WaveLSea: helping experts interactively explore pattern mining search spaces

Etienne Lehembre, Bruno Crémilleux, Albrecht Zimmermann, Bertrand Cuissart, Abdelkader Ouali

WaveLSea: helping experts interactively explore pattern mining search spaces

Etienne Lehembre[1*], Bruno Cremilleux[1], Albrecht Zimmermann[1], Bertrand Cuissart[1], Abdelkader Ouali[1]

[1]UNICAEN, ENSICAEN, CNRS - UMR GREYC, Normandie Univ, 14000, Caen, France.

*Corresponding author(s). E-mail(s): lehembre.etienne@gmail.com;
Contributing authors: bruno.cremilleux@unicaen.fr;
albrecht.zimmermann@unicaen.fr; bertrand.cuissart@unicaen.fr;
abdelkader.ouali@unicaen.fr;

Abstract

This article presents the method Wave Top-k Random-d Lineage Search (WaveLSea) which guides an expert through data mining results according to her interest. The method exploits expert feedback, combined with the relation between patterns to spread the expert's interest. It avoids the typical feature definition step commonly used in interactive data mining which limits the flexibility of the discovery process. We empirically demonstrate that WaveLSea returns the most relevant results for the user's subjective interest. Even with imperfect feedback, WaveLSea behavior remains robust as it primarily still delivers most interesting results during experiments on graph-structured data. In order to assess the robustness of the method we design novel oracles called soothsayers giving imperfect feedback. Finally, we complete our quantitative study with a qualitative study using a user interface to evaluate WaveLSea.

Keywords: Interactive Pattern Mining, Structured Pattern Mining, Data Exploration, Reinforcement Learning, and Human Machine Interface

1 Introduction

In the pharmaceutical industry, measuring the affinity of a drug-like molecule with a biological receptor is both a vital operation to identify drug precursors, and an

expensive one, both financially and resource-wise. As such, it is common practice to use computational methods to suggest molecular substructure patterns associated to strong affinity for a targeted receptor by mining data from previous test series. When these results are a set of salient patterns, or a set of patterns generated by a neural network as with molecular generation [1, 2], a recurring problem is their large number, often impossible to analyze for a human expert. Various approaches address this problem, e.g. *condensed representations* of result sets that synthesize the search space [3], numerous *quality measures* [4], and, more recently, *pattern set mining* techniques [5]. However, even the combination of these methods remains insufficient as the search space remains too large for human experts. Therefore, one proposal is to integrate the expert into the process via a search described as *interactive* [6].

While several interactive pattern mining methods deal with data as itemsets [7, 8], few works focus on interactive pattern mining dedicated to structured data, e.g. graphs [9, 10], topical to process molecular data. Moreover, even in those works, subgraphs are treated the same way as itemsets and the structural relationships between them are not fully used. The methods do not relate the size of the subgraphs to a level of specificity in the pattern space. In addition, as the standard approach in interactive mining learns an approximation of user preferences on patterns by translating them into weights on predefined features, it necessarily narrows down the adaptation to the user's preferences. Finally, most of the current methods either use a binary interaction, a pattern being seen either as a positive one or as a negative one [9–11], which may lack nuance to describe the user interest, or require a full user-derived pattern ranking [7], which quickly becomes challenging.

This work structures the search space to efficiently compute the subgraph samples submitted to an expert for feedback in an iterative setting. After each interaction of the expert with the suggestions sample, the sampling is revised according to her subjective interest guiding her exploration. Our study draws out three crucial points. First, the exploration has to avoid focusing on only a part of the research space partially ordered graph (POG), i.e. a subset of the solution patterns. This point is addressed by the introduction of smooth *exploratory waves* to steer the exploration of the POG. Second, the search has to explore a POG while converging towards the interest of the expert by reducing the syntactic distance between the samples and the expert's pattern subset of interest. The algorithm does so by using the partial order relation derived from graph inclusion to structure the search space and to propagate the expert's subjective interest. This avoids the usage of features and their induced bias. Third, in order to properly spread the subjective interest, the method provides a graduated interaction scheme to the user with more than two options.

Our contribution in this paper is three-fold.

- First, we propose an interactive pattern selection method: Wave Top-k Random-d Lineage Search (WaveLSea), which, in short, exploits the structure of the pattern search space. The WaveLSea method avoids the need to use pre-defined features by embedding the user's interest directly in the partial order graph of the search space. Therefore, the expert's interactions directly modify the search space. Its exploration is done through sampling waves steering the expert in a gentle and steady manner through the search space, avoiding abrupt jumps that make it hard to relate viewed

patterns to each other. Notably, we discuss our method in the context of graph mining, both due to the fact that little work exists in this field, and because of our interest in chemoinformatics. But *whenever* one can structure the pattern search space, as is also the case for itemsets, sequences, or trees, our algorithm can be applied to guide the user through the mining results.
- Second, we propose more realistic and more complex protocols for evaluating interactive mining, as opposed to the perfect oracles that are used in existing work [7, 9, 10, 12–16]. Instead of mapping a predefined quality function one-to-one to simulated responses, we propose to model experts that can get confused, or are biased, or unsure of themselves. This novel evaluation method allows us to observe the robustness of WaveLSea when confronted with expert errors.
- Third, we illustrate the WaveLSea method on a concrete use case arising in therapeutic chemistry, using a primary dataset of interest: BCR-ABL1.

The article is organized as follows. In Section 2 we introduce core notions. Section 3 describes the Wave Top-k Random-d Lineage Search method and in Section 4 we present the corresponding related work. Section 5 defines a novel methodology to evaluate interactive pattern mining processes. In Section 6, we present the quantitative experimental evaluation and discuss the results. In Section 7, we present the package user interface and a small evaluation done with it. Finally, Section 8 concludes.

2 Preliminaries

Let $\mathcal{D}$ a *dataset*, $\mathcal{L}$ a *pattern language*[1], and $\mathbb{G}(\mathbb{V}, \mathbb{E})$ a *graph* where $\mathbb{V}$ is a set of *vertices* and $\mathbb{E}$ a set of *directed edges*. A *Partial Order Graph* (POG) represents the *partially ordered pattern space* (poset) of $\mathcal{L}$ under a given *partial order* $<$. For *graph patterns*, the partial order can be derived from *subgraph inclusion*: $g_1 < g_2 : g_1 \subseteq g_2 \Leftrightarrow g_1$ is isomorphic to a subgraph of g_2. For each vertex $v \in \mathbb{V}$, v contains a pattern set $X = \{x_i \in \mathcal{L}\}, |X| \geq 1$ also called *Equivalence Class* (EC). In our case, x_i are frequent subgraphs having the same support (as defined in the following). We therefore overload the subgraph relation $<$ to sets of subgraphs: $X_1 < X_2 \Leftrightarrow \exists x_1 \in X_1, \exists x_2 \in X_2, x_1 < x_2$.

Each pattern set X has an associated set called *support*, denoted $sup(X)$, containing elements of $\mathcal{D}$ in which X occurs: $sup(X) = \{t \in \mathcal{D} \mid \forall x \in X, x < t\}$, then $X_1 < X_2 \implies sup(X_2) \subset sup(X_1)$. The cardinal of the support is called the *frequency*, noted $frequency(X)$.

We chose to go with the classical anti-monotonic support definition for the sake of descriptive clarity. Nevertheless, as described in the following, the edge definition does not use the support. In fact, any measure that allows for the definition of (syntactic) equality classes, e.g. convertible constraints [17] or even witnesses [18], can be used. Therefore, the method described in this article is also applicable when the support and frequency functions are non anti-monotonic.

[1] As mentioned before, even though we discuss WaveLSea in the context of graph mining, other pattern languages are equally admissible.

As each vertex contains a pattern set, we base our definition of directed edges on the $<$ relation. Thus, the directed edge set is defined as:

$$\mathbb{E} = \{(v_1, v_2) \mid v_1, v_2 \in \mathbb{V}, X_{v_1} < X_{v_2}, \nexists v_3 \in \mathbb{V} : X_{v_1} < X_{v_3} < X_{v_2}\}.$$

To relate the construction of the POG to existing work, in formal concept analysis, it is a lattice [19]. In this work, we call v_1 *parent* and v_2 *child*. We extend the relation by transitivity to parents of parents and children of children called respectively *ancestors* and *descendants*. The *lineage* of $v \in \mathbb{V}$ is the set of its ancestors and descendants in $\mathbb{G}$, denoted $\mathcal{L}i(v)$. We define the *roots* of $\mathbb{G}$ as the set of vertices $v \in \mathbb{V}$ having no parent. The *distance* is the minimal count of directed edges between two vertices of the POG. A *layer* L_i is the set of vertices having the same distance i from the roots of the POG, said distance called *depth* of the layer. We note L when the depth is not relevant.

Let L a layer of $\mathbb{G}$. We call the layer composed of the parents of the vertices of L and the layer formed by their children the *adjacent layers*. Thus, we can build a POG in a *layered view* where the first layer contains the smallest subgraphs with the largest supports and the last layer contains the largest subgraphs with the smallest supports. Intuitively, the first layer contains the most generic patterns and the last layer contains the most specific patterns.

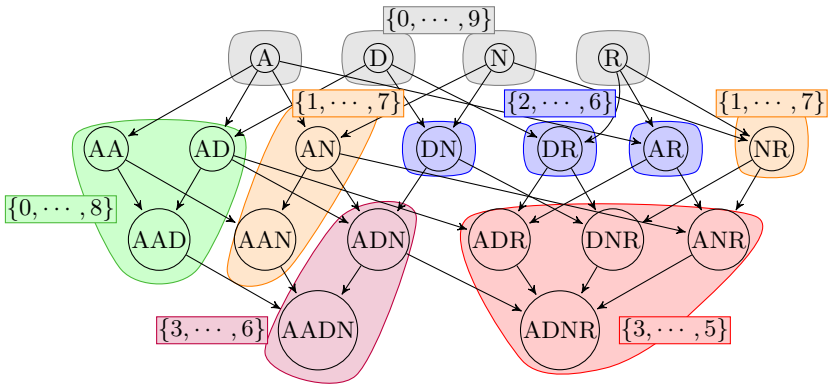


Fig. 1 Building a POG from subgraphs

Example: subgraph pattern mining theory can be used in chemoinformatics through *pharmacophores* [20]. Pharmacophores are complete subgraphs based on pharmacophoric features labeling their vertices. Each pharmacophore appears in a set of molecules, i.e. its support. They are used to study the biological behavior of a molecule fitting a binding site. It allows the expert to establish a link between the biological behavior of the molecule described by the pharmacophore and its molecular structure.

In Figure 1 each vertex is a pharmacophore, molecule identifiers are indicated in colored rectangles and directed edges are represented as arrows. The support of a pharmacophore matches the colored area it is in. Each colored area defines a *Structured Equivalence Class* (SEC) defined as in [21]. There are four pharmacophoric features:

$A, D, N,$ and R. Thus, the subgraph pattern AN is a pharmacophore composed of features A and N. A directed edge links A to AN as $A < AN$. We note that $sup(AN) \subset sup(A)$. Thus, vertices are organized in a layered setting with the most generic patterns at the top and the most specific ones at the bottom. In this article, we drop edge labels from figures for the sake of better readability.

In order to integrate the user's interest into the graph, we have to introduce several notions. The Interest Partial Order Graph (IPOG) is defined as follows. Let $\mathbb{G}(\mathbb{V}, \mathbb{E}, \mathbb{V}^+, \mathbb{V}^-, \mathcal{W})$ an IPOG where $\mathbb{V}^+ \subseteq \mathbb{V}$ is the subset of vertices prioritized for the exploration, $\mathbb{V}^- \subseteq \mathbb{V}$ is the subset of vertices excluded from the exploration, and $\mathcal{W} : \mathbb{V} \to \mathbb{R}$ is the weight function $\mathcal{W}(v)$ associating each vertex $v \in \mathbb{V}$ to a real number $j \in \mathbb{R}$ called *weight*. The next section details our method, which aims to guide an expert in the relational graph of the pattern space by building the IPOG.

3 The Wave Top-k Random-d Lineage Search

This section presents Wave Top-k Random-d Lineage Search (WaveLSea). The method aims to emulate the expert's subjective interest in order to guide her in the exploration of the IPOG. From this perspective, WaveLSea relies on several modules, the first being the *exploratory wave*. The second is the *diffusion of the expert's interest* through the structure, modifying the pattern search space. The last is the *exploitation of the diffused interest* to iteratively sample the result space.

3.1 The exploratory waves

The wave motion guides the expert from the most generic patterns to the most specific ones (the crest of the wave) before rolling back by sampling patterns for the expert in each layer iteratively. The motion leads the expert to contrast her understanding of the studied patterns by comparing them with their ancestors and descendants. It also allows the diffusion of the expert's interest without sampling elements lacking connections to the pattern space the expert just observed. Philosophically, this is similar to the Expectation Maximization (EM), or similar iterated optimization, algorithm(s) that update values then exploit them in the next iteration to inform the process. It also shares the same mindset as iterative redescription mining [22].

In the same way as waves on a beach, the exploratory wave cycles through the layers of the context graph then retraces its steps in the backwash. During this movement, grains of sand will be carried inland before being carried back. These grains are the snippets of knowledge discovered by the expert, which she will carry from the first layers to the last, before confronting this knowledge with their generalizations, like grains of sand trapped in a wave.

Let f be the first layer interesting the expert and l the last one. Let S_i be a sample of pattern drawn from the pattern space. We note $[S_i]$ a sequence composed of samples, and $\cdot$ the concatenation of two sequences. We define an exploratory wave as a sequence of samples composed as follows.

$$[S_i | i \text{ from } f \text{ to } l] \cdot [S_i | i \text{ from } l-1 \text{ to } f+1] \tag{1}$$

As each sample is presented to the expert for labeling before the next one is drawn, the composition of the next sample is affected by previous samples. Each wave also takes into account previous waves for the sake of continuity. We now detail interactions and corresponding labels which will affect the sampling composing the exploration's waves.

Table 1 Expert's interactions and their respective consequences.

Interaction	Consequences	Color
Rejected	Exclusion areas & Weights diminution	Red
Uninterested	Weights diminution	Orange
Unsure	no changes	Purple
Interested	Weights augmentation	Blue
Accepted	Prioritized areas & Weights augmentation	Green

3.2 Expert interactions and propagation of the interest

In order to diffuse the expert's interest and sample informative patterns for her, the method needs means of communication: an *interaction*. The proposed *interactions* and their *consequences* are listed in Table 1.

The different interactions carry different semantics, and, as the table shows, have different effects on the POG:

- "Accepted" patterns are ones the expert intends to keep for further evaluation afterwards, and whose syntactic lineage can be expected to be relevant as well. "Rejected" patterns, on the other hand, are not of any interest to the expert.
- "Interesting" patterns *might* be worthy of further evaluation but at the time of interacting, the expert is not sure yet. In a similar manner, "uninteresting" patterns will probably be rejected but this is not clear yet.
- "Unsure", finally, is the expert's reaction when they really do not know yet what to do with this pattern and need further information later on.

In terms of consequences, these labels transcribe into two major types of modifications: weight modification and area definition.

Area definition

The first and most drastic consequence in Table 1 is the designation of prioritized and excluded areas for the exploration. Patterns sampled for the expert's review are primarily sampled in prioritized area, then in unmarked ones and *never* in excluded areas. Therefore, those consequences have to be restricted to parents and children of the reviewed vertices.

Let v_1 a sampled pattern for the expert:
If the expert *accepts* v_1 then:

$$\mathbb{V}^+ \leftarrow \mathbb{V}^+ \cup \{v_2 \in \mathbb{V} | \exists (v_1, v_2) \in \mathbb{E} \text{ or } \exists (v_2, v_1) \in \mathbb{E}\} \tag{2}$$

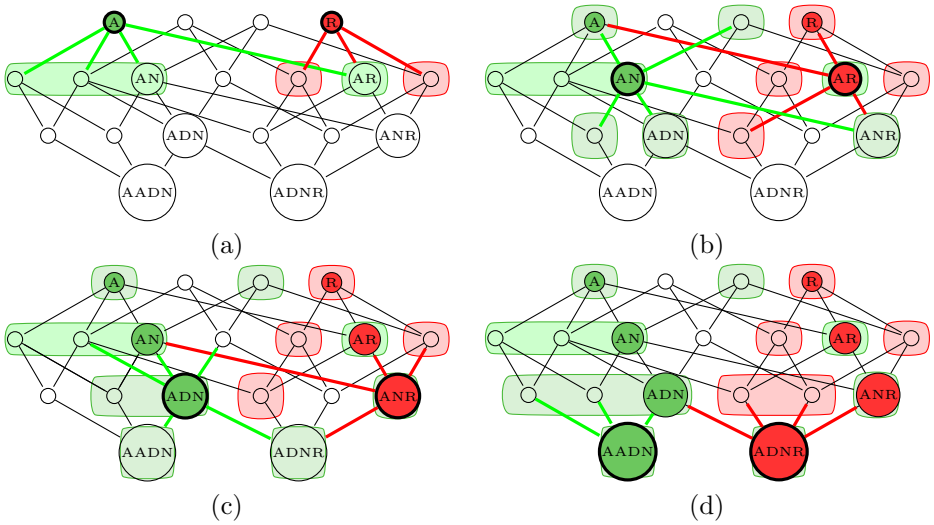


Fig. 2 Defining preferred and excluded research areas in the IPOG.

If the expert *rejects* v_1 then:

$$\mathbb{V}^- \leftarrow \mathbb{V}^- \cup \{v_2 \in \mathbb{V} | v_2 \notin \mathbb{V}^+ \text{ and } \exists (v_1, v_2) \in \mathbb{E} \text{ or } \exists (v_2, v_1) \in \mathbb{E}\} \tag{3}$$

Figure 2 illustrates the area modifications respectively defined in Equations (2) and (3) as consequences of *Acceptance* and *Rejection* by the expert. In this example, the expert searches for pharamacophores similar to $AADN$ and tries to discard pharmacophores similar to $ADNR$ because they induce undesirable behavior. Of course, she has not clearly identified patterns interesting her at the beginning of the exploration, as it would be simpler to use direct search queries in such a case. Subfigure *(a)* places the algorithm in the highest layer of the search space, i.e. the layer containing the most generic subgraphs (pharmacophoric features). A sample of two patterns (A and R) is shown to the expert, who rejects R (in red) and accepts A (in green). As previously described, the WaveLSea method defines an exclusion area in the layer below (in red) as well as a prioritized area (in green). If there is a conflict between the search zones, then the priority zone is favored, so that future ambiguities can be assessed. Subfigure *(b)* places the algorithm in the following layer. The method samples AR (in order to resolve ambiguity) and AN (to better spread the expert's interest) from the prioritized area. The interaction defines areas in the layers above and below narrowing down the search space, but also uncovering new ambiguities. Subfigure *(c)* places the algorithm in the second-to-last layer from which ADN and ANR are sampled. The interactions narrow down the possibilities to the pharmacophores the expert is looking for, while quantifying the expert's interest in them. In the next steps, after confirming the expert's belief, the algorithm will reascend to the middle layer and sample from these areas in order to contrast the found specific knowledge with more generic ones. Subfigure *(d)* displays the final interactions as the expert reaches the end of the process. At this stage, the IPOG is nicely divided into two parts, one corresponding

to the expert's interest (left) and one exhibiting undesired behavior (right). However, there are areas of ambiguity in the right part which have been resolved by the exploration. This example shows that local conflicts, confirmations, and rejections induced by the expert's interest can be exploited to achieve a better global comprehension of the search space.

Weights management

The second algorithm module is more subtle. It modifies the weights of the lineage of vertices reviewed by the expert in order to impact future samples. Let $v \in \mathbb{V}$ a vertex, A an expert's interaction, and λ a modifier.

$$weighting(v, A, \lambda) = \begin{cases} \mathcal{W}(v) + \lambda & \text{if } A \in \{\text{accepted, interested}\} \\ \mathcal{W}(v) - \lambda & \text{if } A \in \{\text{rejected, uninterested}\} \end{cases} \tag{4}$$

However, the greater the distance between two vertices, the more different the contained patterns. Therefore, we have to consider this variation in the application of Equation (4). Moreover, two distinct patterns will probably not have the same interestingness in the eyes of the expert. Hence, the definition of Equation (5), where i is the distance between two vertices v_1 and v_2 and the modifier is the weight of the sampled pattern discounted by the distance, such that:

$$\forall v_2 \in \mathcal{L}i(v_1) \cup \{v_1\},\ weighting(v_2, A, |\mathcal{W}(v_1)| * \frac{1}{2^{i-1}}) \tag{5}$$

We note that the λ used in Equation (4) is in fact the weight of the vertices carrying the novel interaction decreased according to the distance between the vertices. The factor $\frac{1}{2^{i-1}}$ embodies the decreasing correlation between the sampled subgraph and its generalizations and specifications. We chose this discount factor after also evaluating multiplicative and additive weights proposed in the literature [23], which lead however to similar yet less convincing results, something also found in [24]. Moreover, the modification of the vertices' weights also modifies their impact when the expert interacts with them. The higher the absolute value of a weight is, the higher the impact that is obtained from the interaction. Therefore, the larger the vertex's lineage is, the stronger the interaction interest diffusion will be.

In Figure 3, we choose as an example an IPOG where the default weight of a vertex is 1.00. Therefore, each vertex with no link to any of the interactions of the expert has a weight of 1.00. The expert gives one positive interaction to AN (in blue) and one negative interaction to AR (in orange). According to Equation 5, the weights of AN's lineage are increased according to AN's weight. Respectively, the weights of AR's lineage are decreased according to AR's weight. We note that ANR does not propagate the interactions, as they cancel each other out at this point.

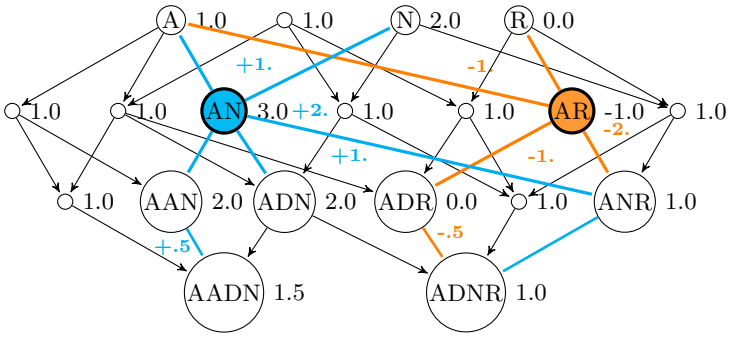


Fig. 3 Weight diffusion under the expert's interest

3.3 Pattern sampling

Now that we possess an efficient way to diffuse the expert's interest into the IPOG, we have to exploit this information in order to sample patterns from our search space. To achieve this, the method uses the IPOG's weights to compute the potential interest of each pattern. A pattern's probability to be sampled increases:

- with the cumulative weight of its *accepted* and *interested* lineage, inducing a higher chance to be interesting or accepted (*exploitation heuristic*).
- with the cumulative weight of its *unexplored* lineage, guiding the expert towards exploring the unexplored space (*exploration heuristic*).
- if the cumulative weights of its positive and negative lineage are *similar*, which indicates a contradiction which requires further exploration (*ambiguity heuristic*).

Following this line of argument, we compute the potential interest of v as follows:

$$I_p(v) = \sum_{i=0}^{k} (\mathcal{W}(\mathcal{L}i_i^+(v)) + \mathcal{W}(\mathcal{L}i_i^?(v)) + \mathcal{W}(\mathcal{L}i_i^U(v)) + 2 * min(\mathcal{W}(\mathcal{L}i_i^+(v)), \mathcal{W}(\mathcal{L}i_i^-(v, \mathbb{G})))) * \frac{1}{2^i} \quad (6)$$

In Equation (6), $\mathcal{L}i_i(v)$ is the lineage of v at depth i in $\mathbb{G}$ with:

- $\mathcal{W}(\mathcal{L}i_i^+(v))$ the sum of weights of the vertices in the lineage with a *positive* interaction,
- $\mathcal{W}(\mathcal{L}i_i^?(v))$ the sum of weights of the vertices in the lineage *without* interaction,
- $\mathcal{W}(\mathcal{L}i_i^U(v))$ the sum of weights of the vertices in the lineage with an *unsure* interaction,
- $\mathcal{W}(\mathcal{L}i_i^-(v))$ the sum of weights of the vertices in the lineage with a *negative* interaction,
- the factor $\frac{1}{2^i}$ reducing the influence of vertices depending on their distance.

Note that $2 * min(\mathcal{W}(L_i^+(v,)), \mathcal{W}(L_i^-(v)))$ corresponds to the logical AND operator, which means that the expression rewards similar values of $\mathcal{W}(L_i^+(v))$ with $\mathcal{W}(L_i^-(v))$, highlighting the lineage's ambiguities. To get another view, we can rewrite

$2 * min(L^+, L^-)$ as $L^+ + L^- - |L^+ - L^-|$. If, $L^+ > L^-$ then $L^+ + L^- - |L^+ - L^-| = L^+ + L^- - (L^+ - L^-) = L^+ - L^+ + L^- + L^- = 2 * L^-$. If $L^+ < L^-$ then $L^+ + L^- - |L^+ - L^-| = L^+ + L^- - (L^- - L^+) = L^+ + L^+ + L^- - L^- = 2 * L^+$. Therefore, if L^- is much larger than L^+ — related patterns have largely been rejected or considered uninteresting — the function does not take it into account. If L^+ is much larger than L^- — the pattern under consideration therefore being likely to be acceptable — the function only counts the (low) L^- which is likely to reduce the interest of the pattern. If L^+ and L^- are similar, then the pattern is ambiguous, and we have to resolve the ambiguity.

The potential interest is exploited in two types of sampling. Firstly, to encourage exploitation, the sampling step selects the top k patterns to be shown to the expert, according to I_p. Secondly, to encourage exploration, the sampling pseudo-randomly draws d additional patterns to be shown to the expert with a selection probability computed from I_p. Therefore, at the beginning of each sampling step, the potential interest of every vertex in the sampled layer not in an excluded area is computed. Then, a biased die is rolled d times without replacement for the random sampling. Let $L \in \mathbb{G}$ a graph layer, sampling probabilities are defined as follows:

$$\forall v \in L, P(v) = \frac{I_p(v)}{\sum\limits_{\forall v' \in L} I_p(v')} \tag{7}$$

The first samples drawn from the IPOG favor vertices with numerous parents, children, and close relatives. As no interaction happened yet, the only non-zero contribution to I_p is $\mathcal{W}(L_i^?(v, \mathbb{G}))$. Furthermore, as each vertex is initialized with the same weight, vertices with strong connectivity have the highest potential interest. This behavior is intended as it favors vertices with a wider, and thus stronger, impact on the IPOG at the beginning of the expert's exploration. It thus leads faster to a good discrimination of the vertices in the IPOG.

3.4 The Wave Top-k Random-d Lineage Search algorithm

Algorithm 1 takes as input an IPOG $\mathbb{G}$, an exploitation factor k setting the number of top-scoring samples to be presented, an exploration factor d setting the number of random samples, and the first and last layers of the exploratory waves. As long as the user does not end the process and there are patterns to explore, the loop, lines 1 to 21, continues. In the loop, lines 2 to 19, the value i starts at depth f and ends at depth l, i is incremented at each iteration if $f < l$ or decremented if $f > l$, passing the waves forwards and backwards through the POG. At line 3, we set L to the layer of $\mathbb{G}$ of depth i. We assign to each vertex v in L its potential interest through the loop, lines 4 to 6, by using Equation (6). At line 7, we assign to S a sample of L by drawing k top patterns and d random patterns based on their potential interest at line 8. The loop, lines 9 to 18, modifies the graph $\mathbb{G}$ for each vertex v and its interaction A obtained at line 10. The vertex lineage's weights are modified at line 11 using Equation (5) and the prioritized areas $\mathbb{V}^+$ and excluded areas $\mathbb{V}^-$ are respectively updated at line 13 and 16 by using Equations (2) and (3). After a full descent or ascent, we swap f and l at line 20 in order to explore layers in the opposite direction. This allows the direction

Algorithm 1 Wave Top-k Random-d Lineage Search

Require: $\mathbb{G}(\mathbb{V}, \mathbb{E}, \mathbb{V}^+, \mathbb{V}^-, \mathcal{W})$ a graph, k the number of top draws, d the number of random draws, f the first layer of the exploratory wave, l the last layer of the exploratory wave.

Ensure: $\mathbb{G}$ the graph shaped by the expert's interactions.

1: **while** $\exists v \in \mathbb{V}$ & $v \notin \mathbb{V}^-$ & v not explored **do**
2: **for** $i : f$ to l **do**
3: $L \leftarrow Layer_i(\mathbb{G})$
4: **for** $v \in L$ **do**
5: $Update_{I_p}(v, I_p(v))$ (Equation (6))
6: **end for**
7: $S = \{\text{Top } k \ v' \text{ with the highest } I_p(v')\}$
8: $S = S \cup \{d \ v'' \text{random patterns following Equation (7)}\}$
9: **for** $v \in S$ **do**
10: $A \leftarrow Interaction(v)$
11: $weighting\_lineage(\mathbb{G}, v, A)$ (Equation (5))
12: **if** $A =$ accepted **then**
13: $\mathbb{V}^+ \leftarrow \mathbb{V}^+ \cup \{\forall v_2 \in \mathbb{V} | \exists (v, v_2) \in \mathbb{E} \text{ or } \exists (v_2, v) \in \mathbb{E}\}$
14: **end if**
15: **if** $A =$ rejected **then**
16: $\mathbb{V}^- \leftarrow \mathbb{V}^- \cup \{\forall v_2 \in \mathbb{V} | v_2 \notin \mathbb{V}^+ \text{ and } \exists (v, v_2) \in \mathbb{E} \text{ or } \exists (v_2, v) \in \mathbb{E}\}$
17: **end if**
18: **end for**
19: **end for**
20: $t \leftarrow f + 1$; $f \leftarrow l - 1$; $l \leftarrow t$ (*Reversing the tide' direction*)
21: **end while**

of the search from the most general patterns to the most specific ones before going backwards, giving the exploration its wave-shape. Once the expert is satisfied or there are no patterns left to explore, we return the updated graph at line 21.

3.5 Algorithmic complexity

Let $\mathbb{G}(\mathbb{V}, \mathbb{E})$ a graph, L a layer of *unexplored* vertices of $\mathbb{G}$ and $m = max_{v \in \mathbb{V}}(\#(lineage(v)))$ the maximum number of vertices in a lineage of $\mathbb{G}$.

Interaction complexity

When the expert interacts with $\mathbb{G}$, the interest value is propagated over all the weights of the ancestors and descendants of the vertex carrying the interaction. Thus, the complexity of the operation is $O(m)$.

Sampling complexity

Before each sampling, the unexplored vertices in the layer in which the sample will be drawn must be updated. The complexity of this operation is $O(m * \#(L))$. Let S be

a sample. When the expert evaluates a sample, she interacts with each vertex in the sample. The complexity of the operation is therefore $O(\#(S) * m)$.

Algorithm complexity

Let w be the number of samples evaluated during exploration and $\#(S)$ their maximum cardinality. The total complexity of sampling and evaluation performed during the exploration is $O(w * \#(S) * \#(L) * m^2)$.

As $\#(S)$ and the number of layers are usually small in (interactive) subgraph pattern mining, especially with pharmacophores, the only complexity breaking point is m. Thus, WaveLSea will be slower in dense structures. Nevertheless, in our experiments, making a hundred queries containing three patterns each in a POG of roughly fifteen thousands vertices and sixty-nine thousands directed edges (BCR-ABL1 data) takes only a few seconds on a personal laptop. Meaning that reporting the consequences of one action is fast enough to not be noticed by the user. This last statement has been verified during the human trials of WaveLSea.

Regarding the space-complexity of the method, the algorithm in our implementation needs the full IPOG to propagate the consequences of interactions. But all the information needed are in the IPOG which means that the used space is predictable on the size of the pattern language.

In conclusion, WaveLSea produces two results. First, the pattern set sampled and labeled by the expert through the exploratory waves, leading the expert to a first grasp of the pattern space. Second, the relational graph shaped by the expert's interest: the IPOG. This graph, through the vertices labels, the labeled areas, the weights, and the potential interest, is a structured representation of the expert's interest. A representation which can be observed and studied, offering a *global picture* of the exploration process in the pattern language based on *local* interactions. Through the IPOG, the expert can explore her own interest, study the relation between the patterns she chose to highlight and those that have been discarded. Furthermore, the expert can leverage the IPOG in her evaluation of unobserved elements.

4 Related work

Feature construction

The usual approach to interactive pattern mining describes each result pattern in terms of features such as the presence of language elements, presence in data transactions, frequency, quality or surprisingness calculated from patterns etc. [7, 9, 12, 13]. Those features are either hard-coded or require user definition beforehand. Regardless which of those methods is used, the underlying assumption is that the expert's interest can be reliably mapped to a fixed feature set, and *only* to that set. One could of course add additional descriptors, but there is no guarantee to capture all semantics and computational effort would increase. An exception is [10], where the authors propose to learn embedding-like representations of patterns for sequential or graph patterns. That method allows for better, and less explicit fitting of the expert's interest to the feature vector corresponding to the latent space. Nevertheless, it is still a features vector and therefore, less expressive than the search space itself. Moreover, due to

the expensiveness of that operation, the representation is only learned once at the beginning. In contrast to this, our method does not re-encode patterns but exploits the structure of the search space to diffuse user feedback directly to result patterns, instead of passing via an intermediate feature vector, allowing local and dynamic evaluation to impact the global sampling in the search space. With our method, the expert *directly* impacts the search space's structure and molds it through her interactions.

Search space exploration

Interactive mining can explore the search space in two ways: post-processing a (partial) result set, or exploiting user feedback during mining. The majority of existing work [7, 10, 12, 13] does the former, essentially re-ranking or filtering patterns, simply because enforcing learned user preferences remains rather challenging. While we intend to push user feedback into the mining process itself in future work, our approach currently also performs post-processing. Works such as [9] use user feedback to perform direct randomized search space exploration, e.g. Monte-Carlo Markov Chain (MCMC) [25, 26] or Monte-Carlo Tree Search (MCTS) sampling [27], which runs the risk, however, of only exploring a subpart of the full pattern space. The work in [28] tackles the issues emerging from MCMC and MCTS exploration by using user feedback to directly remove candidates from a *search beam*, which as a heuristic method again risks missing part of the search space because it considers it in its entirety by not having a wave motion. In interactive redescription mining [22], the beam is also used to sample rules to be refined during the process, but as is usual in beam search, only already sampled rules will be further refined in later iterations. Our method, by using exploratory waves, avoids getting stuck in a subpart of the search space like the MCMC method, or proposing two consecutive patterns that are too different from one another, as can be the case of the MCTS method. Moreover, the wave sampling allows to put a focus on undiscovered patterns linked to previously labeled ones without getting stuck on already annotated ones as in the case of beam search. Nevertheless, the information is not lost as the expert's interaction directly impacts the search space structure permanently.

(Subjective) user interestingness

By involving user feedback, interactive mining arguably sounds similar to work on *subjective interestingness*. Works from that field either assume that the user has *fixed*, prior knowledge about the data [29], or that her knowledge gets updated by each mined pattern [30]. Notwithstanding the name, however, such updates typically happen in an *objective* manner, only exploiting prior patterns, and not in a manner involving the user, similar to work on non-redundant pattern mining [31, 32]. The updates are also *global* in nature. Yet the expert is sensitive to local phenomena and her interest may diverge when studying two distinct regions of the search space. Our method takes this into account by only using relevant information through the concept of pattern lineage, syntactically related patterns. In interactive mining, the user is instead supposed to give binary (like/dislike) [9, 28] or ternary feedback [7], or rank patterns [10, 12, 13, 33]. The relationship between this feedback and the above-mentioned features is then learned via some (often linear) function. In our work, the user selects

from a number of discrete actions and her expressed interest is then instantly diffused to neighboring patterns, instead of learning a preference function. Our work also finds similarities with web page recommendation. A work in spam web page labeling [24] only exploits one of two possible user feedback options, spreads it only to a web page's descendants, and tends to use it without further refinement. In our work, the embedded information is evaluated by a heuristic, to assess the potential interest of patterns regarding their lineage in the search space.

5 Designing oracles for the evaluation

One of the recurring problems with interactive methods lies in their evaluation [34], that is, in setting up experiments to serve as empirical proof of the method's effectiveness. In general, it is difficult to gather enough expert users to achieve a significant number of repetitions of the experiment [35]. The insufficient number of experiments, if possible at all, means that outliers may not be identified as such. This difficulty may stem from a lack of experts in the field under study, an inability of the developers of an interactive method to gather them for an experimental evaluation, or simply the fact that such experts do not have the time to carry out the multiple assessments required.

The limited access to human experts has prompted the authors of interactive methods to develop objects that can be used to evaluate their methods automatically: *oracles*. Nevertheless, these oracles are often simple in their operation. They fall into two main categories. 1) Oracles based on an objective quality measure calculated on the pattern or pattern support [7, 9, 10, 12–16, 33, 36]. This measurement is deterministically translated by the oracle into a "*yes*" or "*no*" response, or into a ranking of patterns proposed for evaluation. 2) Oracles based on a selection of a subset of patterns made by one of the authors of the method [11, 37, 38]. In that case, if the pattern is part of the selected subset then the oracle's response is positive, otherwise the response is negative. In the remainder of this document, this deterministic response will be considered as a *hidden label* related to the evaluated pattern. Some work on the borderline between the two methods calculates the expert's prior on a subset of the dataset [15].

However, the use of such an oracle could lead to an overly optimistic evaluation, especially for an exploratory method like ours. In practice, experts rarely return perfect feedback. They are often biased by previous exploration on other datasets. Moreover, they may not be sure of themselves and thus use lighter interactions. In order to overcome this shortcoming, we test our method with *five types of oracles* simulating different possible expert behaviors. To the best of our knowledge, the only work introducing noise in the evaluation is [39] where the authors added a confusion matrix to blur the line between the positive and the negative class for the label determination. Nevertheless, this is the first time that an interactive mining method has been evaluated in such an extensive way. We call these oracles *soothsayers* to indicate their fallibility.

In order to properly evaluate our method with the help of these soothsayers, we assign to each vertex a *hidden label* determined by the quality of the contained pattern

set. The soothsayer assigns a *discovered label* to the reviewed vertex, determined by the combination of the soothsayer's type and the pattern's quality.

These soothsayers are modeled around two main factors. The first one is the evolution of the probability of making an accurate prediction. The second one is the generation of the error if the soothsayer commits a mistake. For the latter, it is crucial to not generate a mistake randomly, as an expert rarely makes a mistake without an understandable reason.

5.1 Defining soothsayer accuracy

In order to compute the accuracy of the soothsayer, we use two values: the *total knowledge score* and the *gained knowledge score*. The total knowledge represents the available knowledge in the pattern language $\mathcal{L}$, denoted K_t. The gained knowledge is the knowledge cumulatively acquired by the soothsayer reviewing patterns, denoted K_g. If we suppose that a soothsayer begins with a fifty percent chance to make a mistake, we can infer the following formula to characterize the accuracy of the soothsayer, denoted D_{Acc}.

$$D_{Acc} = 0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t}) \tag{8}$$

However, in order to compute the soothsayer's accuracy we have to determine K_t and K_g. To do so, we attribute to each pattern X its *knowledge* $K(X)$. In this work, we consider that the more the entities containing a pattern are diverse, the more the pattern is informative. Thus, we consider the pattern set of all entities containing a given pattern to infer its related knowledge. A well-known option for comparing sets is the Jaccard distance but the main problem with using it is that it is far too expensive to compute. To mitigate this, we use a measure that represents a similar idea to get a quick idea of the diversity of the items in a given pattern X.

Let $d \in \mathcal{D}$ a graph of the data. Then, we define $sup^T(d)$ as the *set of patterns occurring in* d. We consider a *random variable* on X denoted $\mathcal{V}_X$ with patterns contained in $sup^T(sup(X))$ as possible values and a distribution computed on the frequency of a given pattern in $sup^T(sup(X))$. Therefore, the more homogeneous the distribution over the patterns is, the greater the *entropy* associated with X, i.e. if the graphs containing X are diverse in their structure, then the knowledge associated with X is substantial. From this perspective, we can infer that the entropy of $\mathcal{V}_X$, denoted $\mathcal{H}(X)$, is the knowledge $K(X)$. Let $frequency_{sup(X)}(X')$ the frequency of the pattern X' in the graphs containing the pattern X, then we can define the knowledge of X as:

$$K(X) = - \sum_{X' \in sup^T(sup(X))} P_{X'} log(P_{X'}) \tag{9}$$

Where the probability of choosing a pattern X' is defined by its frequency in the set of patterns occurring in the graphs containing X as in the following equation :

$$P_{X'} = \frac{frequency_{sup(X)}(X')}{\sum_{X'' \in sup^T(sup(X))} frequency_{sup(X)}(X'')} \tag{10}$$

Using Equation 9, we possess a faster way of determining the knowledge contained in a subgraph pattern. Furthermore, this equation still offers insight into the diversity of the graph set containing a pattern without comparing every pattern contained in it. To get a better idea of the complexity difference, let $n = \#(\mathcal{L})$, $m = mean(sup(X))$ for $X \in \mathcal{L}$, and $p = mean(sup^T(d))$ with $d \in \mathcal{D}$. The complexity of Equation 9 is $O(n * m * p)$.

In the following, we use Equation 9 to compute the accuracy of the soothsayers in the following.

However, the obtained accuracy with Equation 8 is adapted to the case of a binary decision, if the answer is "yes" or "no". In our method, we use five distinct labels: "*Accepted*", "*Interested*", "*Unsure*", "*Uninterested*", and "*Rejected*". That is why we are going to define a distribution matrix (Table 2) to set the soothsayers' decision for a given hidden label.

Table 2 Labels matrix distribution for hidden labels. Hidden labels are indicated in the first cell of each row, with the conditional probability for discovered labels in the rest of the row.

Hidden \ Discovered	Rejected	Uninterested	Unsure	Interested	Accepted
Rejected	50%	20%	15%	10%	5%
Uninterested	17.5%	50%	17.5%	10%	5%
Unsure	5%	20%	50%	20%	5%
Interested	5%	10%	17.5%	50%	17.5%
Accepted	5%	10%	15%	20%	50%

If we combine the probability matrix in Table 2 with Formula 8, we obtain an evolutionary distribution matrix providing the discovered label distribution vector for each hidden label at a given time of the exploration. Let X a given pattern and M the labels' distribution matrix. Therefore, the probability for the soothsayer of choosing a label for a pattern knowing its hidden label is defined as follows. Let $\Omega = \{Rejected, Uninterested, Unsure, Interested, Accepted\}$ the set of labels and $l, l' \in \Omega$ two labels. Then $M_{l,l'}$ is the probability that the soothsayer returns the label l' if the hidden label of X is l. The probability that the soothsayer attributes the label l' to X is defined as:

$$P_{l'}(X) = \begin{cases} M_{l,l'} + \frac{K_g}{K_t} - (\frac{K_g}{K_t} * M_{l,l'}) & \text{if } l = l' \\ M_{l,l'} - \frac{K_g}{K_t} M_{l,l'} & \text{otherwise} \end{cases} \tag{11}$$

In Equation 11, the probability of choosing the hidden label increases proportionally with the soothsayer's gained knowledge while the probability of choosing the other labels decreases. Meaning that, as the soothsayer learns more about the dataset by reviewing its patterns, it becomes more accurate and makes fewer mistakes.

However, with this method, the soothsayer's errors are only linked to the hidden label of X which omits a part of the pattern information. Thus, we want to use the concrete properties derived from a pattern such as its support or its composition to define the label corresponding to the soothsayer's mistake. By combining the error determination with the accuracy defined in Equation 8, we will design more human-like soothsayers.

5.2 Defining the soothsayers' discovered labels

As we already defined the soothsayers' increasing accuracy, we now define several methods to determine a discovered label when a soothsayer makes a mistake. Including the soothsayer making no mistake at all (the standard solution in the state of the art) and the soothsayer using the distribution matrix to choose his labels, we list six distinct soothsayers.

The omniscient one: makes no errors, is an oracle corresponding to the existing state of the art, assigns to each presented vertex its hidden label.

The probabilistic one: has for each label a dynamic response probability vector inducing a fixed error percentage. In order to avoid improbable answers, each vector contains choice probabilities for each label. The idea is to give the highest probability to the true (hidden) label and positive probabilities to similar labels. The more impact a choice has, the less likely that the experts is wrong, for we consider these choices are made when the expert feels sure of herself. Distributions are indicated in Table 2 where each row corresponds to a probability vector in which each column contains a label's probability of being chosen, given the label in the left-most cell. The discovered label is assigned following Equation 11.

The biased one: models the expert's *prior* coming from her knowledge concerning previously studied datasets. The prior is determined by a second quality measure whose behavior diverges from the one which determines the hidden labels, yet errors made by the oracle are calculated from the same support values. Therefore, errors made by the oracle remain consistent with the underlying information. The measure chosen to represent the bias in this work is the *confidence*. Let l be the hidden labels and l' be the label determined by a second quality measure. Then the probability of choosing l is defined as follows:

$$P_l(X) = \begin{cases} 1 & \text{if } l = l' \\ 0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t}) & \text{elsewise} \end{cases}$$

Analogously, the probability of choosing l' is defined as:

$$P_{l'}(X) = \begin{cases} 1 & \text{if } l = l' \\ 1 - (0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t})) & \text{eslewise} \end{cases}$$

The locally subjective one: models the expert's behavior if she mainly focuses on the samples, leading her to label the sample by considering its top pattern as *at least* interesting and the lowest-scoring as *at least* uninteresting when she makes a mistake. Let $S = (X_1, \cdots, X_i)$ a sample of i patterns sorted in increasing order by the quality measure used to determine the hidden labels. If the hidden label of X_i is *Accepted* or *Interested*, then the soothsayer makes no mistake. Otherwise, the soothsayer can make a mistake and labels the pattern as *Interested*. If the hidden label of X_1 is *Rejected* or *Uninterested*, then the soothsayer makes no mistake. Otherwise, the soothsayer can make a mistake and labels the pattern as *Uninterested*. Therefore, the probability of choosing the right label is defined as follows. Let l be the hidden label of X_i. The probability of labeling X_i with l is:

$$P_l(X_i) = \begin{cases} 1 & \text{if } l = \textit{Accepted} \text{ or } l = \textit{Interested} \\ 0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t}) & \text{elsewise} \end{cases}$$

Let l' be the hidden label of X_1. The probability of labeling X_1 with l is:

$$P_{l'}(X_1) = \begin{cases} 1 & \text{if } l' = \textit{Rejected} \text{ or } l' = \textit{Uninterested} \\ 0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t}) & \text{elsewise} \end{cases}$$

The subjectively surprised one: models an expert exploring patterns that surprise her, whether due to high quality or not. In order to coherently compute this surprisingness, we use the *Outstanding Pattern Selector* introduced in [41]; patterns selected by it are labeled as *Accepted* by the soothsayer when it makes a mistake. Let X a sampled pattern and l its hidden label. Then the probability of discovering l is defined as follows.

$$P_l(X) = \begin{cases} 1 & \text{if } l = \textit{Accepted} \\ 0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t}) & \text{elsewise} \end{cases}$$

The neighborhood biased one: models an expert biased by the quality of a *closely related pattern* having a bigger impact than the reviewed pattern. When it makes a mistake, the soothsayer considers every pattern being a parent, child or sibling of the reviewed patter. It then selects the hidden label of the pattern having the biggest impact, i.e. having the greatest absolute value for the quality measure. Let l the hidden label of the sampled pattern X and l' the hidden label of its neighbor with the highest potential interest. Then, the probability of labeling X with l is:

$$P_l(X) = \begin{cases} 1 & \text{if } l = l' \\ 0.5 + \frac{K_g}{K_t} - (0.5 * \frac{K_g}{K_t}) & \text{elsewise} \end{cases}$$

Now that our six soothsayers are defined, the errors made by the soothsayers are consistent with the patterns sampled and the probability to make mistakes is consistent with the knowledge that the soothsayers possess about the pattern language.

Therefore, we have the opportunity to test if our method is *robust* to the expert's errors in an automated process.

In the following, hidden labels are defined regarding the quality scores of the patterns such that the lowest values are labeled as *Rejected*, the next-lowest are labeled as *Uninterested*, and so on. The quality thresholds are computed for each search space in order to respect as much as possible the following distribution: 2.00% of *Rejected* labels, 18.00% of *Uninterested* labels, 60.00% of *Unsure* labels, 18.00% of *Interested* labels, and 2.00% of *Accepted* labels. This distribution is intended to represent the fact that an expert is not interested in the whole set of results, she is more likely to use actions with a weak impact on the patterns' space and less likely to use actions strongly affecting it.

6 Experiments and results

6.1 Results on publicly available data

Dataset descriptions

Table 3 TUDatasets datasets, their size, the number of extracted subgraphs with a frequency of 10% and the structured equivalence classes composing the POG. Dataset are sorted according to their decreasing number of structured equivalence classes (SEC).

Dataset	Graphs	Subgraphs	SEC	Dataset	Graphs	Subgraphs	SEC
UACC257H	39,988	30,814	30,680	SW-620	40,532	1,005	1,005
YeastH	79,601	24,005	23,970	OVCAR-8	40,516	1,003	1,003
UACC257	39,988	11,075	11,028	SF-295	40,271	1,003	1,003
Yeast	79,601	7,708	7,692	NCI-H23	40,353	1,001	1,001
SF-295H	40,271	3,764	3,764	SN12C	40,004	998	998
SW-620H	40,532	3,757	3,757	P388	41,472	624	624
SN12CH	40,004	3,747	3,747	AIDS	2,000	192	192
BZR_MD	306	3,249	2,147	MUTAG	188	603	110
Mutagenicity	4,337	1,904	1,880	PTC-FM	349	146	96
MCF-7	27,770	1,024	1,024	PTC-MM	336	148	84
PC-3	27,509	1,023	1,023	PTC-FR	351	138	84
MOLT-4	39,765	1,006	1,006	PTC-MR	344	124	83

We selected twenty-four datasets with varying characteristics from the TUDataset repository[2]. Each dataset contains two classes, to render the use of a contrast measure as oracle possible. The frequent subgraphs are extracted by *quickSpan*[3] with a minimal frequency of 10%. We keep the subgraphs' order under seven vertices following recommendations by the chemoinformatician experts with whom we collaborate, a recommendation to which we return in our case study based on BCR-ABL1. Table 3 lists the datasets' names, their size, the number of extracted subgraphs and the number of *structured equivalence classes* (SEC) [21]. Structured equivalence classes are pattern sets computed as follows: if two subgraphs p and q have the same support

[2]https://chrsmrrs.github.io/datasets/docs/datasets/
[3]https://gitlab.inria.fr/Quickspan/quickspan

$sup(p) = sup(q)$ and they are linked in the POG by a path passing only through vertices containing patterns p_i having the same support $sup(p) = sup(p_i)$, i.e. being in the same equivalence class, then they belong to the same structured equivalence class. In Figure 1, structured equivalence classes are defined by the color of the area and the continuity of its border. In the following, we equate structured equivalence classes and pattern sets so each vertex of the POG contains an SEC (i.e. pattern set).

Search space sizes range from about 200 patterns to a few tens of thousand. This variation helps to observe the variable or non-variable behaviors of WaveLSea with respect to its application space and to get an idea about its adaptability. It also gives the opportunity to observe the behavior of the soothsayers when they acquire the total knowledge available in the dataset and when the evolution of the gained knowledge is slow.

In our work, we use the well-known Weighted Relative Accuracy (WRAcc) [42] quality measure based on the graph data classes defined as:

$$WRAcc(X, \mathcal{D}) = \frac{sup(X)}{|\mathcal{D}|} * (\frac{sup(X)^+}{sup(X)} - \frac{|\mathcal{D}^+|}{|\mathcal{D}|}),$$

where $\mathcal{D}^+$ is a subset of $\mathcal{D}$ which contains data graphs from a target class and $sup(X)^+$ the support of X in this subset. As patterns in the same equivalence class have the same support, they will also have the same score, the patterns presented to an expert would be the list of the most *general*, or *generator*, patterns (also known as free sets in itemset mining).

Experimental protocol

In order to evaluate the effectiveness of WaveLSea, each equivalence class in the IPOG is labeled with one of five interactions: *Rejected*, *Uninterested*, *Unsure*, *Interested*, *Accepted*. We query each of the earlier described soothsayers with 100 samples of 3 patterns. We divided the 3 sampled patterns into 2 exploited patterns ($k = 2$) and 1 explored pattern ($d = 1$). As our algorithm (and comparison techniques) contain randomized operations, each (dataset, soothsayer) pair is evaluated 100 times and the observed results averaged.

In order to interpret our results, we study the oracles' discovered labels and the hidden labels. Let A a label type, we define the *Recall* as :

$$Recall(A) = \frac{\text{Discovered}(A)}{\text{Hidden}(A)}$$

The only other existing works on interactive graph mining is [9, 10] but after a number of attempts, we have not been able to acquire an implementation of either of those methods, which unfortunately precludes us from a direct comparison.

Instead, we compare to our implementation of the Monte-Carlo Markov-Chain sampling method proposed in [9] to sample from each layer a wave moves through, as well as a baseline where patterns are purely randomly selected for each wave.

Results

Evaluating 300 patterns is a tedious task. Hence, the need to help the expert to discover as many accepted patterns as possible and at most a small portion of rejected ones in the shortest amount of time. If the presented pattern is neither accepted nor rejected, it should be at least interesting. We aim to keep the number of "*Unsure*" labels at the lowest because they do not interest the expert and contribute the least to the learning process. Table 4 and 5 give the results of the evaluation of each method (WaveLSea, MCMC, and random waves) with the omniscient soothsayer on datasets presented in Table 3.

The beginning of each row corresponds to the pair (*dataset*, *used method*) indicated in the first cell. The following five cells contains the mean recall and recall standard deviation in parenthesis for each label: *Rejected*, *Uninterested*, *Unsure*, *Interested*, and *Accepted*. For a given dataset, the results of the three methods are gathered in groups of adjacent rows always in the following order: random waves, MCMC, and WaveLSea. Considering the first three labels, lower values are better, considering the last two labels, higher values are better. For a given label and a dataset, the bold value indicates the best method.

We observe that the WaveLSea method always has the best values for the *Uninterested*, *Interested* and *Accepted* labels. Comparing the MCMC mean recall to random wave results, we observe that the diffusion of the interest through weights does impact the exploration of the search space favorably. Moreover, the mean recall for *Accepted* is constantly and substantially higher for WaveLSea compared to the MCMC indicating that the MCMC may be stuck in a subpart of the search space or that defining prioritized and excluded area for the exploration leads to better results.

Also, WaveLSea often has the best values for the *Rejected* and *Unsure*. We can infer than in smaller search spaces (roughly a thousand and less), finding more *Rejected* pattern leads to a better discrimination of the search space, helping to reduce the number of sampled *Uninterested* and *Unsure* patterns.

Table 6 gives two-way ANOVA sum of squares p-values. Indicated values correspond to the case with the method as the independent variable for each label recall, providing the degree of significance found among the means of the two groups. In this table, the value $2.2e^{-16}$ correspond to the limit value of R language near 0. We can use the two-way ANOVA because runs for each pair (method, dataset) are repeated one hundred times giving us a statistically reliable sample size. Moreover, methods are used without prior knowledge for each iteration, and label identification restarts from zero each time meaning that results for each iteration are independent. Based on these values, we can affirm that the results of Table 4 and Table 5, indicating that the WaveLSea method performs better than the MCMC method, are statistically robust for all labels except for the *Rejected* label. When considering the *Rejected* label, only the omniscient soothsayer offers statistically robust improvements with WaveLSea.

We note that if we interest ourselves in the p-values for individual datasets, their value is always $2.2e^{-16}$ which is the same as for the combination of dataset and method. If we compute the two-way ANOVA with the results of random waves in addition, then all p-values are equal to $2.2e^{-16}$.

Table 4 Mean recall values and standard-deviation at the end of the exploration for WaveLSea (WLS), Wave random (rand), and Monte-Carlo Markovian-Chain (MCMC) for the first twelfth datasets presented in Table 3. For *Rejected*, *Uninterested*, and *Unsure*, lower values are better, for *Interested* and *Accepted*, higher values are better.

Dataset (method)	*Rejected*	*Uninterested*	*Unsure*	*Interested*	*Accepted*
UACC257H (rand)	7.76 (1.46)	3.73 (0.34)	2.20 (0.12)	2.43 (0.28)	2.88 (0.89)
UACC257H (MCMC)	6.64 (1.88)	2.82 (0.53)	1.73 (0.15)	4.5 (0.50)	7.32 (1.93)
UACC257H (WLS)	**3.63 (0.91)**	**1.46 (0.20)**	**1.08 (0.08)**	**6.29 (0.28)**	**26.28 (3.30)**
YeastH (rand)	12.49 (2.04)	3.24 (0.14)	2.78 (0.14)	3.44 (0.38)	4.68 (1.37)
YeastH (MCMC)	10.84 (2.61)	3.54 (0.57)	2.17 (0.17)	4.34 (0.52)	13.78 (2.85)
YeastH (WLS)	**6.95 (1.08)**	**1.27 (0.24)**	**1.43 (0.11)**	**6.85 (0.51)**	**37.73 (3.92)**
UACC257 (rand)	20.10 (3.28)	8.48 (0.72)	4.87 (0.24)	5.08 (0.61)	6.31 (2.48)
UACC257 (MCMC)	14.33 (4.02)	7.13 (1.02)	4.03 (0.33)	8.62 (1.04)	17.58 (4.14)
UACC257 (WLS)	**11.36 (1.51)**	**4.17 (0.35)**	**2.90 (0.17)**	**13.19 (0.57)**	**40.09 (2.97)**
Yeast (rand)	24.32 (3.85)	8.53 (0.91)	6.71 (0.33)	8.08 (0.82)	11.23 (3.05)
Yeast (MCMC)	21.62 (5.44)	8.55 (1.28)	5.61 (0.43)	10.37 (1.07)	26.04 (4.44)
Yeast (WLS)	**13.69 (1.97)**	**3.73 (0.50)**	**4.61 (0.21)**	**15.90 (0.79)**	**57.38 (4.79)**
SF-295H (rand)	38.89 (4.50)	19.41 (1.59)	14.14 (0.55)	14.82 (1.39)	14.50 (5.14)
SF-295H (MCMC)	**12.26 (6.15)**	12.55 (1.87)	15.58 (0.69)	18.25 (2.08)	25.72 (6.62)
SF-295H (WLS)	16.87 (2.92)	**10.72 (0.70)**	**13.83 (0.46)**	**21.78 (1.70)**	**59.80 (4.91)**
SW-620H (rand)	38.68 (5.31)	20.01 (1.42)	14.10 (0.53)	14.46 (1.31)	13.05 (4.90)
SW-620H (MCMC)	**12.03 (6.06)**	12.46 (1.93)	15.82 (0.64)	17.67 (2.05)	23.95 (6.73)
SW-620H (WLS)	17.26 (2.38)	**11.04 (0.84)**	**13.80 (0.37)**	**22.24 (1.09)**	**52.28 (4.58)**
SN12CH (rand)	39.85 (5.65)	19.08 (1.37)	**14.00 (0.55)**	15.76 (1.40)	13.99 (4.79)
SN12CH (MCMC)	**9.21 (6.14)**	11.51 (2.09)	15.85 (0.80)	19.09 (1.89)	24.23 (6.03)
SN12CH (WLS)	19.97 (2.54)	**9.53 (0.78)**	14.10 (0.35)	**22.04 (1.06)**	**58.69 (4.78)**
BZR_MD (rand)	50.23 (5.91)	28.47 (2.02)	**17.90 (0.69)**	21.94 (1.91)	38.51 (5.56)
BZR_MD (MCMC)	34.46 (8.99)	27.06 (3.22)	18.00 (1.04)	24.20 (2.10)	36.39 (6.75)
BZR_MD (WLS)	**21.12 (4.22)**	**12.05 (1.58)**	17.94 (0.45)	**38.63 (0.91)**	**55.91 (2.45)**
Mutagenicity (rand)	31.92 (9.11)	37.45 (2.83)	28.75 (1.10)	23.94 (2.50)	32.03 (7.87)
Mutagenicity (MCMC)	28.4 (11.30)	24.90 (3.77)	27.57 (1.20)	37.65 (2.93)	50.00 (9.82)
Mutagenicity (WLS)	**7.20 (2.87)**	**19.36 (1.36)**	**27.55 (0.81)**	**45.08 (1.93)**	**56.05 (5.37)**
MCF-7 (rand)	84.62 (8.00)	57.16 (3.07)	**39.73 (1.33)**	41.27 (3.46)	43.22 (12.47)
MCF-7 (MCMC)	**35.46 (11.72)**	41.54 (4.52)	44.26 (1.57)	44.02 (4.00)	62.21 (12.85)
MCF-7 (WLS)	47.92 (3.76)	**34.15 (2.16)**	40.60 (0.85)	**59.41 (1.51)**	**95.71 (3.51)**
PC-3 (rand)	83.37 (7.83)	55.23 (2.93)	**40.36 (1.33)**	41.09 (3.56)	34.27 (10.15)
PC-3 (MCMC)	**36.62 (14.26)**	40.87 (4.22)	43.88 (1.58)	46.20 (4.10)	55.64 (11.27)
PC-3 (WLS)	39.08 (5.21)	**33.47 (1.79)**	40.81 (0.66)	**60.01 (2.10)**	**93.00 (3.04)**
MOLT-4 (rand)	74.83 (8.85)	57.48 (3.11)	42.74 (1.15)	40.22 (3.31)	39.03 (9.75)
MOLT-4 (MCMC)	**37.84 (11.86)**	40.98 (4.47)	45.54 (1.74)	48.41 (4.66)	50.00 (12.18)
MOLT-4 (WLS)	38.69 (4.01)	**38.49 (1.61)**	**42.14 (0.76)**	**57.81 (3.05)**	**96.71 (5.78)**

However, when exploration covers the entire search space as it does for the AIDS and MUTAG datasets, there are few to no differences as observed in Table 5. Results for PTC-FM, PTC-MM, PTC-FR, and PTC-MR are not indicated in Table 5 as for all labels the mean recall is at 100.00. In this case, the interest of an exploration strategy lies in its rapidity for uncovering interesting patterns.

Let us observe the mean recall curves for each soothsayer to illustrate the speed at which interesting patterns are discovered.

Table 5 Mean recall values and standard-deviation at the end of the exploration for WaveLSea (WLS), Wave random (rand), and Monte-Carlo Markovian-Chain (MCMC) for the last twelfth datasets presented in Table 3. For *Rejected*, *Uninterested*, and *Unsure*, lower values are better, for *Interested* and *Accepted*, higher values are better.

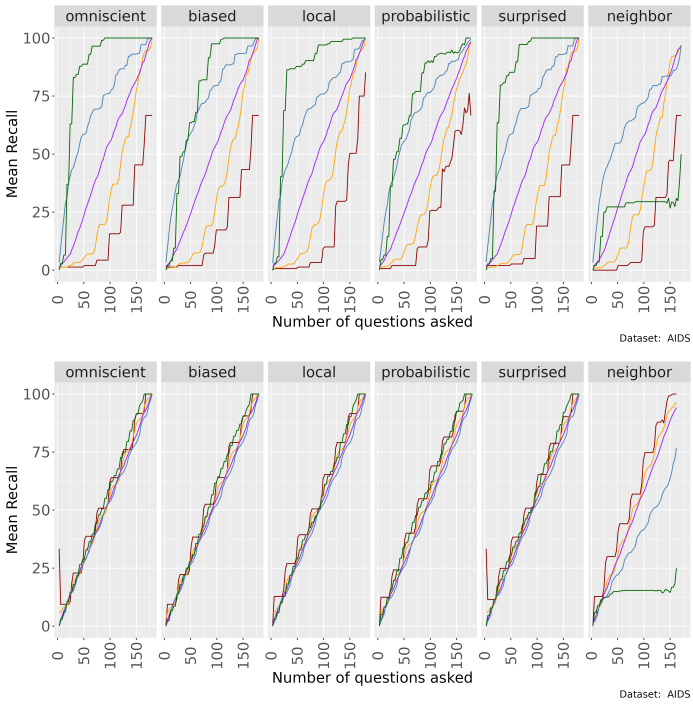
Dataset (method)	*Rejected*	*Uninterested*	*Unsure*	*Interested*	*Accepted*
SW-620 (rand)	74.20 (7.95)	59.55 (2.96)	**42.31 (1.30)**	39.05 (3.35)	44.23 (11.07)
SW-620 (MCMC)	40.31 (11.73)	43.22 (4.57)	45.61 (1.64)	44.97 (4.63)	55.93 (12.80)
SW-620 (WLS)	**39.54 (4.38)**	**37.51 (2.09)**	42.47 (0.77)	**57.18 (2.16)**	**99.00 (2.49)**
OVCAR-8 (rand)	78.55 (8.02)	58.49 (3.12)	**41.57 (1.28)**	42.95 (3.19)	40.26 (11.38)
OVCAR-8 (MCMC)	38.62 (13.98)	41.81 (5.01)	45.56 (1.95)	47.78 (3.94)	50.14 (12.60)
OVCAR-8 (WLS)	**38.54 (3.86)**	**36.59 (1.49)**	42.07 (0.37)	**59.89 (1.50)**	**99.50 (1.83)**
SF-295 (rand)	72.65 (9.21)	58.87 (2.81)	42.15 (1.20)	41.57 (3.45)	40.69 (11.34)
SF-295 (MCMC)	**29.31 (10.81)**	43.68 (4.32)	45.61 (1.72)	47.54 (4.44)	46.93 (11.45)
SF-295 (WLS)	44.31 (4.11)	**36.06 (1.70)**	**40.87 (0.79)**	**65.10 (1.84)**	**90.43 (3.40)**
NCI-H23 (rand)	79.95 (7.44)	58.14 (3.06)	42.47 (1.22)	41.07 (3.22)	38.38 (10.68)
NCI-H23 (MCMC)	**35.23 (11.99)**	43.72 (4.62)	45.61 (1.89)	47.66 (4.81)	43.14 (13.08)
NCI-H23 (WLS)	45.31 (3.26)	**34.82 (1.88)**	**40.93 (0.81)**	**65.47 (2.29)**	**96.29 (3.59)**
SN12C (rand)	79.41 (7.36)	60.37 (3.09)	**41.72 (1.13)**	41.60 (2.98)	39.47 (11.42)
SN12C (MCMC)	**36.31 (12.42)**	40.87 (4.55)	46.00 (1.70)	48.09 (4.75)	55.21 (12.09)
SN12C (WLS)	47.85 (4.17)	**36.83 (2.11)**	42.19 (0.82)	**59.36 (2.64)**	**99.14 (2.33)**
P388 (rand)	100.00 (0.0)	81.76 (3.06)	**61.86 (1.23)**	61.95 (3.68)	69.39 (9.98)
P388 (MCMC)	67.44 (16.51)	**63.47 (5.36)**	66.78 (2.09)	67.88 (4.96)	65.70 (15.13)
P388 (WLS)	**50.33 (5.57)**	67.37 (1.59)	65.06 (0.62)	**72.99 (1.80)**	**70.00 (0.0)**
AIDS (rand)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)
AIDS (MCMC)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)
AIDS (WLS)	**66.67 (0.0)**	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)
MUTAG (rand)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)
MUTAG (MCMC)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)
MUTAG (WLS)	100.00 (0.0)	**90.91 (0.0)**	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)

Table 6 List of the p-values for each label obtained via two-way ANOVA over 100 repetitions of the MCMC and WaveLSea methods with each of the twenty-four datasets and each of the soothsayers.

Soothsayer	*Rejected*	*Uninterested*	*Unsure*	*Interested*	*Accepted*
Omniscient	0.008851	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
Probabilistic	0.0206	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
Biased	0.1564	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
Local	0.639	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
Surprised	0.8763	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$
Neighborhood	$8.746e^{-05}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$	$2.2e^{-16}$

In Figure 4 and Figure 5, we show results for two datasets, *AIDS* and *Mutagenicity*. Complementary studies, datasets, and program executable are available at: https://github.com/Etienne-Lehembre/WaveLSea. The source code is available at: https://hal.science/hal-04057516/.

For Figure 4 and 5, the x-axis indicates the number of patterns proposed to the soothsayer, and the y-axis shows the *Recall*. Curve colors are matching labels as



AIDS

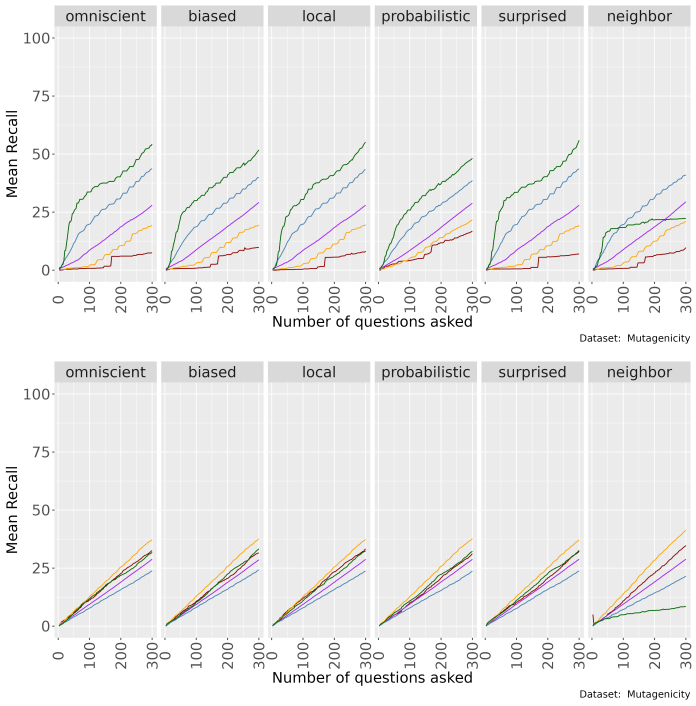
Fig. 4 Comparison of recall curves of accepted labels (green), interested labels (blue), unsure labels (purple), uninterested labels (orange), and rejected labels (red). Results of WaveLSea (top) and waved random sampling (bottom) for the AIDS data set.

follows: green for accepted, blue for interested, purple for unsure, orange for uninterested, and red for rejected. Each column corresponds to a soothsayer type. The colors correspond to the types of labels (see Table 1).[4]

The results show the almost constant progression of the percentage of found *accepted* labels, which is particularly fast in the first 50 queries. Even if in the densest networks these labels are not always all found, we notice that their percentage of discovery remains higher than those of the other labels, no matter the oracle used. Comparing the curves of a *wave* run where the patterns are randomly sampled (bottom-half figures) with the curves with WaveLSea where consequences are applied, we note that the results of WaveLSea are clearly better. We note that for *AIDS*, in the case of random sampling, the progression of the recall is linear and quasi-equivalent for each label type. This means that for each layer, the distribution of labels is equivalent. For *Mutagenicity*, purely random sampling recovers *Uninterested* patterns faster than *Accepted* or *Interested* ones.

Although the omniscient soothsayer often gets the best results, only the neighbor soothsayer clearly degrades the results. This suggests that mistakes induced by the

[4]Taken independently, recall curves are strictly increasing. However, as each run is not identical, all curves are not considered at the same time in the same layer. This is why the average curve of the results is not always strictly increasing.



Mutagenicity

Fig. 5 Comparison of recall curves of accepted labels (green), interested labels (blue), unsure labels (purple), uninterested labels (orange), and rejected labels (red). Results of WaveLSea (top) and waved random sampling (bottom) for Mutagenicity.

neighborhood may be further from ground truth in pattern languages from TUDATA. In addition, the curve for the *Interested* patterns generally increases more rapidly than the others. The curves of the *Rejected* patterns remain low, either for the whole experiment, or for a long period until many of the other types are nearly exhausted.

6.2 Results on experimental data

In the following section, we apply the WaveLSea algorithm to a real-world chemical dataset. As a dataset $\mathcal{D}$ we use BCR-ABL1 from CHEMBL23[5], a chemical graph dataset containing 1,485 molecules. The graph pattern set $\mathcal{L}$ called pharmacophores extracted from it is composed of 112,363 frequent labeled graphs, the orders of which go from 1 to 7 extracted with *Norns*[6] [20].

Pharmacophores are complete graphs, with pharmacophoric features as vertices, mapping given descriptors of chemical molecules in order to study their biological behavior. Each pharmacophore's support is composed of molecules which can be labeled either as active or inactive. The pharmacophores have been grouped into 1,533

[5]https://chembl.gitbook.io/chembl-interface-documentation/downloads
[6]https://valorisation.greyc.fr/catalog/logiciel?identifier=norns

Structured Equivalence Classes (SEC) based on identical support and structural connection in the relational POG. We note that even if we narrow down the set of patterns in the search space, the number of patterns to study still forms a combinatorial explosion. In this dataset, our classes are defined w.r.t. the BCR-ABL1 receptor, the first class consists of active molecules and the second of inactive ones.

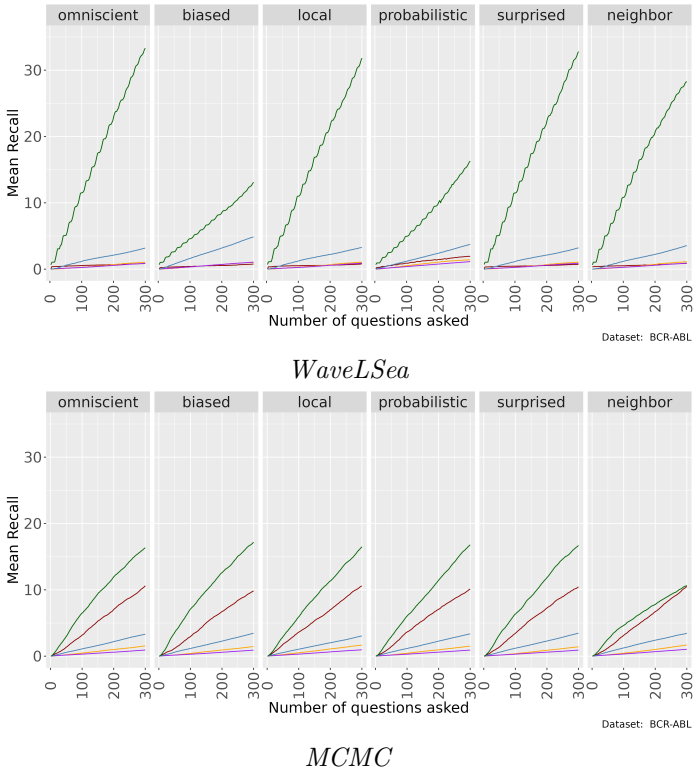


Fig. 6 Comparison of recall curves of accepted labels (green), interested labels (blue), unsure labels (purple), uninterested labels (orange), and rejected labels (red). Results of WaveLSea (top) and Bhuyian MCMC version for POG exploration (bottom).

In Figure 6, we display the mean results of the WaveLSea algorithm on BCR-ABL1 (top) compared to the mean results of our own implementation of the Bhuyian MCMC algorithm on POG [9]. We note that our method has a higher recall value for accepted labels for almost every soothsayer. But even when WaveLSea's accepted labels are fewer, they are traded in for interesting labels. Moreover, the MCMC method has a higher recall for rejected and uninteresting labels, regardless of the soothsayer. Therefore, WaveLSea displays better results than this version of the MCMC algorithm while assuring a stronger coherence between each sampling.

In Figure 7, we interest ourselves in the heuristics' percentage of the potential interest. The higher the percentage of an heuristic is, the higher its implication in the potential interest is. The *exploitation heuristic* is in blue, the *exploration heuristic* in

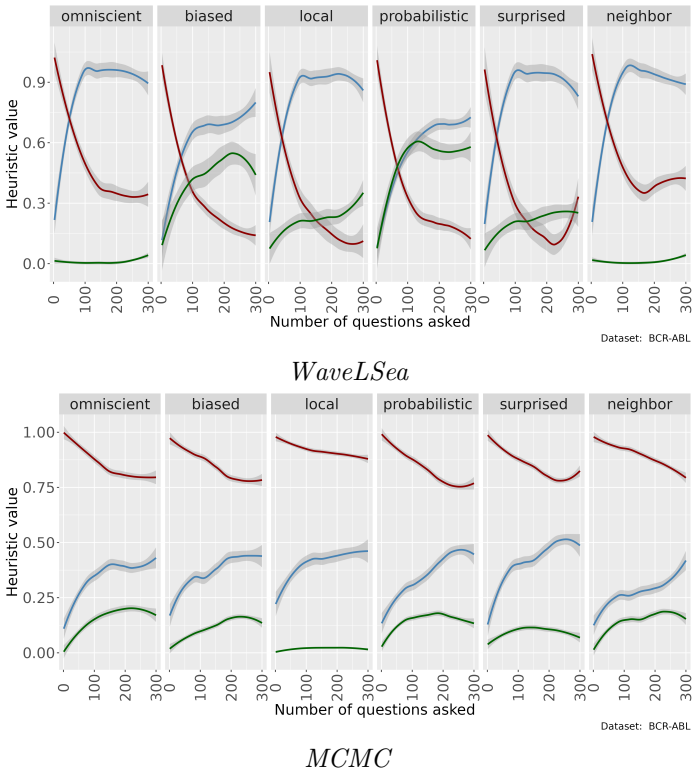


Fig. 7 Comparison of exploration heuristic (red), exploitation heuristic (blue), and ambiguity heuristic (green), between WaveLSea (top) and Bhuyian MCMC version for POG exploration (bottom).

red and the *ambiguity heuristic* in green. We observe that the WaveLSea algorithm swiftly transitions from the exploratory to the exploitative heuristic. The use of the prioritized areas in the top-samples allows to leverage the information transmitted by the expert as soon as possible. Whereas in the MCMC setting we remark that the exploration heuristic continues to make up a high percentage of the potential interest. We also observe that the WaveLSea method has a high percentage of ambiguity for the biased and the probabilistic soothsayers. We attribute this to early mistakes inferring an ambiguous interest for the rest of the exploration, leading to lower performance of WaveLSea with these two soothsayers.

In our implementation of Bhuyian MCMC, we set the exploration limit to 100 because it is not necessary to have a higher exploration limit regarding the size of our data sets. The value used to determined if the pattern is submitted to the expert or not is the potential interest in order to keep the analysis consistent with the WaveLSea algorithm.

For Figure 8, we generated an IPOG using WaveLSea and the omniscient oracle 100 times, each vertex is described by:

- *Potential Interest* — the mean value of I_p,

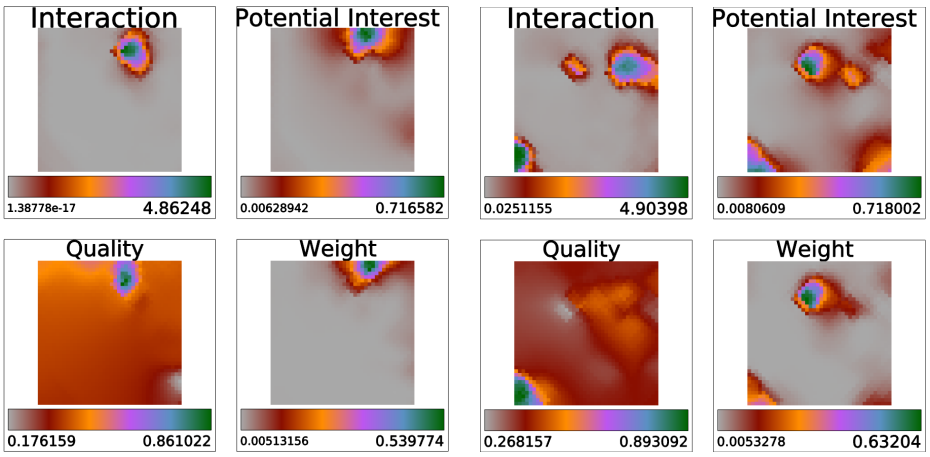


Fig. 8 Correlation between ground truth and normalized metrics used in WaveLSea.

- *Interaction* — the mean oracle feedback where 0 indicates that the vertex has not been sampled,
- *Quality* — the mean WRAcc of the vertex,
- *Weight* — its mean weight.

Then we clustered the IPOG's vertices, using a SOM (Self Organizing Map) in Tulip[7] with a grid of 40x40, and 10 epochs. Figure 8 *(a)* shows how strongly the four measures correlate through its mountain shape, with the best clusters at the center (in green) decreasing towards the worst clusters in the outer layer (in red). Especially *Weight* and *Quality*, except for the lower half of each subfigure, which corresponds to a part of the IPOG not explored. The correlation is even more pronounced in Figure 8 *(b)* where we can observe a clear link between the *Potential Interest* value and *Quality*. We note that *Weight* transcribes the quality by exacerbating the minimal and maximal values, while *Potential Interest* produces a more nuanced representation of the *Quality*. We also see the effects of conflicting interactions, where *accepted* patterns surrounded by *unsure* ones inhibit interest propagation, leading to lower *Potential Interest*.

In Figure 9, also done with Tulip and showing a visualization of the final IPOG, vertex colors correspond to the legend described in Table 1. The size of the vertices is proportional to their weight, and their label indicates the most present set of pharmacophoric features of their pharmacophores. Most visible vertices are green (i.e. accepted patterns) and blue (i.e. interested patterns) which indicates that most of the patterns proposed to the omniscient oracle are indeed interesting ones. It also confirms that our weight correctly maps the user's primary interest. From this type of figure, the expert studies the relation between the elements she finds interesting and the others. If she takes a step back and adds elements she did not already study, it will also help

[7] https://tulip.labri.fr/site/

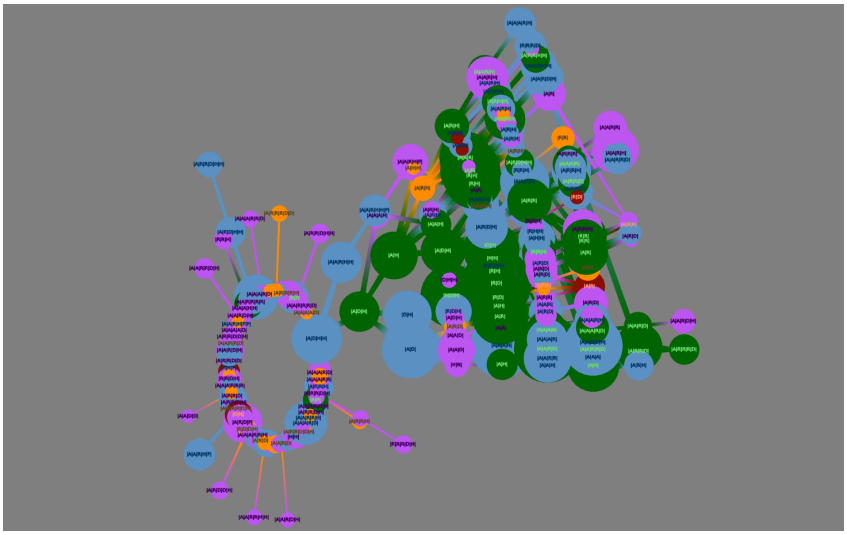


Fig. 9 Visualization of the IPOG explored by an omniscient oracle.

her in a second, more static exploration where she uses the result weights and evaluations of her interactive exploration to infer interestingness upon unexplored patterns. All of which should help her understand her own subjective interest.

7 User interface and human evaluation

7.1 Designing a user interface

Concerning the design of the user interface, several aspects were taken into account. First, the expert has to be able to navigate through the IPOG structure and interact with it at any time. This feature is crucial because, in our opinion, the relation between the studied patterns is one of the best tools for explicability. Siblings and lineage have to be easily accessible in order to help pattern evaluation for the same reason. Vertices should give insights into the potential interest and user interaction in order to give to the expert an interactive exploration of the result space. In this section, we present a user interface prototype designed for chemoinformatics experts.

The Figure 10 is a snapshot of the user interface used for the interactive exploration with WaveLSea. In the frame numbered 1 (in red), the user has access to the IPOG and its vertices. The IPOG is interactive and vertices can be selected to label them, display their lineage or siblings, and inspect the properties of their pattern or equivalence class. In the current snapshot, the user chose to visualize patterns in the layers 3 to 5. The legend of the interface is placed to the left and above it, a button is placed to unroll the access to settings, quality measures, saving options, loading data, and WaveLSea parameters. The frame numbered 2 (in green) shows the molecules containing the pharmacophore selected by the user in the first frame. The study of molecules is one of the main feature of interest for chemoinformaticians because they can infer from them the properties of the pharmacophore(s) contained in the vertex. The frame numbered

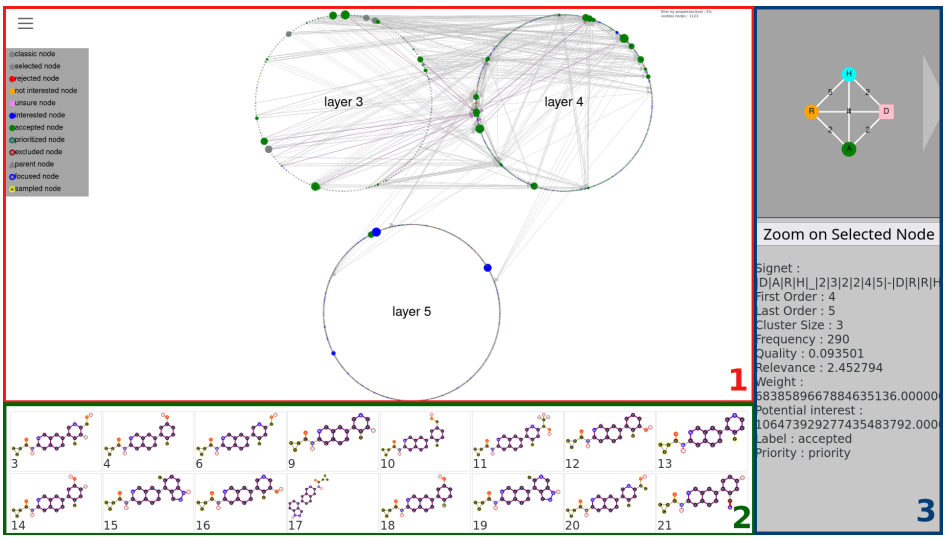


Fig. 10 User interface design

3 (in blue) in addition shows the generators of the equivalence class (pattern) selected by the user. This frame also gives details about the selected patterns such as their quality, relevance, weight, and potential interest. These details constitute the other main feature of interest for experts studying subgraph patterns.

By using this interface, the human expert can inspect the lineage and siblings of the sampled patterns to get a better insight into their attributes and simplify their study. Through the settings, the expert can map the size of the IPOG's vertices to the potential interest in order to draw out the patterns having the highest potential considering the interest currently expressed. These choices allow the expert to explore the result space through the waves of WaveLSea, but also on her own by using the key components of our method. If the expert does so, her intuitive samples will be the patterns with the highest potential interest or weight regarding the chosen settings.

7.2 Human evaluation

As our user interface is designed for chemoinformatics experts, we do not have access to a large potential sample of participants, unlike for less specialized tasks where we could make recourse to Mechanical Turk or similar platforms. Therefore, a lot of evaluation techniques used in Human Computer Interaction (HCI) cannot be applied in our case. Nevertheless, for our study, we can use some of the HCI guidelines for toolkit evaluations to inspire our own. Even if our interface is a prototype meant for experts, we can consider its relevancy towards future usage. In this section we perform a demonstration, more precisely a case study [43], of our method with the previously presented user interface. In this evaluation, we reduced the size of the dataset to explore in order to get a better grasp on the behavior of WaveLSea. The used dataset is a random sample of fifty percent of the molecules comprising the BCR-ABL1 dataset. The search space contains $1,812$ equivalence classes. The *searched corpus* contains 18

equivalence classes with the highest relevance value in the layer 3 and 4. The relevance value is the value used in the Oustanding Pattern Selector [41]. To search patterns in our corpus, we sampled one hundred patterns between the layers two to four containing $1,642$ patterns.

Using the user interface, we study for each sampled vertex its generators. In the current experiment, we express our interest through the subgraph relation. If one of the generators of the sampled equivalence class is a pharmacophore from the corpus or a direct subgraph or supergraph, then the sampled vertex is labeled as *accepted*. If one of the generators is a subgraph or a supergraph of one of the corpus pharmacophore, then the sampled vertex is labeled as *interesting*. If the generators of the sampled vertex are not related to one of the corpus pharmacophore then the vertex is labeled as *rejected*. The *unsure* and *uninteresting* labels are used in cases situated between the case of the *interesting* label and the case of the *rejected* label in a more fluent manner.

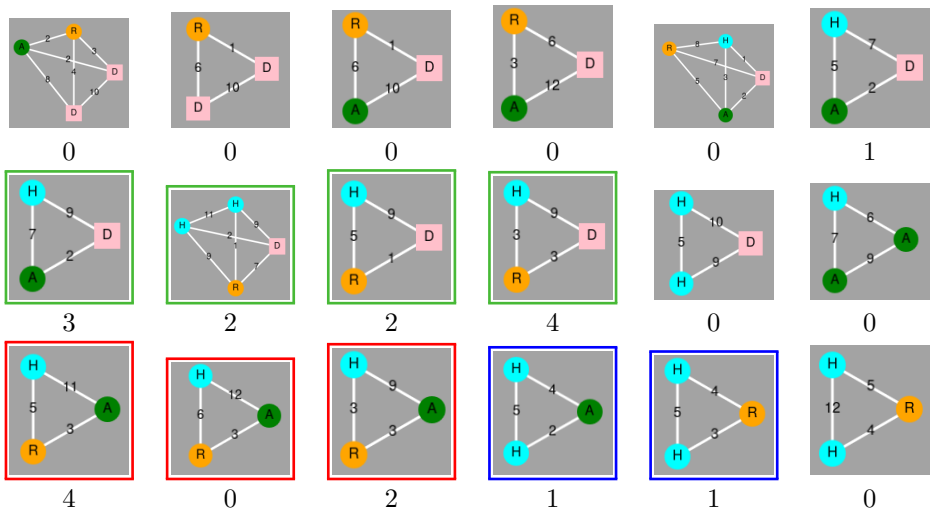


Fig. 11 Pharmacophore corpus and their uncover rate

In Figure 11, we see three main families of pharmacophores. The green family, sharing the subgraph $A-9-D$, are the ones found most often during our evaluation. We assume that their common substructure induces a convergence in the exploration of the IPOG containing the pharmacophores. The next main family is the red family which is characterized by the subgraph $R-3-A$. We note that the most found representative of this family is linked to the green family by the subgraph $R-5-H$. The blue family with the subgraph $H-5-H$ as common pattern, finally, is linked to the green family by the subgraph $H-3-R$. From these three families, we remark that our exploratory strategy leads to the diffusion of our interest concerning subgraphs linked to the corpus. Therefore, in this study, the WaveLSea algorithm behaved as expected, it diffused the expressed interest leading us to find patterns according to our interactions.

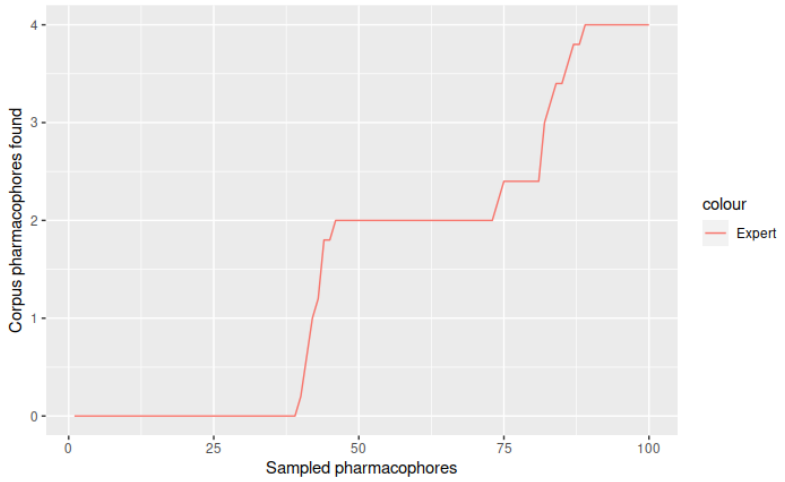


Fig. 12 Mean number of found pharmacophores from the corpus

In Figure 12, the graph shows the mean evolution of the discovered pharmacophores in the corpus during the exploration. At the end of the exploration, four of the eighteen pharmacophores in the corpus have been recovered, which means that we discovered 22.5% of the corpus while exploring only 6% of the dataset. If the pharmacophores had been sampled randomly, we would have found only 1.08 pharmacophores of the corpus. It is interesting to see that the method needs at least thirty samples in order to obtain a good first partition of the pharmacophore pattern space to find the first set of interesting patterns. After, it needs roughly another thirty sample to get a grasp of the new expert interest in this experiment. But interest acquisition time seems to reduce after that cold start.

Finally, to complete this case study, we support the demonstration with a user interview, a standard method of evaluation in HCI. This interview was done with an expert in chemoinformatic research from CERMN[8] laboratory interested in the BCR-ABL1 dataset. From the expert's perspective, the use of color-coded interactions helps in efficiently navigating the complex search space. By assigning distinct colors through each interaction, experts can quickly identify and prioritize areas of interest, enhancing the effectiveness of their exploration. Similarly, the WaveLSea pattern sampling allows the selection of relevant elements within the search space. This approach streamlines the exploration process by focusing attention on areas with the highest potential for valuable insights, saving time and resources.

Despite the initial learning curve, the division of the user interface into three panels proves to be useful for the expert. While it may require some time getting used to, this layout enables instant access to all necessary information, facilitating a comprehensive understanding of the studied pharmacophore. Once mastered, the expert can leverage this design to quickly analyze and interpret complex data, leading to more informed decision-making.

[8] https://cermn.unicaen.fr/

Moreover, the methodology introduced for pharmacophore exploration brings a fresh perspective to the field. By starting with patterns of smaller orders, chemoinformatic researchers can effectively track the evolution of interest and uncover subtle nuances within the search space. This iterative approach ensures that no valuable insights are overlooked, leading to a more thorough and insightful analysis of pharmacophores.

8 Conclusion

In this paper, we present an algorithm whose goal is to guide an expert during her exploration of a search space. Our work focuses on structured patterns and spaces, in particular graph patterns and partial order graphs. The algorithm is divided into three components: structure exploration, sampling, and graduated interaction. We define five interactions and their consequences. Each interaction-consequence pair influences the accessible search space or the pattern sampling.

The structure is explored through waves, going back and forth from the most general patterns to the most specific ones. The goal is to accompany the expert through her understanding of the studied objects and to avoid confusing her. The interactions either modify the reachable search space directly or modify the patterns' emulated interest, which is later used to assess patterns' potential interest through heuristics and thus affect future samples.

To evaluate our method, we simulated expert feedback using novel oracles called soothsayers based on an objective quality measure. We show that the method retrieves numerous high quality patterns, depending on the number of interactions with the oracle, even when the oracle's returns are noisy. Furthermore, we evaluate our method through a human evaluation using a user interface developed for our algorithm. In order to help the expert in her task, the user interface transcribes the components used by the algorithm into the graph view. Moreover, the expert can inspect the lineage and the siblings of the patterns to get a better grasp on its interestingness.

Acknowledgment

This work was partially funded by the ANR project InvolvD (ANR-20-CE23-0023). The user interface has been developed by Julien CRAND under the supervision of Etienne LEHEMBRE. The user interview has been conducted with Ronan BUREAU from CERMN laboratory of Caen Normandy University who has kindly accepted to experiment with our exploration method through the user interface. The selection of the statistical test benefits from the expertise of André SESBOÜÉ from the LMNO laboratory.

References

[1] Wang, Y., Li, Z., Farimani, A.B.: Graph neural networks for molecules, pp. 21–66. Springer (2023)

[2] Xiong, J., Xiong, Z., Chen, K., Jiang, H., Zheng, M.: Graph neural networks for automated de novo drug design. Drug Discovery Today **26**(6), 1382–1393 (2021)

[3] Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: ICDT, pp. 398–416 (1999). Springer

[4] Tan, P., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. Inf. Syst. **29**(4), 293–313 (2004)

[5] Raedt, L.D., Zimmermann, A.: Constraint-based pattern set mining. In: Proceedings of the 2007 SIAM International Conference on Data Mining (SDM), pp. 237–248

[6] Fournier-Viger, P., Gan, W., Wu, Y., Nouioua, M., Song, W., Truong, T., Duong, H.: Pattern mining: Current challenges and opportunities. In: International Conference on Database Systems for Advanced Applications, pp. 34–49 (2022). Springer

[7] Boley, M., Mampaey, M., Kang, B., Tokmakov, P., Wrobel, S.: One click mining: Interactive local pattern discovery through implicit preference and performance learning. In: Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics, pp. 27–35 (2013)

[8] Leeuwen, M.: Interactive data exploration using pattern mining. In: Holzinger, A., Jurisica, I. (eds.) Interactive Knowledge Discovery and Data Mining in Biomedical Informatics: State-of-the-Art and Future Challenges, pp. 169–182. Springer, Berlin, Heidelberg (2014)

[9] Bhuiyan, M., Hasan, M.A.: Interactive knowledge discovery from hidden data through sampling of frequent patterns. Statistical Analysis and Data Mining: The ASA Data Science Journal **9**(4), 205–229 (2016)

[10] Bhuiyan, M.A., Al Hasan, M.: Priime: A generic framework for interactive personalized interesting pattern discovery. In: 2016 IEEE International Conference on Big Data (Big Data), pp. 606–615 (2016). IEEE

[11] Yu, Y., Wang, W., Wu, N., Liu, H., Shao, M.: IISD: Integrated Interaction Subgraph Detection for event mining. Knowledge-Based Systems **240**, 108080 (2022)

[12] Rueping, S.: Ranking interesting subgroups. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 913–920 (2009)

[13] Dzyuba, V., Leeuwen, M.: Learning what matters - sampling interesting patterns. In: PAKDD 2017, Proceedings, Part I, pp. 534–546 (2017)

[14] Dzyuba, V., Leeuwen, M.V., Nijssen, S., Raedt, L.D.: Active Preference Learning for Ranking Patterns. In: 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, pp. 532–539. IEEE, Herndon, VA, USA (2013)

[15] Xin, D., Shen, X., Mei, Q., Han, J.: Discovering interesting patterns through user's interactive feedback. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '06, p. 773. ACM Press, Philadelphia, PA, USA (2006)

[16] Du, B., Zhang, S., Cao, N., Tong, H.: FIRST: Fast Interactive Attributed Subgraph Matching. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery And Data Mining, pp. 1447–1456. ACM, Halifax NS Canada (2017)

[17] Pei, J., Han, J., Lakshmanan, L.V.S.: Pushing convertible constraints in frequent itemset mining. Data Min. Knowl. Discov. **8**(3), 227–252 (2004)

[18] Kifer, D., Gehrke, J., Bucila, C., White, W.M.: How to quickly find a witness. In: PODS, pp. 272–283 (2003)

[19] Kuznetsov, S.O., Obiedkov, S.A.: Algorithms for the construction of concept lattices and their diagram graphs. In: PKDD, pp. 289–300 (2001). Springer

[20] Métivier, J.-P., Cuissart, B., Bureau, R., Lepailleur, A.: The pharmacophore network: a computational method for exploring structure–activity relationships from a large chemical data set. Journal of Medicinal Chemistry **61**(8), 3551–3564 (2018)

[21] Lehembre, E., Giovannini, J., Geslin, D., Lepailleur, A., Lamotte, J.-L., Auber, D., Ouali, A., Cremilleux, B., Zimmermann, A., Cuissart, B., Bureau, R.: Towards a partial order graph for interactive pharmacophore exploration: extraction of pharmacophores activity delta. Journal of Cheminformatics **15**(1), 116 (2023)

[22] Galbrun, E., Miettinen, P.: A Case of Visual and Interactive Data Analysis: Geospatial Redescription Mining. In: Proceedings of the Instant Interactive Data Mining Workshop at ECML-PKDD 2012, IID 12, Bristol, United Kingdom, pp. 1–12 (2012)

[23] Lavrac, N., Flach, P., Todorovski, L.: Subgroup discovery with cn2-sd. J. Mach. Learn. Res. **5**(2), 153–188 (2004)

[24] Gyongyi, Z., Garcia-Molina, H., Pedersen, J.: Combating web spam with trustrank. In: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB) (2004)

[25] Al Hasan, M., Zaki, M.J.: Output space sampling for graph patterns. Proceedings of the VLDB Endowment **2**(1), 730–741 (2009)

[26] Saha, T.K., Al Hasan, M.: Fs3: A sampling based method for top-k frequent subgraph mining. Statistical Analysis and Data Mining: The ASA Data Science Journal **8**(4), 245–261 (2015)

[27] Bosc, G., Boulicaut, J.-F., Raïssi, C., Kaytoue, M.: Anytime discovery of a diverse set of patterns with monte carlo tree search. Data mining and knowledge discovery **32**, 604–650 (2018)

[28] Dzyuba, V., Leeuwen, M.: Interactive discovery of interesting subgroup sets. In: Advances in Intelligent Data Analysis XII: 12th International Symposium, IDA 2013, London, UK, October 17-19, 2013. Proceedings 12, pp. 150–161 (2013). Springer

[29] Leeuwen, M., De Bie, T., Spyropoulou, E., Mesnage, C.: Subjective interestingness of subgraph patterns. Machine Learning **105**(1), 41–75 (2016)

[30] Gallo, A., De Bie, T., Cristianini, N.: Mini: Mining informative non-redundant itemsets. In: Knowledge Discovery in Databases: PKDD 2007: 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007. Proceedings 11, pp. 438–445 (2007). Springer

[31] Xin, D., Cheng, H., Yan, X., Han, J.: Extracting redundancy-aware top-k patterns. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data mining, pp. 444–453. ACM, Philadelphia PA USA (2006)

[32] Hien, A., Loudni, S., Aribi, N., Lebbah, Y., Laghzaoui, M.E.A., Ouali, A., Zimmermann, A.: A relaxation-based approach for mining diverse closed patterns. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I, pp. 36–54 (2021). Springer

[33] Hien, A., Loudni, S., Aribi, N., Ouali, A., Zimmermann, A.: Exploiting complex pattern features for interactive pattern mining. arXiv. arXiv:2204.04242 [cs] (2022)

[34] Amershi, S., Cakmak, M., Knox, W.B., Kulesza, T.: Power to the people: The role of humans in interactive machine learning. Ai Magazine **35**(4), 105–120 (2014)

[35] Wu, J., Liu, D., Guo, Z., Wu, Y.: RASIPAM: Interactive pattern mining of multivariate event sequences in racket sports. IEEE Transactions on Visualization and Computer Graphics, 1–11 (2022)

[36] Vermorel, J., Mohri, M.: Multi-armed bandit algorithms and empirical evaluation. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) Machine Learning: ECML 2005, pp. 437–448. Springer, Berlin, Heidelberg (2005)

[37] De Bie, T.: Subjective interestingness in exploratory data mining. In: Tucker, A., Höppner, F., Siebes, A., Swift, S. (eds.) Advances in Intelligent Data Analysis XII, pp. 19–31. Springer, Berlin, Heidelberg (2013)

[38] Yu, Y., Wang, W., Shao, M., Wu, N., Sun, Y., Sun, Y., Tian, Q.: Multi-users interaction anomalous subgraph detection for event mining. Neurocomputing **509**, 34–45 (2022)

[39] Giacometti, A., Soulet, A.: Interactive Pattern Sampling for Characterizing Unlabeled Data. In: Adams, N., Tucker, A., Weston, D. (eds.) Advances in Intelligent Data Analysis XVI vol. 10584, pp. 99–111. Springer, Cham (2017). Series Title: Lecture Notes in Computer Science

[40] Barr, E.T., Harman, M., McMinn, P., Shahbaz, M., Yoo, S.: The Oracle Problem in Software Testing: A Survey. IEEE Transactions on Software Engineering **41**(5), 507–525 (2015)

[41] Lehembre, E., Bureau, R., Crémilleux, B., Cuissart, B., Lamotte, J.-L., Lepailleur, A., Ouali, A., Zimmermann, A.: Selecting outstanding patterns based on their neighbourhood. In: IDA, pp. 185–198 (2022). Springer

[42] Todorovski, L., Flach, P., Lavrač, N.: Predictive performance of weighted relative accuracy. In: PKDD, pp. 255–264 (2000). Springer

[43] Ledo, D., Houben, S., Vermeulen, J., Marquardt, N., Oehlberg, L., Greenberg, S.: Evaluation Strategies for HCI Toolkit Research. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pp. 1–17. ACM, Montreal QC Canada (2018)