# dask: lazy, parallel and OOC

- xarray runs either numpy or dask under the hood
- if chunks are specified, then dask is the backend
- dask operates in lazy mode, numpy in eager mode
- dask build graph of operations, delays execution
- dask only executes when data is requested (plot,…)
- execution is multi-threaded on cluster (local, k8s, jobqueue)
- can handle dataset size larger than memory (OOC)

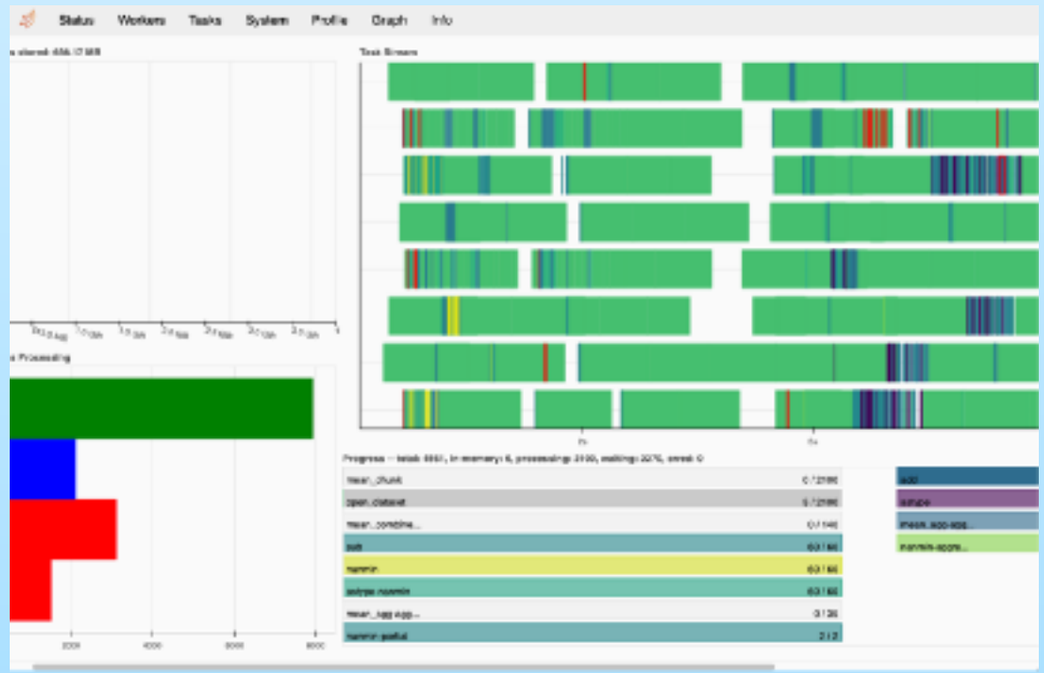
```
from dask.distributed import Client, LocalCluster
cluster = LocalCluster()
client = Client(cluster)
client
```

**Client**

**Scheduler:** tcp://127.0.0.1:63195
**Dashboard:** http://127.0.0.1:63196/status

**Cluster**

**Workers:** 4
**Cores:** 8
**Memory:** 17.18 GB



# Zarr: optimized cloud storage

Why Bother with a new format?

```
dmdu -sh my_OM4p125_run/*

6.8T    history
9.4T    pp
1.8T    restart
2.7T    zstore
```

- zarr have BLOSC compression
- designed for cloud object storage
- chunk size matters (10-100 Mo)
- stores can be of different types (zip/directory/…)

```
temp_tendency
├── temp_tendency/0.0.0.0
├── temp_tendency/0.1.0.0
├── temp_tendency/0.2.0.0
├── temp_tendency/0.3.0.0
├── temp_tendency/0.4.0.0
├── temp_tendency/0.5.0.0
├── temp_tendency/0.6.0.0
├── temp_tendency/0.7.0.0
├── temp_tendency/0.8.0.0
├── temp_tendency/0.9.0.0
├── temp_tendency/1.0.0.0
├── temp_tendency/10.0.0.0
```

```
├── ./tosga
│   └── ./tosga/gn
│       └── ./tosga/gn/v1
│           ├── ./tosga/gn/v1/tosga.yml
│           └── ./tosga/gn/v1/tosga.zip
├── ./umo
│   └── ./umo/gn_d2
│       └── ./umo/gn_d2/v1
│           ├── ./umo/gn_d2/v1/umo.yml
│           └── ./umo/gn_d2/v1/umo.zip
├── ./uo
│   └── ./uo/gn_d2
│       └── ./uo/gn_d2/v1
│           ├── ./uo/gn_d2/v1/uo.yml
│           └── ./uo/gn_d2/v1/uo.zip
```