Parsl – 2024 Open Community Calls

ATTENDEES	Josh Bryan Ben Clifford Kent Lee Yadu Babuji Kevin Hunter Kesling Daniel S. Katz Reid Mello Mark Hereld Chris Janidlo Sophie Bui	**DATE**	1/9/2024
		TIME	11 a.m. CST
		NEXT MEETING	1/23/24
MEETING PURPOSE	An open space to discuss topics surrounding activities such as user projects and needs, what's going on in the community, core development, code review, development infrastructure, software release, etc.		

AGENDA ITEM	MEETING NOTES
"Functional Influence on a Python-embedded Scientific Workflow Library" Lightning talk by Ben Clifford **Original Talk Abstract**: Parsl is a library for implementing scientific workflows in Python. Although we rarely present it this way, there are a lot of functional programming influences in the implementation of Parsl, and in this talk, I'll try to present both scientific workflows and Parsl from that angle. First, I'll introduce the very broad field of scientific workflow and the corner of it that we work in. Then I'll talk about our core execution model, which is based on futures with an applicative functor or monadic structure. I'll talk about how we override some Python syntax to get the behaviour we want while still mostly looking like Python, and I'll talk about some of the history	**Functional programming language and its applications.** ● Ben's experience as a Haskell programmer influenced his thoughts on how Parsl should work ● During his presentation at TUDelft, Ben introduced attendees to scientific computing – using computers to do science, not just programming ● Discussed workflow languages, execution vs. language, and cultural differences between workflow and build systems. **Using Python as a workflow programming language.** ● Workflow language evolved from Virtual Data language to Swift, then to Python as a library for increased functionality and community support. ● Discussed the challenges of formal reasoning in functional programming, particularly in Haskell and Agda, and how they are addressed through language features like futures and overloading function calls.

and human factors that drove us to consider Python as a host language. I'll also touch on some other Parsl topics that overlap with the functional community: for example, trying to understand equality between objects in different Python environments across time and space, especially functions; recording the provenance of generated scientific outputs for reproducibility; how our programming interface has helped other people build compositional libraries for "doing science" on top of Parsl.	● Discussed how the use of expressions in Python can facilitate composition and the creation of libraries, making it easier to package and document code. ● Highlighted the benefits of using syntax similar to functional programming languages, such as being able to run multiple parameters at once. **Python's handling of equality.** ● Discussed the challenges of working with equality in Python, particularly when dealing with objects and their representations. ● Highlighted the differences between object equality and serialization and type checking in Python. Ben gave this talk at TUDelft's Functional Programming Day 2024 event on January 5th in the Netherlands and received enthusiastic feedback from the audience about their interest in Parsl. ● He realized that they could improve the join apps specification in Python by adopting a more theoretical approach, which could make it easier for users to specify their behavior. ● Discussed the possibility of automatically discovering join apps without waiting for their results.
MPI Support State & Batch Applications Updates and discussion by Yadu Babuji ● PR: https://github.com/Parsl/parsl/pull/3016 ● Docs: https://github.com/Parsl/parsl/blob/mpi_experimental_3/docs/userguide/mpi_apps.rst	There's a PR request (https://github.com/Parsl/parsl/pull/3016) that was tested on ALCF's Polaris and NERSC's Perlmutter that basically lets you write batch applications, which are going to then launch MPI applications and have them run on an arbitrary set of nodes within a batch job ● Explained how to use the MPI prefix to launch MPI applications in a batch job, with the ability to specify resource requirements and schedule applications within the job. **PR and Documentation** ● Shows you how to write the application itself that uses the prefixes and specifies the resource specification, which is the way to tell the system how many resources this specific MPI

	incantation requires. The runtime itself is only supported through HTEX ● We have a new option called Enable MPI mode, and there is an MPI launcher that you can specify (allowing you to tell it like you want, such as MPI exec, etc.) and whatever you specify would then get passed in as the MPI prefix. ○ we want the user to specify what launcher they want to use rather than automatically try and figure out what the right launcher ● The example in documentation has been tested with real MPI applications on Polaris and Perlmutter. If anybody has applications that they would like to see tested for these infrastructures, Yadu would be happy to help.

CurveZMQ
Discussion led by Reid Mello
 	
Results Example:
`Polaris, with CurveZMQ, Py3.10, zmq==25.1.2`
`1k, no_data 960Tasks/s`
`1k, with ~1MB, 19.36 Tasks/s`
`10K, no_data, 1301 Tasks/s`
`Polaris, without CurveZMQ, Py3.10, zmq==25.1.2`
`1k, no_data, no_encryption, 1040Tasks/s`
`1k, with ~1MB, no_encryption, 115.9Tasks/s`
`10k, no_data, no_encryption, 1440Tasks/s`
`10k, with ~1MB, no_encryption, 111.5 Tasks/s`
	Reid shared that Globus Computes wants to encrypt our connections between various processes within Parsl. So this is specifically focused on the high throughput executor (HTEX).
	● ZMQ pipes that are unencrypted between the interchange in the manager and then there's other stuff like the interchange, to the executor, and then also the interchange to the monitoring hub
	● Reid's current conclusion – potentially the right path is curves and cubes. Curve is the standard and encryption methodology for ZMQ.
	○ Seems to be working. There were some concerns with the supportability of Curve because it requires certain cryptography libraries that need to be installed on different systems, but when I can tell it's just libsodium.
	● Preliminary testing results on Polaris and parameter, showed a 10x reduction in performance with large payloads.
	● mentioned that using ZMQ within Parsl may not be the best option due to performance concerns and the need for advanced routing capabilities.
	● Approaching this in a modular way to shut off encryption

Outreachy Internship ● Focused on increasing diversity and folks may not have a lot of technical expertise so they may need to start off with something basic ● Outreachy would provide some funding to cover costs to support the intern **Deadlines** ● Jan. 15 at 3pm UTC - Live Q&A for FOSS communities interested in mentoring Outreachy interns ● Jan. 22 at 3pm UTC - Live Q&A for people interested in becoming an Outreachy mentor ● Feb. 15 at 4pm UTC - last day for mentoring organizations to sign up ● Feb. 23 at 4pm UTC - last day for mentors to submit projects ● Mar. 4 at 4pm UTC - contribution period opens ● Apr. 2 at 4pm UTC - contribution period closes ● May 1 at 4pm UTC - interns announced ● May 27, 2024 to Aug. 23, 2024 - internship period	Last year we didn't get any interns through Outreachy and didn't get feedback or advice on what to do to increase our chances of securing interns. ● This is an opportunity for an intern to help with Parsl-related projects without having to spend funds. However, technical experience and skills vary – leaning towards beginner/basic ● Sophie will be introduced to the Outreachy representative to connect and figure out tips for success **Potential Challenges** ● Interns may not have the technical experience to work on complex jobs – mentors will need to be prepared to walk them through the basics ○ Mentors will need to be able to meet with the intern 5 hours/week ○ Will need to provide a project description ● Technical skills and experience needed for more complex jobs can be intimidating a deter applicants **Potential Projects** Based on our experience – we may need to simplify our intern needs and start with the basics. Below is a space for project suggestions ● Updating Parsl website ● Updating Documentation
Python 3.12 Test Run & Removal of Python 3.8 ● Ongoing process that keeps happening	Ben C. shared an announcement via #parsl-hackers channel on 12/27/23 to inform the community that: ● Python 3.12 test run is now compulsory in our GitHub test environment – he edited the GitHub settings to make that the case. ● The next Python version change will be the removal of Python 3.8 at end-of-life sometime later this year ○ When we start getting to use Python 3.9 language

	features
Windows Support Discussion ● GitHub Issue #1878	There's a persistent small community interested in us supporting Windows, but we don't have a dedicated technical sponsor who will help maintain it. ● The main thing that's making it not work is the multiprocessing fork which doesn't exist on Windows, Unix, and recommended to not use it on MacOS ● We want to find someone who is passionate about it ○ Should we put a specific thing in the #general Slack channel to put out a call – looking for someone who is actually interested in testing it out fully and not someone who won't have time to do it ○ HPE Dragon does not work on Windows at all or relies on fork – more interested in having Posix support ○ Ben just wants the basic local things to be able to run ○ Josh shared that their support will most likely run WSL (a Linux kernel alongside the Mac one) ● A lot of work we're doing is leading to supporting Windows eventually but there's not a dedicated person currently working on this
Projects that Use Parsl ● Projects That Utilize Parsl	We'd like to feature your project(s) that use Parsl on our site – please share info about your work in our open Google Doc. We're asking for the following info about your project: ● Project Name/Title ● Website Link ● Contact ● Domain/Discipline/Field ● Short Project Description
Miscellaneous	● Include call-in number for our open community calls in case folks can't join Zoom directly through app