ISIDA/FRAGMENTOR 2017 USER GUIDE

LABORATOIRE DE CHEMOINFORMATIQUE, UMR7140 - STRASBOURG

F. RUGGIU, G. MARCOU, V. SOLOV'EV, D. HORVATH, A. VARNEK

UPDATED NOVEMBER 2017

TABLE OF CONTENTS

INTRODUCTION

Introduction

The ISIDA Fragmentor2017 is a development of the Laboratoire de Chémoinformatique, Chimie de la Matière Complexe (SMS UMR 7140), Université de Strasbourg, France. This program is a part of the ISIDA project, which stands for “In SIlico Design and data Analysis” and aims to develop tools for the calculation of descriptors, the navigation in chemical space, quantitative structure-activity modeling (QSAR) and virtual screening. The ISIDA Fragmentor2017 calculates molecular fragment count descriptors from a Structure-Data File (SDF). It is based on a series of graph algorithm from the book “Algorithmes de graphes” [1].

The ISIDA descriptors have been described in 6 publications:

- ISIDA Substructural Molecular Fragments (SMF)[2, 3]
- ISIDA Fuzzy Pharmacophoric Triplets (FPT) [4, 5]
- ISIDA Property-Labelled Fragments (IPLF) [6]
- Individual hydrogen-bond strength QSPR modelling with ISIDA local descriptors: a step towards polyfunctional molecules [7]

ISIDA Fragmentor2017 is able to calculate SMF and IPLF (with a ChemAxon based java program: CA_Prop_Map2011) as described in the publications. You may also read our Nomenclature document to learn about ISIDA fragment descriptors which is available on our website (http://infochim.u-strasbg.fr/recherche/Download/Fragmentor/Nomenclature_of_ISIDA_fragments_2015.pdf).

The laboratory uses the ChemAxon plugins to map a property on the graph. However, one of the aims of ISIDA Fragmentor2017 is to enable the use of any combination of options and let the user as much freedom as possible to fit his needs. Therefore, the ”coloration“ of the molecular graph can be user-defined - given the Input format is respected.

The next Chapter describes the installation of the Fragmentor2017. Then, we will list all the options, input and output format description, installation and usage of Fragmentor2017 and the corresponding nomenclature of ISIDA fragment descriptors. Chapter 3 is dedicated to our ChemAxon-based property mapping program.

Installation

DOWNLOAD PACKAGES

A few compiled executables are available on request on our website in the Download section (http://infochim.u-strasbg.fr/spip.php?rubrique41). If you need another compiled version or wish to have access to the source code, please read the following instructions.

The package is stored on the SVN repository of the web server of the Laboratory of Chemoinformatics. The prerequisite to obtain the package are (i) an account on the server associated to the users group and (ii) an SVN client.

An account can be requested to Dr. G. Marcou (g.marcou@unistra.fr) or Dr. F. Bonachera (f.bonachera@unistra.fr).

The SVN client is usually natively available on Linux systems. If it is absent, the packaging system frequently names it at subversion. For Max OS X users, the SVN client is provided as a part of xCode later than version 1.7. For Windows, the best option is to use a graphical client. One of the most popular is tortoiseSVN (https://tortoisesvn.net/). For Windows users, be aware that SVN clients usually are invading: they provide additional menus and sets of icons to the whole Windows environment.

Once an SVN client is installed, you can recover the package using the following URL:

```
svn+ssh://<login>@infochim.u-strasbg.fr/home/infochimie/svn/Fragmentor2017
```

using the command line client the full command line is the following:

```
svn checkout svn+ssh://<login>@infochim.u-
strasbg.fr/home/infochimie/svn/Fragmentor2017
```

This will create a new Fragmentor2017 directory. The directory can later be updated to obtain new versions of the package. For the command line client, the update can be done by typing, inside this directory:

```
svn update
```

In the same directory as your Fragmentor2017 directory, acquire the Molecule project, using svn:

```
cd Fragmentor2017
svn checkout svn+ssh://<login>@infochim.u-
strasbg.fr/home/infochimie/svn/Molecule
```

INSTALLATION

CONTENTS

The ISIDA/Fragmentor2017 package contains the following packages, aside from the Free Pascal sources of the actual software:

- Molecule: the package containing all Free Pascal Units for molecules manipulation
- example: an example folder providing several files for tests.
- doc: the documentation
- lnx64 : a folder containing executables for linux 64
- mac: a folder containing executables for mac
- win64: a folder containing executables for windows 64

COMPILATION

The compiled versions of the software, which can be found in the directories lnx64, mac and win64 (Fragmentor for command line and xFragmentor for graphical version) do not require any installation procedure: they can be used immediately and do not require any pre-requisites.

If you need to compile the project, you will need Lazarus 1.6 and FPC 3.0. (https://www.lazarus-ide.org/)

In the IDE Lazarus, open the main project file of the tool you need to compile. The main file of the project is the lpi (Lazarus Project Information) file. For instance to recompile the command line version of the software, you need to open project file (Fragmentor.lpi). Then, compile the software from the tool bar of Lazarus, using the operation Execute/Compile.

There are 2 versions of the software that can be compiled:

- The command line version : open the Fragmentor.lpi project in lazarus
- The graphical version : open the xFragmentor.lpi project in lazarus

You can also compile the Fragmentor using only fpc, with the following options:

```
-MObjFPC -Scgi -O3 -g -gl -vewnhi -l -FuMolecule -Fu
```

Fragmentor2017 – command line

COMMAND LINE

This section describes the command line version of the software. It can be summarized as follows:

```
PATH/Fragmentor -i <SDFile> -o <BaseName>
               [-f <string> -s <string> -h <HeaderFile>]
                -t <integer> [{-l <integer> -u <integer>}
                -c <SDField> -b <SDField> -m <(0,1,2,3)> -d <(0,1,2)>
                -x <(0,1,2,3)>:<XMLFileName>
                --DoAllWays --AtomPairs --UseFormalCharge --StrictFrg
                --GetAtomFragment --Pipe]
```

Options in squared brackets are not mandatory and those in curly brackets are linked to one another. Options are quickly explained in the next section. It is best to keep the options as they are ordered above. In any case, longer options (with --) should always follow the short ones (with only -).

One call to ISIDA Fragmentor2017 may include several different types of fragmentation.

To do so, use several -t options (indicating the type of fragmentation) with the list of corresponding options. For example if you wish to obtain sequences of atoms and bonds ranging from 1 to 4 bonds, and augmented atoms with a distance up to 1 bond, you will use the following command:

```
PATH/Fragmentor -i input.sdf -o output -t 3 -l 2 -u 5 -t 10 -l 2 -u 2
```

Note: The numbers given as lower and upper lengths correspond to the number of atoms included in the sequences. If you wish to include atom counting to the previous command then use:

```
PATH/Fragmentor -i input.sdf -o output -t 3 -l 2 -u 5 -t 10 -l 2 -u 2 -t 0
```

Certain options cannot be used together or require another option:

- Atom-centered fragments (-t 4 to 9) are always shortest path - they cannot be used with the option --DoAllWays.
- --StrictFrg can only be used with the -h option to indicate the header file (.hdr). The outputted svm will be limited to the descriptors indicated in that header file and keeping the same order.
- Marked Atom option (-m) can only be set to 0 or 1 for Triplets calculation (-t 10).

Make sure your input Structure-Data File (SDF) is at the V2000 format, or else it might generate errors, memory leaks or wrong fragmentations. Be aware that ISIDA Fragmentor2017 does not check the input file before treating it!!

FRAGMENTOR2017 – COMMAND LINE

LIST OF OPTIONS

- -i : Input Structure-Data File (SDF) name.
- -o : All output files will have this name and will differ only by their extensions.
- -f : Format of the output. By default SVM - SMF, SVM and CSV are available (see output formats in the *Input and Output formats* section)
- -s : A substring identifying unambiguously a field name in the SDF. The value of the field will be considered as a property to be saved along with set of descriptors of each input compound. Missing values are replaced by "?".
- -h : Name of a header file. If present, the fragmentation will reproduce the list of fragments the header contains. The output header file will match this input concatenated with new fragments discovered at the end.
- -t : Fragmentation type. See below.
- -l : Minimal length of fragments as sequences - Note: a length of 2 corresponds to a sequence with 2 atoms
- -u : Maximal length of fragments as sequences
- -c : Indicate the field name (ATOM_COLOR_NAME) in the SDF of your wished coloration for atoms. Should be of format:

```
> <ATOM_COLOR_NAME>
5 1:P 2:H 4: A/D
95 1:A 2:H 4:D
```

where 5 and 95 are the count to be considered for each species and the following characters are Atom number: Coloration1/Coloration2

- - b : Indicate the field name (BOND_COLOR_NAME) in the SDF of your wished coloration for bonds. Should be of format:

```
> <BOND_COLOR_NAME>
5 2:-up- 4: -down-
? 2:=Z=
```

the first column (containing "5" and "?") is ignored. The following characters are Bond number: Coloration.

- -m :
  - If set to 1 : All sequences must begin or end by a marked atom. A marked atom is an atom that has a label in the 7th column of the atom block in the SDF file.
  - If set to 2 : All fragments containing the marked atom will be generated
  - If set to 3 : A special flag (&MA&) will be added to the marked atom. All fragments are present.
  - If set to 0 : all molecular fragments will be generated - same as without the option.

Fragments composed of several sequences are included if one of the sequences is valid according to this condition.

- -d :
    - If set to 1: When processing Condensed Graph of Reactions (CGRs), only those fragments containing a dynamic bond are kept while the others are discarded.
    - If set to 2 : When processing Condensed Graph of Reactions, only those fragments containing only dynamic bonds are kept while the others are discarded.
    - If set to 0 : all molecular fragments will be generated - same as without the option.

Fragments composed of several sequences are included if one of the sequences is valid according to this condition.

- -x : This option controls the reading/writing of an XML file describing the setup of a fragmentation scheme. The syntax of this option includes an integer between 0 and 3, a column (":") then a string. The string refers to the name of the XML file that will be read or write. The value 0 (default) means that the XML file processing is ignored. The value 1 will create an XML file containing the current setup as interpreted from the other -t options of the command line. The value 2 will read a previously created XML file and interpret it as additional setups to those already interpreted from other -t options of the command line. The value 3 will setup the fragmentation as for the value 2 then save the resulting setup as in a new XML file as for the value 1.
- --DoAllWays : If fragments are sequences, search for all paths connecting two atoms.
- --UseFormalCharge : Charged atoms (column 5 in the SDF file) will be indicated by adding _FC"charge_value"_
- --AtomPairs : All constitutional details of a sequence are removed and only the number of constitutive atoms is given.
- --StrictFrg : Only fragments included in a header file defined by a "-h" option are considered. New fragments are discarded.
- --GetAtomFragment : outputs also for each atom the list of fragments in which it is included.
- --Pipe: the output files are appended to existing files with the same name, otherwise the output files are overwritten.

FRAGMENTOR2017 – COMMAND LINE

TYPE OF FRAGMENTATIONS

(-t option)

-t 0 Count of atoms

-t 1 Sequences of atoms only

-t 2 Sequences of bonds only

-t 3 Sequences of atoms and bonds

-t 4 Atom centered fragments based on sequences of atoms

-t 5 Atom centered fragments based on sequences of bonds

-t 6 Atom centered fragments based on sequences of atoms and bonds

-t 7 Atom centered fragments based on sequences of atoms of fixed length

-t 8 Atom centered fragments based on sequences of bonds of fixed length

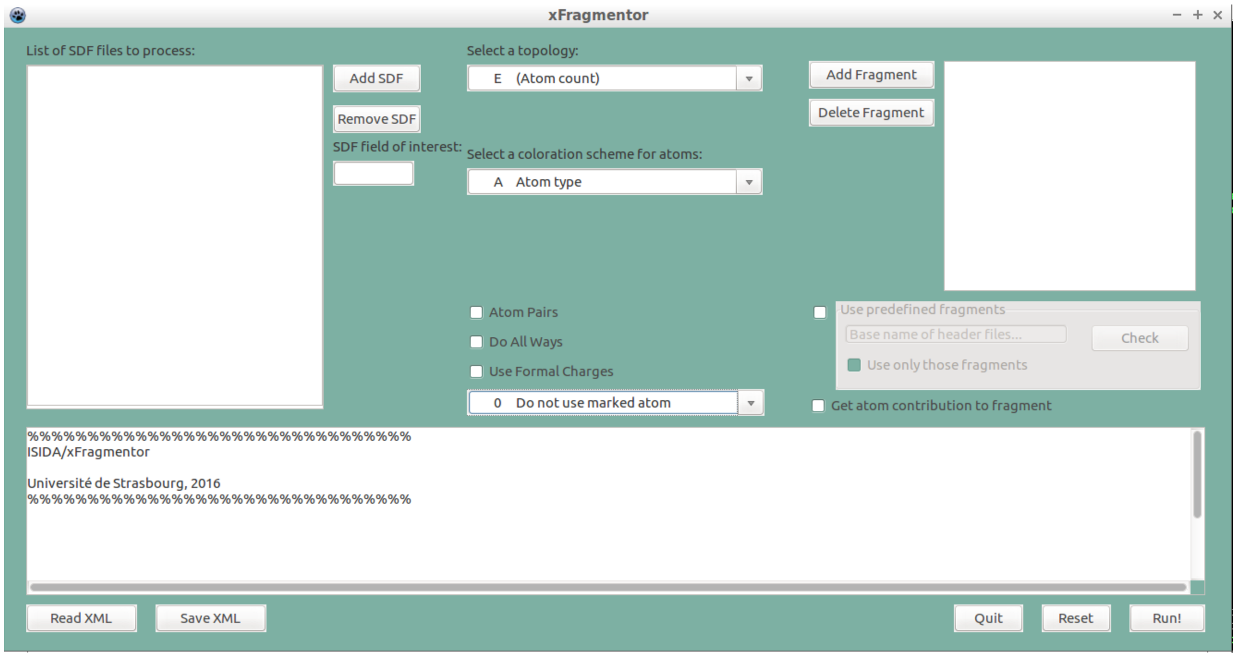
-t 9 Atom centered fragments based on sequences of atoms and bonds of fixed length

-t 10 Triplets

Fragmentor2017 – Graphical user interface

The Fragmentor is also available with a graphical user interface. The software can be found in the folder corresponding to your machine type (currently, 3 versions are available: lnx64, win64 and mac). All graphical user interfaces are named xFragmentor_XXX (XXX=operating system type).

OVERVIEW



Select a topology: E (Atom count)

Select a coloration scheme for atoms: A Atom type

0 Do not use marked atom

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ISIDA/xFragmentor

Université de Strasbourg, 2016
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 1. xFragmentor

OPTIONS

The graphical version of the Fragmentor offers the same options as the command line.

List of SDF files to process

You can input several SDF files, which will be treated sequentially.

SDF field of interest

A substring identifying unambiguously a field name in the SDF. The value of the field will be considered as a property to be saved along with set of descriptors of each input compound. Missing values are replaced by "?".

Topology

E - Atom count (corresponds to -t 0)
I - Atom Path (corresponds to -t 1, 2, 3)
II - Atom-centered fragments, heterogeneous path sizes (corresponds to -t 4, 5, 6)
IIR - Atom-centered fragments, homogeneous path sizes (corresponds to -t 7, 8, 9)
III - Triplets (corresponds to -t 10)
If you select a topology different from Atom path, a selection for minimum length of fragments and maximum length of fragments will appear.

Coloration scheme for atoms

If the option **Custom** is selected, you will be asked to indicate the field name (ATOM_COLOR_NAME) in the SDF of your wished coloration for atoms. Should be of format:

```
> <ATOM_COLOR_NAME>
5 1:P 2:H 4: A/D
95 1:A 2:H 4:D
```

where 5 and 95 are the count to be considered for each species and the following characters are Atom number: Colouration1/Colouration2

Coloration scheme for bonds

If the option Custom is selected, you will be asked to indicate the field name (BOND_COLOR_NAME) in the SDF of your wished coloration for bonds. Should be of format:

```
> <BOND_COLOR_NAME>
5 2:-up- 4: -down-
? 2:=Z=
```

the first column (containing "5" and "?") is ignored. The following characters are Bond number: Coloration

Atom Pairs

All constitutional details of a sequence are removed and only the number of constitutive atoms is given.

Do All Ways

If fragments are sequences, search for all paths connecting two atoms.

Use Formal Charges

Charged atoms (column 5 in the SDF file) will be indicated by adding _FC"charge_value"_

Marked atom dropdown menu

- 0 – Do not use marked atom: all molecular fragments will be generated - same as without the option.
- 1 – Fragments begin or end by marked atom: All sequences must begin or end by a marked atom. A marked atom is an atom that has a label in the 7th column of the atom block in the SDF file.
- 2 – Only fragments containing marked atom: All fragments containing the marked atom will be generated
- 3 – Marked atoms are annotated: A special flag (&MA&) will be added to the marked atom. All fragments are present.

Use predefined fragments

If selected, the fragmentation will reproduce the list of fragments the header contains. The output header file will match this input concatenated with new fragments discovered at the end. You can also choose to use only the list of fragments contained in the header file.

Get atom contribution to fragment

This will allow to keep track of atom contribution for each fragment.

LAUNCHING A FRAGMENTATION

Once all options have been selected and filled, click **Add Fragment**. If you want to launch several fragmentations on the same SDF file(s), you can then modify the fragmentation options and click again on **Add Fragment**. Once each of the fragmentation configurations have been added to the list, click **Run!**.

OUTPUT FILES

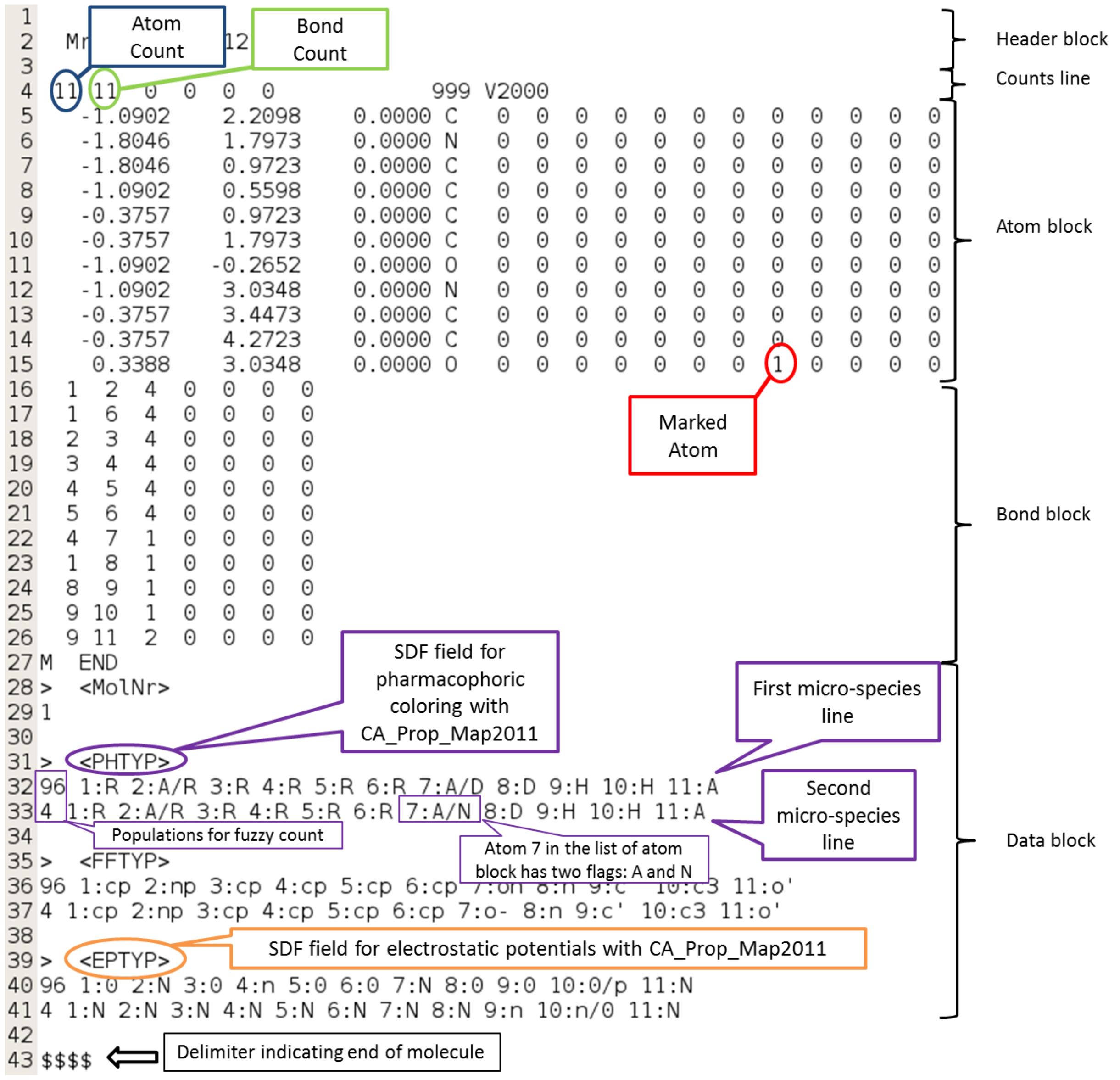
The output files will be located in the same folder in which you have launched the fragmentation. xFragmentor will generate 5 different output files :

- ARFF: Attribute-relation file format, used in WEKA. Follow this link for documentation : https://www.cs.waikato.ac.nz/ml/weka/arff.html
- ATM: A file containing atom contributions to fragments.
- HDR: a header file, containing the index and a string representing each fragment discovered into the SDF
- SVM: descriptor matrix in a file following the libSVM format. The first column contains the values of the field identified using the -s option. Other columns consist in a pair of values separated by a ":". The first value identifies the fragment’s index in the header file; the second one is the fragment count.
- XML: A summary of the fragmentation(s) (options, header file(s), input file(s))

Input and Output formats

INPUT : STRUCTURE DATA FILE (.SDF)

SDF is a format developed by MDL (now part of Accelerys). Its format should be findable on Accelerys' website and a copy of the document is given in the doc folder of the project. The V2000 format is used by ISIDA Fragmentor2017. Here is a quick description of the most important features that an SDF should contain:



```

  Mr             12

 11 11  0  0  0  0          999 V2000
   -1.0902    2.2098    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8046    1.7973    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8046    0.9723    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.0902    0.5598    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.3757    0.9723    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.3757    1.7973    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.0902   -0.2652    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   -1.0902    3.0348    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
   -0.3757    3.4473    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.3757    4.2723    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    0.3388    3.0348    0.0000 O   0  0  0  0  0  0  0  1  0  0  0  0
  1  2  4  0  0  0  0
  1  6  4  0  0  0  0
  2  3  4  0  0  0  0
  3  4  4  0  0  0  0
  4  5  4  0  0  0  0
  5  6  4  0  0  0  0
  4  7  1  0  0  0  0
  1  8  1  0  0  0  0
  8  9  1  0  0  0  0
  9 10  1  0  0  0  0
  9 11  2  0  0  0  0
M  END
>  <MolNr>
1

>  <PHTYP>
96 1:R 2:A/R 3:R 4:R 5:R 6:R 7:A/D 8:D 9:H 10:H 11:A
4 1:R 2:A/R 3:R 4:R 5:R 6:R 7:A/N 8:D 9:H 10:H 11:A

>  <FFTYP>
96 1:cp 2:np 3:cp 4:cp 5:cp 6:cp 7:on 8:n 9:c  10:c3 11:o'
4 1:cp 2:np 3:cp 4:cp 5:cp 6:cp 7:o- 8:n 9:c' 10:c3 11:o'

>  <EPTYP>
96 1:0 2:N 3:0 4:n 5:0 6:0 7:N 8:0 9:0 10:0/p 11:N
4 1:N 2:N 3:N 4:N 5:N 6:N 7:N 8:N 9:n 10:n/0 11:N

$$$$
```

Atom Count
Bond Count
Header block
Counts line
Atom block
Marked Atom
Bond block
SDF field for pharmacophoric coloring with CA_Prop_Map2011
First micro-species line
Second micro-species line
Populations for fuzzy count
Atom 7 in the list of atom block has two flags: A and N
Data block
SDF field for electrostatic potentials with CA_Prop_Map2011
Delimiter indicating end of molecule

Figure 2: Example of SDF

INPUT AND OUTPUT FORMATS

Description of example SDF

- Line 1-3: Header block - contains name of molecule
- Line 4: Counts line - First 3 characters corresponds to the atom count, next 3 is the bond count.
- Line 5- 4 + atom count (15): Atom Block - each line in this block corresponds to an atom and each column corresponds to a different property of the atom. The number of lines corresponds to the atom count read in line 4.
    - Column 1-3: Spatial coordinates x,y,z
    - Column 4: Element
    - Column 6: Formal Charge (1 = +3, 2 = +2, 3 = +1, 4 = doublet radical, 5 = -1, 6 = -2, 7 = -3)
    - Column 12: Not used in MDL format. This column is used by the Fragmentor to indicate marked atoms. To mark an atom, the 0 should be replaced by a 1. Like in the last atom of the atom block in 2.1 (line 15).
- Line 6 + atom count (16) - 6 + atom count + bond count (26): Bond block - each line corresponds to a bond where the two first values correspond to the atoms involved in the bond and the third on is the bond type. ISIDA Fragmentor2017 has special bonds for CGRs outlined in tables 2.2 and 2.3.
- Line 28-42: Data block - contains information separated into fields. In this example the fields generated by CA_Prop_Map2011.java are shown. The names of the fields are given as > <NAME>. The format for property mapping for ISIDA Fragmentor2017 is shown. It should be of format:

```
> <COLOR_NAME>
5 1:P 2:H 4: A/D
95 1:A 2:H 4:D
```

where 5 and 95 are the count to be considered for each species and the following characters are Atom number: Coloration1/Coloration2 OLOR_NAME should be indicated with the option -c.

- Line 43: Delimiter indicating end of molecule - the following lines will be a new molecule in the same format.

Bond types

Here, we will describe the bond types with their respective symbols used in the generated descriptors and the integer used in the SDF to identify them. Note that column 3 corresponds to the character 7 to 9 and column 7 corresponds to character 19 to 21 found in the bond block line. The format for CGRs was modified compared to ISIDA Fragmentor2011: The symbols used in the descriptors changed and therefore ISIDA Fragmentor2017 is not retro-compatible with ISIDA Fragmentor2011 in the case of CGRs as well as for ”Any Bonds“ (column 3 = 8)‘ and ”Special Bond“ (column 3 = 9). A new format permitting the visualization of dynamic bond with ChemAxon was implemented. However the previous format (visualization with Edi SDF) is still readable - that is why each Dynamic bond is found twice in the following table. If a custom coloration is provided through the -b option, then it supersedes all other symbols.

Bond Type	**Symbol**	**SDF bond column 3**	**SDF column 7**
Simple	-	1	0 or 2
Double	=	2	0 or 2
Triple	+	3	0 or 2
Aromatic	*	4	0 or 2
Single or Double	.	5	0 or 2
Single or Aromatic	:	6	0 or 2
Double or Aromatic	”	7	0 or 2
Any bond type	?	8	0 or 2
Special bond type	_	9	0 or 2
Single bond in cycle	.	50	0 or 2
Double bond in cycle	:	60	0 or 2
Triple bond in cycle	#	70	0 or 2
Hydrogen bonds	~	80	0 or 2
Unknown bond	YY		0 or 2

Table 2.1: Bond types in SDF

Bond Type	Symbol	SDF bond column 3	SDF column 7
Single bond creation	81	1	8
Double bond creation	82	2	4
Triple bond creation	83	3	12
Aromatic bond creation	84	4	1
Single bond cut	18	1	-1
Double bond cut	28	2	-1
Triple bond cut	38	3	-1
Aromatic bond cut	48	4	-1
Single bond to double bond	12	2	8
Single bond to triple bond	13	3	8
Single bond to aromatic bond	24	4	8
Double bond to single bond	21	1	4
Double bond to triple bond	23	3	4
Double bond to aromatic bond	24	4	4
Triple bond to single bond	31	1	12
Triple bond to double bond	32	2	12
Triple bond to aromatic bond	34	4	12
Aromatic bond to single bond	41	1	1
Aromatic bond to double bond	42	2	1
Aromatic bond to triple bond	43	3	1

Table 2.2: Visualization of CGRs with ChemAxon

Bond Type	Symbol	SDF bond column 3	SDF column 7
Single bond creation	81	81	4
Double bond creation	82	82	4
Triple bond creation	83	83	4
Aromatic bond creation	84	84	4
Single bond cut	18	18	4
Double bond cut	28	28	4
Triple bond cut	38	38	4
Aromatic bond cut	48	48	4
Single bond to double bond	12	12	8
Single bond to triple bond	13	13	8
Single bond to aromatic bond	24	24	8
Double bond to single bond	21	21	8
Double bond to triple bond	23	23	8
Double bond to aromatic bond	24	24	8
Triple bond to single bond	31	31	8
Triple bond to double bond	32	32	8
Triple bond to aromatic bond	34	34	8
Aromatic bond to single bond	41	41	8
Aromatic bond to double bond	42	42	8
Aromatic bond to triple bond	43	43	8

Table 2.3: Visualization of CGRs with EdiSDF

INPUT AND OUTPUT FORMATS

OUTPUT: HEADER FILE AND SVM, SMF AND CSV FORMATS

ISIDA Fragmentor2017 will always output a header file with the extension .hdr and another file in either SVM, SMF or CSV format. By default, the SVM format is outputted and it can be changed with the option -f.

- **SMF**: The SMF (Substructural Molecular Fragments) format outputs 3 files: a header file .hdr, containing the index and a string representing each fragment discovered into the SDF, a sparse descriptor matrix in a .smf file and a one column file with the values of the field identified using the -s option. The sparse descriptor matrix represents one molecule per line. It is read by pairs of columns, the first one identifies a fragment, the second one how many times this fragment was discovered.
- **SVM**: The SVM (Support Vector Machine) format outputs 2 files: a header file .hdr, containing the index and a string representing each fragment discovered into the SDF, and descriptor matrix in a file .svm following the libSVM format. The first column contains the values of the field identified using the -s option. Other columns consist in a pair of values separated by a ":". The first value identifies the fragment's index in the header file, the second one is the fragment count.
- **CSV**: The CSV (Comma-Separated Values) format outputs 2 files: a header file .hdr, containing the index and a string representing each fragment discovered into the SDF, and a sparse descriptor matrix in a .csv file where each value is separated by a semi-colon ";". The first value corresponds to the activity (given by the -s option), and it is then read by pairs of column, the first one identifies a fragment by its index, the second one how many times this fragment was discovered.

NOMENCLATURE

Nomenclature

DEFINITIONS

To characterize the different fragments, they are coded according to the following:

TopologicalFragmentationColorationTypeBondInclusion(LowerLength-UpperLength)CountingType_Options

Where:

1. TopologicalFragmentation is a roman number and corresponds to the following fragmentation:

   I – Sequences (corresponds to –t 1, 2, 3)
   II – Atom-centered fragments (corresponds to –t 4, 5, 6, 7, 8, 9)
   III – Triplets (corresponds to –t 10)

2. ColorationType is a chain of letters starting with a capital and followed by only lower case letters. The following codes have been used up to now:

   A – Atom symbol (when no special coloration is used)

   Ph – Pharmacophoric typing: this typing is created apart from the Fragmentor. To obtain fragments with pharmacophoric typing, one has to follow these 2 steps:

   - First, use a utility to add to the SDF file the field PHType. The PHTYP is generated by CA_Prop_Map2011.java, a java utility created by Dr. D. Horvath. Please read the chapter called “Mapping properties using ChemAxon” in this manual.
   - Then, use the Fragmentor with the SDF containing these atom colorations.

   Ep – Topological electrostatic potentials: this typing is created apart from the Fragmentor. To obtain fragments with this typing, one has to follow these 2 steps:

   - First, use a utility to add to the SDF file the field EPType. The EPTYP is generated by CA_Prop_Map2011.java, a java utility created by Dr. D. Horvath. Please read the chapter called “Mapping properties using ChemAxon” in this manual.
   - Then, use the Fragmentor with the SDF containing these atom colorations.

   Pc – Partial Charges: this typing is created apart from the Fragmentor. To obtain fragments with this typing, one has to follow these 2 steps:

   - First, use a utility to add to the SDF file the field PCType. The PCTYP is generated by CA_Prop_Map2011.java, a java utility created by Dr. D. Horvath. Please read the chapter called “Mapping properties using ChemAxon” in this manual.

- Then, use the Fragmentor with the SDF containing these atom colorations.

Ff – Force Field coloration: this typing is created apart from the Fragmentor. The force field coloration is produced by the assignment of FF atom types as symbols into the definition of fragments. To obtain fragments with this typing, one has to follow these 2 steps:

- First, use a utility to add to the SDF file the field FFType. The FFTYP is generated by CA_Prop_Map2011.java, a java utility created by Dr. D. Horvath. Please read the chapter called “Mapping properties using ChemAxon” in this manual.
    - Then, use the Fragmentor with the SDF containing these atom colorations.

Lp – LogP increments

Ba – Benson atoms (when - -UseBenson was used)

3. BondInclusion simply indicates the inclusion of bond orders in the string with a capital B. If only bonds are used then no ColorationType will appear.
4. LowerLength and UpperLength are the number of atoms to be included at minimum and maximum respectively. Note that a LowerLength=2 and UpperLength=5 will create fragments with at minimum a topological distance of 1 and maximum a topological distance of 4.
5. CountingType corresponds to the type of weight used to count the occurrences of fragments:

- ms – micro-species (pH dependent counting - PHTYP, EPTYP, PCTYP from CA_Prop_Map2012.java are used)

When none is indicated then the direct count is used (weight =1).

6. Options indicate special options used during the fragmentation and are listed below:

- P – AtomPairs (when - -AtomPairs is used)
- R – Restricted (only for atom-centered fragments - corresponds to -t 7,8,9)
- AP – AllPaths (when - -DoAllWays is used)
- FC – FormalCharge representation (when - -UseFormalCharge is used)
- MA1,MA2,MA3 – MarkedAtom with the used option number (-m 1,2 or 3) following the MA
- SF – StrictFragmentation (when - -StrictFrg is used with a specific header in -h header.hdr)
- AD – AllDynamic (Bonds) (when -d 2 is used)
- OD – OneDynamic(Bond) (when -d 1 is used)

Options are separated by a hyphen (-).

Example: IIPhB(3-5)ms_P-FC

NOMENCLATURE

A FEW EXAMPLES OF CORRESPONDANCE BETWEEN ISIDA/FRAGMENTOR2017 OPTIONS AND NOMENCLATURE OF ISIDA DESCRIPTORS

-t	-c	-l	-u	Other options	Nomenclature
0	/	/	/	/	No nomenclature
1	/	2	5	/	IA(2-5)
1	PHTYP	2	8	/	IPh(2-8)ms
1	PHTYP	3	5	-m 1	IPh(3-5)ms_MA
1	/	2	5	--DoAllWays	IA(2-5)_AP
1	/	2	5	--AtomPairs --UseFormalCharge	IA(2-5)_P-FC
2	/	2	7	/	IB(2-7)
3	/	3	6	/	IAB(2-6)
3	EPTYP	3	6	/	IEpB(2-6)
3	PHTYP	3	6	/	IPhB(2-6)
8	/	2	4	/	IIA(2-4)
9	/	2	4	/	IIB(2-4)
10	/	2	4	/	IIAB(2-4)
11	/	2	4	/	IIA(2-4)_R
12	/	2	4	/	IIB(2-4)_R
13	/	2	4	/	IIA(2-4)_R
14	/	2	4	/	IIIA(2-4)

Table 2.4

Mapping properties using ChemAxon

INTRODUCTION

CA_Prop_Map2011 is a java program part of the Utils package based on ChemAxon's JChem classes and developed by Dragos Horvath and Fiorella Ruggiu. It requires therefore a ChemAxon license for the calculation plugin. Note that the pharmacophoric mapping is available on our Mobyle portal (http://infochim.u-strasbg.fr/mobyle-cgi/portal.py#forms::phtype)

USAGE

```
java Utils/CA_Prop_Map2011 –f <ChemAxonInput>
                          [-o <SDFile>
                           -min_ms_pop <double>
                           -major_ms <toggle>
                           -pH <double>]
```

OPTIONS

- -f <input file> (path): the input file path and name. The input can also be piped into the program. It may be of any readable format by ChemAxon
- -o <output file> (path): the output file path and name. By default Typed.sdf. The generated SDF becomes then the input of the ISIDA Fragmentor2017.
- -min_ms_pop (double): the minimum population level of a microspecie for it to be taken into account. By default min_ms_pop=1.0
- -major_ms (toggle): if activated only the major microspecie will be considered
- -pH (double): indicate the pH at which the microspecies are calculated. By default pH=7.4
- -stdoptions (path): **DEPRECATED!!** (path to the file containing rules for the standardize)

The program does not standardize - it is recommended you standardize the file beforehand.

INSTALLATION AND REQUIREMENTS

Download packages

The package is stored on the SVN repository of the web server of the Laboratory of Chemoinformatics. The prerequisite to obtain the package are (i) an account on the server associated to the users group and (ii) an SVN client.

An account can be requested to Dr. G. Marcou (g.marcou@unistra.fr) or Dr. F. Bonachera (f.bonachera@unistra.fr).

MAPPING PROPERTIES USING CHEMAXON

The SVN client is usually natively available on Linux systems. If it is absent, the packaging system frequently names it at subversion. For Max OS X users, the SVN client is provided as a part of xCode later than version 1.7. For Windows, the best option is to use a graphical client. One of the most popular is tortoiseSVN (https://tortoisesvn.net/). For Windows users, be aware that SVN clients usually are invading: they provide additional menus and sets of icons to the whole Windows environment.

Once an SVN client is installed, you can recover the package using the following URL:

```
svn+ssh://<login>@infochim.u-strasbg.fr/home/infochimie/svn/Utils
```

using the command line client the full command line is the following:

```
svn checkout svn+ssh://<login>@infochim.u-
strasbg.fr/home/infochimie/svn/Utils
```

This will create a new Utils directory. The directory can later be updated to obtain new versions of the package. For the command line client, the update can be done by typing, inside this directory:

```
svn update
```

Contents

The ISIDA/Utils package contains the following java executables:

- Aromatize.java
- Ca_Prop_Map*.java
- PH_Type2011.java
- ShortestPath.java
- Tautomerize.java
- pKaAssign.java

It also contains the following rules files (in .xml format) :

- Standardize.xml
- cvffTemplates.xml
- shortPharmFlags.xml

ChemAxon/JChem

The programs are ChemAxon-based software. You will therefore need to download and install ChemAxon/JChem in order for them to be executed correctly.

You can find the ChemAxon software on their website: https://www.chemaxon.com/download/jchem-suite/#jchem

You will need a license to use these tools. You can ask it to Dr. F. Bonachera (f.bonachera@unistra.fr).

MAPPING PROPERTIES USING CHEMAXON

Java

Install a java runtime environment (JRE) and a java development kit (JDK). They can be found in the official Oracle/Java website (http://www.oracle.com/technetwork/java/javase/downloads/jdk9-downloads-3848520.html).

Once java is installed on linux machines, you need to check if it will be used by default. To do this, execute the following command and follow the instructions to select the java oracle jdk as default java:

```
su
update-alternatives --config java
```

Requirements

You will need to modify your CLASSPATH to where your ChemAxon classes can be found.

Bash example:

```
CLASSPATH=/opt/chemaxon/jchem/lib/jchem.jar:
export CLASSPATH
STANDARD_RULES=/opt/scripts/JavaClasses/Utils/Standardize.xml
export STANDARD_RULES
PHARMAFLAG_RULES=/opt/scripts/JavaClasses/Utils/shortPharmFlags.xml
export PHARMAFLAG_RULES
FORCEFIELD_RULES=/opt/scripts/JavaClasses/Utils/cvffTemplates.xml
export FORCEFIELD_RULES
```

Csh example:

```
setenv CLASSPATH /opt/chemaxon/jchem/lib/jchem.jar
setenv STANDARD_RULES /opt/scripts/JavaClasses/Utils/Standardize.xml
setenv PHARMAFLAG_RULES /opt/scripts/JavaClasses/Utils/shortPharmFlags.xml
setenv FORCEFIELD_RULES /opt/scripts/JavaClasses/Utils/cvffTemplates.xml
```

The Color_Atom programs (CA_Prop*.java) also require rules, which are given as XML files in the directory. Please give the path to the files with the following variable names, modify them in the programs or indicate them with the options:

```
-stdoptions Standardize.xml
```

Input files may be of any kind accepted by ChemAxon (https://docs.chemaxon.com/display/docs/File+Formats+Home).

Compilation

Finally, compile CA_Prop_Map2011 with javac.

```
javac /[full path to your java utils scripts]/CA_Prop_Map2011.java
```

Appendix A

ABBREVIATIONS

- CGRs: Condensed Graph of Reactions
- FPT: Fuzzy Pharmacophoric Triplets (ISIDA descriptors)
- IPLF: ISIDA Property-Labelled Fragments (descriptors)
- ISIDA: In SIlico Design and data Analysis
- JDK: Java Development Kit
- JRE: Java Runtime Environment
- QSAR: Quantitative Structure-Activity Relationship
- SDF: Structure-Data File (from MDL - now Accelerys)
- SMF: Substructural Molecular Fragments (ISIDA descriptors)

Bibliography

[1] P. Lacomme, C. Prins, and M. Sevaux, *Algorithmes de graphes*. Eyrolles, second ed., 2003.

[2] A. Varnek, D. Fourches, F. Hoonakker, and V. Solov'ev, "Substructural fragments: an universal language to encode reactions, molecular and supramolecular structures.," *J. Computer-Aided Molecular Design*, vol. 19, pp. 693–703, Jul 2005.

[3] A. Varnek, D. Fourches, D. Horvath, O. Klimchuk, C. Gaudin, P. Vayer, V. Solov'ev, F. Hoonakker, I. Tetko, and G. Marcou, "ISIDA - platform for virtual screening based on fragment and pharmacophoric descriptors," *Curr Comput Aided Drug Des.*, vol. 4, pp. 191–198, Sept 2008.

[4] F. Bonachera, B. Parent, F. Barbosa, N. Froloff, and D. Horvath, "Fuzzy tricentric pharmacophore fingerprints. 1. Topological fuzzy pharmacophore triplets and adapted molecular similarity scoring schemes.," *J Chem Inf Model.*, vol. 46, pp. 2457–2477, Nov-Dec 2006.

[5] F. Bonachera and D. Horvath, "Fuzzy tricentric pharmacophore fingerprints. 2. application of topological fuzzy pharmacophore triplets in quantitative structureactivity relationships.," *J Chem Inf Model.*, vol. 48, pp. 409–425, Feb 2008.

[6] F. Ruggiu, G. Marcou, A. Varnek, and D. Horvath, "Isida property-labelled fragment descriptors," *Mol Inf*, vol. 29, no. 12, pp. 855–868, 2010.

[7] F. Ruggiu, V. Solov'ev, G. Marcou, D. Horvath, J. Graton, J.-Y. Le Questel, and A. Varnek, "Individual hydrogen-bond strength QSPR modelling with ISIDA local descriptors: a step towards polyfunctional molecules," *Mol Inf*, vol. 33, p. 477-487, June 2014.