batchAssembly

vers 1.0.0.0

2016-07-27

author: Stephan Fuchs (fuchss@rki.de)

Content

1 Short Description

parallelized batch a5 assembly of PE read files

2 Man Page

```
usage: batchAssembly [-h] [-i] [-p INT] [-t INT] [--version] FILE [FILE
                     ...]

parallelized batch a5 assembly of paired-end read data

positional arguments:
  FILE        input fastq file(s)

optional arguments:
  -h, --help  show this help message and exit
  -i          if set paired-end read data is provided in a single file
              (interleaved FASTQ). By default, two files containing forward
              and reverse reads, respectively, are expected.
  -p INT      Number of subprocesses that should be run in parallel.
              Default is 2.
  -t INT      Number of threads/CPUs that should be assigned to each
              subprocess. Default is 2.
  --version   show program's version number and exit  --version
show program's version number and exit
```

3 Function and Input

batchAssembly provides de novo assembling of multiple paired-end read datasets in batch mode. Assembling is based on a5 assembler [1]. Jobs can be parallelized.

3.1 Defining the FASTQ files

The definition of at least one FASTQ file containing the read data is mandatory.
All FASTQ files are defined at the last command position by a space delimited list of file names. If the FASTQ files are not in the current working directory the relative or absolute path has to be included. batchAssembly accepts only paired-end read data. Paired-end read identifiers have to be conforming to the Illumina read pair convention. By default, read data has to be provided in two FASTQ files containing forward and reverse reads, respectively. These files reads have to be submitted in the order *forward file of dataset 1*, *reverse file of dataset 1*, *forward file of dataset 2*, *reverse file of dataset 2*,…
However, it is more easy to use wildcards (*) to submit the file list. In this case, mate files are automatically recognized by means of their names. In general, FASTQ files can be submitted as GZIP archive or plain text file. Example:

```
> batchAssembly *.fq.gz
```

Alternatively, paired-end read data can be provided in a single interleaved FASTQ file (see 3.2).

3.2 Providing interleaved FASTQ files using `-i`

Paired-end read data can be provided in a single interleaved FASTQ file using the option `-i`.

```
> batchAssembly -i *.fq.gz
```

3.3 Parallelization of assembly jobs using `-p`

Assembly jobs can be parallelized. Using the parameter -p followed by an integer number, the number of parallel jobs can be defined. By default, two assembly jobs are processed at the same time. Please make sure, that the number of parallel jobs multiplied by the number of CPUs assigned to each (see 3.4) is not higher than the number of CPUs in your machine. Example:

```
> batchAssembly -p 10 *.fq.gz
```

3.4 Usage of multiple CPU per process using `-t`

Speed of assembling can be increased by the number of CPUs provided to each assembly job (see 3.3). By default, two CPUs are assigned to each parallel job. Please make sure, that the number of parallel jobs multiplied by the number of CPUs assigned to each is not higher than the number of CPUs in your machine. Example:

```
> batchAssembly -p 10 -t 4 *.fq.gz
```

In this example 10 jobs are processed in parallel each using 4 CPUs. Thus the machine has to provide at least 40 CPUs.

4 Output

batchAssembly creates a sub-directory in the *current directory*. This result folder is named according to the pattern *BATCHASSEMBLY _YYYY-MM-DD_hh-mm-ss* and contains:

- *batchAssembly.log* providing detailed information on program versions and submitted files/jobs
- dataset-specific subfolders

4.1 *batchAssembly.log*

This file contains general information on

- used program versions
- successfully completed assembly jobs

4.2 dataset-specific subfolders

Each job generates a dataset-specific subfolder named after the related FASTQ file(s). These folders contain:

- original a5 [1] output

- *stderr.log*
- *stdout.log*

The *stderr.log* file contains the standard error output generated by the a5 assembler [1]. The *stdout.log* file contains the standard output generated by the a5 assembler [1]. In case of failures or unexpected results these files can be helpful and should be considered.

5 Usage Example

Using batchAssembly on three paired-end read datasets (Lib1, Lib2, Lib3) results in the following output (user input is highlighted in blue):

```
> ~/data/scripts/NGS/batchAssembly -t 5 *.fq.gz
target path created:  BATCHASSEMBLY_2016-08-04_07-29-08
preparing assembly job(s)...
working on 0 assembly job(s)...  [3 of 3 done] [3 successfully completed]
writing log file...
3 of 3 jobs successfully completed
>
```

Following folders and files are created:

- *BATCHASSEMBLY_2016-08-04_07-29-08*
  - *batchAssembly.log*
  - *Lib1 _R*
    - *stderr.log*
    - *stdout.log*
    - … a5 output folders and files
  - *Lib2 _R*
    - *stderr.log*
    - *stdout.log*
    - … a5 output folders and files
  - *Lib3_R*
    - *stderr.log*
    - *stdout.log*
    - … a5 output folders and files

6 Requirements

batchAssembly was tested using:

- Debian GNU/Linux 8 (jessie)
- A5-miseq version 20150522

7 License & Disclaimer

8 References

[1] Coil D, Jospin G, Darling AE. A5-miseq: an updated pipeline to assemble microbial genomes from Illumina MiSeq data. *Bioinformatics*. 2015 Feb 15;31(4):587-9. doi:10.1093/bioinformatics/btu661. Epub 2014 Oct 22. PubMed; PMID: 25338718.