FTrees Command Line Documentation Version 6.13

Sascha Jung & Marcus Gastreich

August 19, 2024

Contents

1 Introduction

All links, references, table of contents lines etc. in this document are clickable.

Please note that this package is a command line package.

FTrees is a command line package for very fast, fuzzy similarity searching with query molecules. The search can span vast quantities of molecules ("chemical spaces") — or traditional collections of molecules ("libraries"). FTrees enables the user to search up to trillion sized chemical spaces with a fuzzy similarity metric within seconds to minutes on standard, modest hardware. The FTrees technology is also part of our flagship platform infiniSee and operates in the background of the Scaffold Hopper mode (`https://www.biosolveit.de/download/?product=infinisee`). This document is a brief documentation for the command line package.

FTrees lets you

- conduct very fast, fuzzy **pharmacophore similarity searching** across vast combinatorial Chemical Spaces for your query molecules
- perform fuzzy similarity searches in enumerated library files (SMILES, SDF or MOL2)
- visualize the local similarities of mapped substructures between query and result (Figure 3)

The basic principle behind FTrees is that molecules are represented by trees called "Feature Trees" (Figure 1). Every tree is composed of nodes (colored circles in Figure 1) that represent a particular molecular substructure, and vertices that describe their connectivity. The following attributes are encompassed within a node:

- Spatial volume
- Ring membership
- Pharmacophore-like properties (donor, acceptor, amide-like, aromatic, hydrophobic/hydrophilic)

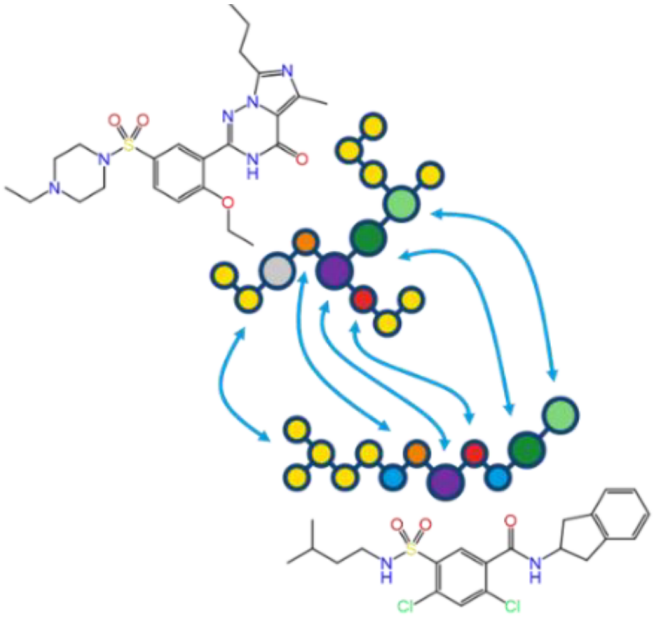

Figure 1: *Example of a reaction and associated fragments encoded with "Lego-like" linkers.*

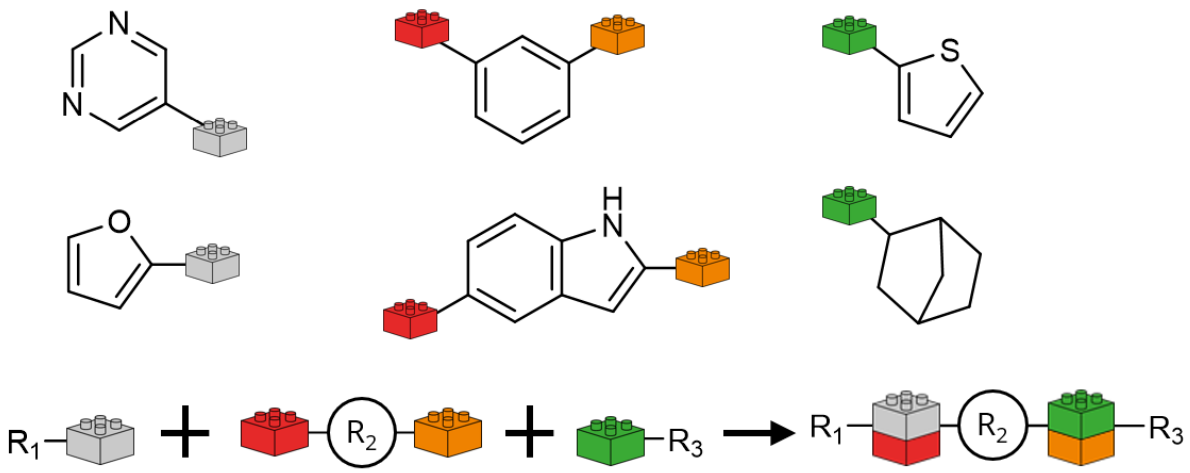

Figure 2: *Example of a reaction and associated fragments encoded with "Lego-like" linkers.*

Two trees are then aligned with each other, similar to a sequence alignment. The alignment provides a mapping of corresponding nodes on either tree (see arrows in Figure 1). Mapped nodes are compared based on their property profiles resulting in a "Local Similarity". The "Global Similarity" is essentially an overall average, it can be used to categorize multiple molecules as to how similar they are, where 0 is dissimilar, and 1 is identical. The coloring of the substructures helps to identify which substructure of the query is matched onto which one of the hit molecule (see Figure 3). The alignment (or mapping) found by FTrees is among all the one with the highest "Global Similarity".

Depending on whether the search is carried out on a standard library or in a chemical space, the algorithm differs slightly:

- Standard molecule libraries: The query and the hit Feature Trees are aligned using a dynamic programming algorithm (see Further Reading at the end). The nodes of the trees, or fragments they represent, are aligned in such a way that the overall similarity of the trees, or molecules, is maximized. Both the overall similarity and the so-called local similarities can be output. Additionally, this alignment can be visualized together with the local similarities and is therefore of great benefit to the user: It shows *why* the computer considers parts of the two molecules to be similar or dissimilar.
- Chemical space navigation: The query Feature Tree is split up into fragments, and — obeying the original connectivity in the query — fragments are searched from the chemical space that have a similarity as close as possible to the target similarity (see further below). The next fragment is added in accordance with the connection/reaction rules encoded in the chemical space to optimize the similarity score. Overall, a new molecule is constructed from the chemical space that overlaps as well as possible with the query molecule (or rather, its Feature Tree), and minimizes the difference between the target similarity and actual similarity (default: identity).

FTrees traverses huge Chemical Spaces using "Lego-like" chemical reaction combinatorics behind the scenes. To conduct quick calculations, we formalize reactions and encode them as linking reactions with associated fragments (see Figure 2). The fragments carry specific linkers, represented by lego bricks in Figure 2. The fragments can only be combined in a chemically meaningful and synthetically accessible way, which is determined by the reaction definitions. For example, fragments with a grey brick can only be attached to fragments with a red brick, those carrying a green brick only to those with an orange brick. Chemical spaces ready to be used with FTrees can be downloaded from our website (`https://www.biosolveit.de/chemical-spaces/`). Alternatively, you can generate your own corporate spaces with the CoLibri toolkit (`https://www.biosolveit.de/products/#CoLibri`).

Please note that FTrees is agnostic with regards to stereo chemistry and *ortho*/*meta*/*para*-substitution

patterns, due to the fuzziness of the descriptor. A hit containing an R stereo center could just as well be an S isomer; cis could just as well be trans!

The overall Feature Tree similarity is normalized from 0 (entirely different) to 1 (identity). As it is derived from Feature Tree nodes that encompass fuzzy, pharmacophore-like properties, it is also referred to as pharmacophore similarity.

Summarizing, with Ftrees you can conduct fuzzy, pharmacophore-based similarity searches in trillion-sized chemical spaces and beyond on modest, standard hardware. Due to the fuzzy approach of the descriptor, you will find hits with relevant chemistry (scaffold hops) that may be missed by other methods (e.g. fingerprint-based similarity methods).[1]

2 Technical Prerequisites

FTrees is a command line application. It needs the following to run:

- The FTrees package
  (`https://www.biosolveit.de/download/?product=ftrees`)
  Depending on your operating system, some libraries may have to be installed (get in touch with us: `mailto:support@biosolveit.com`; and please mention any errors/warnings that you see in your mail)
- A shell (Linux/Unix) or a terminal (macOS), or a command line environment (Windows; e.g.: cmd.exe)
- A valid license (from `mailto:license@biosolveit.com`)

The license setup instructions will come with the license that we send out or have already sent out to you. A "test license" that you can request online and that is sent to you instantaneously can simply be placed next to the executable (ftrees.exe, ftrees, or FTrees — depending on your operating system).

macOS Specialties On macOS, the executable will typically reside inside the *.app package:

```
/Applications/FTrees.app/Contents/MacOS/FTrees
```

To place the short term test license there, you will have to go into the *.app package using a right mouse click on FTrees.app in the Finder, and click on "Show package contents". In there, you will see the Contents/ subfolder, in there the MacOS subfolder, and in there, the FTrees executable. If you are about to use the **test license**, place it right there, next to the executable. A longer term license will be handled separately, we will tell you how when we send that very license.

When you call FTrees for the first time, go to the Finder, and navigate to the Applications folder. Do a right(!) click on FTrees.app, and — if applicable — confirm that you want to open the program. It will flash up once, and you are good to go at the terminal prompt from there on.

To make the first step, call the FTrees within your shell/terminal/environment.

[1] Please have a look at SpaceLight for similarity searches with "classical" fingerprints, e.g. ECFP4, in Chemical Spaces.
`https://www.biosolveit.de/download/?product=spacelight`

3 Jump Start: Finding Scaffold Hops in Large Chemical Spaces

Your license is all set? You unpacked the installation archive? You downloaded a chemical space file (`.space` file, from `https://www.biosolveit.de/chemical-spaces/`)? Then here is a typical call to search a query against a chemical space:

```
./ftrees -i <path/to/query.sdf> -s <path/to/chemical_space.space>
```

The query can be an SD file (`.sdf`), a SMILES file (`.smi` or `.smiles`, containing line-separated SMILES), a `.mol` or `.mol2` file with one or multiple entries. For quick searches, the input can just as well be a SMILES string enclosed by quotation marks, for example:

```
./ftrees -i "CC(C)C(=O)N" -s <path/to/chemical_space.space>
```

By default, the 100 most similar hit molecules are written as SMILES to your console/shell (STDOUT). To write your results to an output file (`.csv` or `.sdf`), additionally use either the `-o / --output-files` option (writes a separate output file for every query) or the `-O / --single-output-files` option (writes a single output file with the concatenated results for all queries), for example:

```
./ftrees -i "CC(C)C(=O)N" -s <path/to/chemical_space.space> -o <path/to/output.csv>
```

The output file contains the structure of the result molecules as well as detailed information on the similarity score (see page 8 for more information). You can adjust the number of results by using the `--max-nof-solutions` option:

```
./ftrees -i <path/to/queries.sdf> -s <path/to/chemical_space.space>
         -o <path/to/output.csv>  --max-nof-solutions 300
```

Additionally, you can limit the output to those results that exceed a certain similarity (value between 0 and 1):

```
./ftrees -i "CC(C)C(=O)N" -s <path/to/chemical_space.space> -o <path/to/output.csv>
         --min-similarity-threshold 0.8
```

4 Command Line Options

4.1 Overview

For an overview of all command line options, call FTrees with `--help`:

```
./ftrees --help

 Program options:
-i [ --input ] arg                    Input query molecule file or single input molecule as smiles.
                                      Supported file types are *.smi, *.smiles, *.mol, *.mol2 and *.sdf.
-s [ --search-files ] arg             Paths to library input molecule files for similarity scoring or to
                                      Fragment Space FSF files or Fragment Spaces. Supported file types are
                                      *.smi, *.smiles, *.mol, *.mol2, *.sdf, *.fsf, *.space and *.zip.
                                      Note: The .flf and fragment files specified in the FSF have to be in
                                      the appropriate relative paths.
-o [ --output-files ] arg             Output base files (suffixes are required). Supported file types are
                                      *.csv and *.sdf.
-O [ --single-output-files ] arg      Output files (suffixes are required). All results are written to a
                                      single output file. Supported file types are *.csv and *.sdf.
-m [ --match-image-base-file ] arg    Output base file name for matching images (suffix required).
                                      Supported file types are *.pdf, *.png and *.svg.
                                      Note: For each match a separate file is created.
--gen-2d-output [=arg(=1)]            Generates 2d coordinates in case of SDF output files.
--gen-mapping-output [=arg(=1)]       Generates detailed Feature Tree similarity descriptors and annotates
                                      them in the output file

Configuration:
--expand-alternative-results [=arg(=1)]
                                      Write alternative results based on alternative reaction paths or
                                      duplicate matchings.
--max-nof-results arg (=100)          Maximum number of top-ranking result molecules [1 to 1000000].
--min-similarity-threshold arg (=0.8) Similarity threshold below which molecules are discarded [0.0 to
                                      1.0].
--target-similarity arg (=1)          Desired target similarity to the query molecule [0.5 to 1.0].
                                      Note: Must be >= '--min-similarity-threshold'
--total-diversity arg (=1)            Required diversity between any two compounds in a solution set [0.9
                                      to 1.0].
                                      Note: Only available if '--max-nof-results' is <= 500.
                                      WARNING: any value below 1.0 drastically extends the run time.

General options:
-h [ --help ]                         Print this help message
--license-info                        Print license info
--thread-count arg                    Maximum number of threads used for calculations. The default is to
                                      use all available cores.
--version                             Print version info
-v [ --verbosity ] arg (=2)           Set verbosity level
                                           0 [silent]
                                           1 [error]
                                           2 [warning]
                                           3 [workflow]
                                           4 [steps]
```

The abbreviated, one-letter options are preceded with one dash `-` whereas the longer, named options are preceded with two dashes: `--`. If an option needs an argument (arg), you can include or omit the equals sign.

4.2 Minimal Required Options

This section describes the arguments you must specify at minimum to successfully run a similarity search. First, you must provide the path to a file containing the query compounds. All well-known data formats are supported (MOL, SDF, SMILES file, MOL2). Instead of a file containing the query molecules you can also specify a single SMILES string enclosed by quotation marks. The SMILES string or the molecule file must be passed to the `-i` option. Additionally, you need to specify the path to a fragment space (`.space` file) to be searched. Alternatively, you can also specify a library file (SDF, MOL2, SMILES file) to perform an enumerated search instead of a space search. Either the space file or library file have to be specified with the `-s` option. The minimal search prompt then has the following general form:

```
<path/to/ftrees/executable> -i <path/to/queries> -s <path/to/space_file>
```

When you specify the required paths the search prompt might look like the following:

```
./ftrees -i my_queries.sdf -s my_space.space
```

Or, if you specified a SMILES string instead of a file:

```
./ftrees -i "CC(C)C(=O)N" -s my_space.space
```

In the examples above, the output is printed on the console by default, which will look similar to the following example:

```
Query:   O(C(C)C)CCC
Rank:   1   sim: 0.933   O=C(N(C(C)C)C)C   WXVL021____AN0073____SB0150
Rank:   2   sim: 0.933   O=C(N(C)C)C(C)C   WXVL021____AN0032____SB0052
Rank:   3   sim: 0.933   O=C(N(C)C)CC(C)C   WXVL021____AN0032____SB0063
Rank:   4   sim: 0.932   O=C(N(C)C)CCC   WXVL021____AN0032____SB0073
Rank:   5   sim: 0.928   ClCCCCC(=O)N(C)C   WXVL021____AN0032____SB0379
...
Rank: 100   sim: 0.899   O=C(N(C)C)C(O)CC(C)C   WXVL021____AN0032____SB4281
```

For every query molecule, the respective hit molecules are printed as SMILES to the console with additional information on the name of the molecule and similarity score. By default, 100 result molecules per query are printed to the console. If you want to increase or decrease that number, please have a look at the `--max-nof-results` option (see page 11).

Of course, you can also write the result molecules to an output file either with the `-o` option (one separate output file per query) or with the `-O` option (a single output file with results molecules for all queries). See page 8 for more information.

4.3 Program Options

–i [––input] arg Specify a file containing the query molecules. Supported file formats are SDF, MOL and MOL2. You can also provide a text file containing multiple line-separated SMILES (file extension must be `.smi` or `.smiles`). Instead of specifying a filename, you can also enter a SMILES string directly. The string must be enclosed by quotation marks. It is also possible to use the `-i` option multiple times in a row (see examples below).
NOTE: The `-i` option is required.

Examples:

```
ftrees -i myquery.sdf

ftrees -i myquery.sdf -i mydrugs.smi

ftrees -i "CC1=CC=CN=C1"
```

–s [––search-files] arg Specify the chemical space file or library file to be searched. Also multiple files can be specified at once or the `-s` option can be used several times in a row. And even libraries and chemical spaces can be mixed (see examples below). Supported file types for libraries are SDF, SMILES (`.smi` and `.smiles` containing line-separated SMILES) and MOL2. For chemical spaces, `.space` and `.fsf` can be used. Please note: If you specify a `.fsf` file you have to make sure that the corresponding `.flf` files and fragment files (`.smi`) are in the appropriate relative paths.
NOTE: The `-s` option is required.

Examples:

```
ftrees -s mylibrary.sdf

ftrees -s myFIRSTlib.sdf mySECONDlib.sdf

ftrees -s myLIBRARY.sdf mySPACE.space

ftrees -s mylibrary.smi

ftrees -s mychemicalspace.space

ftrees -s mychemicalspace1.space  -s mychemicalspace2.space

ftrees -s mychemicalspace.fsf
```

–o [––output-files] arg Specify the base name for the output files as argument. The output will be written either as SD file (`.sdf`) or `.csv` file — or both. Specify the desired output file type by the file extension. The output SD file contains additional information for every result molecule in dedicated SD data fields (see below). The output CSV file contains the result molecules as SMILES together with additional information (see below).
NOTE: If you have multiple queries in your input file, then a separate CSV or SD file will be written per query! To write all results from multi-query input files in a single output file, see `--single-output-files` option below.
NOTE: If you do not specify an output file, reduced output will be written to the command line (STDOUT, see Section 4.2).

Examples:

```
ftrees -o myoutput.sdf
```

```
ftrees -o myoutputtable.csv
```

```
ftrees -o myoutput.sdf myoutputtable.csv
```

The latter example outputs both, one SD and one CSV file per query contained in your input file. The names of the output files will have the following general structure:

`myoutput_{querynumber}.sdf` and `myoutputtable_{querynumber}.csv`

The output file(s) contain additional information for every result molecule:

- **result rank**: rank among all results for this particular query
- **pharmacophore similarity**: FeatureTree similarity value (derived from aligned query molecule FeatureTrees against result molecule FeatureTrees)
- **result name**: name of the result molecule
- **query name**: name of the query molecule
- **query smiles**: SMILES of the query molecule
- **space**: name of the searched space(s) or library
- **reaction name**: name of the reaction that constructs the result molecule
- **reagent(1-5) name**: name of building block (1-5) from which the result molecule is constructed (relevant only for space searches, number depends on reaction)
- **reagent(1-5) smiles**: SMILES of building block (1-5) from which the result molecule is constructed (relevant only for space searches, number depends on reaction)

–O [––single-output-files] arg Specify the name for the output files as argument. The output will be written either as SD file (`.sdf`) or `.csv` file — or both. Specify the desired output file type by the file extension. The SD file will contain additional information for every result molecule in dedicated SD data fields (see below). The CSV file will contain the result molecules as SMILES together with additional information (see below).
As a difference to the `--output-files` option (see above), all results from multi-query input files will be written to a single output file (concatenated results). It is also possible to write both a CSV file and a SD file at the same time (see last example below).
NOTE: If you do not specify an output file, reduced output will be written to the command line (STDOUT, see Section 4.2).

Examples:

```
ftrees -O singleoutput.sdf
```

```
ftrees -O singleoutput.csv
```

```
ftrees -O singleoutput.sdf singleoutput.csv
```

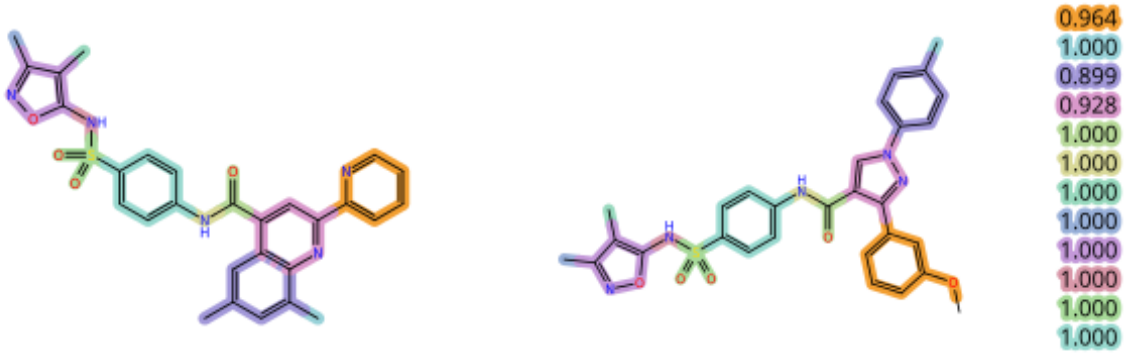

Figure 3: *Example of a match image. The query is on the left side, the hit molecule on the right side.*

–m [––match-image-base-file] arg Specify a base name for the match image output files as argument. This will generate (one per hit molecule, so potentially many!) output images that explain the matching of query versus hit molecule in 2D pictures. Depending on the file extension you specified, images will be written as `.png`, `.pdf`, or vector-based `.svg` files.
NOTE: Generation of match images leads to extended runtimes.

Example:

```
ftrees -m matching.png
```

This call will generate *one* `.png` file per hit molecule(!), the file names will look like:

`matching_{querynumber}_{hitnumber}.png`.

Figure 3 shows an example matching image. The local similarities are listed and color-coded on the right hand side. The query molecule is located on the left side (Tofacinib), the hit molecule on the right side. You can now easily understand why FTrees considers these molecules to be similar: the parts of the aromatic heterocycle are fully identical between query and hit molecule (orange and cyan parts with similarity 1), the aliphatic ringsystems both contain a nitrogen atom and are located at the same place (green part, similarity 0.994) and the carbonyl group (pink, similarity 0.998) is also present in both molecules at a comparable position.

––gen-2d-output If you specify this option, 2D coordinates are generated for your output SD files. Make sure to specify an SD file either using the `--output-files` or `--single-output-files` option.
NOTE: Generation of 2D coordinates will lead to extended runtimes, especially for large numbers of queries and hit molecules.

Example:

```
ftrees --gen-2d-output
```

--gen-mapping-output If you specify this option, for every hit molecule the local similarity values (annotated in the *similarity-descriptor* column/tag) and the corresponding substructures (annotated as semicolon-separated SMILES in the *similarity-descriptor-smiles* column/tag) are written to the output files. This is the same information that is visualized for the hit molecule in a match image (see Figure 3 and the description for the `--match-image-base-file` option above).

Example:

```
ftrees --gen-mapping-output
```

4.4 Configuration

--expand-alternative-results Expands alternative reactions for a hit molecule. In chemical spaces, the same molecule can be formed in different reactions with different reagents/building blocks. If you use this option, these different possibilities are written to the output. Alternative results all have the same rank and the same similarity score but have different names and different reagents. You can also use this option to find identical results in different spaces if you specify multiple spaces at the same time (see `-s` option).

Example:

```
ftrees --expand-alternative-results
```

--max-nof-results arg(=100) Takes a number between 1 and 1,000,000 as argument. This option controls the number of hit molecules per query that will be output. The default is 100, that means 100 hit molecules for every query molecule are written to the output files. The results will always be sorted by descending Feature Tree similarity, so the parameter controls the TOP number of results.

NOTE: An overall maximum of one million results can be written out. This means, for example, if you have an input file containing 1000 query molecules, the value for `--max-nof-results` cannot exceed 1000 !!

Example:

```
ftrees --max-nof-results 1000
```

--min-similarity-threshold arg(=0.8) Takes a number between 0 and 1 as argument. Default is 0.8. This option lets you limit the results to those exceeding a minimum similarity. For example, if you are only interested in highly similar molecules, then you may constrain the similarity to be above 0.95.
NOTE: Feature Trees are "late talkers", i.e., a similarity of 0.3 or 0.5 does NOT mean that half of the molecule is similar. Instead, since the similarity is fuzzy, a tangible similarity starts around 0.8; chemists will agree on a more "obvious" similarity in ranges around 0.85 and higher.

Example:

```
ftrees --min-similarity-threshold 0.95
```

––target-similarity arg(=1) Takes a number between 0.5 and 1 as argument. Default is 1. If you would like your results to stay comparably close to the query molecule you should specify a high target similarity (say 0.95 or above), whereas if you would like to enforce pronounced scaffold hops you can lower the target similarity (say 0.75-0.9). Those results that are closest to the target similarity will be the top hits in your output files.

NOTE: `--target-similarity` must be >= `--min-similarity-threshold`

Example:

```
ftrees --target-similarity 0.9
```

––total-diversity arg(=1) Takes a number between 0.9 and 1 as argument. Default is 1. This option enforces diversity between all hit molecules. The FTrees similarity computed between any two molecules of the result set is not allowed to be greater than the specified value for the total diversity. Consequently, the smaller the value of total diversity, the more diverse the result molecules in the output file will be.
NOTE: If you set this option to a value below 1, pairwise similarities must be computed between all molecules in result set. Therefore, our recommendation is to use it only with small result sets.
NOTE: Only available if `--max-nof-results` is <= 500.
WARNING: Any value below 1.0 will drastically increase the runtime.

Example:

```
ftrees --total-diversity 0.9
```

4.5 General Options

–h [––help] Displays the command line help with short descriptions for every argument option. For more information see Section 4.1.

Example:

```
ftrees --help
```

––license-info Shows command line information about the license setup you currently use. If you have any problems with your license, send an email to `mailto:support@biosolveit.com` and include this information.

Example:

```
ftrees --license-info
```

––thread-count arg Takes a number as argument. Specifies the maximum number of threads used for your similarity searches. By default, all available logical cores of your computer are used. You may want to reduce the number of threads if you want to run other computations on your computer at the same time, or if you share the compute resource.

Example:

```
ftrees --thread-count 4
```

––version Displays information on the version of FTrees on the command line. In quoting FTrees, please mention this version number.

Example:

```
ftrees --version
```

–v [––verbosity] arg(=2) Sets the verbosity level, e.g., the level of console output, with an integer argument. The default value is 2. The following options are available:

0 Silent. No messages will be displayed in the console during the search run. Errors will be ignored whenever possible.

1 Error. Only error messages will be displayed.

2 Warning. The default setting, warnings and error messages will be displayed.

3 Workflow. In addition to errors and warnings, information on the different steps of the search are displayed on the command line.

4 Steps. In addition to the 'Workflow' option, the progress of each step is displayed in detail.

Example:

```
ftrees -v 0
```

5 Further Reading

- The basic ideas behind the FTrees method are covered in the original publication by Matthias Rarey and J. Scott Dixon
  (`https://doi.org/10.1023/A:1008068904628`)
- The dynamic match search is described by Marc Zimmermann et al., 2003, in: 14th European Symposium on Quantitative Structure-Activity Relationships, (Van de Waterbeemd, H. ed.), Blackwell Publishing.
- Chemical space navigation with Feature Trees descriptors in the FTrees program has been published by Matthias Rarey and Martin Stahl
  (`https://doi.org/10.1023/A:1011144622059`)
- A review on chemical space searching has been published by Torsten Hoffmann and Marcus Gastreich in DDT 2019
  (`https://doi.org/10.1016/j.drudis.2019.02.013`).
- CoLibri is our software to generate chemical spaces from your own in-house synthons and reactions.
  `https://www.biosolveit.de/download/?product=colibri` has more information.

If you prefer to control the FTrees algorithm through a graphical interface, download our platform infiniSee (`https://www.biosolveit.de/download/?product=infinisee`). infiniSee uses the FTrees algorithm in the background of the Scaffold Hopper mode.

We wish you great success and much joy with FTrees!