Accelerating Genomics Workflows at NERSC

Nick Tyler, NESAP for Data Postdoc
Data Science Engagement Group
July 26, 2022

NESAP for Data Postdoc

- Working with DOE Joint Genome Institute
- JAWS (JGI Analysis Workflow Service)
  - Runs users genomics workflows across multiple HPC sites
  - Handles data movement with Globus
  - Uses Cromwell to manage workflows
    - Workflow Description Language (WDL)
    - Language used in bioinformatics
    - Parallelism with scatters
- What I've been working on
  - Implement performance monitoring for workflows
  - Helped transition to HTCondor as execution backend
  - Beginning to transition to Perlmutter

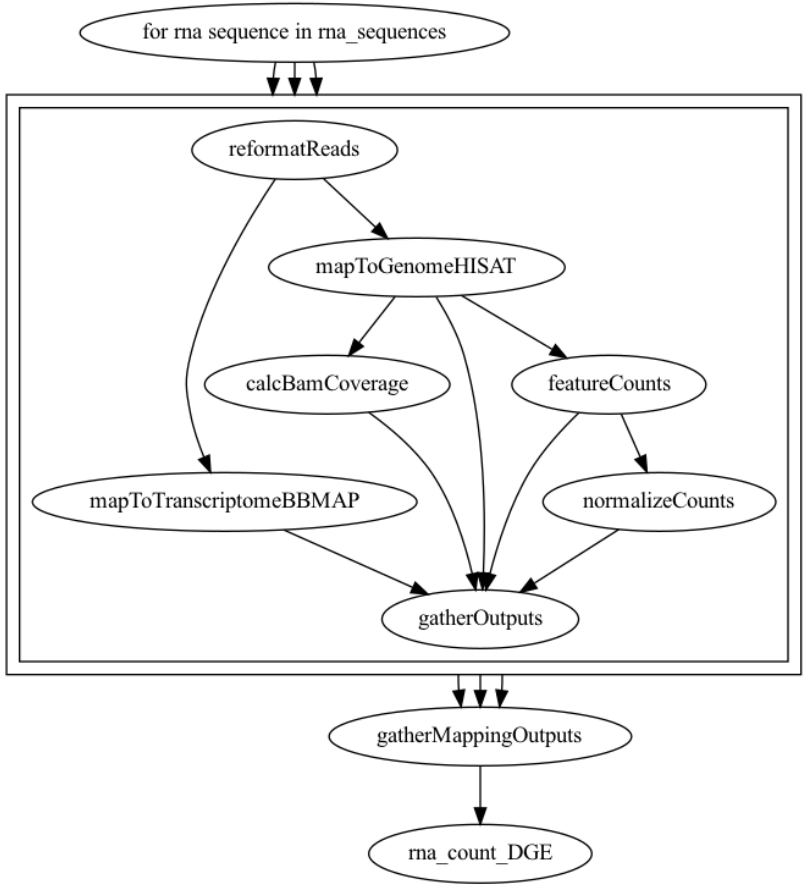
JAWS Workflows

- Workflow tasks are defined in WDL
  - Describes inputs and parameters
  - Command section
    - bash script
    - run inside of a container
  - Define outputs of successful task
  - Request for compute resources
- Inputs are defined in JSON file
  - Same workflow with different data
  - Easy to read and write
  - Automated creation
    - Look for new files in a folder
    - Based on database of data

```
task mapToGenomeHISAT {
    File read1_fastq
    File read2_fastq
    Boolean isSingleEnd
    File genome_index
    String lib_name
    Int n_threads=8
    String run_time

    String reads_input_flag = if(isSingleEnd) then "-U ${read1_fastq}" else "-

    command {
      set -euo pipefail

      cp ${genome_index} .
      tar -xvzf genome_index_files.tar.gz

      hisat2 -p ${n_threads} -k 1 -x genome_index_files/genome_ref ${reads_inp

      samtools index ${lib_name}_hits.bam
      samtools flagstat ${lib_name}_hits.bam > ${lib_name}_map_flag_stats.txt
      samtools depth -q 0 -Q 0 ${lib_name}_hits.bam | pigz > ${lib_name}_hits.
    }
    output {
        File hits_bam = "${lib_name}_hits.bam"
        File hits_bam_index = "${lib_name}_hits.bam.bai"
        File flag_statsfile = "${lib_name}_map_flag_stats.txt"
        File hits_depth = "${lib_name}_hits.bam.depth.gz"
    }
    runtime {
        docker: "danielapeterson/hisat2_samtools:2.2.1"
        time: run_time
        memory: "5G"
        cpu: n_threads
        continueOnReturnCode: true
    }
}
```

JAWS Workflows

- Real workflows are complicated
  - Same steps for multiple files
  - Merge outputs of different programs
  - Conditions on some steps
  - Some parts can be run in parallel
  - Collect outputs for entire run
  - Tasks may need different resources
- Cromwell
  - Workflow engine handles DAG
    - Directed acyclic graph
  - Communicate to compute backend
    - HTCondor

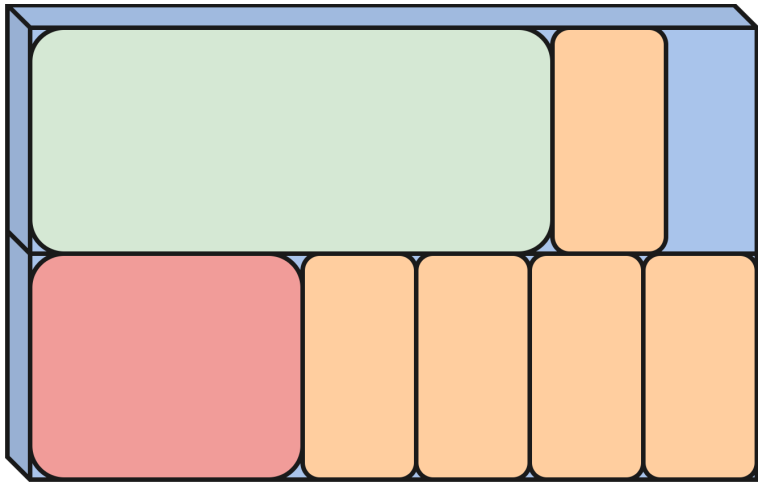


HTCondor Scheduler

- Different type of scheduler from Slurm
  - Designed for high throughput workloads
    - Handles many jobs at once
    - Smaller resource requests then a full node
  - Distributed across different sites
  - Can break up a node into smaller compute “slots”
  - Reuse allocation as much as possible
    - Many users can share the same allocation
- Backbone of the Open Science Pool
  - Project to allow outside access to compute resources
  - Free to US-affiliated research projects and groups
  - Back scheduling on many HPC systems so there is no idle time
  - Used heavily by HEP/NP

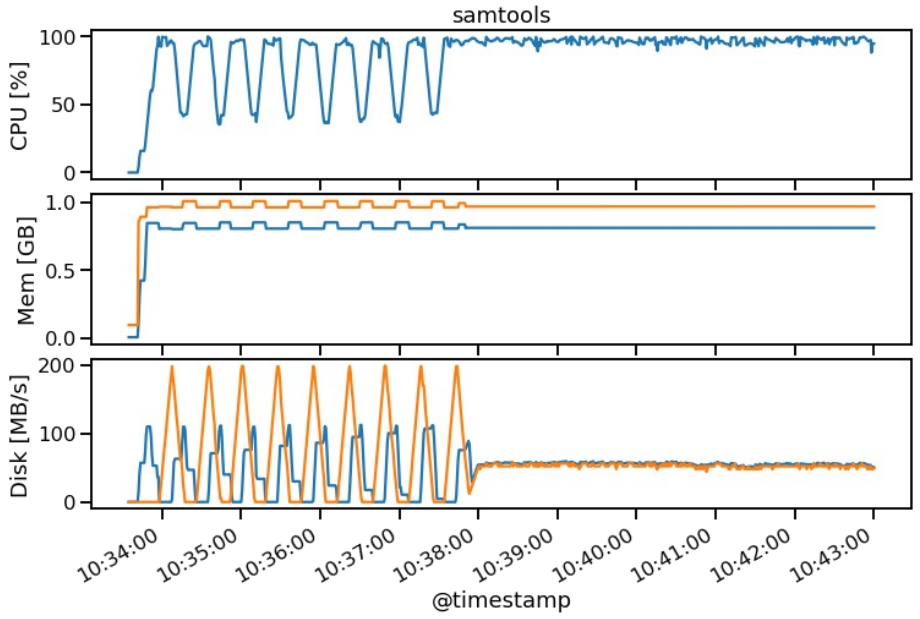
Efficient Use of Resources

- Use HTCondor to create allocation for JAWS
  - WDL runtime section is used to create condor job
  - Fit as many tasks as possible on a node
  - Many workflows can run on a node simultaneously
  - No queue wait time!
- Managing the pool
  - Checks Slurm and HTCondor queue
  - Increase pool by requesting node
    - sbatch
  - Decrease pool when nodes are idle
    - scancel
  - Choose the correct node
    - Large memory nodes, exvivo
    - 1.5TB ram



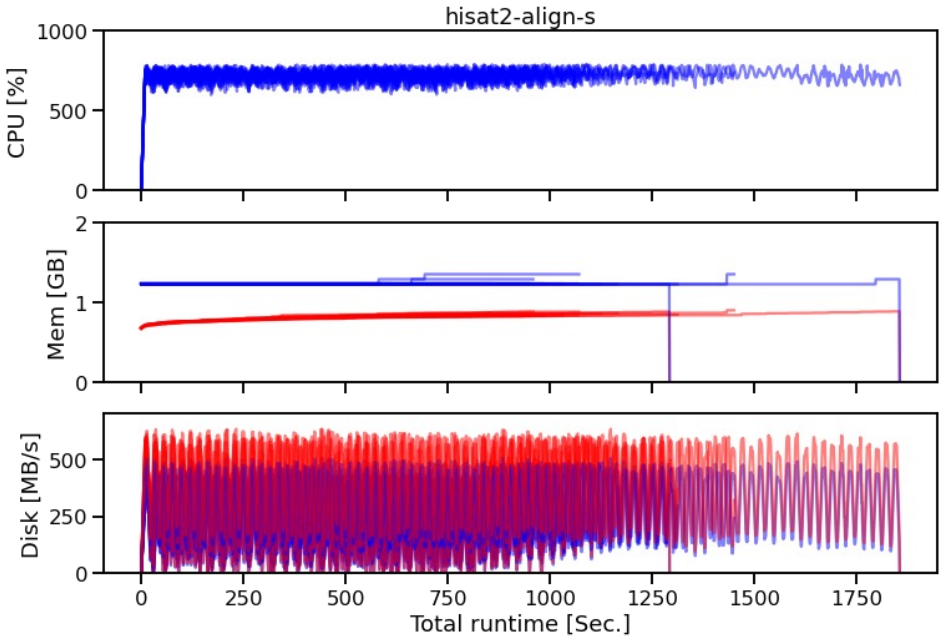
Helping JAWS Users Efficiency

- Want our users to request the correct resources
  - Compare usage with WDL runtime
  - Pack more jobs onto a single node
  - Increase throughput of jobs
- Developers can optimize code
- Tracking performance over multiple sites
  - Insights into a system performance
  - Some workloads might work better at one site over another



Performance Monitoring

- Pagurus
  - github.com/tylern4/pagurus
  - Site agnostic monitoring tool
- Track individual tasks on a node
  - Map data back to the workflow
  - Get individual programs in task
  - Look at usage over multiple runs
- Easy to setup and install
  - Install with pip
  - Uses python psutil library
  - ps (process status) command



Ongoing Work

- Add new sites to JAWS
  - Perlmutter
    - Larger CPU nodes, more memory
    - Allow users to request GPUs
  - JGI compute
    - New system at LabIT
    - Dedicated resource for JGI
- Next steps in my work
  - Current system relies on workflow nodes
    - Runs HTCondor, Cromwell, JAWS
    - Interacts with Slurm
  - Move services to Spin
  - Connect with SuperFacility API

Spin

RANCHER