HHS Public Access

Author manuscript

Pract Exp Adv Res Comput 2018 (2018). Author manuscript; available in PMC 2019 October 09.

Published in final edited form as:

Pract Exp Adv Res Comput 2018 (2018). 2018 July ; 2018: . doi:10.1145/3219104.3219141.

Building a Science Gateway For Processing and Modeling Sequencing Data Via Apache Airavata

Zhong Wang,
Baker Institute for Animal Health, College of Veterinary Medicine, Cornell University, Ithaca, NY, zw355@cornell.edu

Marcus A. Christie,
Science Gateways Research Center, Pervasive Technology Institute, Indiana University, Bloomington, IN, machrist@iu.edu

Eroma Abeysinghe,
Science Gateways Research Center, Pervasive Technology Institute, Indiana University, Bloomington, IN, eabeysin@iu.edu

Tinyi Chu,
Graduate field of Computational Biology, Cornell University, Ithaca, NY, tc532@cornell.edu

Suresh Marru,
Science Gateways Research Center, Pervasive Technology Institute, Indiana University, Bloomington, IN, smarru@iu.edu

Marlon Pierce,
Science Gateways Research Center, Pervasive Technology Institute, Indiana University, Bloomington, IN, marpierc@iu.edu

Charles G. Danko
Baker Institute for Animal Health and Department of Biomedical Sciences, College of Veterinary Medicine, Cornell University, Ithaca, NY, cgd24@cornell.edu

Abstract

The amount of DNA sequencing data has been exponentially growing during the past decade due to advances in sequencing technology. Processing and modeling large amounts of sequencing data can be computationally intractable for desktop computing platforms. High performance computing

CCS CONCEPTS
• **Applied computing** → **Computational biology**; *Recognition of genes and regulatory elements*;

ACM Reference Format:
Zhong Wang, Marcus A. Christie, Eroma Abeysinghe, Tinyi Chu, Suresh Marru, Marlon Pierce, and Charles G. Danko. 2018. Building a Science Gateway For Processing and Modeling Sequencing Data Via Apache Airavata. In *PEARC '18: Practice and Experience in Advanced Research Computing, July 22–26, 2018, Pittsburgh, PA, USA*, editor A, editor B, and editor C (Eds.). ACM, New York, NY, USA, Article 4, 7 pages. https://doi.org/10.1145/3219104. 3219141

(HPC) resources offer advantages in terms of computing power, and can be a general solution to these problems. Using HPCs directly for computational needs requires skilled users who know their way around HPCs and acquiring such skills take time. Science gateways acts as the middle layer between users and HPCs, providing users with the resources to accomplish compute-intensive tasks without requiring specialized expertise. We developed a web-based computing platform for genome biologists by customizing the PHP Gateway for Airavata (PGA) framework that accesses publicly accessible HPC resources via Apache Airavata. This web computing platform takes advantage of the Extreme Science and Engineering Discovery Environment (XSEDE) which provides the resources for gateway development, including access to CPU, GPU, and storage resources. We used this platform to develop a gateway for the dREG algorithm, an online computing tool for finding functional regions in mammalian genomes using nascent RNA sequencing data. The dREG gateway provides its users a free, powerful and user-friendly GPU computing resource based on XSEDE, circumventing the need of specialized knowledge about installation, configuration, and execution on an HPC for biologists. The dREG gateway is available at: https://dREG.dnasequence.org/.

Keywords

Science gateway; cloud computing; software-as-a-service; sequencing data; Apache Airavata; Next Generation Sequencing

1 INTRODUCTION

Next Generation Sequencing (NGS) technology is now widely used in the biological sciences, and is fundamental to the field of genomics [10, 16, 20], epigenomics [17, 32] and transcriptional regulation [23, 26]. Rapid NGS technology development has dramatically reduced the cost and increased the throughput of DNA sequence-based functional assays. Desktop computer systems and even small-scale servers are becoming overpowered by the enormous amount of data generated by NGS experiments. Various methods have been developed to address the increasing demand for computing power. On the software perspective, an increasing number of bioinformatics tools that take advantage of multicore/multithread CPUs to provide more efficient computation have begun to emerge [8, 28]. On the hardware side, high performance computing (HPC) resources equipped with flexible task schedulers, such as PBS and SLURM, have greatly sped up computation. For some extremely computationally intensive applications, such as those using machine learning to predict gene expression patterns [11, 33], identify DNA sequences with regulatory activity [5, 15], and performing molecular dynamics simulations [6, 35], even multicore CPU-based HPCs are being outclassed by GPU-based platforms. In the long run, there will be an increasing complexity in both the software and hardware platforms, as new computational technologies are applied to biological problems.

DNA sequence regions known as transcript regulation elements (TREs), which include promoters and enhancers, are critical components of the genetic regulatory programs of all organisms. We have recently introduced a software tool to identify the location of TREs called the Detection of Regulatory Elements using GRO-seq data (dREG) [15]. dREG

identifies TREs using a multiscale summary of genomic data derived using an experimental technique called global run-on and sequencing as input (including GRO-seq [14], PRO-seq [21], or ChRO-seq [12] data). Genomic data signals are analyzed using a support vector regression (SVR) to detect the characteristic patterns of transcription at TREs. The software allows for high-quality predictions of TREs for any cell type with existing GRO-seq, PRO-seq or ChRO-seq data.

Recently, we have made extensive improvements on the dREG model and implementation that have together significantly increased the sensitivity of TRE identification [37]. When training the new model, we strategically increased the number of training samples (1.65M samples with 360 features), which resulted in an SVR model that contains a large amount of support vectors. The newly trained model yields TREs with a higher resolution and improves accuracy for bona-fide TRE regions. In comparison to alternative methods recently developed for the same task [7], dREG has a much higher sensitivity at the same false discovery rate (FDR). Despite the improved performance, using the new dREG model requires large-scale computing resources, which overwhelms even the multicore CPU clusters. To accelerate the computation, dREG employs a GPU-based SVM package [36]. Using NVIDIA Tesla K80 or NVIDIA Tesla P100-comparable GPU and 16 CPU cores for data I/O, a typical run of our implementation identifies TREs within 6 hours. The GPU-based dREG made the prediction of high quality TRE regions feasible.

Installation and setup of a GPU server is, however, not straightforward for most biologists, nor is it economical if the server is only made for a small amount of dREG runs. To this end, under the grant of Extended Collaborative Support Services [38], we developed a “Science Gateway” that allows the genomics community to access dREG through XSEDE [34] computational resources. The dREG Gateway website (https://dreg.dnasequence.org/) provides a user-friendly portal for biologists. Typically, users only need to upload PRO-seq data as two files representing raw counts mapped to the plus and minus strand of DNA in the standard bigWig format. A typical run takes 2–6 hours depending on the amount of data in the experiment. By signing up for an account, users may keep track of the status of their submitted jobs. Once dREG completes successfully, users may download the output files containing the dREG peak calls and dREG signal. The dREG Gateway also provides the option to visualize the input and output files as a private track hub on the WashU Epigenome Browser [40].

In this paper, we demonstrate the general framework and its detailed workflow of using XSEDE resources to build a gateway for sequencing data processing, and integrate other web applications with the PGA [29]. The paper will provide an introduction to the requirement for analyzing sequencing data using HPC resources, the architecture of dREG gateway, the detailed process of the dREG gateway development, and a discussion about the generalization of PGA to processing other sequencing data.

2 ARCHITECTURE OF DREG GATEWAY

2.1 Architecture

As a science gateway, the dREG gateway is hosted on the Jetstream server and dependent on the GPU resource of XSEDE. Technically, the dREG gateway is built on the PGA front end and uses Apache Airavata middleware to transfer data and submit compute jobs into XSEDE GPU nodes as illustrated in Figure 1.

Figure 1 demonstrates the architecture of the dREG gateway. In brief, it can be divided into two parts. In the first part, the secured web service receives the request and sequence data from the web browser via Apache HTTPD. The Apache server completes the process of user authentication and other related processes with the aid of PGA hosted on JetStream. In the second part, the computing tasks are submitted to the HPC computing resource through the Airavata middleware hosted on the Indiana University's Intelligent Infrastructure (IUII). The HPC employs a job scheduler to call the R programs that complete the computation on GPU nodes provided by XSEDE, such as Comet GPUs at the San Diego Supercomputer Center. Once the computation is completed, Airavata copies the results from the HPC storage into the user's web storage.

Apache Airavata [24, 27, 30] is open-sourced and open-community middleware for providing general purpose services needed by science gateways. It controls metadata and computational tasks and enables the execution of simple applications and workflows on clusters, computational clouds, and HPCs. It is mainly composed of 6 components, 1) registry, which manages metadata, 2) orchestrator, which organizes the execution of tasks, 3) workflow interpreter, 4) an application manager, which interacts with remote computational devices, 5)a messaging system, which sends and receives messages, such as external emails, and 6) the credential store, which controls security credentials needed to access remote resources. In our dREG gateway, these components can be managed, configured and invoked by PGA through the Apache Airavata API.

PGA is an extendable web framework to implement the basic functions of Airavata using Airavata Thrift's PHP client library bindings. PGA can be regarded as an Airavata client, which employs webpage-based user interface to implement basic functions, including user management, system configuration, and data analysis.

2.2 Interaction With PGA

The dREG gateway offers dREG peaking calling services on user-defined data to the genomics community. Similar to most of gateway platforms, users create a dREG task based on a dREG application template, and upload the input data onto the web server. These steps only involve interactions with PGA until the dREG task has been submitted for computation. Upon the dREG task being submitted, the orchestrator component in Apache Airavata initiates the computation by invoking the application manager using Apache Airavata's messaging component to monitor the status of the submitted task, shown in Figure 2.

Airavata uses SSH to access the HPC compute nodes. Public key authentication is used to automate the SSH connection between Airavata and the computing nodes; keys are securely

managed using the credential store component. Airavata generates task scripts that are appropriate for the task schedulers on the target resource, such as PBS and SLURM, to execute bash script in the compute nodes. Schedulers need to specify the account information, task queue, required compute resources (e.g., the number of CPUs, the size of memory and wall time), and the email address used to report the status of job to Airavata's messaging system. The email address is crucial for Airavata to monitor the status of the submitted job. Upon finishing the computation of the submitted job, Airavata uses the scp command to copy the result to web storage, and notify the user by email. The results and input data can be stored in the web storage for 3 months due to the large size of input files even the gateway with its large storage is seprately deployed.

3 IMPLEMENTATION

The dREG gateway provides a powerful online computational tool for the biologist or bioinformatician, which has the following features:

1) The dREG gateway provides a data visualization interface using the WashU EpiGenome Browser to show (a) input bigWig data; (b) signal predicted by dREG; and (c) the genomic coordinates of peaks of transcription initiation identified by the peak calling function.

2) The dREG gateway doesn't need users to install any software. It eliminates the times necessary to become familiar with the dREG package and R development environment, installation and configuration of this complex software package.

3) The dREG gateway provides a GPU computing power to biologists interested in using dREG. dREG uses a SVR model trained by large scale data, the training and prediction are computationally intensive, CPUs take a long time, and are difficult to get the results. The computational speed is increased 100 times with the aid of a GPU SVM package [36]. As a result, the GPU resources on XSEDE enhance data processing capabilities of genomic GRO-seq, PRO-seq and ChRO-seq experiments. With the aid of 16 CPU cores, NVIDIA Tesla K80 and P100 on Comet are able to speed up computation by 48-fold and 80-fold compared with using a multi-core computing environment for a small GRO-seq dataset (6M informative position). Large datasets can achieve much more substantial performance improvements on GPUs.

The implementation of the dREG Gateway began with the installation of the PGA framework. The PGA framework provides the gateway administrator with essential functions of the gateway, such as management of compute nodes, the ability to create and deploy applications, and management of user accounts and computational experiments. It also allows the general user to submit new jobs and review results. Considering the requirements of dREG users, we customized the PGA by 1) creating a dREG web theme after the installation of the latest PGA framework;2) modifying the user interface to make it more user-friendly to a biologist, for instance by automating selection of HPC resources; 3) adding tools for uploading large data sets, which avoids the slow speed and vulnerability in conventional HTTP post-based approaches; 4) providing the function that allows the

downloading through HTTP without the user session, which facilitates collaboration between other web services for data visualization.

3.1 Our Experience Implementing the dREG Gateway

Interdisciplinary collaborations between computer science and computational biologists turned dREG from a UNIX command-line bioinformatics tool to a science gateway with a functional web interface. XSEDE provided the essential infrastructure of the gateway, including 1) a virtual host on JetStream [19] for the web server; 2) hosted Airavata services operated by the IU team [2];3) CPU or GPU nodes on the HPC server. Based on this infrastructure, the following additional steps were required for building the complete gateway.

1) Installation and configuration of PGA.—PGA is a web framework developed on the PHP Laravel framework, and runs in the Apache HTTPD server on a virtual host. The parameters of the hosted Apache Airavata domain need to be configured in PGA after its installation on the virtual host. This configuration indicates the Airavata server that the PGA is dependent on, and specifies the PGA identification. Users may also install their own individual Airavata services and setup the corresponding parameters for PGA. Airavata communicates with the compute nodes in the HPC through SSH. Therefore, the user needs to create a gateway community account, and allow the access of compute nodes from the Airavata server via SSH through this account. To do this, the public key from PGA credential store needs to be added to the authorized key file on the compute nodes for the gateway account. Note that this step may be accomplished with the aid of administrator privileges or management tools in some HPC systems.

2) Theme design.—This step provides dREG with a customized homepage and user interface. We designed the logo, homepage, instruction, document, FAQ, etc. PGA isolates the independent contents into a web theme, and provides gateway designers with the option to either use the default UI interface, or to customize the theme.

3) Install applications on the computing node.—Our dREG model is encapsulated within the dREG package in R. After all dependencies were compiled and installed on the compute node, we used bash scripts to create an application interface, which will be registered in the next step.

4) Registering the application module and interface in PGA.—Gateway developers need to register the application module and interface with its required and optional input and output parameters, including parameter name, type, and other properties, on the PGA administrator dashboard. Parameter types can include typical primitives such as strings and integers, as well as more custom types. Inputs corresponding to uploaded files are specified as Uniform Resource Identifier (URI), and PGA will save the uploaded file on the web storage.

5) Deploying the application to the compute node.—Gateway developers need to specify resource information of the compute resource, including the resource address,

application executable path (generally pointing to bash scripts), computing queue, etc. when deploying applications to the compute resource. After binding the application to the target computing resources, the application is available on the web page of new experiment for general users.

Generally gateway development may involve collaborations among web designers, programmers, and administrators. Web designers may customize home page, programmers wrapper applications or workflows using bash scripts, and the administrators may setup the gateway system, such as registering and deploying the application. Beyond the XSEDE application and PGA installation, it may cost one or two weeks to build a simple gateway, including web pages on the gateway, bash scripts on computing nodes, and configuration of the gateway. However the time investment required may be significantly different depending on the complexity of the workflow and web page design.

3.2 Uploading Big Data

In the current version for PGA, files are uploaded through the default HTML file operation. Although HTML itself does not have limits on the size of the file transfer, the server end has restrictions on it to speed up the response to multiple users. The drawback is that larger files are more prone to upload failure if the network connection is not good, and from an end user's perspective the connection with the server appears to hang while large files are transferred. In the case of dREG, each file can be as large as hundreds of megabytes (MB). Therefore, we assembled an upload scheme based on Tus [3], a protocol for resumable file upload via HTTP/1.1 and HTTP/2. Tus deploys on server and client side, both implemented by many programming languages or frameworks, e.g. JavaScript, Java, Python, PHP. Currently PGA is implemented in the PHP Laravel 4.2 framework, which is not yet supported by Tus community. To address this issue, we implemented this package based on the server code on similar platforms (https://github.com/Danko-Lab/Laravel-tus-server), and added it to the PGA module. We also embedded the JavaScript code as a client for the Tus protocol in the new experiment webpage. In addition we increased the size limit on file uploading in the PGA configuration.

The principle by which Tus uploads files is to break files into smaller pieces, and then consecutively submit each file to the server using JavaScript. The user may monitor the file upload by a progress bar in the web page. In addition to the Tus protocol, the Java Struts2 package, developed by COSMIC2 [13], integrates the Globus's method of uploading files and provides another convenient way of uploading files from Globus network to web servers. Globus is more convenient to transfer big data between data servers than Tus which needs to download data firstly from data server and then upload to the dREG gateway. If PGA can integrate these two methods in the next release, gateway developers and users will have more flexible solutions than the current implementation.

3.3 Displaying Results in a Genome Browser

A genome browser is an online tool for visualizing NGS data along with gene annotations and other information about the target genome. There are two major providers, WashU [40] and UCSC genome browser [18]. All raw data and output files of dREG runs can be

visualized using either genome browser. With the aid of the annotation tools provided by the genome browser, biologists can visualize the dREG analysis results. In order to provide this feature to users in a seamless manner, the gateway needs to provide partial content of sequence data via HTTP or FTP protocol. Although the latest PGA offers HTTP-based data downloading, it has several limitations. First, it requires user authentication, i.e. to verify the user session. Secondly, data downloading has to be performed from the beginning as a whole and cannot be accessed randomly.

To overcome these barriers, we have implemented new ways of accessing data without a valid user session for PGA, and improve the HTTP protocol process for data downloading to implement partial data downloading. This violates the data protection mechanism of PGA. To compensate, we encrypt file names when requesting data download, and grant the peer sites in PGA, i.e. the genome browser website, which allows data downloading without the session file.

The WashU genome browser specifies data formatting using JSON format, and all necessary parameters that describes sequence data. The dREG gateway adds a link to the Genome Browser in the page of the experiment result. After this link is triggered, the user side pops up a new browser window to show the Genome Browser, and then the Genome Browser server launches the track script request for the dREG gateway. The track script is parsed in the Genome Browser server. Then, according to the file list in the track script, the Genome Browser Server then requests partial data for the track visualization. Finally the partial genome view is combined with other annotations is delivered to the client browser, as shown in Figure 3.

4 CONCLUSION

Cloud-based next generation sequencing (NGS) tools are currently at an early stage [9]. There are several examples in industry and academia. Amazon EC2 [1] has deployed comprehensive bioinformatics tools to provide a Platform-as-a-Service (PaaS) cloud environment. The Cancer Genomics Cloud (CGC) and National Cancer Institute jointly provided the web-based workflow design to analyze large-scale sequence data in CGC [31]. XSEDE provides HPC computing resources and gateway solutions to help the research community build Software-as-a-Service (SaaS) gateways on a web platform, including Galaxy [4], CIPRES [25], UltraScan [19], and COSMIC2 [13]. The application and development of NGS increasingly pushes computational requirements, and as a result Cloud-based data analytics platforms will be the preferred option for biologists without extensive IT knowledge.

In general, encapsulating bioinformatics tools and data analytics to build a simple Web service requires extensive expertise. Our dREG gateway takes advantage of XSEDE's services. Firstly, XSEDE provides hardware resources, including the virtual host for web server and computing nodes on HPC. Also, XSEDE offers Extended Collaborative Support Services for science gateways, which connects new gateway providers with experts in the field and with science gateway framework developers,, which greatly reduces the workload of installation and the difficulty of building a gateway. The PGA framework can help

developers make a prototype gateway rapidly. For bioinformatics applications with simple workflows, or for more complex workflows which can be decomposed into a compute job using bash scripts, PGA has the advantage of low development costs and high efficiency. Once such a web service is available, computational resources are concentrated and more easily maintained and efficiently utilized. Thus, our dREG gateway provides a general solution to develop a cloud platform for analyzing and processing the NGS data.

The cloud-based platforms for data analysis are becoming increasingly popular among biologists due to the variety and complexity of emerging NGS computational applications. In particular, as various machine learning and artificial intelligence algorithms [22, 39], which usually require the processing of a large number of training/testing examples, are developed and applied to the field of biology, the high computing power of these cloud-based platforms become advantageous. Additionally, a cloud-based system alleviates the burden of installation, setup and maintenance of bioinformatics software on biologists, letting them focus on the interpretation of the data. Following this trend, we predict that Web-based gateway providing cloud-based SaaS services will become mainstream.

ACKNOWLEDGMENTS

We thank XSEDE allocations TG-BIO160048 and TG-MCB160061 for providing computational resources required in this gateway, including the Web server, Apache HTTPD service, Airavata service, Web storage and GPU nodes. Work in this publication was supported by the NHGRI (National Human Genome Research Institute) grant R01-HG009309; Charles G. Danko is the principal investigator. Apache Airavata development is supported by NSF award # 1547611. The content is solely the responsibility of the authors and does not necessarily represent the official views of the US National Institutes of Health.

REFERENCES

[1]. 2018. Genome Cloud Computing - Amazon Web Service (AWS). (2018). https://aws.amazon.com/en/health/genomics/

[2]. 2018. SciGaP gateway. (2018). https://scigap.org/

[3]. 2018. Tus - resumable file uploads. (2018). https://tus.io/

[4]. Afgan Enis, Baker Dannon, Van den Beek Marius, Blankenberg Daniel, Bouvier Dave, Čech Martin, Chilton John, Clements Dave, Coraor Nate, Eberhard Carl, et al. 2016. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. Nucleic acids research 44, W1 (2016), W3–W10. [PubMed: 27137889]

[5]. Alipanahi Babak, Delong Andrew, Weirauch Matthew T, and Frey Brendan J. 2015. Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. Nature biotechnology 33, 8 (2015), 831.

[6]. Asgari Ehsaneddin and Mofrad Mohammad RK. 2015. Continuous distributed representation of biological sequences for deep proteomics and genomics. PloS one 10, 11 (2015), e0141287. [PubMed: 26555596]

[7]. Azofeifa Joseph G and Dowell Robin D. 2017. A generative model for the behavior of RNA polymerase. Bioinformatics 33, 2 (2017), 227–234. [PubMed: 27663494]

[8]. Bray Nicolas L, Pimentel Harold, Melsted Páll, and Pachter Lior. 2016. Near-optimal probabilistic RNA-seq quantification. Nature biotechnology 34, 5 (2016), 525.

[9]. Celesti Antonio, Celesti Fabrizio, Fazio Maria, Bramanti Placido, and Villari Massimo. 2017. Are next-generation sequencing tools ready for the cloud? Trends in biotechnology 35, 6 (2017), 486–489. [PubMed: 28363406]

[10]. Chae Heejoon, Rhee Sungmin, Nephew Kenneth P, and Kim Sun. 2014. BioVLABMMIA-NGS: microRNA–mRNA integrated analysis using high-throughput sequencing data. Bioinformatics 31, 2 (2014), 265–267. [PubMed: 25270639]

[11]. Chen Yifei, Li Yi, Narayan Rajiv, Subramanian Aravind, and Xie Xiaohui. 2016. Gene expression inference with deep learning. Bioinformatics 32, 12 (2016), 1832–1839. [PubMed: 26873929]

[12]. Chu Tinyi, Rice Edward J, Booth Gregory T, Salamanca Hans H, Wang Zhong, Core Leighton J, Longo Sharon L, Corona Robert J, Chin Lawrence S, Lis John T, et al. 2017 Chromatin run-on reveals nascent RNAs that differentiate normal and malignant brain tissue. bioRxiv (2017), 185991.

[13]. Cianfrocco MA, Wong-Barnum M, Youn C, Wagner R, and Leschziner A. 2017 COSMIC2: A Science Gateway for Cryo-Electron Microscopy Structure Determination In Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact. ACM, 22.

[14]. Core Leighton J, Waterfall Joshua J, and Lis John T. 2008. Nascent RNA sequencing reveals widespread pausing and divergent initiation at human promoters. Science 322, 5909 (2008), 1845–1848. [PubMed: 19056941]

[15]. Danko Charles G, Hyland Stephanie L, Core Leighton J, Martins Andre L, Waters Colin T, Lee Hyung Won, Cheung Vivian G, Kraus W Lee, Lis John T, and Siepel Adam. 2015. Identification of active transcriptional regulatory elements from GRO-seq data. Nature methods 12, 5 (2015), 433. [PubMed: 25799441]

[16]. Desai Narayan, Antonopoulos Dion, Gilbert Jack A, Glass Elizabeth M, and Meyer Folker. 2012. From genomics to metagenomics. Current opinion in biotechnology 23, 1 (2012), 72–76. [PubMed: 22227326]

[17]. Kartashov Andrey V and Barski Artem. 2015. BioWardrobe: an integrated platform for analysis of epigenomics and transcriptomics data. Genome biology 16, 1 (2015), 158. [PubMed: 26248465]

[18]. Kent W James, Sugnet Charles W, Furey Terrence S, Roskin Krishna M, Pringle Tom H, Zahler Alan M, and Haussler David. 2002. The human genome browser at UCSC. Genome research 12, 6 (2002), 996–1006. [PubMed: 12045153]

[19]. Knepper Richard, Coulter Eric, Pierce Marlon, Marru Suresh, and Pamidighantam Sudhakar. 2017 Using the Jetstream Research Cloud to provide Science Gateway resources In Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. IEEE Press, 753–757.

[20]. Kudtarkar Parul, DeLuca Todd F, Fusaro Vincent A, Tonellato Peter J, and Wall Dennis P. 2010. Cost-effective cloud computing: a case study using the comparative genomics tool, roundup. Evolutionary Bioinformatics 6 (2010), EBO–S6259.

[21]. Kwak Hojoong, Fuda Nicholas J, Core Leighton J, and Lis John T. 2013. Precise maps of RNA polymerase reveal how promoters direct initiation and pausing. Science 339, 6122 (2013), 950–953. [PubMed: 23430654]

[22]. Libbrecht Maxwell W and Noble William Stafford. 2015. Machine learning applications in genetics and genomics. Nature Reviews Genetics 16, 6 (2015), 321.

[23]. Lohse Marc, Bolger Anthony M, Nagel Axel, Fernie Alisdair R, Lunn John E, Stitt Mark, and Usadel Björn. 2012. RobiNA: A user-friendly, integrated software solution for RNA-Seq-based transcriptomics. Nucleic acids research 40, W1 (2012), W622–W627. [PubMed: 22684630]

[24]. Marru Suresh, Gunathilake Lahiru, Herath Chathura, Tangchaisin Patanachai, Pierce Marlon, Mattmann Chris, Singh Raminder, Gunarathne Thilina, Chinthaka Eran, Gardler Ross, et al. 2011 Apache airavata: a framework for distributed applications and computational workflows In Proceedings of the 2011 ACM workshop on Gateway computing environments. ACM, 21–28.

[25]. Miller Mark A, Pfeiffer Wayne, and Schwartz Terri. 2010 Creating the CIPRES Science Gateway for inference of large phylogenetic trees In Gateway Computing Environments Workshop (GCE), 2010. IEEE, 1–8.

[26]. Niknafs Yashar, Molen Nicholas, Pandian Balaji, Iyer Matthew, and Chin-naiyan Arul. 2017. Bridging the gap between NGS data and its usability: cancer gene discovery through massive-scale transcriptomic analyses and development of a powerful web-tool for dissemination of these findings. (2017).

[27]. Pamidighantam Sudhakar, Nakandala Supun, Abeysinghe Eroma, Wimalasena Chathuri, Rathnayaka Yodage Shameera, Marru Suresh, and Pierce Marlon. 2016. Community science exemplars in seagrid science gateway: Apache airavata based implementation of advanced infrastructure. Procedia Computer Science 80 (2016), 1927–1939.

[28]. Patro Rob, Duggal Geet, Love Michael I, Irizarry Rafael A, and Kingsford Carl. 2017. Salmon provides fast and bias-aware quantification of transcript expression. Nature methods 14, 4 (2017), 417. [PubMed: 28263959]

[29]. Pierce Marlon, Marru Suresh, Demeler Borries, Singh Raminderjeet, and Gorbet Gary. 2014 The apache airavata application programming interface: overview and evaluation with the UltraScan science gateway In Gateway Computing Environments Workshop (GCE), 2014 9th. IEEE, 25–29.

[30]. Pierce Marlon E, Marru Suresh, Gunathilake Lahiru, Wijeratne Don Kushan, Singh Raminder, Wimalasena Chathuri, Ratnayaka Shameera, and Pamidighantam Sudhakar. 2015 Apache Airavata: design and directions of a science gateway framework. Concurrency and Computation: Practice and Experience 27, 16 (2015), 4282–4291.

[31]. Reynolds Sheila M, Miller Michael, Lee Phyliss, Leinonen Kalle, Paquette Suzanne M, Rodebaugh Zack, Hahn Abigail, Gibbs David L, Slagel Joseph, Longabaugh William J, et al. 2017. The ISB Cancer Genomics Cloud: a flexible cloud-based platform for cancer genomics research. Cancer research 77, 21 (2017), e7–e10. [PubMed: 29092928]

[32]. Sarda Shrutii and Hannenhalli Sridhar. 2014. Next-generation sequencing and epigenomics research: a hammer in search of nails. Genomics & informatics 12, 1 (2014), 2–11. [PubMed: 24748856]

[33]. Singh Ritambhara, Lanchantin Jack, Robins Gabriel, and Qi Yanjun. 2016. Deepchrome: deep-learning for predicting gene expression from histone modifications. Bioinformatics 32, 17 (2016), i639–i648. [PubMed: 27587684]

[34]. Towns John, Cockerill Timothy, Dahan Maytal, Foster Ian, Gaither Kelly, Grimshaw Andrew, Hazlewood Victor, Lathrop Scott, Lifka Dave, Peterson Gregory D, et al. 2014 XSEDE: accelerating scientific discovery. Computing in Science & Engineering 16, 5 (2014), 62–74.

[35]. Wang Sheng, Peng Jian, Ma Jianzhu, and Xu Jinbo. 2016. Protein secondary structure prediction using deep convolutional neural fields. Scientific reports 6 (2016), 18962.

[36]. Wang Zhong, Chu Tinyi, Choate Lauren A, and Danko Charles G. 2017. Rgtsvm: Support Vector Machines on a GPU in R. arXiv preprint arXiv:1706.05544 (2017).

[37]. Wang Zhong, Chu Tinyi, Choate Lauren A, and Danko Charles G. 2018. Identification of regulatory elements from nascent transcription using dREG. bioRxiv (2018), 321539.

[38]. Wilkins-Diehr Nancy, Sanielevici Sergiu, Alameda Jay, Cazes John, Crosby Lonnie, Pierce Marlon, and Roskies Ralph. 2015 An Overview of the XSEDE Extended Collaborative Support Program In International Conference on Supercomputing. Springer, 3–13.

[39]. Yue Tianwei and Wang Haohan. 2018. Deep Learning for Genomics: A Concise Overview. arXiv preprint arXiv:1802.00810 (2018).

[40]. Zhou Xin, Li Daofeng, Zhang Bo, Lowdon Rebecca F, Rockweiler Nicole B, Sears Renee L, Madden Pamela AF, Smirnov Ivan, Costello Joseph F, and Wang Ting. 2015. Epigenomic annotation of genetic variants using the Roadmap Epigenome Browser. Nature biotechnology 33, 4 (2015), 345.

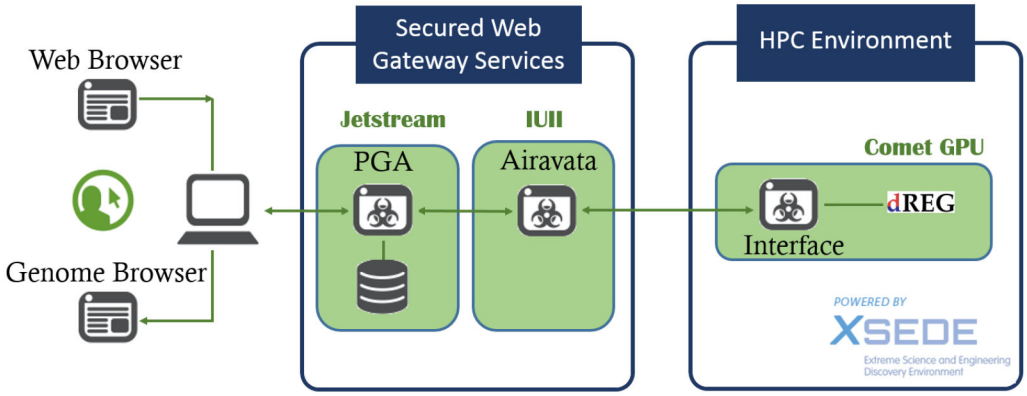

Figure 1:
Architecture of dREG gateways.

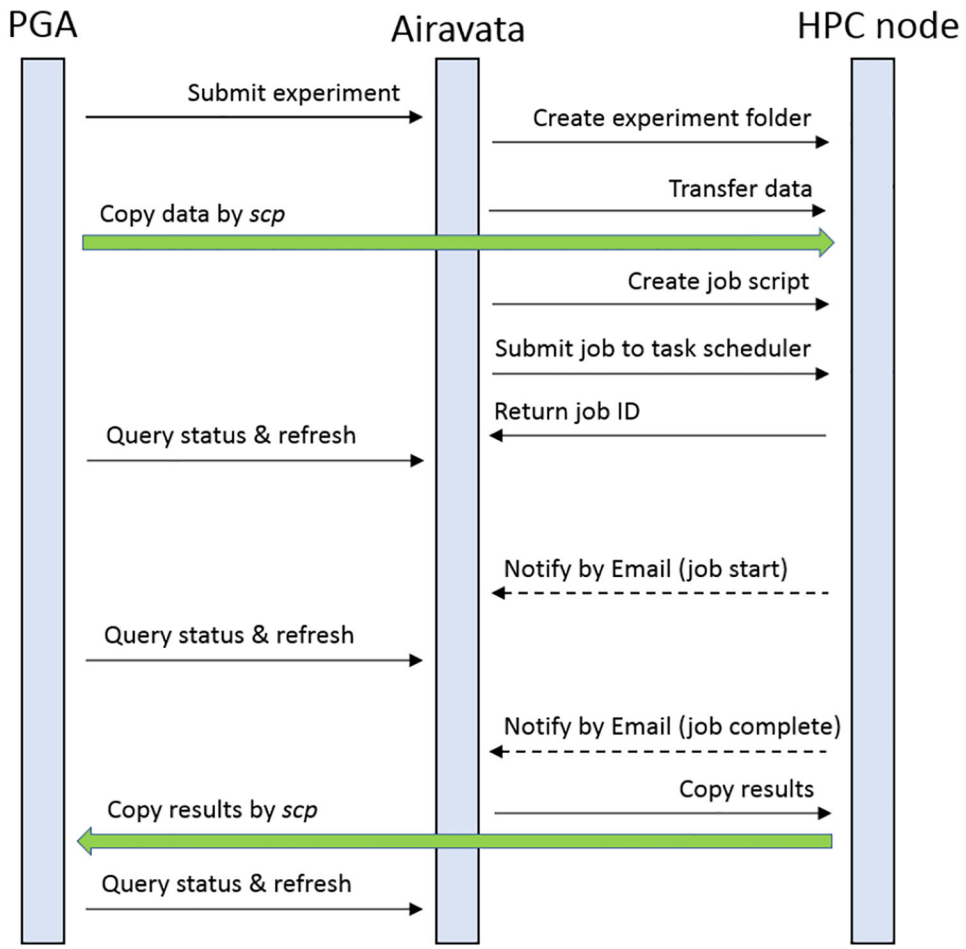

Figure 2:
PGA interaction with Airavata and compute nodes.

Figure 3:
dREG results shown in WashU genome browser.