Supplementary Data to: "SFQ: Constructing and querying a succinct representation of FASTQ files"

Robert Bakarić, Damir Korenčić, Dalibor Hršak and Strahil Ristov

1 SFQ usage and options

SFQ supports various options both in the processing/compression stage and in the reading stage. In the processing phase, SFQ accepts as an input FASTQ or FASTA files and produces a compressed sFASTQ archive. The type of compression can be lossless, or one of the four variants of lossy compression. In the compression stage, the user can determine the type of the input format, the type of compression and the amount of used RAM. The result of the compression is a directory with multiple subdirectories and files. If the processing is interrupted, and some of the files are already produced, the `--restart` option can be enabled to re-use the previously created files.

As an input in the reading stage, SFQ accepts sFASTQ directory, and, in the random access mode, a text file with a list of records' IDs. sFASTQ directory can be read from disk, or loaded into RAM. The output are streams of FASTQ or FASTA records printed to STDOUT or to an optional output file. The default output format is the standard FASTQ or FASTA organization of streams, but SFQ can optionally produce various combinations of the FASTQ record components. If the external list of record IDs is missing in the random access mode, the SFQ can, mainly for the test purposes, generate random IDs.

SFQ software is implemented as a combination of the command line interface coded in Rust and the LZ trie core coded in C++. It requires C++11 compiler, or higher. The command line options are listed below.

```
USAGE:
    sfq [OPTIONS]

FLAGS:
    -h, --help                          Prints help information
    -V, --version                       Prints version information

OPTIONS:
    -a, --action <c|d|g>                Action: (c) compress, (d) decompress, (g) grep <requires --list >  [default: c]
    -s, --compression-mode <0-4>        Compression mode [1-4 produce different lossy versions, default: 0]
    -F, --fragment-size <Max|integer>   Amount of RAM in MB allocated for the compression. Max = use all available RAM.
                                        [default: Max]
    -t, --infmt <fastq|fasta>           File types supported [default: fastq]
    -i, --input <FILE>                  Input file (fasta,fastq,sfastq)
    -j, --input-rev <FILE>              Filename of a reverse file (fastq, fasta)
    -l, --list <filename|rand(10)>      Please provide a list of prefixes (numbers or ranges), in separate lines. SFQ
                                        returns records associated with the input prefixes. Works only with -a g.
                                        [default: rand(10)]
    -m, --memory-mode <D|R>             Memory mode: defines memory type  (D - disk, R - RAM) [default: D]  [possible
                                        values: D, R]
    -f, --outfmt <fq|fa|s|q|h|...>      Output format:

                                          fq      :fastq,
                                          fa      :fasta,
                                          s   :sequence,
                                          q   :quality,
                                          h   :head,
                                          s+q     :sequence quality,
                                          h+q     :head quality,
                                          h+s     :head sequence,
```

```
                                        h+s+q   :head sequence quality,
                                        s+h+q   :sequence head quality,
                                        ...
                                      [default: fq]
    -o, --output <FILE>               Output file; interleaved if input is two paired end fastq files
    -r, --restart <no|yes>            Restart compression from temporary files. Works only with -a c. NOTE: Temporary
                                      files must be complete and correct! [default: no]
```

1.1 Hidden -e option

When SFQ reads a contiguous block of records, the IDs are forwarded to LZ trie core in batches of 4^N consecutive numbers, which leads to a better cache efficiency. We have empirically established that for the most datasets the fastest average response is reached when $N = 6$, which we have hard-coded as a default value. This leads to a somewhat staccatoed output, as a result of the time needed for the successive initializations of 4^6 values. The output can be smoothed using the hidden "-e" option, with the synopsis "-e N", to set N to a smaller value. The output to STDOUT will look nicer, but the reading speed will be somewhat impaired.

The RAM usage is also dependent on the size of N. Therefore, the smallest memory footprint in the reading stage is achieved with $N = 0$.

1.2 SFQ usage examples

```
Example No.1 - Compress single stranded fastq file

          sfq -i ./data/fwd.fq -a c -t fastq -o FwdIdx

Example No.2 - Compress paired-end fastq file

          sfq -i ./data/fwd.fq -j ./data/rev.fq -a c -t fastq -o FwdRevIdx

Example No.3 - Decompress single stranded fastq files by printing full records

          sfq -i FwdIdx -a d -f fq -t fastq -o fw.fq

Example No.4 - Decompress paired-end fastq files by printing full records

          sfq -i FwdRevIdx -a d -f fq -t fastq -o fw_rv.fq

Example No.5 - Decompress single stranded fastq files by printing fasta records

          sfq -i FwdIdx -a d -f fa -t fastq -o fw.fa

Example No.6 - Decompress paired-end fastq files by printing fasta records

          sfq -i FwdRevIdx -a d -f fa -t fastq -o fw-re.fa

Example No.7 - Decompress single stranded fastq files by printing tsv formatted: head \tab seq

          sfq -i FwdIdx -a d -f "h+s" -t fastq -o fw_hs.tsv

Example No.8 - Decompress paired-end fastq files by printing tsv formatted: quality \tab head

          sfq -i FwdRevIdx -a d -f "q+h" -t fastq -o fw_qh.tsv

Example No.9 - Extract a specific set of records listed in list.file as sequence only format

          sfq -i FwdRevIdx -a g -f "s" -t fastq -o s.out -l list.file

Example No.10 - Compress paired-end fasta file while limiting available RAM to 8 GB

          sfq -i ./data.in/fa.fa -a c -t fasta -o fa.out -F 8000

Example No.11 - Extract a random set of 4 records in sequence + head format

          sfq -i ./FwdRevIdx -a g -t fastq -o fa.out -l "rand(4)" -f "s+h"
```

2 Datasets

The experimental datasets are a selection of datasets from [1], [2] and [3]. They are available at the following links.

P.aeruginosa

```
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR554/SRR554369/SRR554369_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR554/SRR554369/SRR554369_2.fastq.gz
```

Metagenomic

```
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR532/ERR532393/ERR532393_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR532/ERR532393/ERR532393_2.fastq.gz
```

H.sapiens1 (ERP001775)

```
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR174/ERR174324/ERR174324_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR174/ERR174325/ERR174325_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR174/ERR174324/ERR174324_2.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR174/ERR174325/ERR174325_2.fastq.gz
```

The first two files were concatenated to obtain FASTQ file with forward reads, and the second two files are concatenated to obtain the FASTQ file with paired reverse reads.

H.sapiens2 (NA12878 Rep 1, Lane 1)

```
NA12878-Rep-1_S1_L001_R1_001.fastq
NA12878-Rep-1_S1_L001_R2_001.fastq
```

Downloaded from Illumina's BaseSpace public data (https://basespace.illumina.com/datacentral) from the project NovaSeq S2: Nextera DNA Flex (8 replicates of NA12878).

ERR194146

```
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR194/ERR194146/ERR194146_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR194/ERR194146/ERR194146_2.fastq.gz
```

ERR174310

```
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR174/ERR174310/ERR174310_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR174/ERR174310/ERR174310_2.fastq.gz
```

SRR065390

```
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR065/SRR065390/SRR065390_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR065/SRR065390/SRR065390_2.fastq.gz
```

SRR689233

ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR689/SRR689233/SRR689233_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR689/SRR689233/SRR689233_2.fastq.gz

SRR635193

ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR635/SRR635193/SRR635193_1.fastq.gz
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR635/SRR635193/SRR635193_2.fastq.gz

MiSeq

ftp://webdata:webdata@ussd-ftp.illumina.com/Data/SequencingRuns/MG1655/MiSeq_Ecoli_MG1655_110721_PF_R1.fastq.gz
ftp://webdata:webdata@ussd-ftp.illumina.com/Data/SequencingRuns/MG1655/MiSeq_Ecoli_MG1655_110721_PF_R2.fastq.gz

SRR1536586

Downloaded with sra-tools (`https://github.com/ncbi/sra-tools`) "fastq-dump SRR1536586" command.

References

[1] S. Chandak, K. Tatwawadi, I. Ochoa, M. Hernaez, T. Weissman, SPRING: a next-generation compressor for FASTQ data, Bioinformatics 35 (15) (2018) 2674–2676.

[2] T. M. Kowalski, S. Grabowski, PgRC: pseudogenome-based read compressor, Bioinformatics 36 (7) (2019) 2082–2089.

[3] X. Xia, Arsda: A new approach for storing, transmitting and analyzing transcriptomic data, G3: Genes, Genomes, Genetics 7 (12) (2017) 3839–3848.