Planning the official release of RMG-Py

issues to resolve and issues to put off

Connie Gao

4/11/2014

RMG Study Group

What do users need?

- Easy installation
- Comprehensive documentation and examples
- Transparent and easy to manipulate databases
- Code stability
- Something to cite

How do we give users what they need?

- Easy installation
  - Package RMG as an executable for Windows (py2exe or NSIS), Linux (Freeze or PyInstaller), and Mac (py2app)
- Comprehensive documentation and examples
  - Write them!
- Transparent and easy to manipulate databases
  - Upgrade to universal database and provide GUI for manipulation
- Stability
  - Stable but extensible input files and adjacency lists (strive for backwards compatibility from version 1.0 onwards)
  - Finalize RMG's current features and hold off on developing features
- Something to cite
  - RMG-Py software paper planned in conjunction with release

Universal database: requirements

- User friendly both in raw form and through a GUI
- Capable of storing lengthy reference information
- Extensive error checking

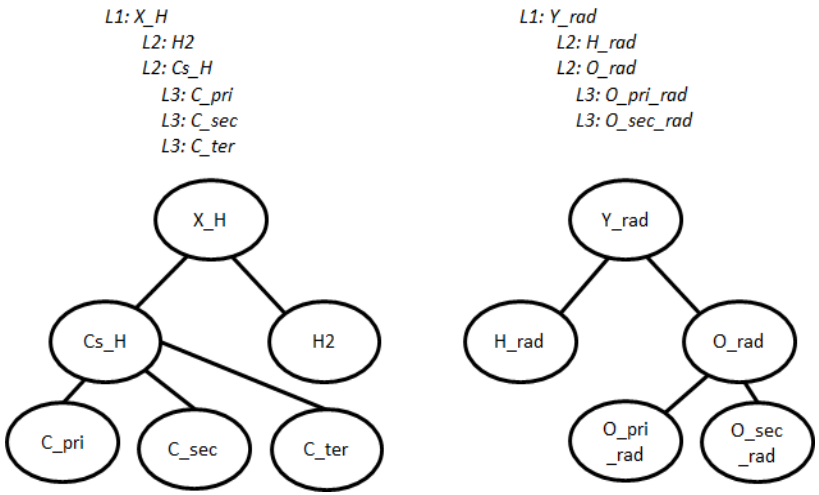
Universal database: making it user-friendly

- Python style raw database files modified according to user suggestions:
  - Separate dictionaries from list of reactions
  - Reactions and rate rules searchable by string
  - Allows user to easily compare values without extensive scrolling
  - Capability of storing long comments

Universal database: making it user-friendly

- Use website as a GUI portal for users to modify the database (capabilities are already online but need maintenance)
  - Django allows individual user accounts: make the website a portal for *modifying* personal copies of the database
  - Better visualization and displaying rates for comparisons
  - Input file creation through the website (capability already there, just not up to date)
- Advantage of using the website is that we can *actively maintain* a single working copy— the user does not have extra software or patches to install

Universal database: error-checking



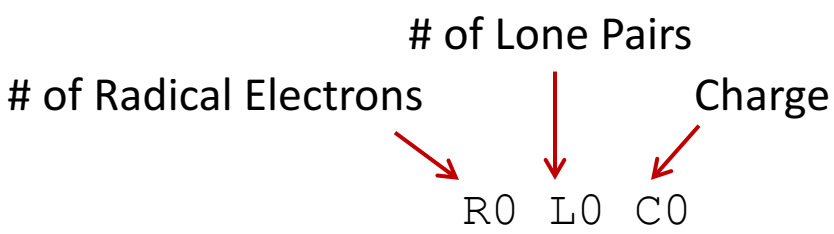
- Nathan has been working on a test script that should be cleared whenever the database is modified.
  - Child checking in trees
  - Identification of duplicates
  - Cross checks of names and adjlists
- We now have internal mass balance and duplicate checks for thermo and reaction libraries

Code stability: finalizing the adjacency list

Multiple ways to represent methane (CH_4)

Radicals only No explicit hydrogens	Radicals and lone pairs No explicit hydrogens	Radicals, lone pairs, and charges Explicit hydrogens

`1 C R0`
 	
Minimum Representation
	`1 C R0 L0`	`1 C R0 L0 C0 {2,S} {3,S} {4,S} {5,S}`
		`2 H R0 L0 C0 {1,S}`
		`3 H R0 L0 C0 {1,S}`
		`4 H R0 L0 C0 {1,S}`
		`5 H R0 L0 C0 {1,S}`



- Hydrogens can be either explicit or non-explicit
- Assume default values for charges and lone pairs when unspecified
- Letter flags make adjlists extensible for new attributes

- Devise rigorous unit tests and write clear documentation

Code stability: finalizing RMG's current features and holding off on developmental features

- Gas and liquid phase chemistry
- Hydrocarbon (C, H, O) and heteroatom (N, S) support
- Parameter estimation
  - Thermochemistry: update aromaticity perception
  - Kinetics: eliminate cyclic transition state double-counting
  - Transport
- *Spin state conservation (in progress)*
- Pressure dependent networks
- On-the-fly quantum mechanics
  - Thermo
  - *Kinetics (in progress)*
- *Sensitivity analysis (in progress)*
- CanTherm

Final task list

1. Universal raw database with error checking
2. Finalize current features (make up to date with Java)
3. Devise examples and write RMG-Py paper
4. Package and release

Lower priority tasks

5. Make website a functional GUI for database
6. Development of additional features