

# Nispero: a cloud-computing based Scala tool specially suited for bioinformatics data processing

Evdokim Kovach, Alexey Alekhin, Marina Manrique, Pablo Pareja-Tobes,  
Eduardo Pareja, Raquel Tobes, and Eduardo Pareja-Tobes

Oh no sequences! Research Group, Era7 bioinformatics  
`eparejatobes@ohnosequences.com`

**Abstract.** Nowadays it is widely accepted that the bioinformatics data analysis is a real bottleneck in many research activities related to life sciences. High-throughput technologies like Next Generation Sequencing (NGS) have completely reshaped the biology and bioinformatics landscape. Undoubtedly NGS has allowed important progress in many life-sciences related fields but has also presented interesting challenges in terms of computation capabilities and algorithms. Many kinds of tasks related with NGS data analysis, as well as other bioinformatics data analysis, can be computed in a parallel, independent way; taking the maximum advantage of this can obviously help in leveraging the analysis bottleneck.

Given the way NGS data is generated scalability plays also an important role in its analysis. NGS data is not generated in a continuous fashion but in a batch way, thus the computation needs can be dramatically different at different points.

Cloud computing provides a perfect framework for systems with these two requirements: parallel and scalable. Besides, it allows adjusting the computation power on demand, and thus not being attached to (and paying for) a fixed compute infrastructure.

Nispero is a Scala library for declaring stateless computations and scaling them using cloud computing, in particular a combination of services from AWS (Amazon Web Services). Some highlights are:

- strongly typed configuration based on Scala code
- CRDT-like semantics (a nispero instance is essentially a morphism between idempotent commutative monoids)
- automatic deploy/undeploy

Nispero relies on the EC2 service (Elastic Compute Cloud) to carry out the computations, on the S3 service (Simple Storage Service) for data storage and on SQS (Simple Queue Service) and SNS (Simple Notification Service) for communication between the different system components.

A Nispero system is composed by:

- a “console” instance that tracks at any moment the status of the whole system giving the user the opportunity to check at any point the current status of the computations, workers, etc.

- a “manager” instance that is in charge of deploying and undeploying the group of workers
- a set of “workers” that performs the computations/tasks in a parallel, independent way
- SQS queues for “input”, “output” and “error” messages
- S3 objects for “input” and “output” files

The lifecycle of a Nispero system is simple but robust. It starts with the launch of the “console” and “manager” instances, the “manager” then takes the tasks from an S3 object, publishes them in a SQS queue and launches the workers. The workers take the messages with the tasks from the corresponding SQS queue (i.e. the “input” queue) in an independent, parallel way. Once they have finished the computation they put the results of the computation in S3 objects, publish a message in the “output” SQS queue and delete the input message of the corresponding task from the “input” queue.

Nispero is an open-source project released under AGPLv3 license.

The source code is available at <https://github.com/ohnosequences/nispero>.

This project is funded in part by the ITN FP7 project INTERCROSSING (Grant 289974)