Mining (Soft-) Skypatterns Using Dynamic CSP

Willy Ugarte Rojas[1], Patrice Boizumault[1], Samir Loudni[1],
Bruno Crémilleux[1], and Alban Lepailleur[2]

[1] GREYC (CNRS UMR 6072) – University of Caen
Campus II Côte de Nacre, 14000 Caen - France
[2] CERMN (UPRES EA 4258 - FR CNRS 3038 INC3M) – University of Caen
Boulevard Becquerel, 14032 Caen Cedex - France

Abstract. Within the pattern mining area, skypatterns enable to express a user-preference point of view according to a dominance relation. In this paper, we deal with the introduction of softness in the skypattern mining problem. First, we show how softness can provide convenient patterns that would be missed otherwise. Then, thanks to Dynamic CSP, we propose a generic and efficient method to mine skypatterns as well as soft ones. Finally, we show the relevance and the effectiveness of our approach through a case study in chemoinformatics and experiments on UCI benchmarks.

1 Introduction

Discovering useful patterns from data is an important field in data mining for data analysis and is used in a wide range of applications. Many approaches have promoted the use of constraints to focus on the most promising knowledge according to a potential interest given by the final user. As the process usually produces a large number of patterns, a large effort is made to a better understanding of the fragmented information conveyed by the patterns and to produce *pattern sets* i.e. sets of patterns satisfying properties on the whole set of patterns [5]. Using the dominance relation is a recent trend in constraint-based data mining to produce useful pattern sets [19].

Skyline queries [3] enable to express a user-preference point of view according to a *dominance* relation. In a multidimensional space where a preference is defined for each dimension, a point p_i *dominates* another point p_j if p_i is better (i.e., more preferred) than p_j in at least one dimension, and p_i is not worse than p_j on every other dimension. However, while this notion of skylines has been extensively developed and researched for database applications, it has remained unused until recently for data mining purposes. Computing *skylines of patterns* from a database is clearly much harder than computing *skylines* in database applications due to the huge difference between the size of search spaces (we explain this issue in Section 5). The inherent complexity on computing skylines of patterns may explain the very few attempts in this direction.

A pioneering work [17] proposed a technique to extract skyline graphs maximizing two measures. Recently, the notion of skyline queries has been integrated into the constraint-based pattern discovery paradigm to mine skyline patterns (henceforth called *skypatterns*) [19]. As an example, a user may prefer a pattern with a high frequency, large length and a high confidence. In this case, we say that a pattern x_i dominates another pattern x_j if $freq(x_i) \geq freq(x_j)$, $size(x_i) \geq size(x_j)$, $confidence(x_i) \geq$

H. Simonis (Ed.): CPAIOR 2014, LNCS 8451, pp. 71–87, 2014.

$confidence(x_j)$ where at least one strict inequality holds. Given a set of patterns, the skypattern set contains the patterns that are not dominated by any other pattern (we formally introduce the notions in the following sections). Skypatterns are interesting for a twofold reason: they do not require any threshold on the measures and the notion of dominance provides a global interest with semantics easily understood by the user.

Nevertheless, skypatterns queries, like other kinds of queries, suffer from the stringent aspect of the constraint-based framework. Indeed, a pattern satisfies or does not satisfy the constraints. But, what about patterns that slightly miss a constraint? A pattern, close to the frontier of the dominance area, could be interesting although it is not a skypattern. In the paper, we formally introduce soft skypatterns. Note that there are very few works such as [2,21] dealing with softness into the mining process.

The contributions of this paper are the following. First, we introduce the notion of soft skypattern. Second, we propose a flexible and efficient approach to mine skypatterns as well as soft ones thanks to the Dynamic CSP (Constraint Satisfaction Problems) framework [22]. Our proposition benefits from the recent progress on cross-fertilization between data mining and Constraint Programming (CP) [4,9,7]. The common point of all these methods is to model in a declarative way pattern mining as CSP, whose resolution provides the complete set of solutions satisfying all the constraints. We show how the (soft-) skypatterns mining problem can be modeled and solved using dynamic CSPs. A major advantage of the method is to improve the mining step during the process thanks to constraints dynamically posted and stemming from the current set of candidate skypatterns. Moreover, the declarative side of the CP framework leads to a unified framework handling softness in the skypattern problem. Finally, the relevance and the effectiveness of our approach is highlighted through a case study in chemoinformatics for discovering toxicophores and experiments on UCI benchmarks.

This paper is organized as follows. Section 2 presents the context and defines skypatterns. Section 3 introduces soft skypatterns. Section 4 presents our flexible and efficient CP approach to mine skypatterns as well as soft ones. We review some related work in Section 5. Finally, Section 6 reports in depth a case study in chemoinformatics and describes experiments on UCI benchmarks.

2 The Skypattern Mining Problem

2.1 Context and Definitions

Let $\mathscr{I}$ be a set of distinct literals called *items*. An itemset (or pattern) is a non-null subset of $\mathscr{I}$. The language of itemsets corresponds to $\mathscr{L}_{\mathscr{I}} = 2^{\mathscr{I}} \backslash \emptyset$. A transactional dataset $\mathscr{T}$ is a multiset of patterns of $\mathscr{L}_{\mathscr{I}}$. Each pattern (or transaction) is a database entry. Table 1 (left side) presents a transactional dataset $\mathscr{T}$ where each transaction t_i is described by items denoted $A, \ldots, F$. The traditional example is a supermarket database in which each transaction corresponds to a customer and every item in the transaction to a product bought by the customer. An attribute (*price*) is associated to each product (see Table 1, right side).

Constraint-based pattern mining aims at extracting all patterns x of $\mathscr{L}_{\mathscr{I}}$ satisfying a query $q(x)$ (conjunction of constraints) which is usually called *theory* [12]: $Th(q) = \{x \in \mathscr{L}_{\mathscr{I}} \mid q(x) \; is \; true\}$. A common example is the frequency measure leading to the

Table 1. Transactional dataset $\mathscr{T}$

Trans.	Items
t_1	$B \quad E \; F$
t_2	$B \; C \; D$
t_3	$A \quad E \; F$
t_4	$A \; B \; C \; D \; E$
t_5	$B \; C \; D \; E$
t_6	$B \; C \; D \; E \; F$
t_7	$A \; B \; C \; D \; E \; F$

Item	A	B	C	D	E	F
Price	30	40	10	40	70	55

minimal frequency constraint ($freq(x) \geq \theta$). The latter provides patterns x having a number of occurrences in the dataset exceeding a given minimal threshold θ. There are other usual measures for a pattern x:

- $size(x)$ is the number of items that pattern x contains.
- $area(x) = freq(x) \times size(x)$.
- $min(x.att)$ (resp. $max(x.att)$) is the smallest (resp. highest) value of the item values of x for attribute att.
- $average(x.att)$ is the average value of the item values of x for attribute att.
- $mean(x) = (min(x.att) + max(x.att))/2$.

Considering the dataset described in Table 1, we have: $freq(BC)$=5, $size(BC)$=2 and $area(BC)$=10. Moreover, $average(BCD.price)$=30 and $mean(BCD.price)$=25.

In many applications, it is highly appropriated to look for contrasts between subsets of transactions, such as toxic and non toxic molecules in chemoinformatics (see Section 6.1). The growth-rate is a well-used contrast measure highlighting patterns whose frequency increases significantly from one subset to another [14]:

Definition 1 (Growth rate). *Let $\mathscr{T}$ be a database partitioned into two subsets $\mathscr{D}_1$ and $\mathscr{D}_2$. The growth rate of a pattern x from $\mathscr{D}_2$ to $\mathscr{D}_1$ is:*

$$m_{gr}(x) = \frac{|\mathscr{D}_2| \times freq(x, \mathscr{D}_1)}{|\mathscr{D}_1| \times freq(x, \mathscr{D}_2)}$$

The collection of patterns contains redundancy w.r.t. measures. Given a measure m, two patterns x_i and x_j are said to be equivalent if $m(x_i) = m(x_j)$. A set of equivalent patterns forms an equivalent class w.r.t. m. The largest element (i.e. the one with the highest number of items) of an equivalence class is called a **closed pattern**. More formally, a pattern x_i is closed w.r.t. m iff $\forall x_j \supsetneq x_i, m(x_j) \neq m(x_i)$. The set of closed patterns is a compact representation of the patterns (i.e we can derive all the patterns with their exact value for m from the closed ones). This definition is straightforwardly extended to a set of measures M.

2.2 Skypatterns

Skypatterns have been recently introduced by [19]. Such patterns enable to express a user-preference point of view according to a dominance relation. Given a set of patterns, the skypattern set contains the patterns that are not dominated by any other pattern.

Definition 2 (Dominance). *Given a set of measures M, a pattern x_i dominates another pattern x_j with respect to M (denoted by $x_i \succ_M x_j$), iff $\forall m \in M, m(x_i) \geq m(x_j)$ and $\exists m \in M, m(x_i) > m(x_j)$.*

Consider the example in Table 1 with M=$\{freq, area\}$. Pattern *BCD* dominates pattern *BC* because $freq(BCD)$=$freq(BC)$=5 and $area(BCD)$>$area(BC)$. For M=$\{freq, size, average\}$, pattern *BDE* dominates pattern *BCE* because $freq(BDE)$=$freq(BCE)$=4, $size(BDE)$=$size(BCE)$=3 and $average(BDE.price)$>$average(BCE.price)$.

Definition 3 (Skypattern operator). *Given a pattern set $P \subseteq \mathscr{L}_{\mathscr{I}}$ and a set of measures M, a skypattern of P with respect to M is a pattern not dominated in P with respect to M. The skypattern operator Sky(P,M) returns all the skypatterns of P with respect to M: Sky(P,M) = $\{x_i \in P \mid \nexists x_j \in P, x_j \succ_M x_i\}$.*

The skypattern mining problem is thus to evaluate the query $Sky(\mathscr{L}_{\mathscr{I}}, M)$. For instance, from the data set in Table 1 and with M=$\{freq, size\}$, $Sky(\mathscr{L}_{\mathscr{I}}, M) = \{ABCDEF, BCDEF, ABCDE, BCDE, BCD, B, E\}$ (see Fig. 1a). The shaded area is called the *forbidden area*, as it cannot contain any skypattern. The other part is called the *dominance area*. The edge of the dominance area (bold line) marks the boundary between them.

3 The Soft Skypattern Mining Problem

This section introduces the notion of softness in the skypattern mining problem. As the skypatterns suffer from the stringent aspect of the constraint-based pattern framework, we propose to capture valuable patterns occurring in the forbidden area (that we call soft skypatterns). We define two kinds of soft skypatterns: the *edge-skypatterns* that belongs to the edge of the dominance area (see Section 3.1) and the δ-*skypatterns* that are close to this edge (see Section 3.2). The key idea is to strengthen the dominance relation in order to soften the notion of non dominated patterns.

3.1 Edge-Skypatterns

Edge-skypatterns are defined according to a dominance relation and a *Sky* operator.

Definition 4 (Strict Dominance). *Given a set of measures M, a pattern x_i strictly dominates a pattern x_j with respect to M (denoted by $x_i \gg_M x_j$), iff $\forall m \in M$, $m(x_i) > m(x_j)$.*

Definition 5 (Edge-skypattern operator). *Given a pattern set $P \subseteq \mathscr{L}_{\mathscr{I}}$ and a set of measures M, an edge-skypattern of P, with respect to M, is a pattern not strictly dominated in P, with respect to M. The operator Edge-Sky(P,M) returns all the edge-skypatterns of P with respect to M: Edge-Sky$(P,M) = \{x_i \in P \mid \nexists x_j \in P, x_j \gg_M x_i\}$*

Given a set of measures M, the edge-skypattern mining problem is thus to evaluate the query *Edge-Sky*(P,M). Fig. 1a depicts the 28=7+(4+8+3+4+2) edge-skypatterns extracted from the example in Table 1 for M=$\{freq, size\}$. Obviously, all edge-skypatterns belong to the edge of the dominance area, and seven of them are skypatterns.

Proposition 1. *For two patterns x_i and x_j, $x_i \gg_M x_j \implies x_i \succ_M x_j$. So, for a pattern set P and a set of measures M, $Sky(P,M) \subseteq$ Edge-Sky(P,M).*

3.2 δ-Skypatterns

In many cases the user is interested in patterns close to the border of the dominance area because they express a trade-off between the measures. The δ-skypatterns address this issue where δ means a percentage of relaxation allowed by the user. Let $0 < \delta \leq 1$.

Definition 6 **(δ-Dominance).** *Given a set of measures M, a pattern x_i δ-dominates another pattern x_j w.r.t. M (denoted by $x_i \succ^{\delta}_{M} x_j$), iff $\forall m \in M$, $(1-\delta) \times m(x_i) > m(x_j)$.*

Definition 7 **(δ-Skypattern operator).** *Given a pattern set $P \subseteq \mathscr{L}_{\mathscr{I}}$ and a set of measures M, a δ-skypattern of P with respect to M is a pattern not δ-dominated in P with respect to M. The δ-skypattern operator δ-$Sky(P,M)$ returns all the δ-skypatterns of P with respect to M: δ-$Sky(P,M) = \{x_i \in P \mid \nexists x_j \in P : x_j \succ^{\delta}_{M} x_i\}$.*

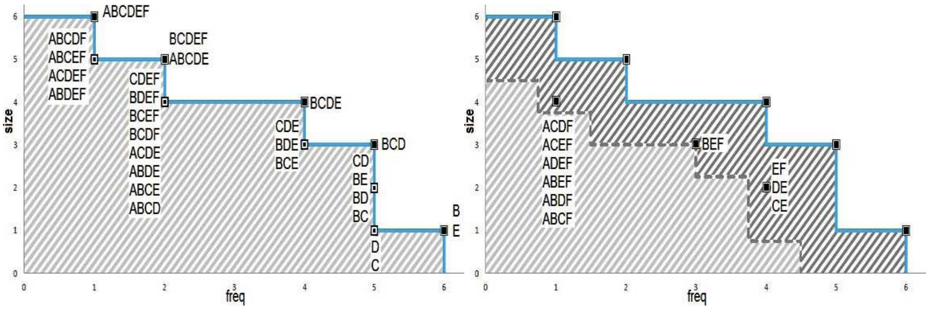
The δ-skypattern mining problem is thus to evaluate the query δ-$Sky(P,M)$. There are 38 (28+10) δ-skypatterns extracted from the example in Table 1 for $M=\{freq, size\}$ and δ=0.25. Fig. 1b only depicts the 10 δ-skypatterns that are not edge-skypatterns. Intuitively, the δ-skypatterns are close to the edge of the dominance relation, the value of δ is the maximal relative distance between a skypattern and this border.

Proposition 2. *For two patterns x_i and x_j, $x_i \succ^{\delta}_{M} x_j \implies x_i \gg_M x_j$. So, for a pattern set P and a set of measures M, $Edge$-$Sky(P,M) \subseteq \delta$-$Sky(P,M)$.*

To conclude, given a pattern set $P \subseteq \mathscr{L}_{\mathscr{I}}$ and a set of measures M, the following inclusions hold: $Sky(P,M) \subseteq Edge$-$Sky(P,M) \subseteq \delta$-$Sky(P,M)$.

4 Mining (Soft-) Skypatterns Using Dynamic CSP

This section describes how the skypattern and the soft skypattern mining problems can be modeled and solved using Dynamic CSP [22]. The implementation was carried out in `Gecode` by extending the CP-based pattern extractor developed by [9]. The main idea of our of approach is to improve the mining step during the process thanks to constraints dynamically posted and stemming from the current set of the candidate skypatterns. This process stops when the forbidden area cannot be enlarged. Finally, the completeness of our approach is ensured by the completeness of the CP solver.



(a) Edge-skypatterns.

(b) δ-skypatterns (that are not edge ones).

Fig. 1. Soft-skypatterns extracted from the example in Table 1

4.1 Dynamic CSP

A Dynamic CSP [22] is a sequence $P_1, P_2, ..., P_n$ of CSP, each one resulting from some changes in the definition of the previous one. These changes may affect every component in the problem definition: variables, domains and constraints. *For our approach, changes are only performed by adding new constraints.* Solving such dynamic CSP involves solving a single CSP with additional constraints posted during search. Each time a new solution is found, new constraints are imposed. Such constraints will survive backtracking and state that next solutions should verify both the current set of constraints and the added ones.

4.2 Mining Skypatterns

Constraints on the dominance relation are dynamically posted during the mining process and softness is easily introduced using such constraints. Variable x will denote the (unknown) skypattern we are looking for. Changes are only performed by adding new constraints. So, we consider the sequence $P_1, P_2, ..., P_n$ of CSP where M is a set of measures, each $P_i = (\{x\}, \mathscr{L}, q_i(x))$ and:

- $q_1(x) = closed_M(x)$
- $q_{i+1}(x) = q_i(x) \wedge \phi_i(x)$ where s_i is the first solution to query $q_i(x)$

First, the constraint $closed_M(x)$ states that x must be a closed pattern w.r.t M, it allows to reduce the number of redundant patterns (see Section 2.1). Then, the constraint $\phi_i(x) \equiv \neg(s_i \succ_M x)$ states that the next solution (which is searched) will not be dominated by s_i. Using a short induction proof, we can easily argue that query $q_{i+1}(x)$ looks for a pattern x that will not be dominated by any of the patterns s_1, s_2, $\ldots$, s_i.

Each time the first solution s_i to query $q_i(x)$ is found, we dynamically post a new constraint $\phi_i(x)$ leading to reduce the search space. This process stops when we cannot enlarge the forbidden area (i.e. there exits n s.t. query $q_{n+1}(x)$ has no solution). For skypatterns, $\phi_i(x)$ states that $\neg(s_i \succ_M x)$ (see Definition 2):

$$\phi_i(x) \equiv \left(\bigvee_{m \in M} m(s_i) < m(x) \right) \vee \left(\bigwedge_{m \in M} m(s_i) = m(x) \right)$$

But, the n extracted patterns s_1, s_2, $\ldots$, s_n are not necessarily all skypatterns. Some of them can only be "intermediate" patterns simply used to enlarge the forbidden area. A post processing step must be performed to filter all candidate patterns s_i that are not skypatterns, i.e. for which there exists s_j $(1 \leq i < j \leq n)$ s.t. s_j dominates s_i. So mining skypatterns is achieved in a two-steps approach:

1. Compute the set $S = \{s_1, s_2, \ldots, s_n\}$ of candidates using Dynamic CSP.
2. Remove all patterns $s_i \in S$ that are not skypatterns.

While the number of candidates (n) could be very large (the skypattern mining problem is NP-complete), it remains reasonably-sized in practice for the experiments we conducted (see Table 2 for the case study in chemoinformatics).

4.3 Mining Soft Skypatterns

Soft skypatterns are processed exactly the same way as skypatterns. Each kind of soft skypatterns has its own constraint $\phi_i(x)$ according to its relation of dominance.

For edge-skypatterns, $\phi_i(x)$ states that $\neg(s_i \gg_M x)$ (see Definition 4):

$$\phi_i(x) \equiv \bigvee_{m \in M} m(s_i) \leq m(x)$$

For δ-skypatterns, $\phi_i(x)$ states that $\neg(s_i \succ_M^\delta x)$ (see Definition 6):

$$\phi_i(x) \equiv \bigvee_{m \in M} (1-\delta) \times m(s_i) < m(x)$$

As previously, the n extracted patterns are not necessarily all soft skypatterns. So, a post processing is also required as for skypatterns. Once again, the number of candidates (n) remains reasonably-sized in practice for the experiments we conducted (see Table 2 for the case study in chemoinformatics and Figure 5 for UCI benchmarks).

4.4 Pattern Encoding

Let $\boldsymbol{d}$ be the 0/1 matrix where $\forall t \in \mathscr{T}, \forall i \in \mathscr{I}$, $(d_{t,i} = 1) \Leftrightarrow (i \in t)$. Pattern variables are set variables represented by their characteristic function with boolean variables. [4,7] model an unknown pattern x and its associated dataset $\mathscr{T}$ by introducing two sets of boolean variables: $\{X_i \mid i \in \mathscr{I}\}$ where $(X_i = 1) \Leftrightarrow (i \in x)$, and $\{T_t \mid t \in \mathscr{T}\}$ where $(T_t = 1) \Leftrightarrow (x \subseteq t)$. Each set of boolean variables aims at representing the characteristic function of the unknown pattern.

The relationship between x and $\mathscr{T}$ is modeled by posting reified constraints stating that, for each transaction t, $(T_t = 1)$ iff t is covered by x:

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

4.5 Closedness Constraints

Section 2.1 recalls the definition of closed patterns satisfying closedness constraints. Let M=$\{min\}$ and $val(j)$ a function that associates an attribute value to each item j. If item i belongs to x, then its value must be greater than or equal to the min. Conversely, if this value is greater than or equal to the min, i must belong to x (if not, x would not be maximal for inclusion). So, x is a closed pattern for the measure min iff:

$$\forall i \in \mathscr{I}, (X_i = 1) \Leftrightarrow val(i) \geq min\{val(j) \mid j \in x\} \tag{2}$$

Let M=$\{freq\}$, the closedness constraint ensures that a pattern has no superset with the same frequency. So $closed_M(x)$ is modeled using Equation 1 and Equation 3.

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

There are equivalences between closed patterns according to measures: the closed patterns w.r.t *mean* and *min* are the same and the closed patterns w.r.t *area*, *growth-rate* and *frequency* are the same [19]. The constraint $closed_M(x)$ states that x must be a closed pattern w.r.t M (the closed patterns w.r.t M gather the closed patterns w.r.t each measure of M i.e. x is closed w.r.t M iff x is closed for at least one measure $m \in M$).

5 Related Work

Computing skylines is a derivation from the maximal vector problem in computational geometry [13], the Pareto frontier [10] and multi-objective optimization. Since its rediscovery within the database community by [3], several methods have been developed for answering skyline queries [15,16,20]. These methods assume that tuples are stored in efficient tree data structures. Alternative approaches have also been proposed to help the user in selecting most significant skylines. For example, [11] measures this significance by means of the number of points dominated by a skyline.
Introducing softness for skylines. [8] have proposed thick skylines to extend the concept of skyline. A thick skyline is either a skyline point p_i, or a point p_j dominated by a skyline point p_i and such that p_j is close to p_i. In this work, the idea of softness is limited to metric semi-balls of radius $\varepsilon{>}0$ centered at points p_i, where p_i are skylines.
Computing skypatterns is different from computing skylines. Skyline queries focus on the extraction of tuples of the dataset and assume that all the elements are in the dataset, while the skypattern mining task consists in extracting patterns which are elements of the frontier defined by the given measures. The skypattern problem is clearly harder because the search space for skypatterns is much larger than the search space for skylines: $O(2^{|\mathscr{I}|})$ instead of $O(|\mathscr{T}|)$ for skylines.
Computing skypatterns. [19] have proposed `Aetheris`, an approach taking benefit of theoretical relationships between pattern condensed representations and skypatterns. `Aetheris` proceeds in two steps. First, condensed representations of the whole set of patterns (i.e. closed patterns according to the considered set of measures) are extracted. Then, the operator *Sky* (see Definition 3) is applied. Nevertheless, this method can only use a crisp dominance relation. [17] deals with skyline graphs but their technique only maximizes two measures (number of vertices and edge connectivity).
CP for computing the Pareto frontier. [6] has proposed an algorithm that provides the Pareto frontier in a CSP. This algorithm is based on the concept of nogoods and uses spatial data structures (quadtrees) to arrange the set of nogoods. This approach deals for computing skylines and cannot be directly applied to skypatterns. The application is not immediate since several different patterns may correspond to a same point (they all have the same values for the considered measures). As experiments show the practical efficiency of our approach, we have considered that adding [6] to a constraint solver would require an important development time compared to the expected benefits.

6 Experimental Study

First, we report in depth a case study in chemoinformatics by performing a CPU time analysis as well as a qualitative analysis that demonstrates the usefulness and the interest of soft skypatterns (see Section 6.1). Then, using experiments on UCI benchmarks, we show and discuss the practical issues of our approach (see Section 6.2).

`Aetheris` *and* `CP+SKY` *(hard version of the skypatterns) produce exactly the same set of skypatterns.* So, the same outputs are compared Section 6.1.2 (Table 2, skypatterns part) and Section 6.2.1 (Fig 4 and Fig 5). Up to now, there is a single work (`Aetheris` [19]) to extract skypatterns, no other comparison is possible on skypatterns. Finally, soft skypatterns are completely new and there is no other competitor.

All experiments were conducted on a computer running Linux operating system with a core i3 processor at 2.13 GHz and a RAM of 4 GB. `Aetheris` was kindly provided by A. Soulet and used in [19]. The implementation of `CP+SKY` was carried out in `Gecode` by extending the CP-based patterns extractor developed by [9].

6.1 Case Study: Discovering Toxicophores

A major issue in chemoinformatics is to establish relationships between chemicals and their activity in (eco)toxicity. Chemical fragments[1] which cause toxicity are called *toxicophores* and their discovery is at the core of prediction models in (eco)toxicity [1,18]. The aim of this study, which is part of a larger research collaboration with the CERMN Lab, is to investigate the use of softness for discovering toxicophores.

6.1.1 Experimental Protocol. The dataset is collected from the ECB web site[2]. For each chemical, the chemists associate it with hazard statement codes (HSC) in 3 categories: H400 (very toxic, CL50 $\leq$ 1 mg/L), H401 (toxic, 1 mg/L $<$ CL50 $\leq$ 10 mg/L), and H402 (harmful, 10 mg/L $<$ CL50 $\leq$ 100 mg/L). We focus on the H400 and H402 classes. The dataset $\mathscr{T}$ consists of 567 chemicals (transactions), 372 from the H400 class and 195 from the H402 class. The chemicals are encoded using 1450 frequent closed subgraphs (items) previously extracted[3] with a 1% relative frequency threshold.

In order to discover patterns as candidate toxicophores, we use both measures typically used in contrast mining [14] such as the *growth rate* (see Definition 1) since toxicophores are linked to a classification problem and measures expressing the background knowledge such as the *aromaticity* because chemists consider that this information may yield promising candidate toxicophores. Now, we describe these three measures.

- *Growth rate.* When a pattern has a frequency which significantly increases from the H402 class to the H400 class, then it stands a potential structural alert related to an excess of the toxicity: if a chemical has, in its structure, fragments that are related to an effect, then it is more likely to be toxic. Emerging patterns embody this natural idea by using the growth-rate measure.

- *Frequency.* Real-world datasets are often noisy and patterns with low frequency may be artefacts. The minimal frequency constraint ensures that a pattern is representative enough (i.e., the higher the frequency, the better is).

- *Aromaticity.* Chemists know that the aromaticity is a chemical property that favors toxicity since their metabolites can lead to very reactive species which can interact with biomacromolecules in a harmful way. We compute the aromaticity of a pattern as the mean of the aromaticity of its chemical fragments.

We consider four sets of measures: $M_1{=}\{growth\text{-}rate, freq\}$, $M_2{=}\{growth\text{-}rate, aromaticity\}$, $M_3{=}\{freq, aromaticity\}$ and $M_4{=}\{growth\text{-}rate, freq, aromaticity\}$. Redundancy is reduced by using closed skypatterns (see Section 4.2). For δ-skypatterns, we consider two values: $\delta = 0.1$ and $\delta = 0.2$. The extracted skypatterns and soft skypatterns are made of molecular fragments. To evaluate the presence of toxicophores, an expert analysis leads to the identification of well-known toxicophores.

[1] A fragment denotes a connected part of a chemical structure having at least one chemical bond.

[2] European Chemicals Bureau: `http://echa.europa.eu/`

[3] A chemical Ch contains an item A if Ch supports A, and A is a frequent subgraph of $\mathscr{T}$.

Table 2. Skypattern mining on ECB dataset

	Skypatterns					Edge-Skypatterns			δ-Skypatterns					
		`CP+SKY`		`Aetheris`		`CP+Edge-SKY`			`CP+`δ`-SKY`					
									$\delta = 0.1$			$\delta = 0.2$		
	# of Skypatterns	# of Candidates	CPU-Time	# of closed patterns	CPU-Time	# of Edge-skypatterns	# of Candidates	CPU-Time	# of δ-skypatterns	# of Candidates	CPU-Time	# of δ-skypatterns	# of Candidates	CPU-Time
M_1	**8**	613	**18m:34s**	41,887	**19m:20s**	**24**	1,746	**19m:02s**	**25**	4,204	**20m:48s**	**87**	6,253	**22m:36s**
M_2	**5**	140	**15m:32s**	53,201	**21m:33s**	**76**	688	**17m:51s**	**354**	1,678	**18m:14s**	**1,670**	2,816	**23m:44s**
M_3	**2**	456	**16m:45s**	157,911	**21m:16s**	**72**	1,726	**16m:50s**	**352**	4,070	**19m:43s**	**1,654**	6,699	**22m:25s**
M_4	**21**	869	**17m:49s**	12,126	**21m:40s**	**144**	3,021	**20m:27s**	**385**	6,048	**23m:36s**	**1,724**	8,986	**30m:14s**

6.1.2 Performance Analysis. Table 2 reports, for each set of measures M_i: (i) the number of skypatterns that is the same for both approaches, (ii) for `CP+SKY`, the number of candidates (see Section 4.2) and the associated CPU-time and (iii) for `Aetheris`, the number of closed patterns and the associated CPU-time, (iv) the number of edge-skypatterns that are not skypatterns, the number of candidates and the required CPU-time, and (v) the number of δ-skypatterns that are not edge-skypatterns, the number of candidates and the required CPU-time. For each method, reported CPU-times include the two steps.

`CP+SKY` outperforms `Aetheris` in terms of CPU-times (see Table 2, skypatterns part). Moreover, the number of candidates generated by our approach remains small compared to the number of closed patterns computed by `Aetheris`. `Aetheris` applies the skypattern operator on the whole set of closed patterns (column 4) whereas `CP+SKY` applies the skypattern operator on a subset of the closed patterns (column 2). That explains why the numbers in column 2 are lower than the numbers in column 4. It shows the interest of the CP approach: thanks to the filtering of dynamically posted constraints, the search space is drastically reduced.

Finally, the number of soft skypatterns remains reasonably small. For edge skypatterns, there is a maximum of 144 patterns, while for δ-skypatterns, there is a maximum of $1,724$ patterns ($\delta = 0.2$).

6.1.3 Qualitative Analysis. In this subsection, we show that soft skypatterns enable (i) to efficiently detect well-known toxicophores emphasized by skypatterns, and (ii) to discover new and interesting toxicophores that would be missed by skypatterns.

- Growth rate and frequency measures (M_1). Only 8 skypatterns are found, and 3 well-known toxicophores are emphasized (see Figure 2). Two of them are aromatic compounds, namely the chlorobenzene (p_1) and the phenol rings (p_2). The third one, the organophosphorus moiety (p_3) is a component occurring in numerous pesticides.

Soft skypatterns confirm the trends given by skypatterns: the chloro-substituted aromatic rings (e.g. p_4), and the organophosphorus moiety (e.g. p_5) are detected by both the edge-skypatterns and by the δ-skypatterns.

- Growth rate and aromaticity measures (M_2). As results for M_2 and M_3 are similar, we only report the qualitative analysis for M_2. Edge-skypatterns leads to the extraction

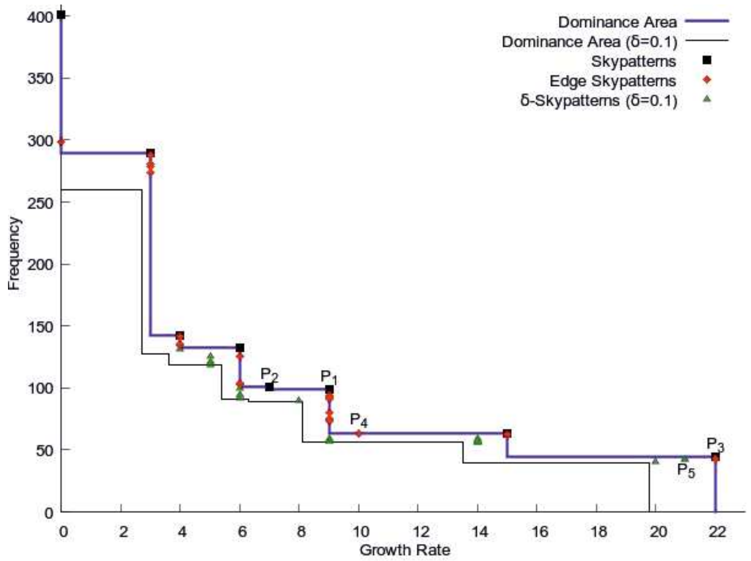


Fig. 2. Analysing the (soft-) skypatterns for M_1

of four new toxicophores: (i) nitrogen aromatic compounds: indole and benzoimidazole, (ii) S-containing aromatic compounds: benzothiophene, (iii) aromatic oxygen compounds: benzofurane, and (iv) polycyclic aromatic hydrocarbons: naphthalene. δ-skypatterns complete the list of the aromatic rings, which were not found with the skypatterns, namely biphenyl.

- Growth rate, frequency and aromaticity measures (M_4)**.** The most interesting results are provided using M_4. Table 3 shows the ratios analysis for the (soft-) skypatterns. Col. 1 provides the name of toxicophores. Col. 2-5 give the number of (soft-) skypatterns containing one complete[4] representative fragment of each toxicophore and, between parentheses, their ratios (# of (soft-) skypatterns containing this toxicophore divided by the total # of (soft-) skypatterns, in bold at the 2nd row). Col. 6 (resp. Col. 7) gives the number of chemicals classified H400 i.e. high toxicity (resp. H402 i.e. harmful) containing at least one representative fragment of the toxicophore. Col. 8-10 show the gains provided by using soft skypatterns for discovering toxicophores (ratio soft skypatterns divided by ratio skypatterns). Bold numbers denote a gain greater than 1 and ∞ means that the toxicophore is only found by soft skypatterns.

21 skypatterns are mined (see Figure 3), and several well-known toxicophores are emphasized: the phenol ring (e_4), the chloro-substituted aromatic ring (e_3), the alkyl-substituted benzene (e_2), and the organophosphorus moiety (P_1). Besides, information dealing with nitrogen aromatic compounds are also extracted (e_1). Table 3 details the repartition of the skypatterns containing only one complete toxicophore compound, according to the toxicophores discussed above. We can observe that very few patterns are extracted.

[4] Patterns with only sub-fragments of a toxicophore are not taken into account.

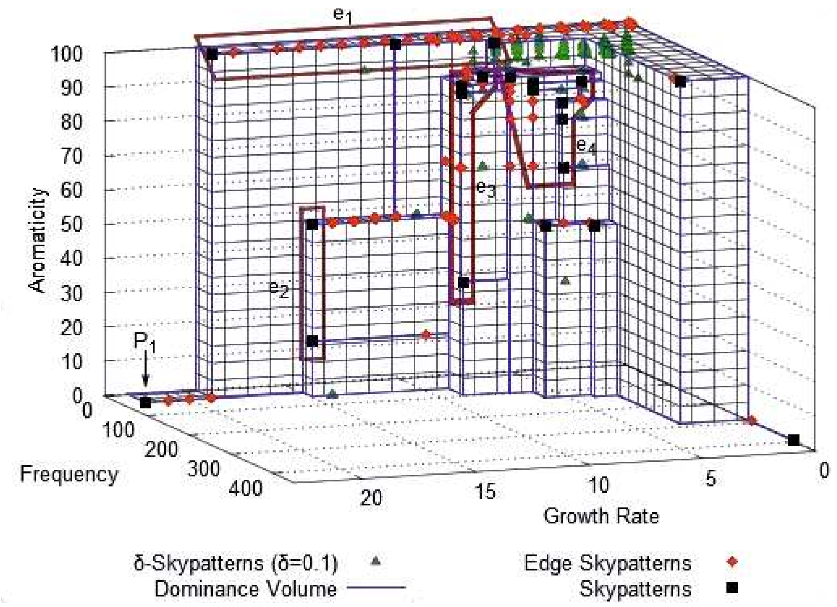


Fig. 3. Analysing the (soft-) skypatterns for M_4

Table 3. Ratio analysis of (soft-)skypattern mining

Chemical		(2)	(3)	(4)	(5)	(6)	(7)	Gain		
								(3)	(4)	(5)
		21	**165**	**550**	**1889**					
Benzene		4 (0.19)	68 (0.41)	322 (0.59)	1373 (0.73)	63.7	18.9	**2.16**	**3.11**	**3.84**
Chlorobenzene		1 (0.05)	2 (0.01)	51 (0.09)	311 (0.16)	22.5	2.5	0.20	**1.80**	**3.20**
Phenol		1 (0.05)	11 (0.07)	32 (0.06)	302 (0.16)	25.2	3.5	**1.40**	**1.20**	**3.20**
Organophosphate	**Basic**	2 (0.10)	18 (0.11)	30 (0.05)	40 (0.02)	18.0	2.5	**1.10**	0.50	0.20
	Exotic		38 (0.23)	66 (0.12)	112 (0.06)	18.0	2.5	∞	∞	∞
Nitrogen aromatic rings			15 (0.09)	74 (0.13)	175 (0.09)	8.6	2.0	∞	∞	∞
Polycyclic aromatic rings			12 (0.07)	178 (0.32)	302 (0.16)	7.2	3.5	∞	∞	∞
Alkyl-substituted benzene			4 (0.02)	64 (0.12)	649 (0.34)	30.9	11.7	∞	∞	∞
Aniline				15 (0.03)	259 (0.14)	24.7	11.3		∞	∞
Alkyl-substituted aniline					157 (0.08)	12.0	7.1			∞
Chlorophenol					168 (0.09)	9.6	1.5			∞
Alkyl phenyl ether					106 (0.06)	9.9	3.0			∞
Alkyl-substituted phenol					61 (0.03)	9.6	1.5			∞
Dichlorobenzene					59 (0.03)	9.9	1.5			∞

(2) Skypatterns (3) Edge-Skypatterns
(4) δ-Skypatterns ($\delta = 0.1$) (5) δ-Skypatterns ($\delta = 0.2$)
(6) coverage rate on H400 (%) (7) coverage rate on H402 (%)

Soft skypatterns enable to detect more precisely the first four toxicophores (see Table 3). For instance, 41% of edge-skypatterns extracted contain the benzene ring, against 19% for hard skypatterns (gain of 2.16: egde-skypatterns detect 2.16 times more patterns containing this fragment compared to hard ones). This gain reaches about 3.11 (resp. 3.84) for $\delta = 0.1$ (resp. 0.2). The same trends hold for chlorobenzene and phenol rings, where 16% of extracted δ-skypatterns ($\delta = 0.2$) include such fragments, against 5% in the hard case (gain of 3.20). From a chemical point of view, these fragments cover well the H400 molecules (from 18% to 63.7%), as is shown in Col. 6, thus demonstrating the toxic nature of the extracted patterns, particularly in the soft case.

Regarding the aromatic rings previously discussed (gray lines of Table 3), *several new patterns containing these toxicophores are only mined by soft skypatterns*. δ-skypatterns (with δ=0.1) allow to better discover these toxicophores compared to edge-skypatterns (average gain of about 4). *Moreover, several patterns with novel fragments of a great interest are solely detected by δ-skypatterns* (yellow lines in Table 3), particularly with δ=0.2. It is important to note that 22% of these patterns include aromatic amines (12% for aniline and 8% for substituted anilines). These two toxicophores, which cover respectively 24.7% and 12% of molecules classified H400, are very harmful to aquatic organisms. The other toxicophores are extracted by δ-skypatterns with ratios ranging from 3% to 9%.

To conclude, soft skypatterns enable to efficiently detect well-known toxicophores emphasized by skypatterns, and to discover new and interesting toxicophores that would be missed by skypatterns.

6.2 Experiments on UCI Benchmarks

Our experiments on UCI[5] benchmarks thoroughly investigate the behavior on `CP+SKY` and `Aetheris` with sets of 4 or 5 measures. We made this choice because the user often handles a limited number of measures when dealing with applications on real-world datasets (see for instance our case study in chemoinformatics in Section 6.1).

Experiments were carried out on 23 various (in terms of dimensions and density) datasets (see Col 1 of Table 4). We considered 5 measures M_6=$\{freq, max, area, mean, growth\text{-}rate\}$ and 6 sets of measures: M_6 and all the combinations of 4 measures from M_6 (noted M_1, M_2, M_3, M_4 and M_5). Measures using numeric values, like *mean*, were applied on attribute values that were randomly generated within the range $[0..1]$. For each method, reported CPU-times include the two steps.

6.2.1 Mining Skypatterns. Figure 4 shows a scatter plot of CPU-times for `CP+SKY` and `Aetheris`. Each point represents a skypattern query for a dataset: its x-value is the CPU-time the `CP+SKY` method took to mine it, its y-value is the CPU-time of `Aetheris`. We associate to each dataset a color. Moreover, we only report CPU-times for the 6 datasets requiring more than 30 seconds, either for `CP+SKY` or `Aetheris`. For both approaches, CPU times are very small and quite similar on the remaining 17 datasets.

`CP+SKY` outperforms `Aetheris` on many datasets (e.g. almost all of the points are in the left part of the plot field of Figure 4). The only exception is the dataset mushroom. This dataset, which is the largest one (both in terms of transactions and items) and with

[5] `http://www.ics.uci.edu/ mlearn/MLRepository.html`

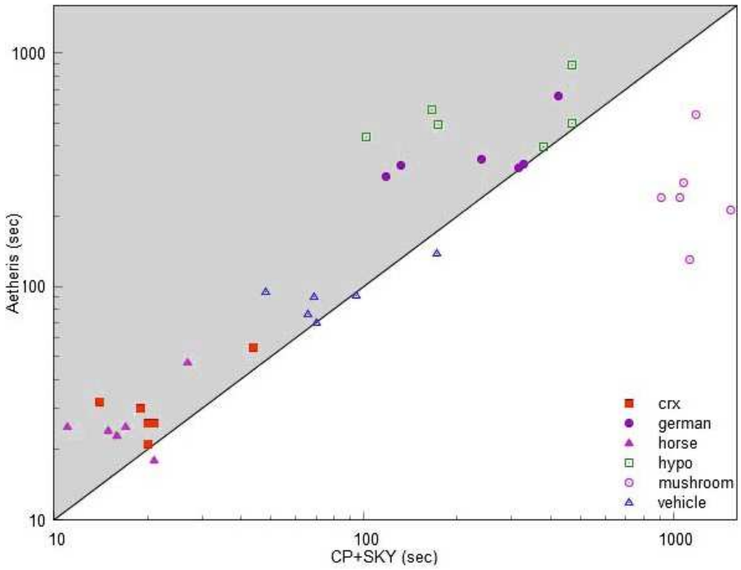


Fig. 4. Comparing CPU times on 6 UCI datasets for $M_1, \ldots, M_6$

the lowest density (around 18%), leads to the extraction of a relatively small number of closed patterns. This greatly promotes Aetheris.

Figure 5 compares, for each set of measures M_i ($1 \leq i \leq 6$), the number of closed patterns for Aetheris with the number of candidates for CP+SKY. We also report the

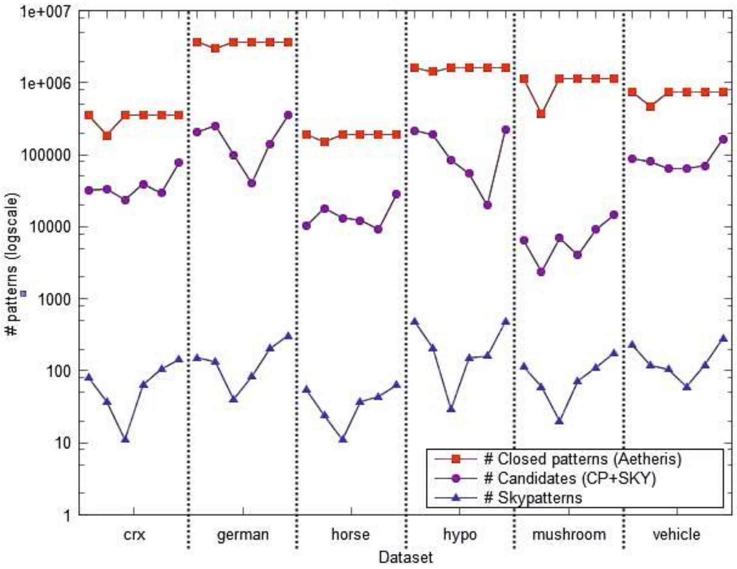


Fig. 5. Comparing # of closed patterns, candidates and skypatterns on 6 datasets

Table 4. Analysis of soft skypattern mining on UCI benchmarks for M_6

				CP+Edge-Sky		CP+δ-Sky ($\delta = 5\%$)		CP+δ-Sky ($\delta = 10\%$)		CP+δ-Sky ($\delta = 15\%$)		CP+δ-Sky ($\delta = 20\%$)	
Dataset	# items	# transactions	density	# of Edge-skypatterns	Time (sec)	# of δ-skypatterns	Time (sec)	# of δ-skypatterns	Time (sec)	# of δ-skypatterns	Time (sec)	# of δ-skypatterns	Time (sec)
abalone	28	4,178	0.321	2,634	36	1,373	38	1,432	38	6,303	38	7,256	42
anneal	68	798	0.195	3,162	28	8,184	34	20,242	35	24,029	37	27,214	36
austral	55	690	0.272	11,714	69	34,205	70	68,855	99	69,487	102	70,652	113
breast	43	286	0.231	1,409	1	17	1	1,651	1	2,429	1	2,443	1
cleve	43	303	0.325	14,636	19	4,466	19	30,605	22	30,952	22	50,275	23
cmc	28	1,474	0.357	14,406	31	3,297	32	3,351	32	11,848	33	14,020	33
crx	59	690	0.269	29,068	134	73,627	151	73,707	159	105,344	166	165,782	167
german	76	1,000	0.276	93,087	1,157	170,169	2,614	230,457	2,995	270,435	3,439	290,654	3,483
glass	34	216	0.295	2,296	1	109	1	1,531	1	2,491	1	4,035	1
heart	38	270	0.368	15,563	15	644	16	34,136	16	42,685	18	44,114	18
hepatic	45	155	0.421	15,002	24	6,122	24	45,572	25	50,686	25	60,857	26
horse	75	300	0.235	13,068	54	39,149	60	43,073	66	55,175	68	74,275	71
hypo	47	3,163	0.389	278,625	1,343	104,147	1,387	115,126	1,402	116,654	1,463	117,089	1,487
iris	15	151	0.333	55	1	20	1	27	1	49	1	67	1
lymph	59	142	0.322	8,286	19	49,846	19	59,753	20	62,143	20	65,946	21
mushroom	119	8,124	0.193	21,639	3,241	33,757	3,328	99,852	3,336	129,383	3,407	150,965	3,614
new-thyroid	21	216	0.287	119	1	41	1	137	1	154	1	173	1
page	35	941	0.314	2,675	18	7,136	19	9,714	19	17,387	21	19,094	22
pima	26	768	0.346	1,778	5	518	5	3,439	5	4,308	5	4,358	6
tic-tac-toe	29	259	0.344	6,800	16	4,078	18	18,584	19	20,130	20	22,576	21
vehicle	58	846	0.327	76,732	687	716	689	2,457	751	3,789	782	4,369	787
wine	45	179	0.311	3,155	5	2,490	5	4,422	5	7,507	5	13,407	6
zoo	43	102	0.394	2,254	2	3,361	2	4,829	2	7,724	2	8,986	2

number of skypatterns. The number of candidates generated by our approach remains very small (some thousands) compared to the huge number of closed patterns computed by `Aetheris` (about millions). Finally, the number of skypatterns remains small.

6.2.2 Mining Soft Skypatterns. This section shows the feasibility of mining soft skypatterns on UCI Benchmarks (for these experiments, parameter δ has been set to $\{0.05, 0.1, 0.15, 0.2\}$). As our proposal is the only approach able to mine soft skypatterns, it is no longer compared with `Aetheris`. Table 4 reports, for each dataset (i) the number of edge-skypatterns that are not (hard) skypatterns, the number of candidates and the required CPU-time, (ii) for δ in $\{0.05, 0.1, 0.15, 0.2\}$ the number for δ-skypatterns that are not edge-skypatterns, the number of candidates and the required CPU-time. Even if the number of soft patterns increases with δ, our approach remains efficient: there are only 8 experiments out of 115 requiring more than 3,000 seconds.

7 Conclusion

We have introduced soft skypatterns and proposed a flexible and efficient approach to mine skypatterns as well as soft ones thanks to Dynamic CSP. The relevance and the effectiveness of our approach have been highlighted through experiments on UCI datasets and a case study in chemoinformatics.

In the future, we would like to continue to investigate where the CP approach leads to new insights into the underlying data mining problems. Thanks to CP, we would particularly like to introduce softness within other tasks such as clustering, and study the contribution of soft skypatterns for recommendation.

Acknowledgments. We would like to thank Arnaud Soulet for providing the `Aetheris` program and his highly valuable comments. This work is partly supported by the ANR (French Research National Agency) funded project FiCoLoFo ANR-10-BLA-0214.

References

1. Bajorath, J., Auer, J.: Emerging chemical patterns: A new methodology for molecular classification and compound selection. J. of Chemical Information and Modeling 46, 2502–2514 (2006)
2. Bistarelli, S., Bonchi, F.: Soft constraint based pattern mining. Data Knowl. Eng. 62(1), 118–137 (2007)
3. Börzönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: 17th Int. Conf. on Data Engineering, pp. 421–430. Springer (2001)
4. De Raedt, L., Guns, T., Nijssen, S.: Constraint programming for itemset mining. In: KDD 2008, pp. 204–212. ACM (2008)
5. De Raedt, L., Zimmermann, A.: Constraint-based pattern set mining. In: 7th SIAM International Conference on Data Mining. SIAM (2007)
6. Gavanelli, M.: An algorithm for multi-criteria optimization in csps. In: van Harmelen, F. (ed.) ECAI, pp. 136–140. IOS Press (2002)
7. Guns, T., Nijssen, S., De Raedt, L.: Itemset mining: A constraint programming perspective. Artif. Intell. 175(12-13), 1951–1983 (2011)
8. Jin, W., Han, J., Ester, M.: Mining thick skylines over large databases. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 255–266. Springer, Heidelberg (2004)
9. Khiari, M., Boizumault, P., Crémilleux, B.: Constraint programming for mining n-ary patterns. In: Cohen, D. (ed.) CP 2010. LNCS, vol. 6308, pp. 552–567. Springer, Heidelberg (2010)
10. Kung, H.T., Luccio, F., Preparata, F.P.: On finding the maxima of a set of vectors. Journal of ACM 22(4), 469–476 (1975)
11. Lin, X., Yuan, Y., Zhang, Q., Zhang, Y.: Selecting stars: The *k* most representative skyline operator. In: ICDE 2007, pp. 86–95 (2007)
12. Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. Data Mining and K. Discovery 1(3), 241–258 (1997)
13. Matousek, J.: Computing dominances in E^n. Inf. Process. Lett. 38(5), 277–278 (1991)
14. Kralj Novak, P., Lavrac, N., Webb, G.I.: Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. Journal of Machine Learning Research 10, 377–403 (2009)
15. Papadias, D., Tao, Y., Fu, G., Seeger, B.: Progressive skyline computation in database systems. ACM Trans. Database Syst. 30(1), 41–82 (2005)
16. Papadias, D., Yiu, M., Mamoulis, N., Tao, Y.: Nearest neighbor queries in network databases. In: Encyclopedia of GIS, pp. 772–776 (2008)

17. Papadopoulos, A.N., Lyritsis, A., Manolopoulos, Y.: Skygraph: an algorithm for important subgraph discovery in relational graphs. Data Min. Knowl. Discov. 17(1), 57–76 (2008)
18. Poezevara, G., Cuissart, B., Crémilleux, B.: Extracting and summarizing the frequent emerging graph patterns from a dataset of graphs. J. Intell. Inf. Syst. 37(3), 333–353 (2011)
19. Soulet, A., Raïssi, C., Plantevit, M., Crémilleux, B.: Mining dominant patterns in the sky. In: ICDM, pp. 655–664 (2011)
20. Tan, K.-L., Eng, P.-K., Ooi, B.C.: Efficient progressive skyline computation. In: VLDB, pp. 301–310 (2001)
21. Ugarte, W., Boizumault, P., Loudni, S., Crémilleux, B.: Soft threshold constraints for pattern mining. In: Discovery Science, pp. 313–327 (2012)
22. Verfaillie, G., Jussien, N.: Constraint solving in uncertain and dynamic environments: A survey. Constraints 10(3), 253–281 (2005)