University of Southern Denmark

DM840

Algorithms in Cheminformatics autumn 2019

Assignment 2

Author:
Sebastian Eklund Larsen
<selar16>

January 8, 2020

Introduction

In this assignment we try to implement a program which can create valid mappings in regards to mono-cyclic transitions.

Valid mapping

The majority of chemical reactions are said to have a mono-cyclic transition, which means that given a set of educts and a set of products, there exists a mapping from the atoms of the educts to the atoms of the products, where the bindings only found in the educt and the bindings only found in the products, create a cycle of alternating bindings. This means that all bindings not found in both educt and product must be a part of the cycle. Given a valid mapping to a cycle, one can infer the rules of the transition. When searching for rules we have the variable k, which determines the maximum distance an atom has to be from the cycle to be included in the rule. The variable c determines the size of the cycle we are looking for.

Implementation

In program we try to create possibly valid mappings, and then verify if they have a mono-cyclic transition. We try every possible unique mapping where atoms of the same type are mapped to each other, and run a verification algorithm an the mapping.
In order to verify we first the set of edges that only appear in the educt and do the same for the edges in from the products. We make sure that these two sets are of the same size, as they otherwise would not make an alternating cycle. Then we traverse the edges, alternating between the two sets. If we end with the same node the first edge started with, both sets are empty and the number of edges traversed is divisible by two. Then we know that the mapping contains a cycle of the vertices of the two sets in the order we traversed them.
When we have a valid mapping we add the rule depending on the parameters k and c.

Result

The program has been tested by manually changing the graphs and variables in test.py, granted the the program does not have pruning implemented, so

testing took a long while, meaning not many results have been gathered.

Educts	Products	k	c	n	t
C=C, C=C	C1CCC1	0	4	2	319
C=C, C=C	C1CCC1	1	4	2	318
O, Cl, CC(=O)OCC	Cl, OCC, CC(=O)O	0	6		
O, Cl, CC(=O)OCC	Cl, OCC, CC(=O)O	1	6		
O, Cl, CC(=O)OCC	Cl, OCC, CC(=O)O	2	6		
C1C(O)CC(O)C(O)C1	C=CO, C=CO, C=CO	0	6		
C1C(O)CC(O)C(O)C1	C=CO, C=CO, C=CO	1	6		
CC=CC=CC, OC1C=CC=CC=1	C=CC=CC=C, OC(=C)C=CC=C	0	6		
CC=CC=CC, OC1C=CC=CC=1	C=CC=CC=C, OC(=C)C=CC=C	1	6		
CC=CC=CC, OC1C=CC=CC=1	C=CC=CC=C, OC(=C)C=CC=C	2	6		
CC, OC1C=CC=CC=1	C=C, OC(=C)C=CC=C	0	6		
CC, OC1C=CC=CC=1	C=C, OC(=C)C=CC=C	1	6		
CC, OC1C=CC=CC=1	C=C, OC(=C)C=CC=C	2	6		
OP(=O)(O)OP(=O)(O)O, O	O=P(O)(O)O, O=P(O)(O)O	0	4		
OP(=O)(O)OP(=O)(O)O, O	O=P(O)(O)O, O=P(O)(O)O	1	4		
OP(=O)(O)OP(=O)(O)O, O	O=P(O)(O)O, O=P(O)(O)O	0	6		
OP(=O)(O)OP(=O)(O)O, O	O=P(O)(O)O, O=P(O)(O)O	1	6		
C#N, C#N	N=CC#N	0	4		
C#N, C#N	N=CC#N	1	4		

Rules according to k

When one finds a rule, one would be able to apply it on any set of molecules where the left side of the rule appears. When k is small the rule is very general and only the atoms which are mapped to the atoms in the cycle need to appear in the molecule. As k increases one is less able to apply the rule. As an example take a symmetric educt, where the symmetry is broken on one side in the product. If the mono-cyclic transition could be applied to both sides, one would be able to apply the rule twice with a low k. but if k was big enough to make the rule have some of the transitioned parts in the context, one might not be able to apply it again.

Possible improvements

Right now the program does not do a good job with pruning duplicate mappings, and thus creates the same rule many times. One way to deal with this issue would be to use canonicalisation. By the use of canonicalisation we could reduce our educt and product graphs into their canonical form. In the canonical form we would have a better view to compare the two graphs, and thereby be better able to generate a valid mapping.