

Welcome! Parsl and funcX Fest 2022

Kyle Chard

September 13-14, 2022

Parsl Code of Conduct

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free and bullying-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members
- Respecting the work of others by recognizing acknowledgment/citation requests of original authors
- Being explicit about how we want our own work to be cited or acknowledged

This meeting will follow the same Code of Conduct.

Issues: contact Dan Katz (dskatz@illinois.edu)

https://github.com/Parsl/parsl/blob/master/CODE_OF_CONDUCT.md



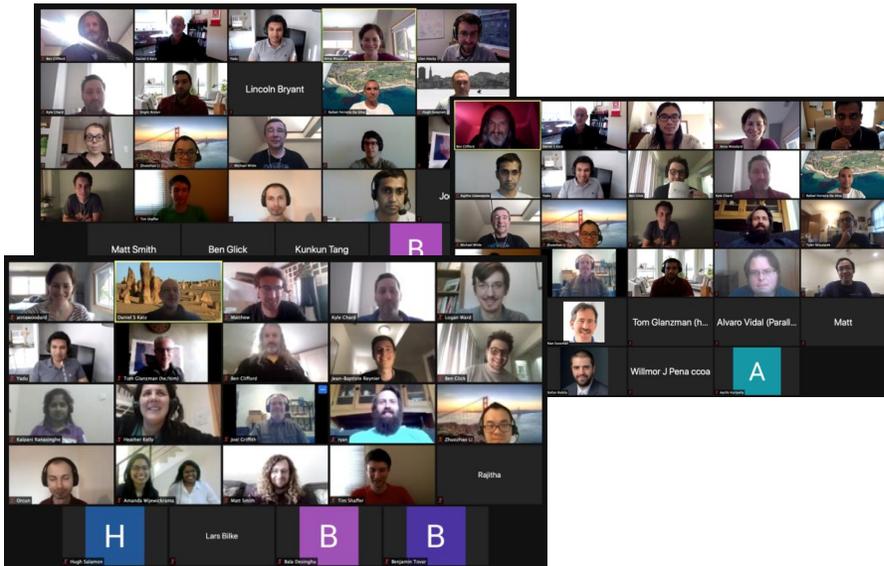
Parslfest 2019



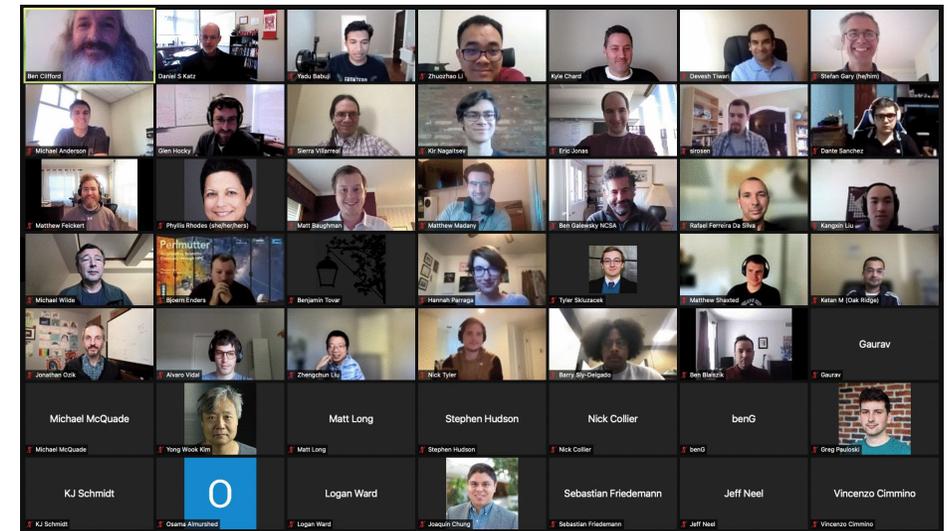
Parsl & funcX Fest 2022



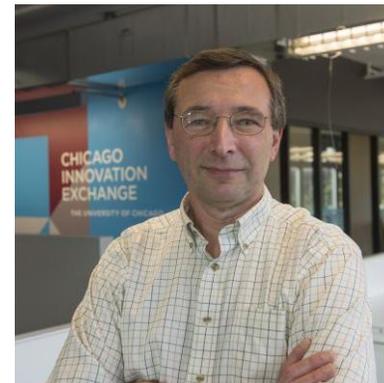
ParslFest 2020



Parsl & funcX Fest 2021



Introducing the Parsl and funcX teams



And many others...

Anna Woodard
Stephen Rosen
Kir Nagaitsev

Logan Ward
Theodore Ando

Thank you funding agencies and project partners



2209919 (UChicago)
2209920 (UIUC)
1550588 (UChicago/UIUC)
1550476 (Notre Dame),
1550475 (Colorado State)
1550562 (Northern Arizona)
1550528 (College of New Jersey)



2004894 (UChicago)
2004932 (UIUC)

Argonne LDRDs

- 2022-0230 Productive Exascale Analysis Workflows for Numerical Cosmology
- 2021-0152 Creating a Robust and Scalable Framework for On-demand Analysis and AI-based Experiment Steering
- 2019-0217 Establishing a Usable, Scalable, and Reproducible Computational Ecosystem for Dark Energy Science

LSST Dark Energy Science Collaboration (DESC)

DOE ECP PRJ1008564 ExaWorks project

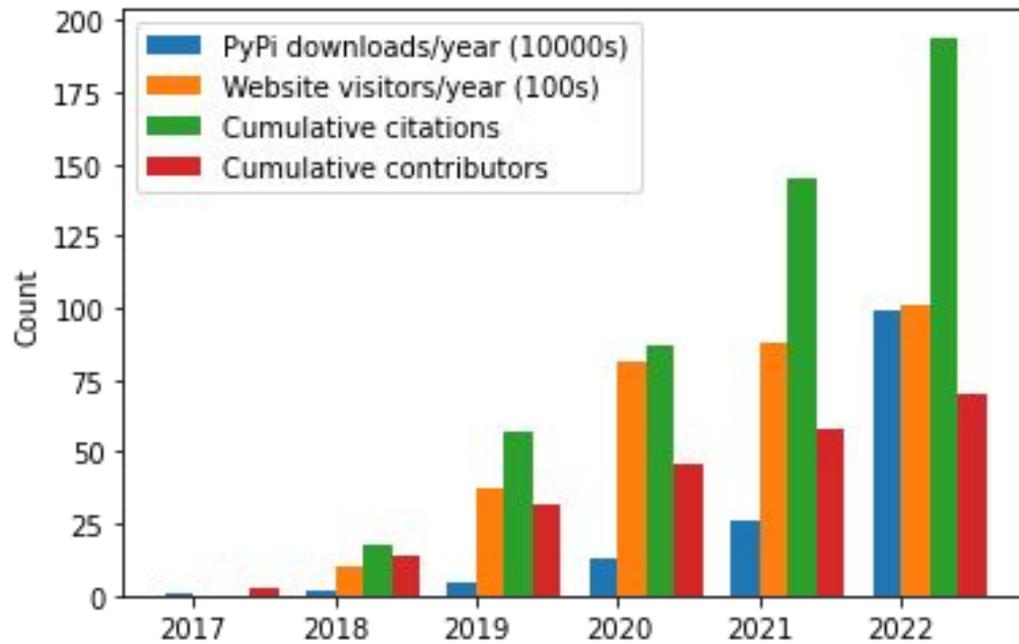
DOE DE-NA0003963 Center for Exascale-enabled Scramjet Design (CEESD)

Discovery Partners Institute (DPI): Airborne-Satellite-AI-HPC integrative framework (ASAI)

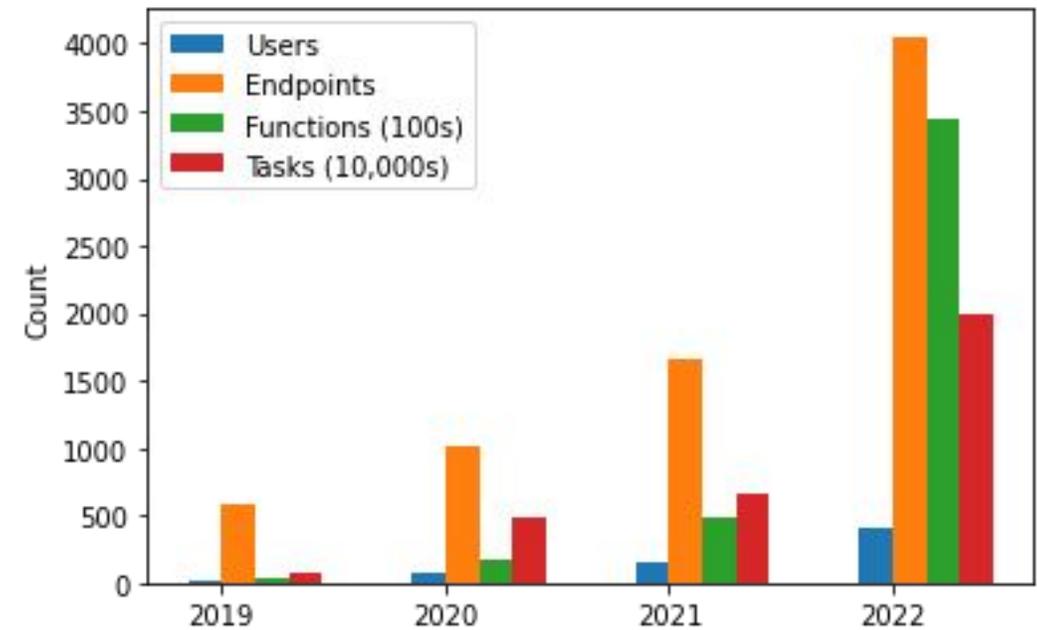


U.S. DEPARTMENT OF
ENERGY

Parsl and funcX continue to grow!



70 contributors, >1.4M PyPI downloads



>20M tasks, >340K functions, >4000 endpoints

Goals for this meeting

- Meet the Parsl and funcX communities
- Learn about how people are using Parsl & funcX
- Find out how to contribute to Parsl/funcX
 - Help us develop and better engage the Parsl & funcX community
- Provide feedback to the Parsl/funcX team
 - Help us prioritize development activities
 - Help us identify shortcomings
 - Understand what needs work
- Form new collaborations

Brief Introduction to Parsl and funcX

Motivation: The modern research computing landscape

Software is increasingly *assembled* rather than written

- High-level language to integrate components from many sources

Parallel and distributed computing is ubiquitous

- Increasing data sizes & plateauing sequential processing

Resources are increasingly heterogeneous (distributed)

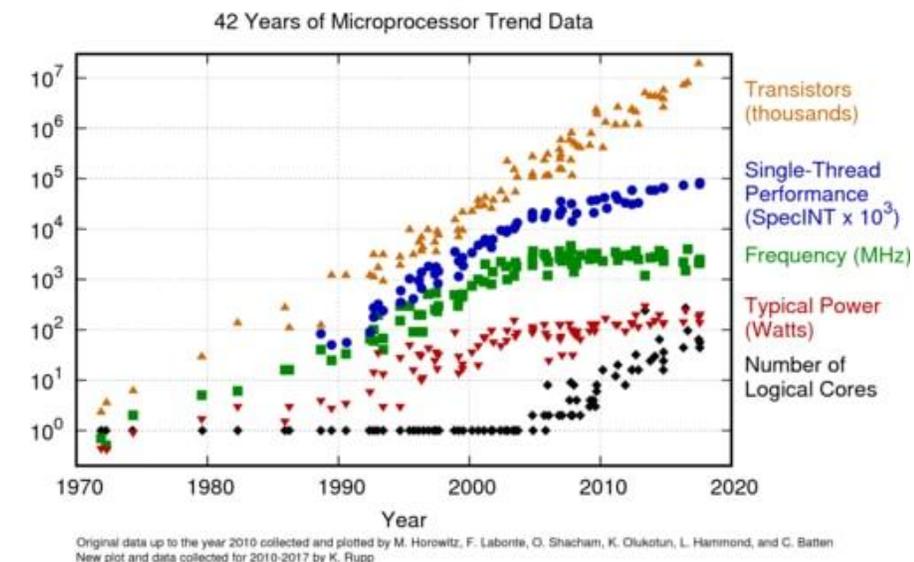
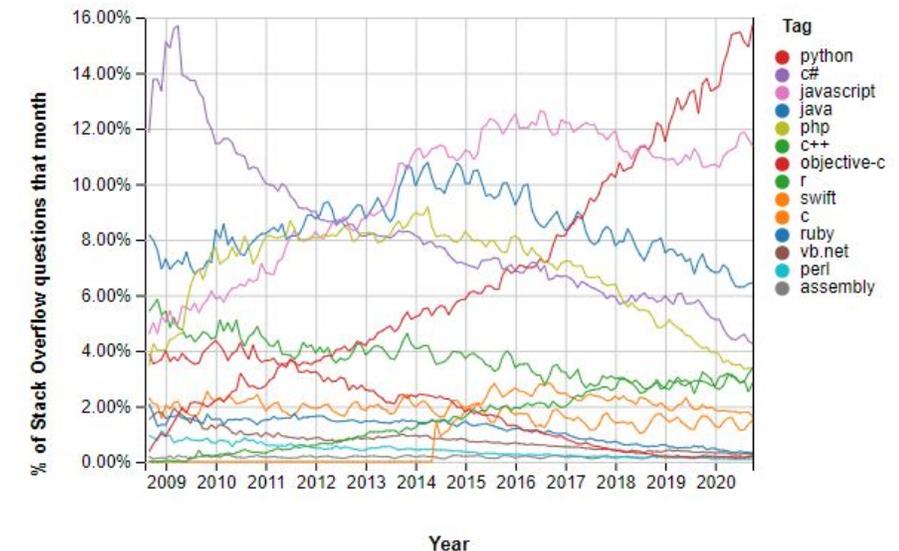
- Application components best run in different places

Python (& SciPy ecosystem) de facto standard language

- Libraries, tools, Jupyter, etc.

Parsl allows for the natural expression of parallelism in Python

funcX enables fire-and-forget remote and distributed execution of Python functions



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labadie, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp



ParSl: a parallel programming library for Python

Apps define opportunities for parallelism

- Python apps call Python functions
- Bash apps call external applications

Apps return “futures”: a proxy for a result that might not yet be available

Apps run concurrently respecting data dependencies. Natural parallel programming!

ParSl scripts are independent of where they run. Write once run anywhere!

```
pip install parsl
```

```
@python_app
def hello():
    return 'Hello World!'

print(hello().result())
```

Hello World!



```
@bash_app
def echo_hello(stdout='echo-hello.stdout'):
    return 'echo "Hello World!"'

echo_hello().result()

with open('echo-hello.stdout', 'r') as f:
    print(f.read())
```

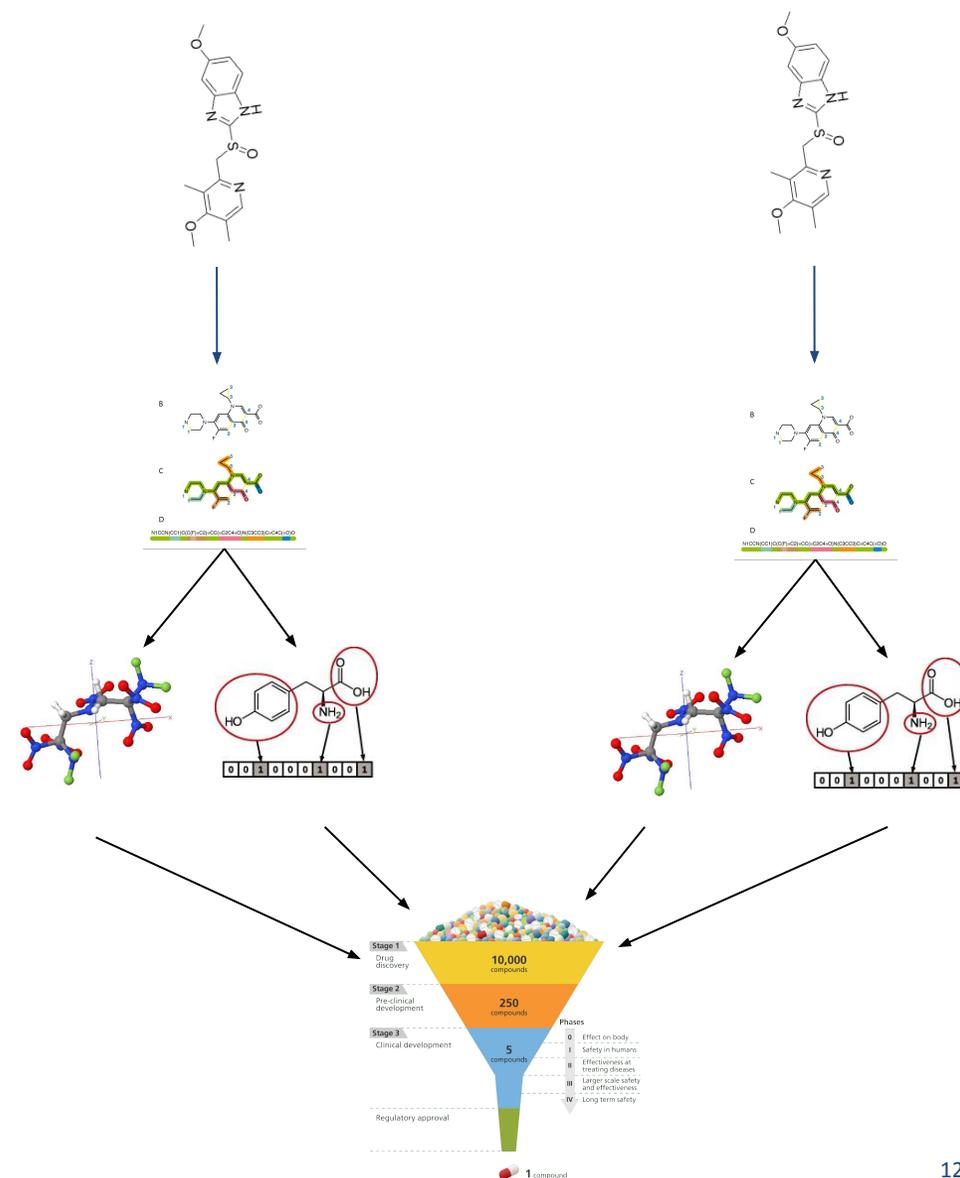
Hello World!



Try ParSl: <https://parsl-project.org/binder>

Parsl's dataflow programming model delivers intuitive parallel programming

- Programming paradigm in which program is assembled as a directed graph of data flowing between tasks
- Intuitive way to think about parallelism (tasks run independently when data slice ready)
- Parsl's dataflow model allows data to be passed between Apps
 - Python types and objects
 - Files (local or via HTTP, FTP, or Globus)



Parsl highlights for 2022

Award Abstract # 2209920

Collaborative Research: Sustainability: A Community-Centered Approach for Supporting and Sustaining Parsl

NSF Org: [OAC](#)
[Office of Advanced Cyberinfrastructure \(OAC\)](#)

Awardee: UNIVERSITY OF ILLINOIS

Award Abstract # 2209919

Collaborative Research: Sustainability: A Community-Centered Approach for Supporting and Sustaining Parsl

NSF Org: [OAC](#)
[Office of Advanced Cyberinfrastructure \(OAC\)](#)

Awardee: UNIVERSITY OF CHICAGO

Initial Amendment Date: July 20, 2022

Latest Amendment Date: July 20, 2022

Award Number: 2209919

Award Instrument: Standard Grant

Funding focuses on sustainability

- Governance
- Maintenance and testing
- Building the Parsl community

→ See Dan Katz's talk at 2:30 today ("Parsl Sustainability")

Code highlights

- 1.2 release (June 2022)
- WQEX production
- Monitoring updates
- Prototype Flux executor
- Retry scoring
- Bug fixes, documentation, testing

Collaboration

- Ongoing integrations with WQ, RCT, Flux
- Working with ExaWorks on testing
- Prototype integration with ExaWorks PSI/J

Community

- 70 contributors
- Biweekly open developer meeting (see #parsl-hackers on slack)



*func*x

The image features the text "funcx" in a dark blue, italicized sans-serif font. The "x" is a large, stylized graphic composed of numerous small, overlapping hexagons in a variety of colors including purple, blue, cyan, green, yellow, orange, red, and pink. Two thin, light gray lines intersect at the center of the "x" to form its structure.

funcX: Federated Function-as-a-Service



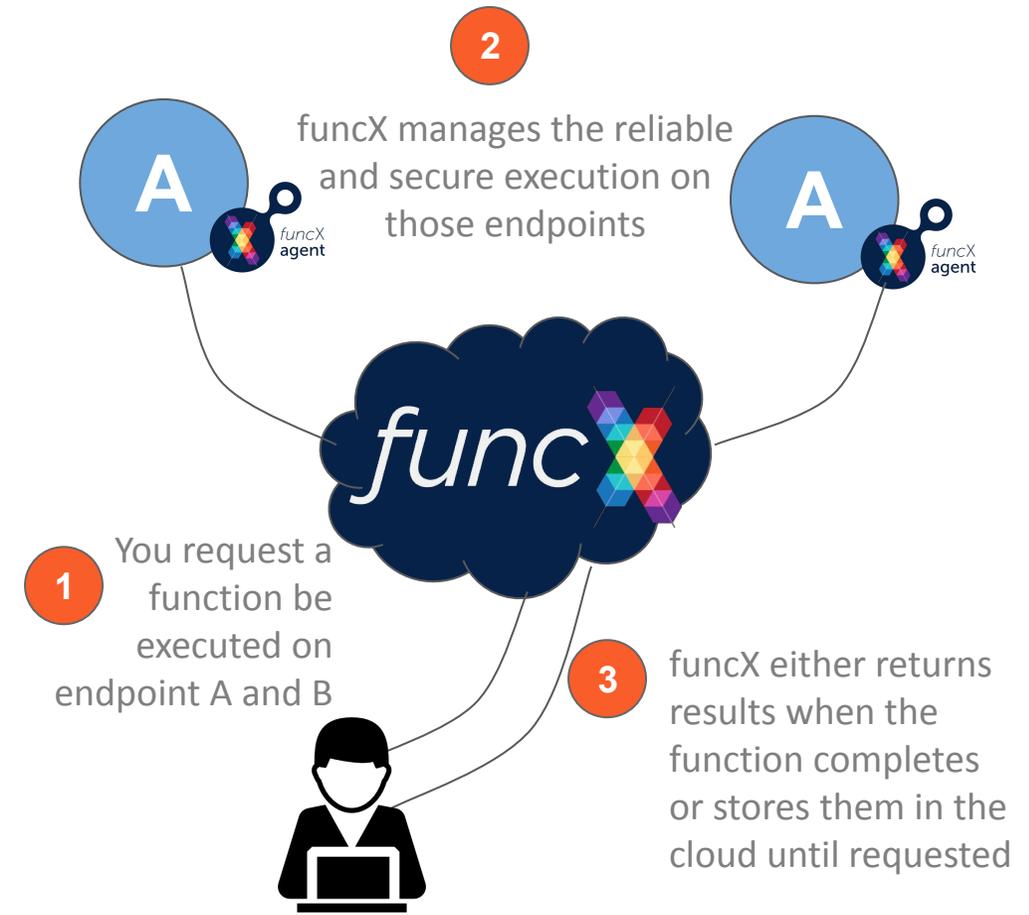
One of the biggest challenges with Parsl is the need to coordinate work over several resources

- Where do you host the Parsl process? how do you reliably connect to remote resources? how do you manage data between them?

funcX services Parsl:

- funcX endpoints (building on Parsl) enable scalable execution of tasks
- funcX service provides reliable management of tasks across endpoints

funcX leverages the Globus security model



Try funcX: <https://funcx.org/binder>

Transform laptops, clusters, clouds into function serving endpoints



- Python-based agent (pip or Conda) installable in user space
- Elastically provisions resources from local, cluster, kubernetes, or cloud system (using Parsl)
- Manages concurrent execution on provisioned resources
- Optionally manages execution in containers
- Share endpoints with collaborators

```
$ pip install funcx-endpoint  
  
$ funcx-endpoint configure myep  
  
$ funcx-endpoint start myep
```



Execute tasks on any accessible endpoint

Choose a function, endpoint ID, and input arguments

```
from funcx.sdk.client import FuncXClient
from funcx.sdk.executor import FuncXExecutor

fx = FuncXExecutor(FuncXClient())

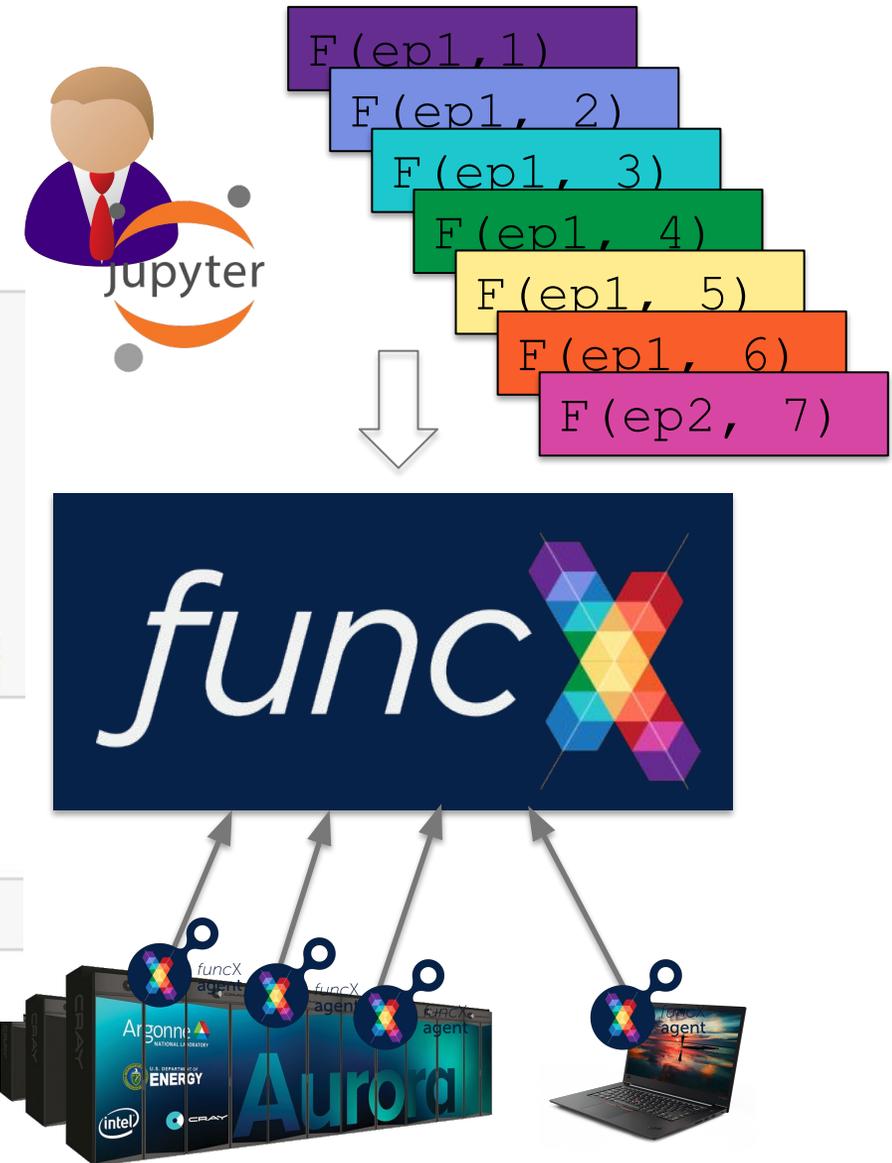
def hello():
    return 'Hello World'

future = fx.submit(hello, endpoint_id='4b116d3c-1703-4f8f-9f6f-39921e5864df')
```

Asynchronously retrieve results

```
print(future.result())

Hello World
```



funcX highlights for 2022

New funcX architecture based on cloud-hosted RabbitMQ queues

- Horizontal scaling of cloud infrastructure, more robust task processing, improved security model, and improved endpoint performance

funcX executor interface (implementing Python's concurrent.futures executor interface)

New S3-based task storage model

Update to Globus SDK 3 for improved token management

New internal task lifecycle and logging model

Workflows community events

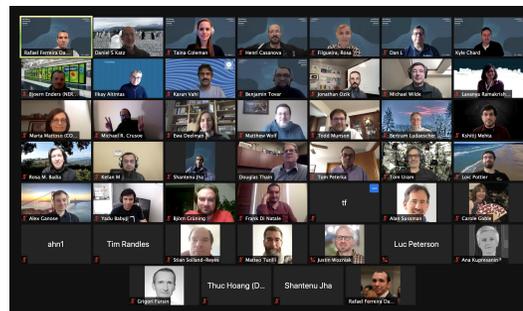


Supercomputing 22 Birds of a Feather “Community Roadmap for HPC and AI Scientific Workflows Research and Development”

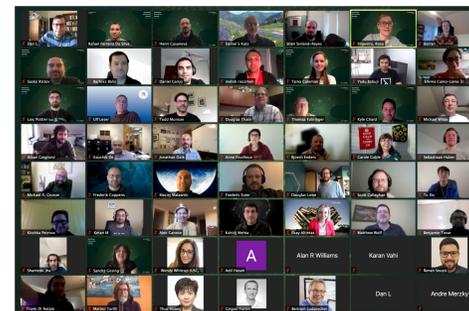
- <https://workflows.community/bof/sc22/>

4th workflows community summit will be held in the fall:

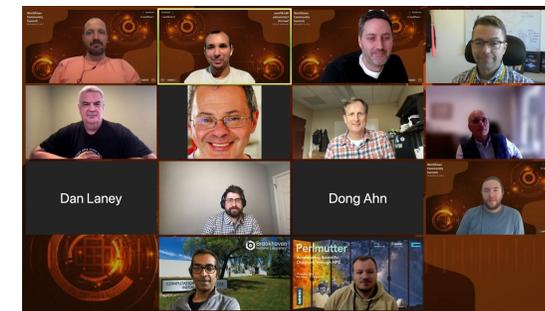
- <https://workflows.community/summits>



First Workflows Community Summit:
45 participants, 27+ workflow systems



Second Workflows Community Summit:
75 participants



Third Workflows Community Summit:
9 participants, 8 facilities/centers (ALCF, OLCF, NERSC, LC, BNL, PSC, NREL, NCSA)

Parsl & funcX Fest 2022

Tuesday September 13

1:00 pm- Welcome!

1:20 pm - Session 1 (Chair: Daniel S. Katz)

2:30 pm - Tech talks: Dan Katz, Alvaro Vidal Torreira

3:00 pm - Break

3:30 pm - Session 2 (Chair: Kevin Hunter Kesling)

4:30 pm - Tech talks: Yadu Babuji, Ben Galewsky

5:00 pm - Day 1 Closing

Wednesday September 14

8:00 am - Breakfast

9:00 am - Session 3 (Chair: Yadu Babuji)

10:15 am - Tech Talk: Ben Clifford

10:30 am - Break

11:00 am - Session 4 (Chair: Kyle Chard)

12:00 - Closing



<https://parsl-project.org/parslfest2022.html>

