ISIDA Fragmentor2015 - User Manual

Fiorella Ruggiu, Gilles Marcou,
Vitaly Solov'ev, Dragos Horvath, Alexandre Varnek

Contents

Chapter 1

Introduction

The ISIDA Fragmentor2015 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 Fragmentor2015 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 Fragmentor2015 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/spip.php?rubrique49`).

The laboratory uses the ChemAxon plugins to map a property on the graph. However, one of the aims of ISIDA Fragmentor2015 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 all the options, input and output format description, installation and usage of Fragmentor2015 and the corresponding nomenclature of ISIDA fragment descriptors. Chapter 3 is dedicated to our ChemAxon-based property mapping program.

Chapter 2

Fragmentor2015

2.1 Command line

The ISIDA Fragmentor2015 is a command line only program. You may call upon it using:
PATH/Fragmentor -i <SDFile> -o <BaseName>
[-f <string> -s <string> -h <HeaderFile>]
-t <integer> [{-l <integer> -u <integer>}
-c <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 Fragmentor2015 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 outputed 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, else it might generate errors, memory leaks or wrong fragmentations. Beware that ISIDA Fragmentor2015 does not check the input file before treating it!!

2.2 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 2.4.2)
- -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 (COLOR_NAME) in the SDF of your wished coloration. 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: Colouration1/Colouration2

- -m : If set to 1: All fragments 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)

- -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)

- -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 refer 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.

Type of fragmentation (-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

2.3 Installation

The ISIDA Fragmentor2015 project is versionned with subversion on the infochim server. A few compiled executables are available on our website `http://infochim.u-strasbg.fr` in the Download then Fragmentor section (`http://infochim.u-strasbg.fr/spip.php?rubrique49`). If you need another compiled version or wish to have access to the source code, please contact Pr. A. Varnek (varnek@unistra.fr).

2.3.1 Steps for installation

1. Acquire Fragmentor2015 project using subversion (svn):

svn checkout svn+ssh://yourlogin@infochim.u-strasbg.fr/home/infochimie/svn/Fragmentor2015 Fragmentor2015

2. In the same directory as your Fragmentor2015 directory (cd Fragmentor2015), acquire the Molecule project using svn:

svn checkout svn+ssh://yourlogin@infochim.u-strasbg.fr/home/infochimie/svn/Molecule Molecule

1. Compile the project using preferably Lazarus with fpc or just fpc with the following options: -MObjFPC -Scgi -O3 -g -gl -vewnhi -l -FuMolecule -Fu.

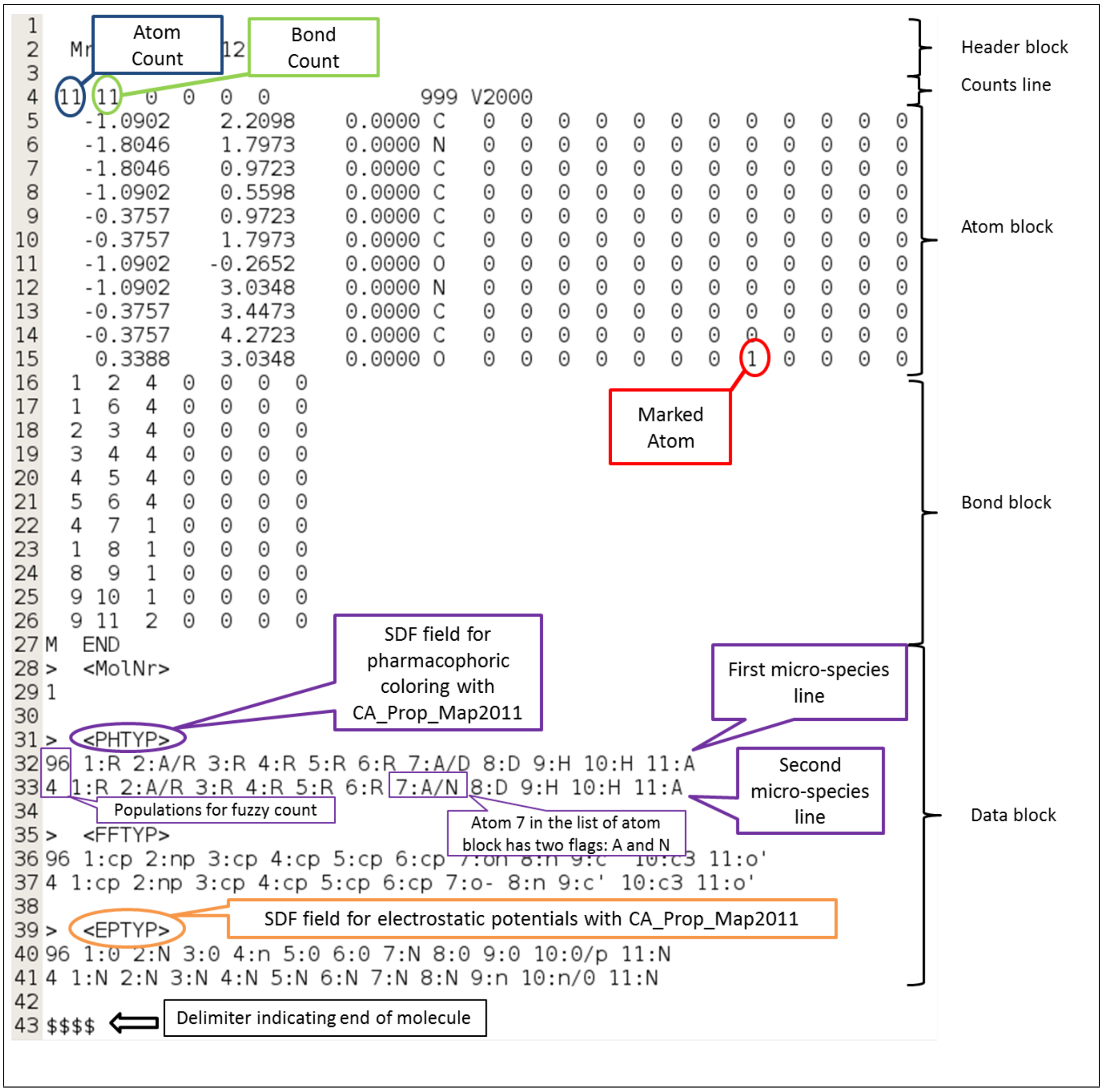

Figure 2.1: Example of SDF

2.4 Input and output formats

2.4.1 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 Fragmentor2015. Here is a quick description of the most important features that an SDF should contain:

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 corresponds to the atoms involved in the bond and the third on is the bond type. ISIDA Fragmentor2015 has special bonds for CGRs outlined in the following table.
- 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 Fragmentor2015 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: Colouration1/Colouration2
COLOR_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 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 Fragmentor2015 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 visualisation of dynamic bond with ChemAxon was implemented. However the previous format (visualisation with Edi SDF) is still readable - that is why each Dynamic bond is found twice in the following table.

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		

2.4.2 Output: Header file and SVM, SMF and CSV formats

ISIDA Fragmentor2015 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 outputed 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 descripor matrix represent one molecule per line. It is read by pairs of column, 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.

Table 2.1: Visualisation of CGRs with ChemAxon

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

The first column contains the values of the field identified using the -s option. Other columns consists 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.

Table 2.2: Visualisation of CGRs with EdiSDF

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

2.5 Nomenclature

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

TopologicalFragmentationColourationTypeBondInclusion (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-centred fragments (coressponds to -t 4, 5, 6, 7, 8, 9)

   III - Triplets (corresponds to -t 10)

2. ColourationType 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 colouration is used)
   - Ph – Pharmacophoric typing (PHTYP generated by CA_Prop_Map2011.java)

- Ep – Topological electrostatic potentials (EPTYP generated by CA_Prop_Map2011.java)
- Pc – Partial Charges (PCTYP generated by CA_Prop_Map2011.java)
- 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 ColourationType 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-centred 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

2.5.1 A few examples of correspondance between ISIDA Fragmentor2015 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)

Chapter 3

Mapping properties using ChemAxon

3.1 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/spip.php?rubrique14

3.2 Usage

textbfjava Utils/CA_Prop_Map2011 -f <ChemAxon input> [-o <SDF> -min_ms_pop <double> -pH <double> -major_ms]
Options in squared brackets are not mandatory.

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 Fragmentor2015.
- -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 (toogle): 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.

3.3 Installation

3.3.1 Steps for installation:

1. Download JChem from ChemAxon's website (`http://www.chemaxon.com/download/jchem/jchem-for-java/`)
2. Install JChem and its licence with the LicenseManager
3. Install a java runtime environment (JRE) and a java development kit (JDK)
4. Download the Utils package (with svn)
5. Edit your shell configuration file (.bashrc for a bash shell) to define the java CLASSPATH and eventually the path to your JRE
6. Compile CA_Prop_Map2011 with javac

3.3.2 ChemAxon JChem

To use this package you will need an installed version of JChem with licence, allowing you to use the calculation plugin. Download JChem from ChemAxon's website (http://www.chemaxon.com/download/jchem/jchem-for-java/). You will need an account on their website to do so. It is easier to use the installation with the JRE. Then install the program and run the LicenseManager to register you license. By default, it should be placed in the .chemaxon directory found in the user's home.

3.3.3 Java

To run and compile the classes, a JRE and a JDK are needed.
For linux, choose the java-sun packages. Configure your media to contain the non-free packages and updates in your mirror list (For Mandriva/GNOME, got to Administration→Configure your system → Software Management → Configure media sources). In the Software Manager, search for the following two packages and install them: java-1.6.0-sun and java-1.6.0-sun-devel. Note: If you installed JChem with a JRE, the java-1.6.0-sun will already be installed.

3.3.4 Utils package

In order to obtain the package, use subversion. To install it on linux, use the Software Manager and install the package. For Windows, use TortoiseSVN (http://tortoisesvn.net). The deposit is on infochimie on the following path: /home/infochimie/svn/Utils. To acquire it, you will need to use the following command:

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

3.3.5 Java CLASSPATH

To compile the java programs using ChemAxon's classes, the CLASSPATH needs to contain the path to them. CA_Prop_Map2012 also requires the definition of variables to find its configuration files. You may define them just before using the program or integrate them into your shell configuration file.
Example of .bashrc:
CLASSPATH=/opt/chemaxon/jchem/lib/jchem.jar:/opt/scripts/JavaClasses
export CLASSPATH
STANDARD_RULES=/opt/scripts/JavaClasses/Utils/Standardize.xml
export STANDARD_RULES
SH_PHARMAFLAG_RULES=/opt/scripts/JavaClasses/Utils/shortPharmFlags.xml
export SH_PHARMAFLAG_RULES
FORCEFIELD_RULES=/opt/scripts/JavaClasses/Utils/cvffTemplates.xml
export FORCEFIELD_RULES

Example of .cshrc:
setenv CLASSPATH /opt/chemaxon/jchem/lib/jchem.jar:/opt/scripts/JavaClasses
setenv STANDARD_RULES /opt/scripts/JavaClasses/Utils/Standardize.xml
setenv SH_PHARMAFLAG_RULES /opt/scripts/JavaClasses/Utils/shortPharmFlags.xml
setenv FORCEFIELD_RULES /opt/scripts/JavaClasses/Utils/cvffTemplates.xml

3.3.6 Javac Compilation

Compile the program using the following command:
javac /opt/scripts/JavaClasses/Utils/CA_Prop_Map2012.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 structure-activity 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," *Molecular Informatics*, 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. tbp, p. tbp, tbp 2014.