ngVLA Data Processing - Dask Domain Model

Dask Domain Model

Here is what I think our domain model should be in the context of the dask ecosystem.

I would like to find a new name for the "Cluster Resource Manger" and rethink the functions allocated to it in the previous post.

I think we should consider how various use cases map to underlying compute resources and separate out issues related to deployment in order to help guide and organize our prototyping efforts.

Resource Managers

Resource Managers manage work on compute resources.

- Compute Resource : Resource Manager Example
- local : laptop OS
- HCP : Slurm, PBS, etc (AKA job queuing systems)
- HTC : HTCondor
- Cloud: AWS, GCP, etc

Cluster Managers

Cluster Managers are dask abstractions included in dask_jobqueue that deploy a scheduler and workers as determined by communicating with resource managers. e.g. dask_jobqueue.SLURMCluster.

A worker is a Python object and node in a dask cluster that serves two purposes, 1) serve data, and 2) perform computations.

Jobs are resources submitted to and managed by resource managers.

For job queuing system Resource Managers, the Cluster Manager configures each node and generates a job script for the underlying resource manager.

dask_distributed provides a client that interacts with Cluster Managers. The client abstraction provides a consistent interface to the user for any Cluster /Resource Manager combination.

Dask Gateway

Dask Gateway provides a secure, multi-tenant server for managing dask clusters. It allows users to launch and use dask clusters in a shared, centrally managed cluster environment, without requiring users to have direct access to the underlying cluster backend (e.g. Kubernetes, Hadoop/YARN, HPC Job queues, etc…).

HTC

dask_jobqueue provides an HTCondor Cluster Manger.

dask-CHTC customizes dask_jobqueue to fit CHTC’s needs.

Cloud

dask_cloudprovider provides abstractions for constructing and managing ephemeral Dask clusters on various cloud platforms.