The Journal of Open Source Software

SCAS dashboard: A tool to intuitively and interactively analyze Slurm cluster usage

Thomas Walzthoeni [1¶], **Bom Bahadur Singiali** [2], **N. William Rayner** [3], **Francesco Paolo Casale** [4,5,6], **Christoph Feest** [4,7], **Carsten Marr** [4], and **Alf Wachsmann** [2]

1 Core Facility Genomics, Helmholtz Zentrum München - German Research Center for Environmental Health, 85764 Neuherberg, Germany **2** Digital Transformation & IT, Helmholtz Munich, Helmholtz Zentrum München - German Research Center for Environmental Health, 85764 Neuherberg, Germany **3** Institute of Translational Genomics, Helmholtz Zentrum München - German Research Center for Environmental Health, 85764 Neuherberg, Germany **4** Computational Health Center, Helmholtz Zentrum München - German Research Center for Environmental Health, 85764 Neuherberg, Germany **5** Helmholtz Pioneer Campus, Helmholtz Zentrum München - German Research Center for Environmental Health, 85764 Neuherberg, Germany **6** School of Computation, Information and Technology, Technical University of Munich, Munich, Germany **7** Helmholtz AI, Helmholtz Zentrum München - German Research Center for Environmental Health, 85764 Neuherberg, Germany ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Daniel S. Katz

Reviewers:

- @aturner-epcc
- @apjez
- @verolero86

Submitted: 01 November 2023
Published: unpublished

Summary

Many organizations offer High Performance Computing (HPC) environments as a service, hosted on-premises or in the cloud. Compute jobs are commonly managed via Slurm (Yoo et al., 2003), but an intuitive, easy-to-use and interactive visualization has been lacking. To fill this gap, we developed a Slurm Cluster Admin Statistics (SCAS) dashboard. SCAS provides a means to analyze and visualize data of compute jobs and includes a feature to generate presentations for cluster users. It thus allows HPC stakeholders to easily analyze and identify bottlenecks of Slurm-based compute clusters in a timely fashion and provides decision-making support for managing cluster resources.

Statement of need

Slurm (Yoo et al., 2003) is an open-source cluster management and job scheduling system for Linux-based compute clusters and is widely used for High Performance Computing (HPC). It offers command line tools to export and analyze cluster use and various applications have been developed to monitor the current state of the cluster (e.g. live dashboards using Grafana (Dessalvi, 2021)). A feature rich tool for the analysis of cluster performance is Open XDMoD (Palmer et al., 2015) that supports various schedulers and metrics. Open XDMoD uses 3rd party software libraries that are not free for commercial use. Open OnDemand (Hudak et al., 2018) allows users to access a HPC cluster using a web portal, it provides various apps to facilitate HPC usage and can integrate the Open XDMoD for usage statistics. Both, Open XDMoD and Open OnDemand require continuous support and extensive configurations and therefore, intuitive, responsive, easy-to-install and easy-to-use applications that enable HPC administrators and managers to analyze and visualize cluster usage in detail and over time are highly complementary. This information is crucial to identify bottlenecks in compute clusters and make informed strategic decisions regarding their future development.

To address this, we developed the Slurm Cluster Admin Statistics (SCAS) dashboard, a scalable and flexible dashboard application to analyze completed compute jobs on a Slurm-based cluster. The dashboard offers various statistics, visualizations, and insights to HPC stakeholders and cluster users. Additionally, we engineered the software to have a low-memory footprint and to be fast and responsive to user queries. The software stack is provided in an easy-to-use and easy-to-deploy manner using docker containers and a docker-compose implementation.

Description

SCAS Dashboard overview

The SCAS dashboard architecture consists of a nginx web server as a router (reverse proxy), a frontend based on R-Shiny (Chang et al., 2023; Chang & Borges Ribeiro, 2021; R Core Team, 2023), a backend based on Python using the Django REST framework to provide an API, and a PostgreSQL database as backend (see Figure 1). The dashboard is intended for the HPC stakeholders and therefore includes secure user authentication. The frontend is a user-friendly interface for filtering and visualizing the Slurm data. The backend provides an admin interface via Python Django Admin and a web API that is used by both the frontend and a script for uploading new data. Additionally, the backend creates a daily index of the data, enabling the software to maintain a low memory footprint while being fast and responsive. Furthermore, a presentation can be generated automatically and viewed by various stakeholders, including the cluster users, via a web browser.

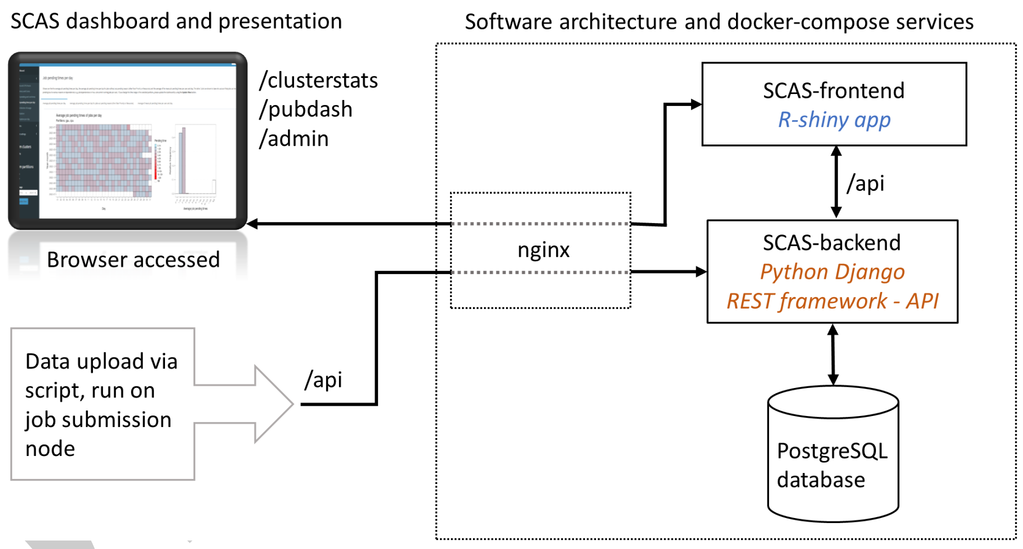

Figure 1: Architecture of the SCAS dashboard. The dashboard and the presentation are accessed by the user through a web browser. New data can be uploaded to the SCAS dashboard by executing a script that regularly fetches the latest data from a job submission node. On the server side, the architecture is organized into separate components (shown in dashed box): nginx (reverse proxy), SCAS-frontend, SCAS-backend and PostgresSQL database. A docker-compose implementation of the services is provided.

SCAS dashboard workflow

Completed compute jobs and available node configurations are submitted to the SCAS-backend API with a script that utilizes the Slurm's *sacct* tool. This script can be run as a daily or weekly *cron* job on a job submission node. The backend then generates the daily statistics that are stored in the database. This preprocessed indexed data enables the app to have a low memory footprint and high responsiveness, as no calculations are required when the data is fetched from the API. Upon filtering a date range in the frontend, a request is sent to the backend which retrieves the data for the selected days and aggregates the statistics to generate the visualizations.

Frontend – dashboard user interface

Figure 2 displays some example views of the user interface. The date range, the cluster, and the partitions that should be analyzed can be selected from the menu (Figure 2a). Data tables and visualizations are then updated accordingly and displayed to the user.

For the selected date range, the visualizations include:

- Number of jobs, CPU and GPU hours per month (Figure 2b,c)
- Memory and cores requested by users, displayed as contingency graphs
- Average job pending and runtimes per month and per day (Figure 2d,e,f)
- Distribution of CPU hours used vs. the percentage of users
- Total cluster utilization per day and per month, individual node utilization per month, summaries of utilization per CPU/GPU or memory type of the nodes per month (Figure 2g)

The data can also be downloaded for use in spreadsheet applications.

Frontend – automated presentation

For presenting key figures to the cluster users, a feature is available to generate a browser-based presentation in carousel mode. The presentation is auto-updated and customization settings are available via the admin interface.

SCAS dashboard - example use case

To exemplify an analysis with the SCAS dashboard, we assumed that users reported longer pending times for GPU resources in recent months. We have simulated this case by increasing the number of GPU jobs (and their pending times) for GPU servers, with 16 GPUs, over a time frame of 1 year. As shown in Figure 2b,c, the increase in the number of GPU jobs and CPU hours for the GPU partition is visible and confirms the assumption. By inspecting the pending times per day (Figure 2d) there is a general, unbiased increase of the pending times visible for the last few months. From Figure 2e we can then see an increase of the pending times for the GPU partition for the previous 6 months. Figure 2f shows that the increase of the pending times is only seen for servers with >10 GPUs, and the utilization of the nodes with 16 GPUs has increased while those with 2 and 4 GPUs were stable (Figure 2g). This analysis can be used to draw concrete conclusions. In this case to either inform the users that resources are available if up to 4 GPUs are requested, or to make the decision to invest in new GPU servers to achieve shorter pending times and higher throughput.

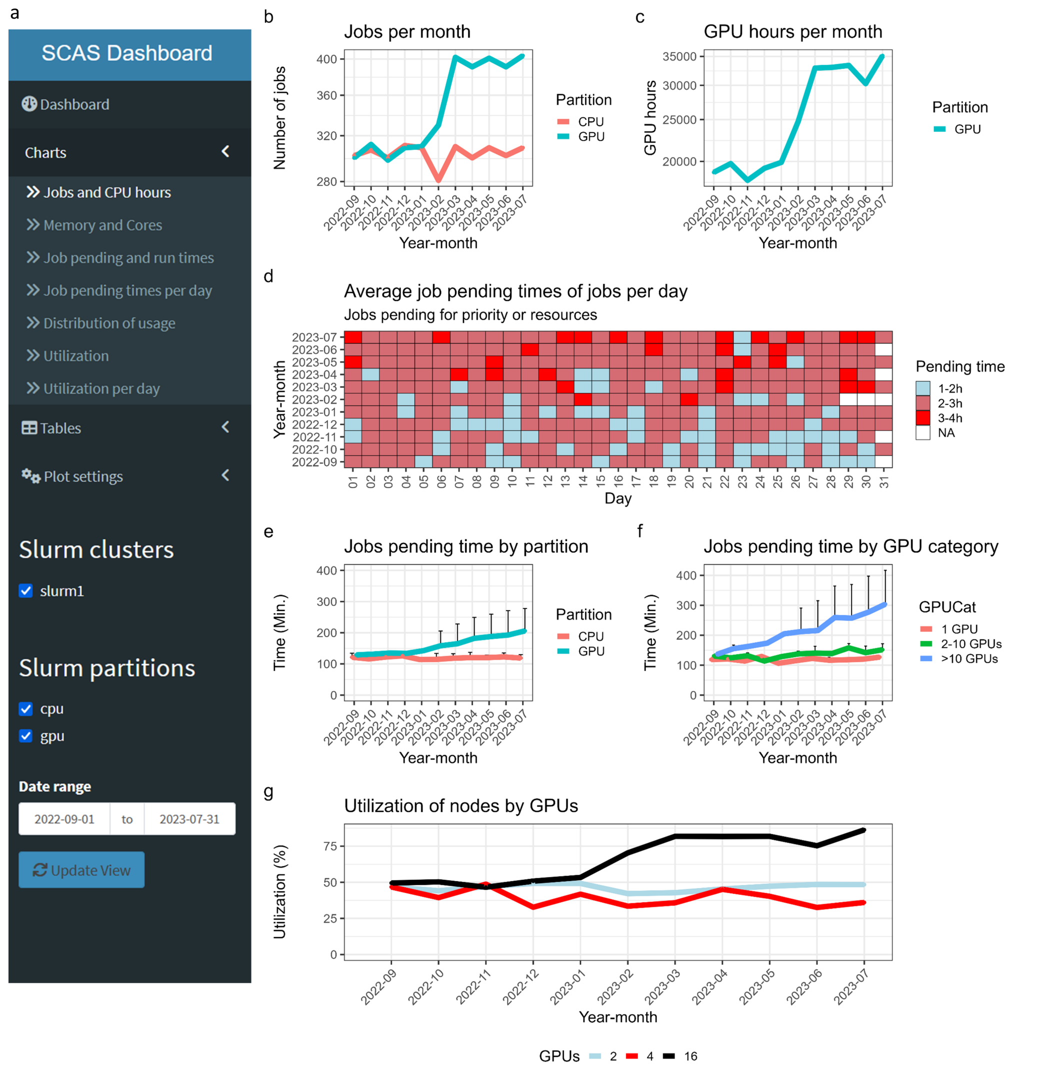

Figure 2: **a**. User interface of the SCAS Dashboard featuring navigation and the selection menu. The central panel displays statistics and graphics. **b**. Line plot showing the jobs run per month. **c**. Line plot showing the GPU hours per month. **d**. Heatmap plot showing the average daily pending times of the jobs. **e**. Line plot with the average jobs pending times per month. The positive error bars indicate the standard deviation. **f**. Line plot with the average jobs pending times per month separated by GPU categories. The positive error bars indicate the standard deviation. **g**. Line plot showing the utilization of nodes with different numbers of GPUs.

Conclusion and Availability

The SCAS dashboard enables rapid and responsive analysis of Slurm-based cluster usage. This allows stakeholders: I) to identify current bottlenecks of CPU and GPU utilization, II) to make informed decisions to adapt SLURM parameters in the short term and III) to support strategic decisions, all based on user needs. The SCAS dashboard, code, and the documentation are hosted on a publicly available GitHub repository (https://github.com/Bioinformatics-Munich/scas_dashboard). The repository also contains a docker-compose file for rapid deployment and testing of the software, as well as a program to generate test data for the dashboard.

Acknowledgements

We acknowledge the Institute of Computational Biology (Prof. Dr. Dr. Fabian Theis) at Helmholtz Munich for supporting the development of the software. We thank Dr. Bastian Rieck, Helmholtz Munich, for valuable contributions and comments to the manuscript.

Chang, W., & Borges Ribeiro, B. (2021). *Shinydashboard: Create dashboards with 'shiny'*. http://rstudio.github.io/shinydashboard/

Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., & Borges, B. (2023). *Shiny: Web application framework for r*.

Dessalvi, M. (2021). *SLURM Dashboard*. https://grafana.com/grafana/dashboards/4323.

Hudak, D., Johnson, D., Chalker, A., Nicklas, J., Franz, E., Dockendorf, T., & McMichael, B. L. (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

Palmer, J. T., Gallo, S. M., Furlani, T. R., Jones, M. D., DeLeon, R. L., White, J. P., Simakov, N., Patra, A. K., Sperhac, J., Yearke, T., Rathsam, R., Innus, M., Cornelius, C. D., Browne, J. C., Barth, W. L., & Evans, R. T. (2015). Open XDMoD: A tool for the comprehensive management of high-performance computing resources. *Computing in Science & Engineering*, *17*(4), 52–62. https://doi.org/10.1109/MCSE.2015.68

R Core Team. (2023). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Yoo, A. B., Jette, M. A., & Grondona, M. (2003). SLURM: Simple linux utility for resource management. In D. Feitelson, L. Rudolph, & U. Schwiegelshohn (Eds.), *Job scheduling strategies for parallel processing* (pp. 44–60). Springer Berlin Heidelberg. https://doi.org/10.1007/10968987_3