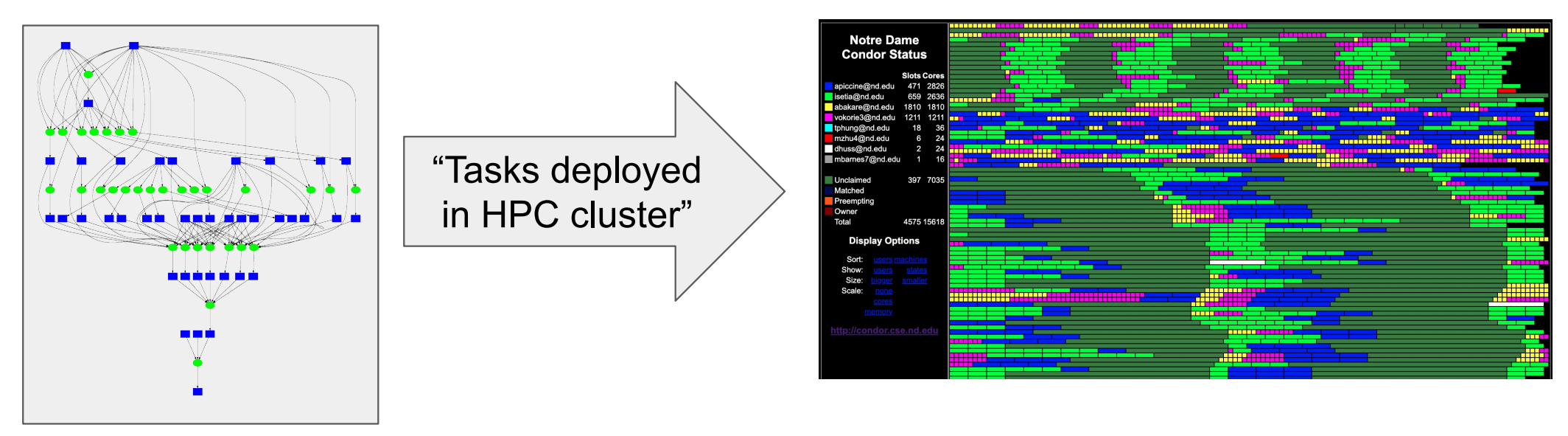
Floability: Enabling Portable Scientific Workflows Across HPC Facilities

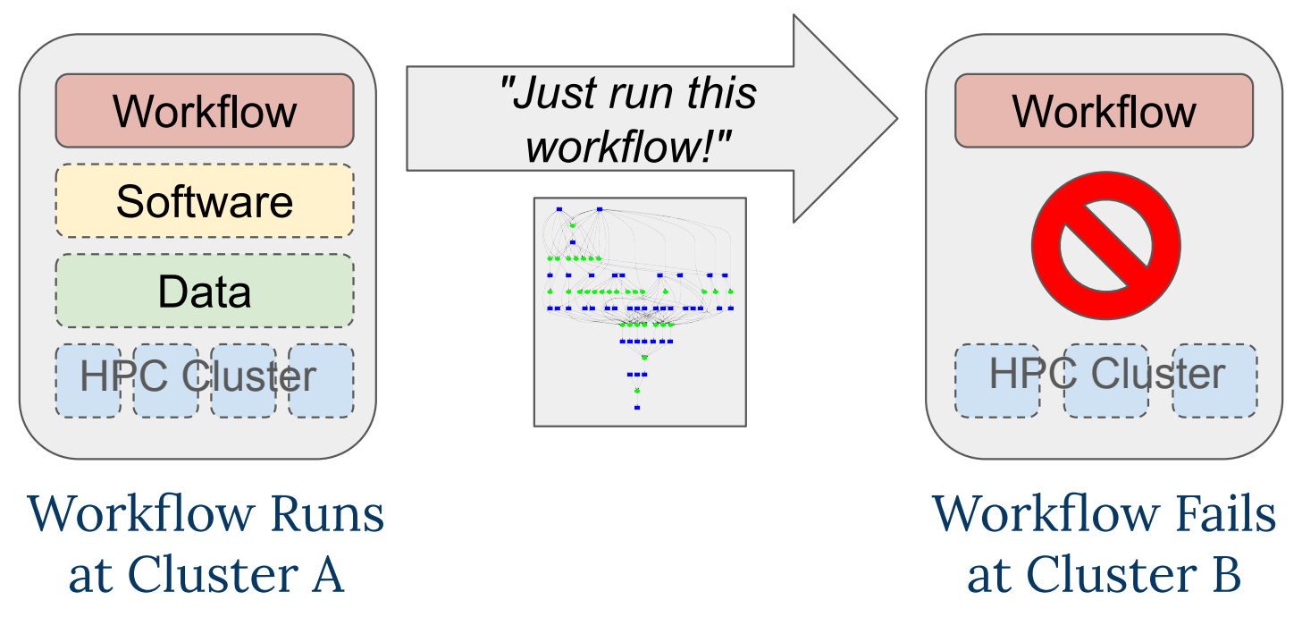
Md Saiful Islam (mislam5@nd.edu)

On behalf of the Floability team: Prof. Tanu Malik, Prof. Kevin Lannon, Prof. Shaowen Wang, Prof. Douglas Thain,
Ben Tovar, Md Saiful Islam, Talha Azaz, Dr. Furqan Baig, A S M Shahadat Hossain, and Raza Ahmad

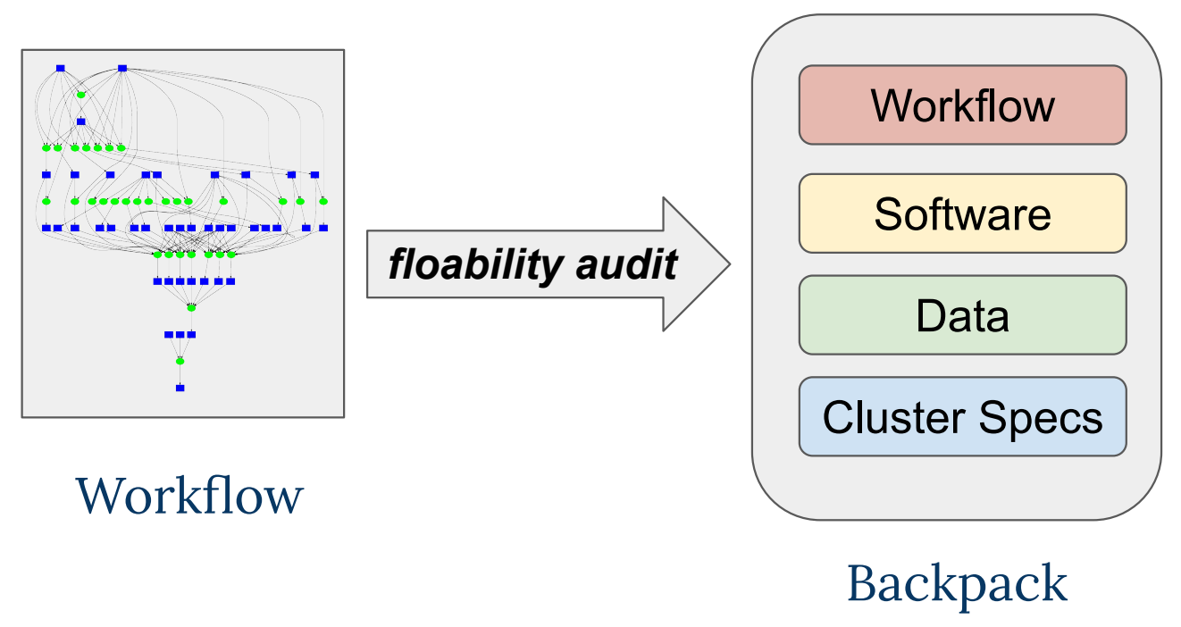
Scientific workflows often consist of millions of tasks arranged in a DAG.



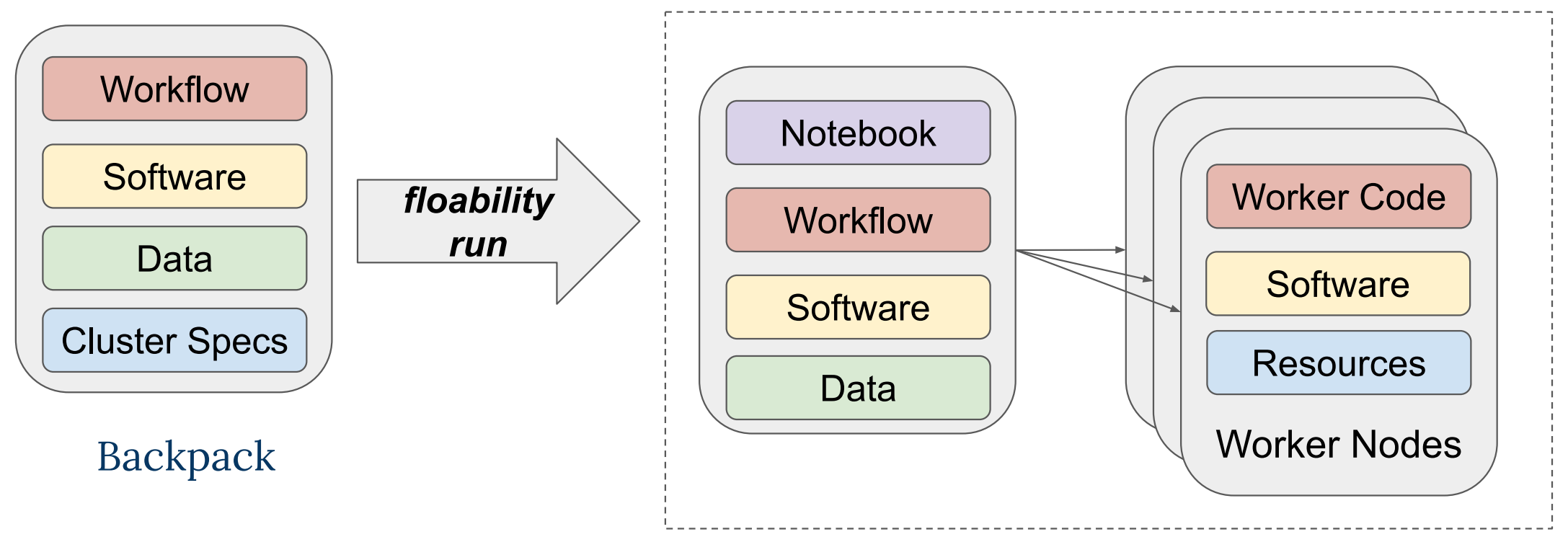
A workflow by itself cannot run without the data and its supporting environment.


Workflow
Software
Data
HPC Cluster
"Just run this workflow!"
Workflow
HPC Cluster
Workflow Runs at Cluster A
Workflow Fails at Cluster B

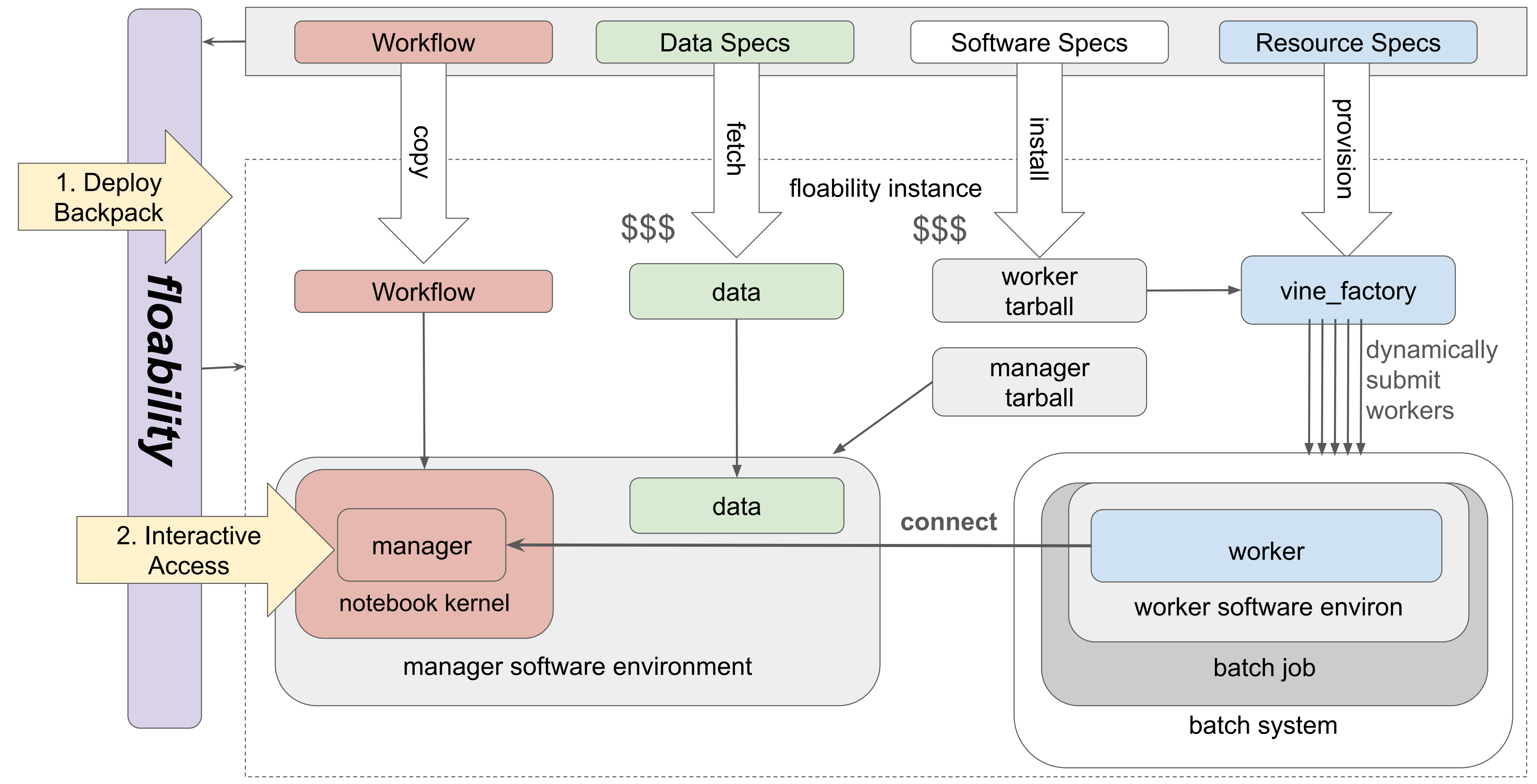
A backpack specifies all dependencies needed to execute a notebook-based workflow.



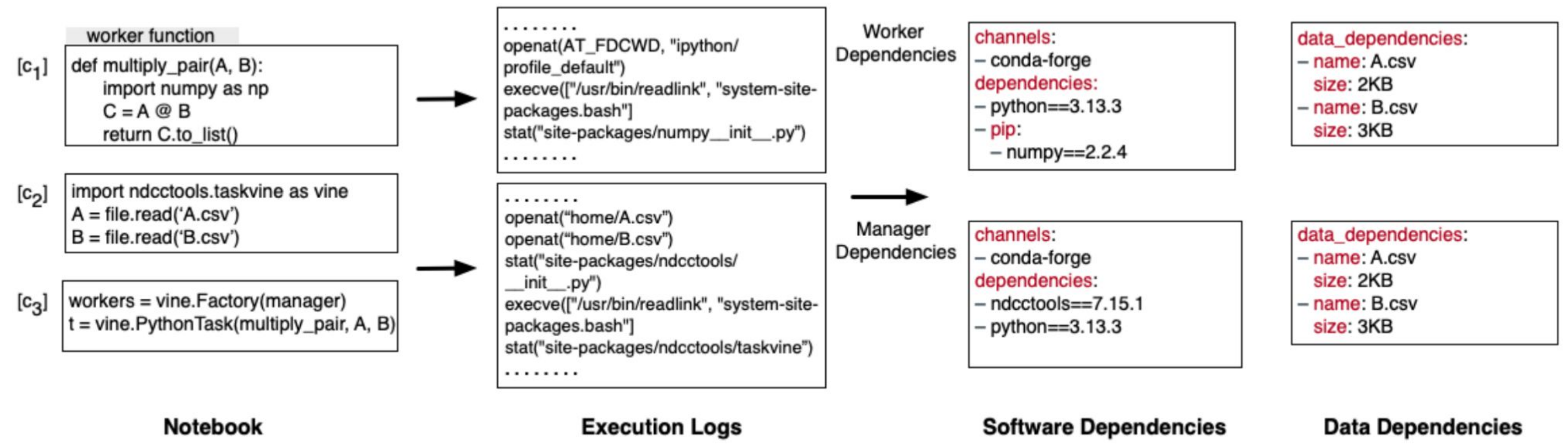
Floability deploys a backpack into an HPC cluster.



HPC Cluster
(HTCondor, SLURM, UGE,...)



Floability architecture consists of a manager running on the head node and multiple worker processes running on compute nodes. A factory process is responsible for launching and dynamically scaling workers in the cluster.



```
def multiply_pair(A, B):
    import numpy as np
    C = A @ B
    return C.to_list()
```

```
import ndcctools.taskvine as vine
A = file.read('A.csv')
B = file.read('B.csv')
```

```
workers = vine.Factory(manager)
t = vine.PythonTask(multiply_pair, A, B)
```

```
........
openat(AT_FDCWD, "ipython/
profile_default")
execve(["/usr/bin/readlink", "system-site-
packages.bash"]
stat("site-packages/numpy__init__.py")
........
```

```
........
openat("home/A.csv")
openat("home/B.csv")
stat("site-packages/ndcctools/
__init__.py")
execve(["/usr/bin/readlink", "system-site-
packages.bash"]
stat("site-packages/ndcctools/taskvine")
........
```

```
channels:
– conda-forge
dependencies:
– python==3.13.3
– pip:
  – numpy==2.2.4
```

```
data_dependencies:
– name: A.csv
  size: 2KB
– name: B.csv
  size: 3KB
```

```
channels:
– conda-forge
dependencies:
– ndcctools==7.15.1
– python==3.13.3
```

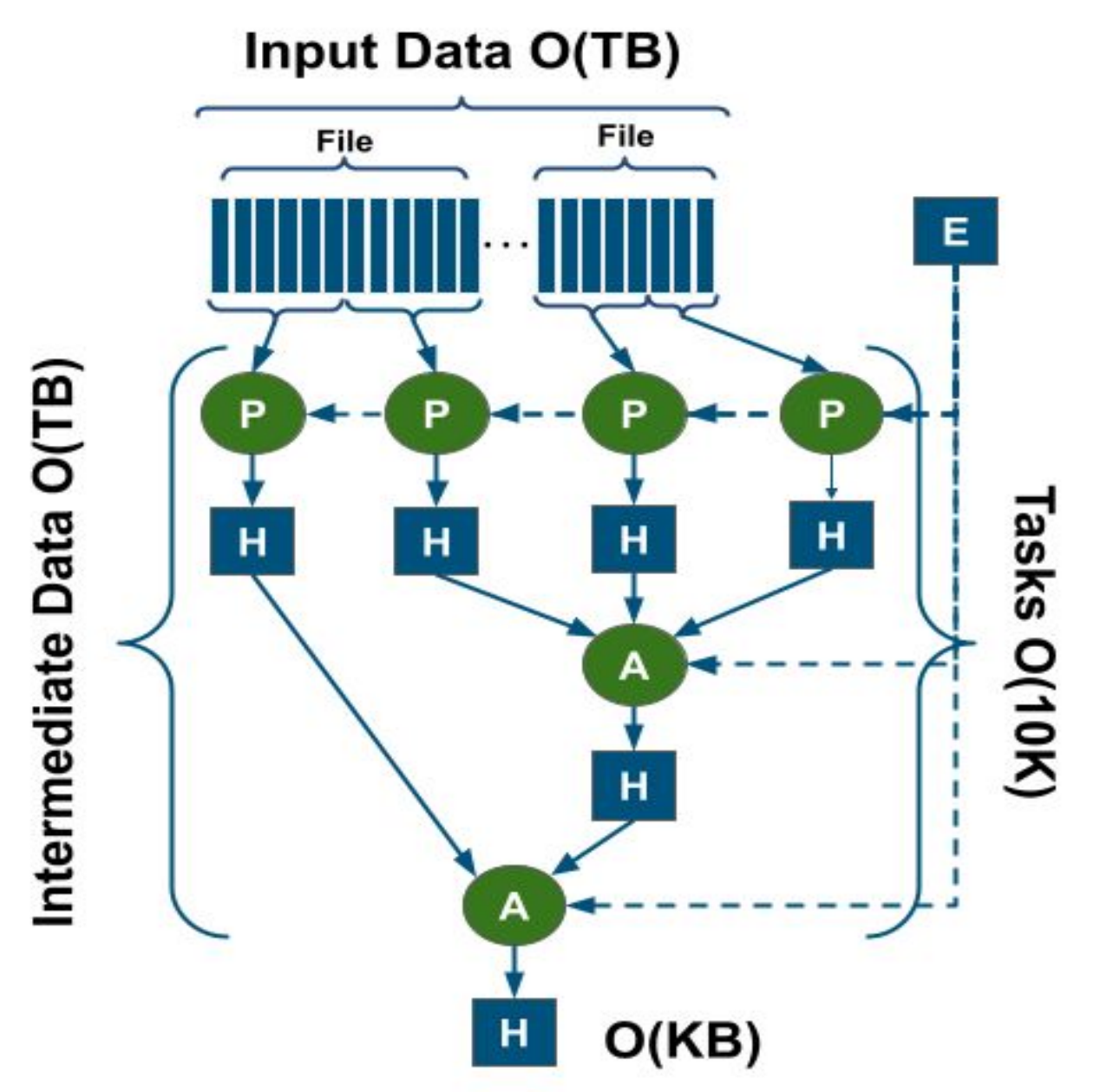
```
data_dependencies:
– name: A.csv
  size: 2KB
– name: B.csv
  size: 3KB
```

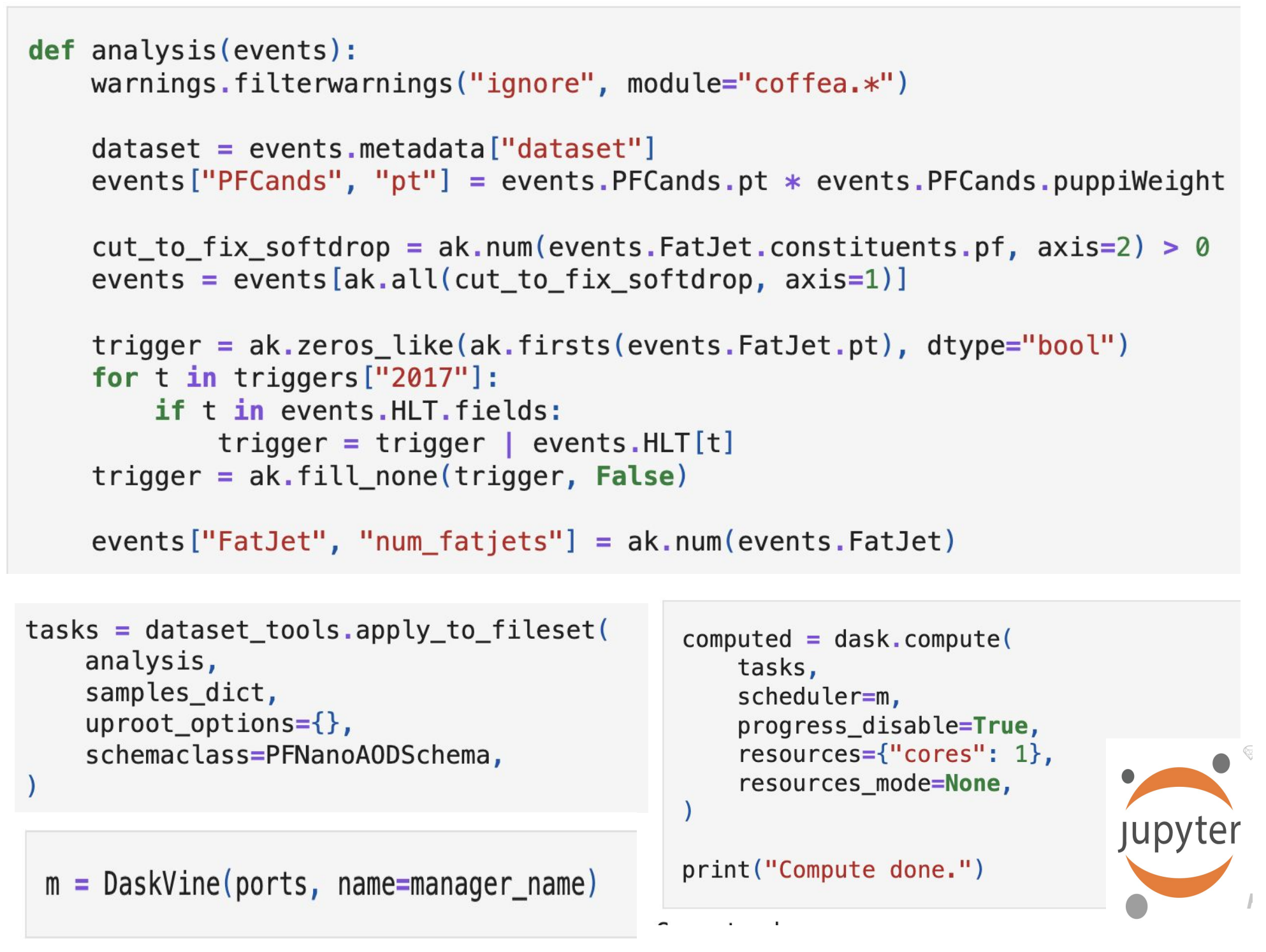
Floability ***audit*** uses system call tracing to extract manager and worker software packages, which are then validated to produce an environment file with precise versions.

Example Applications

DV5 CMS Analysis Application

Consumes 1.5TB Data
Submits 17K Tasks, Uses 2400 cores, 200 nodes.





```
def analysis(events):
    warnings.filterwarnings("ignore", module="coffea.*")

    dataset = events.metadata["dataset"]
    events["PFCands", "pt"] = events.PFCands.pt * events.PFCands.puppiWeight

    cut_to_fix_softdrop = ak.num(events.FatJet.constituents.pf, axis=2) > 0
    events = events[ak.all(cut_to_fix_softdrop, axis=1)]

    trigger = ak.zeros_like(ak.firsts(events.FatJet.pt), dtype="bool")
    for t in triggers["2017"]:
        if t in events.HLT.fields:
            trigger = trigger | events.HLT[t]
    trigger = ak.fill_none(trigger, False)

    events["FatJet", "num_fatjets"] = ak.num(events.FatJet)
```

```
tasks = dataset_tools.apply_to_fileset(
    analysis,
    samples_dict,
    uproot_options={},
    schemaclass=PFNanoAODSchema,
)
```

```
m = DaskVine(ports, name=manager_name)
```

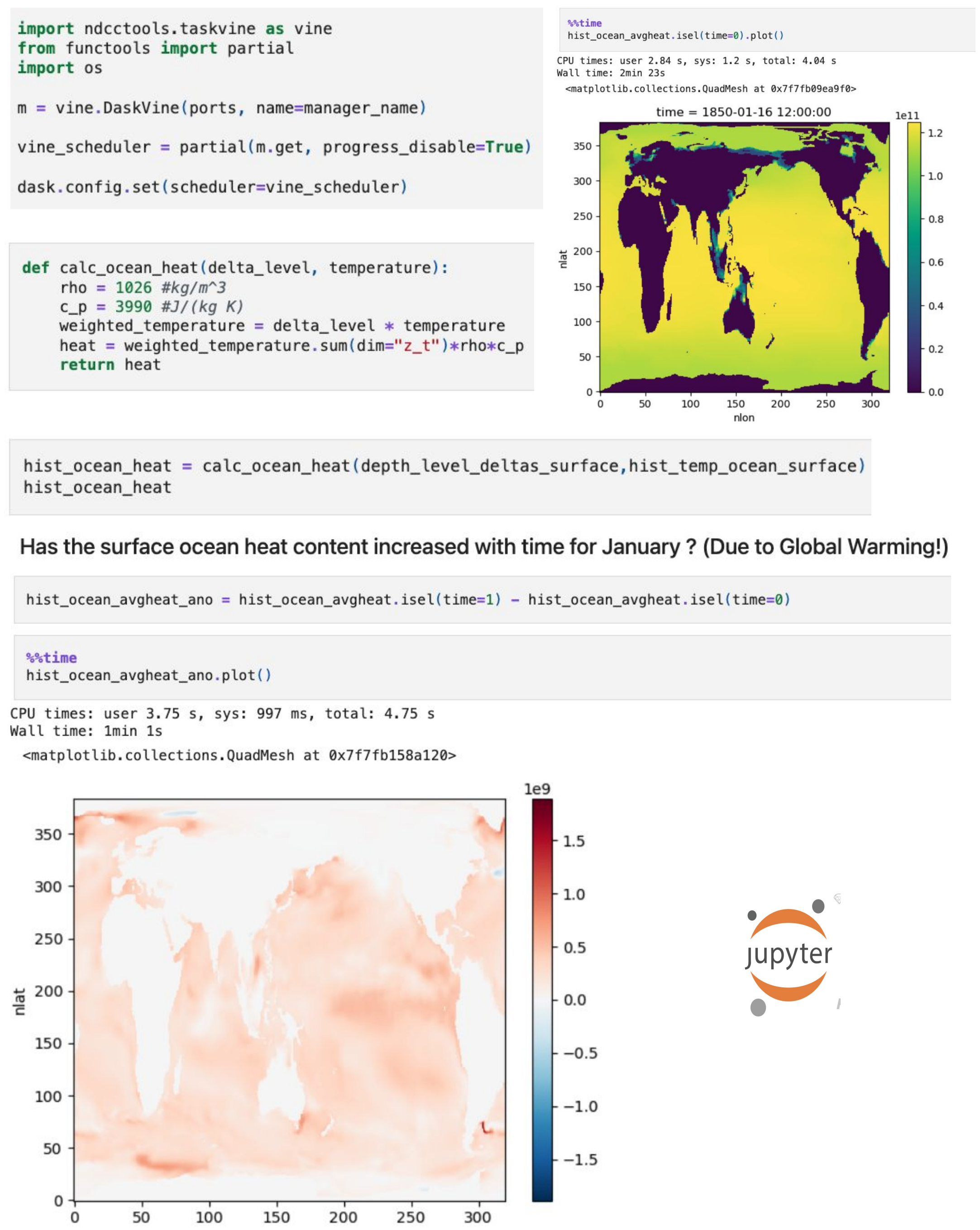
```
computed = dask.compute(
    tasks,
    scheduler=m,
    progress_disable=True,
    resources={"cores": 1},
    resources_mode=None,
)

print("Compute done.")
```

Provided By: Kevin Lannon and Connor Moore

Surface Ocean Heat (CESM2)

Data: CESM2 LENS 1850-2100
Tasks: 2800+ parallel jobs for each plot



```
import ndcctools.taskvine as vine
from functools import partial
import os

m = vine.DaskVine(ports, name=manager_name)

vine_scheduler = partial(m.get, progress_disable=True)

dask.config.set(scheduler=vine_scheduler)
```

```
def calc_ocean_heat(delta_level, temperature):
    rho = 1026 #kg/m^3
    c_p = 3990 #J/(kg K)
    weighted_temperature = delta_level * temperature
    heat = weighted_temperature.sum(dim="z_t")*rho*c_p
    return heat
```

```
%%time
hist_ocean_avgheat.isel(time=0).plot()
```

```
CPU times: user 2.84 s, sys: 1.2 s, total: 4.04 s
Wall time: 2min 23s
<matplotlib.collections.QuadMesh at 0x7f7fb09ea9f0>
```

```
hist_ocean_heat = calc_ocean_heat(depth_level_deltas_surface,hist_temp_ocean_surface)
hist_ocean_heat
```

Has the surface ocean heat content increased with time for January ? (Due to Global Warming!)

```
hist_ocean_avgheat_ano = hist_ocean_avgheat.isel(time=1) - hist_ocean_avgheat.isel(time=0)
```

```
%%time
hist_ocean_avgheat_ano.plot()
```

```
CPU times: user 3.75 s, sys: 997 ms, total: 4.75 s
Wall time: 1min 1s
<matplotlib.collections.QuadMesh at 0x7f7fb158a120>
```

Provided By: Harsha Hampapura

UNIVERSITY OF NOTRE DAME

https://floability.github.io

This work was supported by NSF Grant OAC-02411436

floability