

```
Vector ringSets = RingPartitioner.partitionRings(
    sssrf.findSSSR(molecule)
);
```

After a scan of all 249,071 NCI molecules, we collected 11610 unique ring systems. In order to build a 3D structure for a new modeling candidate, we first examined its molecular structure for the existence of one of the template ring systems. To map the template ring systems onto a query structure the **UniversallIsomorphismTester** class should be used. With the **QueryAtomContainer** class one is now able to configure a substructure or isomorphism search for your own requirements (e.g. search only for equivalent bond systems). If a template ring system could be mapped, its coordinates were assigned to the modeling candidate and aliphatic chains were layed out thereafter.

Currently, molecules with unknown ring systems cannot be handled by this approach.

Dr. Christian Hoppe
 Universität zu Köln, Germany
 christian.hoppe@uni-koeln.de

Bibliography

- [1] E.L. Willighagen. What's 2004 going to bring? *CDK News*, 1(1):3–4, 2004.
- [2] N.L. Allinger. *Force Fields: A Brief Introduction*, volume 2 of *Encyclopedia of Computational Chemistry*, pages 1013–1015. 1998.
- [3] J.R. Maple. *Force Fields: A General Discussion*, volume 2 of *Encyclopedia of Computational Chemistry*, pages 1015–1024. 1998.
- [4] J. Sadowski. *3D Structure Generation*, volume 1 of *Handbook of Chemoinformatics*, pages 231–261. 2003.
- [5] Rudolph C. Sadowski J. Gasteiger, J. *Automatic Generation of 3D-Atomic Coordinates for Organic Molecules.*, volume 3 of *Tetrahedron Comput. Methodol.*, pages 537–547. 1990.
- [6] Takahasi Y. Abe H. Sasaki S. Ihlenfeldt, W.D. *CACTVS: A Chemistry Algorithm Development Environment*, pages 102–105. Daijuukagakutouronkai Dainijuukai Kouzoukassiseisoukan Shinpojiumu Kouenyoushishuu. 1992.

Spok - The Spectrum Organisation Kit

An introductory overview on the current state of the Spectrum Organisation Kit.

by Tobias Helmus

The Spectrum Organisation Kit is a program written in Java for the organisation and visualisation of spectral data. It is now in an alpha state and is intended to be used by experimental scientists for the analysis of spectral data, the assignment of these data to structural information and the management of larger amounts of spectral/structural data. It will be published under an open source license.

At the moment the program is able to display continuous and peak data in form of charts by using the JFreeChart library [1]. A comparable view of the peak and the continuous chart is provided, and zooming into all charts is already implemented. For giving the possibility to enter a structure for every spectrum the JChemPaint editor for 2D molecular structures [2] is embedded into the application.

Persistence of the spectral and structural data is realised by serialisation of the spectrum objects to XML files via the XStream library [3], whereas the structural information is written to CML files using the CDK **CMLWriter**. The code for the latter looks like this:

```
FileWriter out = new FileWriter("Filename");
```

```
CMLWriter cmlwriter = new CMLWriter(out);
cmlwriter.write((SetOfMolecules) structure);
cmlwriter.close();
```

where structure is the **SetOfMolecules** constructed by painting a structure in JChemPaint [2].

For the spectrum object, the `toXML()` method of the XStream package returns the java object in one XMLString which then can be written to a file with a standard **FileWriter**:

```
File file = new File("Filename");
FileWriter out;
XStream xstream = new XStream();
String xml = xstream.toXML((Spectrum) spectrum);
out = new FileWriter(file);
out.write(xml);
out.close();
```

At the time of writing the program was capable of reading spectral data in the jcamp-dx format. For this purpose the jcamp-dx library hosted on SourceForge [4] was used, which will be the reference implementation of the IUPAC JCAMP-DX spectroscopy data standard [5]. This library offers routines for reading and writing all spectrum types and data formats defined in this standard. Using the **JCampReader** is very straightforward and can easiest be done by creating a spectrum from a string containing the whole jcamp-dx file:

```
JCampReader jcamp = JCAMPReader.getInstance();
jcampSpectrum = jcamp.createSpectrum(jcampString);
```

If a file without a peak table is imported, a peak extraction can be performed by one of the two peak picking algorithms provided by the jcamp-dx package. Spok will shortly be downloadable in a first version at SourceForge and on our group website at <http://almost.cubic.uni-koeln.de/jrg>.

Tobias Helmus
Junior Research Group for Applied Bioinformatics
University of Cologne, Germany
tobias.helmus@uni-koeln.de

Bibliography

[1] The JFreeChart Homepage. <http://www.jfree.org/jfreechart/>.

<http://www.jfreechart.org/jfreechart/>.

[2] C. Steinbeck S. Krause, E. Willighagen. JChemPaint - Using the Collaborative Forces of the Internet to Develop a Free Editor for 2D Chemical Structures. *Molecules*, 5:93-98, 2000.

[3] The XStream Homepage. <http://xstream.codehaus.org/>.

[4] The JCAMP-DX Library Homepage. <http://sourceforge.net/projects/jcamp-dx/>.

[5] R.S. McDonald P.S. McIntyre D.N. Rutledge A.N. Davies P. Lampen, R.J. Lancashire. A Generic JCAMP-DX Standard File Format, JCAMP-DX V.6.00. 2002.

Frequently asked Questions

"Frequently asked Questions" is the first article of a series in the CDK newsletter. It is compiled of selected questions and answers that are taken from the CDK user mailing list (cdk-user@lists.sourceforge.net). Additionally, the author considers questions that are of general interest for CDK users. All credits for the expert answers go to the helpful developers and users who are contributing to the user mailing list.

by Uli Fechner

What exactly is the difference between the SaturationChecker, the ValencyChecker, and the ValencyHybridChecker?

All three classes provide methods for checking whether the valences of an atom are saturated with regard to a particular atom type. The oldest class is **SaturationChecker**. It is exclusively able to deal with neutral atoms. To overcome this limitation the **ValencyChecker** has been written. It not only adds the facility to handle charged atoms but also implements a slightly different algorithm. During the development of the SMILES parser another problem occurred: the SMILES 'c1ccccc1' was not parsable into the aromatic 'benzene', unless an atom type list was used that considers hybridization states. Such a list, together with an enhanced algorithm compared to the **ValencyChecker**, is employed by **ValencyHybridChecker** so that hybridization states can be dealt with. The atom type lists of **SaturationChecker**, **ValencyChecker** and **ValencyHybridChecker** are accessible over the URL <http://cdk.sourceforge.net/atlists.html> or in the directory 'cdk/src/org/openscience/cdk/config' of an installed distribution.

Where can I find the API documentation in the CDK source package?

The API HTML documentation is created when the command `ant -f javadoc.xml` is executed in the CDK root directory. This creates a directory '/doc/api' in the CDK root directory that contains the complete API documentation.

Do canonical SMILES created by the CDK match the Daylight canonical SMILES?

No, they do not match each other. The CDK method implements the algorithm described by Weininger *et al.* in 1989 [1]. After several years of this publication Daylight changed their rules for generating canonical SMILES. To date their current method is not published. This lack of information does not allow the implementation of the new Daylight rules for canonical SMILES.

According to the API the class Molecule extends the class AtomContainer but it does not add anything at all to what is provided in AtomContainer. What is the purpose of Molecule?

The class **Molecule** exists to give a meaning to the content of an **AtomContainer**. Strictly speaking, a **Molecule** should not allow fragmented moieties. But this is not enforced at the moment.