NGS4Cloud: Cloud-based NGS Data Processing*

João Forja[1,2], Alexandre Almeida[1,2], Alexandre P Francisco[1,3], José Simão[1,2], and Cátia Vaz[1,2]

[1] INESC-ID Lisboa
[2] Instituto Superior de Engenharia de Lisboa (ISEL / IPL)
[3] Universidade de Lisboa / Instituto Superior Técnico

Motivation and challenges: Next-Generation Sequencing (NGS) technologies are greatly increasing the amount of genomic computer data, revolutionizing the biosciences field and leading to the development of more complex NGS Data Analysis techniques [2]. These techniques, known as pipelines or workflows, consist of running and refining a series of intertwined computational analysis and visualization tasks on large amounts of data. These pipelines involve the use of multiple software tools and data resources in a staged fashion, with the output of one tool being passed as input to the next one. To simplify the design and execution of biomedical workflows by end users, especially those that use multiple software tools and data resources, a number of scientific workflow systems have been developed over the past decade. Examples include Galaxy [1] and Swift [3]. However, most of these scientific workflow systems cannot be easily deployed and most of the times are only available to users with access to specialized IT support. There are two main issues to address in the design of an execution environment to these pipelines. First, due to the complexity of configuring and parametrizing pipelines, the use of NGS Data Analysis techniques is not an easy task for a user without IT knowledge. Second, knowing input data can be as much as terabytes and petabytes, pipelines execution require, in general, a great amount of computational resources.

System Organization: Regarding the first challenge, NGS4Cloud is devised to allow easy design and use pipelines, without users need to configure, install and manage tools, servers and complex workflow management systems. The system offers a DSL (domain-specific language), to describe pipelines[4]. The DSL's syntax provides primitives to specify the sequence through which each tool command is executed, to specify arguments, and to chain commands' inputs and outputs. To know each tools' properties and commands, NGS4Cloud uses tools' meta-data which is saved in a user-provided repository.[5]

Regarding the second challenge, Cloud technologies are sought as a solution to solve the lack of resources of an average computer, providing users with big clusters of powerful machines to run pipelines more efficiently. NGS4Cloud deploys pipelines in a remote cluster. It analyses the pipeline's description and automatically explores multiples cores of the same machine or different machines, as well as data partitioning. We developed an execution engine that analyses the

*Extended Abstract. This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013.

[4] `https://github.com/ngspipes/dsl/wiki`, visited 15-06-2016

[5] `https://github.com/ngspipes/tools/wiki`, visited 15-06-2016

pipeline and builds a graph of tasks. This graph of tasks reflects the dependency among tasks and allows to infer what can be executed in parallel and what can only be executed in serial. It is organized as a directed acyclic graph (DAG) where vertices of the DAG are the tasks and edges represent the dependencies. From it, the engine deploys the tasks in a cluster of machines governed by the Mesos's batch job scheduling framework Chronos.

Parallelism is made simple to end-users. We consider three levels: i) multi-core execution of tool commands that support it; ii) parallel execution of independent tasks of a pipeline; iii) data partitioning in order to process the fragments in parallel executions of the same tool command. The optimal partitioning of the input files for parallel execution is not solved by NGS4Cloud. This responsibility is entirely delegated to the users when they describe the pipeline using the DSL. If a user decides to split a file, then it will be partitioned as requested. To support the parallelization of independent tools we automatically infer from the pipeline description dependencies based on the outputs and inputs of commands.

NGS4Cloud is currently deployed at the Portuguese National Distributed Computing Infrastructure (IaaS). We have configured the Mesos kernel at this infrastructure, which allows to easily manage clusters of machines assuring an efficient use of resources, provides fault tolerance and high availability, and has support for running Docker images.[6,7] NGS4Cloud uses Docker natively to isolate each tool for easy and up-to-date execution. Since pipeline tasks are batch jobs, we are using the Chronos framework, a batch job scheduler framework that runs on top of Mesos and offers a REST interface for scheduling jobs with dependencies.[8] The cluster has a distributed file system which serves as working directory for the execution of the pipeline, allowing all cluster machines to access the files being produced. A software component called `Monitor` is responsible for reading a description of the tasks and its dependencies. Chronos is used to schedule these tasks at a Mesos' controlled cluster. The use of these frameworks decouples our solution from any concrete cloud service. Since Mesos and Chronos are currently being actively developed, we hope NGS4Cloud will be able to take advantage of improvements to this technologies.

References

1. Giardine, B., Riemer, C., Hardison, R.C., Burhans, R., Elnitski, L., Shah, P., Zhang, Y., Blankenberg, D., Albert, I., Taylor, J., et al.: Galaxy: a platform for interactive large-scale genome analysis. Genome research 15(10), 1451–1455 (2005)
2. Shuster, S.C.: Next-generation sequencing transforms today's biology. Nature Methods 5(1), 16–18 (Jan 2008)
3. Wozniak, J.M., Wilde, M., Foster, I.T.: Language features for scalable distributed-memory dataflow computing. In: Data-Flow Execution Models for Extreme Scale Computing (DFM), 2014 Fourth Workshop on. pp. 50–53. IEEE (2014)

[6] `http://mesos.apache.org/`, visited 15-06-2016

[7] `https://hub.docker.com/`, visited 15-06-2016

[8] `https://mesos.github.io/chronos/`, visited 15-06-2016