Connecting Workflows To Mesos

Charles Zheng
czheng2@gmail.com
Mesos architecture

Framework 1
Job 1 | Job 2
FW Scheduler

Framework 2
Job 1 | Job 2
FW Scheduler

Allocation module
Mesos master

Agent 1
Executor
Task 1 | Task

Agent 2
Executor
Task | Task

CTTools
Why Mesos

- Mesos uses container technology for resource management that has lower overhead comparing to system using virtual machine
- Support Containers (Mesos container and Docker)
- The two level architecture is extensible that allow different users to build their own applications on the same platform
- It has been deployed on many commercial clouds
A typical HPC workflow

The diagram shows a sequence of steps:

1. **Input Sequence**: 30 GB
2. **Split 8 mins**
3. **Subsequences**:
   - subseq1: 30 MB
   - subseq2: 30 MB
   - subseqm: 30 MB
   - subseqn: 30 MB
4. **BWA**:
   - bwa: 8 secs
5. **Outputs**:
   - out1: 23 MB
   - out2: 23 MB
   - outm: 23 MB
   - outn: 23 MB
6. **Combine 8 mins**
7. **Result**: 20 GB

The diagram also includes references to:

- **bwa index**
- **Sequence Index**
- **Standard Sequence**: 35 MB

The workflow is indicated by the arrows labeled S1, S2, and S3.
Design challenges

- Synchronize workflow status between workflow engine and Mesos
- Avoid starvation caused by the default resource allocation policy
- Executors do not transfer results back to user
- Garbage collection policy used by Mesos will cause the disk be filled up quickly
- Tasks resource requirements are not known in advance
Makeflow and Mesos

1. Write ready tasks info
2. Get ready tasks info
3. Notice Scheduler to submit ready tasks
4. Submit tasks to Mesos Master
5. Assign tasks to Executors and sync the inputs
6. Execute tasks and send output back
7. Write the finished tasks info
8. Read the finished tasks info
9. Notice Makeflow the task is finished

Mesos Master
Mesos Scheduler
Task Info Monitor
Makeflow
mesos Task Info
mesos Task state
Slave1
Slave2
Slave3
Executor
Executor
Executor
How to use

● Makeflow and Mesos
  ○ `makeflow -T mesos --mesos-master=<ip:port> --mesos-path=/path/to/mesos/python/site-packages`

● Makeflow, Work Queue and Mesos
  ○ `makeflow -T wq -N <project_name>` Makeflow
  ○ `work_queue_factory -T mesos -M <project_name> --mesos-master=<ip:port> --mesos-path=/path/to/mesos/python/site-packages`
Lessons learned

● A workflow monitor is required to communicate between workflow engine and Mesos
● Uncarefully task arrangement of workflows can cause starvation
● Remember to delete results!
● An intermediate task scheduling framework (Work Queue) is preferred
Where to get it

A pull request is currently available.