NGS4Cloud: Cloud-based NGS Data Processing

Projeto e Seminário

Licenciatura em Engenharia Informática e Computadores

Alexandre Almeida
alex_til@hotmail.com
915363779

João Forja
jnforja@gmail.com
915658139

Supervisors:
Cátia Vaz, ISEL, cvaz@cc.isel.ipl.pt
José Simão, ISEL, jsimao@cc.isel.ipl.pt
Alexandre P. Francisco, IST, aplf@ist.utl.pt

April 4, 2016

1 Introduction

Next-Generation Sequencing (NGS) technologies are greatly increasing the amount of genomic computer data, revolutionizing the biosciences field leading to the development of more complex NGS Data Analysis techniques (Shuster, 2008). 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.

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. Moreover, knowing input data can be as much as terabytes and petabytes, pipelines execution require, in general, a great amount of computational resources.

NGSPipes framework is devised to solve the first issue, allowing to easily design and use pipelines, without users need to configure, install and manage tools, servers and complex workflow management systems (Dantas and Fleitas, 2015). However, as of now, it operates only in a single machine, which will most likely lack the necessary resources to execute the pipeline in a reasonable period of time, if at all.

Cloud technologies are sought as a solution to solve the lack of resources of an average computer, a feature that would substantially improve NGSPipes solution, providing users with big clusters of powerful machines to run pipelines more efficiently (Armbrust et al., 2010). We plan to integrate NGSPipes with the capability of running pipelines in a remote cluster, while standing to its current usability standards.

2 Requirements

To extend NGSPipes into a Cloud-based NGS Data Processing framework, challenges like managing cluster resources and scheduling tasks across the cloud environment need to be overcome. In addition to enabling NGSPipes's pipelines to be executed on a remote cluster, it would also be interesting to provide the pipelines with automatic parallelization in order to remove that burden from users hands.

Apache Mesos technology is gazed upon as a tool to solve the remote execution challenges.[1] It allows programmers to code as if against a single pool of resources, abstracting CPU, memory, storage, and other compute resources away from machines (physical or virtual). Mesos architecture has three main entities: frameworks, masters, and slaves. Slaves are the cluster machines where tasks will be executed. Masters are responsible for managing the resources provided by the slaves. Frameworks receive information from masters about resources offered by slaves, and make use of those resources to execute jobs posted by users. Custom frameworks can be created in order to produce one that interacts with users and uses resources as seen fit for a specific domain.

Adding up, Mesos supports the same container technologies as NGSPipes uses to install, set up, and run the pipeline tools on a machine: Docker[2]. Docker is an open-source project which wraps and extends Linux containers technology to bring a complete solution for the creation and distribution of containers. The Docker platform provides a vast number of commands to easily manipulate containers. Containers are a technology which isolates an application in an environment configured with all the dependencies needed for its execution; an encapsulation of an application together with its dependencies. The use of containers brings advantages such as:

- Having little to no overhead comparing to running an application natively, due to interacting directly with the host OS kernel, not existing any layer between the application running inside the container and the OS.
- Providing high portability since the application runs in the environment provided by the container; bugs related to runtime environment configurations will almost certainly not occur.

[1]`http://mesos.apache.org/`, visited in 30-03-2016
[2]`https://www.docker.com/`, visited in 02-04-2016

- Running dozens of containers at the same time, thanks to their lightweight nature.
- Executing an application by downloading the container and running it, avoiding going through possible complex installations and set up.

As for the parallelization of a pipeline, this can be split into three different degrees:

1. Some tools may take advantage of the multi-core architecture of most present machines.
2. Some tools allow for breaking down a collection of data in smaller chunks, processing them in parallel and joining the results once completed.
3. Some pipelines contain tasks that may run independently of each other.

The requirements for this project will be:

1. **The execution of pipelines on clusters of machines** - To execute the pipeline in a cluster, and efficiently use its resources, it is required a batch job scheduling Mesos framework that provides:

    - Support for Docker.
    - A client API to manage jobs, allowing a user to specify the relationship between them, to provide the pipeline input and output locations, to give users feedback about the jobs's state, and to specify the resources required to run each component of the pipeline.
    - Fault tolerance, meaning that even if a machine of the cluster where a task is being executed were to go off-line, the task will be relaunched in other available machine. It would be interesting if the new executing machine could relaunch the task from a point where it did not have to start over again, but proceed from a previously saved checkpoint.

    A component to coordinate the workflow's execution on the framework will also be necessary. If a framework matching all of the criteria is not found, this component will also have to implement the missing features.

2. **The parallelization of the execution of a pipeline** - In order to accomplish this requirement we will need to analyse the pipeline and determine which tools can be parallelized and to what degree. To determine to what degree a tool is parallelizable in runtime, tools will need to have meta-data associated which provides that information. Graph theory will further be used to analyse the pipelines and know what tasks are parallelizable. There may be occasions when the parallelization of the pipeline cannot be automatically deduced, hence changes to the NGSPipes's DSL may be necessary in order to allow users to explicitly describe the parallelization of pipeline steps.

This way, two main components will be developed: An analyser and a coordinator. The analyser is occupied with analysing the description of the pipeline, provided by a user, producing jobs representations which the framework understands and a workflow. The workflow produced by the analysis of the description of the pipeline will contain the information about which jobs are parallelizable. The coordinator's main responsibility is to coordinate the execution of the jobs specified in the pipeline. The coordinator will interact directly with the framework, scheduling the execution of the workflow.

3 Schedule

Date	Duration(in weeks)	Assignments
03/03/2016	1	- Introduction to Docker and Mesos technologies
10/03/2016	1	- Further study of Mesos technology - Introduction to Marathon and other Mesos frameworks
17/03/2016	1	- Write project proposal - Investigate existent batch job scheduling Mesos frameworks - Study Mesos custom frameworks
24/03/2016	1	- Write project proposal - Further study Mesos custom frameworks
31/03/2016	1	- Deliver project proposal - Study Chronos framework
07/04/2016	2	- Develop coordinator
21/04/2016	1	- Test coordinator - Wrap up progress report
28/04/2016	1	- Deliver progress report
05/05/2016	3	- Develop analyser
26/05/2016	1	- Test analyser
02/06/2016	1	- Create poster - Prepare Beta version for delivery
09/06/2016	1	- Deliver poster - Deliver Beta version
16/06/2016	2	- Test NGS4Cloud - Finish the report
30/06/2016	1	- Final delivery

References

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., and Zaharia, M. (2010). A view of cloud computing. *Commun. ACM*, 53(4):50–58.

Dantas, B. and Fleitas, C. (2015). Infraestrutura de suporte à execução de fluxos de trabalho para bioinformática. Diploma Thesis, Instituto Superior de Engenharia de Lisboa.

Shuster, S. C. (2008). Next-generation sequencing transforms today's biology. *Nature Methods*, 5(1):16–18.