Substructure search optimizations

2 апреля 2023 г.

1 Abstract

TODO

2 Abount Fingerprint-based screening

Что-то примерно, как описано в Sachem поиске

3 Algorithm description

3.1 Notation

- $\mathbb{M}$ — множество тех молекул, среди которых хотим организвовать поиск.
- $\mathbb{F}$ — множество fingerprints построенных по множеству $\mathbb{M}$.
- $\mathbb{T}$ — бинарное дерево поиска для фингерпринтов.
- $fp : \mathbb{M} \to \mathbb{F}$ — функция, которая строит fingerprint по молекуле
- $F_M = \{F' \in \mathbb{F} \mid fp(M) \text{ is submask of } F'\}$ — множество всех фингерпринтов, являющихся надмаской фингерпринта $fp(M)$. стоит ли формально определить, что такое«is submask of»?

3.2 Общая идея

- Не будем работать напрямую с молекулами. Для заданного множества $\mathbb{M}$ построим множество $\mathbb{F}$ и будем решать задачу о поиске $\{F' \in \mathbb{F} \mid F \text{ is submask of } F'\}$ для заданного фингерпринта F
- Тогда для поиска всех надструктур молекулы M сначала множество $F_M = \{F' \mid F' \text{ is submask of } fp(M)\}$ Стоит ли формальнее определить, что такое «is submask of»?, с помощью BallTree. Тогда ответом будет $\bigcup_{F' \in F_M} fp^{-1}(F') = \{M' \mid M' \text{ is substructure of M }\}$, где $fp^{-1}(F') =$

$\{M' \in \mathbb{M} \mid fp(M') = F'\}$, а проверка «$M'$ is substructure of M» осуществляется с помощью сторонних алгоритмов.

- Для эффективного поиска F_M будем использовать BallTree с метрикой Russel-Rao для множества $\mathbb{F}$. В общем-то, это довольно частный случай BallTree, поэтому будем описывать ниже нашу идею, без привязки к обобщённой версии BallTree. слишком неаккуратно написана связь нашего дерева с BallTree

3.3 Описание дерева

- В листьях хранятся некоторые множества фингерпринтов. Обозначим множество в листе l как S_l
- В каждой вершине хранится OR всех фингерпринтов, лежащих в листах данного поддерева. Обозначим данное значение в вершине v как C_v

хочется более формально определить структуру дерева

3.4 Поиск в дереве

- Для полученной молекулы M строим $F = fp(M)$. И для F запускаем поиск в дереве.
- Рекурсивно спускаемся в обоих детей, начиная с корня.
- Если оказались в врешине v для которой F is not submask of C_v, то прекращаем рекурсивный спуск из v.
- Если дошли таким образом до листа l и F is submask of C_l, то далее перебираем все элементы S_l и добавим в F_M те фингерпринты, которые которые оказались надмаской F.

Псевдокод процедуры поиска описан в алгоритме 2. Вспомогательная функция для поиска фингерпринтов в поддереве описана в алгоритме 1.

Таким образом, для эффективного поиска надо построить дерево так, чтобы как можно больше листов было отсеено при спуске. Как построить такое дерево обсудим ниже.

в какое-то место надо засунуть про то, что такой подход можно распараллелить

3.5 Построение дерева

Будем индукционно строить дерево, увеличивая его глубину. Для начала сделаем одну вершину, в которую положим все фингерпринты. Далее будем выполнять процедуру разделения попалам для всех листов, пока не достигнем достаточной глубины.

Algorithm 1 Поиск всех подходящих фингерпринтов в поддереве

Require: v — вершина в дереве. F — фингерпринт, надмаски которого ищем

Ensure: $\{F' \mid F'$ находится в листе в поддереве $v \ \wedge \ F$ is submask of $F'\}$

procedure FindInSubtree(v, F)
  if F is not submask of C_v **then**
    return $\varnothing$
  else if v is leaf **then**
    return $\{F' \in S_v \mid F$ is submask of $F'\}$
  else
    $left \leftarrow$ FindInSubtree(LeftChild(v), F)
    $right \leftarrow$ FindInSubtree(RightChild(v), F)
    return Concatenate($left, right$)
  end if
end procedure

Algorithm 2 Поиск всех надструктур заданной молекулы

Require: M — молекула

Ensure: $\{M' \in \mathbb{M} \mid M$ is substructure of $M'\}$

procedure FindMetaStructures(M)
  $F \leftarrow fp(M)$
  $F_M \leftarrow$ FindInSubtree($treeRoot, F$)
  return $\bigcup_{F' \in F_M} fp^{-1}(F')$ ▷ $fp^{-1}(F) = \{M' \in \mathbb{M} \mid fp(M) = F\}$
end procedure

4 Benchmarks

время работы на базе pubchem на разных aws машинах. Какой процент молекул отсекаем, сколько в среднем «бесполеных» вершин, в которых придётся идти и влево и вправо, какой процент работает наша часть, а какой процент работает «чёрный ящик»

5 References

TODO

- Найти что-то про описание ball tree и метрику Rassel-Rao
- Pubchem
- Indigo fingerprint, RDKit fingerprint
- Что-то про AWS