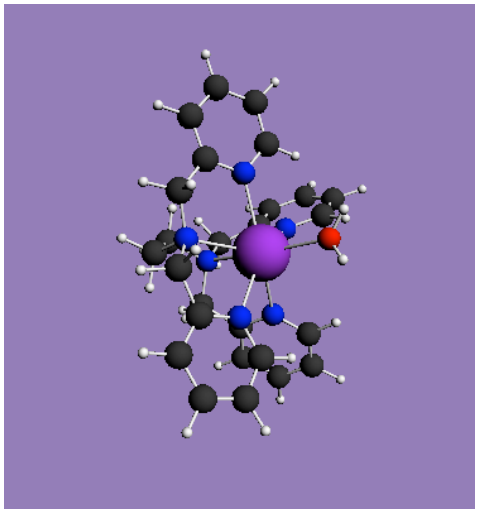
Quantum Chemistry on a Grid

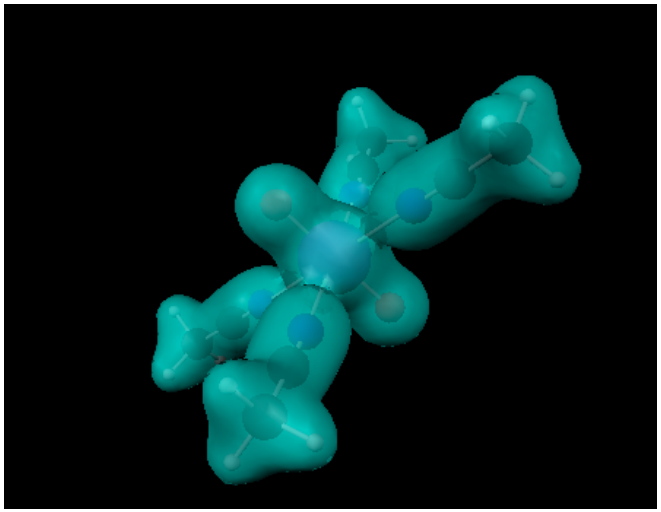
Lucas Visscher
Vrije Universiteit Amsterdam

Outline

- Why Quantum Chemistry needs to be gridified
  - Computational Demands
  - Other requirements
  - What may be expected ?
- Program descriptions
  - Dirac
    - Organization
    - Distributed development and testing
  - Dalton
  - Amsterdam Density Functional (ADF)
- An ADF-Dalton-Dirac grid driver
  - Design criteria
  - Current status

Computational (Quantum) Chemistry





QC is traditionally run on supercomputers

Large fraction of the total supercomputer time annually allocated

- Characteristics
  - Compute Time scales cubically or higher with number of atoms treated. Typical : a few hours to several days on a supercomputer
  - Memory Usage scales quadratically or higher with number of atoms. Typical : A few hundred MB till tens of GB
  - Data Storage variable. Typical: tens of Mb till hundreds of GB.
  - Communication
    - Often serial (one-processor) runs in production work
    - Parallelization needs sufficient band width. Typical lower limit 100 Mb/s.

Changing paradigms in quantum chemistry

- Computable molecules are often already “large” enough
- Many degrees of freedom
  - Search for global minimum energy is less important
  - Statistical sampling of different conformations is desired

- Let the nuclei move, follow molecules that evolve in time
  - 2 steps : (Quantum) dynamics on PES
  - 1 step : Classical dynamics with forces computed via QM

- Changing characteristics
  - General: more similar calculations
  - Compute time : Increases : more calculations
  - Memory usage : Stays constant : typical size remains the same
  - Data storage : Decreases : intermediate results are less important
  - Communication: Decreases : only start-up data and final result

Computers : characteristics

- Supercomputers
  - ☹ Few machine: not much compute time
  - ☺ Resources per machine are large
- Clustercomputers
  - 😐 More machines: more compute time
  - ☺ Time per machine: weeks (if necessary)
  - 😐 Resources per machine are adequate
- Grids
  - ☺ Flexible setup,can be used to bundle small clusters
  - ☺ In principle many machines: abundant compute time
  - ☹ Time per machine is limited
  - ☹ Memory per machine may be limited, typical <1 GB
  - ☹ Data storage per machine is limited

The Past ....

The Future ?










1: Independent calculations

PES calculations for quantum dynamics

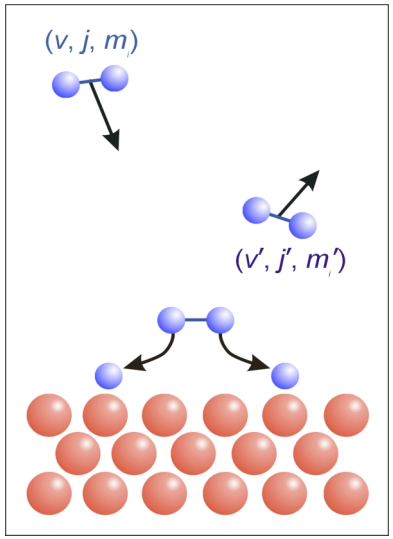
- First step is the calculation of hundreds or thousands of different structures
- Each calcuation is independent and gives only one real number as final answer
- Grid can be used to speed up the calculations by distributing them over several individual machines

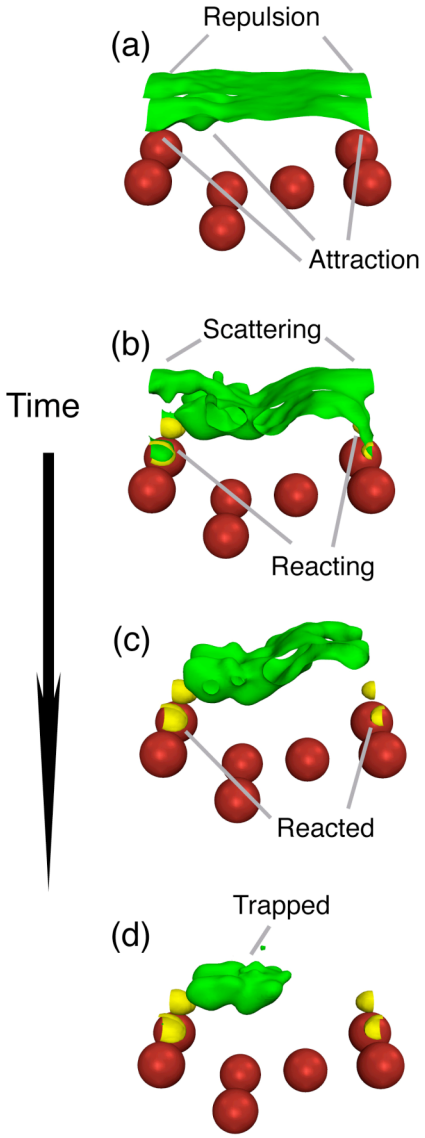
Application area

- Heterogenic catalysis
- Hydrogen Storage

Active groups

- VUA (Baerends)
- UL (Kroes, Olsen)





2: Sequences

Refining a Potential Energy Surface

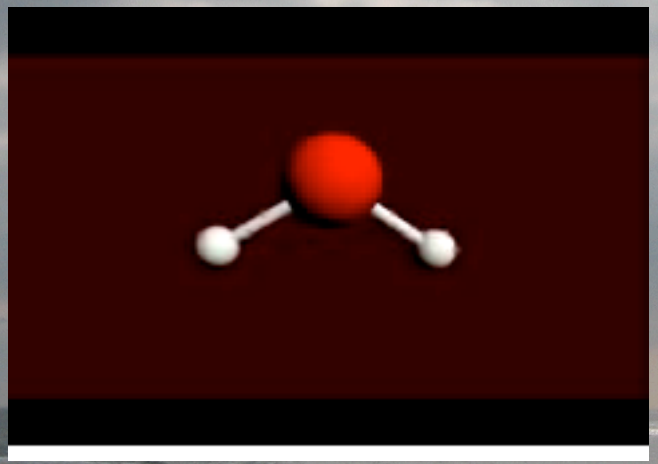
- After a quick first scan, make more detailed maps

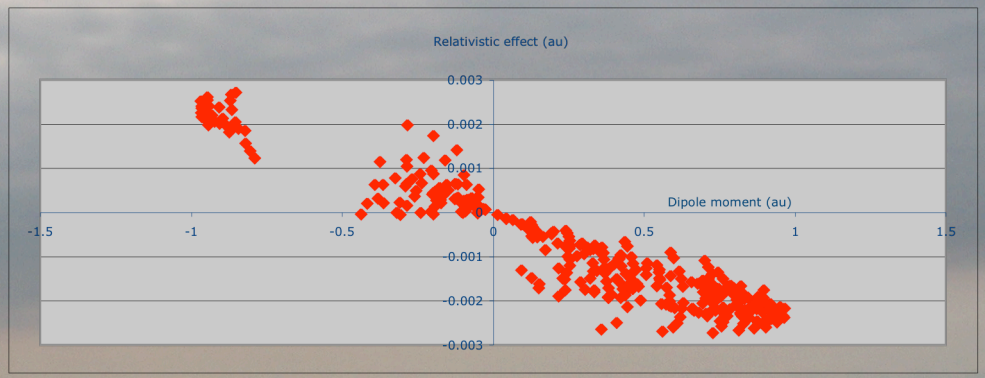
Sequence of different methods

- Structure with method A
- Relative energies with method B
- Molecular properties with method C

Application

- Water dipole moment surface (van Stralen, Visscher, Tennyson, Cszazar)

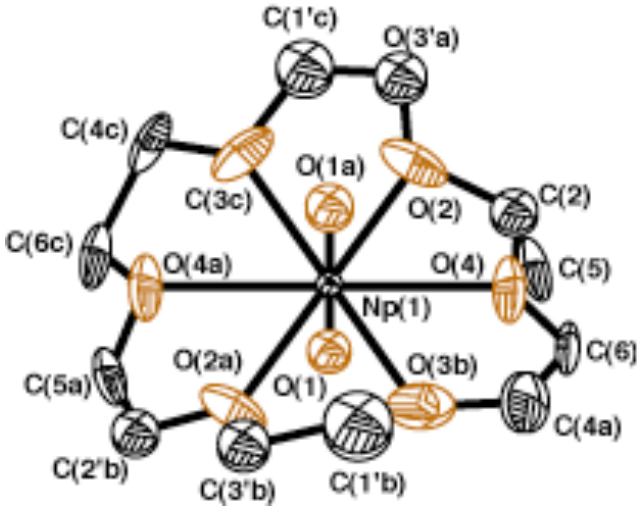




3: Parallel calculations

“Divide and Conquer” methods

- Independent calculation of subsystems
- Periodic “couplings” (update) steps

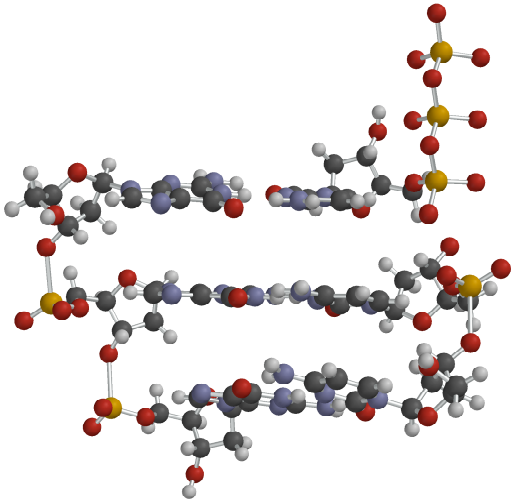


Structure optimization of large molecules

- Should compete with supercomputer approach
- Present day parallel implementations need to be modified (less communication, less memory per calculation)

Toepassingen

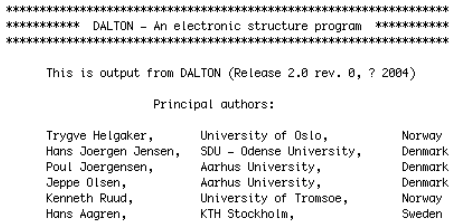
- Ln/Ac complexes (Visscher, VUA)
- Simulation DNA replication (Bickelhaupt, VUA)



Programs targeted

DALTON

- Free for academic users
- Relatively small molecules
- Many molecular properties
- Many methods



DIRAC

- Free for academic users
- Small molecules
- Specialized in heavy element chemistry

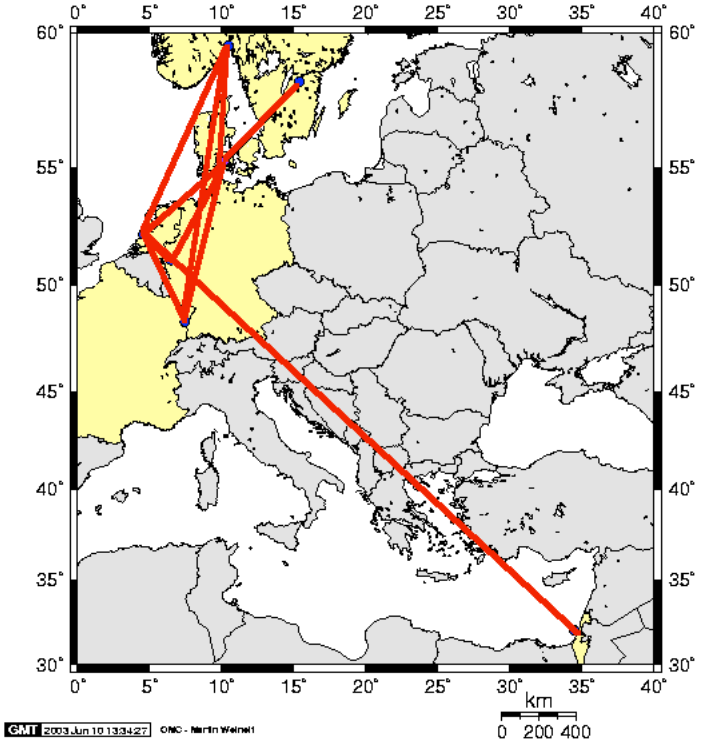
ADF

- Commercial (SCM: spin-off TC group VUA)
- Relatively large molecules
- Many molecular properties
- Exclusively Density Functional Theory

Scientific Computing & Modelling

DIRAC - a European metalaboratory for developing computational techniques and software for 4-component relativistic quantum chemical calculations

Collaborations 2000-2005



Groups involved

1. Faegri / Helgaker (Department of Chemistry, University of Oslo)
2. Fleig (Institut für Theoretische Chemie, Düsseldorf)
3. Jensen (Department of Chemistry, University of Southern Denmark)
4. Kaldor (School of Chemistry, University of Tel Aviv)
5. Norman (Department of Physics and Measurement Technology, University of Linköping)
6. Saue - (Laboratoire de Chimie Quantique et Modélisation Moléculaire, University of Strasbourg
7. Visscher - (Faculty of Sciences, Vrije Universiteit Amsterdam)

Mission statement

The purpose of this collaboration is to establish a metalaboratory coordinating a joint European effort to develop a state-of-the-art program system for 4-component relativistic quantum chemical calculations that extend the realm of high accuracy quantum chemical calculations to the lower regions of the periodic table where relativistic effects play an important role. Special effort is dedicated to make this program system easily accessible by non-expert users and guarantee its free distribution throughout the entire research area.

DIRAC@dirac.chem.sdu.dk

- CVS server : authenticated access for developers
  - Automatic backup of repository
  - Authenticated access to registered developers
  - Automatic notification of commits via author mailing list
- Web server : authenticated access for developers
  - Latest code documentation
  - Archive of authors mailing lists
  - List of license holders
  - Notes from author meetings and internal documents
- Web server : non-authenticated access for general public
  - Code documentation of the latest released version
  - Archive of users mailing list
  - Information about distribution
- FTP server : authenticated access for registered users
  - Download of release version and patches
  - Download additional data (basis sets)

Development & Testing

1. Definition of representative and sensitive tests
   - Economical sample case for specific functionality
   - Specify the lines of output that give the critical numbers
   - Choose the precision in the filter file
   - Condensed information in a report file

2. Automatic testing
   - Test runs are stored in the cvs repository
   - Nightly testing of updated code on two platforms, using three different compilers (one using parallel runs)
   - Weekly test of CPU-intensive runs

3. Benchmarking & profiling
   - GFlop counters at selected (critical) spots
   - MPI communication time analysis

Current distribution and use

- Distribution of source code : aimed at experienced computational chemists who often add their own extensions
- Installation of the code via Install script that handles machine-dependent features. Public domain configure tools used
- A script to run the code (one single executable) both interactively and for most batch systems
- Documentation and support available via public website and mailing list for registered users.

- Typical runs are still at high end of computing spectrum, large memory, fast CPU and/or fast communication is required
- Installation and execution are not blackboxed, in contrast to commercial codes

Design general grid driver

Easy of use

- Should be intuitive for computational chemists
- Should offer acces to all resources available to the user
- Should work with mentioned three programs and be extendible
- Should have added value relative to the normal mode of operation

Grids

- Should handle heterogenous grids
- Should not require an executable on all machines
- Should handle license requirements (ADF : machine-specific license)

Technical Aspects

- Use existing middleware
- Testbed on grid environment inside our university and outside (DAS2)
- Use expertise of local computational science group (Bal, Merzky)

Choice of middleware

- Use a high-level middleware to access the functionality of Globus
- QC codes have a long lifetime, need to select an APIs that is reasonably mature, to minimize updating time
- Competing standards: select a middleware that can be transfed to other environments

- Other important selection criteria
  - Sufficient functionality and stability
  - Local expertise available

Chosen solution

- Gridlab (http://www.gridlab.org): EU project that aims to provide tools to make using grid services simpler
- Use “Gridlab Application Toolkit’’ (GAT)
- Interface to different “middlewares” via a higher level API
- GAT Implemented in C, wrappers in Python

- QC specific code in Python, makes combination with other non-grid scripts easier

User Interface

- User interface to resources: written in python, using GAT to interact with grid resources
- Basic operations: submit/stop/query/list jobs
- Other operations: job creation, configuration
- Current form: command-line tool
- GUI: planned after first evaluation of the driver by expert users

Command-line Interface

- grid-cli.py --help

```
grid-cli.py --configure [--project <project name>]

grid-cli.py --create        [--type <job_type>]
                            [--project <project name>]
                            [--location <project location>]
                            [--dalton <file(s)> [ --molecule <file(s)> ] |
                            --dirac <file(s)>    --molecule <file(s)>    |
                             --adf <file(s)   ]

grid-cli.py --submit  [-m <max jobs running>] [--project <project name>]
                      [--location <project location>] [--job <job name>]
grid-cli.py --stop    [--id <job id> | --project <project name> | --job <job name>]
grid-cli.py --query   [--id <job id> | --project <project name> | --job <job name>]
grid-cli.py --clear   [--id <job id> | --project <project name> | --job <job_name>]
grid-cli.py --list    [ --project <project name> ]
```

Command-line Interface

Configure step

> grid-cli.py --configure [--project <project name>]

This “setup” creates directories and sets global preferences for the project

Some information about the grid resources needs to be provided if no resource management and detection framework exists (for e.g. scratch dirs)

Command-line Interface

Job creation step

> grid-cli.py --create [options]

Three files created

- script to run the calculation(s)
- software resource descriptions
- hardware resource descriptions

The script contains a compressed tarfile that will create a “sandbox” for the job

Binaries fetched later or included in this tarfile

Command-line Interface

Next step is job submission:

> grid-cli.py --submit [options]

GAT provides the frontend to submission and keeps a database entry about this job

This database holds information such as job status and any other information the user likes to add

Upper limit to the maximum number of jobs running at the same time might be specified

Command-line Interface

Other management activities through GAT:

(a) Information can be queried from the database

> grid-cli.py --query [options]

(b) Jobs can be stopped/cancelled

> grid-cli.py --stop [options]

(c) List jobs (or clean the job list)

> grid-cli.py --list [options]

> grid-cli.py --clear [options]

QC Infrastructure

Current setup:

- Executables for different architectures available via web server.
- Non-executable data (basis sets) comes from the submitters machine

Planned:

- One gridftp/scp service for both executables and supplementary constant data
- Variable data to be sent with the jobscript

Open Issues

- Where/How to maintain the code-related infrastructure (binaries, basis sets)?
- Authentication issues (certificates expire rather quickly for QC standards)

- "Proof-of-concept" application started: improvement of relativistic corrections in W-n (n=1,2,3) thermochemistry methods

Experiences of a quantumchemist.....

Getting access to a grid

- Not very difficult, requires identification (passport) to have a certificate valid for a limited time (6 months)

Setting up a grid (or hooking up your system)

- “Ask a computational scientist”
- Need to install Globus, Gridlab

Using the middleware

- Not all adapters work properly (but you can run locally...)
- Plethora of options may be bewildering at first sight

Doing calculations

- Testruns are typically limited to a few minutes
- Production grids: need to apply for compute time, difference with traditional cluster computing is marginal
- Alternative: convince others (non-computational chemists) to pool (unused) CPU time, but what’s in it for them ?

Unsolved Problems

Licensing and safety

- Guest misbehaviour : Illegal actions on host machine
- Host misbehaviour : Illegal copying of software possible
- Vendor restrictions : Installation only on specific machines

(Con)Temporary Solutions

- Current grids are still small, a pioneering “trustworthy” community
- Temporary license, grid certificate is valid for limited time anyhow

Better solutions

- Run virtual machine on the host (solves guest problem)
- Software companies should revise their licensing policy

New Possibilities

- Companies may offer computer and or software usage time

Conclusions

Use of computational grid for quantum chemistry

- Requires little code adapation, mostly done on the scripting level
- Asks for organizational talent in the set-up of the grid and the associated virtual organizations

Possibilities

- Large scans of interesting molecules
- Reduced elapsed times
- A much larger range of applications if the prophecies come true

Short range goals

- Finalize the grid driver
- Compute relativistic effects in G2/G3 set

Thanks to

People

- **Andre Gomes**
- Andre Merzky, Rob van Nieuwpoort

Money

- National Computing Facilities (NCF, Netherlands)
- COST-D23