paleomix, Next-Generation Sequencing wrapper

this framework is open-source and available at GitHub and wrap all steps from `fastq` to `bam` files. Actually, this tool can do much more but the rest is out of scope. Its major drawback is that it is dedicated to one machine. For clusters, you are then limited to one node since memory are not shared by default.

check if paleomix is available

```
paleomix -h
```

test your install

fetch the example, reference is the human mitochondrial genome

```
mkdir -p ~/install/paleomix/example
cp -r /work/users/aginolhac/chip-seq/paleomix/examples/bam_pipeline/00* ~/install/paleomix/example
cd ~/install/paleomix/example
```

run the example, start by a `dry-run`, adjust the number of threads accordingly.

```
paleomix bam_pipeline run --bwa-max-threads=1 --max-threads=2 --dry-run 000_makefile.yaml
```

If all fine, re-rerun the command without the `--dry-run` option

Generate a makefile

Trimming, mapping imply a lot of steps and it is hard to be sure that everything goes well. Paleomix works in temporary folder, check the data produced and then copy back files that are complete. Plus, you want to test different parameters, add a new reference without having to redo earlier steps while being sure that all files are up-to-date. This goes through a `YAML` makefile. The syntax is pretty forward.

Create a generic makefile

```
cd ~/chip-seq
paleomix bam_pipeline mkfile > mouse.makefile
```

Edit the makefile

using your favorite editor, edit the `mouse.makefile`. For example `vim mouse.makefile` or `kate` or `nano`.

Options

For duplicates, change the default behaviour from `filter` to `mark`

```
  PCRDuplicates: mark
```

Features

Under the `Features` section, comment with a `#` the part that should be run to fit the following

```
Features:
  - Raw BAM         # Generate BAM from the raw libraries (no indel realignment)
                    #   Location: {Destination}/{Target}.{Genome}.bam
#    - Realigned BAM  # Generate indel-realigned BAM using the GATK Indel realigner
                    #   Location: {Destination}/{Target}.{Genome}.realigned.bam
#    - mapDamage      # Generate mapDamage plot for each (unrealigned) library
                    #   Location: {Destination}/{Target}.{Genome}.mapDamage/{Library}/
  - Coverage        # Generate coverage information for the raw BAM (wo/ indel realignment)
                    #   Location: {Destination}/{Target}.{Genome}.coverage
#    - Depths         # Generate histogram of number of sites with a given read-depth
                    #   Location: {Destination}/{Target}.{Genome}.depths
  - Summary         # Generate target summary (uses statistics from raw BAM)
                    #   Location: {Destination}/{Target}.summary
```

Prefixes

These are the references to align read to.

```
Prefixes:
  # Name of the prefix; is used as part of the output filenames
  mouse_19:
    # Path to .fasta file containg a set of reference sequences.
    Path: /work/users/aginolhac/chip-seq/references/chr19.fasta
```

Samples

enter at the end of the makefile, the following lines, according to your login. Do use **spaces** and not tabs for the indentation.

```
TC1-I-A-D3:
  TC1-I-A-D3:
    TC1-I-A-D3:
      "14s006680-1-1":
        /home/users/student01/chip-seq/raw/C53CYACXX_TC1-I-A-D3_14s006682-1-1_Sinkkonen_lane114s006682_seque

TC1-H3K4-A-D3:
  TC1-H3K4-A-D3:
    TC1-H3K4-A-D3:
      "14s006647-1-1":
        /home/users/student01/chip-seq/raw/C51C3ACXX_TC1-H3K4-A-D3_14s006647-1-1_Sinkkonen_lane514s006647_s

TC1-I-ST2-D0:
  TC1-I-ST2-D0:
    TC1-I-ST2-D0:
      "14s006677-1-1":
        /home/users/student01/chip-seq/raw/C51C3ACXX_TC1-I-ST2-D0_14s006677-1-1_Sinkkonen_lane814s006677_se

TC1-H3K4-ST2-D0:
  TC1-H3K4-ST2-D0:
    TC1-H3K4-ST2-D0:
      "14s006644":
        /home/users/student01/chip-seq/raw/C51C3ACXX_TC1-H3K4-ST2-D0_14s006644-1-1_Sinkkonen_lane514s006644_
```

Perform the trimming / mapping

```
paleomix bam_pipeline run --bwa-max-threads=2 --max-threads=12 --dry-run mouse.makefile
```

check trimming

First of all, check using `fastqc` that the trimming did remove the adapters that were contaminated the reads.

```
find . -name "reads.truncated.bz2" | parallel "fastqc {}" &
```

using the character `&` tells the shell that we want the processes to run in the background. Meaning that you can still run more things while the 4 tasks are running. Check them using `htop`.

check especially, the input for ST2, day0 before and after trimming. Did it solve the issue with adapters?

filter for unique reads

Uniqueness of reads refers to mappability. The less locations a read has in a genome, the higher is mappability will be. A common filter is to use **30** as a threshold for filtering reads:

```
samtools view -b -q 30 file.bam > file.q30.bam
```

Filter in parallel

```
parallel "samtools view -b -q 30 {} > {.}.q30.bam" ::: *.bam
```

Since we are using only the chr19 for this tutorial, do you think the mappability score is correct? Why?

filter for duplicates?

A duplicate is a bias that comes from PCR amplification. Reads then stack at the same location and create artificial high coverages. Duplicates have a unclear definition in a mapped file. Usually, single-end reads that are mapped at the same 5' end are considered as duplicates. External coordinate are used for paired-end reads.
For regular NGS, filtering for duplicates is mandatory. However, for chip-seq since the reads are by nature clustered location this is not recommended. If duplication is observed at the reads level, such as in `fastqc` output, then filtering may be necessary. Marking duplicates allows to keep track of them without losing them.

Previous Next