PythonCyc

and other APIs

A Python package to access Pathway Tools and its data using the Python programming language

Mario Latendresse

March 2016

APIs for Pathway Tools

- An Application Programming Interface (API) is a set of methods, or functions, to programmatically interact with an application such as Pathway Tools
- Pathway Tools have (foreign) APIs for Perl, Java, and Python
- For PerlCyc and JavaCyc, see the solgenomics (SGN) web site:

  `https://solgenomics.net/downloads/index.pl`

- Stay tune for PythonCyc

What is PythonCyc?

- It is a Python package to access Pathway Tools using the Python programming language
- It allows access to the data (PGDBs) and modify it using Python
- It also allows access to many functions defined in Pathway Tools to manipulate genes, proteins, reactions, pathways, compounds and more.

Online Documentation

- http://bioinformatics.ai.sri.com/ptools/pythoncyc.html
- You will see links for a tutorial and a complete API documentation
- Installation instructions of PythonCyc are also provided
- The PythonCyc package is on github.com
  https://github.com/latendre/PythonCyc

Installing PythonCyc

- Download the .zip file from github.com
- Unzip the file
- Cd to the directory created
- Execute

```
sudo python setup.py install
```

- You will be asked for your password
- The PythonCyc package is globally installed on your computer

Python Server

- Pathway Tools needs to start the Python Server
- You can allow or deny **remote** access to the Python server
- Deny remote access:

```
./pathway-tools -python-local-only
```

- Accept remote access:

```
./pathway-tools -python
```

- Once started, Pathway Tools is ready to communicate with PythonCyc

Starting PythonCyc

- As any Python package, you import it

```
import pythoncyc
```

- You then select the databases desired

```
meta = pythoncyc.select_organism('meta')
ecoli = pythoncyc.select_organism('ecoli')
```

- All future accesses to these databases use variables meta and ecoli

Getting Data the Lazy Way

- All genes

```
>>> genes = ecoli.genes
>>> genes
<PFrame class |Genes| currently with 4500 instances
(ecoli)>
```

- Printing the frame ids

```
>>> for gene in genes.instances:
...     print gene.frameid
```

- Printing their common names

```
 ...    print gene.common_name
```

The Lazy Way Cached the Data

- The previous examples cached (keep) the data local in Python
- The data is not transferred from Pathway Tools a second time
- Example: (assuming product has not be accessed yet)

```
>>> genes.instances[0]
```

- Product attribute is not displayed

```
>>> Genes.instances[0].product
```

- The product value is transferred and displayed

```
>>> Genes.instances[0]
```

- Product value is now kept in that instance 0.
- Data modified in these instances does not modify the corresponding data in Pathway Tools

Other Classes of Objects

- The previous discussion applies to all other classes of objects

```
>>> reactions = ecoli.reactions
>>> compounds = ecoli.compounds
>>> pathways = ecoli.pathways
>>> proteins = ecoli.proteins
>>> people = ecoli.people
```

Other Way to Transfer Data

- The methods get_slot_value and get_slot_values always transfer the data
- We need to use frame ids
- Example with the left attribute of all reactions of ecoli

```
>>> reactions = ecoli.reactions
>>> for reaction in reactions.instances:
>>> ...  print ecoli.get_slot_values(reaction.frameid, 'left')
```

Modifying Data in a PGDB

- Two methods are available

  `put_slot_values and put_slot_value`

- Example with MetaCyc and one reaction

```
meta.put_slot_value('RXN-9000','GIBBS-0',2.1)
```

- The attribute GIBBS-0 for reaction RXN-9000 is modified with value 2.1 directly in the PGDB stored in Pathway Tools (not locally!)
- Saving a PGDB (bound to var pgdb) by Pathway Tools can be done by

```
pgdb.save_pgdb()
```

Callable Pathway Tools Functions

- There are over 150 functions defined in Pathway Tools that can be called in PythonCyc
- All these functions are described in the API documentation at

  http://pythoncyc.readthedocs.org/en/latest/

- Examples:

```
pgdb.all_operons()
pgdb.genes_of_pathway(pathway)
pgdb.enzymes_of_gene(gene)
pgdb.reactions_of_compound(compound)
pgdb.reactions_of_pathway(pathway)
```

Complete Example

```
def gather_gibbs_substrates_of_reactions(orgid):
    """
    Return a dictionary of all reactions of PGDB orgid with
    the Gibbs free energies of formation
    of their substrates. The keys are the frame ids of the
    reactions and the values are dictionaries
    of compound frame ids to Gibbs free energies of formation.
    """
    pgdb = pythoncyc.select_organism(orgid)
    rxn_frameids = pgdb.all_rxns(type='all')
    gibbs_dict = {}
    for rxn_fid in rxn_frameids:
        cpds_fids = pgdb.get_slot_values(rxn_fid,'SUBSTRATES')
        gibbs_dict[rxn_fid] = dict([(cpd_fid,
                      pgdb.get_slot_value(cpd_fid, 'GIBBS-0'))
                      for cpd_fid in cpds_fids])
    return gibbs_dict
```