GrunDB: a tool for validating QM algorithms in GAMESS(US)

Riccardo Murri
<riccardo.murri@uzh.ch>

Grid Computing Competence Centre, University of Zurich
http://www.gc3.uzh.ch/

Swiss Grid Day, Nov. 30, 2010

University of Zurich

What is GAMESS?

GAMESS(US) is a program for ab initio molecular quantum chemistry. It can perform a number of general computational chemistry calculations (too many to list here!).

GAMESS runs adopts a single program model: one command to perform all these computations.

The input file combines both the molecule geometry and the specification of what to compute.

For more information, see `http://www.msg.chem.iastate.edu/gamess/` and the following journal articles:

- "General Atomic and Molecular Electronic Structure System", M. W. Schmidt, K. K. Baldridge, J. A. Boatz, S. T. Elbert, M. S. Gordon, J. H. Jensen, S. Koseki, N. Matsunaga, K. A. Nguyen, S. Su, T. L. Windus, M. Dupuis, J. A. Montgomery, *J. Comput. Chem.*, 14, 1347-1363(1993).
- "Advances in electronic structure theory: GAMESS a decade later", M. S. Gordon, M. W. Schmidt, pp. 1167-1189, in *Theory and Applications of Computational Chemistry: the first forty years*, C. E. Dykstra, G. Frenking, K. S. Kim, G. E. Scuseria (editors), Elsevier, Amsterdam, 2005.

A problem in algorithm validation

Baldridge Group @ UZH: development of new algorithms and procedures within the GAMESS suite.

Problem: how to check that the new algorithms' results are consistent with what we already know?

Solution: compare results with known data. The more extensive the comparison, the better the validation.

A problem in algorithm validation

Baldridge Group @ UZH: development of new algorithms and procedures within the GAMESS suite.

Problem: how to check that the new algorithms' results are consistent with what we already know?

Solution: compare results with known data. The more extensive the comparison, the better the validation.

Ideal high-throughtput processing case!

- *Many independent* jobs
- Spread them around SMSCG

What is GRunDB then?

GRunDB is a tool to automate:

- running GAMESS on a data-bank of molecule geometries
- comparing computed stoichiometry results with known-good ones

Functional high-level view

GRunDB is implemented as a Linux command-line tool on top of the GC3Libs/GC3Utils toolkit.

GRunDB takes as input:

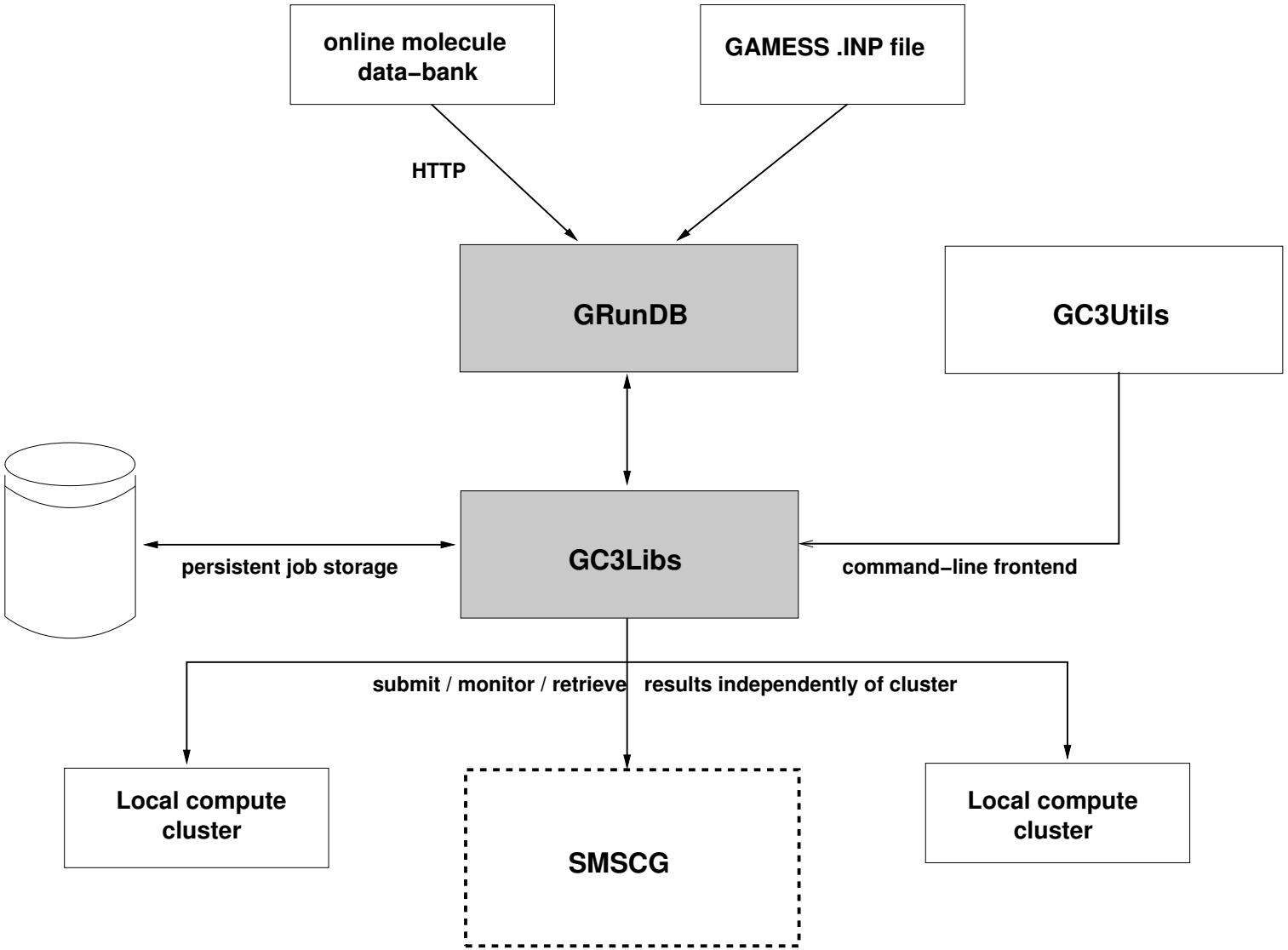
- a set of molecules (a subset of an online data-bank)
- a GAMESS input file

It creates a GAMESS job for each molecule in the subset, plugging its geometry in the template input file.

When all jobs are done:

- results are extracted from the output files
- a summary table compares computed energy with reference data

Architecture



The GC3Libs framework

GC3Libs is a set of Python classes to submit and control batch jobs to clusters and grid resources.

GC3Libs takes care of the job processing and computational resource control; GRunDB does the pre- and post-processing.

GC3Libs has an *application-oriented* paradigm: each application has its own specialized job class, that knows how to cope with *that* application's own features.

Find out more at: `http://gc3pie.googlecode.com/`

The GAMESS.UZH molecule database: Subset index

University of Zurich > Organic Chemistry Institute

Baldridge Research Group | Contact | Sitemap

University of Zurich

The GAMESS.UZH molecule database

Edit | RecentChanges | Preferences | Discussion

Welcome to the GAMESS.UZH molecule database

The available molecules are divided into subsets; the list of all subsets is available below. For each molecule, a table of reference stoichiometry data is given, accompanied by the molecule geometry in GAMESS `.inp` file format.

This site is organized as a Wiki; only authorized users can edit pages to add or modify content, but anyone can view the pages and download data files.

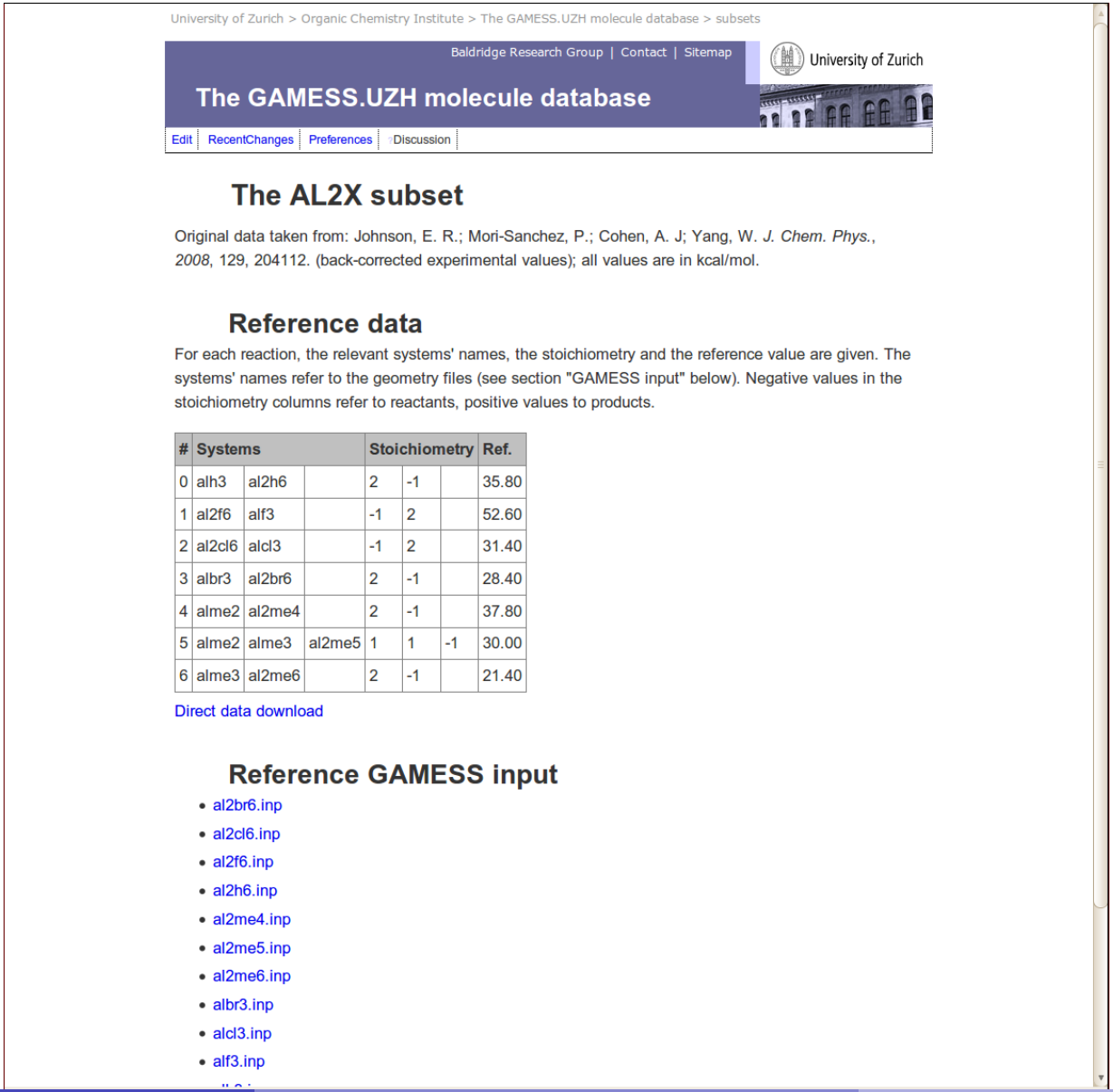
The database is maintained by people in the Baldridge group at the Organic Chemistry Institute of the University of Zürich; IT support is provided by the Grid Computing Competence Centre.

Available molecule sets

The available molecules are divided into subsets; the list of all subsets is available below. For each molecule, a table of reference stoichiometry data is given, accompanied by the molecule geometry in downloadable GAMESS `.inp` file format.

- ACONF
- AL2X
- BH76
- BH76RC
- BHPERI
- CYCONF
- DARC
- DC9
- G21EA
- G21IP
- G2RC
- IDISP
- ISO34
- MB08-165
- NBRC
- O3ADD6
- PA
- PCONF

The GAMESS.UZH molecule database: The AL2X subset



University of Zurich > Organic Chemistry Institute > The GAMESS.UZH molecule database > subsets

Baldridge Research Group | Contact | Sitemap

University of Zurich

The GAMESS.UZH molecule database

Edit | RecentChanges | Preferences | Discussion

The AL2X subset

Original data taken from: Johnson, E. R.; Mori-Sanchez, P.; Cohen, A. J; Yang, W. *J. Chem. Phys.*, *2008*, 129, 204112. (back-corrected experimental values); all values are in kcal/mol.

Reference data

For each reaction, the relevant systems' names, the stoichiometry and the reference value are given. The systems' names refer to the geometry files (see section "GAMESS input" below). Negative values in the stoichiometry columns refer to reactants, positive values to products.

#	Systems			Stoichiometry			Ref.
0	alh3	al2h6		2	-1		35.80
1	al2f6	alf3		-1	2		52.60
2	al2cl6	alcl3		-1	2		31.40
3	albr3	al2br6		2	-1		28.40
4	alme2	al2me4		2	-1		37.80
5	alme2	alme3	al2me5	1	1	-1	30.00
6	alme3	al2me6		2	-1		21.40

Direct data download

Reference GAMESS input

- al2br6.inp
- al2cl6.inp
- al2f6.inp
- al2h6.inp
- al2me4.inp
- al2me5.inp
- al2me6.inp
- albr3.inp
- alcl3.inp
- alf3.inp

Start analysis of a molecule set

GRunDB creates a job for each molecule in the specified subset. A *session* name must be given to record the current analysis.

Subsets can be specified by their name on the web page. More than one (or *ALL*) can be analyzed in a single GRunDB go.

```
$ ./grundb.py new AL2X session1
Input file name  State (JobID)       Info
================================================================================
al2br6           NEW (job.680)       New at Mon Nov 29 14:06:46 2010
al2cl6           NEW (job.681)       New at Mon Nov 29 14:06:46 2010
al2f6            NEW (job.682)       New at Mon Nov 29 14:06:46 2010
al2h6            NEW (job.683)       New at Mon Nov 29 14:06:46 2010
 ...
alf3             NEW (job.689)       New at Mon Nov 29 14:06:47 2010
alh3             NEW (job.690)       New at Mon Nov 29 14:06:47 2010
alme2            NEW (job.691)       New at Mon Nov 29 14:06:47 2010
alme3            NEW (job.692)       New at Mon Nov 29 14:06:47 2010
```

Submit jobs to the Grid

Once a session has been created, a single command invocation is needed to submit jobs to SMSCG or University clusters.

```
$ ./grundb.py progress session1
Insert AAI/Switch password for user  m1058036 :
Queue selected: all.q@idgc3grid01.uzh.ch
File uploaded: /tmp/rmurri/rsl.qOp5wn
File uploaded: /home/rmurri/gc3/gc3utils/0.10/grundb/take1.inp.d/AL2X/al2f6
   ...
Input file name  State (JobID)       Info
================================================================================
al2br6           SUBMITTED (job.680)  Submitted at Mon Nov 29 14:08:04 2010
al2cl6           SUBMITTED (job.681)  Submitted at Mon Nov 29 14:07:53 2010
al2f6            SUBMITTED (job.682)  Submitted at Mon Nov 29 14:07:29 2010
al2h6            SUBMITTED (job.683)  Submitted at Mon Nov 29 14:07:41 2010
   ...
alh3             SUBMITTED (job.690)  Submitted at Mon Nov 29 14:07:50 2010
alme2            SUBMITTED (job.691)  Submitted at Mon Nov 29 14:07:59 2010
alme3            SUBMITTED (job.692)  Submitted at Mon Nov 29 14:08:02 2010
```

Monitor job progress and execution

The same command is used to monitor job execution. This makes it easy to use any periodic command scheduler to carry out the task.

```
$ ./grundb.py progress session1
Input file name  State (JobID)        Info
================================================================================
al2br6           SUBMITTED (job.680)  Submitted at Mon Nov 29 14:08:04 2010
al2cl6           RUNNING (job.681)    Running at Mon Nov 29 14:08:40 2010
al2f6            RUNNING (job.682)    Running at Mon Nov 29 14:08:40 2010
al2h6            RUNNING (job.683)    Running at Mon Nov 29 14:08:40 2010
   ...
alh3             RUNNING (job.690)    Running at Mon Nov 29 14:08:40 2010
alme2            SUBMITTED (job.691)  Submitted at Mon Nov 29 14:07:59 2010
alme3            SUBMITTED (job.692)  Submitted at Mon Nov 29 14:08:02 2010
```

Output retrieval and post-processing

Again, `grundb progress` will automatically retrieve and post-process results of jobs that have finished execution, extracting the energy values needed to compute stoichiometry results.

```
$ ./grundb.py progress session1
File downloaded: gsiftp://idgc3grid01.uzh.ch:2811/jobs/17057129103608390550
File downloaded: gsiftp://idgc3grid01.uzh.ch:2811/jobs/17057129103608390550
   ...
Input file name  State (JobID)       Info
================================================================================
al2br6           RUNNING (job.680)   Running at Mon Nov 29 14:09:00 2010
al2cl6           RUNNING (job.681)   Running at Mon Nov 29 14:08:40 2010
al2f6            DONE (job.682)      Final-b2plyp energy= -1084.1929109480
al2h6            DONE (job.683)      Final-b2plyp energy= -488.2225951188
   ...
alh3             DONE (job.690)      Final-b2plyp energy= -244.0835095562
alme2            DONE (job.691)      Final-b2plyp energy= -322.6947308201
alme3            DONE (job.692)      Final-b2plyp energy= -361.9996423212
```

Interoperability / 1

All the data related to the jobs created by GRunDB is stored on the filesystem. This is a design decision: users must retain full control of the data and be able to inspect every sibngle job.

session.csv This is where job IDs and their statuses are recorded. It's the same information you see on the video, but it can be imported into a spreadsheet.

session.inp.d Input files for all GAMESS jobs, named after the molecule they represent. Can be edited, and then resubmit the job.

session.out.d Output and error files from a job; this is the place to inspect if something has gone wrong, or if you want to read the full log file of the computations.

Interoperability / 2

GRunDB creates jobs and stores them using GC3Libs mechanisms, so you can operate on jobs with the GC3Utils command-line tools as well.

Using GC3Utils allows for operations on a single job:

- What's the running status of the largest job of all?
  → Use the `gstat` command.
- Why is this molecule taking so long to compute?
  → Use the `gtail` command to peek at the output/error log.
- Oops! I need to change parameters for one molecule only.
  → Use the `gresub` command to re-start a job from the beginning.

Finally...

When all jobs are done, GRunDB computes stoichiometry data and compares it to the reference data.

```
$ ./grundb.py progress session1
Input file name  State (JobID)       Info
================================================================================
al2br6           DONE (job.682)      Final r-m06 energy is -15929.9575541891
   ...                                                     after 19 iterations
alme3            DONE (job.694)      Final r-m06 energy is -362.1226427375
                                                           after 18 iterations
STOICHIOMETRY DATA
Reaction                                Comp. energy  (Ref. data; deviation)
================================================================================
                                          AL2X
--------------------------------------------------------------------------------
2*alh3 + -1*al2h6                              +35.70  (+35.80; -0.10)
-1*al2f6 + 2*alf3                              +48.46  (+52.60; -4.14)
-1*al2cl6 + 2*alcl3                            +28.15  (+31.40; -3.25)
2*albr3 + -1*al2br6                            +25.41  (+28.40; -2.99)
2*alme2 + -1*al2me4                            +34.52  (+37.80; -3.28)
1*alme2 + 1*alme3 + -1*al2me5                  +28.55  (+30.00; -1.45)
2*alme3 + -1*al2me6                            +21.72  (+21.40; +0.32)
```

Thank you!

(Any questions?)

References

GRunDB `http://gc3pie.googlecode.com/svn/branches/0.10/grundb`

GC3Libs `http://gc3pie.googlecode.com/`

GAMESS(US) `http://www.msg.chem.iastate.edu/gamess/`

"General Atomic and Molecular Electronic Structure System", M. W. Schmidt, K. K. Baldridge, J. A. Boatz, S. T. Elbert, M. S. Gordon, J. H. Jensen, S. Koseki, N. Matsunaga, K. A. Nguyen, S. Su, T. L. Windus, M. Dupuis, J. A. Montgomery, *J. Comput. Chem.*, 14, 1347-1363(1993).

"Advances in electronic structure theory: GAMESS a decade later", M. S. Gordon, M. W. Schmidt, pp. 1167-1189, in *Theory and Applications of Computational Chemistry: the first forty years*, C. E. Dykstra, G. Frenking, K. S. Kim, G. E. Scuseria (editors), Elsevier, Amsterdam, 2005.

Well, and the errors...?

Errors happen, and it's not necessarily a users' fault. (e.g., disk full on a remote cluster)

How does GRunDB deal with this?

Well, and the errors...?

Errors happen, and it's not necessarily a users' fault. (e.g., disk full on a remote cluster)

How does GRunDB deal with this?

Use GC3Utils to get control over the *individual* job:

- Inspect job output and logs in the *session.out.d* directory (or use the `gtail` command)
- Take measures against the failure
- Re-submit the failed job with `gresub`
- GRunDB will notice and re-process the results when the new job is done.