Synt-On tutorial

Synt-On is a knowledge-base toolkit for the design of organic chemistry libraries, and the analysis of building blocks (BBs). It combines two types of chemicals called "synthons" into a single space. Synthons are fragments with specific properties and labels that indicate where reactive centers are located. Synthons can be obtained by breaking down existing compounds or by transforming reagents. This toolkit helps design libraries, generate similar compounds, perform scaffold analysis, and analyze building blocks in medicinal chemistry[1]. You can find the Synth-On code on GitHub.

Prerequisites

SyntOn is a suite of scripts written in python. It should be used with the following dependencies: python 3.9.0, rdkit 2021.03.1, matplotlib 3.4.2 and numpy 1.20.2.

Several build-in python modules are also used, but they are usually installed by default (datetime, os, time, random, re, resource, sys, multiprocessing, collections, xml). All other modules are custom written and provided within the package.

Installation

The full list of packages is listed in the file SyntOn_environment.yml. You can create required environment using this command, in which you need to replace "user" by your username:

```
 conda env create -f SyntOn_environment.yml  -p
/home/[user]/anaconda3/envs/synton_env
```

Activate the environment using the following command :

```
 conda activate synton_env
```

Example of usage

In this example, we will generate a small library of compounds using the provided list of building blocks. Figure 1 provides an illustration of the compound generation process.

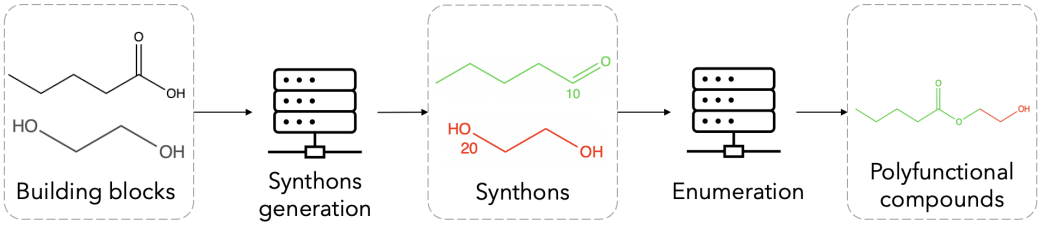

Figure 1. Workflow of compounds enumeration. (20 - Nucleophilic centre, 10 - Electrophilic centre)

Steps for the compound's enumeration:

1. Install the software.
2. Activate your conda environment.
3. Open the terminal and navigate to the "data" folder.
4. Locate the file "BBs.cxsmiles" in the "data" folder. This file contains the list of available building blocks (BBs).
5. To synthonize the building blocks, run the following command:

```
python3 [path_to_script]/SynthOn_BBsBulkClassificationAndSynthonization.py -i BBs.cxsmiles -o outSynth
```

   - Use the "-i" parameter followed by the input file name ("BBs.cxsmiles").
   - Use the "-o" parameter followed by the desired prefix for the output file names.
6. After running the command, four files will be generated:
   - outSynth_BBmode.smi: Contains SMILES, IDs, assigned classes, and generated synthons for each classified building block.
   - outSynth_Synthmode.smi: Similar to the previous file, but focusing on unique synthons.
   - outSynth_NotClassified: Contains building blocks not classified by RdKit.
   - outSynth_NotProcessed: Contains building blocks not processed by RdKit.
7. To enumerate the compounds from the defined library of synthons, run the following command:

```
python3 [path_to_script]/SyntOn_BulkFragmentationEnumerationAndAnaloguesDesign.py -i outSynth_Synthmode.smi -oD [your_working_directory]/data --enumerationMode --nCores 16
```

   - Use the "-i" parameter followed by the input synthons file ("outSynth_Synthmode.smi").
   - Use the "-oD" parameter followed by the output directory ("[your_working_directory]/data").
   - Use the "--enumerationMode" parameter to create all chemically possible combinations from the input synthons.
   - To accelerate the process, use the "--nCores" parameter followed by the number of available CPUs for parallelization. Approximately, calculations should be completed within 1-2 minutes using 16 cores.

After executing the command, temporary files will be generated and combined into a final file called "FinalOut_allEnumeratedCompounds_DuplicatesCanBePresent.smi". It is important to note that this file may contain duplicate compounds. After removing the duplicates, a total of 155 compounds were enumerated from the initial 10 building blocks.

References

(1) Zabolotna, Y.; Volochnyuk, D. M.; Ryabukhin, S. V.; Gavrylenko, K.; Horvath, D.; Klimchuk, O.; Oksiuta, O.; Marcou, G.; Varnek, A. SynthI: A New Open-Source Tool for Synthon-Based Library Design. *J. Chem. Inf. Model.***2022**, *62* (9), 2151–2163. https://doi.org/10.1021/acs.jcim.1c00754.