1 Introduction

Elastreaming is a library intended for running executing bioinformatics tools and workflows on Amazon EC2 clusters. The main advantage of elastreaming is minimizng the execution times and thus the running costs by overlapping the uploading and computing times. There are two main components in elastreaming for:

- Creating and managing clusters
- Tools and workflows Execution

2 Administration

2.1 Requirements

- Unix-like OS
- Python > 2.5, the following modules are needed:
  - pyCrypto ≥ 2.3
  - simplejson
  - elementTree
- Sun Java > 1.5
- Ruby 1.8

2.2 Installation

- Download the client library from www.nubios.nileu.edu.eg/elastream/Downloads/elastream.tar.gz.
- Extract the package
- Navigate to the extracted directory and run the following command:

  ```
  python setup.py
  ```

2.3 AMI Image

This is a public AMI on Amazon AWS that contains

- A set of bioinformatics tools, complete list can be found in the resources page at
  http://www.nubios.nileu.edu.eg/elastream
- Tavaxy engine and Taverna Workflow System installed.
- Server side of Elastic Cloud library installed.

The image can be edited either to add more tools or databases, this can be done by the following:

1. Instantiate a single machine instance of the image (recommended machine type is m1.large)
2. Perform the needed changes.
3. Execute the following the code

   ```
   sudo /root/clear
   ```

4. Create a new image of the instance
5. Terminate the instance
6. Use the new AMI-ID for creating clusters for further use

3 Creating and Managing Clusters

This component is repsonsible for two main functions:

- Creating a cluster from EC2 compute nodes.
- Terminating an already establised cluster.

The following describes how to use the command line interface for managing the clusters: running:

```
python cloudWrapper.py [options]
-m [create] either create or terminate
-d [""] required when -m is terminate
-n [1] number of nodes
-t [m1.large] type of node (m1.large, m1.xlarge, c1.xlarge)
-s [default] security group for compute nodes
-r [EU1] region to install the cluster in
-a [XXX] ami ID to use in the cluster
-k [required] key pair to start compute nodes with
-p [required] path to private key file
-c [required] path ot certificate
```

-m

This defines which mode is requried, *create*, to create a new cluster, or *terminate* to terminate an already established cluster.

-d

This is required when the the mode is *termiante*, this is the domain of the main node of the cluster to be terminated.

-n

This defines the number of nodes the cluster will be composed of.

-t

This defines the type of nodes that will compose the cluster.

-s

This is the name of the security group to be used for the cluster nodes.

-r

This is the region at which the cluster will be installed, it take one of the following value:

- EU1
- USE1
- USW1
- Asia1

-a

This is the amazon id for the image to be used for the cluster nodes, you can check the website www.nubios.nileu.edu.eg/elastream for the latest supported ami id.

-k

This is the name of the key pair used to access the nodes.

-p

This is the path to the private key file

-c

This is the path to the certificate

4 Modes of Operation

There are different modes of operation where elastreaming can be used to run tools or workflows:

- running single tool on the cloud from command line
- running a single tool within a workflow on the cloud

5 Usage

5.1 Single Tool

There is a command line interface for running tools on the cloud using Elastreaming. The following describes how it is used:

```
python StreamingWrapper [options]
--infrastrucutre, -i
--domain, -d
--inputsmap, -n
--outputsmap, -o
--command, -c
--inputX
--inputXsplitter
--inputXchunkSize
--outputX
```

5.1.1 –infrastructure

This defines whether the tool is to run on an EC2 cluster (–infrastructure=cloud) of an EMR cluster (–infrastrucutre=emr).

5.1.2 –domain

This is needed when infrastructe is cloud, the tool is to be executed on EC2 cluster. It defines the domain of the cluster's main node. In case of EMR is going to be used, this options defines the limitations in the following syntax:

```
--domain={max num of clusters}_{num of nodes per cluster}_{node type}
```

Such that if the job is to be executed using a maximum of 2 clusters, each cluster composed of 4 c1.xlarge nodes it will be:

```
--domain=2_4_c1.xlarge
```

5.1.3 –command

This defines a path to the file containg the command of the tool to be executed. Inputs and outputs within the command are replaced with unique variable names starting with $.

5.1.4 –inputX

X is a number from 1 to 10. This defines the path of the file containg the data needed to be passed to the input port X.

5.1.5 –inputXsplitter, –inputXchunkSize

These two paramters define how the chunk is going to be prepared on the cloud side. Splitter can be a substring used to divide the input data or size and chunk size defines either how many parts are to be within each chunk in case of splitting on a substring or the size of the data within each chunk in case of splitting data according to size.

5.1.6 –inputmaps

This defines the mapping of input ports to the inputs names within the command.

5.1.7 –outputmaps

This defines the mapping of output ports to the outputs names within the command.

This is needed when the to

5.2 Single Tool within a Workflow

For this scenario currently there are two supported workflow systems but it can be extended to support more systems by adding their interface to Elastreaming.

5.2.1 Taverna

This is supported in any version of Taverna supporting the use of beanshell processors.

1. Add a beanshell processor to the workflow

2. Copy the beanshell script inside from the resources page at
   http://www.nubios.nileu.edu.eg/elastream
   into the added beanshell processor.

3. Add ten input ports and ten output ports with 0 depth.

4. Add the following input ports:

   - domain
   - infrastrucutre
   - command
   - inputsmap
   - outputsmap
   - inputXSplitter
   - inputXChunks

5.2.2 Galaxy

Elastreaming provides an interface for Galaxy workflow system, where two nodes have been developed, one to create a cluster and the other to run a job. To install those two nodes to Galaxy:

1. Download the following package http://www.nubios.nileu.edu.eg/elastream/galaxy.tar.gz.

2. Extract the downloaded package to a folder called *elastream* in the *tools* folders.

3. Copy the contents *dependencies* folder from the package to the *tool-data* folder inside a folder called *elastream*.

4. Navigate to the *elastream* folder inside the *tool-data* folder and run the following command:

   ```
   python setup.py
   ```

5. Add reference to the two nodes to the tool_conf.xml file

6. Restart Galaxy