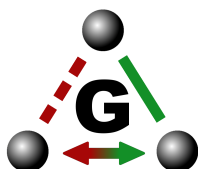


GGL Tutorial: Visualization



Norah Schnorr², Christoph Flamm^{1,*} and Martin Mann²

¹ Institute for Theoretical Chemistry, Vienna University



² Bioinformatics Group, University of Freiburg



<http://www.tbi.univie.ac.at/software/GGL/>

Version April 24, 2015

Built for GGL version 4.1.0-trunk

*Send comments to xtof@tbi.univie.ac.at or mmann@informatik.uni-freiburg.de

Contents

1	In General	1
2	molGml2pic.pl - Molecule visualization	1
2.1	Call	1
2.2	Example	2
3	reactionGml2pic.pl - Reaction visualization	3
3.1	Call	4
3.1.1	Educt-Product Visualization	5
3.1.2	Reaction ITS Visualization	5
3.2	Example	6

1 In General

Within the GGL, graphs are specified using the GML language. Since these GML definitions can become quite large and hard to read the GGL provides several tools to visualize molecules and reactions within its perl module.

Internally the visualization scripts call the **Graphviz** library and therefore require its presence. From the **Graphviz** library the **neato** layout program is used (reference to graphviz.org).

Important **neato** parameters are:

- “-T” specifies the output format, such as SVG, PNG, PDF. All scripts need to be called accordingly with “-Tsvg > outimage.svg” for example.
- “-h”, “-help” and “-?” display the **neato** help message.
- “-index” displays node indices in the node labels. This option is not available in **reactionGml2pic.pl** and **reactionItsGml2pic.pl** which by default display node indices to ease the mapping between educt and product graph.

See section 2.1 and 3.1 for further examples.

The different visualization tools all need a special GML input which are explained in the corresponding sections 2.1 and 3.1.

2 molGml2pic.pl - Molecule visualization

This section shows how to use the **molGml2pic.pl** script to visualize molecules. Given a molecule in GML notation the script produces a graphical depiction using the **Graphviz** **neato** layouting program. The following section explains how to call **molGml2pic.pl** and section 2.2 shows an example.

2.1 Call

The **molGml2pic.pl** script needs a GML as input representing a SMILES encoding of a molecule. Therefore GGL provides the **molTool** to transfer SMILES encodings into GML format. The **molTool** is called with the following parameters:

- input mode: “-inMode=S” for SMILES encodings

- output mode: “-outMode=G” to generate GML format or “-outMode=L” to generate ... ?
- input GML: “-in=MOLECULE.gml”

Using the `molTool` and `molGml2pic.pl` direct piping is possible:

```
echo 'c1ccccc1' | molTool -inMode=S -outMode=G \
| perl molGml2pic.pl -Tsvg > benzene.svg
```

For the parameters of the `molGml2pic.pl` script which are required by the `Graphviz` call see section 1.

2.2 Example

In this section the usage of the `molGml2pic.pl` script is explained by an example.

In the following we present an example that uses the `molTool` to convert the ...?? molecule into GML and then calls the `molGml2pic.pl` to generate a graphical depiction in Scalable Vector Graphics (SVG) format. The corresponding GML is shown underneath and the resulting image is shown in Fig. 1.

```
\$ echo c1ccccc1 | molTool -inMode=S -outMode=G \
| perl molGml2pic.pl -Tsvg > benzene.svg
```

The resulting graph in GML format:

```

graph [
  node [ id 0 label "C" ]
  node [ id 1 label "N" ]
  node [ id 2 label "c" ]
  node [ id 3 label "c" ]
  node [ id 4 label "c" ]
  node [ id 5 label "c" ]
  node [ id 6 label "n" ]
  node [ id 7 label "c" ]
  edge [ source 1 target 0 label "-" ]
  edge [ source 2 target 1 label "-" ]
  edge [ source 3 target 2 label ":" ]
  edge [ source 4 target 3 label ":" ]
  edge [ source 5 target 4 label ":" ]
  edge [ source 6 target 5 label ":" ]
  edge [ source 7 target 6 label ":" ]
  edge [ source 7 target 2 label ":" ]
]

```

3 reactionGml2pic.pl - Reaction visualization

To create graphical depictions of molecule reactions GGL provides two visualization scripts:

- reactionGml2pic.pl (3.1.1)
- reactionItsGml2pic.pl (3.1.2)

Both show a depiction of the educt and product graphs of the reaction. The `reactionItsGml2pic.pl` script additionally depicts a imaginary transition state (ITS).

Via node indices the vizualization includes additional mapping information.

In the educt graph, bond and charge changes occuring in the reaction are colored in red and in the product graph in green, respectively.

In `reactionItsGml2pic.pl` the coloring of the nodes encodes the charge changes in the reaction. Negative charge changes are shown with green node color and red labels. Positive charge changes with red node color and green labels respectively. Black nodes and labels denote no charge change of the atom.

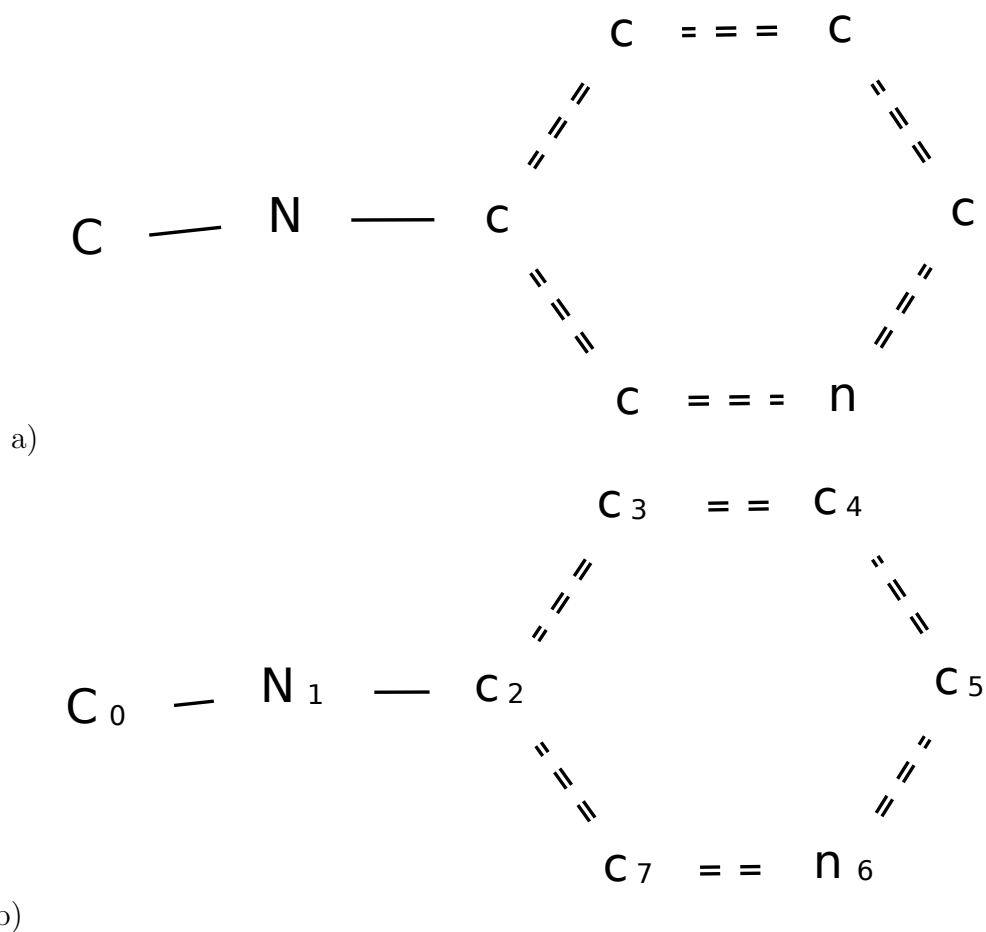


Figure 1: The benzene molecule depicted with `molGml2pic.pl` (a) and using the “-index” parameter (b)

Both scripts need a special **GML** encoding which specifies the bond and charge changes of the reaction. The next section explains how to call the scripts and section 3.2 shows an example.

3.1 Call

This sections explains how the `reactionGml2pic.pl` and `reactionItsGml2pic.pl` can be called. The input of both scripts is a compacted **GML** which includes all bond and charge changes occuring along a reaction. Therefor the **GGL** provides the `reactionTool`. The `reactionTool` takes a reaction in **GML** format as input and generates a compacted version of it. It is called with the

following parameters:

- input mode: “-inMode=G”
- output mode: “-outMode=C”
- input GML: “-in=REACTION.gml”

For the parameters of `reactionItsGml2pic.pl` and `reactionGml2pic.pl` which are required by the `Graphviz` call see section 1.

3.1.1 Educt-Product Visualization

This section explains the usage of the `reactionGml2pic.pl` script. This script depicts a molecule reaction as educt and product graphs including atom mapping information via node indices which are depicted as subscripts of the node labels.

The input of the `reactionGml2pic.pl` is a compacted GML which can be created with the `reactionTool` (3.1). An example is shown in section 3.2.

Using the `reactionTool` and the `reactionGml2pic.pl` direct pipelining is possible:

```
reactionTool -in=dielsalder\_rule.gml -inMode=G -outMode=C \  
| reactionGml2pic.pl -Tsvg > diels.alder.svg''
```

3.1.2 Reaction ITS Visualization

The `reactionItsGml2pic.pl` script depicts additionally to educt and product graph and imaginary transition state (ITS). This ITS represents an overlap of the educt and product molecules and highlights all changes occurring along the reaction.

Via node indices which are depicted as subscripts of the node labels additional mapping information is provided.

The input of the `reactionItsGml2pic.pl` script is a compacted GML which can be created using the `reactionTool` (3.1). An example is shown in section 3.2.

Using the `reactionTool` and `reactionItsGml2pic.pl` direct pipelining is possible:

```
reactionTool -in=dielsalder\_rule.gml -inMode=G -outMode=C \
| reactionItsGml2pic.pl -Tsvg > diels.alder.svg
```

3.2 Example

In this section we present an example that uses the `reactionTool` to convert the Diels-Alder reaction given in GML format into a compacted GML. Then the `reactionGml2pic.pl` and the `reactionItsGml2pic.pl` are called with this compacted GML as input to generate a graphical depiction in Scalable Vector Graphics (SVG) format. The corresponding GML and compacted GML are shown in the following and the resulting images can be seen in Fig. 2 for the `reactionGml2pic.pl` and in Fig. 1 for the `reactionItsGml2pic.pl` respectively.

The Diels-Alder rule in GML:

```
rule [
  ruleID "Diels-Alder reaction"
  context [
    node [ id 0 label "C" ]
    node [ id 1 label "C" ]
    node [ id 2 label "C" ]
    node [ id 3 label "C" ]
    node [ id 4 label "C" ]
    node [ id 5 label "C" ]
  ]
  left [
    edge [ source 1 target 2 label "=" ]
    edge [ source 4 target 5 label "-" ]
    edge [ source 3 target 4 label "=" ]
    edge [ source 0 target 5 label "=" ]
    constrainNoEdge [ source 0 target 2 ]
    constrainNoEdge [ source 3 target 1 ]
  ]
  right [
    edge [ source 0 target 1 label "-" ]
    edge [ source 1 target 2 label "-" ]
    edge [ source 2 target 3 label "-" ]
    edge [ source 3 target 4 label "-" ]
    edge [ source 4 target 5 label "=" ]
    edge [ source 5 target 0 label "-" ]
  ]
]
```

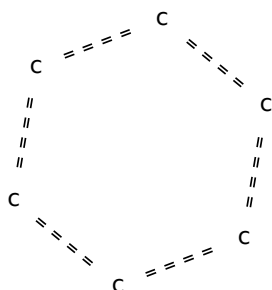



Figure 2: The Diels-Alder reaction depicted using the `reactionGml2pic.pl`.

The compacted GML after calling “\$ reactionTool -in=diels-alder.gml -inMode=G -outMode=C”:

```
graph [
  node [ id 0 label "C" ]
  node [ id 1 label "C" ]
  node [ id 2 label "C" ]
  node [ id 3 label "C" ]
  node [ id 4 label "C" ]
  node [ id 5 label "C" ]
  edge [ source 3 target 4 label "=|-" ]
  edge [ source 0 target 5 label "=|-" ]
  edge [ source 1 target 2 label "=|-" ]
  edge [ source 4 target 5 label "-|=" ]
  edge [ source 2 target 3 label "|-" ]
  edge [ source 0 target 1 label "|-" ]
]
```

The resulting figures:

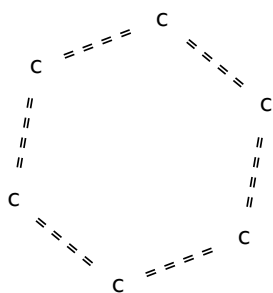


Figure 3: The Diels-Alder reaction depicted using the `reactionItsGml2pic.pl`.