SciApps: a Bioinformatics Workflow Platform Powered by XSEDE and CyVerse

Liya Wang
Cold Spring Harbor Laboratory
Cold Spring Harbor, NY
USA
wangli@cshl.edu

Zhenyuan Lu
Cold Spring Harbor Laboratory
Cold Spring Harbor, NY
USA
luj@cshl.edu

Peter Van Buren
Cold Spring Harbor Laboratory
Cold Spring Harbor, NY
USA
vanburen@cshl.edu

Doreen Ware
Cold Spring Harbor Laboratory
Cold Spring Harbor, NY
USDA ARS, USA
ware@cshl.edu

ABSTRACT

SciApps[1], a lightweight bioinformatics workflow system powered by the CyVerse infrastructure, uses the Agave Science API to manage the entire cycle of analysis jobs between XSEDE HPC and the CyVerse Data Store. SciApps provides a graphical user interface for job submission, workflow creation, and management of both jobs and workflows. Each reproducible workflow, along with all inputs and results, is retrievable with a unique ID.

CCS CONCEPTS

• **Information systems** → Data management systems; computing platforms • **Software and its engineering** → 3-tier architectures

KEYWORDS

CyVerse, Agave science API, workflow, cloud computing, infrastructure, science gateway

ACM Reference format:

L. Wang, Z. Lu, P. Van Buren, and D. Ware. 2018. SciApps: a Bioinformatics Workflow Platform Powered by XSEDE and CyVerse. In *Proceedings of PEARC18, Pittsburgh, PA, USA, July 22-26, 2018,* 7 pages. https://doi.org/10.1145/3219104.3219109

[1] https://de.sciapps.org

© 2018 Association for Computing Machinery. ACM acknowledges that this contribution was authored or co-authored by an employee, contractor or affiliate of the United States government. As such, the United States Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.

PEARC '18, July 22–26, 2018, Pittsburgh, PA, USA
© 2018 Association for Computing Machinery
ACM ISBN 978-1-4503-6446-1/18/07…$15.00
https://doi.org/10.1145/3219104.3219109

1 INTRODUCTION

CyVerse [1], which is funded by the National Science Foundation (NSF), aims to provide life scientists with a powerful computational infrastructure to handle big datasets and complex analyses, thereby enabling data-driven discovery. The foundation of the CyVerse infrastructure is the iRODS [2]-based Data Store, which provides users great flexibility and control over their data. The CyVerse Discovery Environment (DE) [3] provides a graphical interface for sophisticated data management, along with hundreds of curated applications that run on either CyVerse Condor cluster [4] or XSEDE resources [5] at the Texas Advanced Computing Center (TACC). The XSEDE resources are leveraged through the Agave Science API [6], which virtualizes TACC clusters as execution systems and the Data Store as a storage system. To date, CyVerse has supported ~50,000 users in management and analysis of more than 2.5 petabytes (PB) of data.

To enhance support for complex analysis of big data with CyVerse and XSEDE, and fully utilize the modular Agave apps developed by the CyVerse project and its community members, we developed SciApps [7], a workflow management system, to chain these apps into automated workflows. Like the DE, SciApps handles the interaction between XSEDE and the Data Store for each analysis, but also has several enhanced features: a graphical user interface for building, modifying, and sharing scientific workflows; automation of individual Agave app submissions; and a graphical workflow diagram for visualizing real-time job status and relationships among individual analysis steps.

Comparing with SciApps (www.sciapps.org) that utilizes a local federation system [8], in this work, we have developed SciApps2 (de.sciapps.org) that operates entirely in the cloud. It uses the CyVerse Data Store for data storage and management, XSEDE for computing, and CyVerse Data Common (DC) landing pages for

displaying data and metadata (http://datacommons.cyverse.org/). For both platforms, workflows can now be shared with an URL containing the unique workflow ID. SciApps2 also supports MaizeCODE, an NSF-funded project aimed at creating a comprehensive reference encyclopedia for maize. Through SciApps2' ready-to-use platform, breeders and plant scientists can easily reproduce the MaizeCODE analysis, use the results from any step to perform downstream analysis, apply the same workflow to new datasets, or extend a workflow with new analyses.

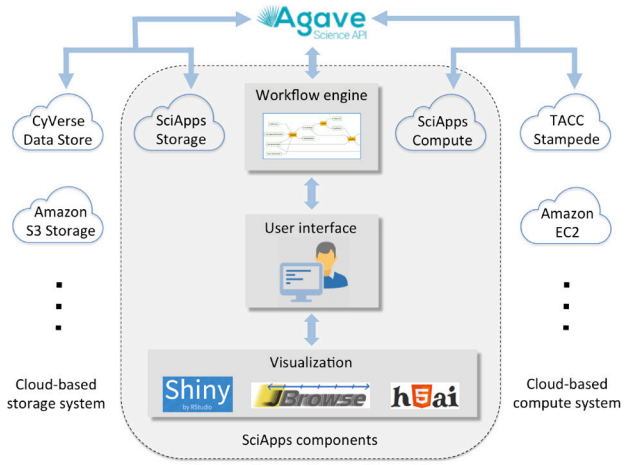

Figure 1: Components of SciApps and connections to cloud-based systems through the Agave Science API.

2 BACKGROUND

2.1 CyVerse Platforms

CyVerse has several platforms to support various types of data analysis. Analysis with DE or SciApps is submitted using important modular components called applications or apps. In the following sections, we will briefly discuss how apps are built with Agave and integrated into DE. In addition to these app-based platforms, CyVerse also developed Atmosphere [9], a cloud computing platform for command-line savvy users.

2.2 Agave App

To build an Agave app, users either need to have their own XSEDE allocation or can be added to CyVerse's allocation. Subsequently, each user can create a private virtual execution system to test the app they have built. To run an app, users need to create a wrapper template that calls the executable code, which can be pre-built binaries or Singularity images [10]; the latter is preferable for tools that have complicated dependencies. An app JSON file is required to define the execution system, the storage system, the path to the wrapper script, inputs, parameters, and outputs, along with other metadata describing the app. The wrapper script (or template file) is needed to pass the inputs and parameters from the app JSON to the tool, and to execute the tool. After testing, users can share the app with either selected users or the public by contacting the CyVerse team. More details on building Agave apps are available online: https://cyverse.github.io/cyverse-sdk/docs/cyversesdk.html

2.3 DE App

Both DE and SciApps can automatically render a web-based app form from the Agave app JSON file. In addition, DE supports Docker-based app integration. First, users need to create a Dockerfile, a text document that contains all the commands a user can call on the command line to assemble an image. For DE app integration, the Dockerfile is expected to include commands for downloading required packages for a particular tool from reliable online resources, installing them, and defining an ENTRYPOINT that will allow the Docker image to be run like an executable. The user can then build and test the Docker image with the Dockerfile. If the test is successful, the user can request installation of the Dockerized tool in DE and create the app interface in DE. After testing, the app can also be shared with either selected users or the public.

3 ARCHITECTURE

The architecture of SciApps is shown in Fig. 1. Both platforms consist of a web interface for executing apps and building workflows, a workflow engine to streamline and automate job submissions, a storage system, a computing system, and a web server for various visualization services. While SciApps is optimized to handle the analysis of large-scale datasets locally with a CyVerse federation system at CSHL [8], SciApps2 is scaled by leveraging cloud resources for storage and computing. Both platforms share the same design of front-end interface and workflow construction. Therefore, in following sections, we will use SciApps instead of SciApps2 to describe the platform.

3.1 Web Interface

The SciApps web interface has four areas, head navigation menu and three panels below it, as shown in Fig. 2. Apps in the left panel are categorized according to the EDAM ontology [11]. Apps are also searchable by names. The app form is loaded in the main panel with default (or previously used) inputs and parameters (if reloaded from the history panel), and once submitted, the analysis/job history is displayed in the history panel, which can be selected to build a workflow.

All app categories are closed by default, as shown in Fig. 2. The app search function is interactive: when any letters are typed into the search box, categories with a matched app or apps will be expanded with matches. Clicking on an app will bring up the app form in the main panel, along with a short description of the app below the app form. For each job in the History panel, there are three icons next to the job name: from left to right, a checkbox for adding the job into a workflow, an info icon for a popup window with detailed job information, and a reload icon for loading the app form with the same inputs and parameters. The History panel

only displays outputs predefined in the JSON file of the app, as shown for step 1, 3, 4 in Fig. 2 (expanded upon clicking on job name). If the user aims to build a workflow, these predefined outputs can be used as inputs for subsequent analysis tasks. For further explanation, check Section 4: Implementation, below.

With no jobs displaying in the History panel, the info message on the top of the panel prompts users to start testing with a public workflow. When loading a workflow (public or private), all job histories will be loaded into the History panel (as shown in Fig. 2), and app forms with parameters used in the workflow will be loaded into the main panel in sequential order (not shown). The message is also automatically changed to prompt users to build a workflow by selecting two or more jobs when the History panel is not empty (for example see Fig. 2). Clicking on the link embedded in the info message (or from the top bar, "Workflow" then "Build a Workflow") will bring up the workflow builder page in the main panel. For convenience, the user has the option to select or deselect all jobs from the History panel. Completed jobs can be saved as 'pseudo' workflows if the user wants to load them later (and/or add more steps) to build a real workflow. If the user wants to start building a new workflow, simply refreshing the browser window will clear out the history panel with new jobs. More details can be found in the platform guide (https://www.sciapps.org/page/help).

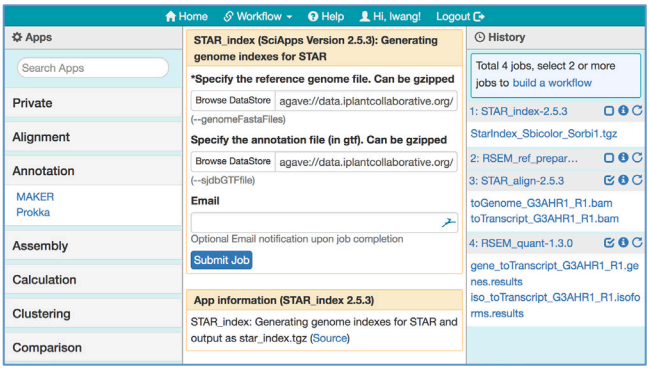

Figure 2: SciApps web interface showing a four-step RNA-Seq quantification workflow loaded in the history panel, the app form of the first step loaded in the main panel, the results from step 1, 3, and 4 clicked and expanded in the history panel, and step 3 and 4 selected to build a new workflow.

3.2 Authentication

To grant access to multiple cloud-based systems, SciApps adopts the CyVerse Central Authentication Service (CAS) for authorization. When a user logs in, they are directed to the CyVerse user portal to enter their username and password, and are then redirected to SciApps if successfully authenticated. Through authentication, the CyVerse username is captured by SciApps, for two reasons: first, it will be used to direct SciApps to the sci_data folder of the user that logged in; and second, once an analysis job is submitted, the job is shared with the user through the Agave API. If necessary, users can check the detailed job information through Agave's CyVerse SDK (https://cyverse.github.io/cyverse-sdk). Sharing is needed here because SciApps2 uses a designated CyVerse user account (a superuser) to execute apps and workflows. The superuser also gains full access to each user's sci_data folder once the folder is created, either manually or automatically when users enable their SciApps service from the CyVerse user portal.

SciApps adopts a platform-centric approach by designating a superuser to manage the entire analysis cycle. The major advantage of such an approach, in contrast to a user-centric approach in which each login user is in charge of everything, is that it greatly simplifies the management of apps, systems, analysis jobs, inputs, and workflows. For example, when user A wants to share a workflow with user B, the only thing they need to share is a workflow URL; there is no need to grant user B permissions to any workflow components, such as apps, systems where the apps are defined to execute, analysis jobs, or user A's sci_data folder. User B can load the workflow in SciApps for execution through the superuser, who has gained accesses to all of the workflow components either through sharing (apps, systems, sci_data folder) or being the owner (of all analysis jobs).

For security reasons, workflows and related output data are protected with randomly generated ids or folder names, therefore only the superuser and the owner of workflows can access them. However, users need to realize that, when a workflow is shared, there is no additional authentication required to access all data associated with the workflow.

3.3 Automated Workflows

When users submit an analysis task, the job history is recorded in the History panel, and a workflow can be built by selecting two or more jobs using the checkboxes (Fig. 2). The input/output relationships among individual tasks are built by tracing the origin of intermediate output, which is available in the job history metadata of the Agave API. Once built, a graphic workflow diagram is shown for verification. After visual inspection, users can choose to save the workflow.

The workflow diagram is interactive. The user can click on both data and app nodes to check related metadata and full names of apps and data files (long names are truncated in the diagram). For a running workflow, the diagram provides real-time job status updates through automatic updating the color of the app node. Alternatively, the status of an individual job can be accessed by clicking on the info icon. SciApps workflows are implemented as directed acyclic graphs. The execution of a step is only dependent on the availability of its required input(s), making it possible to exploit parallelism.

SciApps supports management of both workflows and analysis jobs. On the 'My Jobs' page, analysis jobs are listed with name, submit time, end time, and status. All jobs are searchable by names and can be deleted or loaded into the History panel to build workflows. On the 'My Workflows' page, completed or running workflows are listed with names and descriptions, and are searchable by name, description, and workflow ID. Four operations are supported for workflows: loading the workflow to re-run, visualizing the workflow diagram directly, obtaining a link containing the workflow ID for sharing the workflow, and deleting the workflow from the user's account. Both loading and visualizing will load jobs into the History panel if they are not yet loaded, but the visualizing operation will not load the app forms.

4 IMPLEMENTATION

The back end of SciApps was built using Perl and MySQL database, and the front-end was built with React, an open-source JavaScript library. The workflow engine uses the MySQL database to track job status and perform the submission of a job once its inputs are ready. Most components of the SciApps web interface are rendered from JSON data, and the schemas of all JSON data are custom-designed for fast rendering or easy sharing. One exception to this is the app JSON schema, which is inherited from the Agave API. In addition to defining inputs and parameters for rendering the app form, the app JSON specifies the execution system where the app will be executed. This makes it possible for SciApps to submit jobs to both local and cloud-based systems. To add a new app, storage, or execution system, the user can follow the CyVerse SDK tutorial (https://github.com/cyverse/cyverse-sdk). The workflow diagram was built using Mermaid (https://knsv.github.io/mermaid/) and modified for reflecting real-time job status via dynamically changing colors. The latter information is acquired through Agave API's Webhook notification for jobs, which is also used to automatically update the MySQL database and execute multi-step workflows.

For an Agave app to be compatible with SciApps, its outputs, which are optional by default, must be defined within the app JSON files, and the output IDs must be manually appended to output file names as prefixes in the wrapper script. The reason for this is that output IDs are NOT passed to the wrapper script by the Agave API, but the SciApps workflow engine needs the output ID to build the connection between two analysis steps. The majority of Agave apps supported by SciApps are constructed by deriving output file names from input file names (e.g., prefix_inputFileName), which is done with the wrapper script of each Agave app. Such modifications ensure that the SciApps workflows are not only reproducible, but also reusable for different datasets, without requiring the user to manually process the output file names.

SciApps uses the CyVerse DC landing pages to display both inputs and outputs. Users need to decide whether to make private input data public or not. SciApps archives outputs into users' sci_data folder and makes them publically available, but the data are protected by a randomly generated job folder name. The DC page displays both metadata and a preview of the first 8 KB of text-based files, and also provides a link for opening the file in CyVerse DE. If the file size is smaller than 2 GB, a direct download link is provided; otherwise, a link is provided to instructions for acquiring the file through either CyberDuck (https://cyberduck.io) or icommands (https://docs.irods.org/master/icommands/user).

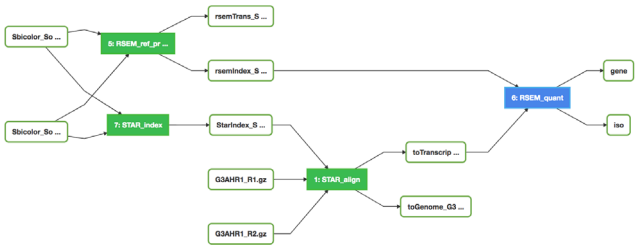

Figure 3: **RNA-Seq workflow showing the reference genome and annotation file being passed to two apps, RSEM_ref_prepare and STAR_index, to build indexes that will speed up alignments with the STAR_align app and quantification with the RSEM_quant app, with the color of app node representing status of the analysis task: Pending (yellow, not shown), Running (blue), Completed (green), or Failed (red, not shown).**

5 MAIZECODE DATA AND WORKFLOWS

The MaizeCode project will generate hundreds of experiments aimed at identifying functional elements in the Maize Genome. All data sets are being processed with SciApps workflows and released to the public. As an example, Fig. 3 shows the workflow for RNA-Seq, which uses next-generation sequencing (NGS) to reveal the presence and quantity of RNA in a biological sample at a given moment. The workflow uses four apps built from two different tools: STAR [12] for alignment of short reads to the genome, and RSEM [13] for quantifying RNA expression levels. The metadata are attached to the input data, and the relationships among data sets are captured by the workflow. Given a unique workflow ID (which is attached to the input data), one can easily check how an experiment is designed and how the data are analyzed, and if needed reproduce the analysis using XSEDE resources.

By design, every experiment is organized as a workflow on the SciApps platform. For RNA-Seq, the most common type of downstream analysis is comparing expression levels between two experiments. This can be achieved by loading both workflows and feeding their quantification outputs (output of RSEM_quant) to RSEM_de, a downstream app, for differential expression analysis (DEA). With a large number of experiments, It is critical to ensure that all experiments are processed consistently before performing DEA or other downstream analyses. Using SciApps not only ensures consistency but also conveniently chains together all

inputs, associated metadata, results, and analysis jobs, making both data and analysis findable, accessible, interoperable, re-usable, and reproducible.

MaizeCode data are described with rich metadata that are attached to raw reads before they are submitted to the NCBI Short Read Archive (SRA) [14] via CyVerse's SRA pipeline (https://learning.cyverse.org/projects/sra_submission_quickstart). Both data and metadata are searchable within CyVerse DE. As mentioned above, users can also search workflows by name, description, or IDs on the SciApps platform.

6 USING SCIAPPS

SciApps workflows are available to all 50,000 CyVerse users and CyVerse account is free for all users. To develop their own workflows, users can utilize the SciApps2 site (de.sciapps.org). For workflow development and testing, users need to have their own XSEDE allocation or they can contact CyVerse to be added to CyVerse's allocation to access over 1 million shared CPUs from XSEDE clusters at TACC. Additionally, users can set up their own SciApps with an XSEDE allocation or a local cluster.

7 RELATED WORKS

SciApps' front-end interface is designed with three panels aligned similarly to those of the Galaxy [15] and GenePattern [16] platforms. On the back end, SciApps is designed to be compatible with Agave API and CyVerse CI and is flexible in regard to data placement and mixing of local and cloud-based computing resources. Specifically, the Agave job JSON specifies where to retrieve the data and where to archive the results, whereas the Agave app JSON, in addition to descriptions of each step in the Common Workflow Language format or CWL [17], specifies where to execute an app and how many processors and how much RAM are needed. Therefore, although the SciApps workflow JSON is designed with CWL in mind, it is not interoperable with CWL. However, for single-node jobs, its possible to convert SciApps workflows into CWL format by combining the workflow JSON with the Agave app JSON and job JSON; however, this will eliminate the flexibility regarding where and how to execute the app and where the data can be placed. Another challenge is that CWL does not yet support Singularity containers [7], which are adopted by SciApps applications to enable simple switching between local and cloud-based computing clusters.

To build a workflow, CyVerse DE allows the user to define an output file name through an output parameter, and then uses the value of the parameter to chain apps together as a workflow. This design allows the user to manually retain the names of the original samples. When analyzing new datasets, users need to manually change output file names because, if the output file names are fixed, a similar collision will occur for datasets as described above for replicates, making it hard to reuse the same workflow on different datasets. On the contrary, by automatically retaining the input file names in an app's wrapper script, SciApps workflows can be applied to different datasets without requiring manual changes.

In summary, SciApps automates the execution of series of Agave apps, which is not supported by the DE. Comparing with Galaxy and GenePattern, SciApps has the advantage of supporting mixing both could and local computing resources in the same workflow, which is sometimes critical for projects that need large scale computation or collaboration.

8 CONCLUSIONS

SciApps provides a web-based workflow platform accessible to all CyVerse users for automating the execution of modular Agave apps on XSEDE resources. In addition to its reproducibility, SciApps workflow can be used to organize both data and metadata, and to process a large amount of data either remotely in the cloud or locally; the latter has the benefit reducing the need for cross-country data transfers.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation (DBI-1265383 and 1445025).

REFERENCES

[1] Goff, S.A. et al. 2011. The iPlant Collaborative: Cyberinfrastructure for Plant Biology. Frontiers in plant science. 2, (Jul. 2011), 34.
[2] Xu, H. et al. 2017. iRODS Primer 2: Integrated Rule-Oriented Data System. Synthesis Lectures on Information Concepts, Retrieval, and Services. 9, 3 (2017), 1–131.
[3] Oliver, S.L. et al. 2013. Using the iPlant Collaborative Discovery Environment. Current Protocols in Bioinformatics.
[4] Thain, D. et al. 2005. Distributed computing in practice: the Condor experience. Concurrency and computation: practice & experience. 17, 2-4 (2005), 323–356.
[5] Towns, J. et al. 2014. XSEDE: Accelerating Scientific Discovery. Computing in science & engineering. 16, 5 (2014), 62–74.
[6] Dooley, R. et al. 2012. Software-as-a-Service: The iPlant Foundation API. 5th IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS). IEEE (2012).
[7] Wang, L. et al. 2018. SciApps: A cloud-based platform for reproducible bioinformatics workflows. Bioinformatics (2018), https://doi.org/10.1093/bioinformatics/bty439
[8] Wang, L. et al. 2015. Architecting a Distributed Bioinformatics Platform with iRODS and iPlant Agave API. 2015 International Conference on Computational Science and Computational Intelligence (CSCI) (2015).
[9] Skidmore, E. et al. 2011. iPlant atmosphere. Proceedings of the 2011 ACM workshop on Gateway computing environments - GCE '11 (2011).
[10] Kurtzer, G.M. et al. 2017. Singularity: Scientific containers for mobility of compute. PloS one. 12, 5 (May 2017), e0177459.
[11] Ison, J. et al. 2013. EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. Bioinformatics . 29, 10 (May 2013), 1325–1332.
[12] Dobin, A. et al. 2012. STAR: ultrafast universal RNA-seq aligner. Bioinformatics . 29, 1 (2012), 15–21.
[13] Li, B. and Dewey, C.N. 2011. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC bioinformatics. 12, 1 (2011), 323.
[14] Leinonen, R. et al. 2010. The Sequence Read Archive. Nucleic acids research. 39, Database (2010), D19–D21.
[15] Giardine, B. et al. 2005. Galaxy: a platform for interactive large-scale genome analysis. Genome research. 15, 10 (Oct. 2005), 1451–1455.
[16] Reich, M. et al. 2006. GenePattern 2.0. Nature genetics. 38, 5 (May 2006), 500–501.
[17] Common Workflow Language: 2016. https://doi.org/10.6084%2Fm9.figshare.3115156.v2.