DATMA USER MANUAL

DESCRIPTION:

DATMA is a distributed automatic pipeline for fast metagenomic analysis that includes: sequencing quality control, 16S-identification, reads binning, de novo assembly, ORF detection and taxonomic annotation. DATMA uses a distributed computing model that allows that different stages can be executed in multiple resources reducing the analysis time. DATMA is a freely available at https://github.com/andvides/DATMA

QUICK START:

DATMA is written in Python and has been tested in Linux with Ubuntu distribution.

- For a basic installation run the script install_datma with sudo properties. It configures DATMA with a basic configuration (NON COMPSs support) using the custom tools.

```
$sudo ./install_datma.sh
```

- To execute the simple test:
    - Download the controllledMetagenome.zip
    - Modified the configuration file according the path used in the installation process. It can be found into the controllledMetagenome folder.
    - Run the runDATMA.sh script use as arguments: the path to the configuration file and the running mode sequential (seq) or distributed (compss), it requires that COMPSs has been installed.

```
$ runDATMA.sh controlledMetagenome.txt seq
```

MANUAL INSTALLATION:

1. Download tools that form DATMA.

i) Quality Trimming and Filtering of Raw Reads tools:

Prinseq can be downloaded from http://prinseq.sourceforge.net/.

Fastx can be downloaded from http://hannonlab.cshl.edu/fastx_toolkit/.

RAPIFILT can be downloaded from https://github.com/andvides/RAPIFILT.

ii) 16S-identification

DEG, RDP, GreenGenes Grd, NCBI-16s, RDP, RFam, rnammer and silvaPFAM databases can be selected as reference. The FM-Index representation of each one can be downloaded from: https://github.com/andvides/DATMA/16Sdatabase

iii) CLAME binning

CLAME can be downloaded from: https://github.com/andvides/DATMA/tree/master/tools.

iv) Bin de novo assembly and annotation

Newbler can be downloaded from: http://sequencing.roche.com.

Velvet can be downloaded from: https://www.ebi.ac.uk/~zerbino/velvet/.

Spades can be downloaded from: http://cab.spbu.ru/software/spades/.

MegaHit can be downloaded from: https://github.com/voutcn/megahit.

Prodigal can be downloaded from: https://github.com/hyattpd/Prodigal.

GeneMark can be downloaded from: https://ngs.csr.uky.edu/GeneMark.

v) Final report

Krona can be downloaded from: https://github.com/marbl/Krona.

vi) COMPSs

COMPS can be downloaded from: https://www.bsc.es/research-and-development/software-and-apps/software-list/comp-superscalar.

2. Install all the tools according the author instruction and recommendations and add each one to the execution PATH. Probe that each tool can be executed from the DATMA path.

```
$ nano ~/.bashrc
# Add the following to the end of your .bashrc file
export PATH="/home/$USER/$Tool/bin:$PATH"
```

3. Download DATMA source codes into a specific folder. Source code can be download from: https://github.com/andvides/DATMA.

4. Configuration files:
   a. DATMA configuration file: In this file, the user specifies the input-sequences file, the output directory, the workflow stages, the databases directories, the number of threads to use, CLAME's bases parameter, etc.

```
################################################################################
################################################################################
##DATMA CONFIGURATION FILE
##
##USE THIS FILE TO PASS THE ARGUMENTS TO EACH
##TOOL USED IN THE DATMA WORK FLOW
##VERSION_2: 31-10-2018
##Uncomment the lines that you wish modify
################################################################################
################################################################################
##GENERAL PARAMETERS
#-start_in: INITIAL STAGE for the pipeline
#-inputFile: Full path and name for the input reads
#-outputDir: Full path and name for the output directory, <default ./output>
#-cpus: Number of threads used for each tool
#-typeReads Reads type <fasta,fastq, Illumina or SFF>
#
##QUALITY CONTROL
#-cleanTool: Select rapifilt, prinseq, or fastx (default rapifilt)
#-te: remove n bases from the end <only for rapifilt>
#-tb: remove n bases from the begin <only for rapifilt>
#-lq: Set lef-cut value for quality scores (int default 30)
#-rq: Set right-cut value for quality scores (int default 30)
#-m:  Delete sequences with size minor that (int default 70)
#-wq: Winwdows to check quality (int defatult 2)
#
##MERGE ILLUNIMA FILES USING FLASH TOOL
#-fb: Number of bases for set a merge (default 5)
#
##16S-REMOVE
#-database_16s_fasta: fasta sequences (default '~/DATMA/16sDatabases/rfam/RFAM_db.fasta')
#-database_16s_fm9: fm-index representation (default ~/DATMA/16sDatabases//rfam/rfam.fm9)
#-RDP_path: path to the RDP classifier tool (default '~/DATMA/tools/RDPTools')
#
##CLAME PARAMETERS
#-bases: Number of bases to use in the alignment stage <default 70,60,50,40,30,20>
#-sizeBin: Number of reads to report a bin <default 2000>
#-ld: Descrimine reads with a number of edges minor than this value <default 2>
#-nu: MAD distance <default 3>
#-w: windows offset for the alignment
#
#ASSEMBLY OPTIONS
#-assembly: Select newbler, velvet, spades, megahit (default spades)
#
##BLAST PARAMETERS
#-database_nt: Full path to the NT database of NCBI (default ~/DATMA/blastdb/nt)
#
#Kaiju
#database_kaiju: Full path to the Kaiju database (default ~/DATMA/tools/kaiju/kaijudb)
#
#Krona
#-combine: Uncomment this to merge bins output
```

b. resources.xml: provides information about the available execution resources.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResourcesList>
   <ComputeNode Name="172.16.7.105">
      <Processor Name="aletoso">
         <ComputingUnits>1</ComputingUnits>
      </Processor>
      <Adaptors>
         <Adaptor Name="integratedtoolkit.nio.master.NIOAdaptor">
            <SubmissionSystem>
               <Interactive/>
            </SubmissionSystem>
            <Ports>
               <MinPort>40001</MinPort>
               <MaxPort>43002</MaxPort>
            </Ports>
         </Adaptor>
         <Adaptor Name="integratedtoolkit.gat.master.GATAdaptor">
            <SubmissionSystem>
               <Batch>
                  <Queue>sequential</Queue>
               </Batch>
               <Interactive/>
            </SubmissionSystem>
            <BrokerAdaptor>sshtrilead</BrokerAdaptor>
         </Adaptor>
      </Adaptors>
   </ComputeNode>
   <ComputeNode Name="172.16.7.104">
      <Processor Name="mostro">
         <ComputingUnits>1</ComputingUnits>
      </Processor>
      <Adaptors>
         <Adaptor Name="integratedtoolkit.nio.master.NIOAdaptor">
            <SubmissionSystem>
               <Interactive/>
            </SubmissionSystem>
            <Ports>
               <MinPort>40001</MinPort>
               <MaxPort>43002</MaxPort>
            </Ports>
         </Adaptor>
         <Adaptor Name="integratedtoolkit.gat.master.GATAdaptor">
            <SubmissionSystem>
               <Batch>
                  <Queue>sequential</Queue>
               </Batch>
               <Interactive/>
            </SubmissionSystem>
            <BrokerAdaptor>sshtrilead</BrokerAdaptor>
         </Adaptor>
      </Adaptors>
   </ComputeNode>
</ResourcesList>
```

c. project.xml: provides information about the resources used in a specific execution.

```
<Project>
   <MasterNode/>
   <ComputeNode Name="172.16.7.105">
      <InstallDir>/home/users/andresb/opt/COMPSs/</InstallDir>
      <WorkingDir>/tmp/COMPSsWorker/</WorkingDir>
      <User>andresb</User>
      <LimitOfTasks>1</LimitOfTasks>
      <Application>
         <AppDir>/home/users/andresb/datma/codes/</AppDir>
         <Pythonpath>/home/users/andresb/datma/codes/</Pythonpath>
      </Application>
   </ComputeNode>
   <ComputeNode Name="172.16.7.104">
      <InstallDir>/home/users/andresb/opt/COMPSs/</InstallDir>
      <WorkingDir>/tmp/COMPSsWorker/</WorkingDir>
      <User>andresb</User>
      <LimitOfTasks>1</LimitOfTasks>
       <Application>
         <AppDir>/home/users/andresb/datma/codes/</AppDir>
         <Pythonpath>/home/users/andresb/datma/codes/</Pythonpath>
      </Application>
   </ComputeNode>
</Project>
```

RUNNING:

1. Generate the 16S database index cat <install_path>/datma/16sDatabases/README
2. DATMA can be executed using the runcompss script of COMPSs.

```
$ runcompss --project=project_solo1.xml --resources=resources_solo1.xml --summary -d --lang=python
/datma/codes/datma.py -f configurationFile.txt

$python /datma/codes/finalReport.py -f configurationFile.txt
```

3. To easy the execution, DATMA provides a run scripts to execute the complete workflow. Just type the run script name and specify the DATMA configuration file.

```
$ runDATMA.sh configurationFile.txt compss
```

RESULTS:

DATMA generates the follow directories.

- 16sSeq: 16S rna Ribosomal sequences detected.
- bins: All bin generated by CLAME.
- clean: Quality filter results.
- readsForbin.fastq: Balance reads that were not binned.
- round_*_b*: Report for the bin stages.
- *.html: Report files in HTML format.

AUTHORS:

Benavides A(1), Sanchez F (2), Alzate JF (3),(4) and Cabarcas F (1),(3)

Grupo Sistemic, Departamento de Ingeniería Electrónica, Facultad de Ingenieria, Universidad de Antioquia.

Smart Variable S.L, Barcelona, Spain

Centro Nacional de Secuenciacion Genomica-CNSG, Sede de Investigación Universitaria SIU, Universidad de Antioquia

Grupo de Parasitología, Departamento de Microbiología y Parasitología, Facultad de Medicina, Universidad de Antioquia

FAQ:

Please contact us:

bernardo.benavides@udea.edu.co

felipe.cabarcas@udea.edu.co