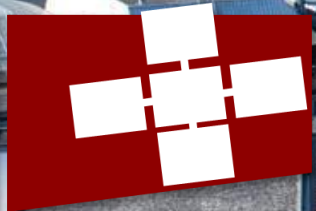





DaFLEX Progress report










Fortran merged!


master ▾
dace / dace / frontend / fortran /


t
Add file ▾
...

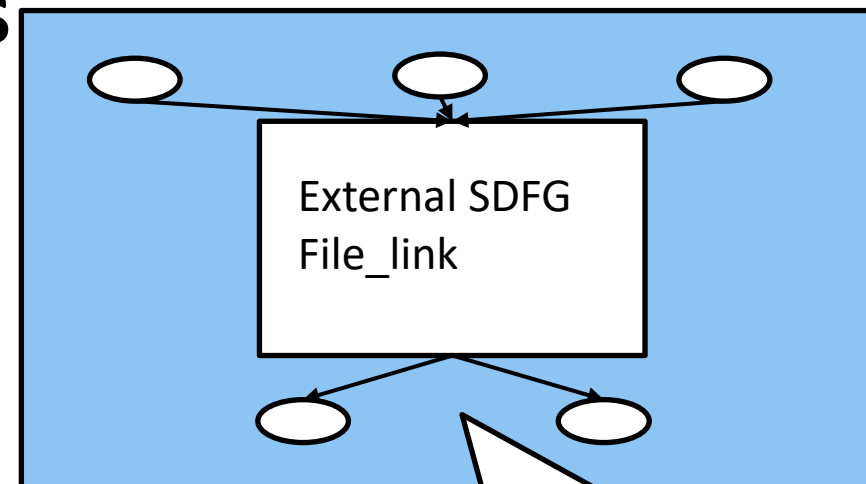

Alexandru Calotoiu
testing new options to get CI to work
×
3143c90 · last week
History

Name	Last commit message	Last commit date
 ..		
 <code>__init__.py</code>	adding tests for fortran frontend	3 weeks ago
 <code>ast_components.py</code>	testing new options to get CI to work	last week
 <code>ast_internal_classes.py</code>	initial commit with commented, formatted core of the fortran...	3 weeks ago
 <code>ast_transforms.py</code>	resolving comments	3 weeks ago
 <code>ast_utils.py</code>	resolving comments	3 weeks ago
 <code>fortran_parser.py</code>	testing new options to get CI to work	last week

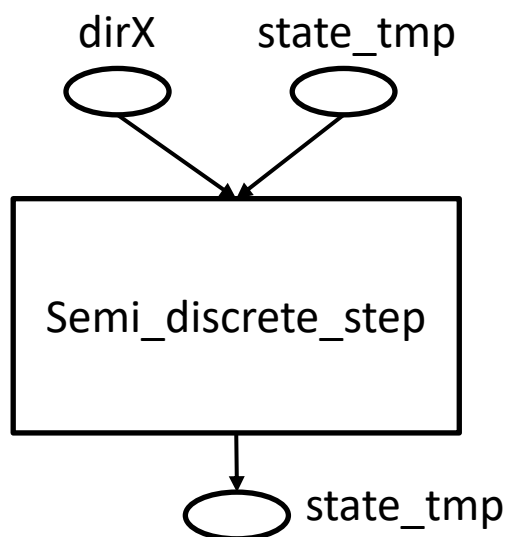
External nested SDFGs – In progress

■ Goal – New workflow:

- create SDFGs for each function first
- do local optimization
- load them together (potentially hierarchically)
- do global optimization



```
call semi_discrete_step( state , dirX , state_tmp)
```



Not used

Read

Read &
Written

“Slotting” the external SDFG in is not necessarily trivial:
 What if the NestedSDFG was simplified and no longer uses all arguments?
 We can leverage the lessons of the frontends!

ICON and Cloverleaf

- Adding more Fortran features
- Increasing robustness