Breaking the Barriers to Effective Supercomputing: Web Dashboard for Job Accounting and Performance Metrics

Richie Tan
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN
tan447@purdue.edu

Anjali Rajesh
School of Arts and Sciences
Rutgers University
New Brunswick, NJ
anjali.rajesh@rutgers.edu

Guangzhen Jin
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN
jin456@purdue.edu

Rajesh Kalyanam
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN
rkalyana@purdue.edu

***Abstract*—The NSF-funded Anvil supercomputer, built and maintained by the Purdue University Rosen Center for Advanced Computing, enables efficient research computing across a variety of scientific domains nation-wide. In addition to the traditional terminal interface, Anvil uses the open-source web portal framework Open OnDemand to provide a low barrier web interface to the Anvil supercomputer. In this poster we describe enhancements that were made to Anvil's Open OnDemand dashboard to provide a clean, well-structured, and extensible interface for researchers to visualize useful information about their utilization of Anvil without accessing the terminal. Our enhancements include various apps that provide detailed statistics about user jobs and their respective performance metrics while also focusing on data query performance. The information in these apps will enable users to identify and debug any jobs that are noticeably resource-inefficient, as well as improve the queue wait time and efficiency of their jobs in the future.**

***Index Terms*—HPC, Open OnDemand, Anvil, Ruby on Rails, Slurm, dashboard**

Introduction

With the increasing scale of modern data-driven workflows, researchers require efficient yet easy-to-use HPC resources to effectively conduct their research. Many HPC centers typically provide terminal access and utilities for their computing resources but have limited graphical interfaces for researchers with less terminal experience. This project focuses on enhancing the default web-based dashboard for Purdue University's NSF-funded Anvil supercomputer [1], built with Open OnDemand [2]. The goal of the enhanced dashboard is to offer a user-friendly alternative to command-line HPC management, providing an intuitive interface for seamless access to similar data and tools. By developing a user-friendly, productivity-focused dashboard displaying job/allocation accounting data and job performance metrics in an intuitive layout, researchers with varying command line expertise can still easily interact with HPC systems, analyze their workflow performance and resource usage.

System Design

Data Sources

The dashboard retrieves data from two main sources: the Slurm Workload Manager [4] and Purdue Rosen Center for Advanced Computing's (RCAC) Halcyon service [5]. Data from the Slurm database such as job status, accounting information, and partition status is obtained via Slurm shell commands, while Anvil announcements, outages, and maintenance news feeds are retrieved via the Halcyon news API. The disk usage is the only exception, which pulls data from Anvil's ZFS and GPFS servers using a Bash script wrapper called *myquota* (Tab. 1).

Middleware

The middleware for the dashboard consists of a data API and in-memory cache that serves data from the data sources to the Open OnDemand client. Each dashboard feature has a corresponding API endpoint that queries the data sources and stores the data in a memory cache with a certain expiration period. The caching layer ensures that users see as close to real-time data as possible without excessively querying the Slurm or Halcyon databases (Fig. 1).

Dashboard Features

The new dashboard homepage features a welcoming and user-friendly design with widgets and tables that display key metrics: Service Unit (SU) Usage, Disk Usage, the user's current Job Queue, and the status of Anvil's partitions (Fig. 2). Users can monitor resource usage, export data for further analysis, and receive alerts when quotas are reached. The

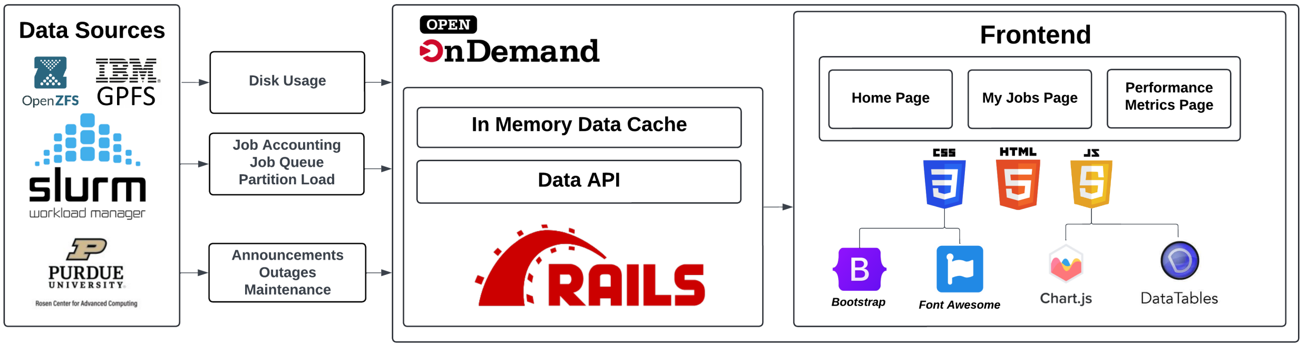


Fig. 1: Flowchart diagram of how the dashboard technologies work together

homepage also provides updates on outages or maintenance that may impact the user.

The *My Jobs* page, accessible from the homepage, displays a table of both the user's own jobs and jobs submitted by other users on the same allocation, including details such as wait time, job efficiency metrics and alerts, real-time SU usage, and a user-friendly job status (e.g. the actual reason a job is pending). Various filtering options are provided in addition to the ability to expand to reveal more information for a job. The page also features graphs depicting SU Usage and Job State distribution for both personal and allocation jobs (Fig. 3).

The *Performance Metrics* page, accessible from the *My Jobs* page, gathers information from the Slurm database to present various job statistics. Users can adjust the data range using tabs for all-time, one year, one month, one week, or one day. The page shows metrics such as the number of jobs within the selected timeframe, the average memory, running time, and CPU efficiencies. The least efficient job ID for each metric is provided so users can further analyze the job to optimize future resource usage (Fig. 4).

Performance Optimizations

The naive solution to showing Slurm job accounting data on the dashboard is to query the Slurm shell commands every time the page is loaded. However, commands like *sacct* and *squeue* fetch data from slurmdbd (Slurm database daemon) and slurmctld (Slurm management daemon) which can become unresponsive under high usage, leading to lags on each web page request. In order to alleviate this, we implemented an in-memory cache that helps to reduce the number of Slurm queries by storing data that doesn't change frequently, including information about completed jobs, disk usage, and service unit usage, while discarding stale data after an expiration period. The My Jobs page first loads the latest cached jobs into the job table before refreshing the data from *sacct*, essentially showing the user nearly up-to-date job data while new data is being fetched while the page finishes loading.

TABLE I: List of Data Sources and Query Commands

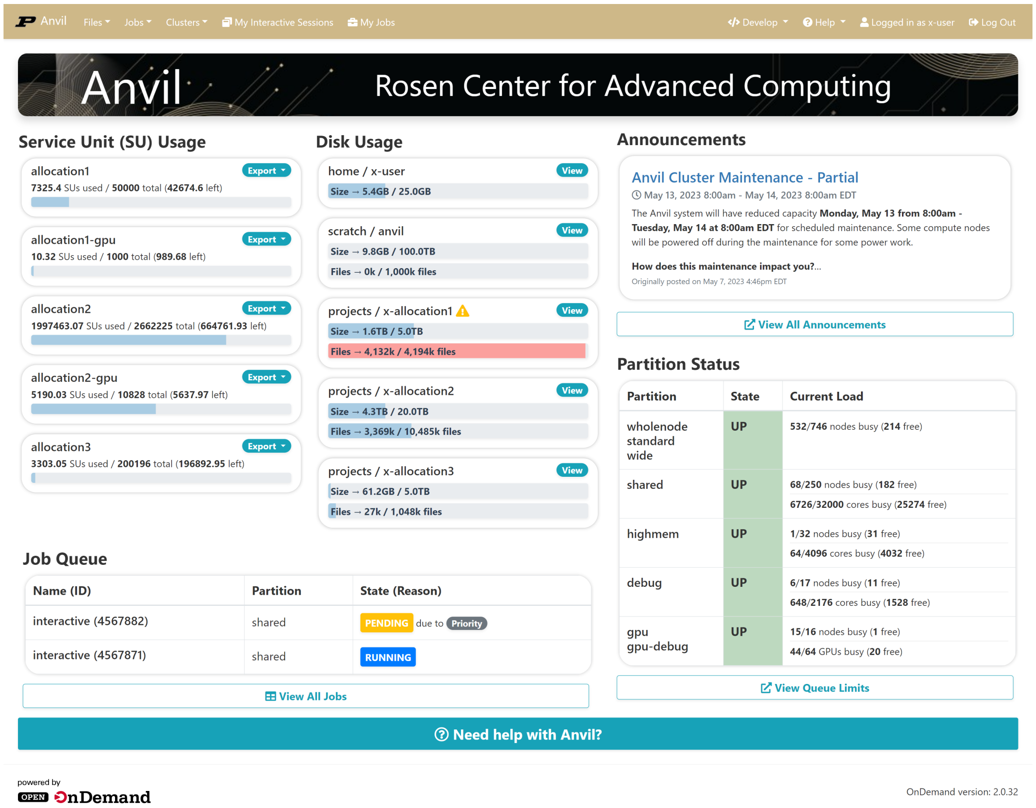
<table>
<tr><th>Feature</th><th>Source</th><th>Query</th></tr>
<tr><td>News Feed</td><td>Halcyon</td><td>GET https://rcac.purdue.edu/api/news</td></tr>
<tr><td>Service Units</td><td>Slurm</td><td>scontrol show assoc flags=assoc -o</td></tr>
<tr><td>Disk Usage</td><td>Slurm</td><td>myquota</td></tr>
<tr><td>Partition Status</td><td>Slurm</td><td>sinfo -h -o ‘%R|%a|%F|%C’</td></tr>
<tr><td>GPU Count</td><td>Slurm</td><td>scontrol show node -oneliner | grep -E ‘NodeName=g0’| grep -Eo ‘CfgTRES=.* AllocTRES=.* CapWatts’</td></tr>
<tr><td>Job Queue</td><td>Slurm</td><td>squeue -t all -h -o ‘%i|%P|%j|%u|%T|%r’</td></tr>
<tr><td>Allocations</td><td>Slurm</td><td>sacctmgr show user {username} withassoc format=account -P -n -r | xargs | tr ‘’ ‘,’| tr -d ‘\n’</td></tr>
<tr><td>My Jobs</td><td>Slurm</td><td>sacct -S now-7days -E now -P -n -a -o {comma separated list of fields[1]} -A {comma separated list of user’s allocations}</td></tr>
<tr><td>Performance Metrics</td><td>Slurm</td><td>sacct -u {username} -S now-100000days[2] -X -o JobID,Submit,Start,End</td></tr>
</table>

Discussion

Through further developments and optimizations, the dashboard can serve as a user-friendly alternative to command-line HPC usage. The dashboard is designed to be adaptable and scalable, allowing for easy configuration for use with other clusters. The data API and caching solution can be used by any HPC resource that utilizes Slurm as its job scheduler or easily adapted to other schedulers such as PBS. HPC centers

[1] jobid,jobname,user,account,start,end,elapsed,submit,planned,partition, state,reason,timelimit,reqmem,alloccpus,totalcpu,workdir,maxrss, maxdiskwrite,maxdiskread,nodelist,reqtres

[2] This represents jobs data from all time.



Service Unit (SU) Usage

allocation1 — Export
7325.4 SUs used / 50000 total (42674.6 left)

allocation1-gpu — Export
10.32 SUs used / 1000 total (989.68 left)

allocation2 — Export
1997463.07 SUs used / 2662225 total (664761.93 left)

allocation2-gpu — Export
5190.03 SUs used / 10828 total (5637.97 left)

allocation3 — Export
3303.05 SUs used / 200196 total (196892.95 left)

Disk Usage

home / x-user — View
Size → 5.4GB / 25.0GB

scratch / anvil — View
Size → 9.8GB / 100.0TB
Files → 0k / 1,000k files

projects / x-allocation1 ⚠ — View
Size → 1.6TB / 5.0TB
Files → 4,132k / 4,194k files

projects / x-allocation2 — View
Size → 4.3TB / 20.0TB
Files → 3,369k / 10,485k files

projects / x-allocation3 — View
Size → 61.2GB / 5.0TB
Files → 27k / 1,048k files

Announcements

Anvil Cluster Maintenance - Partial

May 13, 2023 8:00am - May 14, 2023 8:00am EDT

The Anvil system will have reduced capacity **Monday, May 13 from 8:00am - Tuesday, May 14 at 8:00am EDT** for scheduled maintenance. Some compute nodes will be powered off during the maintenance for some power work.

How does this maintenance impact you?...

Originally posted on May 7, 2023 4:46pm EDT

View All Announcements

Partition Status

Partition	State	Current Load
wholenode standard wide	UP	532/746 nodes busy (214 free)
shared	UP	68/250 nodes busy (182 free) 6726/32000 cores busy (25274 free)
highmem	UP	1/32 nodes busy (31 free) 64/4096 cores busy (4032 free)
debug	UP	6/17 nodes busy (11 free) 648/2176 cores busy (1528 free)
gpu gpu-debug	UP	15/16 nodes busy (1 free) 44/64 GPUs busy (20 free)

View Queue Limits

Job Queue

Name (ID)	Partition	State (Reason)
interactive (4567882)	shared	PENDING due to Priority
interactive (4567871)	shared	RUNNING

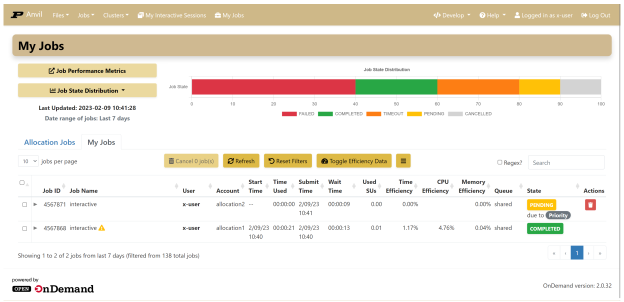
View All Jobs

Need help with Anvil?

powered by OPEN OnDemand

OnDemand version: 2.0.32

Fig. 2: Anvil dashboard homepage with widgets



My Jobs

Last Updated: 2023-02-09 10:41:28

Date range of jobs: Last 7 days

Job State Distribution

10 jobs per page

□ Regex?

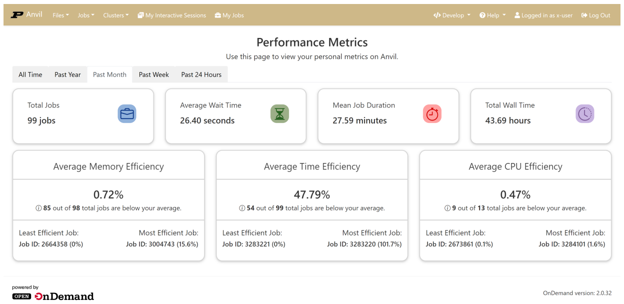
□	Job ID	Job Name	User	Account	Start Time	Time Used	Submit Time	Wait Time	Used SUs	Time Efficiency	CPU Efficiency	Memory Efficiency	Queue	State	Actions
□	4567871	interactive	x-user	allocation2	--	00:00:00	2/09/23 10:41	00:00:09	0.00	0.00%		0.00%	shared	PENDING due to Priority	🗑
□	4567868	interactive ⚠	x-user	allocation1	2/09/23 10:40	00:00:21	2/09/23 10:40	00:00:13	0.01	1.17%	4.76%	0.04%	shared	COMPLETED	

Showing 1 to 2 of 2 jobs from last 7 days (filtered from 138 total jobs)

powered by OPEN OnDemand

OnDemand version: 2.0.32

Fig. 3: *My Jobs* app with table of user's jobs



Performance Metrics

Use this page to view your personal metrics on Anvil.

Total Jobs: 99 jobs

Average Wait Time: 26.40 seconds

Mean Job Duration: 27.59 minutes

Total Wall Time: 43.69 hours

Average Memory Efficiency
0.72%
85 out of 98 total jobs are below your average.
Least Efficient Job: Job ID: 2664358 (0%)
Most Efficient Job: Job ID: 3004743 (15.6%)

Average Time Efficiency
47.79%
54 out of 99 total jobs are below your average.
Least Efficient Job: Job ID: 3283221 (0%)
Most Efficient Job: Job ID: 3283220 (101.7%)

Average CPU Efficiency
0.47%
9 out of 13 total jobs are below your average.
Least Efficient Job: Job ID: 2673861 (0.1%)
Most Efficient Job: Job ID: 3284101 (1.6%)

powered by OPEN OnDemand

OnDemand version: 2.0.32

Fig. 4: *Performance Metrics* app with job efficiency and wait time info

that use Open OnDemand can adapt our dashboard framework with modifications to the data sources as needed.

Acknowledgements

This work was supported in part by the National Science Foundation (NSF) grant no. 2005632 and the Anvil Research Experience for Undergraduates (REU) program.

References

[1] X. Carol Song, Preston Smith, Rajesh Kalyanam, Xiao Zhu, Eric Adams, Kevin Colby, Patrick Finnegan, Erik Gough, Elizabett Hillery, Rick Irvine, Amiya Maji, and Jason St. John. 2022. Anvil - System Architecture and Experiences from Deployment and Early User Operations. In Practice and Experience in Advanced Research Computing (PEARC '22). Association for Computing Machinery, New York, NY, USA, Article 23, 1–9. https://doi.org/10.1145/3491418.3530766

[2] Hudak et al., (2018). Open OnDemand: A web-based client portal for HPC centers. Journal of Open Source Software, 3(25), 622, https://doi.org/10.21105/joss.00622

[3] Jette, M.A., Wickberg, T. (2023). Architecture of the Slurm Workload Manager. In: Klusáček, D., Corbalán, J., Rodrigo, G.P. (eds) Job Scheduling Strategies for Parallel Processing. JSSPP 2023. Lecture Notes in Computer Science, vol 14283. Springer, Cham. https://doi.org/10.1007/978-3-031-43943-81

[4] K. D. Colby, D. T. Dietz, P. M. Smith and D. D. Cumberland, "Self-Service Queue and User Management in Shared Clusters," 2014 First International Workshop on HPC User Support Tools, New Orleans, LA, USA, 2014, pp. 22-31, doi: 10.1109/HUST.2014.5. keywords: Delays;Manuals;Engineering profession;Portals;Access control;Business;Conferences,

[5] Rice, S., Colby, K., (2022) Halcyon: Unified HPC Center Operations: An extensible web framework for resource allocation, documentation, and more., Practice and Experience in Advanced Research Computing (PEARC 2022).