Mining (Soft-) Skypatterns Using Constraint Programming

Willy Ugarte, Patrice Boizumault, Samir Loudni, Bruno Crémilleux, Alban Lepailleur

Mining (Soft-) Skypatterns using Constraint Programming

Willy Ugarte, Patrice Boizumault, Samir Loudni,
Bruno Crémilleux, and Alban Lepailleur

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 Constraint Programming, 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 experiments on UCI benchmarks and a case study in chemoinformatics for discovering toxicophores.

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 determined effort has been made towards 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 [De Raedt and Zimmermann, 2007]. Using the dominance re-

Willy Ugarte · Patrice Boizumault · Samir Loudni · Bruno Crémilleux
GREYC (CNRS UMR 6072) – University of Caen
Campus II Côte de Nacre, 14000 Caen - France

Alban Lepailleur
CERMN (UPRES EA 4258 - FR CNRS 3038 INC3M) – University of Caen
Boulevard Becquerel, 14032 Caen Cedex - France

e-mail: firstname.lastname@unicaen.fr

lation is a recent trend in constraint-based data mining to produce useful pattern sets [Soulet et al., 2011].

Skyline queries [Börzsönyi et al., 2001] enable to express a user-preference point of view according to a *dominance* relation. Such queries have attracted considerable attention due to their importance in multi-criteria decision and are usually called *"Pareto efficiency or optimality queries"* 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 [Papadopoulos et al., 2008] 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*) [Soulet et al., 2011]. Briefly, given a set of measures, skypatterns are patterns based on a Pareto-dominance relation for which no measure can be improved without degrading the others. 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 $freq(x_i) \geq freq(x_j)$, $size(x_i) \geq size(x_j)$, $confidence(x_i) \geq confidence(x_j)$ 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 [Bistarelli and Bonchi, 2007, Ugarte et al., 2012] 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 [Verfaillie and Jussien, 2005]. Our proposition benefits from the recent progress on cross-fertilization between data mining and Constraint Programming (CP) [De Raedt et al., 2008, Khiari et al., 2010, Guns et al., 2011]. 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

Trans.	Items
t_1	B E F
t_2	B C D
t_3	A 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

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

and solved using Dynamic CSP. 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 describes experiments on UCI benchmarks and reports in depth a case study in chemoinformatics by performing both a performance and a qualitative analysis.

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-empty subset of $\mathscr{I}$. The language of itemsets corresponds to $\mathscr{L}_{\mathscr{I}} = 2^{\mathscr{I}} \setminus \emptyset$. 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 gathers articles 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* [Mannila and Toivonen, 1997]: $Th(q) = \{x \in \mathscr{L}_{\mathscr{I}} \mid q(x) \, is \, true\}$. A common example is the frequency measure leading to the minimal frequency constraint. The latter provides patterns x having a number of occurrences in the dataset exceeding a

given minimal threshold min_{fr}: $freq(x) \geq min_{fr}$. There are other usual measures for a pattern x:

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

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). We will use the growth rate, a well-known contrast measure [Novak et al., 2009]:

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 w.r.t. the set inclusion of an equivalence class is called a *closed pattern*.

Definition 2 (Closed pattern). A pattern $x_i \in \mathscr{L}_{\mathscr{I}}$ is closed w.r.t. a measure m iff $\forall x_j \in \mathscr{L}_{\mathscr{I}}$, $x_j \supsetneq x_i, \Rightarrow m(x_j) \neq m(x_i)$.

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, thus we define the constraint $closed_M(x)$ stating that x must be a closed pattern w.r.t all the measures of M.

2.2 Skypatterns

Skypatterns have been recently introduced by [Soulet et al., 2011]. Such patterns enable to express a user-preference point of view according to a dominance relation. 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 $freq(x_j) \geq freq(x_i), size(x_j) \geq size(x_i), confidence(x_j) \geq confidence(x_i)$ 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.

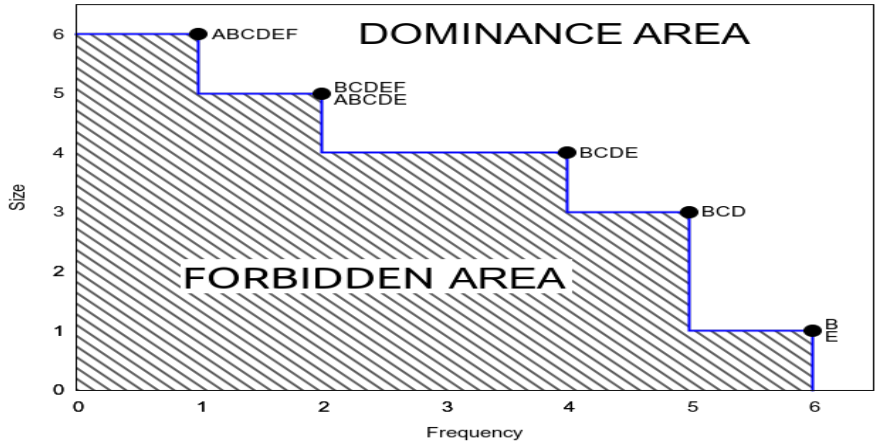


Fig. 1: Skypatterns extracted from the dataset shown in Table 1.

Given a set of measures M, if a pattern x_j is dominated by another pattern x_i according to all measures of M, x_j is considered as irrelevant. This idea is at the core of the notion of skypattern.

Definition 3 (Pareto 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. For M=$\{freq, area\}$, pattern BCD dominates pattern BC since $freq(BCD)$=$freq(BC)$=5 and $area(BCD)$>$area(BC)$. For M=$\{freq, size, average\}$, pattern BDE dominates pattern BCE since $freq(BDE)$ = $freq(BCE)$=4, $size(BDE)$=$size(BCE)$=3 and $average(BDE.price)$>$average(BCE.price)$.

Definition 4 (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 of P 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 Figure 1).

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 these two areas.

[Soulet et al., 2011] have proposed an efficient approach taking benefit of theoretical relationships between pattern condensed representations and skypatterns and making the process feasible when the pattern condensed representation can be extracted. Nevertheless, this method can only use a crisp dominance relation.

3 The soft skypattern mining problem

This section introduces the softness within the skypattern mining problem. The skypatterns suffer from the stringent aspect of the constraint-based framework. In order to introduce softness in this context, we propose 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. The goal is to capture valuable skypatterns occurring in the forbidden area.

3.1 Edge-skypatterns

Similarly to skypatterns, edge-skypatterns are defined according to a dominance relation and a *Sky* operator. These two notions are reformulated as follows:

Definition 5 (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 6 (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 of P not strictly dominated in P, with respect to M. The edge-skypattern operator $Edge\text{-}Sky(P,M)$ returns all the edge-skypatterns of P with respect to M:
$Edge\text{-}Sky(P,M) = \{x_i \in P \mid \nexists x_j \in P, x_j \gg_M x_i\}$

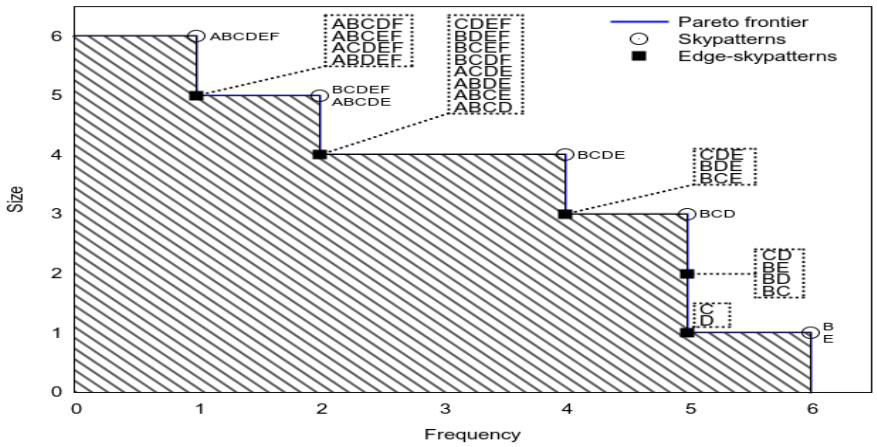


Fig. 2: Edge-skypatterns extracted from the dataset shown in Table 1.

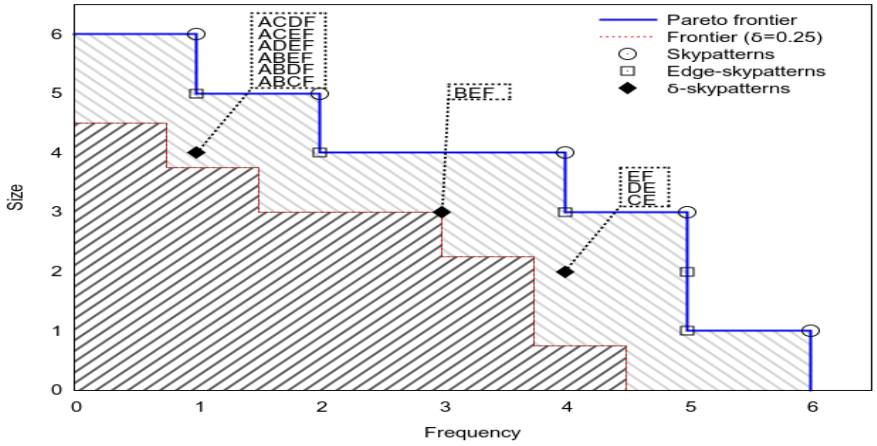


Fig. 3: δ-skypatterns (that are not edge ones) extracted from the dataset (Table 1).

Given a set of measures M, the edge-skypattern mining problem is thus to evaluate the query $Edge\text{-}Sky(P,M)$. Figure 2 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 (hard) skypatterns (see Figure 1).

Proposition 1. *For two patterns x_i and x_j, $x_i \gg_M x_j \implies x_i \succ_M x_j$.*

Proposition 2. *For a pattern set P and a set of measures M, $Sky(P,M) \subseteq Edge\text{-}Sky(P,M)$.*

Proofs are obvious and thus omitted.

3.2 δ-skypatterns

In many cases the user may be interested in skypatterns expressing a trade-off between the measures. The δ-skypatterns address this issue. Let $0 < \delta \leq 1$.

Definition 7 (δ-Dominance). Given a set of measures M, a pattern x_i δ-dominates another pattern x_j with respect to M (denoted by $x_i \succ_M^\delta x_j$), iff $\forall m \in M$, $(1-\delta) \times m(x_i) > m(x_j)$.

Definition 8 (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 of P not δ-dominated in P with respect to M. The δ-skypattern operator $\delta\text{-}Sky(P,M)$ returns all the δ-skypatterns of P with respect to M:
$\delta\text{-}Sky(P,M) = \{x_i \in P \mid \nexists x_j \in P : x_j \succ_M^\delta x_i\}$.

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. Figure 3 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 expressing the maximal relative distance between a skypattern and this border.

Proposition 3. *For two patterns x_i and x_j, $x_i \succ_M^{\delta} x_j \implies x_i \gg_M x_j$.*

Proposition 4. *For a pattern set P and a set of measures M, $Edge\text{-}Sky(P,M) \subseteq \delta\text{-}Sky(P,M)$.*

Proofs are obvious and thus omitted.

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

4 Mining (soft-) skypatterns using CP

This section describes how the skypattern and the soft skypattern mining problems can be modeled and solved using Dynamic CSP [Verfaillie and Jussien, 2005]. A major advantage of this 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. The purpose of adding constraints dynamically is to enlarge the forbidden area until it could not be expanded (Section 4.4 provides a detailed example).

Each time a solution is found, we dynamically post a new constraint leading to reduce the search space. This process stops when we cannot enlarge the forbidden area. Moreover, the declarative side of the CP framework easily enables us to manage constraints providing several kinds of softness and leads to a unified framework handling softness in the skypattern mining problem.

Our proposition benefits from the recent progress on cross-fertilization between data mining and CP [De Raedt et al., 2008, Khiari et al., 2010, Guns et al., 2011]. 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. The implementation of our approach has been carried out in `Gecode`[1].

Sections 4.1 and 4.2 briefly recall the notions of CSP and Dynamic CSP in Constraint Programming. Section 4.3 describes how mining skypatterns can can be performed using Dynamic CSP. Section 4.4 provides an example. Section 4.5 shows that soft skypatterns can be mined in the same way as skypatterns. Section 4.6 presents the boolean pattern encoding. Finally, Section 4.7 is devoted to closedness constraints.

[1] http://www.gecode.org/

4.1 CSP

A CSP P=$(\mathscr{X},\mathscr{D},\mathscr{C})$ is defined by:

- a finite set of variables $\mathscr{X} = \{x_1, x_2, \ldots, x_k\}$,
- a domain $\mathscr{D}$, which maps every variable $x_i \in \mathscr{X}$ to a finite set of values $D(x_i)$,
- a finite set of constraints $\mathscr{C}$.

The problem is to find a mapping from variables to values such that each variable x_i is mapped to a value in its domain $D(x_i)$ and such that all constraints of $\mathscr{C}$ are satisfied.

Algorithm 1 shows how a CSP can be solved using a depth-first search. D and C denote respectively the current domains and the current set of constraints. In each node of the search tree, the algorithm branches by assigning values to a variable that is unfixed (line 7). It backtracks when a violation of constraints is found, i.e. at least one domain is empty (line 2). The search is further optimized by carefully choosing the variable that is fixed next (line 6); for instance, heuristics dom/deg selects the variable x_i having the smallest ratio between the size of its current domain and the number of constraints it occurs.

Algorithm 1 Depth-First(D)

```
1: D ← Filter(D,C)
2: if there exists x_i ∈ 𝒳 s.t. D(x_i) is empty then
3:     return failure
4: end if
5: if there exists x_i ∈ 𝒳 s.t. |D(x_i)| > 1 then
6:     Select x_i ∈ 𝒳 s.t. |D(x_i)| > 1
7:     for all v ∈ D(x_i) do
8:         Depth-First(D ∪ {x_i− > {v}})
9:     end for
10: else
11:     Manage-Solution(D,C)
12: end if
```

The main concept used to speed-up the search is filtering (procedure $Filter(D,C)$ at line 1). Filtering reduces the domains of variables such that the domain remains locally consistent. A solution is obtained (line 11) when each domain $D(x_i)$ is reduced to a singleton and all constraints are satisfied. For CSP, $Manage\text{-}Solution(D,C)$ simply consists in outputting the obtained solution D.

4.2 Dynamic CSP

A Dynamic CSP [Verfaillie and Jussien, 2005] 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 (addings or removals), domains (value addings or removals), constraints (addings or removals).

For our approach, *variables and domains remain the same* and the changes are only performed by *adding new constraints*. Solving such Dynamic CSP can be considered as a backtracking algorithm that, each time a new solution is found, imposes new constraints $\phi(\mathscr{X})$ that survive backtracking, stating that next solutions should verify both the current set of constraints and $\phi(\mathscr{X})$. Dynamic constraints $\phi(\mathscr{X})$ are added to the constraint store (see Algorithm 2) in order to hold in all the branches of the search tree, surviving backtracking. (Note that C is a global variable to all calls to the *Depth-First* procedure).

Algorithm 2 Manage-Solution(D, C)

1: Output solution D
2: $C \leftarrow C \cup \{\phi(\mathscr{X})\}$

4.3 Mining skypatterns using Dynamic CSP

This subsection describes our CP approach for mining both skypatterns and soft skypatterns. Constraints on the dominance relation are dynamically posted during the mining process and softness is easily introduced using such constraints. The purpose of adding constraints dynamically is to enlarge the forbidden area until it could not be expanded.

Variable x will denote the (unknown) skypattern we are looking for. Changes are only performed by adding new constraints (see Section 4.2). So, we consider the sequence $P_1, P_2, ..., P_n$ of CSP where each $P_i = (\{x\}, \mathscr{L}, q_i(x))$ and:

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

First, the constraint $closed_M(x)$ states that x must be a closed pattern w.r.t all the measures of M (see Definition 2). It allows to reduce the number of redundant patterns[2].

Then, the constraint $\phi(s_i, x) \equiv \neg(s_i \succ_M x)$ 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, ..., s_i.

Each time the first solution s_i to query $q_i(x)$ is found, we dynamically post a new constraint $\phi(s_i, x)$, based on the values of the measures for s_i, leading to reduce the

[2] The *closed* constraint is used to reduce pattern redundancy. Indeed, **closed skypatterns** make up an exact condensed representation of the whole set of skypatterns [Soulet et al., 2011].

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

Item	A	B	C	D	E	F
Price	10	55	70	30	15	25

Table 2: Example of a toy dataset.

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, $\phi(s_i,x)$ states that $\neg(s_i \succ_M x)$ (see Definition 3):

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

However, 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. Filter all patterns $s_i \in S$ that are not skypatterns.

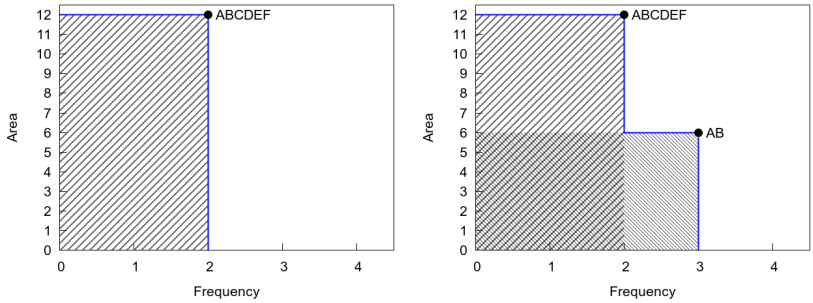
While the number of candidates (n) could be very large, it remains reasonably-sized in practice for the experiments we conducted (see Table 3 and Table 4 for UCI benchmarks, and Table 5 for the case study in chemoinformatics for discovering toxicophores).

4.4 Example

This subsection gives an example of computing skypatterns using a Dynamic CSP and shows how the forbidden area is successively enlarged. We consider the dataset depicted in Table 2 and the set of measures $M = \{freq, area\}$.

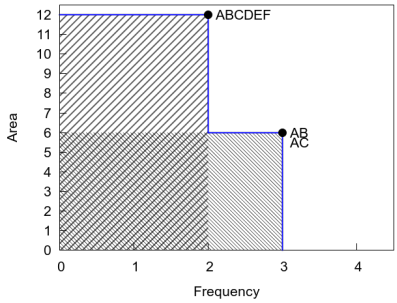
Let P_1 be the associated Dynamic CSP (see Section 4.3). $P_1 = (\{x\}, \mathscr{L}, q_1(x))$ where query $q_1(x) = closed_M(x)$. Its first solution is pattern $s_1 = ABCDEF$ (with $freq(s_1) = 2$ and $area(s_1) = 12$), see Figure 4a.

So, we consider query $q_2(x) = closed_M(x) \wedge \neg(s_1 \succ_M x)$ stating that we are looking for a closed pattern x not dominated by $s_1 = ABCDEF$. Its first solution is pattern $s_2 = AB$ (with $freq(s_2) = 3$ and $freq(s_2) = 6$), see Figure 4b.

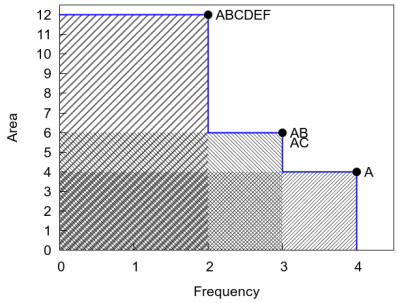


(a) 1st solution: $s_1 = ABCDEF$.

(b) 2nd solution: $s_2 = AB$.



(c) 3rd solution: $s_3 = AC$.



(d) Last solution: $s_4 = A$.

Fig. 4: Solving the toy example using Dynamic CSP.

Then, the next query is $q_3(x) = closed_M(x) \wedge \neg(s_1 \succ_M x) \wedge \neg(s_2 \succ_M x)$ stating that we are looking for a closed pattern x neither dominated by s_1 nor s_2. Its first solution is pattern $s_3 = AC$ (with $freq(s_3) = 3$ and $area(s_3) = 6$), see Figure 4c.

The next query is $q_4(x) = q_3(x) \wedge \neg(s_3 \succ_M x)$ whose first solution is $s_4 = A$ (see Figure 4d) and then query $q_5(x) = q_4(x) \wedge \neg(s_4 \succ_M x)$. $q_5(x)$ has no solution since the dominated area cannot be enlarged. So, the process ends for $n = 5$.

In this example, note that all extracted patterns are skypatterns (i.e., there is no intermediate patterns). The CSP system did not generate solution that does not satisfy the dominance relation. Experiments in the next section provide examples with intermediate patterns.

4.5 Mining soft skypatterns using Dynamic CSP

Soft skypatterns are processed exactly the same way as skypatterns (see Section 4.3). Each kind of soft skypatterns has its own constraint $\phi(s_i,x)$ according to its relation of dominance.

For edge-skypatterns, $\phi(s_i,x)$ states that $\neg(s_i \gg_M x)$ (see Definition 5):

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

For δ-skypatterns, $\phi(s_i,x)$ states that $\neg(s_i \succ_M^{\delta} x)$ (see Definition 7):

$$\phi(s_i,x) \equiv \bigvee_{m\in M} (1-\delta) \times m(s_i) \leq m(x)$$

However, the n extracted patterns s_1, s_2, …, s_n are not necessarily all soft skypatterns. Some of them can only be "intermediate" patterns simply used to enlarge the forbidden area. So, a post processing is required as for skypatterns (see Section 4.3). Mining soft skypatterns is also achieved in a two-steps approach:

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

Once again, the number of candidates (n) remains reasonably-sized in practice for the experiments we conducted (see Table 4 for UCI benchmarks, and Table 5 for toxicophores).

4.6 Pattern encoding

We now introduce the model of a pattern that can be provided to the constraint programming system. Let d be the 0/1 matrix where, 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. [De Raedt et al., 2008, Guns et al., 2011] 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 $(X_i = 1)$ iff $(i \in x)$,
- $\{T_t \mid t \in \mathscr{T}\}$ where $(T_t = 1)$ iff $(x \subseteq t)$.

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 x is a subset of t:

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

A reified constraint associates a 0/1 variable to a constraint reflecting whether the constraint is satisfied (value 1) or not (value 0). Such constraints are useful for expressing propositional formulas over constraints and for expressing that a certain number of constraints hold.

Reified constraints do not enjoy the same level of propagation as simple constraints, but if the solver deduces $T_t = 1$ (resp. $T_t = 0$), then the sum must be equal to 0 (resp. must be different from 0). The propagation is also performed, in a same way, from the sum constraint toward the equality constraint. Using these reified constraints, some measures are easy to encode: $freq(x) = \sum_{t \in \mathscr{T}} T_t$ and $size(x) = \sum_{i \in \mathscr{I}} X_i$. The minimal frequency constraint $freq(x) \geq \theta$ (where θ is a threshold) is encoded by the constraint $\sum_{t \in \mathscr{T}} T_t \geq \theta$.

4.7 Closedness constraints

This subsection provides the encoding of closedness constraints (see Definition 2).

Let M=$\{freq\}$, the closedness constraint ensures that a pattern x has no superset with the same frequency. So, x is a closed pattern w.r.t. the measure *freq* iff:

$$\forall i \in \mathscr{I}, (X_i = 1) \Leftrightarrow \sum_{t \in \mathscr{T}} T_t \times (1 - d_{t,i}) = 0 \tag{2}$$

Let M=$\{min\}$, and *val* be an attribute (e.g. see Table 1 where *val*=*price*). Let $min(x.val)$ be the smallest value of the item values of x for attribute *val* (see Section 2.1). If item i belongs to x, then its value for attribute *val* ($i.val$) must be greater than or equal to the minimal value $min(x.val)$. Conversely, if $i.val$ is greater than or equal to $min(x.val)$, i must belong to x (if not, x would not be maximal for inclusion). So, x is a closed pattern w.r.t. the measure *min* iff:

$$\forall i \in \mathscr{I}, (X_i = 1) \Leftrightarrow i.val \geq min(x.val) \tag{3}$$

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 [Soulet et al., 2011]. 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

The notion of dominance that we introduced in Section 2.2 is at the core of the skyline processing and the recent notion of skypattern that integrates into the pattern discovery process the idea of skylines.

Computing skylines is a derivation from the maximal vector problem in computational geometry [Matousek, 1991], the Pareto frontier [Kung et al., 1975] and multi-objective optimization [Steuer, 1992].

Since its rediscovery within the database community by [Börzsönyi et al., 2001], several methods have been developed for answering skyline queries [Börzsönyi et al., 2001, Papadias et al., 2005, Papadias et al., 2008, Tan et al., 2001]. These methods assume that tuples are stored in efficient tree data structures, such as *B-Tree* (allowing search and sequential access in logarithmic time) or *R-Tree* (for indexing multi-dimensional information). Alternative approaches have also been proposed towards helping the user in selecting most significant skylines. For example, [Lin et al., 2007] measure this significance by means of the number of points dominated by a skyline.

Introducing softness for skylines. [Jin et al., 2004] 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: $O(2^{|\mathscr{S}|})$ instead of $O(|\mathscr{T}|)$ for skylines.

There are only very few works dealing with skypatterns. As already said, [Soulet et al., 2011] have proposed an approach taking benefit of theoretical relationships between pattern condensed representations and skypatterns and making the process feasible when the pattern condensed representation can be extracted. To the best of our knowledge, it is the single work addressing a large set of measures. Nevertheless, this method only uses a crisp dominance relation. Other works address specific measures. A trade-off between quality and diversity measures is introduced in [van Leeuwen and Ukkonen, 2013] for subgroup discovery. [Papadopoulos et al., 2008] and [Shelokar et al., 2013] are interested in graph analysis. [Papadopoulos et al., 2008] discover subgraphs maximizing the number of vertices and the edge connectivity whereas [Shelokar et al., 2013] adapt the framework of the subdue method to the extraction of graph patterns satisfying the Pareto dominance on two to three measures.

CP for computing the Pareto frontier. [Gavanelli, 2002] 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 [Gavanelli, 2002] to a constraint solver would require an important development time compared to the expected benefits.

6 Experimental study

In this section, we report an experimental study on several benchmarks and a case study from chemoinformatics.

6.1 Experiments on UCI benchmarks

This section compares our approach (noted `CP+SKY`) with `Aetheris`, which is the only other method able to mine skypatterns [Soulet et al., 2011]. As our proposal, `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 sky operator (see Definition 4) is applied.

Experiments we performed on UCI datasets show that:

1. `CP+SKY` and `Aetheris` obtain similar CPU-times for mining skypatterns (see Section 6.1.2).
2. As the number of extracted skypatterns is very low, mining soft skypatterns enables to emphasize interesting knowledge that could be missed by skypatterns (see Section 6.1.3).

6.1.1 Experimental protocol

We carried out experiments on several datasets from the UCI repository[3]. We considered two sets of measures: M_1=$\{frequency, growth\text{-}rate, area\}$ and M_2 = $\{frequency, max, area, mean\}$. 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 both steps.

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` CPU-times are obtained by the programs kindly provided by A. Soulet and used in [Soulet et al., 2011]. The implementation of `CP+SKY` was carried out in `Gecode` by extending the CP-based patterns extractor developed by [Khiari et al., 2010].

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

6.1.2 Mining skypatterns

Table 3 compares CP+SKY with Aetheris on several datasets for the two sets of measures M_1 and M_2. For each dataset, and each set of measures, we report:

- the number[4] of skypatterns,

Dataset				$M_1 = \{freq, growth\ rate, area\}$					$M_2 = \{freq, max, area, mean\}$				
					CP+SKY		Aetheris			CP+SKY		Aetheris	
	# items	# transactions	density	# of Skypatterns	# of Candidates	Time (sec)	# of Closed Patterns	Time (sec)	# of Skypatterns	# of Candidates	Time (sec)	# of Closed Patterns	Time (sec)
abalone	28	4,178	0.321	29	2,669	11	38,996	1	44	5,094	16	30,800	1
anneal	68	798	0.195	38	555	1	9,601	1	76	14,098	12	66,974	2
austral	55	690	0.272	22	9,015	6	136,010	4	38	16,231	18	785,005	19
breast	43	286	0.231	13	801	1	6,210	1	17	1,471	1	25,702	1
cleve	43	303	0.325	22	5,896	3	64,016	2	27	8,387	6	242,228	6
cmc	28	1,474	0.357	38	4,209	2	48,923	1	6	8,749	10	69,306	1
crx	59	690	0.269	27	10,035	6	130,768	5	15	35,931	31	1,035,271	29
german	76	1,000	0.276	35	111,099	26	3,215,841	118	40	56,973	120	6,198,069	182
glass	34	216	0.295	24	1,332	1	44,605	1	18	1,112	1	17,660	1
heart	38	270	0.368	26	6,018	2	58,706	1	20	8,662	4	237,586	5
hepatic	45	155	0.421	34	8,209	1	100,105	4	84	40,450	11	643,543	14
horse	75	300	0.235	14	3,968	6	124,368	6	16	15,446	20	1,244,224	35
hypo	47	3,163	0.389	209	273,430	244	673,102	65	37	198,459	435	2,316,785	323
iris	15	151	0.333	3	63	1	287	1	10	111	1	231	1
lymph	59	142	0.322	18	4,359	1	38,888	1	83	18,477	6	408,507	11
mushroom	119	8,124	0.193	25	1,130	550	227,699	24	102	9,380	594	2,736,405	230
new-thyroid	21	216	0.287	7	99	1	593	1	14	161	1	1,218	1
page	35	941	0.314	14	1,197	2	32,904	1	26	2,251	3	94,512	1
pima	26	768	0.346	14	786	1	14,798	1	15	217	1	42,554	1
tic-tac-toe	29	259	0.344	26	4,906	3	42,711	1	9	4,499	6	95,798	2
vehicle	58	846	0.327	57	29,088	6	358,357	16	106	65,400	73	2,291,888	71
wine	45	179	0.311	13	2,129	1	24,010	1	35	6,438	3	113,886	2
zoo	43	102	0.394	33	1,199	1	4,567	1	54	4,290	1	34,588	1

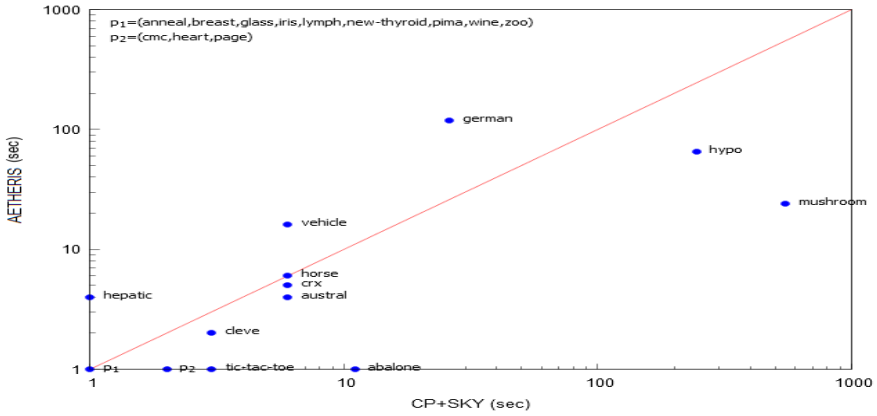
Density of a dataset: The ratio of the number of present items in the dataset (i.e. $\sum_{i \in \mathscr{I}} \sum_{t \in \mathscr{T}} d_{i,t}$) w.r.t. the size of the dataset (i.e. $|\mathscr{I}| \times |\mathscr{T}|$)

Table 3: Comparing CP+SKY with Aetheris on UCI Benchmarks.

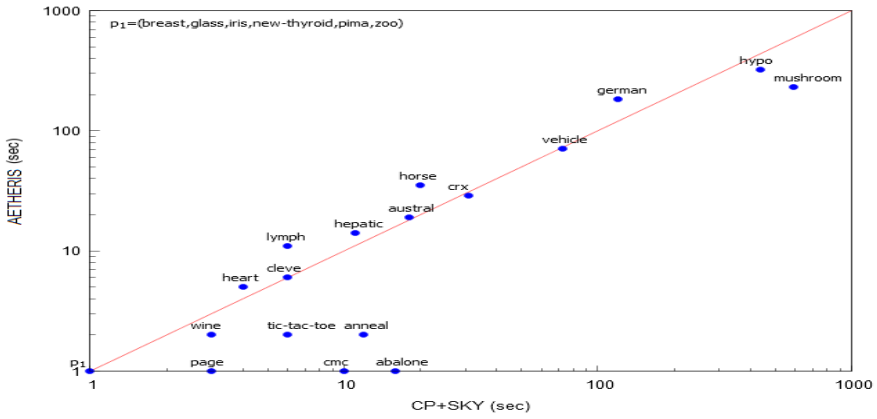
[4] Obviously, it is the same for both methods.

- for CP+SKY, the number of candidates (i.e. intermediate patterns required to determine the forbidden area (see Section 4.3)) and the associated CPU-time,
- for Aetheris, the number of closed patterns of the condensed representation and the associated CPU-time.

First, the number of skypatterns is always very low. There are less than 40 skypatterns for M_1 except for vehicle (57) and hypo (209). It is the same for M_2, for which the number of skypatterns does not exceed 106, thus highlighting the interest of extracting soft skypatterns. Second, on most of the datasets (except for german and hypo for M_1, and german, hypo and mushroom for M_2), required CPU-times for mining all the skypatterns are very low (less than one minute). Third, CP+SKY and Aetheris perform quite similarly on most of the datasets. For the dataset german (resp. vehicle), with M_1, CP+SKY is 4.5 (resp. 2.5) faster



(a) $M_1 = \{frequency, growth\text{-}rate, area\}$



(b) $M_2 = \{frequency, max, area, mean\}$

Fig. 5: Scatter plots comparing CPU-times on UCI datasets.

than Aetheris, while on the three datasets abalone, hypo and mushroom, Aetheris is clearly better (with a factor of 3).

Figure 5a and Figure 5b provide two scatter plots of CPU-times for CP+SKY and Aetheris (see Table 3). Each point represents a dataset: its x-value (log-scale) is the CPU-time for CP+SKY to mine it, its y-value (log scale) is the CPU-time for Aetheris to mine it. A point at the beginning of an axis means that the considered approach requires 1 second or less to mine it.

Figure 5a and Figure 5b show that CP+SKY and Aetheris obtain similar CPU-times. For M_1, CP+SKY is faster than Aetheris on three datasets (e.g., points above the red line, i.e. $y{=}x$). On the other hand, Aetheris clearly dominates CP+SKY on three datasets (e.g. points near the right border of the figure). For the other datasets, the two approaches are quite similar (e.g., points near the red line and points in the bottom of the figure). For M_2, most of the points tend to concentrate in the vicinity of the red line: CP+SKY and Aetheris solve many datasets within similar CPU-times. Moreover, the gap between the two methods on datasets hypo and mushroom for M_2 is greatly reduced with respect to M_1.

These results show that our approach, though integrating softness, obtains similar performances as Aetheris for mining skypatterns.

6.1.3 Mining soft skypatterns

This section shows the feasibility of mining soft skypatterns on UCI Benchmarks (for these experiments, parameter δ was set to 5%). 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, and each set of measures:

- for edge-skypatterns: their number[5], the number of candidates and the required CPU-time,
- for δ-skypatterns: their number[6], the number of candidates and the required CPU-time.

Finally, our CP-based approach enables to mine both skypatterns and soft ones in a same way. This cannot be performed by [Soulet et al., 2011] that can only handle a crisp dominance relation.

6.2 Case study: discovering toxicophores

Toxicology is a scientific discipline involving the study of the toxic effects of chemicals on living organisms. A major issue in chemoinformatics is to establish re-

[5] They correspond to edge-skypatterns that are not hard skypatterns.

[6] They correspond to δ-skypatterns that are neither hard skypatterns neither edge-skypatterns.

Dataset				$M_1 = \{freq, growth\ rate, area\}$						$M_2 = \{freq, max, area, mean\}$					
				CP+Edge-Sky			CP+δ-Sky ($\delta = 5\%$)			CP+Edge-Sky			CP+δ-Sky ($\delta = 5\%$)		
	# items	# transactions	Density	# of Edge-skypatterns	# of Candidates	Time (sec)	# of δ-skypatterns	# of Candidates	Time (sec)	# of Edge-skypatterns	# of Candidates	Time (sec)	# of δ-skypatterns	# of Candidates	Time (sec)
abalone	28	4,178	0.321	834	15,896	38	39	18,031	38	3,447	8,718	24	2,422	11,730	29
anneal	68	798	0.195	38	4,843	3	3	5,769	3	1,495	25,275	20	6,952	33,909	26
austral	55	690	0.272	14	30,402	27	17	47,288	34	10,847	59,980	43	67,904	124,531	74
breast	43	286	0.231	49	2,496	1	3	3,211	1	2,184	4,044	1	2,585	7,131	1
cleve	43	303	0.325	15	22,407	8	6	31,393	10	15,613	27,102	10	9,324	45,301	14
cmc	28	1,474	0.357	87	18,802	15	16	28,970	21	19,737	28,049	29	4,685	32,330	30
crx	59	690	0.269	10	30,851	26	27	49,081	34	51,912	159,831	116	134,398	279,219	150
german	76	1,000	0.276	164	400,028	650	54	945,432	1647	96,743	377,821	477	176,616	835,909	782
glass	34	216	0.295	29	5,160	1	16	5,737	1	706	4,103	1	2,091	4,974	1
heart	38	270	0.368	11	16,472	6	8	27,841	8	20,958	36,740	11	31	63,097	16
hepatic	45	155	0.421	1179	35,111	7	21	43,155	8	20,846	119,983	35	17,105	157,261	33
horse	75	300	0.235	12	17,606	15	3	26,412	18	13,938	69,042	39	50,151	98,491	45
hypo	47	3,163	0.389	9147	472,434	823	2,918	672,019	875	215,789	764,023	8,256	765,715	1,956,947	22789
iris	15	151	0.333	27	128	1	2	128	1	46	129	1	32	151	1
lymph	59	142	0.322	31	8,399	2	7	13,522	3	6,886	74,370	19	66,930	97,640	20
mushroom	119	8,124	0.193	28	11,314	609	114	12,761	614	209,695	600,450	7,021	378,536	719,762	18360
new-thyroid	21	216	0.287	5	247	1	1	277	1	98	371	1	61	418	1
page	35	941	0.314	10	3,775	5	3	5,634	7	3,483	8,207	8	8,397	15,154	13
pima	26	768	0.346	7	8,120	5	13	12,320	6	2,438	2,566	2	80	5,143	3
tic-tac-toe	29	259	0.344	95	27,478	19	26	37,182	22	11,473	17,087	13	6,730	39,053	24
vehicle	58	846	0.327	366	226,070	162	111	296,149	211	73,583	194,280	195	531	460,716	372
wine	45	179	0.311	130	17,831	3	9	18,970	3	2,825	18,883	5	4,879	25,119	5
zoo	43	102	0.394	55	3,779	1	41	3,951	1	2,301	13,286	2	4,096	14,263	2

Table 4: Analysis of soft skypattern mining on UCI benchmarks.

lationships between chemicals and a given activity (e.g., CL50[7] in ecotoxicity). Chemical fragments[8] which cause toxicity are called *toxicophores* and their discovery is at the core of prediction models in (eco)toxicity [Auer and Bajorath, 2006, Poezevara et al., 2011]. The aim of this present study, which is part of a larger research collaboration with the CERMN Lab, a laboratory of medicinal chemistry, is to investigate the use of softness for discovering toxicophores.

[7] Lethal concentration of a substance required to kill half the members of a tested population after a specified test duration.

[8] A fragment denominates a connected part of a chemical structure containing at least one chemical bond.

6.2.1 Experimental protocol

The dataset is collected from the ECB web site[9]. 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, 372 from the H400 class and 195 from the H402 class. The chemicals are encoded using 1,450 frequent closed subgraphs previously extracted from $\mathscr{T}$[10] with a 1% relative frequency threshold.

In order to discover patterns as candidate toxicophores, we use both measures typically used in contrast mining [Novak et al., 2009] such as the growth rate since toxicophores are linked to a classification problem with respect to the HSC and measures expressing the background knowledge such as the aromaticity or rigidity because chemists consider that this information may yield promising candidate toxicophores. Our method offers a natural way to simultaneously combine in a same framework these measures coming from various origins. We briefly sketch these measures and the associated threshold constraints.

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 the toxicity: if a chemical has, in its structure, fragments that are related to a toxic effect, then it is more likely to be toxic. Emerging patterns embody this natural idea by using the growth-rate measure (see Definition 1).

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 denote by m_a the aromaticity measure of a pattern.

Redundancy is reduced by using **closed skypatterns** which are an exact condensed representation of the whole set of skypatterns (see Footnote 2). We consider four sets of measures: M_1, M_2, M_3 and M_4 (see Table 5). For δ-skypatterns, we consider two values: δ=10% and δ=20%. The extracted skypatterns and soft skypatterns are made of molecular fragments. To evaluate the presence of toxicophores in their description, an expert analysis leads to the identification of well-known environmental toxicophores. A few examples are depicted in Table 6.

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

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

	Skypatterns				
		CP+SKY		Aetheris	
	# of Skypatterns	# of Candidates	CPU-Time	# of closed patterns	CPU-Time
M_1={*growth-rate*, *freq*}	**8**	613	**18m:34s**	41,887	**19m:20s**
M_2={*growth-rate*, *aromaticity*}	**5**	140	**15m:32s**	53,201	**21m:33s**
M_3={*freq*, *aromaticity*}	**2**	456	**16m:45s**	157,911	**21m:16s**
M_4={*growth-rate*, *freq*, *aromaticity*}	**21**	869	**17m:49s**	12,126	**21m:40s**

(a) Skypatterns

	Edge-Skypatterns			δ-Skypatterns					
				$\delta = 10\%$			$\delta = 20\%$		
	CP+Edge-SKY			CP+δ-SKY					
	# of Edge-skypatterns	# of Candidates	CPU-Time	# of δ-skypatterns	# of Candidates	CPU-Time	# of δ-skypatterns	# of Candidates	CPU-Time
M_1={*growth-rate*, *freq*}	**24**	1,746	**19m:02s**	**25**	4,204	**20m:48s**	**87**	6,253	**22m:36s**
M_2={*growth-rate*, *aromaticity*}	**76**	688	**17m:51s**	**354**	1,678	**18m:14s**	**1,670**	2,816	**23m:44s**
M_3={*freq*, *aromaticity*}	**72**	1,726	**16m:50s**	**352**	4,070	**19m:43s**	**1,654**	6,699	**22m:25s**
M_4={*growth-rate*, *freq*, *aromaticity*}	**144**	3,021	**20m:27s**	**385**	6,048	**23m:36s**	**1,724**	8,986	**30m:14s**

(b) Soft skypatterns

Table 5: Analysis of (soft-) skypattern mining on ECB dataset.

6.2.2 Mining the (soft-) skypatterns

This section evaluates the interest of using (soft-) skypatterns for discovering toxicophores. Table 5a compares CP+SKY with Aetheris for different combinations of measures. For each set of measures, we report:

- the number of skypatterns,
- for CP+SKY, the number of candidates (i.e. the number of intermediate patterns, see Section 4.3) and the associated CPU-time,
- for Aetheris, the number of closed patterns of the condensed representation and the associated CPU-time.

Table 5b reports, for each set of measures:

- for edge-skypatterns: their extra-number (see footnote #7), the number of candidates and the required CPU-time,
- for δ-skypatterns: their extra-number (see footnote #8), the number of candidates and the required CPU-time.

`CP+SKY` outperforms `Aetheris` in terms of CPU-times (see Table 5a). Moreover, the number of candidates generated by our approach remains small compared to the number of closed patterns computed by `Aetheris`. Thanks to constraints added dynamically, our CP approach enables to drastically reduce the number of candidates.

Moreover, increasing the number of measures leads to a higher number of (soft-) skypatterns, particularly for high values of δ. In fact, a pattern rarely dominates all other patterns on the whole set of measures. Nevertheless, in our experiments, 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 (for $\delta = 20\%$). Moreover, regarding the CPU-times, our approach is very effective: the soft skypatterns computation requires less than 30 minutes.

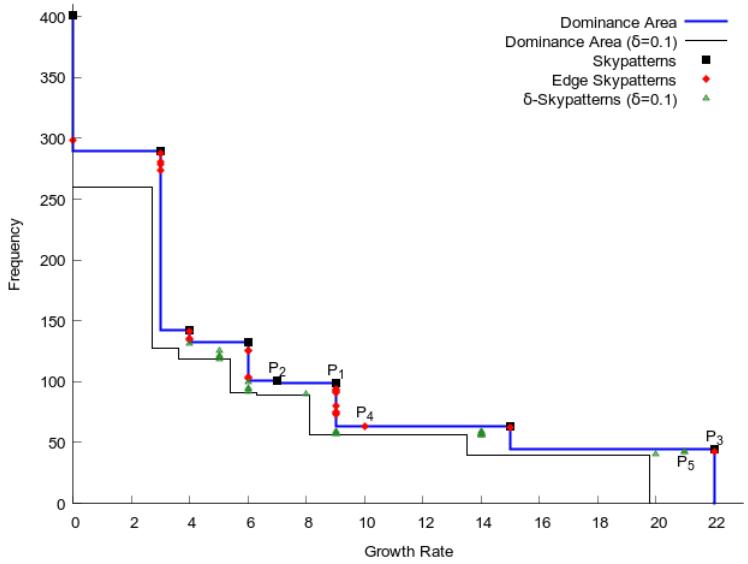
6.2.3 Qualitative Analysis

In this section, we analyse qualitatively the (soft-) skypatterns by evaluating the presence of toxicophores in their description, according to well-known environmental toxicophores.

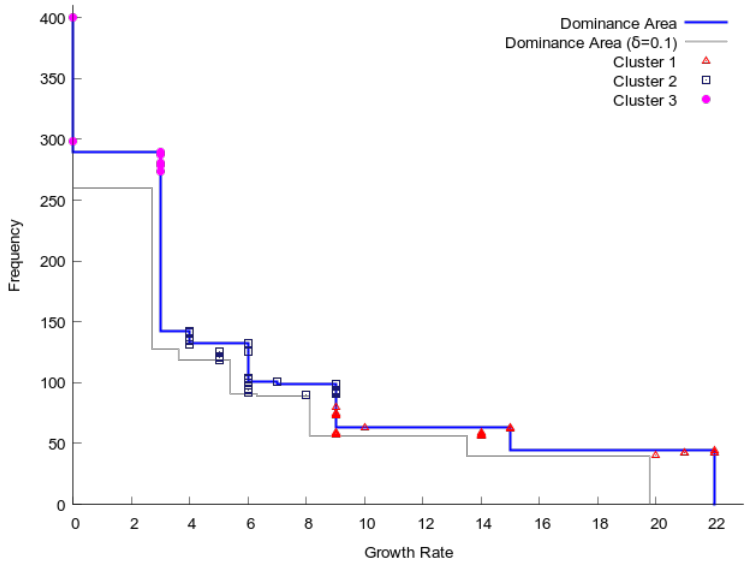
For M_1=$\{growth\text{-}rate, frequency\}$, soft skypatterns enable to efficiently detect well-known toxicophores emphasized by skypatterns, while for M_2=$\{growth\text{-}rate, aromaticity\}$ and M_4=$\{growth\text{-}rate, frequency, aromaticity\}$, soft skypatterns enable to discover (new) interesting toxicophores that would not be detected by skypatterns.

(a) Growth rate and frequency measures (M_1)**.** Figure 6a shows the distribution of (soft-) skypatterns for M_1.

- **Skypatterns**. Only 8 skypatterns are found, and 3 well-known toxicophores are emphasized. Two of them are aromatic compounds, namely the chlorobenzene (pattern p_1: {`Clc`}) and the phenol rings (pattern p_2: {`c1(ccccc1)O`}). The contamination of water and soil by organic aromatic chemicals is widespread as a result of industrial applications ranging from their use as pesticides, solvents to explosives and dyestuffs. Many of them may bioaccumulate in the food chain and have the potential to be harmful to living systems including humans, animals, and plants. The third one, the organophosphorus moiety (pattern p_3: {`OP`, `OP=S`}) is a component occurring in numerous pesticides.
- **Soft skypatterns** confirm the trends given by skypatterns. However, the chloro-substituted aromatic rings (e.g. pattern p_4: {`Clc(ccc)c`, `Clcccc`}), and the organophosphorus moiety (e.g. pattern p_5: {`OP(=S)O)`, `COP(=S)O`}) are detected by the edge-skypatterns and by the δ-skypatterns. Indeed, several patterns containing these toxicophores are extracted.



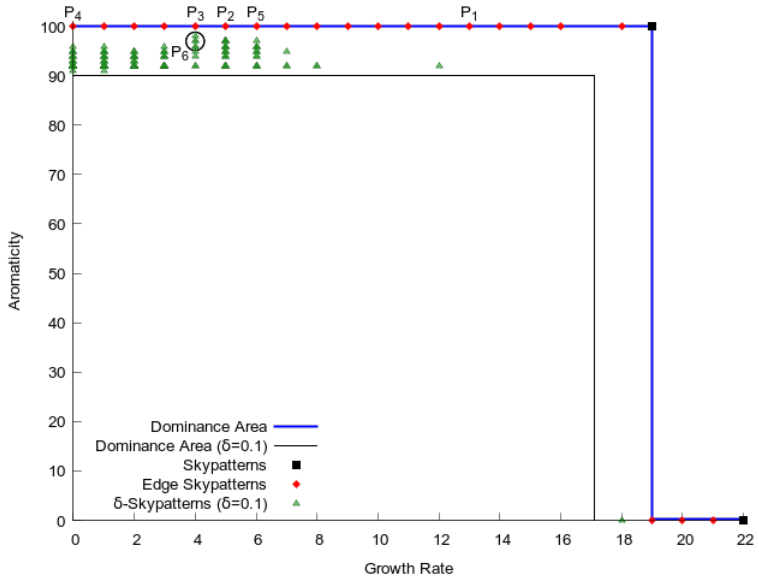
(a) Distribution of the (soft-) skypatterns.



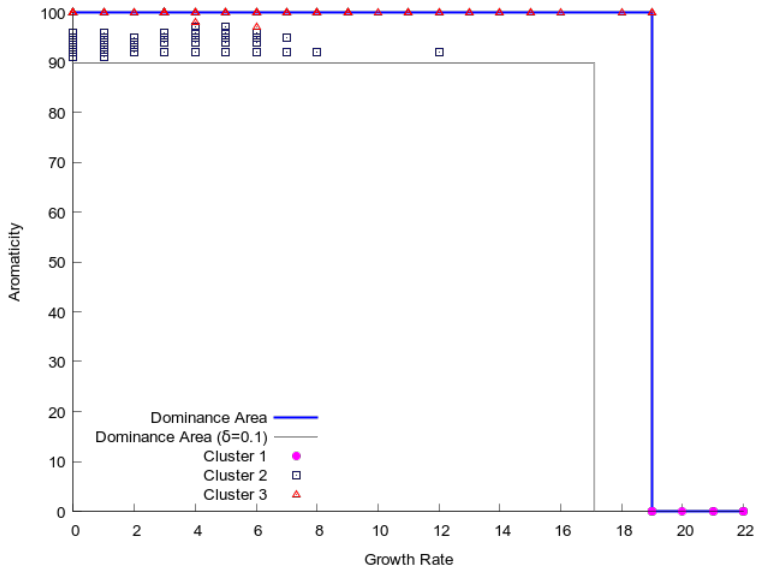
(b) Clustering using *k-means* (k=3).

Fig. 6: Analysing the (soft-) skypatterns for M_1 .

In order to partition the (soft-) skypatterns, we used the *k-means* clustering method with the euclidean metric. The computed solution consists in k=3 distinct clusters. Figure 6b highlights these clusters:



(a) Distribution of the (soft-) skypatterns.



(b) Clustering using *k-means* (k=3).

Fig. 7: Analysing the (soft-) skypatterns for M_2 .

1. Cluster #1 is made of patterns with a high growth rate and a low frequency. It contains 2 skypatterns and 23 soft ones: 8 of them are edge-skypatterns and 15 are δ-skypatterns. From a chemical point of view, most of these patterns contain the organophosphorus moiety and few sub-fragments of *alkyl-substituted benzene* (e.g. {ccC, cccC}).
2. Cluster #2 contains 4 skypatterns and 21 soft ones: 11 of them are edge-skypatterns and 10 are δ-skypatterns. From a chemical point of view, it emphasizes two well-known toxicophores, namely the chlorobenzene and the phenol rings.
3. Cluster #3 comprises 2 skypatterns and 5 edge-skypatterns. Most of them are aromatic compounds, namely the benzene ring (i.e. with high frequency and low growth rate).

(b) Growth rate and aromaticity measures (M_2)**.** As results for M_2 and M_3 are similar, Figure 7a only reports the distribution of the (soft-) skypatterns for M_2.

- **Skypatterns** for M_2 are less informative than the ones mined for M_1.
- **Soft skypatterns** lead to the discovery of several different aromatic rings. In fact, the nature of these chemicals can vary in function of i) the presence/absence of heteroatoms (e.g. N, S), ii) the number of rings, and iii) the presence/absence of substituents. Regarding the two kinds of soft skypatterns:
  - edge-skypatterns lead to the extraction of (i) *nitrogen aromatic compounds*: indole (pattern p_1: {ncc, c1ccccc1}) and benzoimidazole (pattern p_2: {ncnc, c1ccccc1}), (ii) *S-containing aromatic compounds*: benzothiophene (pattern p_3: {ccs, c1ccccc1}), (iii) *aromatic oxygen compounds*: benzofurane (pattern p_4: {coc, c1ccccc1}), and (iv) *polycyclic aromatic hydrocarbons*: naphthalene (pattern p_5: {c1ccc2ccccc2cc1}).
  - δ-skypatterns complete the list of the aromatic rings which were not enumerated during the extraction of the skypatterns, namely biphenyl (pattern p_6: {c1ccccc1c2ccccc2}).

In order to partition the (soft-) skypatterns, we used once again the *k-means* clustering method with the euclidean metric. The computed solution consists in k=3 distinct clusters. Figure 7b highlights these clusters:

1. Cluster #1 is made of 3 skypatterns and 6 edge ones, with very high growth rate and aromaticity equal to zero. They correspond to organophosphorus moieties.
2. Cluster #2 contains only δ-skypatterns. From a chemical point of view, it emphasizes several different aromatic rings.
3. Cluster #3 comprises 2 skypatterns and several edges ones which correspond to nitrogen aromatic compounds.

(c) Growth rate, frequency and aromaticity measures (M_4)**.** The most interesting results are provided using M_4 (see Figure 8).

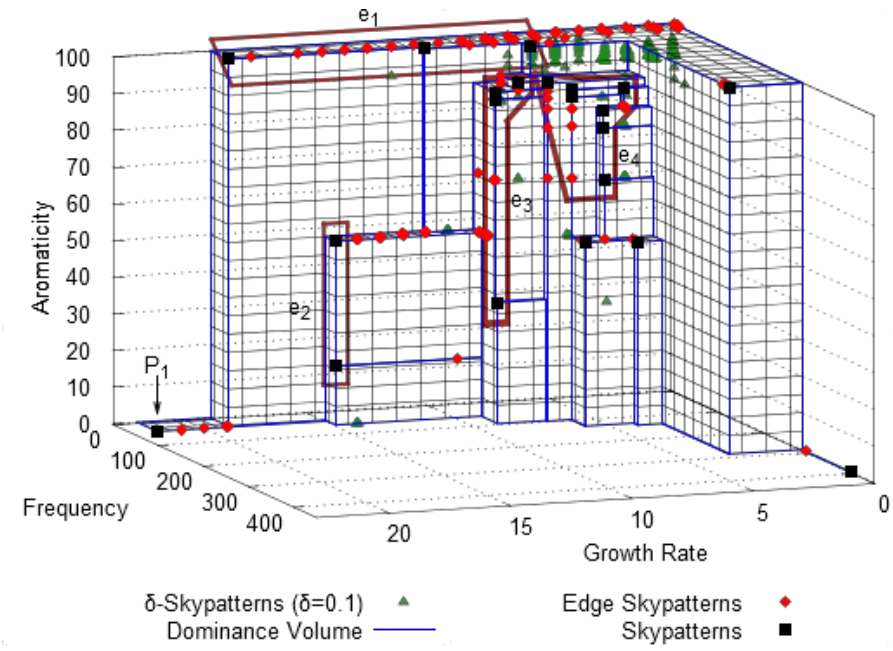


Fig. 8: Analysing the (soft-) skypatterns for M_4.

- **Skypatterns**. 21 skypatterns are mined, and several well-known toxicophores are emphasized: the phenol ring (see e_4), the chloro-substituted aromatic ring (see e_3), the alkyl-substituted benzene (see e_2), and the organophosphorus moiety (see pattern P_1). Besides, information dealing with nitrogen aromatic compounds are also extracted (see e_1).
- **Soft skypatterns** enable to mine several exotic aromatic rings (previously discussed), namely nitrogen and S-containing aromatic compounds, polycyclic aromatic hydrocarbons. Moreover, edge-skypatterns enable to detect more precisely the chloro-substituted aromatic ring and the organophosphorus moiety (e.g. $\{$`COP(=S)O`, `O(P(OC)=S)C`, `O(CC)P=S`$\}$ which are located near p_1). For $\delta \in \{10\%, 20\%\}$, the extraction of the δ-skypatterns leads to the extraction of new several interesting patterns, particularly substituted nitrogen aromatic rings and substituted anilines ($\{$`c1(ccccc1)N`$\}$).

Table 6 gives a classification of all the (soft-) skypatterns extracted according to well-known toxicophores. The introduction of the softness (via soft skypatterns) enables to discover interesting toxicophores previously discussed that would not be detected by skypatterns.

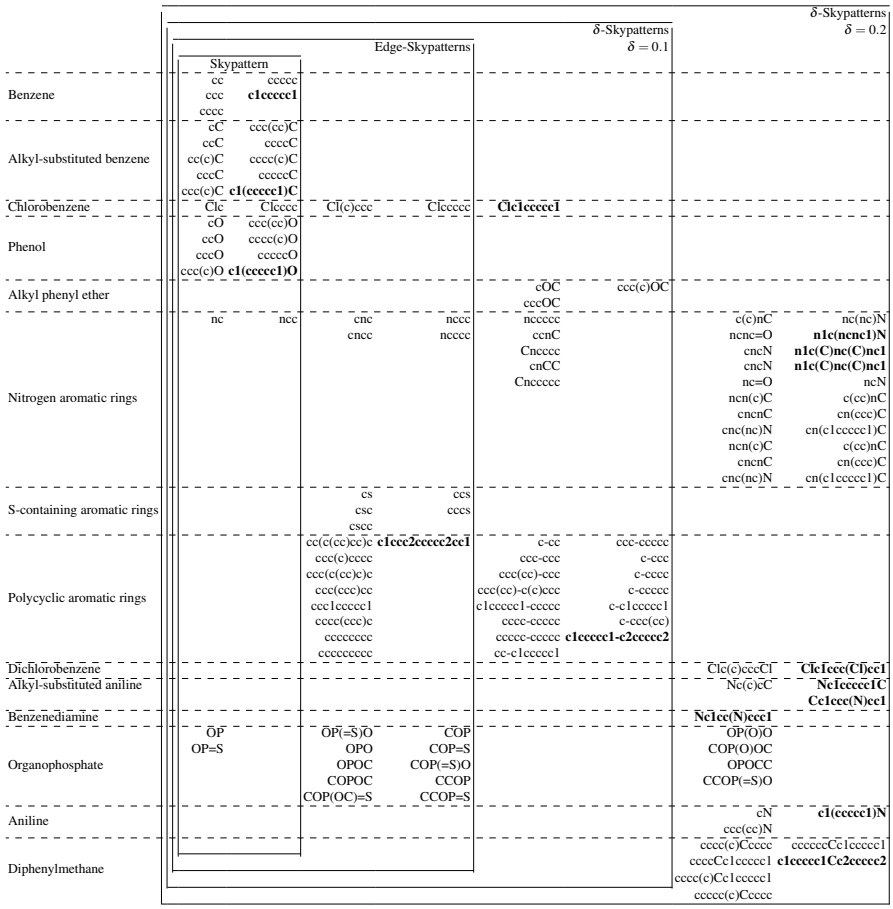


Table 6: Repartition of soft skypatterns for M_4.

7 Conclusion

We have introduced the notion of soft skypattern and proposed a flexible and efficient approach to mine skypatterns as well as soft ones thanks to Dynamic CSP. Moreover, the declarative side of the CP framework easily enables us to manage constraints providing several kinds of softness and leads to a unified framework handling softness in the skypattern problem. Finally, the relevance and the effectiveness of our approach has been highlighted through experiments on UCI benchmarks and a case study in chemoinformatics for discovering toxicophores.

In the future, we would like to study the introduction of softness on other tasks such as clustering, study the contribution of soft skypatterns for recommendation and extend our approach to skycubes. Another direction is to improve the solving

stage by designing a one-step method: each time a new solution s_i is found, all candidates that are dominated by s_i can be removed (see Section 4.3). Another idea is to hybridize our CP approach with local search methods [Drugan and Thierens, 2012] to improve the efficiency of the method.

Acknowledgements This work is partly supported by the ANR (French Research National Agency) funded project FiCOLOFO ANR-10-BLA-0214. The authors would like to thank Arnaud Soulet (University Franois Rabelais of Tours, France), for providing the `Aetheris` program and his highly valuable comments.

References

[Auer and Bajorath, 2006] Auer, J. and Bajorath, J. (2006). Emerging chemical patterns: A new methodology for molecular classification and compound selection. *Journal of Chemical Information and Modeling (JCIM)*, 46(6):2502–2514.

[Bistarelli and Bonchi, 2007] Bistarelli, S. and Bonchi, F. (2007). Soft constraint based pattern mining. *Data & Knowledge Engineering (DKE)*, 62(1):118–137.

[Börzsönyi et al., 2001] Börzsönyi, S., Kossmann, D., and Stocker, K. (2001). The skyline operator. In *Proceedings of the 17th International Conference on Data Engineering (ICDE'2001)*, pages 421–430. IEEE Computer Society.

[De Raedt et al., 2008] De Raedt, L., Guns, T., and Nijssen, S. (2008). Constraint programming for itemset mining. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'2008)*, pages 204–212. ACM.

[De Raedt and Zimmermann, 2007] De Raedt, L. and Zimmermann, A. (2007). Constraint-based pattern set mining. In *Proceedings of the Seventh SIAM International Conference on Data Mining (SDM'2007)*, pages 237–248. SIAM.

[Drugan and Thierens, 2012] Drugan, M. M. and Thierens, D. (2012). Stochastic pareto local search: Pareto neighbourhood exploration and perturbation strategies. *Journal of Heuristics*, 18(5):727–766.

[Gavanelli, 2002] Gavanelli, M. (2002). An algorithm for multi-criteria optimization in CSPs. In *Proceedings of the 15th Eureopean Conference on Artificial Intelligence (ECAI'2002)*, pages 136–140. IOS Press.

[Guns et al., 2011] Guns, T., Nijssen, S., and De Raedt, L. (2011). Itemset mining: A constraint programming perspective. *Artificial Intelligence*, 175(12-13):1951–1983.

[Jin et al., 2004] Jin, W., Han, J., and Ester, M. (2004). Mining thick skylines over large databases. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'2004)*, volume 3202 of *Lecture Notes in Computer Science*, pages 255–266. Springer.

[Khiari et al., 2010] Khiari, M., Boizumault, P., and Crémilleux, B. (2010). Constraint programming for mining n-ary patterns. In *Proceedings of the 16th International Conference in Principles and Practice of Constraint Programming (CP'2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 552–567. Springer.

[Kung et al., 1975] Kung, H. T., Luccio, F., and Preparata, F. P. (1975). On finding the maxima of a set of vectors. *Journal of the ACM*, 22(4):469–476.

[Lin et al., 2007] Lin, X., Yuan, Y., Zhang, Q., and Zhang, Y. (2007). Selecting stars: The k most representative skyline operator. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE'2007)*, pages 86–95. IEEE.

[Mannila and Toivonen, 1997] Mannila, H. and Toivonen, H. (1997). Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258.

[Matousek, 1991] Matousek, J. (1991). Computing dominances in Eˆn. *Information Processing Letters (IPL)*, 38(5):277–278.

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

[Papadias et al., 2005] Papadias, D., Tao, Y., Fu, G., and Seeger, B. (2005). Progressive skyline computation in database systems. *ACM Transactions on Database Systems (TODS)*, 30(1):41–82.

[Papadias et al., 2008] Papadias, D., Yiu, M. L., Mamoulis, N., and Tao, Y. (2008). Nearest neighbor queries in network databases. In *Encyclopedia of GIS*, pages 772–776. Springer.

[Papadopoulos et al., 2008] Papadopoulos, A. N., Lyritsis, A., and Manolopoulos, Y. (2008). Skygraph: an algorithm for important subgraph discovery in relational graphs. *Data Mining and Knowledge Discovery*, 17(1):57–76.

[Poezevara et al., 2011] Poezevara, G., Cuissart, B., and Crémilleux, B. (2011). Extracting and summarizing the frequent emerging graph patterns from a dataset of graphs. *Journal of Intelligent Information Systems (JIIS)*, 37(3):333–353.

[Shelokar et al., 2013] Shelokar, P., Quirin, A., and Cordón, O. (2013). Mosubdue: a pareto dominance-based multiobjective subdue algorithm for frequent subgraph mining. *Knowledge and Information Systems (KAIS)*, 34(1):75–108.

[Soulet et al., 2011] Soulet, A., Raïssi, C., Plantevit, M., and Crémilleux, B. (2011). Mining dominant patterns in the sky. In *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM'2011)*, pages 655–664. IEEE Computer Society.

[Steuer, 1992] Steuer, R. E. (1992). *Multiple Criteria Optimization: Theory, Computation and Application*. Radio e Svyaz, Moscow, 504 pp. (in Russian).

[Tan et al., 2001] Tan, K., Eng, P., and Ooi, B. C. (2001). Efficient progressive skyline computation. In *Proceedings of 27th International Conference on Very Large Data Bases (VLDB'2001)*, pages 301–310. Morgan Kaufmann.

[Ugarte et al., 2012] Ugarte, W., Boizumault, P., Loudni, S., and Crémilleux, B. (2012). Soft threshold constraints for pattern mining. In *Proceedings of the 15th International Conference in Discovery Science (DS'2012)*, volume 7569 of *Lecture Notes in Computer Science*, pages 313–327. Springer.

[van Leeuwen and Ukkonen, 2013] van Leeuwen, M. and Ukkonen, A. (2013). Discovering skylines of subgroup sets. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD'2013)*, volume 8190 of *Lecture Notes in Computer Science*, pages 272–287. Springer.

[Verfaillie and Jussien, 2005] Verfaillie, G. and Jussien, N. (2005). Constraint solving in uncertain and dynamic environments: A survey. *Constraints*, 10(3):253–281.