# Tibanna: software for scalable execution of portable pipelines on the cloud

Soohyun Lee[1], Jeremy Johnson[1], Carl Vitzthum, Koray Kırlı, Burak H. Alver[*], Peter J. Park[*]

[1]Equally contributed

Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA.

*To whom correspondence should be addressed.

**Summary:** We introduce Tibanna, an open-source software tool for automated execution of genomic pipelines on Amazon Web Services (AWS). Tibanna accepts portable pipeline standards including Common Workflow Language (CWL) and Docker, and it can be used through a simple command line interface (CLI) or Python Application Programming Interface (API). It adopts a strategy of isolation and optimization of individual executions, without pre-configured clusters. Pipelines are automatically downloaded, executed and monitored on a single local command. The versatility of Tibanna makes it ideal both for Data Commons, where data to be processed come in sparsely or in bursts, and for stand-alone large-scale parallel batch processing. Tibanna has been used to reproducibly process terabytes of genomic data for the 4D Nucleome (4DN) Network.
**Availability:** Source code is available on GitHub at https://github.com/4dn-dcic/tibanna.
**Contact:** peter_park@hms.harvard.edu

## 1 Introduction

Efficient execution of data processing and analysis pipelines is essential in many areas of research that involve large datasets. However, integration of a working pipeline with local computing infrastructures is often a painstaking process. The need for such integration also leads to redundant creation of seemingly identical pipelines, reducing reproducibility and efficiency.

Cloud platforms present a number of advantages for large-scale data analysis, including scalability and efficient data sharing. However, the frequent lack of separation between pipeline content and platforms impedes transition to the cloud. Having to handle detailed aspects of the cloud platform renders the advantages of cloud computing impractical for many pipeline developers.

For effective separation of pipelines from platforms, pipelines should be made portable and standardized, and all platform-specific tasks must be delegated to a separate pipeline management tool. To enable this framework, there have been efforts to create a standard for portable pipelines. Common Workflow Language (CWL) (https://www.commonwl.org/) and Workflow Description Language (WDL) (https://software.broadinstitute.org/wdl/) describe the basic structure of a pipeline (e.g. steps, inputs and outputs), whereas Docker and Singularity (Kurtzner et al., 2017) enable generation of portable images of executable components and dependencies.

At the Data Integration and Coordination Center (DCIC) for the NIH-sponsored 4D Nucleome (4DN) (Dekker at al. 2017) consortium, our aim was to quickly process a large number of datasets from multiple experimental types (Hi-C, ChIA-PET, RNA-seq, ChIP-seq, etc) submitted by member laboratories. We therefore needed a workflow management system that supports standardized and portable pipelines on a public cloud, specifically Amazon Web Services (AWS), the most widely used commercial cloud platform. In particular, having an open-source implementation was critical not only to the consortium but also to the scientific community so that they can easily run the same standardized 4DN pipelines on their own data.

## 2 Results

We have developed and used Tibanna as an open-source pipeline manager and automated cloud resource allocator for running CWL- and Docker-based pipelines on AWS. Tibanna adopts a versatile, individual optimization approach that overcomes the limitations of an existing cluster-based approach. It also integrates easily with other systems.

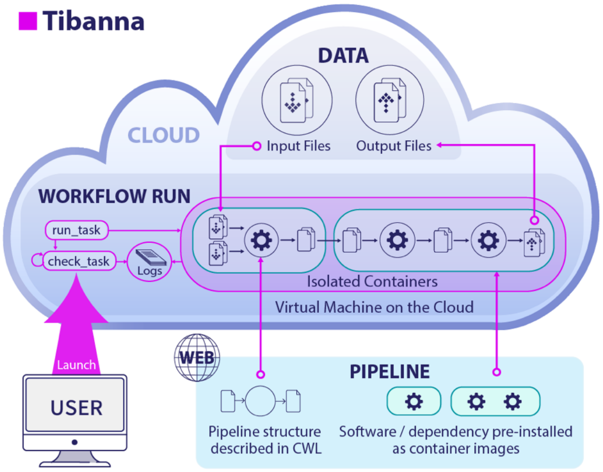
Tibanna
DATA
CLOUD
Input Files
Output Files
WORKFLOW RUN
run_task
check_task
Logs
Isolated Containers
Virtual Machine on the Cloud
Launch
USER
WEB
PIPELINE
Pipeline structure described in CWL
Software / dependency pre-installed as container images

**Fig 1. Tibanna**. For each pipeline execution, Tibanna creates a workflow run by launching an individually scheduled and optimized virtual machine (VM), which takes data and the pipeline, runs the pipeline and destroys itself when finished.

### 2.1 Overview

Tibanna (named after the gas fuel processed in the Cloud City in Star Wars) creates an instance of a pre-configured virtual machine (VM) for each job (Figure 1). The instance receives user input at launch, and then autonomously fetches data and the pipeline, runs the pipeline, sends logs and output files to permanent storage on the cloud and finally terminates itself. Instead of depending on a master queue, each job gets its own scheduler, which is deployed and managed by local code. The scheduler is comprised of serverless functions called AWS Lambdas that are coordinated using a state machine called AWS Step Function. The Lambdas initiate a run and check logs. We use additional Lambdas for metadata handling specifically for integration with the 4DN data portal.

### 2.2 Comparison to other tools

Tibanna offers critical advantages compared to existing workflow management tools, specifically for our need to both automatically process data on the 4DN Data Portal and to use it as a stand-alone tool for pipeline development. We needed an automated and integrable tool that supports portable pipeline standards including CWL/Docker, which allocates optimal resources to individual executions and provides detailed real-time monitoring.

Some tools (e.g. Cwltool, Rabix (Kaushik 2016)) execute CWL locally or in a pre-allocated virtual machine on the cloud but do not perform cloud resource allocation like Tibanna. CWL-Airflow (Kotliar et al. 2017) provides only a library for AWS integration.

Other tools perform cloud resource allocation but do not support CWL. AWS Batch (https://aws.amazon.com/batch/), an AWS service to run batch jobs on AWS, requires a substantial amount of development work to be used with CWL. Nextflow (Tommaso et al., 2017) uses AWS Batch and requires a CWL pipeline to be converted to the custom Nextflow format. However, its companion converter is not functional at this point. Nextflow is a Docker-based pan-platform workflow management system, but its custom format is tied to its own system and it is difficult to develop other applications around. Funnel (https://ohsu-comp-bio.github.io/funnel/), also based on AWS Batch, requires users to do extra work such as building a custom Amazon Machine Image, to be used with CWL. Cromwell (https://software.broadinstitute.org/wdl/) runs WDL on the Google Cloud Platform, but not CWL or on AWS yet.

StarCluster (http://star.mit.edu/cluster/docs/latest/index.html), an older tool without CWL/Docker support, and Toil (Vivian et al., 2017) work by creating a resizable cluster of VM instances of the same CPU and memory, and an additional master VM. They require extra work for configuring and managing clusters. The cluster approach does not make the best use of the elasticity of the cloud and is not efficient for automated processing of various types and sizes of data that may come in sparsely or in bursts. Tibanna uses a more versatile approach of launching an individually optimized VM instance per execution, and terminating it when the job is completed. Seven Bridges and DNA Nexus also adopt this approach. They also offer full automation. However, as a commercial portal service, they charge extra fees and they do not easily integrate with other systems.

### 2.3 How to use

To use Tibanna, a pipeline must be packaged as a Docker image, described in CWL and be accessible on the Web. Input data must be on the AWS cloud and each job should be described in standard JSON format, which specifies the pipeline, input files and parameters. Submission and monitoring of a job can be performed locally from the command line or using Tibanna’s API. Logs can be retrieved either using command-line tools or through the AWS Web Console. A user with a private key file can connect via ssh into a running instance for more detailed monitoring.

## 3 Conclusion

Tibanna is a lightweight pipeline management system for the AWS cloud. Tibanna has been used to process 4DN data and to support other AWS-based projects. The 4DN pipelines are available on GitHub and Docker Hub; Tibanna connects to these repositories to get the pipelines. We have successfully executed over 1,000 jobs simultaneously on several occasions for projects involving, e.g., whole-genome sequencing data. We believe Tibanna will be useful for many future AWS projects.

## Acknowledgements

We thank the members of the 4DN DCIC, the Park lab and the Avillach lab at the Department of Biomedical Informatics at Harvard Medical School for valuable feedback and the E. Alice Lee lab at Boston Children's Hospital for beta-testing Tibanna. We are especially thankful to Andy Schroeder, Peter Kerpedjiev, Martin Owens and Chuck McCallum for code review/contribution, Shannon Ehmsen for preparing the figure, and Alon Galor, Dhawal Jain, Su Wang and Geoff Nelson for comments on the manuscript.

## Funding

This work has been supported by the National Institutes of Health (5U01CA200059) to PJP.

*Conflict of Interest:* none declared.

## References

Dekker J et al. (2017), "The 4D nucleome project", *Nature*, 549: 219–226.

Gaurav K (2016), "Rabix: an open-source workflow executor supporting recomputability and interoperability of workflow descriptions" *Pac Symp Biocomput.* 22:154–165.

Kotliar M et al. (2017) "CWL-Airflow: a lightweight pipeline manager supporting Common Workflow Language", *bioRxiv* https://doi.org/10.1101/249243

Kurtzner GM et al. (2017), "Singularity: Scientific containers for mobility of compute", *PLoS ONE*, 12(5): e0177459

Tommaso PD et al. (2017), "Nextflow enables reproducible computational workflows", *Nat Biotechnol* 35:316–319

Vivian J et al. (2017), "Toil enables reproducible, open source, big biomedical data analyses", *Nat Biotechnol* 35:314–316