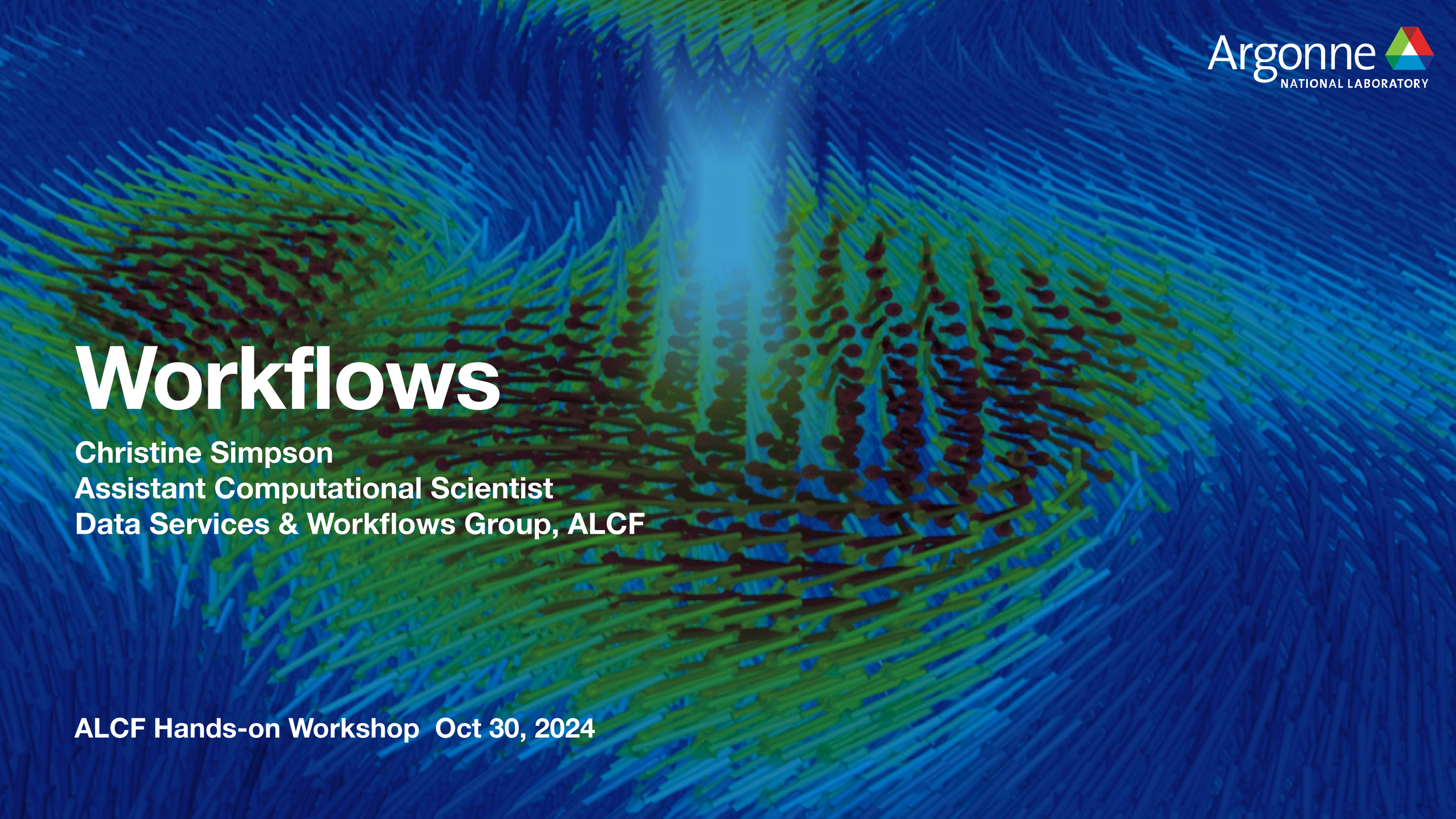


Workflows

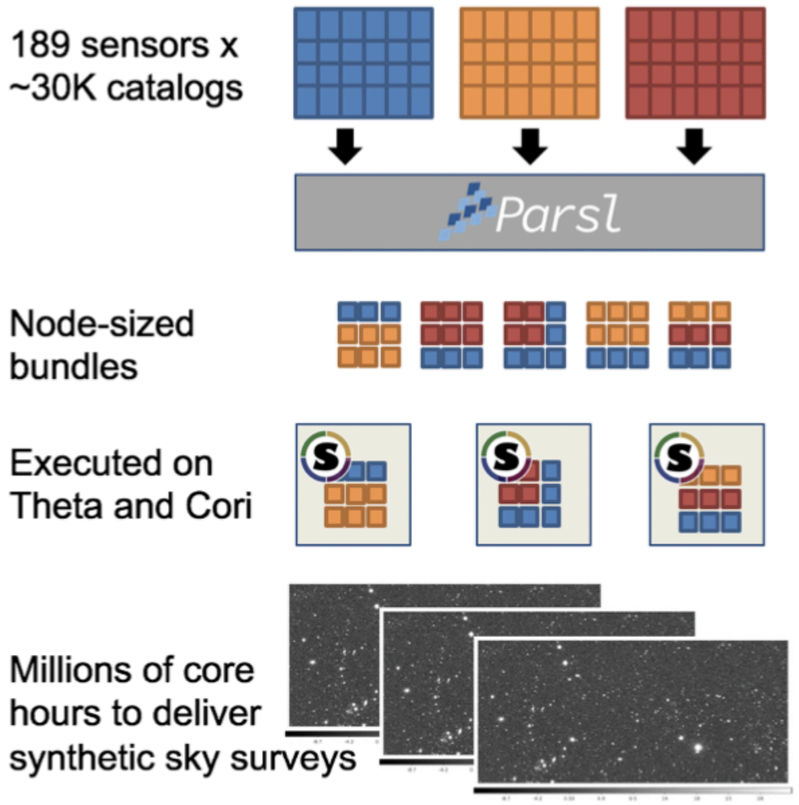
Christine Simpson
Assistant Computational Scientist
Data Services & Workflows Group, ALCF

ALCF Hands-on Workshop Oct 30, 2024

What is a workflow tool?

Why should you consider using one?

- A workflow tool is a piece of software that orchestrates the execution of large numbers of tasks on compute resources, handling dependencies, data flows, and errors/timeouts
- What a workflow tool can do for your workload:
  - Run many tasks concurrently and/or one after another asynchronously across one or many batch jobs
  - Manage task dependencies
  - Automate error handling and restarts of tasks
  - Manage data movement into/out of the file system needed for tasks
- ALCF and ANL have developed tools at the lab and in partnership with Globus Labs that run effectively on our machines



Villarreal et al. “Extreme Scale Survey Simulation with Python Workflows.” Proceeding for eScience 2021

Workflow tools at ALCF

Balsam, Parsl & Globus Compute

- Today, we will cover 2 tools commonly used at the facility for managing workflows
  - **Balsam** - developed at ALCF; a good choice for deploying multi-node MPI jobs and users looking for a database model; can also execute tasks remotely
  - **Parsl** - developed by Globus Labs, UChicago and ANL; a good choice for locally executed, high throughput workflows executing tasks on single cores or nodes
- If time permits: **Globus Compute** - developed by Globus Labs; a good choice for remote execution of tasks
- There are many tools out there! If you are interested in tools we don't cover today, please come talk to us and we can work with you

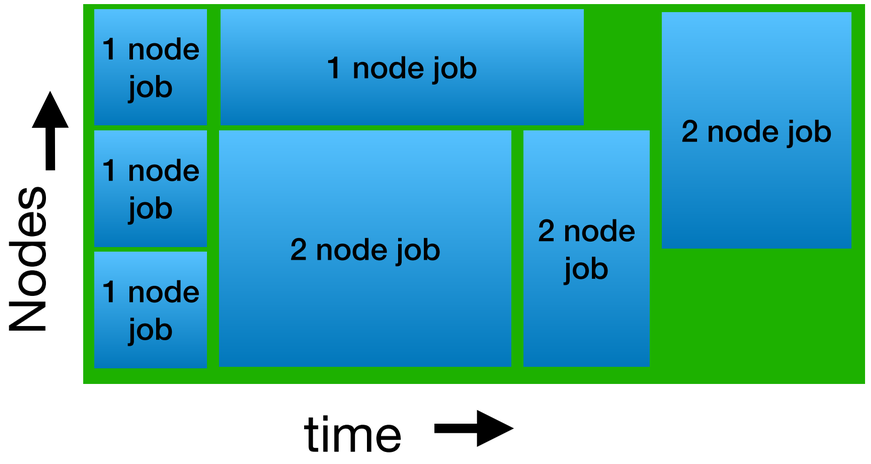
Balsam Workflow Management Tool

A unified platform to manage high-throughput workflows across the HPC landscape

- Balsam was developed at ALCF and is used for deploying workflows on DOE HPC machines
- Balsam uses a **database model**, applications and tasks are stored in a centralized database that tracks the progress of tasks, called jobs
- Install with pip, has a **Python API** and **command line interface**
- Can execute external apps and native python apps
- Optimized for running **MPI** applications
- Centralized server allows for inter-machine workflows
- Database hosted for the user at ALCF (requires ALCF account)
- Supported configurations for ALCF machines, and machines at NERSC & OLCF

To use Balsam, request access to Balsam server by email: support@alcf.anl.gov or drop a request in the #technical-q-a channel

3 Node Batch Job running 7 Balsam jobs requiring different run times and node numbers



Balsam Apps and Jobs

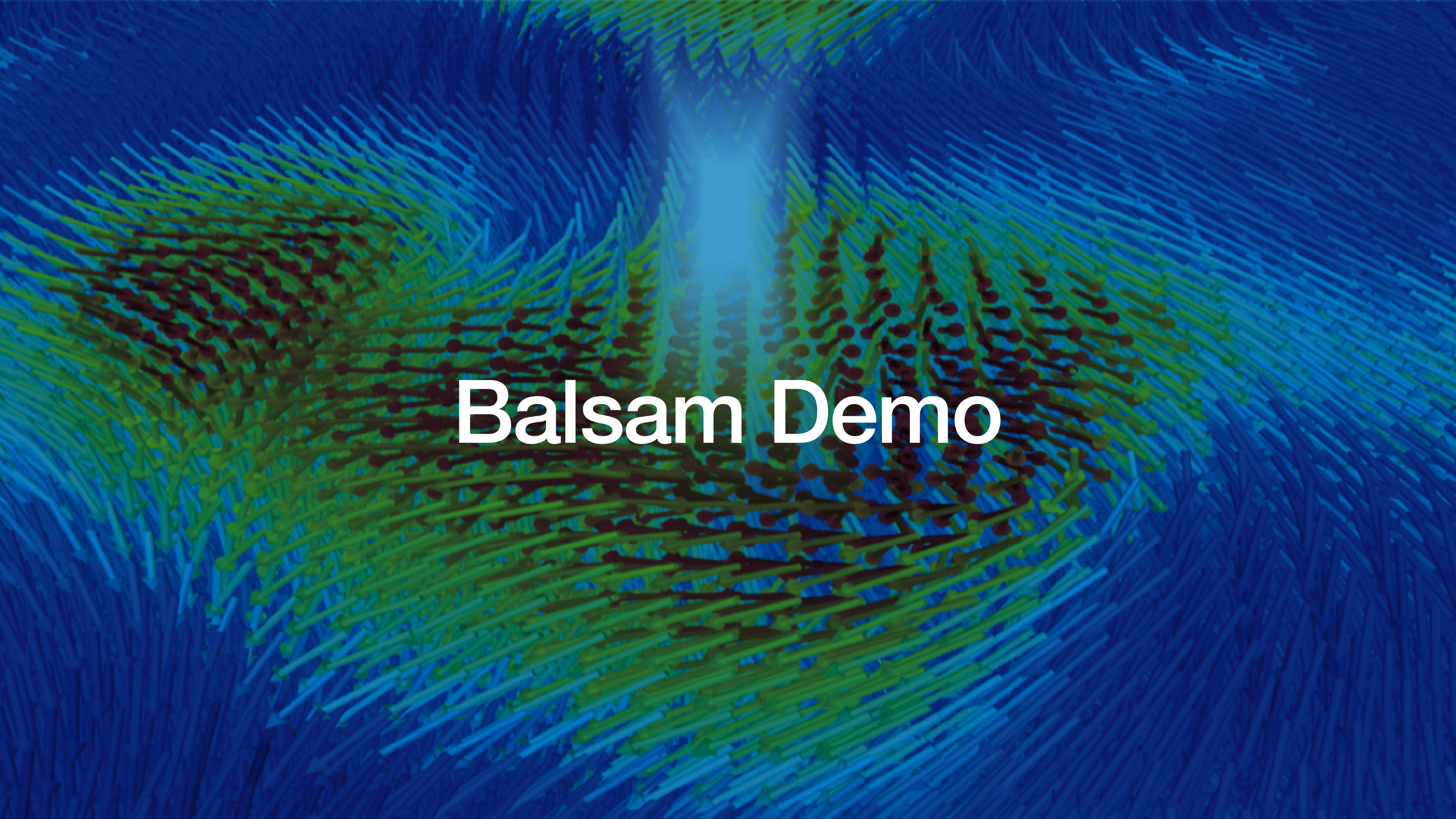
How to manage work

Define applications as Python classes, e.g.:

```
from balsam.api import ApplicationDefinition, Job, BatchJob

class Lammps(ApplicationDefinition):

    site = “polaris_tutorial"

    def shell_preamble(self):
        return f’export PATH=/path/to/lmp:$PATH’

    command_template = 'lmp -in /path/to/input.in -var tinit
{{tinit}}'

Lammps.sync()
```

Query, track, and execute Jobs from the command line (or through python API), e.g.:

```
> balsam job ls
ID         Site                App      Workdir      State          Tags
34017534   polaris_tutorial    Lammps   lat_1/run0   PREPROCESSED   {'case': 'lattice_1'}
34017535   polaris_tutorial    Lammps   lat_1/run1   PREPROCESSED   {'case': 'lattice_1'}
34017536   polaris_tutorial    Lammps   lat_2/run0   JOB_FINISHED   {'case': 'lattice_2'}
34017537   polaris_tutorial    Lammps   lat_2/run1   JOB_FINISHED   {'case': 'lattice_2'}
34017538   polaris_tutorial    Vasp     vasp/test0   PREPROCESSED   {'compound': 'test'}
34017539   polaris_tutorial    Vasp     vasp/test1   PREPROCESSED   {'compound': 'test'}
```


Balsam Demo

Parsl

A parallel programming library for Python

- Simple installation with pip
- Apps define how to run tasks
  - Python apps call python functions
  - Bash apps call external applications
- Workflow contained within memory (no database)
- Configuration (assignment of tasks to hardware) set by user, separate from workflow logic and application definitions
- Apps return futures: a proxy for a result that might not yet be available
- Apps run concurrently, respecting dependencies
- Community of 70+ developers, several at UChicago & ANL, part of Globus Labs

```
@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!
```

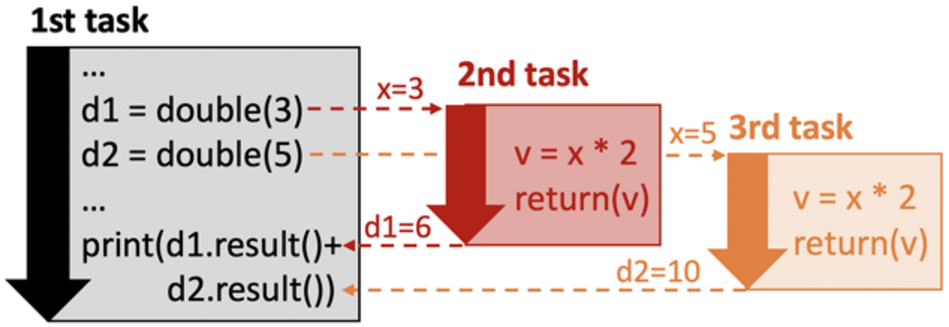
Parsl Apps and Futures

How tasks are made and linked

- Parsl extends the Python `concurrent.futures` module
- Tasks are created by invoking apps that return an AppFuture
- Task dependencies can be created by passing the AppFuture from one task to another

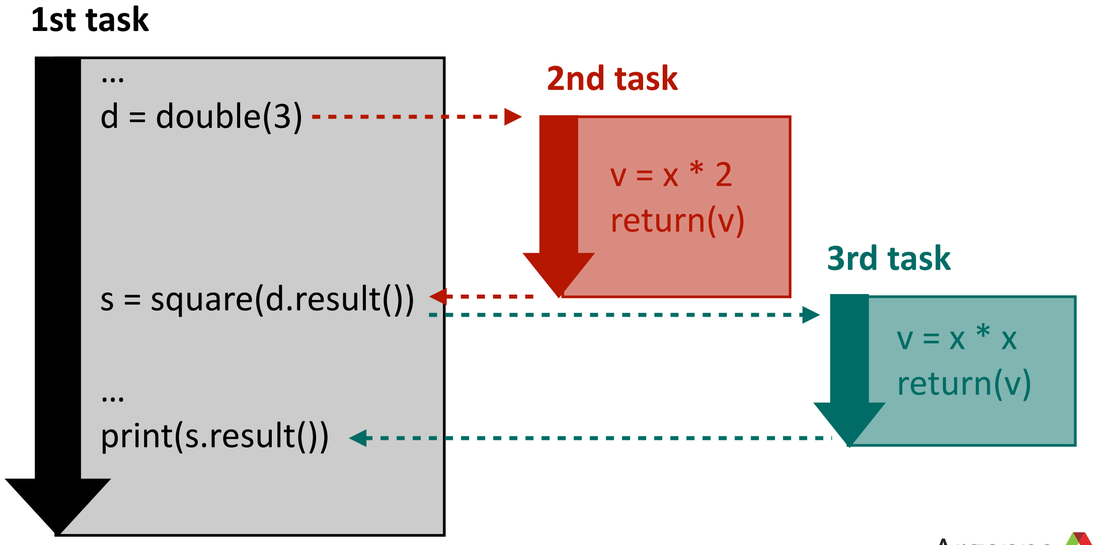
Concurrent Tasks

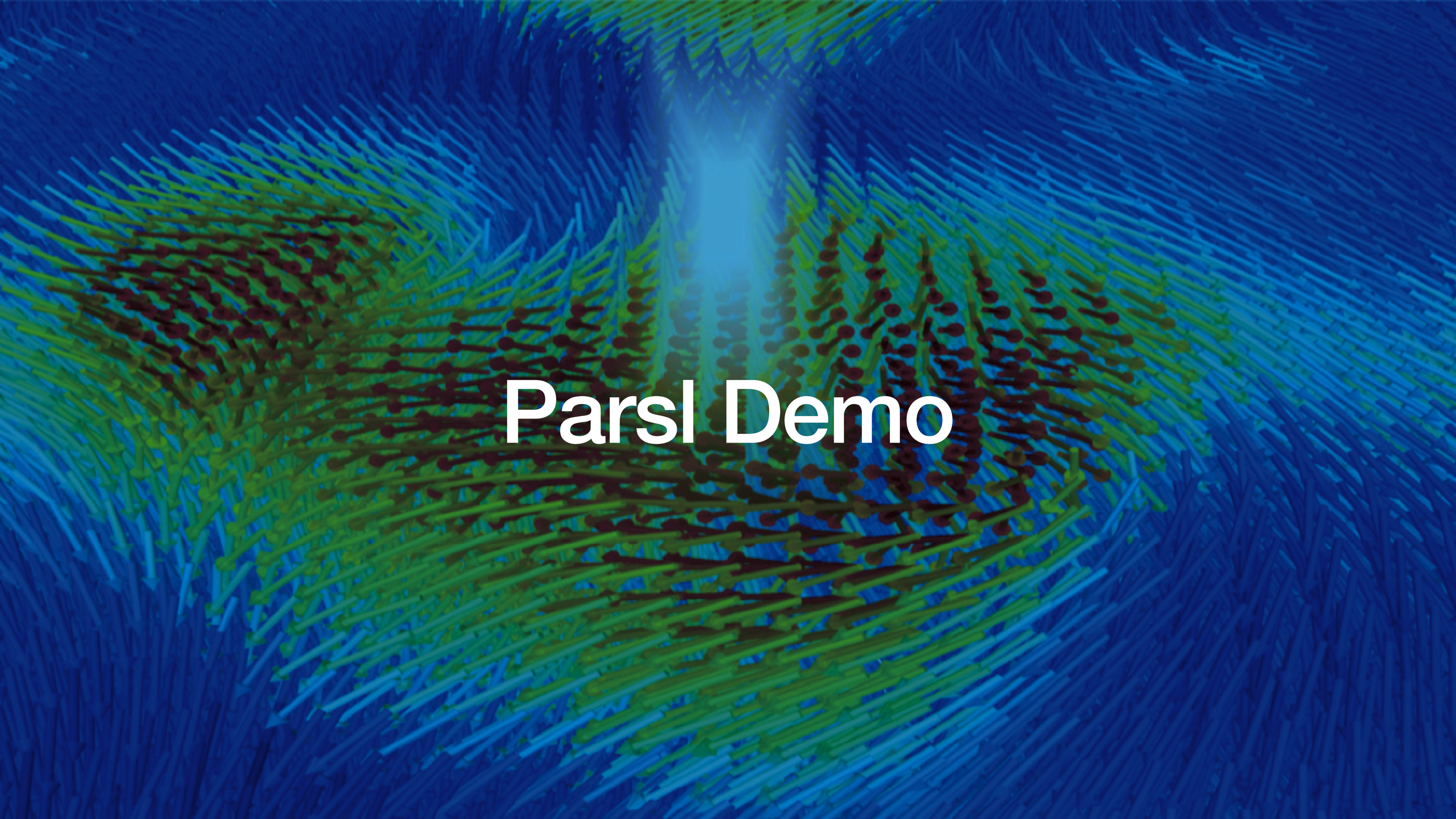
```
@python_app
def double(x):
  v = x * 2
  return v
```



Dependent Tasks

```
@python_app
def double(x):
  v = x * 2
  return v

@python_app
def square(x):
  v = x * x
  return v
```

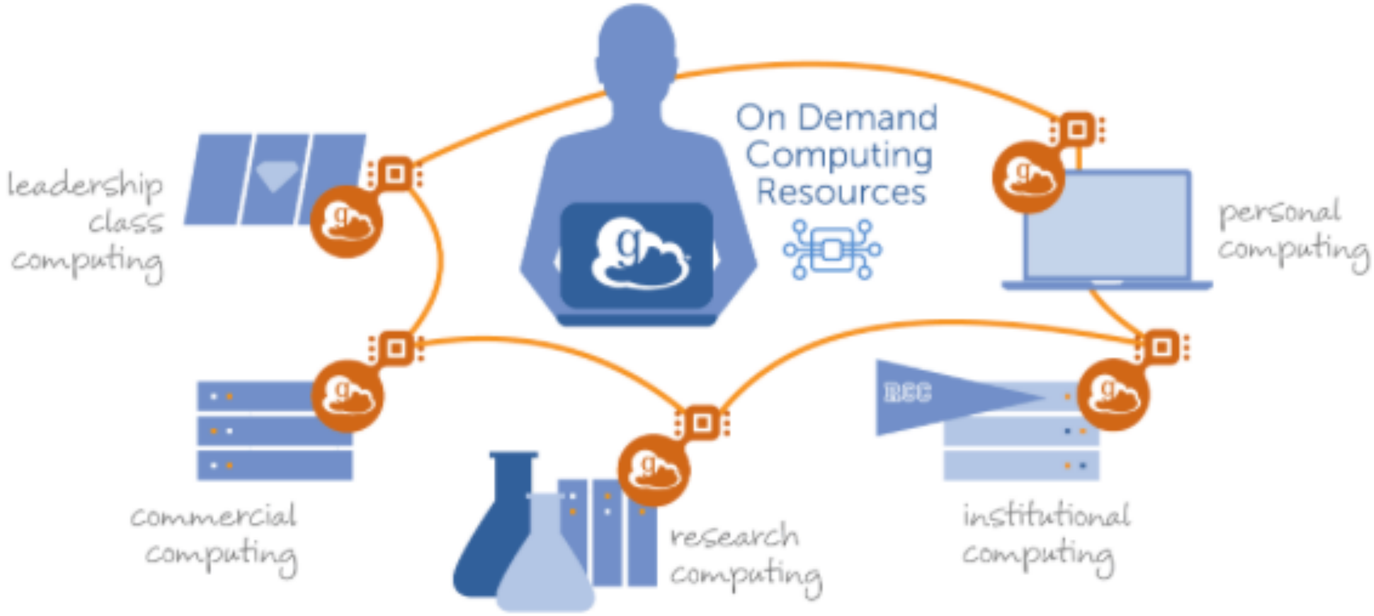



Parsl Demo

Globus Compute

“fire-and-forget” execution of tasks

- Allows users to launch applications remotely from laptop, other machine, etc.
- Built on top of Parsl, similar configuration
- Allows users to launch applications remotely from laptop, external machine, anywhere
- Requires setup of a Compute Endpoint on the target machine (e.g. Polaris) beforehand
- Globus Compute functions can be integrated with data transfers with Globus Flows



Different Tools offer Different Capabilities

- Remote execution of tasks: Balsam, Globus Compute
- Purely local execution of tasks: Parsl
- Multi-node MPI tasks: Balsam does this well, Parsl & Globus Compute in development
- Database: Balsam uses a database, Parsl keeps tasks in memory
- Portability: All can run anywhere (however, Balsam requires an ALCF account)
- Data Transfers: All have integration with Globus data transfers
- AI/ML steering tools: Many tools used at ALCF for this including DeepHyper, Colmena, libEnsemble & SmartSim that leverage workflow tools like Parsl & Balsam.

More Resources

- **Parsl**
  - docs: https://parsl.readthedocs.io/en/stable/
  - github: https://github.com/Parsl/parsl
  - slack: https://parsl-project.org/support.html
  - Globus Compute (formally funcX): https://funcx.org/
- **Balsam**
  - docs: https://argonne-lcf.github.io/balsam/
  - github: https://github.com/argonne-lcf/balsam
  - slack: https://join.slack.com/t/balsam-workflows/shared_invite/zt-1t0736hsz-6hxsmC~0MBFpuP~WvouwWQ
- **Globus Compute**
  - docs: https://globus-compute.readthedocs.io/en/latest/quickstart.html
- Recent workflows workshop materials (includes materials on how to run GNU Parallel, Parsl, Balsam & Fireworks on Polaris): https://github.com/CrossFacilityWorkflows/DOE-HPC-workflow-training
- Workflows community (group where you can discover new workflow tools & connect with workflows community) : https://workflows.community/


Thank-you!
Questions?