Code and Community

(Parsl)

Ben Clifford

benc@hawaga.org.uk

code is a communal artwork…

… not a toxic byproduct of the grant funding process

what does the Parsl code-as-communal-artwork look like?

Python components

DataFlowKernel

SlurmProvider

python_app

Features

one task can depend on another

grid/cloud execution

https://github.com/Parsl/parsl/issues/2554

"Research" Code

Production Code

Research Code

Production Code

Cloud/grid

Workflow progress visualization

Globus file staging

Work Queue integrated with Parsl monitoring

SlurmProvider

Task dependencies

Research Code

Production Code

```
# this is a hack
```

long-standing issues that say "it is broken"

"When (not if) it hangs or breaks, just restart it"

race conditions

integration with other features

Auto testing: pytest, mypy, coverage

Documentation

error handling (vs happy path)

Multiple contributors

power users

fast response to issues [eg. technical sponsor]

serious review from other devs

Research Code

Production Code

Integrate with your main project

trying a new algorithm

one-off summer project

checkmarketing

exploratory prototyping for feedback and understanding

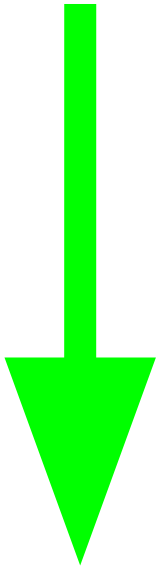
5 years

Research Code

Production Code

All of this has a valuable place in the Parsl community

How do you fit in? Issues



I'll hack around problem

I'll report it in the issue tracker

I'll fix it

A one character typo-fix PR in an error message is not "too small!"

Also covers new features, not just bugs

How do you fit in? New features work better with user collaboration

Friendly criticism:

Missing usecases - "this almost addresses my usecase… but not quite… so I can't use it at all"

Bug-finding & usability

eg MPI (PR #2905) - Yadu's talk

eg DESC collaboration

(Developing in a vacuum is quite lonely)

How do you fit in? Big Contributions (~ WorkQueueExecutor)

- trying a new algorithm
- Integrate with your main project
- one-off summer project
- checkmarketing
- exploratory prototyping for feedback and understanding

- integration with other features
- Auto testing: pytest, mypy, coverage
- error handling (vs happy path)
- Documentation
- Multiple contributors
- power users
- fast response to issues [eg. technical sponsor]
- serious review from other devs

Community Profile: Cooperative Computing Lab

Building "cooperative computing tools"

mostly nothing to do with Parsl - even in competition

CCTools is the package that holds Work Queue and Task Vine

This group contributed initial WQExecutor and Task Vine executor

This group very responsive to issues about WQ and TaskVine

sharing responsibility for the WQExecutor and TaskVineExecutor - for maintenance and for introducing new features

What we're doing - cleaner plugin points

Helping to hack on and extend Parsl: cleaner interfaces:

- inside the codebase (for hacking more easily)

- to external components (for developing/packaging/distributing separately - eg github repo and/or pip package)

Plugins in this parslfest - TaskVine (executor), ProxyStore (serialization), Falcon (file staging), Dragon (executor)

Executors, Providers, Launchers, File Staging, Serialization, Channels, Monitoring

How do you fit in? Technical sponsor / subject champion

Someone who cares about a particular component or feature or science domain

Knows how it works… fixes bugs…

https://github.com/Parsl/parsl/issues/2554

You can do this informally too…

Community Profile: Ryan Chard

from Globus Compute

Technical sponsor for PBS Provider

Only 2 PRs against PBS ever! … easy

Motivation: wants Globus Compute (via Parsl code) to submit to PBS

How do you fit in? Tutorials, documentation, demos, howtos, blogs, configs

Touches the code…

… but isn't PRs into our GitHub repo

eg. NERSC and ALCF - hosts configs and instructions - positive site-driven contribution

Community Profile: Logan Ward

Power user - materials science / molecules

Minor htex features (core allocation)

Video tutorials that cover Parsl and other stuff

Documentation

Tutorial notebooks

Honest feedback

Shares experience with other users in #parsl-help

What we're doing - untangling code

make the code more maintainable and more fun by making it simpler to understand and hack on

clearer types - type hints and automated checking (mypy, typeguarD)

deleting lots of code that has served its purpose

Conclusion: How do you fit in?

- Issues
- New features work better with user collaboration
- Big contributions
- Technical sponsor / subject champion
- Tutorials, documentations, demos, howtos, blogs, configs