GSoC21 W5: LFortran Design Details and minidftatom

Rohit Goswami · 2021-07-09 · gsoc21 · fortran · ramblings

> *Project scaffolding and compiler design*

Background

Serialized update for the 2021 Google Summer of Code under the fortran-lang organization, mentored by Ondrej Certik.

Logistics

- Met with Ondrej on Tuesday, Wednesday and Thursday
  - Discussed the AST, ASR and backends in more detail
- Zeroed in on several alternate designs implementations of intrinsic functions
  - We have prototype calls for the two kinds of compile evaluation
  - My `sin` implementation (pre-GSoC21) is pretty unwieldy and a cleaner method which we elected to pursue is the `minval` strategy
- Discussed concrete methods of working
- Talked about the runtime library
  - This is a late stage issue, after the ASR generation

Overview

This week focused around a subset of `dftatom` which Ondrej prepared. In fact the main highlight was working through possible code design directions and iterating on these with Ondrej (and Gagandeep asynchrously) to come up with the intrinsic function design methods and the compile time evaluation of expressions plan issue (discussed below in issue 420).

New Merge Requests

Restructuring of the ASR (1024)

A code cleanup task, to reduce code smell from `using` statements and to refactor into more logical file structures

Grammar modifications (1037)

Name changes to keep a consistent clear implementation structure

Freshly Assigned Issues

minidftatom Roadmap (401)

A clone of the `SNAP` and `dftatom` issues

size() not handled correctly for arrays (416)

This is an ASR bug, with a fix which should not be too difficult to track down

Compile time evaluation of expressions (420)

This is an internal implementation design specification document, which essentially introduces a `value` option for the ASR

Additional Tasks

Continue down the road for `401` and also consider perhaps, more runtime library logic.

Misc Logs

Since running on a new project is roughly similar, I will only briefly go over the steps.

```
git clone git@gitlab.com:lfortran/examples/minidftatom.git
cd minidftatom
FC=gfortran cmake .
make -j$(nproc)
./F_atom_U
```

Which generates:

```
 Z=          92 N=        5500
E_tot=     -25658.417889
 state    E            occupancy
1s       -3689.355140    2.000
2s        -639.778728    2.000
2p        -619.108550    6.000
3s        -161.118073    2.000
3p        -150.978980    6.000
3d        -131.977358   10.000
4s         -40.528084    2.000
4p         -35.853321    6.000
4d         -27.123212   10.000
4f         -15.027460   14.000
5s          -8.824089    2.000
5p          -7.018092    6.000
5d          -3.866175   10.000
5f          -0.366543    3.000
6s          -1.325976    2.000
6p          -0.822538    6.000
6d          -0.143190    1.000
7s          -0.130948    2.000
 Print the first 10 values of the 1st and 2nd orbitals:
   1753.2892641883584        1753.2812175211841        1753.2731512531607        1753.2650645481160
1753.2569495840532        1753.2488203554758        1753.2406710770906        1753.2325014795395
1753.2243110250588        1753.2160993495806
   587.50702296395195        587.50432661503442        587.50162369317059        587.49891391928361
587.49619467268258        587.49347064291044        587.49073989144335        587.48800232804547
587.48525777252200        587.48250610287175
```

Parsing and AST Generation

The test for the parser is essentially to check if the output before and after formatting with `lfortran` still compiles to the same result. Note that because of the nature of this, we need to ensure `-ffree-line-length-none` is passed when working with the `gfortran` compiler.

```
cd minidftatom
# -i --> Inplace modification
for file in *.f90; do lfortran fmt -i $file; done
make clean
FC=gfortran cmake .
make -j$(nproc)
./F_atom_U
```

Dependency Tree

Before going on to the ASR, a dependency tree is required. As before, it is reproduced in excruciating detail in the issue, and represented by a pretty picture shown in Fig. mdftatomgit for this post.

```
pip install fortdepend
cd minidftatom/src
fortdepend -g -c
```

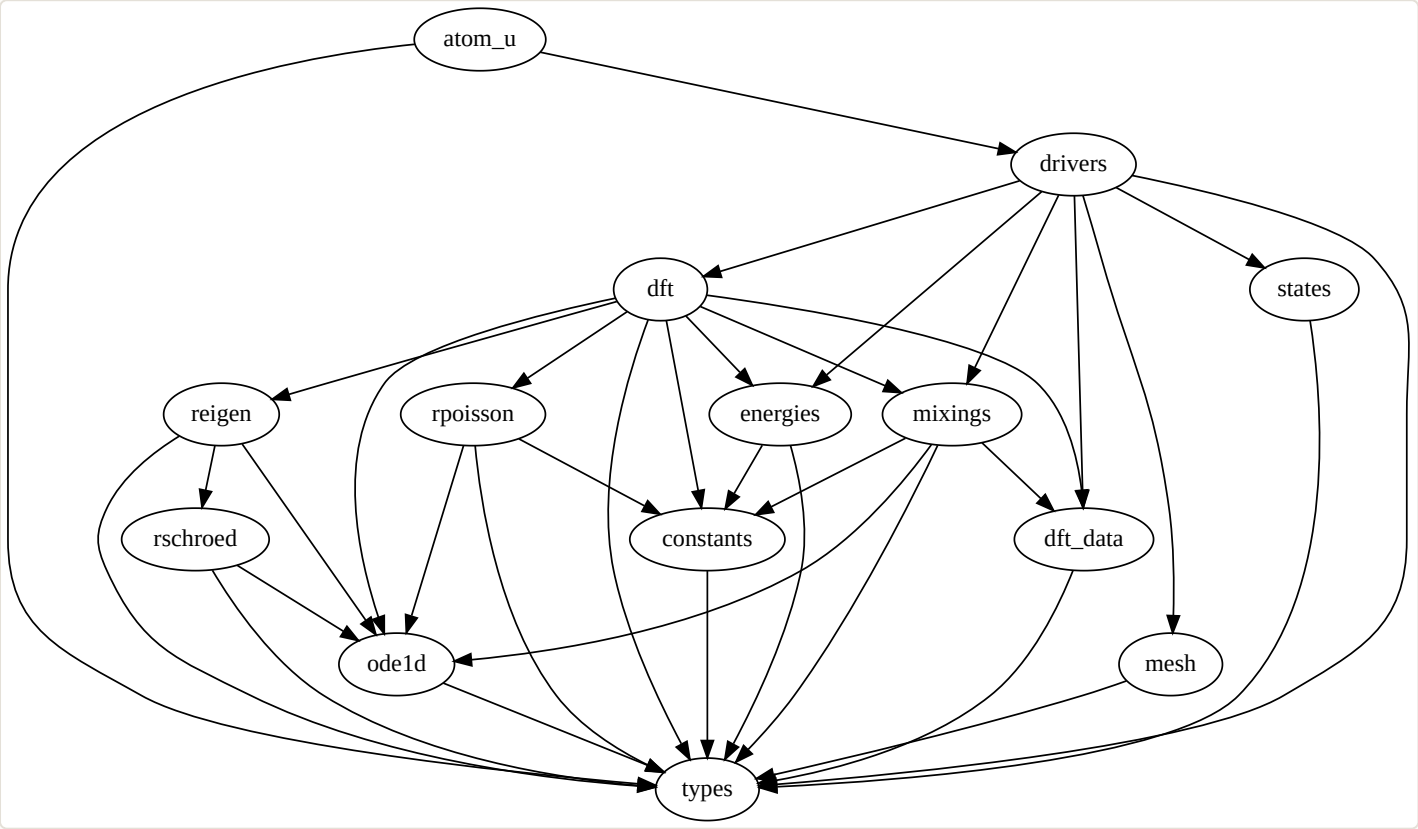

Figure 1: `minidftatom` compilation dependency graph

ASR Generation

I found a new favorite command, which I should have remembered while it was being implemented (by Thirumalai Shaktivel in 765); namely the `--indent` option which makes for a much more readable ASR output. Apart from that `git diff --color-words` works well enough as well.

Conclusions

The main work plan outcomes for the week were the new design specifications for the compile time evaluation of expressions. It is always an enriching experience to work with Ondrej directly (even over the ether) and this week felt more productive than most. I hope to keep the same momentum going. This will allow faster progress on the main issue of getting ASR generated for `minidftatom`.