Drug-seq UMI count Workflow on HPC

Overview

Workflow enables automated umi count processing of the drugseq data using HPC cluster resources.

It consists of 2 main steps :

- aligner_task.sh - This merges umi read information as part of transcript read description and triggers STAR alignment.
- bamchunkprocess_task.sh- Splits STAR bam file based on individual chromosomes, filters bam files to retain only primary alignments and triggers parallel task for each chromosome for umi count processing.

User provides the 'SampleSheet.csv' (example: SampleSheet.csv) containing sample/experiment metadata information, along with other input/output command line options and workflow steps are submitted to the cluster for parallel data processing.

Steps run in the background:

- Accept command line arguments and executes sanity checks.
- Creates run set-up and output directory structure.
- Prepares for sample wise parallel execution of fastq creation and counts steps, and aggregate group wise parallel execution of aggregation and normalization steps.
- Creates cluster submission scripts for automated or manual job submission.

Setup

Source the lines below in your .bashrc file to set-up the test environment.

.bashrc content

```
# .bashrc

. /etc/profile

#"load module environment"

export MODULEPATH=${MODULEPATH}:/usr/prog/modules/all:/cm/shared/modulefiles

#add umi script to env path

UMI_HOME=/usr/prog/scicomp/drug-seq

export PATH=$PATH:$UMI_HOME

#load cluster queue environment

module load uge

#needs python2, load python 2.7.9

module load python/2.7.9-goolf-1.5.14-NX

#load samtools

module load samtools

#load bamtools

module load bamtools

#load "pysam-0.11.2.2" or higher (ns python virtualenv already has necessary version installed)

#source virtualenv
```

```
source /usr/prog/ns/python_env/bin/activate
```

sourcing the environment

source .bashrc

To check to see if your environment is correct, please issue the following commands

Check your environment

```
> which run_umi_workflow.sh [ should display /usr/prog/scicomp/drug-seq/run_umi_workflow.sh ]

>which bamtools [should display /usr/prog/bamtools/2.4.1-goolf-1.5.14-NX/bin/bamtools]

> which samtools [ should display /usr/prog/samtools/1.4.1-goolf-1.5.14-NX/bin/samtools ]

> which python  [ should display /usr/prog/ns/python_env/bin/python]

> python -c "import pysam; print (pysam.__version__)" [should display 0.11.2.2]
```

Run umi count workflow

run_umi_workflow.sh - Script requires SampleSheet.csv along with other command line arguments to trigger scrna workflow steps on the cluster.

run_umi_workflow.sh

```
run_umi_workflow.sh --help
Usage: run_umi_workflow.sh
        --samplesheet  <comma separated file containing
[file_path,transcript_read_file,umi_read_file,reference_env_file],
required. Provide one such line for eachsample>
 --email <use your novartis emailid [required]>
        --outdir <output directory [default: current directory]>
        --scriptdir <scripts directory path, default:
/da/NS/tnt/prog/countumi>
        --clip5prime <number of bases to clip from five prime end (for star
aligner), [default: 5]>
        --clip3prime <number of bases to clip from three prime end (for
star aligner), [default: 1]>
        --umiruntime <numberofHours|unlimited> Provide estimated max umi
processing runtime in hours[default: 48 ]>
        --umimemory <total memory per job> [optional, default: 12],
considered in gigabytes
        --runmode <run|check> [run: submits job to the cluster, check:
creates all the sge template scripts, requires manual submission [default:
run]
```

SampleSheet.csv

Each line in the file represents one sample, requires four comma separated fields.

1) input file path where fastq files are. Example: /usr/prog/ns/ngs_workflow_templates/countumi

2) fastq name of the trascript read file. Example : drug_seq_384_R2.fastq

3) fastq name of the umi-bacode read file. Example: drug_seq_384_R1.fastq

4) Reference environemnt file containing star index, gene_pos.dat file information etc. Example: /usr/prog/ns/ngs_workflow_templates/countumi/star_ref_env.sh

Columns:

fastq_path	transcript_read_file	umi_read_file	reference_env_file
/usr/prog/ns/ngs_workflow_templates/countumi	drug_seq_384_R1.fastq	drug_seq_384_R1.fastq	/usr/prog/ns/ngs_workflo

Content of reference environment file

reference information

```
export REF_STAR=/da/NS/yech3/STAR_ref_hg38_mm10

export STAR_EXEC=/usr/prog/ns/STAR-2.5.1b/source/STAR

#basefiles

export REF_GTF=/da/NS/yech3/STAR_ref_hg38_mm10/hg38_mm10.gtf

export REF_GENE_POS=/da/NS/yech3/STAR_ref_hg38_mm10/hg38_mm10_gene_start_stop.dat #genome indices
```

Trigger the run

Click here to see how to trigger the run

-bash-4.1$ run_umi_workflow.sh --samplesheet /usr/prog/ns/ngs_workflow_templates/countumi/input.txt --email tripti.kulkarni@novartis.com --outdir /clscratch/kulkatr1/umiout --clip5prime 5 --clip3prime 0 --runmode run

```
[STATUS][24-06-17,12:33:28] Checking for arguments .....

[STATUS][24-06-17,12:33:28] SampleSheet found :
/usr/prog/ns/ngs_workflow_templates/countumi/samplesheet.csv

[STATUS][24-06-17,12:33:28] umi runtime provided: 48 hours [STATUS][24-06-17,12:33:28] Checking for
files and creating output structure   [STATUS][24-06-17,12:33:28] Outputdirectory:
/clscratch/kulkatr1/umiout created successfully

[STATUS][24-06-17,12:33:28] Samplesheet provided for the run :
/usr/prog/ns/ngs_workflow_templates/countumi/samplesheet.csv

[STATUS][24-06-17,12:33:28] Email id provided : tripti.kulkarni@novartis.com [STATUS][24-06-17,12:33:28]
Output directory : /clscratch/kulkatr1/umiout [STATUS][24-06-17,12:33:28] Script directory :
/usr/prog/scicomp/drug-seq

[STATUS][24-06-17,12:33:28] Run mode provided : run [STATUS][24-06-17,12:33:29] UMI process max run time
: 172800 seconds

[STATUS][24-06-17,12:33:29] Number of bases to clip from five prime end : 5

[STATUS][24-06-17,12:33:29] Number of bases to clip from three prime end : 0

[STATUS][24-06-17,12:33:29] Checking for scripts and executables.....

[STATUS][24-06-17,12:33:29] script base: /usr/prog/scicomp/drug-seq

[STATUS][24-06-17,12:33:29] Found all scripts! /usr/prog/samtools/0.1.19-goolf-1.5.14-NX/bin/samtools
/usr/prog/bamtools/2.4.1-goolf-1.5.14-NX/bin/bamtools

[STATUS][24-06-17,12:33:29] bamtools found!

[STATUS][24-06-17,12:33:29] samtools found!

[STATUS][24-06-17,12:33:29] Creating cluster submission scripts for drug_seq_384_R2.fastq
/clscratch/kulkatr1/umiout/drug_seq_384_R2
```

```
[STATUS][24-06-17,12:33:29] Cluster submission script:
/clscratch/kulkatr1/umiout/drug_seq_384_R2/clusterSubmission.sh

[STATUS][24-06-17,12:33:29] Submitting jobs to the cluster... STAR alignment job submitted, JOBID:
1893575

[STATUS][24-06-17,12:33:29] Done... /home/kulkatr1/projects/drugseq/countumi


-bash-4.1$ qstat job-ID     prior   name       user         state submit/start at     queue
jclass         slots ja-task-ID

-----------------------------------------------------------------------------------------------------------
---------------------

            1893575        0.50371 drug_seq_3 kulkatr1       r    06/24/2017 12:33:29         12



Note:  After alignment job is done, umi count processing parallel task is triggered. UMI job id will be
captured in process_logs directory for tracking purpose.
```

important "--runmode check"
To create the run set-up but not to trigger the run on the cluster

- `Use "--runmode check" in the command line.`
- `This allows for  any extra verification/changes and manual cluster submission.`
- `Go to run directory and edit the clusterSubmission.sh script and execute the script for cluster submission.`

Output files

Workflow creates output directory label from --outdir parameter provided.

For example:

In this previous run example, /clscratch/kulkatr1/umiout output directory will be created

Click here for Output structure

Main output directory will contain output directories for each sample

- `ls /clscratch/kulkatr1/umiout/`

  `drug_seq_384_R2`

for each sample output

- `ls /clscratch/kulkatr1/umiout/drug_seq_384_R2`

  `aligner_task.sh  bam_files   process_logs  sge_logs bamchunkprocess_task.sh  clusterSubmission.sh restart.sh   umi_counts`

```
bam_files - folder containing chromosome split bam files

umi_counts - folder containing umi count for bam files

process_logs - folder containing  detailed log information from each stage of the workflow/executables,
useful in checking the process

sge_logs - log files generated from cluster, useful in checking if job encountered errors in cluster,
memory issues, node failure etc
```

To check the samples/steps which were finished successfully

```
ls /clscratch/kulkatr1/umiout/drug_seq_384_R2/Successful_*
```

To check the samples/steps which encountered errors

```
ls /clscratch/kulkatr1/umiout/drug_seq_384_R2/Failed_*
```

Restart steps

In case a workflow has encountered errors in any of the steps, there will be "Failed*.txt" files in the output directory. These files capture information on sample and steps which failed.

To rerun, simply execute the restart.sh script with a command line option which specifies the step to be rerun and sample name. This will overwrite failed sample output.

```
bash-4.1$ ./restart.sh
Usage: restart.sh <align|count> <sample_name>

Example:
restart.sh align drug_seq_384_R2
or
restart.sh count drug_seq_384_R2
```