Mining Molecular Fragments: Finding Relevant Substructures of Molecules

Christian Borgelt, Michael R. Berthold

ICDM 2002.

Lecturers: Carlo Cagli and Alessandro Poli

Milano, 28 Aprile 2008
Goal

Analyze large collections of molecules for finding some regularities among molecules of a specific class

Application Example: *find new drug candidates based on experimental evidence of activity against a certain disease gathered by screening hundreds of thousands of molecules*

Presented method generates fragments by embedding them in all appropriate molecules in parallel

Fast search and suppression of redundant search
Previous Approaches

Regularities among molecules searched by using descriptors representing:

- certain substructures of interests, such as aromatic rings or some other predefined small group of atoms [*]
- pairwise atom distances
- 3D molecule arrangements

Similarity between molecules computed as a distance function on these descriptors

Algorithms that finds linear fragments i.e. chains of atoms [**]

Approaches that find arbitrary connected substructures, but relying on frequent reembeddings of fragments [***]


Association Rules and Frequent Itemsets

Association rules is a data mining method for *market basket analysis*:
- aims at finding regularities in shopping behavior of customers
- find sets of products that are frequently bought together
- from the presence of certain products in a shopping cart one can infer that certain other products are present
- e.g.: bread => butter

Association rule algorithms work in two steps:
1. *Frequent itemsets* are determined. These are sets of items that have at least a given minimum *support*, i.e., occur in at least a given percentage of all transactions
2. Association rules are generated from frequent itemsets

Here we focus on the first step, because we are not concerned with generating rules

In molecular substructure analysis we have to take the chemical connectivity (bonds) into account as well
Frequent Itemsets Identification

In order to find frequent itemsets, we have to count the transactions containing them. This task consists in traversing a tree structure and determining the values of the counters in its nodes. The tree is unbalanced, because we are dealing with sets, not sequences.

Two algorithms:

* **Apriori** [*]
  - breadth first search
  - determines the support of an itemset by explicit subset tests on the transactions
  - the tree data structure can consume a lot of memory
  - the subset tests can be costly

* **Eclat** [**]
  - does a depth first search
  - determines the support of an itemset by intersecting the transaction lists for two subsets, the union of which is the itemset
  - the advantage is that not all counters have to be kept in memory
  - several transaction lists have to be kept in memory at the same time—lists that can be very long, especially for small itemsets


Molecules modeled as attributed graphs

- Vertex ➔ atom (attributes: atom type, charge, etc.)
- Edge ➔ bond between atoms (attributes: bond type, single, double, triple, or aromatic)

Goal: find substructures that have a certain minimum support in a given set of molecules, i.e., are part of at least a certain percentage of the molecules

The graphs may be chains or trees or may contain an arbitrary number of cycles

The search is carried out by traversing a tree of fragments of molecules. The root of the tree is the core structure to start from
Frequent Substructures of Molecules (2)

Going down one level in the search tree means to extend a substructure by a bond (and maybe an atom, if the bond does not close a ring)

With a single atom at the root of the tree, the root level contains the substructures with no bonds, the second level those with one bond, the third level those with two bonds and so on

Eclat approach (depth first search and intersections of transaction lists) is preferable, because substructure tests (check whether a given attributed graph is a subgraph of another attributed graph), are extremely costly, and even storing only the topmost levels of the tree can require a prohibitively large amount of memory
Frequent Substructures of Molecules (3)

The given core structure is embedded into all molecules, resulting in a list of embeddings.
Each embedding consists of references to a molecule that point out the atoms and bonds that form the substructure.
In a second step each embedding is extended in every possible way, by considering all bonds that start from an atom already in the embedding.
Explored atoms and bonds are marked and only unmarked bonds from marked atoms are considered as possible extensions.
The resulting extended embeddings are then sorted into equivalence classes, each of which represents a new substructure.
Each of these new substructures corresponds to a child node in the search tree, each of which is then recursively processed.
Search Tree Pruning

*Support based pruning*: subtrees of the search tree can be pruned if they refer to substructures not having enough support.

*Size based pruning*: the search tree is pruned if a user-defined threshold for the number of atoms in a fragment has been reached.

*Structural pruning*: ensures that every itemset is considered in one branch only, even though adding items in different orders can yield the same itemset.
We cannot define a *global* order of the atoms of the molecules, which would correspond directly to the order of the items. An atom is assigned a number reflecting the step in which it was added. That is, the core atom is numbered 0, the atom added with the first bond is numbered 1 and so on. When the extended embedding is to be extended itself, only bonds that start from atoms having numbers no less than this recorded number are considered. Order is furthermore determined by bond type and atom type.
Example
An example-rules

Brief summary of the algorithm rules:

• Starting from a seed each atom is labeled with a progressive number:
  seed labeled “0”
  atoms added at successive steps are labeled “1”, “2”, … according to the step progression.
  single bonds come before than double ones

• Each embedding is extended in every possible way according to the following rules

• Only the last added atom or the one preceding it can be a starting point for the next atom: at the step #3, atom labeled 1 cannot be connected anymore.

• The search algorithm is depth-first
An example-molecules

Six molecules research database
The search algorithm starts with the Sulfur atom
Let define a minimum support of 50%: a substructure must appear at least in three of the six molecules above
An example - generating the search tree

- SUBSTRUCTURE
- DATABASE’S MOLECULES
- NUMBER OF OCCURRENCES
An example - generating the search tree

1° step: Sulfur seed
2° step: first extension (S-C; S-N; S-O; S=N)

Notes:
The order of extension reflects the search algorithm order (see rules).
At the third stage:
C-S-N has no child in which a second carbon atom is attached to S
C-S=N has no child in which another atom is attached to S by a single bond
C-C-S has no child at all
C-S-C has a double occurrence in $b$ and $c$ molecules
An example-results

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1)</td>
<td>C-S-N</td>
<td>2)</td>
</tr>
<tr>
<td>3</td>
<td>(50%)</td>
<td>3</td>
</tr>
<tr>
<td>4)</td>
<td>C-S-O</td>
<td>5)</td>
</tr>
<tr>
<td>4</td>
<td>(67%)</td>
<td>3</td>
</tr>
</tbody>
</table>

Results of the fragments search with absolute and relative occurrence
Note that fragments like C-C-S is not supported because of C-C-S-N fragment.
Likewise O-S-N, S=N, S
C-S-C has a double occurrence because in order to correct the pruning error search rules have been relaxed
An example-embedding a core structure

Use a seed constituted by a rare element is a good point.

In order to start with a seed which is already a fragment it is useful the following observation:
Embedding a core structure is the same as finding a common substructure of the molecule and the core that is as big as the core itself.
An example-experimental results

Dataset from National Cancer Institute:
DTP AIDS Antiviral Screen dataset [*]

**CA** Compounds providing at least 100% protection to the CEM cells
**CM** Compounds providing at least 50% protection to the CEM cells
**CI** Compounds not answering to the previous constraints

46,316 compounds available [**]
37,171 used

Belongings:
**CA**(325), **CM**(877), **CI**(35,969)


[**] http://dtp.nci.nih.gov/docs/aids/aids_data.html
First stage: start with single atoms seeds

<table>
<thead>
<tr>
<th>Atom</th>
<th>CA</th>
<th>CM and CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Carbon</td>
<td>325 (100.0%)</td>
</tr>
<tr>
<td>O</td>
<td>Oxygen</td>
<td>311 (95.7%)</td>
</tr>
<tr>
<td>N</td>
<td>Nitrogen</td>
<td>276 (84.9%)</td>
</tr>
<tr>
<td>S</td>
<td>Sulfur</td>
<td>143 (44.0%)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Se</td>
<td>Selenium</td>
<td>6 (1.9%)</td>
</tr>
</tbody>
</table>
An example-Nitrogen based fragments

A *contrast structures* search is performed, with a minimum support for the compounds and a maximum support on the complement.

The first two fragments have the same coverage, differing actually just for one bond.

The third fragment is almost identical, but a missing bond makes a slightly smaller fragment with a much higher coverage.
An example-Sulfur based fragments

10% minimum compound coverage, 0.5% complement

<table>
<thead>
<tr>
<th>Fragment 1</th>
<th>CA: 11.9%</th>
<th>Fragment 2</th>
<th>CA: 11.9%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cl/CM: 0.3%</td>
<td></td>
<td>Cl/CM: 0.4%</td>
<td></td>
</tr>
</tbody>
</table>

The compounds differ only for the $\text{SO}_3$ group location
These fragments are common to 11 of the 13 Dyes and Polyanions
An example-Selenium based fragments

30% minimum compound coverage, 5% complement

<table>
<thead>
<tr>
<th>Fragment 1</th>
<th>Fragment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>freq. CA: 33.3%</td>
<td>freq. CA: 33.3%</td>
</tr>
<tr>
<td>freq. Cl/CM: 3.8%</td>
<td>freq. Cl/CM: 3.0%</td>
</tr>
</tbody>
</table>

The first fragment picks out all the seven compound classes, while the second picks out one compound less.
An example - Treatment of aromatic bonds

Aromatic bonds are actually not uniquely modelled.

The algorithm models the aromatic bonds as either single or double bonds, using a flag to indicate aromaticity.
The search can be performed either with or without the flag.

Results on fragment extraction from a set of steroids made of 4-ring in which one is single bonded.

\[ \text{single bond} \neq \text{aromatic bond} \]
\[ \text{single bond} = \text{aromatic bond} \]
Conclusions

The algorithm maintains parallel embeddings of a fragment into all molecules throughout the growth process and exploits a local order of the atoms and bonds of a fragment to prune the search tree, which results in faster search and allows for a restricted depth first search algorithm.

Proposals:
- Awareness of interesting fragments (not atom by atom search).
- Uncompleted matching (wild cards, bond patterns, other constraints)
- Search of functionally equivalent structure