The `bam-utils` Software Package

Daniel Ortiz

January 15, 2019

Table of Contents

Introduction

Introduction

- Executing `bam` file pipelines is a tedious task
  - Downloading of very large files
  - Combination of tools with different input requirements
  - Existence of dependencies between tools
  - Tools may need to be added or removed
  - Each tool has specific computational requirements
  - Pipeline may need to be executed for hundreds of files
  - Parallelism should be exploited when possible
  - ...
- `bam-utils` has been created as a highly portable, configurable and extensible solution

Package Overview

Package Dependencies

- Shell Bash
- Python
- Conda
- Database download clients
- Slurm Workload Manager (optional)

Package Installation

- Obtain the package using git:

```
git clone gitlab@fsupeksvr.irbbarcelona.pcb.ub.es:dortiz/bam-utils.git
```

- Change to the directory with the package's source code and type:

```
./reconf
./configure
make
make install
```

NOTE: use `--prefix` option of `configure` to install the package in a custom directory

Additional `configure` Options

- `--with-icgcstor=DIR`: sets location of ICGC's storage client
- `--with-aspera=DIR`: enables Aspera Connect download client
- `--with-egadecrypt=DIR`: location of EGA decryptor tool

Functionality

- Automate execution of general pipelines
- Built-in support for pipelines processing normal-tumor `bam` files
- Automate processing of all of the samples of a dataset
- Handle file downloading as part of pipeline execution
- Keep track of which analysis steps of a pipeline for a pair of `bam` files have been completely executed and which haven't
- Specification of computational resources for each step

Execution Model

- `bam-utils` follows a simple execution model based on a file containing a list of analysis steps to be executed
- All of the steps defined in the list are by default executed simultaneously unless dependencies between them are specified
- Computational resources can be explicitly allocated if a workload manager is available

Supported Databases and Download Clients

- **Databases**
  - EGA
  - ICGC
- **Download clients**
  - `aspc`
  - `icgc-storage-client`
    - Amazon cloud
    - Collaboratory cloud
  - `pyega3`

Implemented Analysis Steps

- `bam` **file downloading**:
  - `download_ega_{norm|tum}_bam`
  - `download_ega_asp_{norm|tum}_bam`
  - `download_aws_{norm|tum}_bam`
  - `download_collab_{norm|tum}_bam`
- `bam` **file manipulation**:
  - `sort_{norm|tum}_bam`
  - `index_{norm|tum}_bam`
  - `delete_bam_files`

Implemented Analysis Steps

- **Small Indels and Single Nucleotide Variant Callers**:
  - `manta_somatic`
  - `platypus_germline`
  - `strelka_somatic`
- **Copy Number Variant Callers**:
  - `ascatngs`
  - `cnvkit`
  - `facets`
  - `wisecondorx`
- **MSI Analyzers**:
  - `msisensor`

Main Tools and File Formats

Main Tools

- `exec_pipeline`
- `query_ega_metadata`
- `query_icgc_metadata`
- `analyze_dataset`

exec_pipeline

- Automates execution of general pipelines
- Main input parameters:
  - `-a <string>`: file with analysis steps to be performed
  - `-o <string>`: output directory
  - `--showopts`: show pipeline options
  - `--debug`: check pipeline options but skip execution

exec_pipeline

- Content of output directory:
  - `scripts`: directory containing the scripts used for each analysis step
  - `<analysis_step_name>`: directory containing the results of the analysis step of the same name
- Additional directories may be created depending on the pipeline
  - `data`: directory containing the normal-tumor `bam` files

query_ega_metadata

- Extracts information from EGA metadata
- Main input parameters:
  - `-s <string>`: file with sample information
  - `-a <string>`: file with analysis information
  - `-t <string>`: file with study information
  - `-p <string>`: file listing Aspera box content
  - `-f <int>`: output format

query_icgc_metadata

- Extracts information from ICGC metadata
- Main input parameters:
  - `-d <string>`: file with donor information
  - `-a <string>`: file with aws manifest
  - `-t <string>`: table file in json format
  - `-f <int>`: output format:

analyze_dataset

- Uses metadata information to automate analysis of a whole dataset
- Main input parameters:
  - `-r <string>`: file with reference genome
  - `-m <string>`: file with metadata, one entry per line
  - `-a <string>`: file with analysis steps to be performed
  - `-p`: only print the commands executing the analysis

The `bam_utils_lib.sh` Library

- Shell library containing functions used by the previously described tools
- Functions can be classified as follows:
  - Implementation of the package execution model
  - Automated creation of scripts executing analysis steps
  - Helper functions to implement analysis steps

Additional Tools

- Reference genome operations:
  - `filter_contig_from_genref`
  - `gen_bed_for_genome`
- Data preparation for analysis steps:
  - `convert_snppos_to_snpgcc`
  - `create_snv_pos_ascat`
  - `gen_wisecondorx_ref`
- Reporting tools:
  - `get_analysis_status`

File Formats

- **Analysis file**: file describing all of the analysis steps to be carried out when processing a normal-tumor sample
- **Module file**: file defining the code of the analysis steps

File Formats

- **EGA/ICGC metadata**: information regarding a whole dataset that is typically spread out in a set of files
- **Analysis metadata**: file providing all the information of a given dataset that is relevant to automate its analysis
- **Analysis automation script**: file with a sequence of commands automating the analysis of a dataset

Analysis File

- **Module import** (module names separated by commas)
- **Entry format** (one entry per line)

  Step name, Slurm account, Slurm partition, CPUs, Memory limit, Time limit, Dependencies
- **Dependency types:** `none`, `after`, `afterok`, `afternotok`, `afterany`

```
#import bam_analysis
#
download_ega_norm_bam dortiz normal_prio 1 2048 10:00:00 jobdeps=none
download_ega_tum_bam dortiz normal_prio 1 2048 10:00:00 jobdeps=none
index_norm_bam dortiz normal_prio 1 1G 4:00:00 jobdeps=afterok:download_ega_norm_bam
index_tum_bam dortiz normal_prio 1 1G 4:00:00 jobdeps=afterok:download_ega_tum_bam
manta_somatic dortiz normal_prio 8 3G 6:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam
strelka_somatic dortiz normal_prio 8 6G 6:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam,
      afterok:manta_somatic
delete_bam_files dortiz normal_prio 1 1G 0:10:00 jobdeps=afterok:manta_somatic,afterok:strelka_somatic,
      afterok:msisensor,afterok:cnvkit,afterok:facets,afterok:ascatngs,afterok:platypus_germline
```

Module File

- Contains the definition of the different steps
- Written in `bash`
- Three `bash` functions should be defined for each step:
  - `stepname_explain_cmdline_opts()`
  - `stepname_define_opts()`
  - `stepname()`

Module File: `stepname_explain_cmdline_opts()`

- This function documents the command line options that the step needs to work
- The aggregated documentation for the different steps is used when executing `exec_pipeline --showopts`
- Whenever two steps share the same option, it is important to give it the same name
- Each option is documented using the `explain_cmdline_opt` function

Module File: `stepname_explain_cmdline_opts()`

```
manta_germline_explain_cmdline_opts()
{
    # -r option
    description="Reference genome file (required)"
    explain_cmdline_opt "-r" "<string>" "$description"

    # -n option
    description="Normal bam file (required if no downloading steps have been defined)"
    explain_cmdline_opt "-n" "<string>" "$description"
}
```

Module File: `stepname_define_opts()`

- This function should create a string containing the options that are specific to the step
- Such options may include:
  - command line options passed when executing `exec_pipeline`
  - options related to computational resources extracted from the corresponding entry of the analysis file
  - completely new options required by the step (typically, those are required to communicate different steps)
- The package provides multiple built-in functions to make the implementation of this function easier

Module File: `stepname_define_opts()`

- The function by default gets the `exec_pipeline` command line string and the corresponding entry of the analysis file
- Using these two parameters as well as other information handled in the module, the function should incrementally add the options
- Once the options are stored in the string, they should be saved by means of the `save_opt_list` function

Module File: stepname_define_opts()

```
stepname_define_opts()
{
    # Initialize variables
    local cmdline=$1
    local jobspec=$2
    optlist=""

    # Use built-in functions to add options to optlist variable
    ...

    # Save option list
    save_opt_list optlist
}
```

Module File: stepname_define_opts()

```
manta_germline_define_opts()
{
    # Initialize variables
    local cmdline=$1
    local jobspec=$2
    optlist=""

    # Define the -step-outd option, the output directory for the step,
    # which will have the same name of the step
    define_default_step_outd_opt "$cmdline" "$jobspec" optlist || exit 1

    # -r option
    define_cmdline_infile_opt "$cmdline" "-r" optlist || exit 1

    # -normalbam option
    local normalbam
    normalbam=`get_normal_bam_filename "$cmdline"` || exit 1
    define_opt "-normalbam" $normalbam optlist || exit 1

    # -cpus option
    local cpus
    cpus=`extract_cpus_from_jobspec "$jobspec"` || exit 1
    define_opt "-cpus" $cpus optlist

    # Save option list
    save_opt_list optlist
}
```

Module File: stepname_define_opts()

```
get_normal_bam_filename()
{
    local cmdline=$1
    local given=0
    local normalbam
    normalbam=`read_opt_value_from_line "$cmdline" "-n"` && given=1
    if [ $given -eq 1 ]; then
        # -n option was given
        file_exists $normalbam || { errmsg "file $normalbam does not exist" ; return 1; }
        echo $normalbam
    else
        # Check -extn option
        check_opt_given "$cmdline" "-extn" || { errmsg "-n or -extn option should be given" ; return 1;
              }
        local bamdir_fullname
        bamdir_fullname=`get_default_shdirname "${cmdline}" "-bamdir"` || { errmsg "-bamdir option not
              given" ; return 1; }
        normalbam=${bamdir_fullname}/normal.bam
        echo $normalbam
    fi
}
```

Module File: `stepname()`

- Implements the step
- The function should incorporate code at the beginning to read the options defined by `stepname_define_opts()`

Module File: stepname()

```
manta_germline()
{
    # Initialize variables
    local ref=`read_opt_value_from_line "$*" "-r"`
    local step_outd=`read_opt_value_from_line "$*" "-step-outd"`
    local normalbam=`read_opt_value_from_line "$*" "-normalbam"`
    local cpus=`read_opt_value_from_line "$*" "-cpus"`

    # Activate conda environment
    logmsg "* Activating conda environment..."
    conda activate manta 2>&1 || exit 1

    # Configure Manta
    logmsg "* Executing configManta.py..."
    configManta.py --bam ${normalbam} --referenceFasta ${ref} ${call_reg_opt} --runDir ${step_outd}
          2>&1 || exit 1
    # Execute Manta
    logmsg "* Executing runWorkflow.py..."
    ${step_outd}/runWorkflow.py -m local -j ${cpus} 2>&1 || exit 1

    # Deactivate conda environment
    logmsg "* Deactivating conda environment..."
    conda deactivate 2>&1
}
```

EGA Metadata

- Sample information (`Sample_File.map`)
  - contains file name info
- Analysis information (`Analisys_Sample_meta_info.map`)
  - contains donor and phenotype information
- Study information (`Study_analysis_sample.map`)
  - contains EGA id information
- Aspera box content (`dbox_content`)

ICGC Metadata

- Donor information (`donor.<study_name>.tsv`)
  - contains gender information
- AWS manifest (`manifest.aws-virginia.<code>.tsv`)
  - contains object id, file name and donor id information
- JSON table file (`icgc_table.json`)
  - contains phenotype information

Analysis Metadata (EGA)

- Created with the query_ega_metadata tool
- Example entries:

```
EGAF00001664282 phenotype=Blood|Normal_blood gender=male ; EGAF00001664327 phenotype=Skin|
      Tumour_metastasis_to_local_lymph_node gender=male

EGAF00001670586 phenotype=Blood|Normal_blood gender=male ; EGAF00001664289 phenotype=Skin|
      Tumour_metastasis_to_local_lymph_node gender=male

EGAF00001664356 phenotype=Skin|Tumour_metastasis_to_distant_location gender=male ; EGAF00001670533
      phenotype=Blood|Normal_blood gender=male

EGAF00001661882 phenotype=Blood|Normal_blood gender=male ; EGAF00001661538 phenotype=Skin|
      Tumour_metastasis_to_local_lymph_node gender=male
...
```

Analysis Metadata (EGA Aspera)

- Created with the query_ega_metadata tool
- Example entries:

```
EGAD00001003388/PART_2/EGAZ00001300436_20170516_AWS_MELA_3c3ed66c-1505-4614-ac9d-575a6713b06a.bam.crypt
       phenotype=Blood|Normal_blood gender=male ; EGAD00001003388/PART_3/
      EGAZ00001300354_20170516_AWS_MELA_daf1ffd8-0a0f-4869-abc8-5be0b4fc1a21.bam.crypt phenotype=Skin|
      Tumour_metastasis_to_local_lymph_node gender=male

EGAD00001003388/PART_3/EGAZ00001303407_20170516_AWS_MELA_a197619e-f3e2-41f6-aef7-d1fadf3c1f5b.bam.crypt
       phenotype=Blood|Normal_blood gender=male ; EGAD00001003388/PART_2/
      EGAZ00001300389_20170516_AWS_MELA_3a9bf676-1a7b-4718-8396-fb36cc89b688.bam.crypt phenotype=Skin|
      Tumour_metastasis_to_local_lymph_node gender=male

EGAD00001003388/PART_3/EGAZ00001300416_20170516_AWS_MELA_f64eba46-d8a1-46f2-ba66-1b509e16c946.bam.crypt
       phenotype=Skin|Tumour_metastasis_to_distant_location gender=male ; EGAD00001003388/PART_3/
      EGAZ00001303394_20170516_AWS_MELA_7bb66858-7533-4f96-9cd4-41aae2fe18b2.bam.crypt phenotype=Blood|
      Normal_blood gender=male
...
```

Analysis Metadata (ICGC)

- Created with the `query_icgc_metadata` tool
- Example entries:

```
34fa2369-424f-5886-9d23-6d19f8f15278 tumor female ; d759d07f-330c-5d0c-bd28-af72147dfb17 normal female

284f1424-d250-59cf-b105-da277b061e4a normal female ; e7e69d23-fb0d-5d3d-9027-ebf355053dbf tumor female

c42fffad-4ffd-59ba-93f1-2c573547369c normal female ; 3a33ef20-dfd0-50b0-afc2-38de9a5baa32 tumor female

37f076d6-fa64-5b5d-a0d0-b5cd7428d4a2 normal female ; 2c34270b-98d2-54b9-bdd3-068c6a9d858f tumor female
...
```

Analysis Automation Script

- Created with the `analyze_dataset` tool (`-p` option)
- At each entry (one per line), `exec_pipeline` tool is used to analyze a normal-tumor bam file pair
- Entry example:

```
/home/dortiz/bio/software/bam-utils/bin/exec_pipeline -r /home/dortiz/bio/data/genome_references/
    refseq_hg19_filt.fa -extn d759d07f-330c-5d0c-bd28-af72147dfb17 -extt 34fa2369-424f-5886-9d23-6
    d19f8f15278 -a /home/dortiz/bio/software/bam-utils/share/bam-utils/examples/basic_analysis_test.
    csv -g XX -o /mnt/raid/dortiz/bio/tasks/bam_analysis_testing_pipeline/d759d07f-330c-5d0c-bd28-
    af72147dfb17_34fa2369-424f-5886-9d23-6d19f8f15278 -cr /home/dortiz/bio/data/genome_references/
    refseq_hg19_filt.fa.bed -sv /home/dortiz/bio/data/facets_info/00-common_all.vcf -sg /home/dortiz/
    bio/data/ascatngs_info/r93/SnpGcCorrections_GRCh37_1000g.tsv -mc chrY -egastr 50 -egacred /home/
    dortiz/bio/software/ega-download-client-python/dortiz_cred.json
```

Extending Package Functionality

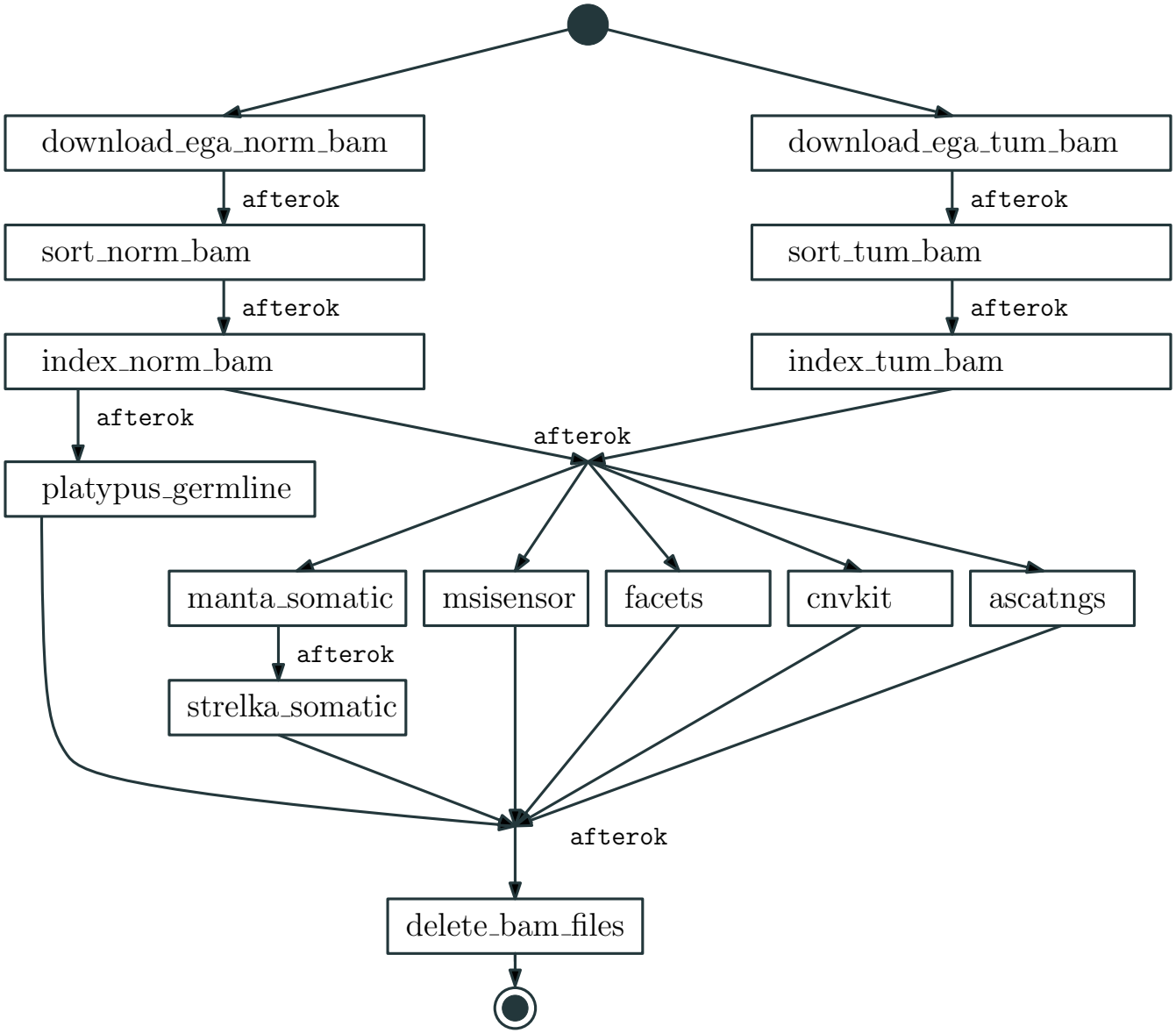
- The package can be easily extended by defining new modules
- After defining a module, it should be imported into the analysis file
- Since multiple imports are permitted, a new module may contain step definitions missing in another one
- The order in which modules are imported is relevant
  - if two modules define the same function, the definition in the module imported last will prevail
  - the previous property can be used to modify a specific step without repeating the code of the whole module

Whole Pipeline Example

Analysis File

```
#import bam_analysis
#
download_ega_norm_bam dortiz normal_prio 1 2048 10:00:00 jobdeps=none
download_ega_tum_bam dortiz normal_prio 1 2048 10:00:00 jobdeps=none
sort_norm_bam dortiz normal_prio 1 4G 10:00:00 jobdeps=afterok:download_ega_norm_bam
sort_tum_bam dortiz normal_prio 1 4G 10:00:00 jobdeps=afterok:download_ega_tum_bam
index_norm_bam dortiz normal_prio 1 1G 4:00:00 jobdeps=afterok:sort_norm_bam
index_tum_bam dortiz normal_prio 1 1G 4:00:00 jobdeps=afterok:sort_tum_bam
manta_somatic dortiz normal_prio 8 3G 6:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam
strelka_somatic dortiz normal_prio 8 6G 6:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam,
      afterok:manta_somatic
msisensor dortiz normal_prio 8 6G 5:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam
facets dortiz normal_prio 1 20G 4:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam
cnvkit dortiz normal_prio 8 8G 10:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam
ascatngs dortiz normal_prio 8 25G 12:00:00 jobdeps=afterok:index_norm_bam,afterok:index_tum_bam
platypus_germline dortiz normal_prio 1 4G 5:00:00 jobdeps=afterok:index_norm_bam
delete_bam_files dortiz normal_prio 1 1G 0:10:00 jobdeps=afterok:manta_somatic,afterok:strelka_somatic,
      afterok:msisensor,afterok:cnvkit,afterok:facets,afterok:ascatngs,afterok:platypus_germline
```

Pipeline


download_ega_norm_bam
download_ega_tum_bam
afterok
afterok
sort_norm_bam
sort_tum_bam
afterok
afterok
index_norm_bam
index_tum_bam
afterok
afterok
platypus_germline
manta_somatic
msisensor
facets
cnvkit
ascatngs
afterok
strelka_somatic
afterok
delete_bam_files