Wave Top-k Random-d Family Search: how to guide an expert in a structured space (supplementary material)

Abstract. In this paper, we develop a method Wave Top-k Random-d Family Search (WTRFS) that include the user interaction in order to guide her through data mining results. This work aims to replace the descriptor declaration step used in interactive data mining. For this we exploit the hypothetical relation between experts' patterns of interest. We empirically demonstrate that WTRFS returns first the most relevant results for the user. Moreover, even if the users' interactions are not perfect, we observed that WTRFS behavior isn't altered.

Appendices

A Weighting Algorithm

Algorithm 1 Lineage weighting

Require: $\mathbb{V}$ a vertex set, $v \in \mathbb{V}$ a vertex, and A an expert interaction.
Ensure: The set of vertices where the ancestors and descendants of v with modified weights regarding A.

$\mathbb{V}' \leftarrow \mathbb{V}$; $L_a \leftarrow \mathcal{P}(\{v\})$; $L_d \leftarrow \mathcal{C}(\{v\})$
$w \leftarrow |poids(v)|$
$poids(v) =$ Weighting(v, A, w)
$k \leftarrow 2$
while $L_a \neq \emptyset$ or $L_d \neq \emptyset$ **do**
    for $a \in L_a$ **do**
        $poids(a) =$ Weighting$(a, A, w * \frac{1}{2^k})$
    end for
    for $d \in L_d$ **do**
        $poids(d) =$ Weighting$(d, A, w * \frac{1}{2^k})$
    end for
    $L_a \leftarrow \mathcal{P}(L_a)$; $L_d \leftarrow \mathcal{C}(L_d)$
    $k \leftarrow k + 1$
end while

In Algorithm 1, we begin by initializing the set of vertices carrying the modifications $\mathbb{V}'$ with the set of vertices $\mathbb{V}$, the set of ancestors of the vertex v noted L_a with the set of parents of v noted $\mathcal{P}(v)$, and the set of descendants of the vertex v noted L_d with the set of children of v noted $\mathcal{C}(v)$ (line 1). We recover the absolute value of the weight of v (line 2) then we update the weight of v (line 3) (which can either double in the negative, or become null, or double in the positive). Line 5 to 15, we repeat a loop as long as the set of ancestors and the set of descendants of v has not been processed. Line 6 to 8 and 9 to 11, we iterate on all the ancestors and descendants to modify their weight. The impact of the weight of v is decreased by $\frac{1}{2^k}$ according to the distance to v, with k initialized to 2 (line 2). This value was chosen empirically. We also experimented with a variant where the scattered weight was decreased as a function of the number of children/parents of a vertex without noticing any practical differences. Line 13 we retrieve the parents of the ancestors contained in L_a to update the set and the children of the descendants in L_d to do the same. Then we increase k by 1 in line 14 to reduce the effect of changing weights on additional depth layers.

B Sampling Algorithm

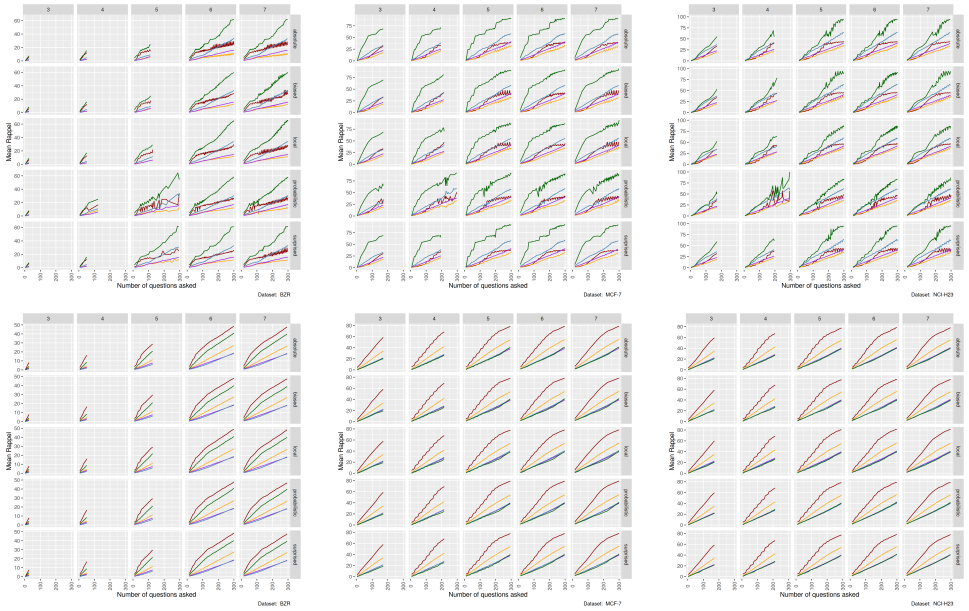
Algorithm 2 Sampling$_{k,d}(L, \mathbb{G})$

Require: $\mathbb{G}$ the studied graph, L a layer of $\mathbb{G}$, k the number of top draws, d the number of random draws.
Ensure: $\mathbb{S}$ the vertices sampled set L.

1: $L^+ \leftarrow \{\forall v \in L | v \in \mathbb{V}^+\}$
2: $L^? \leftarrow \{\forall v \in L | v \notin \mathbb{V}^+ \ \ \& \ \ v \notin \mathbb{V}^-\}$
3: $\mathbb{S} \leftarrow \emptyset, \mathbb{S}' \leftarrow \emptyset$
4: **if** $|L^+| \geq k$ **then**
5: $\quad \mathbb{S} \cup \{v_1, \ldots, v_k \in L^+ \mid \nexists v_{k+1} : f_p(v_{k+1}, \mathbb{G}) > f_p(v_i, \mathbb{G}), 1 \leq i \leq k\}$
6: **else**
7: $\quad \mathbb{S} \cup \{v_1, \ldots, v_{k-|L^+|} \in L^+ \mid \nexists v_j : f_p(v_j, \mathbb{G}) > f_p(v_i, \mathbb{G}), 1 \leq i \leq k - |L^+|\}$
8: $\quad \mathbb{S} \cup \{v_1, \ldots, v_{k-|\mathbb{S}|} \in L^? \mid \nexists v_j : f_p(v_j, \mathbb{G}) > f_p(v_i, \mathbb{G}), 1 \leq i \leq k - |\mathbb{S}|\}$
9: **end if**
10: **if** $|L^+| \geq d$ **then**
11: $\quad \mathbb{S}' \cup \{v_1, \ldots, v_d \in L^+$randomly drawn based on Equation (4)$\}$
12: **else**
13: $\quad \mathbb{S}' \cup \{v_1, \ldots, v_{d-|L^+|} \in L^+$randomly drawn based on Equation (4)$\}$
14: $\quad \mathbb{S}' \cup \{v_1, \ldots, v_{d-|\mathbb{S}'|} \in L^?$randomly drawn based on Equation (4)$\}$
15: **end if**
16: **return** $\mathbb{S} \cup \mathbb{S}'$

In the algorithm 2 we start by initializing the sets L^+ and $L^?$ containing respectively the elements with priority and without opinion with respect to the exploration line 1 and 2. We initialize the set $\mathbb{S}$ sampled at line 3. We perform the k Top draws in line 4 to 9. The pseudo-random draws are performed from line 10 to line 15. The draws are performed in the priority elements in the conditions lines 4 and 10. We return the set of samples on line 16.

C Courbes de rappel pour la sélection aléatoire



BZR *MCF-7* *NCI-H23*

FIG. 1 – *Mean recall for BZR, MCF-7, and NCI-H23 with WTRFSon top and a wave course with random sampling on the bottom.*

In Figure 1, for each illustration, the x-axis shows the number of patterns offered to the oracle, and the y-axis shows the percentage of labels discovered by label type. The columns correspond to the IPOG layers and the rows correspond to the oracle types. The colors correspond to the label types.

We notice that the curve of the labels *Rejected* is always above the others during wave run with random sampling. This curve is followed either by that of the elements *Accepted* or by that of the elements *Interesting*. Finally we see that, except for the smallest spaces, no curve reaches the 100

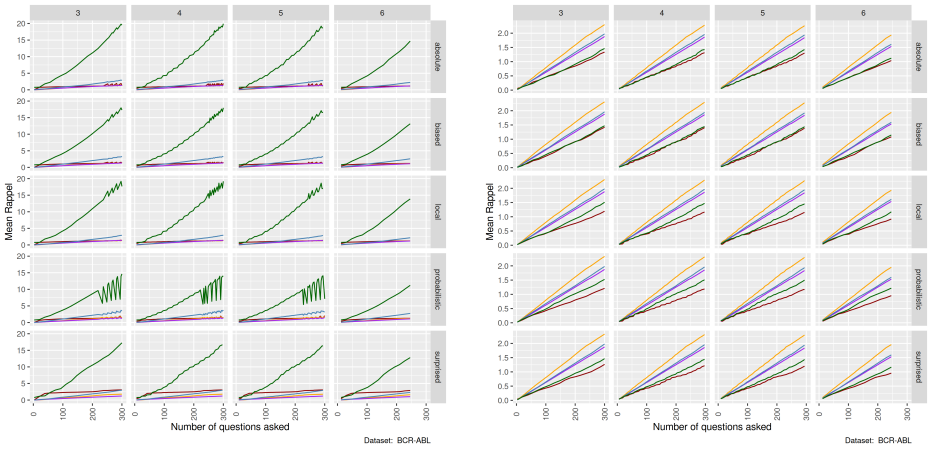
On the whole, the results of Wave Top-k Random-d Family Searchare better than the results obtained by a wave run with random sampling.

D Results on BCR-ABL.

In this section, we apply our method to a chemical dataset studied at CERMN[1]. The studied dataset is BCR-ABL obtained from CHEMBL23[2], is a set of chemical graphs containing 1485 molecules. The set of extracted pattern subgraphs is composed of 112,363 frequent labeled

[1] http://cermn.unicaen.fr/

[2] https://chembl.gitbook.io/chembl-interface-documentation/downloads



(a) *(b)*

FIG. 2 – *Average recall of tags discovered in BCR-ABL with WTRFS(a) and random sample with wave path (b).*

subgraphs. The extraction of subgraphs with order between 1 and 7 was performed with a frequency of 10 occurrences by *Norns*[3] (Métivier et al., 2018).

The frequent subgraphs, called *pharmacophores*, are complete graphs whose vertices are pharmacophore markers. Pharmacophore markers represent chemical properties of molecules that influence their biological behavior. Each pharmacophore has its support composed of molecules that can be classified as active or inactive. The pharmacophores are grouped into 1,533 equivalence classes identified from identical carrier and structural linkage in IPOG.

In this dataset, our first class will therefore be formed of active molecules and the second class of inactive molecules. The activity being determined in relation to a receptor.

In the figure 2, for each illustration, the x-axis shows the number of patterns proposed to the oracle, and the y-axis shows the percentage of discovered labels by label type. The columns correspond to the POG layers and the rows correspond to the oracle types. The colors correspond to the types of labels shown in Table 1 of the paper.

We note, on this set of graphs, a strong difference between the results of WTRFS*(a)* and those of random sampling *(b)*. In random sampling, the most present labels (Interesting, Uncertain, Interesting) are the ones that are most proposed to the oracles, resulting in a very low proportion of labels that are discovered. The recall of the labels *Rejected* and *Accepted* has thus difficulty to exceed 1.

References

Métivier, J.-P., B. Cuissart, R. Bureau, and A. Lepailleur (2018). 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.

[3] https://valorisation.greyc.fr/catalog/logiciel?identifier=norns