TOPMed Analysis Pipeline on the Cloud

Analysis Pipeline on Google Cloud Platform (GCP)

Presentation

- Analysis Pipeline background
  - Design
  - Delivery
- Cluster in the cloud
  - Requirements
  - Creating a cluster in the cloud
- Analysis Pipeline on GCP
  - Preliminaries
  - Analysis Pipeline support of Slurm on GCP
  - Issues encountered
  - Benchmarks, tests, and results
  - Closing remarks

Analysis Pipeline Background - Design Objectives

- ❑ Support multiple workflows
- ❑ Extensible framework supporting multiple clusters for parallelization
  - Local UW Cluster
  - Cloud vendors
- ❑ Pipeline execution examples:

```
# single variant pipeline on local cluster
$ analysis_pipeline/assoc.py single assoc_single.config

# single variant pipeline on aws batch cluster
$ analysis_pipeline/assoc.py single assoc_single.config --cluster_type  AWS_Batch

# single variant pipeline on gcp slurm
$ analysis_pipeline/assoc.py single assoc_single.config --cluster_type Slurm_Cluster
```

Analysis Pipeline Background - Design

- Two major components
  - Workflow system (Python code)
  - Analyses (R code)

Analysis Pipeline Background - Design (cont)

- Workflow system (Python code)
  - Provides multi-step analyses' workflows
    - computing GRM
    - fitting a null model
    - association tests (single, aggregate, window)
  - Framework to parallelize workflows on local and cloud clusters
    - Base class providing attributes and methods common to all clusters
    - Unique cluster classes derived from the base class and providing cluster specific attributes and methods

Analysis Pipeline Background - Design (cont)

- Analyses (R code)
  - Provides internal R code to execute each analysis
  - Requires various R packages (mostly via Bioconductor) in executing analyses
  - Executed on the cluster

Analysis Pipeline Background - Delivery

- Github
  - Pipeline code repository
    - Python workflows
    - Pipeline R code
    - Installation scripts
  - Required software tools not in repository
    - R
    - SAMtools
    - PLINK
    - KING
    - LocusZoom
- Docker images containing
  - Analysis Pipeline (based on github branches)
  - Required software tools

Clusters in the Cloud

- ❑ Cluster requirements
- ❑ Cloud requirements
- ❑ Creating a cluster in the cloud

Clusters in the Cloud - Cluster Requirements

- A shared file system to share data on the cluster master and compute nodes
- Support of docker
- Heterogeneous compute environment (based on cores and memory)
- Optimized job scheduling based on a job’s required max memory and number of cores
- Support dependent job synchronization
- Support array jobs

Clusters in the Cloud - Cloud Requirements

- ❏ Discount pricing options (e.g., Spot pricing or Preemptible VMs)
- ❏ Autoscaling
- ❏ Creating custom compute configurations
  - ❏ Preloading docker images
  - ❏ Access to shared file system

Clusters in the Cloud - Creating a Cluster

- Create and configure the cloud environment
  - IAM accounts
  - Permissions
  - Security
  - VPC (networks)
  - Shared data means (NFS or other)

Clusters in the Cloud - Creating a Cluster (cont)

- Install and configure the cluster
  - Controller/Login VM instance(s)
  - Queues ("partitions" in Slurm)
  - Resource (memory, cpu) settings
  - Shared storage
  - Compute node VM images
  - Customization installations
    - Python and appropriate packages
    - Docker
    - Preload pipeline docker image

Analysis Pipeline on GCP

- ❏ Preliminaries
- ❏ Create GCP Slurm cluster
- ❏ Extend Analysis Pipeline for Slurm
- ❏ Issues encountered
- ❏ Benchmarks and tests
- ❏ Concluding Remarks

Analysis Pipeline on GCP - Preliminaries

- ❏ Review GCP technology and architecture
- ❏ Integrate UW accounts/G Suite into the NIH STRIDES account
- ❏ Learn Slurm technology especially juxtaposed to SGE
- ❏ Install Slurm on our local cluster
- ❏ Understand impact of specific Slurm features on GCP
  - Support for multiple clusters
  - Multiple queues per cluster
  - One machine type per queue (partition)
- ❏ Established naming convention of clusters and queues

Analysis Pipeline on GCP - Create Slurm Cluster

- ❑ Create the GCP environment
- ❑ Deploy and configure various Slurm clusters
  - ■ I/O Performance
    - ❑ Self-deployed NFS shared data
    - ❑ Googles Filestore
    - ❑ Qumulo File system
  - ■ Cost Optimization
    - ❑ GCS Fuse / Google Storage
    - ❑ Standard (Non-Preemptible) VMs
    - ❑ Preemptible VMs

Analysis Pipeline on GCP - Extend Analysis Pipeline

- Python code to support Slurm
  - Define a Python class (Slurm_Cluster) derived in the cluster class hierarchy implementing the cluster specific command to submit a job
  - Provide a Slurm configuration file to dynamically specify submit options
  - Provide a Slurm partitions configuration file

Analysis Pipeline on GCP - Issues Encountered

- Early adopters of Slurm on GCP
- Docker performance
- Mounting NFS problems
- Operating system updates during job executions (causing jobs to fail)
- Lack of GCP resource quotas
- Dependent jobs not starting for job arrays
- Cannot run multiple jobs on the same compute node (when there are sufficient resources on the compute node)
- Idiosyncrasies submitting a Python script to Slurm (script copied to /var/...)

Note: all of these issues were resolved with bug fixes or work-arounds

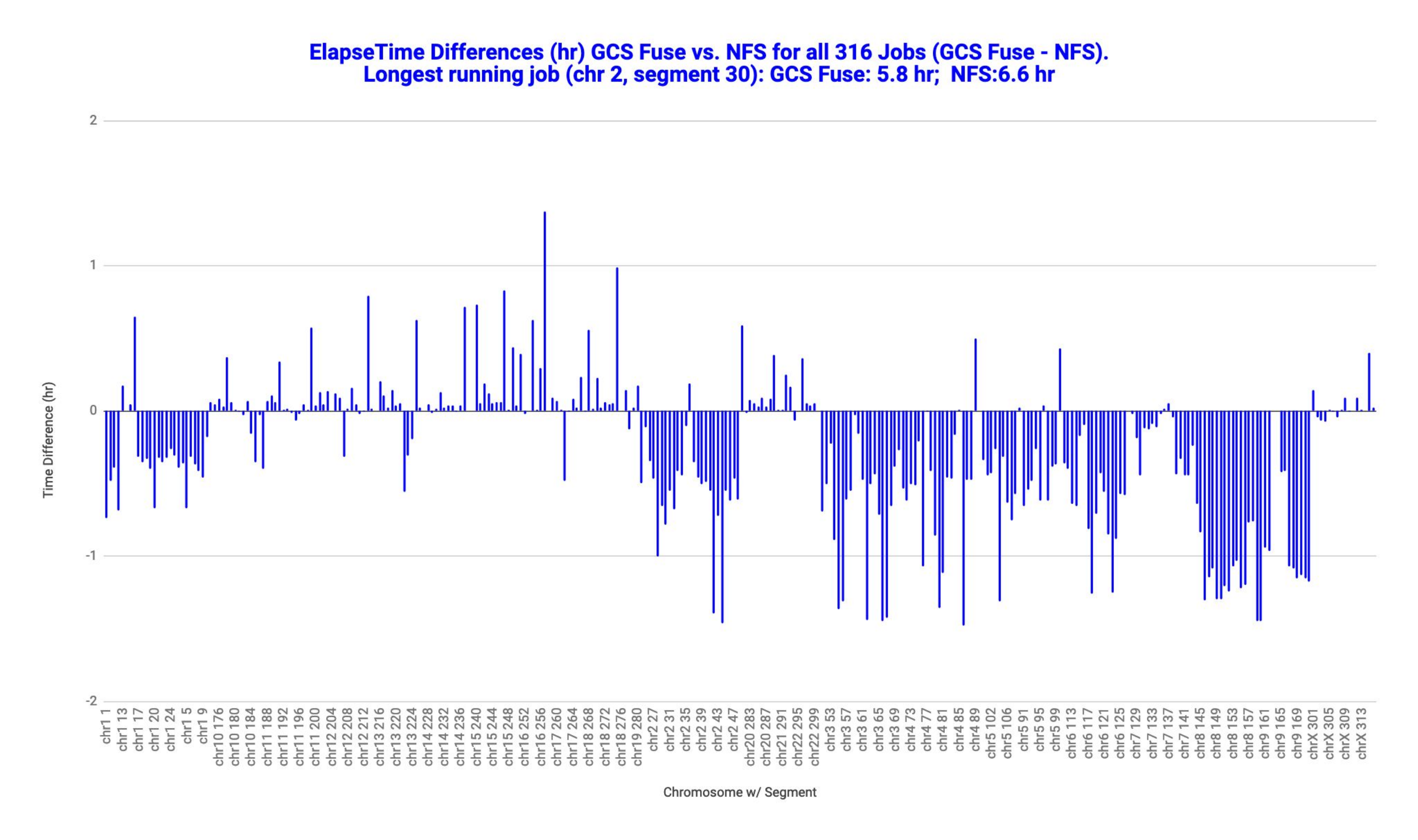
Analysis Pipeline on GCP - Benchmarks and Tests

- Compared performance for different shared data systems (Slurm only)
  - Self-deployed NFS
  - Filestore (Google’s managed NFS service)
  - Qumulo File System (Third-party NFS service)
  - GCS Fuse / Google Storage (file system access to object stored data)
- Compared performance and cost of standard vs. preemptible VMs
- Compared performance association analyses
  - UW
  - AWS Batch
  - Slurm

Analysis Pipeline on GCP - Benchmarks Results

- Compared performance for different shared data systems (Slurm only)
  - No performance difference between self-deployed NFS and:
    - Google’s Filestore
    - Qumulo File System
  - For read-only GCS Fuse / Google Storage and self-deployed NFS performance may have been slightly faster on GCS Fuse (see next slide)
  - Read-write GCS Fuse / Google Storage was not workable for saving results of analysis

Analysis Pipeline on GCP - Benchmark Results (Fuse/NFS)



Analysis Pipeline on GCP - Benchmark Results: Preemptible vs Non-preemptible

Analysis	Data	Condition	Elapse Time (hr)	Cost
Association - Single	Freeze 8	Non-Preemptible	5	$80
Association - Single	Freeze 8	Preemptible	7.5	$23
Association - Aggregate	Freeze 8	Non-Preemptible	7.4	$240
Association - Aggregate	Freeze 8	Preemptible	16.6	$63

Note: Cost is for compute engine only and does not include NFS storage cost

Analysis Pipeline on GCP - Benchmarks Results for Different Clusters

		UW Cluster		AWS Batch		Slurm	
Analysis	Data	Elapse Time (day:hr:min)	Cost	Elapse Time (hr:min)	Cost	Elapse Time (hr:min)	Cost
Assoc single sparse	Freeze 8	21:24	n/a	4:43	$169.63	4:55	$196.65
Assoc smmat sparse	Freeze 8	1:10:43	n/a	6:00	$287.88	6:53	$212.19
Assoc smmat nocoding sparse	Freeze 8	5:10:00	n/a	7:00	$975.00	7:45	$2,084.00
Assoc smmat var6 sparse	Freeze 8	not done	not done	not done	not done	6:45	$3,238.00

Closing Remarks

- Porting Analysis Pipeline to GCP went very well and was relatively easy
- Slurm is a very good job scheduler with more useful features than SGE
  - Various options to specify how dependents jobs should proceed if parent job fails
  - More robust way to specify job arrays (e.g., 1-4, 9, 11)
- GCS Fuse / Google Storage is a cost effective alternative to NFS in sharing read-only data (but we only had < 400GB of read-only data)
- Our self-deployed NFS performs as well as other similar Google services (that are more expensive)
- Access to Google support was critical ($)

Questions