Bioinformatics Workflows With NoSQL Database in Cloud Computing

Evolutionary Bioinformatics
Volume 15: 1–11
© The Author(s) 2019
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/1176934319889974

Polyane Wercelens[1], Waldeyr da Silva[1,2], Fernanda Hondo[1], Klayton Castro[1], Maria Emília Walter[1], Aletéia Araújo[1], Sergio Lifschitz[3] and Maristela Holanda[1]

[1]Department of Computer Science, University of Brasília, Brasília, Brazil. [2]NEPBIO (Group of Biological Studies and Research on Cerrado), Federal Institute of Goiás (IFG), Formosa, Goiás, Brazil. [3]Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil.

ABSTRACT: Scientific workflows can be understood as arrangements of managed activities executed by different processing entities. It is a regular Bioinformatics approach applying workflows to solve problems in Molecular Biology, notably those related to sequence analyses. Due to the nature of the raw data and the in silico environment of Molecular Biology experiments, apart from the research subject, 2 practical and closely related problems have been studied: reproducibility and computational environment. When aiming to enhance the reproducibility of Bioinformatics experiments, various aspects should be considered. The reproducibility requirements comprise the data provenance, which enables the acquisition of knowledge about the trajectory of data over a defined workflow, the settings of the programs, and the entire computational environment. Cloud computing is a booming alternative that can provide this computational environment, hiding technical details, and delivering a more affordable, accessible, and configurable on-demand environment for researchers. Considering this specific scenario, we proposed a solution to improve the reproducibility of Bioinformatics workflows in a cloud computing environment using both Infrastructure as a Service (IaaS) and Not only SQL (NoSQL) database systems. To meet the goal, we have built 3 typical Bioinformatics workflows and ran them on 1 private and 2 public clouds, using different types of NoSQL database systems to persist the provenance data according to the Provenance Data Model (PROV-DM). We present here the results and a guide for the deployment of a cloud environment for Bioinformatics exploring the characteristics of various NoSQL database systems to persist provenance data.

KEYWORDS: Bioinformatics workflows, reproducibility, data provenance, cloud computing, NoSQL

RECEIVED: October 27, 2019. **ACCEPTED:** October 29, 2019.

TYPE: Original Research

FUNDING: The author(s) received no financial support for the research, authorship, and/or publication of this article.

DECLARATION OF CONFLICTING INTERESTS: The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

CORRESPONDING AUTHOR: Polyane Wercelens, University of Brasília, Brasília 70910-900, Brazil. Email: polyane.wercelens@gmail.com

Introduction

In silico scientific experiments are usually performed through workflows, which may be viewed as a chain of software executions.[1] The activities in a workflow may have particular characteristics and purposes, considering an order of execution, demanding ad hoc modeling and management.[2,3]

In Bioinformatics, workflows usually perform experiments on data from the so-called *omics* (genomics, transcriptomics, metabolomics, etc.),[4] whose sequencing technologies have become more affordable. It has made the amount of omics data become a Big Data matter,[5] for which the workflows often require a high-performance computing environment.[6] Maintaining such an environment requires resources, time, and skilled operators, which makes cloud computing an emerging alternative to Bioinformatics workflows. Along with guaranteeing the required reproducibility inherent in scientific experiments, constraints such as the data provenance and technical restrictions based on the user profile must be addressed.[7]

Given this scenario, this study presents a solution for cloud-based Bioinformatics workflows using data provenance to enable the reproducibility, including the computational environment itself. We discuss the effectiveness of the proposed solution using 3 typical Bioinformatics workflows and 3 distinct families of Not only SQL (NoSQL) database systems to store and retrieve the data provenance.

In section "Background," we introduce the concepts involved in this study, followed by the related works and their contributions. Then, section "Method" describes our method, followed by section "Results," where we show the results and their practical implications for improving the reproducibility of workflows in Bioinformatics.

Background

In this section, we introduce essential concepts of data provenance, cloud computing, and the Infrastructure as a Service (IaaS), and NoSQL database systems.

Data provenance

Reproducibility is a fundamental matter in the production of scientific knowledge. The phases of a scientific experiment are often defined in a protocol, or in the case of in silico experiments, in a workflow. Data provenance must provide a lineage or history of how the data were created, used, and modified preserving the data source and the process employed to transform it into a final product.[8,9] For this reason, data provenance is closely related to reproducibility.

Creative Commons Non Commercial CC BY-NC: This article is distributed under the terms of the Creative Commons Attribution-NonCommercial 4.0 License (http://www.creativecommons.org/licenses/by-nc/4.0/) which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).

Data provenance has been traditionally used in the workflow context,[10] and its utility resides in the possibility of easily creating, evaluating, or modifying the computational models or scientific experiments once the provenance expresses the accumulation of knowledge of what was done.[2] Studies in Scientific Workflow Management Systems (SWfMS) have collaborated on suitable solutions such as a taxonomy definition to classify those systems and the understanding of the workflow life cycle.[2]

Over time, some provenance models were proposed, such as the W7 Model,[11] Provenir Ontology,[12] Provenance Vocabulary,[13] Open Provenance Model (OPM),[14] and Provenance Data Model (PROV-DM).[15] PROV-DM is a generic model that has been proposed to capture the data source in such a way that different systems can import and export their specifications. It represents the provenance data through an acyclic and directed graph describing the involved agents (e.g. people), entities, and activities. In a data provenance graph, nodes can represent objects, such as files or programs, and edges represent dependencies between these objects.

Cloud computing

Cloud computing defines resources on-demand to its users from a service provider, abstracting the infrastructure details. Several paradigms of computing have converged to that. They express mainly as virtualization, service orientation, and distributed systems, allowing a conceptually agnostic way to operate concerning the hardware that supports its applications and data in an almost unlimited way. The provisioned infrastructure is both accessible over the internet and geographically hosted in distributed data centers.

High availability, fault tolerance mechanisms, and, notably, elasticity, are some features provided by cloud computing services.[16] Elasticity is the ability to provision resources quickly and, in some cases, automatically.[17] So, provisioning computational resources can be done to increase or flexibly decrease them, consistent to the user needs.

On one hand, service models describe an architectural standard for the operation of cloud-based solutions. These models are defined as Platform as a Service (PaaS), IaaS, and Software as a Service (SaaS).[18] On the other hand, the deployment models can be defined as public, private, community, or hybrid[17] and are determined by who afford to manage and operate the cloud facilities.

Although there are various technologies and hardware involved in these environments, resource management is unified. Many service providers come up with custom interfaces for handling their resources, such as Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), DigitalOcean, and others. These providers offer a toolset that integrates the view of the user resources and enables suitable operation of the hosted environment.

Several threats can occur in a cloud computing environment in the different service and deployment models.[19] So, cloud security mechanisms describe a set of policies, technologies, and control, aiming to protect the information and the hosted services.[20] A pivotal concept is confidentiality, which indicates that only authorized individuals or systems can access data and computational resources,[20] even in a multi-tenant infrastructure.

We have chosen to use IaaS, considering that in this service model, the user has more control over the cloud service stack than in PaaS and SaaS. The data are stored in block devices deployed in virtual machines (VMs), and not directly deal with the object storage system of the cloud provider, which usually is less performative than the other one.

Furthermore, when running a workflow, the computational resources can be expanded using the elasticity mechanisms of the chosen cloud provider. Thus, a cloud computing-based approach embraces seemly that the capturing and the storing provenance data enable the reproduction of the experiments and also include information concerning the computational environment.

NoSQL databases

The NoSQL databases have appeared with the increasing demand for databases that are capable of storing, processing, reading, and writing significant amounts of data with high performance[21] and offer a feasible alternative to the traditional and well-established Relational Database Management Systems (RDBMS).[22]

NoSQL database systems function with a flexible schema (or schemaless), able to handle both structured and unstructured data. NoSQL database systems distinguish in 4 leading families, and some of them are hybrid and implement more than 1.[21-24] These families are as follows:

- Key-value database systems: store data in a pair of parts (keys and values) where unique key indexes each value. Values are isolated and independent of each other, while the application logic treats relationships. Some examples of key-value stores are Voldemort[25] and Redis.[26]
- Column-oriented database systems: a predefined set of columns rules the structure of the values. It works as an orthogonal view of data regarding the tuple (line)-oriented relational database systems. HBase[27] and Cassandra[28] are examples of column-oriented database systems.
- Document-oriented database systems: store documents as collections of attributes and values and may contain multivalued attributes. They established that keys and values could apply to each document, identified by a key which is unique within a collection. Usually, the data artifacts are stored adopting formats such as JavaScript Object Notation (JSON) or eXtensible Markup Language (XML). Some document-oriented database systems are MongoDB[29] and CouchDB.[30]
- Graph-based database systems: they use a graph schema composed of nodes and edges to represent the data that can be stored both in the nodes and in the edges between

them. Some examples of Graph-based systems are Neo4J,[31] ArangoDB,[,32] and OrientDB.[33]

Concerning Bioinformatics, when analyzing the innovations that have emerged since the Next Generation Sequencing (NGS), NoSQL has served to deal with the problems and limitations that a Relational Database encounters in this research field.[34,35] Due to the relevant relationship between NoSQL database systems and cloud computing, which is the environment used for this study, we chose to evaluate the persistence of data provenance in 3 distinct NoSQL families.

Related Works

In this section, we present related works involving data provenance, its storage mode, and the computational environment of the experiments.

De Paula et al[36] proposed the management of data provenance for genome projects using PROV-DM. Their results demonstrated PROV-DM as a suitable model for storing properties for each execution of Bioinformatics workflows, and one which also provided graphical representation for the large volumes of data generated by genome projects using the Entity Collections.

Two research studies were performed using the NoSQL Column-Oriented Cassandra database system for data provenance management using the PROV-Wf model.[37,38] The researchers analyzed the performance by running a Bioinformatics workflow. Ferreira et al[37] compared relational and NoSQL DBMS approaches by migrating source data from a PostgreSQL RDBMS to Cassandra, a column-based NoSQL system. Aniceto et al[38] extended this research by including MongoDB, a Document-Oriented NoSQL, comparing both Cassandra and MongoDB regarding PostgreSQL performance.

Also, other studies have used the Document-Oriented concept. Li et al[8] suggested the *Provenance Lens*, a framework that provides data source management in cloud environments and compares its performance when using RDBMS and NoSQL Document-based and Graph-based systems. Sempéré et al[39] introduced *Gigwa*, a sharing resources web tool, that bases on MongoDB, providing a way to explore large amounts of genotyping data using filters of different combinations. Chacko et al[40] introduced *PERM*, a provenance management system that uses the idea of *Data Foreign Wrappers*, also with MongoDB.

Costa et al[41] presented the GeNNET, a platform that is capable of unifying scientific workflows running Docker containers employing the NoSQL Graph-Oriented Neo4J database system to integrate transcriptome analysis and select relevant genes.

Likewise, Almeida et al[42] conducted a study on the Graph-Oriented model, in which researchers presented AProvBio, an architecture that can perform the data provenance of scientific experiments in Bioinformatics automatically, using PROV-DM and Neo4J.

In another research, Costa et al[43] also captured the provenance data from a workflow using the PROV-Wf model. The workflow was executed both in desktop-based and cloud-based environments, whereas provenance was captured and stored in a relational database system.

Kanwal et al[7] captured provenance data from a workflow based on the Genome Analysis Tool Kit (GATK) using 3 different workflow definition approaches: *Galaxy* (graphical user interface-based integrative framework),[44] *Cpipe*[45] (Bioinformatics-specific pre-built pipelines), and Common Workflow Language (CWL), a declarative approach to workflow definition. Their conclusion reported assumptions and recommendations on reproducibility and data provenance.

There are some bioinformatics workflow management systems that deal with parallel tasks, including the development and management of graphics processing unit (GPU)-accelerated and distributed computing-based workflows.[46,47]

Incompatible data formats can hamper chaining outputs to each task in a workflow. Lenadora et al[48] addressed this problem through a suitable solution using utility and conversion functions, achieving an average reduction of size by 40% in data storage.

Curcin et al[49] provided an overview of scientific workflows and their capabilities of answering questions, reusability, and adaptability. They described the challenges and the steps involved when performing a study using primary care databases when related to provenance capture, component integration, and high-level informatics challenges.

Cloud computing for Bioinformatics is also a natural solution for throughput analysis.[50] It was used by Ko et al[51] while developing *Closha*, a hybrid automatic cloud-based workflow management system capable of running Hadoop-based and general-purpose applications, as well as performing a pipeline-based analysis service for massive biological data.

Regarding the provenance data in a cloud environment, 2 approaches stand out. The first solution is Galaxy,[44] which can be classified as an SaaS and represents data provenance in an OPM model.[14] Another solution is *CloudBioLinux*, which offers genome analysis resources through software images, and specific data repositories for cloud computing platforms.[52] A summary of the significant contributions of these related works is presented in Table 1.

Method

To evaluate our solution for data provenance management, we performed 3 different typical Bioinformatics workflows. The workflows executed on 1 private and 2 public computational cloud environments. The data provenance was captured and persisted in 3 distinct NoSQL database systems, with a mainly designed schema according to the family to which they belong. The provisioning of the VMs for the environment was done using Docker[53] as a container platform. The following

Table 1. Summary of the central contributions of the related works.

WORK	PROVENANCE MODEL	DATABASE TYPE	DATABASE INSTANCE	ENVIRONMENT
Curcin et al[49]	–	RDBMS	Oracle	–
Schatz et al[50]	–	–	–	Cloud
Krampis et al[52]	–	–	–	Cloud
De Paula et al[36]	PROV-DM	–	–	–
Costa et al[43]	PROV-Wf	RDBMS	*	Local and Cloud
Ferreira et al[37]	PROV-Wf	Column and RDBMS	Cassandra and PostgreSQL	Local
Aniceto et al[38]	–	Column	Cassandra	Local
Chacko et al[40]	–	Document	MongoDB	Local
Sadedin et al[45]		RDBMS	SQLite	Local
Li et al[8]	–	RDBMS, Document, and Graph	MySQL, MongoDB, and Neo4J	Local
Sempéré et al[39]	–	Document	MongoDB	Cloud
Afgan et al[44]	OPM	–	–	Cloud
Costa et al[41]	–	Graph	Neo4J	Cloud
Almeida et al[42]	PROV-DM	Graph	Neo4J	Local
Kanwal et al[7]	OPM	–	–	Local and Cloud
Ko et al[51]	–	–	–	Cloud

Abbreviations: OPM, Open Provenance Model; PROV-DM, Provenance Data Model; RDBMS, Relational Database Management Systems.
The symbol "–" means not applicable and "*" means not identified.

subsections describe the workflows, the data provenance schemas for each NoSQL database, and the cloud environment.

Bioinformatics workflows

The raw material for Bioinformatics workflows are "reads," which consist in strings of a limited alphabet (ACGT) representing DNA fragments obtained from NGS. Specific alignments of those reads give rise to an assembly producing contiguous sequences that represent samples of the original DNA.[54] *Greedy* algorithms, OLC (overlap-layout-consensus) methods, and k-mer-based *De Bruijn* graphs are the most used assembly strategies.[55]

A typical workflow for genome assembly is composed of the sequential phases filtering, assembly, and analysis, which includes annotation.[56] The filtering phase cleans the reads using specific quality parameters.[57] The assembly phase may be a reference-based or a de novo assembly. On one hand, the reference-based assembly of fragments uses a reference genome, aligning the reads of the target organism with its genome or a genome of a phylogenetically close organism. On the other hand, the so-called *de novo* assembly is performed without a reference genome.[55] The analysis phase is very diverse and depends on the biological response sought in the experiment, and the data are processed to validate the initial hypothesis of the experiment.[57]

Three real Bioinformatics workflows were used in this work as didactic examples. The first one consists of mapping reads of expressed DNA (cDNA) of human primary cardiac microvascular endothelial cells and hPSC-derived endocardial endothelial cells to the human chromosome 22 as the reference (Supplementary material Workflow for mapping reads to a reference). This workflow used the software Sickle,[58] SAMtools,[59] Hisat,[60] and HTSeq.[61]

The second workflow consists of a de novo genome assembly of a multidrug-resistant *Enterobacter kobei* isolate (Supplementary material Workflow for de novo bacterial assembly). The DNA of the *E. kobei* was sequenced using the HiSeq platform (Illumina) generating 100 bp paired-end reads.[62] This workflow used the Trimmomatic[63] for the filtering and trimming, Abyss[64] for the assembly, and Quast[65] for the statistical analysis of the results.

The third workflow consists of finding drug-resistant genes in an isolated pathogen species *Enterobacter cloacae subsp. cloacae NCTC 9394* (Supplementary material Workflow for identifying bacterial drug-resistant Genes). The reads filtering was performed using Trimmomatic,[63] and its assembly in contigs using SPAdes,[66] and again statistical analysis with Quast.[65] Then, the genes were predicted using with Glimmer[67] and annotated using the MEGARes antimicrobial resistance database[68] through Blast alignments.[69]

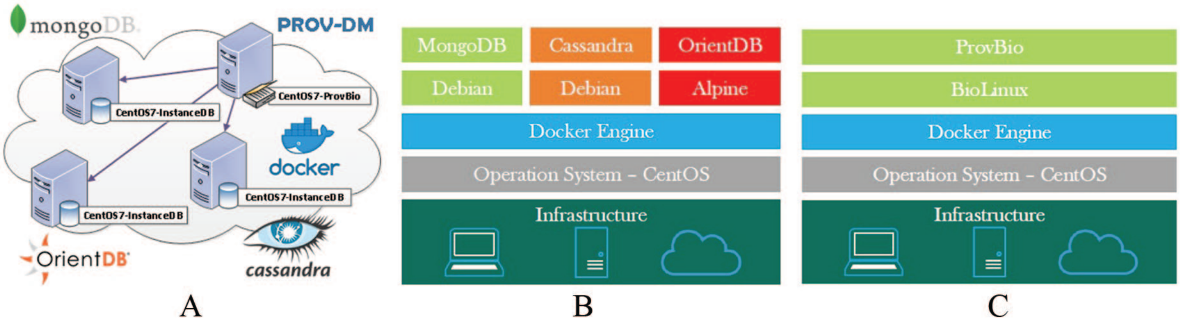


Figure 1. The cloud environment: (A) cloud environment instances, (B) database instances, and (C) workflow instance. PROV-DM indicates Provenance Data Model.

Table 2. Configuration of instances.

HOST NAME	ROLE	OS VERSION	DOCKER CONTAINER	CPU	RAM	HARD DISK
Instance-1	Cassandra N1, OrientDB N1 and MongoDB S1	CentOS 7.4	Cassandra and MongoDB (extended from Debian Linux) OrientDB (extended from Alpine Linux)	2	4 GB	30-GB SSD
Instance-2	Cassandra N2, OrientDB N2 and MongoDB S2	CentOS 7.4	Cassandra and MongoDB (extended from Debian Linux) OrientDB (extended from Alpine Linux)	2	4 GB	30-GB SSD
Instance-3	MongoDB Primary Node	CentOS 7.4	Alpine Linux	2	4 GB	30-GB SSD
Instance-4	PROV-DM Workflow Runner	Ubuntu 14.04	ProvBio (extended from Biolinux)	2	4 GB	50-GB SSD

Abbreviations: CPU, central processing unit; PROV-DM, Provenance Data Model; RAM, random access memory.

Cloud environments

Three different computational resource providers were used running in 2 public cloud providers (DigitalOcean and Google Cloud) and a private corporate cloud. Each environment has VMs using Docker, which is an open-source technology that both allows the management of application in containers and can be used together with IaaS for provisioning workflow environments with a negligible performance impact.[41,70]

A container-based application fully encapsulates the shipped software, providing all the required dependencies and ensuring the same libraries and software packages during the building and running processes.[41,71] A BioLinux[72] Docker image forked to a new image called ProvBio, which contains all the tools and settings used by each workflow. The entire environment can be reproduced by following the steps available at https://github.com/polyane/dataprovenance

Regarding the VM environment, we chose 4 Centos7 VMs running Docker, where we kept the workflow and the NoSQL systems in separate containers, as presented in Figure 1A. The databases (MongoDB, OrientDB, and Cassandra) were hosted in instances 1, 2 and 3, which ran over Alpine Linux and Debian Docker containers (Figure 1B).

The Cassandra and OrientDB platforms were configured with 2 nodes (N1 and N2), whereas MongoDB was set up with 1 primary node and 2 secondary nodes (S1 and S2). The workflow machine "ProvBio" is hosted in instance 4. It has the raw data and runs a ProvBio Docker container forked from Biolinux with additional tools sraToolkit, sickle, Hisat2, SAMtools, Quast, and Trimmomatic (Figure 1C). During the executions, there was no computational resource allocation beyond that was granted initially. The instance specifications are described in Table 2.

Data provenance schemas

Based on the PROV-DM, we defined schemas for persisting the data provenance in NoSQL Column-oriented, Document-oriented, and Graph-based families considering specially designed schemas for each of them. In addition to the typical workflow entities: *Agent*, *Experiment*, *Activity*, and *Project*, we included the *Machines* and *Environment* entities related to the execution of the experiments in the cloud environment.

The *Environment* entity has information about the cloud as the cloud provider, the location, and the number of machines in the environment. The *Machine* entity has information pertinent to the machines present in the execution environment of the workflow such as price, type of billing, operating system, among others. These components are vital for managing the data provenance of an experiment executed in the cloud. It makes it possible to reproduce both the experiment and the conditions of its environment. Besides, it permits us to get tracking the origin of the data.

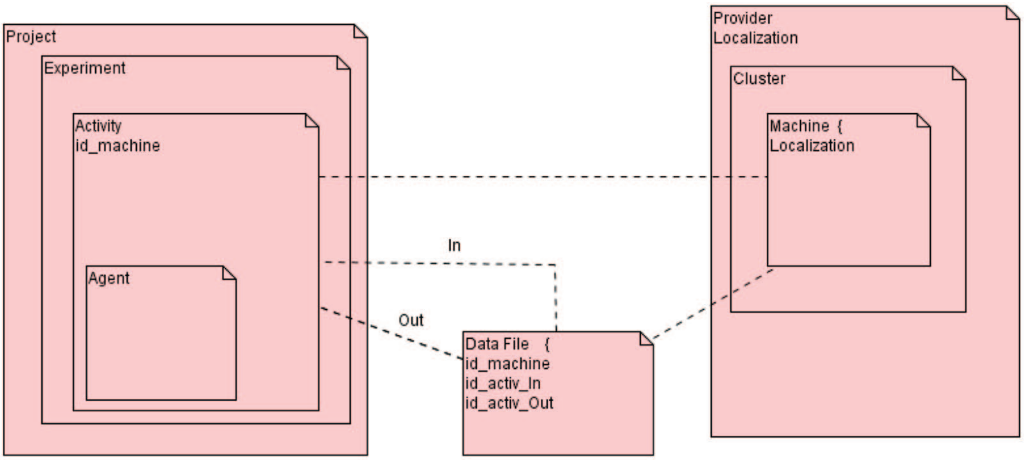


Figure 2. Document data schema.

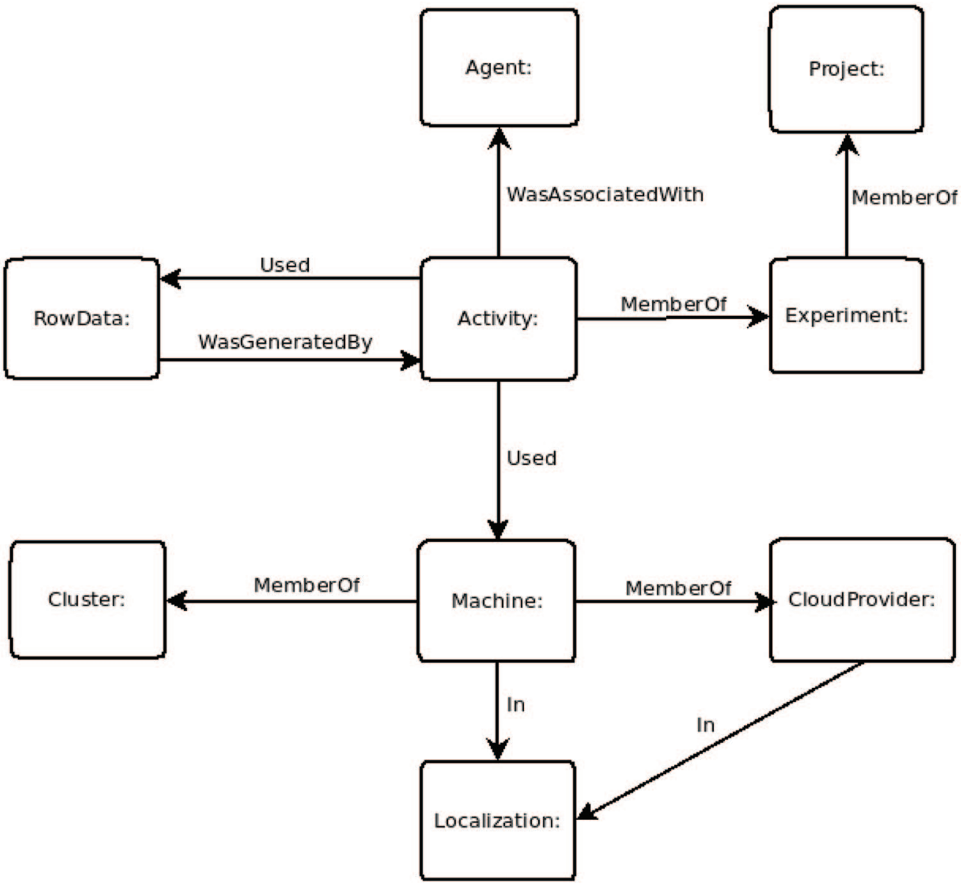
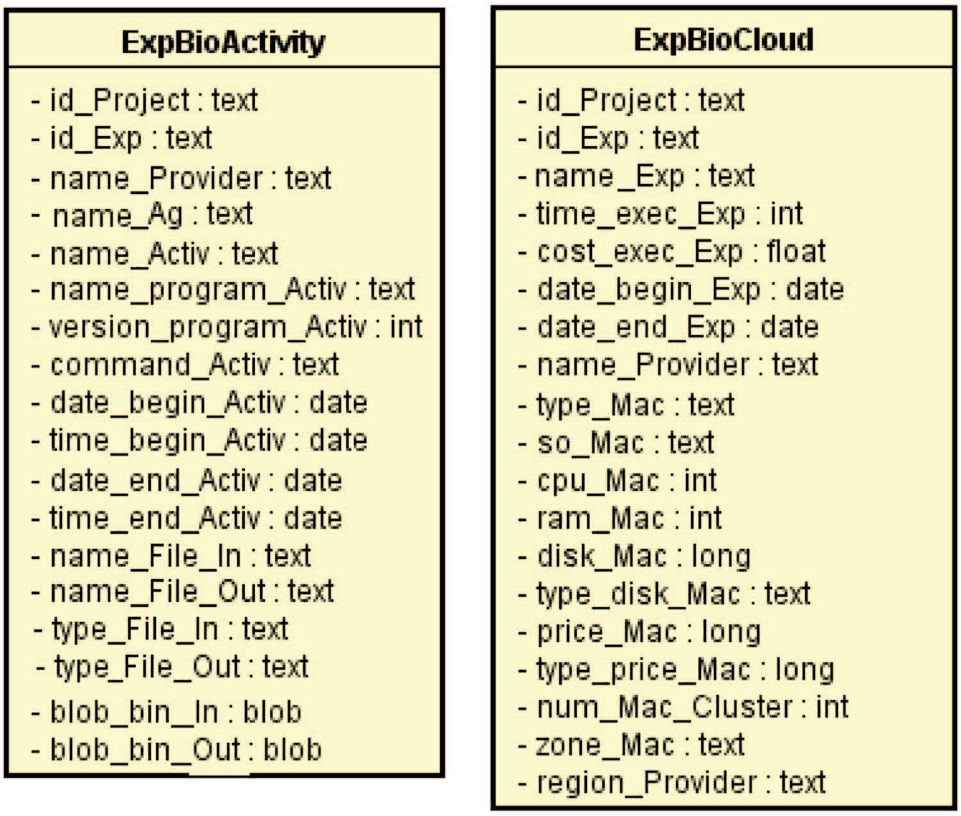


Figure 3. Graph database schema.

MongoDB can store documents under collections in 2 ways: either reference, when there is a reference or link between 2 related documents, or by embedded documents, which arranged in fields or arrays. In the document-based schema proposed here, we used both approaches. Figure 2 shows the embedded documents between the Project, Experiment, Activity, and Agent entities, as well as between the Provider, Location, Cluster, and Machine entities. Activity and Machine entities represent the relationship by reference, describing workflow activity running on a particular machine in the cloud, as well as stored used/generated files on that machine.

OrientDB is a hybrid database system that implements key-value, document, and graph models. Graph databases allow the storage of entities in nodes and edges, which can have properties. The edges have directional significance and indicate how the connected nodes are related. Figure 3 shows the proposed graph database schema for the storage of provenance in OrientDB using GRAPHED notation.[73]



ExpBioActivity

- id_Project : text
- id_Exp : text
- name_Provider : text
- name_Ag : text
- name_Activ : text
- name_program_Activ : text
- version_program_Activ : int
- command_Activ : text
- date_begin_Activ : date
- time_begin_Activ : date
- date_end_Activ : date
- time_end_Activ : date
- name_File_In : text
- name_File_Out : text
- type_File_In : text
- type_File_Out : text
- blob_bin_In : blob
- blob_bin_Out : blob

ExpBioCloud

- id_Project : text
- id_Exp : text
- name_Exp : text
- time_exec_Exp : int
- cost_exec_Exp : float
- date_begin_Exp : date
- date_end_Exp : date
- name_Provider : text
- type_Mac : text
- so_Mac : text
- cpu_Mac : int
- ram_Mac : int
- disk_Mac : long
- type_disk_Mac : text
- price_Mac : long
- type_price_Mac : long
- num_Mac_Cluster : int
- zone_Mac : text
- region_Provider : text

Figure 4. Column family data schema.

Cassandra is a non-relational database system based on columns.[74] A good practice when using Cassandra is to denormalize the database making it devoid of relationships. In Cassandra, it is important to analyze and implement the queries that will be performed by the application in the database to generate suitable and efficient tables. Data model using Cassandra was designed to achieve the minimum response time, even in complex operations, without joins or aggregations present in the SQL language. Thus, to analyze all the information necessary to answer biologists' questions, a query-oriented model was proposed based on how the data will be accessed. Figure 4 shows the model developed for Cassandra.

Table 3. Total times for workflow execution, capture and storage of data provenance.

	DIGITALOCEAN	GOOGLE CLOUD	PRIVATE CLOUD
W1	1h40m11s	20m06s	32m34s
W2	1h27m29s	18m07s	41m35s
W3	2h09m49s	1h15m18s	1h33m01

Results and Discussion

There are technical and practical aspects of the findings of this article. It was achievable to implement the PROV-DM model in different NoSQL database systems to store the data provenance. Unlike relational databases, where data modeling is well established through normal forms, NoSQL databases demand a particular study to achieve a suitable data model. NoSQL databases are schemaless, and notwithstanding providing flexibility for the data model, it brings a significant responsibility to the database administrator. Here, we presented data models for 3 types of NoSQL to store data provenance according to PROV-DM properly.

PROV-DM has already been shown to be suitable for storing data sources from Bioinformatics workflows,[36,42] even though it was not created for this purpose explicitly. With its components, PROV-DM is capable of, respectively, dealing with agents, entities, activities, and the precedence at which they were created, used, or ended.

The Workflows 1 (W1), 2 (W2), and 3 (W3) in each cloud environment were repeatedly executed and returned consistent results regarding the generated data in the workflows for each execution. Google Cloud had an advantage in execution time when compared with DigitalOcean and the private cloud, delivering a better performance, as can be seen in Table 3. Regarding the execution time, and consequently, the costs, there was only a little variation detected. These results enable an evaluation of the cost based on the execution history, which will be a target issue for future works.

Among the service models available in cloud computing, the use of IaaS confers advantages: it can collaborate with information security issues as the data are less exposed than in other service models such as SaaS, for instance, and increase

Table 4. Query examples in each NoSQL for common questions about data provenance.

WHAT ARE THE NAMES AND VERSIONS OF THE PROGRAMS USED TO PERFORM THE WORKFLOW 1 ACTIVITIES?	
MongoDB	db.project.aggregate([{$match:{$and:[{id:”1”}, {“experiment.id”:”1”},]}}, {$unwind:”$experiment.activity”}, {$group:{_id:{program:”$experiment.activity.program_name”, version:‘$experiment.activity.program_version’}}}])

Cassandra	SELECT name_program_Activ, version_program_Activ
	FROM provenance.ExpBioActivity WHERE id_Exp=1;

OrientDB	SELECT program, version FROM activity GROUP BY program WHERE id_Exp=1;
WHAT IS THE GOOGLE CLOUD CONFIGURATION USED TO RUN THE WORKFLOW (MACHINES, PROCESSORS, MEMORY, ETC.)?	
MongoDB	db.provider.find({name_Provider:”Google Cloud”, “cluster.name_Cluster”:”provBio”}, {_id:0, “cluster.num_Machine_Cluster”:1, machine:1})

Cassandra	SELECT num_Mac_Cluster, type_Mac, so_Mac, cpu_Mac, ram_Mac, disk_Mac,
	disk_type_Mac, price_Mac, region_Mac, zone_Mac
	FROM provenance.ExpBioCloud WHERE name_Provider=“Google Cloud”;

OrientDB	SELECT num_mac_cluster, type, operation_system, cpu, ram_memory, disk,
	disk_type, localization, price, billing_type FROM machine
	WHERE name_Provider=“Google Cloud”;

the control and flexibility of the computational environment. The flexibility of an IaaS environment deployment brings an increase in configuration work compared with SaaS, and this is precisely a point where the seized solution proves useful deploying the Bioinformatics workflows to the cloud with minimal technical effort. This study conceptually differs from the service model of proposals such as Galaxy,[44] as it works with IaaS instead of SaaS.

Some of the Bioinformaticians' most common questions about data provenance were answered using the query languages of each database. Each database had its own query language and, in some cases, it was necessary to create specialized functions, such as to obtain the total cost of the workflow, and that required changing some initial configuration parameters of the databases. MongoDB, which has native support for the arithmetic operators, was an exception. Table 4 shows samples for queries according to the query language of each NoSQL database system.

Beyond the technical results, the findings also lead us to some practical considerations this work cover. The omics data sequencing is expanding the demand for biological data analyses, including small labs, or even individuals, who can now access facilities or even have a portable sequencing platform as MinION[75] for an affordable price. In contrast, maintain a data center is not affordable, and an alternative solution is to hire a cloud service to run the analyses.

The presented provenance data schemas have included workflow data as well as metadata for cloud configuration. In this way, once the cloud contract is terminated, all the workflow and its data will be stored in a chosen NoSQL database to be easily deployed in any new cloud. This is a key contribution of this work, making it possible to reproduce the experiment in different clouds with limited effort for computational environment configuration. The presented conceptual data schemas efficiently meet for the reproducibility requirements of the experiment in the cloud by ensuring the compatibility of the computing platform deployed by a researcher following good practices proposed by Kanwal et al.[7]

From the chosen NoSQL families, the graph database system was the most adherent to the PROV-DM model, allowing persistence of the provenance graph in its native form. Moreover, it is possible to generate a picture of the data provenance aligned to the graphical representations of the PROV-DM from a query using an application programming interface (API) as Prefuse.[76] Figure 5 shows the data provenance for the W1. It is possible to see Ag_Fernanda, the executor agent of the At _Map_Hisat2 mapping activity, whose input data are chromosome.22.hisat2.idx and SRR5181508_FILTERED.fastq, and the result data SRR5181508.sam. Figure 6 shows the data provenance for the W2, where Ag_Fernanda executed the At_Filt_Trimmomatic activity using the inputERR885455_1.fastq and ERR885455_2.fastq, resulting in the filtered files.

Behind, Figure 7 shows the filtering phase data provenance for the W3. The input data are ERR037801_1.fastq and ERR037801_2.fastq. It is possible to see the executor agent Ag_Polyane of the At_Filt_Trimmomatic activity, resulting in the filtered files.

Conclusions

Bioinformatics workflows have contributed significantly to solving biological problems through omic data analysis. As with every scientific experiment, reproducibility is an important factor, plus in in silico experiments, extra elements need to be considered to ensure that factor. Two of these elements are the computational environment and the data provenance. Along with these elements, there are practical issues that

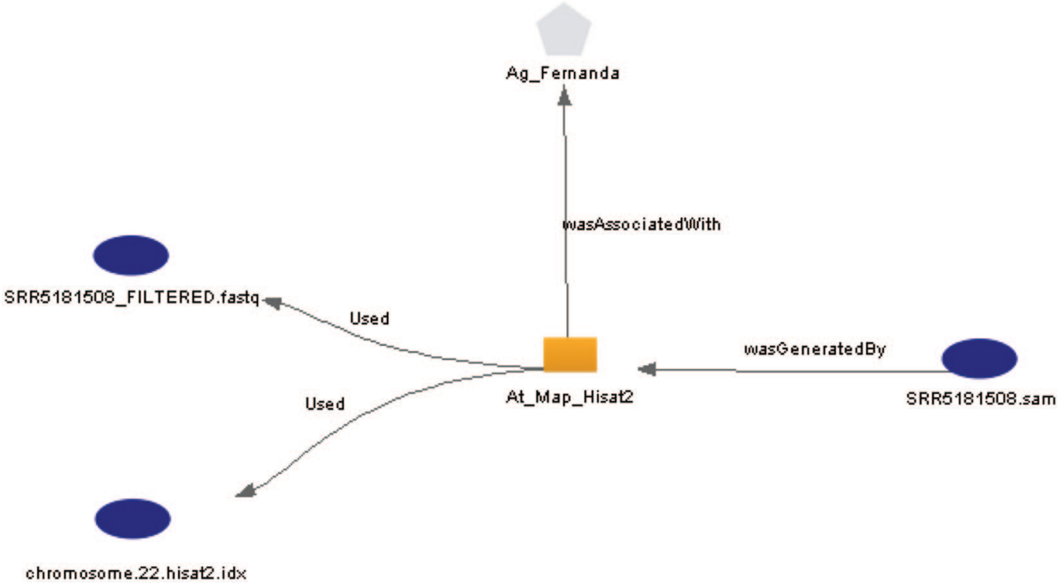


Figure 5. Graph of provenance generated from the mapping of Workflow 1.

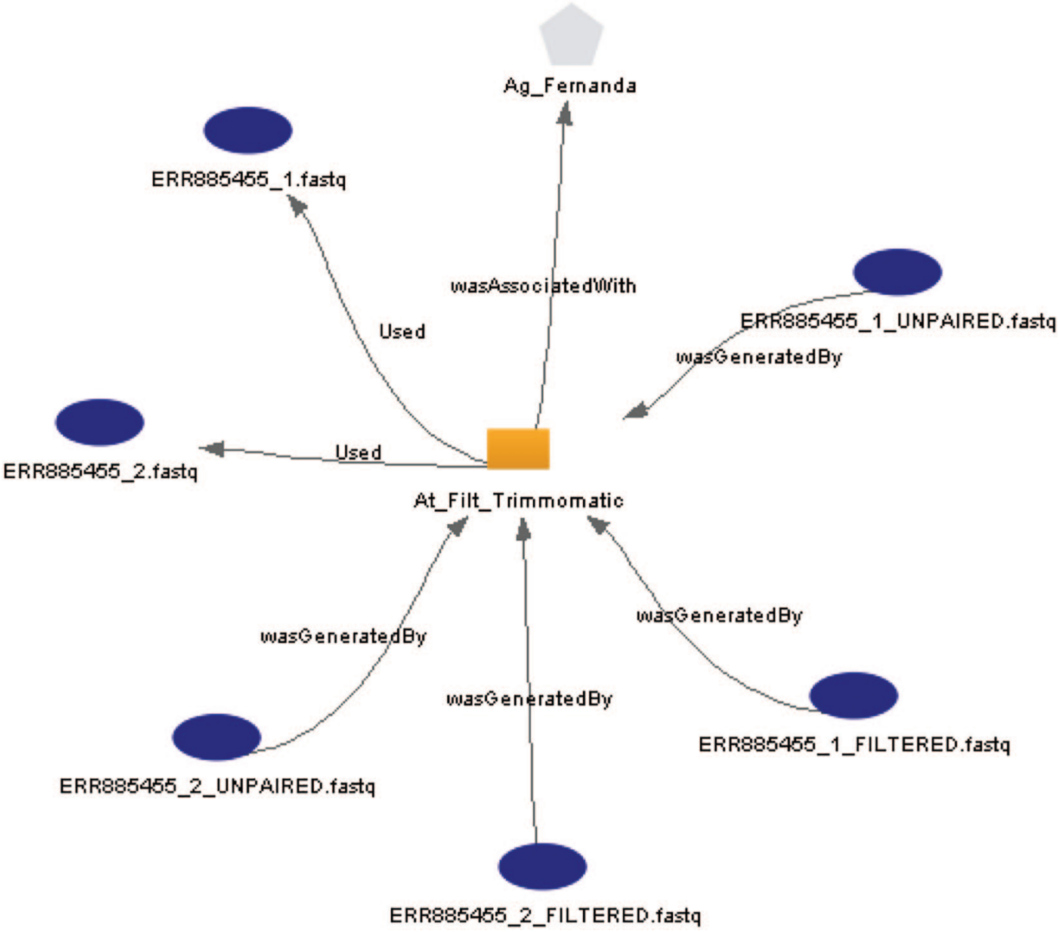


Figure 6. Graph of provenance generated from the filtering of Workflow 2.

impose some restrictions, such as resources for maintaining such an environment, time, skilled operators, and the storage of data provenance in a retrieval way.

In this study, we explored the service model IaaS of cloud computing environment while capturing and persisting data provenance of typical Bioinformatics workflows in NoSQL databases from 3 different families. The data provenance supports the capture of both data and its trajectory, as well as metadata about the computational environment. Thus, the findings showed that the proposed schemas for storing the

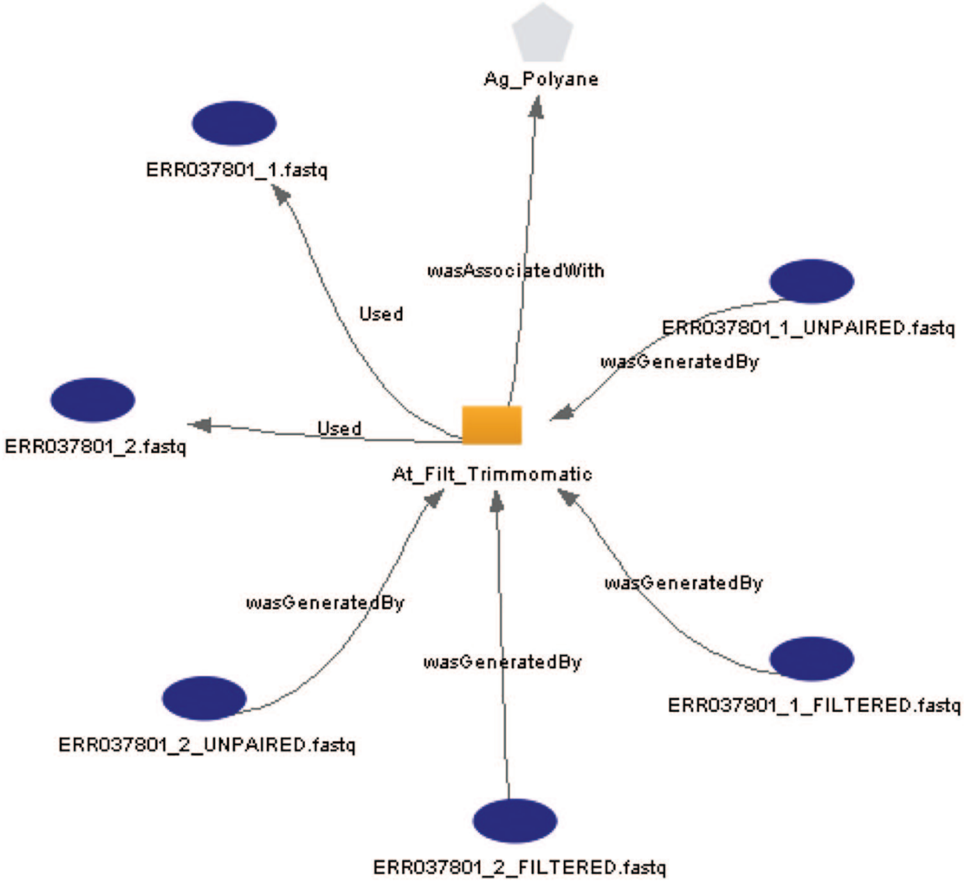


Figure 7. Graph of provenance generated from the filtering of Workflow 3.

provenance data in NoSQL databases enabled replication of the experiments by allowing them to be deployed in a cloud environment with low technical efforts, offering a viable alternative.

Regarding the time, all used NoSQL databases presented good performance for storing the data, displaying an insignificant difference in the data storage time. In our evaluation, the graph database is the best choice among the examined databases because it is more adherent to the PROV-DM. In future works, we intend to investigate issues related to data security and cost prediction based on the history of executions stored from the data source.

Author Contributions

PW, MH, and WDS were conceptualized the article; PW, WDS, FH, KC, MEW, and SL were methodology of this research; PW and FH implemented the software; WDS pipelined this study; all experiments were supervised by AA and MH; and PW wrote and first drafted the manuscripts. All authors contributed to writing and editing.

ORCID iDs

Polyane Wercelens https://orcid.org/0000-0002-8494-1267
Waldeyr da Silva https://orcid.org/0000-0002-8660-6331
Sergio Lifschitz https://orcid.org/0000-0003-3073-3734
Maristela Holanda https://orcid.org/0000-0002-0883-2579

Supplemental Material

Supplemental material for this article is available online.

REFERENCES

1. Leipzig J. A review of bioinformatic pipeline frameworks. *Brief Bioinform*. 2017;18:530-536.
2. Mattoso M, Werner C, Travassos G, Braganholo V, Murta L. Gerenciando experimentos científicos em larga escala. *SBC-SEMISH*. 2008;8:121-135.
3. Rosa M, Moura B, Vergara G, et al. Bionimbuz: a federated cloud platform for Bioinformatics applications. In *International Conference on Bioinformatics and Biomedicine (BIBM)*. New York, NY: IEEE; 2016:548-555.
4. Kohl M, Megger DA, Trippler M, et al. A practical data processing workflow for multi-omics projects. *Biochim Biophys Acta*. 2014;1844:52-62.
5. Stephens ZD, Lee SY, Faghri F, et al. Big data: astronomical or genomical? *PLoS Biol*. 2015;13:e1002195.
6. Schadt EE, Linderman MD, Sorenson J, Lee L, Nolan GP. Computational solutions to large-scale data management and analysis. *Nat Rev Genet*. 2010;11:647.
7. Kanwal S, Khan FZ, Lonie A, Sinnott RO. Investigating reproducibility and tracking provenance—a genomic workflow case study. *BMC Bioinformatics*. 2017;18:337.
8. Li T, Liu L, Zhang X, Xu K, Yang C. Provenancelens: service provenance management in the cloud. In *Proceedings of the 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*. New York, NY: IEEE; 2014:275-284.
9. Buneman P, Khanna S, Wang-Chiew T. Why and where: a characterization of data provenance. In *International Conference on Database Theory*. Cham, Switzerland: Springer; 2001:316-330.

10. Cheah Y, Canon R, Plale B, Ramakrishnan L. Milieu: lightweight and configurable big data provenance for science. In *IEEE International Congress on Big Data*. New York, NY: IEEE; 2013:46-53.
11. Ram S, Liu J. Understanding the semantics of data provenance to support active conceptual modeling. In *International Workshop on Active Conceptual Modeling of Learning*. Cham, Switzerland: Springer; 2006:17-29.
12. Sahoo SS, Sheth AP. Provenir ontology: towards a framework for escience provenance management. https://corescholar.libraries.wright.edu/knoesis/80/. Updated 2009.
13. Hartig O, Zhao J. Publishing and consuming provenance metadata on the web of linked data. *IPAW*. 2010;6378:78-90.
14. OPM. http://openprovenance.org. Updated 2018.
15. W3C. PROV-DM. www.w3.org/TR/prov-dm. Updated 2018.
16. Da Rosa Righi R. Elasticidade em cloud computing: conceito, estado da arte e novos desafios. *Revista Brasileira de Computação Aplicada*. 2013;5:2-17.
17. Mell P, Grance T. The NIST definition of cloud computing. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf. Updated 2011.
18. Loeffler B. Cloud computing: what is infrastructure as a service. *Technet Magazine*. https://docs.microsoft.com/en-us/previous-versions/technet-magazine/hh509051(v=msdn.10)?redirectedfrom=MSDN. Updated 2011.
19. Singh S, Jeong YS, Park JH. A survey on cloud computing security: issues, threats, and solutions. *J Netw Comput Appl*. 2016;75:200-222.
20. Singh A, Chatterjee K. Cloud security issues and challenges: a survey. *J Netw Comput Appl*. 2017;79:88-115.
21. Han J, Haihong E, Le G, Du J. Survey on NoSQL database. In *Proceedings of the 6th International Conference on Pervasive computing and applications (ICPCA)*. New York, NY: IEEE; 2011:363-366.
22. Corbellini A, Mateos C, Zunino A, Godoy D, Schiaffino S. Persisting big-data: the NoSQL landscape. *Inform Syst*. 2017;63:1-23.
23. Gessert F, Wingerath W, Friedrich S, Ritter N. NoSQL database systems: a survey and decision guidance. *Comput Sci Res Develop*. 2017;32:353-365.
24. Hecht R, Jablonski S. NoSQL evaluation: a use case oriented survey. In *International Conference on Cloud and Service Computing (CSC)*. New York, NY: IEEE; 2011:336-341.
25. Project Voldemort. Project Voldemort: a distributed database. http://project-voldemort.com. Updated 2019.
26. Salvatore Sanfilippo, et al. Redis. https://redis.io. Updated 2019.
27. Apache. HBase. https://hbase.apache.org. Updated 2019.
28. Apache. Cassandra. http://cassandra.apache.org. Updated 2019.
29. MongoDB Inc. MongoDB. https://www.mongodb.com. Updated 2019.
30. Apache. CouchDB. http://couchdb.apache.org. Updated 2019.
31. Neo4j Inc. Neo4j graph database. https://neo4j.com. Updated 2019.
32. ArangoDB Inc. ArangoDB. https://www.arangodb.com. Updated 2019.
33. Callidus Software Inc. OrientDB. https://orientdb.com. Updated 2019.
34. De Brevern AG, Meyniel JP, Fairhead C, Neuveglise C, Malpertuy A. Trends in it innovation to build a next generation Bioinformatics solution to manage and analyse biological big data produced by ngs technologies. *Biomed Res Int*. 2015; 2015:904541.
35. Da Silva WMC, Wercelens P, Walter MEM, et al. Graph databases in molecular biology. In *Brazilian Symposium on Bioinformatics*. Cham, Switzerland: Springer; 2018:50-57.
36. De Paula R, Holanda M, Gomes LS, Lifschitz S, Walter ME. Provenance in Bioinformatics workflows. *BMC Bioinformatics*. 2013;14:S6.
37. Ferreira GR, Filipe C Jr, de Oliveira D. Uso de SGBDs NoSQL na gerência da proveniência distribuída em workflows científicos. In *XXIX Simpósio Brasileiro de Bancos de Dados*. https://pdfs.semanticscholar.org/e313/8a19f09f514af7e4de11ccae40cc0d0745f6.pdf. Updated 2014.
38. Aniceto R, Xavier R, Guimaraes V, et al. Evaluating the cassandra NoSQL database approach for genomic data persistency. *Int J Genomics*. 2015;2015:502795.
39. Sempéré G, Philippe F, Dereeper A, Ruiz M, Sarah G, Larmande P. Gigwa-genotype investigator for genome-wide analyses. *Gigascience*. 2016;5:25.
40. Chacko AM, Basheer AM, Kumar SDM. Capturing provenance for big data analytics done using SQL interface. In *UP Section Conference on Electrical Computer and Electronics (UPCON)*. New York, NY: IEEE; 2015:1-6.
41. Costa RL, Gadelha L, Ribeiro-Alves M, Porto F. Gennet: an integrated platform for unifying scientific workflows and graph databases for transcriptome data analysis. *PeerJ*. 2017;5:e3509.
42. Almeida R, da Silva W, Castro K, et al. Aprovbio: an architecture for data provenance in Bioinformatics workflows using graph database. In *International Conference on Bioinformatics and Biomedicine (BIBM)*. New York, NY: IEEE; 2017:2139-2144.
43. Costa F, Silva V, De Oliveira D, et al. Capturing and querying workflow runtime provenance with PROV: a practical approach. In *Proceedings of the Joint EDBT/ICDT Workshops*. New York, NY: ACM Press; 2013:282-289.
44. Afgan E, Baker D, van den Beek M, et al. The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic Acids Res*. 2016;44:W3-W10.
45. Sadedin SP, Dashnow H, James PA, et al. Cpipe: a shared variant detection pipeline designed for diagnostic settings. *Genome Med*. 2015;7:68.
46. Welivita A, Perera I, Meedeniya D. An interactive workflow generator to support bioinformatics analysis through GPU acceleration applications. In *International Conference on Bioinformatics and Biomedicine (BIBM)*. New York, NY: IEEE; 2017:457-462.
47. Welivita A, Perera I, Meedeniya D, Wickramarachchi A, Mallawaarachchi V. Managing complex workflows in bioinformatics: an interactive toolkit with GPU acceleration applications. In *International Conference on Transactions on Nanobioscience*. New York, NY: IEEE; 2018:199-208.
48. Lenadora D, Wickramarachchi A, Meedeniya D, Mallawaarachchi V, Perera I. An adapter architecture for heterogeneous data processing in bioinformatics pipelines applications. In *International Conference on 2019 Moratuwa Engineering Research Conference (MERCon)*. New York, NY: IEEE; 2019:692-697.
49. Curcin V, Bottle A, Molokhia M, Millett C, Majeed A. Towards a scientific workflow methodology for primary care database studies. *Stat Methods Med Res*. 2010;19:378-393.
50. Schatz MC, Langmead B, Salzberg SL. Cloud computing and the DNA data race. *Nat Biotechnol*. 2010;28:691-693.
51. Ko G, Kim PG, Yoon J, et al. Closha: Bioinformatics workflow system for the analysis of massive sequencing data. *BMC Bioinformatics*. 2018;19:43.
52. Krampis K, Booth T, Chapman B, et al. Cloud biolinux: pre-configured and on-demand Bioinformatics computing for the genomics community. *BMC Bioinformatics*. 2012;13:42.
53. Docker Inc. Docker. https://www.docker.com. Updated 2019.
54. Zerbino DR, Birney E. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome Res*. 2008;18:821-829.
55. Bleidorn C. Assembly and data quality. In Bleidorn C, ed. *Phylogenomics*. Cham, Switzerland: Springer; 2017:81-103.
56. Haas BJ, Papanicolaou A, Yassour M, et al. De novo transcript sequence reconstruction from RNA-seq using the trinity platform for reference generation and analysis. *Nat Protoc*. 2013;8:1494-1512.
57. Guo Y, Ye F, Sheng Q, Clark T, Samuels DC. Three-stage quality control strategies for DNA re-sequencing data. *Brief Bioinform*. 2014;15:879-889.
58. Joshi N, Fass J. *Sickle: A Sliding-Window, Adaptive, Quality-Based Trimming Tool for FastQ Files*. Version 1.33 (software). https://github.com/najoshi/sickle. Updated 2011.
59. Li H, Handsaker B, Wysoker A, et al. The sequence alignment/map format and SAMtools. *Bioinformatics*. 2009;25:2078-2079.
60. Kim D, Langmead B, Salzberg SL. HISAT: a fast spliced aligner with low memory requirements. *Nat Methods*. 2015;12:357-360.
61. Anders S, Pyl PT, Huber W. HTSeq—a python framework to work with high-throughput sequencing data. *Bioinformatics*. 2015;31:166-169.
62. Judge K, Hunt M, Reuter S, et al. Comparison of bacterial genome assembly software for minion data and their applicability to medical microbiology. *Microb Genom*. 2016;2:e000085.
63. Bolger AM, Lohse M, Usadel B. Trimmomatic: a flexible trimmer for illumina sequence data. *Bioinformatics*. 2014;30:2114-2120.
64. Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJ, Birol I. Abyss: a parallel assembler for short read sequence data. *Genome Res*. 2009;19:1117-1123.
65. Gurevich A, Saveliev V, Vyahhi N, Tesler G. Quast: quality assessment tool for genome assemblies. *Bioinformatics*. 2013;29:1072-1075.
66. Bankevich A, Nurk S, Antipov D, et al. SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J Comput Biol*. 2012;19:455-477.
67. Delcher AL, Harmon D, Kasif S, White O, Salzberg SL. Improved microbial gene identification with GLIMMER. *Nucleic Acids Res*. 1999;27:4636-4641.
68. Lakin SM, Dean C, Noyes NR, et al. MEGARes: an antimicrobial resistance database for high throughput sequencing. *Nucleic Acids Res*. 2016;45:D574-D580.
69. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. *J Mol Biol*. 1990;215:403-410.
70. Cheng G, Lu Q, Ma L, Zhang G, Xu L, Zhou Z. BGDMdocker: a docker workflow for data mining and visualization of bacterial pan-genomes and biosynthetic gene clusters. *PeerJ*. 2017;5:e3948.
71. Chamberlain R, Schommer J. Using docker to support reproducible research. https://doi.org/10.6084/m9.figshare.1101910. Updated 2014.
72. EON Network. Bio-linux 8. http://environmentalomics.org/whats-new-in-bio-linux-8/. Updated 2018.
73. Van Erven G, Silva W, Carvalho R, Holanda M. Graphed: a graph description diagram for graph databases. In Rocha Á, Adeli H, Reis LP, Costanzo S, eds. *Trends and Advances in Information Systems and Technologies*. Cham, Switzerland: Springer; 2018:1141-1151.
74. Chebotko A, Kashlev A, Lu S. A big data modeling methodology for Apache Cassandra. In *International Congress on Big Data (BigData Congress)*. New York, NY: IEEE; 2015:238-245.
75. Oxford Nanopore Technologies. MinION. https://nanoporetech.com/products/minion. Updated 2019.
76. Heer J, Card SK, Landay J. Prefuse: a toolkit for interactive information visualization. In *Human Factors in Computing Systems (CHI)*. New York, NY: ACM Press; 2005:421-430.