Dynamic provisioning and execution of HPC workflows using Python

Chris Harris, Patrick O'Leary, Michael Grauer, Aashish Chaudhary, Chris Kotfila and Robert O'Bara

Overview

- Motivation
- HPC Workflows
- HPC Resources
- Cluster provisioning
- Data management
- Job submission
- Workflow orchestration
- Result/Applications
- Conclusion

Motivation

- HPC workflows have enabled significant research advances
- Barriers to widespread adoption remain
  - Complex to use
  - Require specialist local expertise
  - Expensive dedicated hardware

Cumulus

- Platform for dynamic provisioning and execution of HPC workflows
- Intended to make HPC workflows more accessible to developers
- Key functionality
  - Cluster provisioning
  - Data management
  - Job submission
  - Workflow orchestration

HPC Workflows

- Are tasks executed in order to carry out some computation on a HPC resource
- Jobs running on HPC resources
  - Simulation code
  - Data processing
- Auxiliary task run outside HPC resources
  - Transferring input data to HPC resource
  - Post-processing of results

HPC Resources

- “Traditional” HPC Resources
  - Dedicated hardware using sophisticated interconnects
- “Dynamic” HPC Resources
  - Built on demand from virtual server in public or private cloud
    - AWS EC2
    - OpenStack
  - Size and characteristics tailored to workflow
  - Only pay for what you use
  - Interconnects are significantly slower

Design principles

- Hide complexity associated with HPC workflows
  - Application development rather than infrastructure
- Allow workflows to be portable across HPC resources
- Expose RESTful endpoints
  - Language agnostic for clients

Cluster provisioning

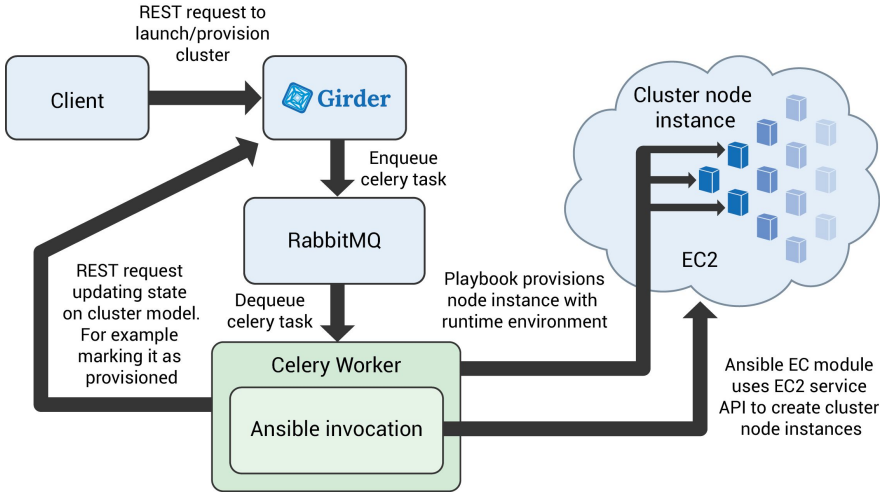
- Launch and provision dynamic clusters tailored to a specific workflow
- Process composed of two steps
  - Launching
  - Runtime Provisioning
- Ansible
  - Automation tool for system configuration and software deployment
  - Declarative operations defined through
    - Reusable roles
    - Use case specific playbooks

Cluster provisioning - Launching

- Creating the virtual servers in the cloud environment
  - Tailor machine type and cluster size
- Machine images
  - Template from which virtual servers are created
  - Base operating system and software
  - Workflow specific images
    - Pre-installed software stack
    - Reproducible environment
    - Reduce cluster startup time

Cluster provisioning - Runtime provisioning

- Runtime configuration
  - E.g. configuration involving network topology
- Built-in support for MPI environment using SGE
- Additional playbooks can be added
  - E.g. Apache Spark.



Data management

- HPC workflows are data driven
  - Cluster and input configurations
  - Output dataset
  - Performance statistics
- Appropriate access controls needed
- Girder
  - Open-source web-based data management platform
  - Exposes RESTful endpoint
  - Provides cumulus with three key pieces of functionality
    - Data organization and access
    - User management and authentication
    - Authorization management

Job submission

- Cumulus using conventional job schedulers
  - SGE, PBS and Slurm (+NEWT)
- Provides a scheduler provides abstraction
- Access to HPC resources through SSH
  - Key-based authentication
  - Provides a secure and standard interface to a variety of
    - Public and private traditional HPC resources
    - Cloud based HPC resources

Workflow orchestration

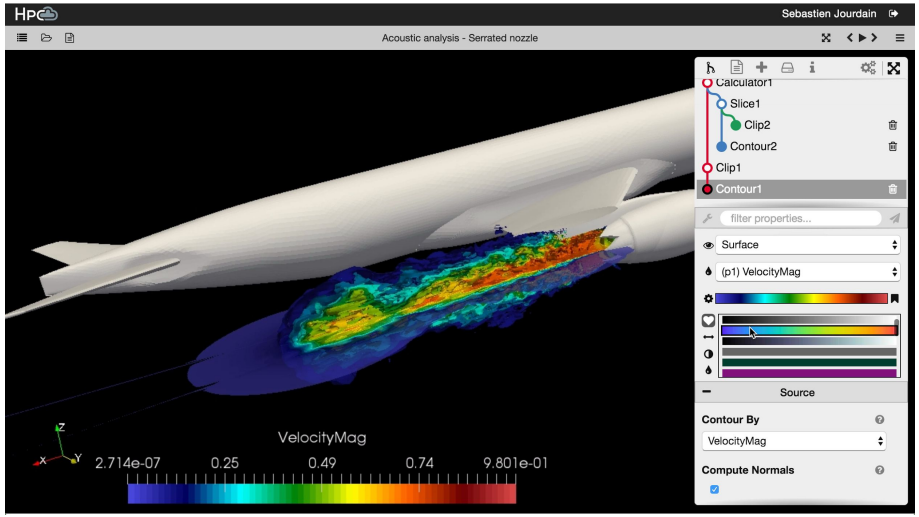
- Combines the *cluster provisioning, data management and job submission* into a workflow
- Workflow topology
  - Simple linear flows
  - Complex flows containing branches and loops
- Efficient and scalable
  - Workflows are potentially very long lived
  - Consume minimal resources while monitoring HPC jobs

Workflow orchestration - TaskFlow

- TaskFlow - A simple yet powerful workflow engine built on Celery
- Celery
  - Open-source asynchronous task queue
  - Tasks are simple Python functions
  - Simple linear scaling

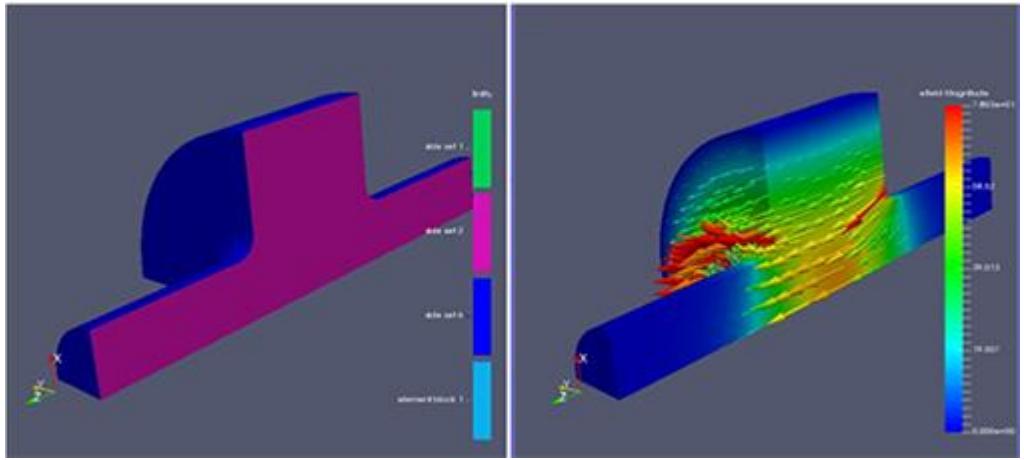
Applications - HPCCloud

- Web-based simulation environment
  - High-level workflows
  - Simple intuitive web UI
- Motivated Cumulus development
- Implements a number of workflows
  - PyFR simulations
  - ParaViewWeb visualization



Applications - ModelBuilder

- Computational Model Builder (CMB) framework
  - Advanced simulation workflows on the desktop
- Multiphysics workflows
  - Particle accelerator simulations
- Qt desktop application
  - API validation in non-web environment



Conclusion

- Cumulus is a novel platform for developing end-to-end HPC workflows
  - Targeting traditional and cloud-based HPC resources
- The platform provides
  - Cluster provisioning
  - Data management
  - Job submission
  - Workflow orchestration
- Its capabilities have been demonstrated in a variety of end-user applications