Max Webinar

BigDFT

Software approach of BigDFT: from modularization to containers. Aiida workflows with PyBigDFT

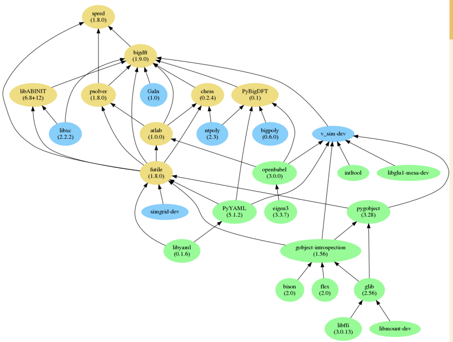
Augustin Degomme

Laboratoire de Simulation Atomistique - L_Sim

November 12, 2020

Virtual Room

Code release and distribution



Modularity first

BigDFT-suite : collection of different independent libraries with own build system.
Third-party libraries (green) and upstream modules (blue)
☛ Dependencies expressed easily in the jhbuild-based bundler.

- Lots of possible options
- Very versatile
- Python configuration files can be shared, many provided
- Good or Expert knowledge often required - not very user friendly

Code release and distribution

Provide new alternatives to users

- Packaged versions
- Virtual machines
- Containers

Which users ? which usages ?

- Development
- HPC
- Analysis

Requirements

We need

- System packages (`bison, flex, cmake,...`)
- Linear algebra packages (`blas, lapack, MKL?,...`)
- Upstream packages (`libyaml, glib, libffi,...`)
- Community packages (`simgrid, ntpoly, libxc,...`)
- BigDFT packages

Moreover

- CUDA has to be installed
- The MPI layer should be CUDA-aware (GPUdirect)
- On workstation and frontends we use `jupyter` notebooks
- The compilation instructions are cumbersome. Difficult to control all these things for non-expert developers.

Containers as a possible solution

Our container history

- Development of GPU acceleration for exact exchange with GPUdirect
- Development of PyBigDFT API
- CI

Flavours

- SDK: large, with everything to build, no BigDFT ▸ SDK
- runtime: stripped, with bigdft/MPI built from SDK ▸ runtime
- Also available on NVidia NGC repository ▸ NGC

Containers as a possible solution

Features

- Built using Nvidia HPC container maker toolkit :
- Comes with CUDA/OpenCl, MKL, either MVAPICH2 or OpenMPI, Jupyter server
- Works on non-GPU systems, ARM/x86 platforms, Windows (using WSL2).
- BigDFT libraries with/without vectorized instructions, dynamically selected
- Tested with GPUDirect on singularity and shifter

Example of a running command

```
nvidia-docker run -it --rm
       -v $(pwd):/host_pwd -w /host_pwd \
       nvcr.io/hpc/bigdft:cuda10-1804-mkl \
       bigdft
```

Other packages

Virtual Machine

- BigDFT is part of MaX flagship codes, is available on the Quantum Mobile virtual machine. QM
- Great for training/schools

Package

- Debian package in the making: Easy to install, less optimized.
- Python package for BigDFT run analysis (futile and PyBigDFT)

PyBigDFT API

Developed in 2016

SystemCalculator: the CalcJob equivalent

```
from BigDFT import Calculators as C
from BigDFT import Inputfiles as I
single_point=C.SystemCalculator()
inp = I.Inputfile()
inp.set_xc('LDA')
inp.write_orbitals_on_disk()
log=single_point.run(input=inp,posinp='mol.xyz')
print (log.energy)
```

Dataset: a small equivalent of a WorkChain

```
from BigDFT import Datasets as D
hgrid_cv=D.Dataset('h_set')
for h in [0.5,0.45,0.4,0.35,0.3]:
    inp.set_hgrid(h)
    hgrid_cv.append_run(id={'h':h},input=inp,runner=
                                        single_point)
results=hgrid_cv.run()
energs=hgrid_cv.fetch_results(attribute='energy')
```

Advantage of aiida technology

We have implemented the “traditional” flavour of AiiDA plugin.

`pip install aiida-bigdft`

▸ BigDFT plugin

`AiidaCalculator` ☛ used to remotely submit the job

```
from BigDFT import AiidaCalculator as A
study=A.AiidaCalculator(code="bigdft@localhost",
        num_machines=1,mpiprocs_per_machine=1,
        omp=1,walltime=3600)
%load_ext jupyternotify
%notify
hgrid_cv.wait()
>>> '0 processes still running'
```

Integrated in PyBigDFT

▸ PyBigDFT AiiDA runs

A technology that makes the notebook a *console* to launch the job and to analyze production data

The console container

AiiDA requirements

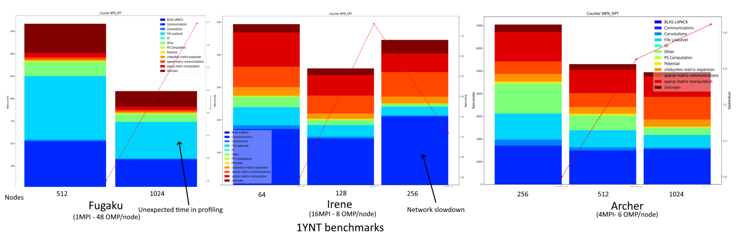
- Installation of the database-related packages
- Working on virtual machine (may still interefere with user's installation/distribution)
- Configuration of the remote machines still to be completed

The Console container

▸ Console

- Install AiiDA, Aiidalab with (Py)BigDFT plugin and analysis tools, Jupyter directly
- No need to install low level layers (MPI, CUDA,. . .) or BigDFT
- Can dialog with supercomputers where the HPC installation is performed by system's administrators
- Control center for BigDFT experiments

Benchmarking with Aiida and PyBigDFT (20k atom system)



Next

- Compute hours have been granted on Fugaku. More performance data to follow in the forthcoming months.
- Emulation/tests on SVE architectures
- LibConv benchmarking