, and H. Arimura, "An efficient algorithm for enumerating closed patterns in transaction databases," in *DS*, LNCS, vol. 3245, 2004, pp. 16–31.

[26] B. Kane, B. Cuissart, and B. Crémilleux, "Minimal jumping emerging patterns: Computation and practical assessment," in *PAKDD*, LNCS, vol. 9077, 2015, pp. 722–733.

[27] M. Khiari, A. Lallouet, and J. Vautard, "Extraction de Motifs sous Contraintes Quantifiées," in *JFPC*, 2012.