GMcloser User Manual

Ver. 1.6.2 March 1, 2018

Contents

A. GMcloser

1. About GMcloser

GMcloser fills and closes the gaps present in scaffold assemblies, especially those generated by the *de novo* assembly of whole genomes with next-generation sequencing (NGS) reads. Unlike other gap-closing tools that use only NGS reads, GMcloser uses preassembled contig sets or long read sets (e.g., error-corrected PacBio reads) as the sequences to close gaps and uses paired-end (PE) reads to improve the accuracy and efficiency of gap closure. The efficiency of gap closure can be increased by successive treatments with different contig sets. To obtain accurate gap-closed assemblies with this strategy, a contig set containing fewer errors should be used in the first step. Alternatively, when using a long read set (must be specified with the long-read option, -lr), iterative gap-closing processes can be automatically performed by specifying the --iterate option. GMcloser assumes that the coverage of a long read set is ≥ 2x and that of a contig query set is < 2x. When using a contig set with ≥ 2x coverage, the job may be run in the long-read mode. GMcloser is implemented in the following five steps:

(1) Alignment of a contig (or long read) set to subcontigs in the scaffold.

The scaffold is split into subcontigs (i.e., contigs contained in scaffolds) and aligned with another contig or long read set using the Nucmer aligner. Alignment data (i.e., coords files) with Nucmer are obtained with the following commands:

```
nucmer -maxmatch -banded -D 5 -l <min_match>
delta-filter -o 95 -i <min_identity>
show-coords -THrcl
```

where <min_match> is 30, 40, or 50, depending on assembly size, and <min_identity> is set with the value specified with the -mi option.

This alignment can be also conducted with another aligner BLASTn by setting an option --blast, which is faster when aligning assemblies of large genomes. When using a long read set, the BLAST aligner is automatically selected.

(2) Alignment of PE reads to both the contig (long read) and subcontig sets.

The first and second sets of PE reads, with inserts of 300–800 bp and with 20–100x coverage, are aligned separately to both the contig (long read) and subcontig sets with the Bowtie2 aligner in the single-end mode.

(3) Likelihood-based selection of the correct contig–subcontig alignment pairs using contig- and read-alignment statistics.

To judge whether a contig–subcontig alignment is correct, a likelihood-based estimate is made using predetermined likelihood ratios of true test alignments (see our paper for details). This estimate is made by calculating a score for each alignment, based on the likelihood ratios for the alignment overlap length, alignment overlap identity, and PE-read mapping rate.

(4) Filling and closing the gaps with selected contigs.

Using the selected contig (long read)–subcontig alignment data, the gaps present in the scaffold dataset are filled and the scaffold termini are extended. For filled but not completely closed gaps, the gaps are closed according to the result for the pairwise alignment of the two subcontigs that encompass the gap. This pairwise alignment is performed with the YASS aligner.

(5) Connection of the subcontig pairs that encompass a gap.

We have observed that a number of subcontig pairs that encompass gaps overlap between the terminal regions of the pair. Connection of the neighboring subcontig pairs can be attempted as a selectable option. The pairwise alignment of neighboring subcontig pairs is performed with YASS.

Although GMcloser can close gaps in the absence of PE reads, the efficiency and accuracy of gap closure may be considerably decreased. In the absence of PE reads, potentially correct contig–subcontig alignments are selected using only empirical settings (i.e., values specified with the --min_match_len and --min_identity options) for the alignment overlap length and overlap identity.

[Important points for running time]

The entire process can be accomplished within a few hours to half a day for genomes of ~100 Mb, but may take several days for large genomes or for a long read set with a high coverage. When closing the gaps in assemblies from a large genome, we recommend using a large value (e.g., >15) for the --thread option. For larger genomes (of >600 Mb), the alignment process with Nucmer may take a considerably longer time or fail to finish the job. Thus, for larger genomes you may enable to shorten runtime and reduce memory consumption by using BLAST instead of Nucmer (by setting the --blast option).

2. Prerequisites

GMcloser can be conducted on the standard Linux and Mac OSX machines.

(1) Perl 5.6 or later

All components of GMcloser are written in Perl. If your Perl is not in the standard location (/usr/bin/perl), edit the first line of all the perl scripts or create a link of your perl executable to /usr/bin.

(2) MUMmer 3.23 (required command: nucmer, delta-filter, show-coords)
MUMmer 3.22 is not allowed

MUMmer/Nucmer (http://mummer.sourceforge.net) is required for the alignment of a query contig set to a target subcontig set.

(3) blast+ 2.2.18 or later (makeblastdb, blastn)

blast+ (ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+) is optional when the alignment of a query contig set to a target subcontig set is performed with BLASTn.

(4) Bowtie 2 (bowtie2-build, bowtie2) (tested on ver. 2.1.0)

Bowtie 2 (http://bowtie-bio.sourceforge.net/bowtie2/index.shtml) is required for the alignment of PE reads to a contig set and a subcontig set.

(5) YASS (yass) (tested on ver. 1.14)

YASS (http://bioinfo.lifl.fr/yass/) is required for the pairwise alignment of subcontigs and contigs or subcontigs. An executable command for downloaded binaries may differ from 'yass'. If so, the executable command should be linked to 'yass' to allow the implementation with a 'yass' command.

The PATH environmental variable for the commands of all the tools cited above must be set to your system.

<For tcsh or csh shell>
Add the following lines in your .cshrc.

```
setenv PATH {$PATH}:path_to_MUMmer3.23
(e.g., setenv PATH {$PATH}:/home/myname/tools/MUMmer3.23)
setenv PATH {$PATH}:path_to_ncbi-blast-2.2.??+/bin
setenv PATH {$PATH}:path_to_bowtie2-2.?.?
setenv PATH {$PATH}:path_to_yass
```

<For bash shell>
Add the following lines in your ./bashrc or .bash_profile.

```
PATH=$PATH:path_to_MUMmer3.23
PATH=$PATH:path_to_ncbi-blast-2.2.??+/bin
PATH=$PATH:path_to_bowtie2-2.?.?
PATH=$PATH:path_to_yass
```

export PATH

3. Command line options

--target_scaf or -t <STR> input target scaffold fasta file [mandatory]

--query_contig or -q <STR> input query fasta file of a contig or long read set (if long reads are used, -lr option must be specified) [mandatory]

--prefix_out or –p <STR> prefix name of output files to be output in the working directory (must not contain directory names) [mandatory]

--read_file or -r <STR> space-separated list of fastq or fasta files of paired-end reads (e.g., -r read_pe1.fq read_pe2.fq) [gzip compressed files (*.gz) are acceptable]

--read_len or -l <INT> read length of the paired-end reads specified with the –r, -st, -sq, or –sd option (mean read length if read lengths are variable) [mandatory]

--insert or -i <INT> average insert size of paired-end reads. INT must be > read_len and < 20001. [default: 400]

--sd_insert or -d <INT> standard deviation of insert sizes of paired-end reads [default: 40]

--read_format or -f <STR> fastq or fasta [default: fastq]

--sam_talign or -st <STR> space-separated list of sam alignment file(s) for target scaffolds, created in a single-end read alignment mode for paired-end reads (e.g., -sa tPE1.sam tPE2.sam, for paired-end read files PE1.fq and PE2.fq) [optional]

--sam_qalign or -sq <STR> space-separated list of sam alignment file(s) for query sequences, created in a single-end read alignment mode for paired-end reads (e.g., -sa qPE1.sam qPE2.sam, for paired-end read files PE1.fq and PE2.fq) [optional]

--sam_dir or –sd <STR> path of directory (i.e., bowtie_align) containing sam alignment files generated from a previous job with GMcloser (this can be used in place of -st and -sq option)

--align_file or -a <STR> Nucmer or BLAST alignment file for query against target. If a BLAST alignment file is specified, the --blast option must be specified [optional]

--connect_subcon or -c connect between gap-encompassing subcontig pairs with their original (not merged with query contigs) termini [default: false]

--extend or –et extend scaffold termini with aligned contig or long-read sequences (This option cannot be used when --long_read option is specified) [default: false]

--blast or -b align target and query contigs with BLAST [default: false] (Nucmer alignment by default)

--min_match_len or -mm <INT> minimum overlap length to be filtered for Nucmer or BLAST alignments. Contig alignments that satisfy both the values specified with -mm and -mi options are selected, irrespective of the mapping rates of PE reads. INT must be ≥ 50. [default: 300]

--min_identity or -mi <INT> minimum overlap identity (%) to be filtered for Nucmer or BLAST alignments. Contig alignments that satisfy both the values specified with -mm and -mi options are selected, irrespective of the mapping rates of PE-reads. INT must be ≥ 95 and ≤ 100. [default: 95]

--min_len_local or -ml <INT> minimum overlap when merging between neighbor subcontigs with YASS aligner. When the alignment identity is ≥ 97% and the mapping rate of PE reads are ≥ 0.2, the specified value is neglected. INT must be ≥ 15. [default: 20]

--min_subcon or -ms <INT> minimum length of subcontigs, to be used for gap closing [default: 100 for Nucmer, 150 for Blast]

--min_gap_size or -g <INT> minimum length of gap, when splitting the target scaffold sequences into subcontigs [default: 1]

--max_indel or -is <INT> maximum length of indels, observed in alignments between target subcontigs and query contigs. The alignments separated by the indel will be merged [default: 70]

--max_qsc or -qsc <INT> maximum alignment coverage (%) of query singletones for target subcontigs (query with ≥ INT is excluded from the query singletone output) [default: 60]

--base_qual or -bq <STR> base call quality format of fastq read file; illumina (phred64) or sanger (phred33) [default: auto]

--nuc_len or -nl <INT> nucmer exact match length, a value specified with the '-l' option of the Nucmer aligner [default: auto, increased from 30 to 50 depending on the total contig length]

--ad_score or -as <FLOAT> score to add to (subtract from) the standard threshold score for the selection of the correct contig–subcontig alignments (e.g., 1 or −1) [default: 0]

--hetero or -ht heterozygosity factor (specify this when the input assemblies and PE reads are heterozygous, i.e., (differ by > 0.2% of the haploid size). This increases the threshold value for filtering based on the mismatch number of aligned PE reads by 1 and reduces the threshold value for the initial filtering based on the Nucmer (BLAST) alignment identity by 1% [default: false]

--thread or -n <INT> number of threads (for machines with multiple processors), allows all the alignment processes to be performed in parallel [default: 5]

--thread_connect or -nc <INT> number of threads (for machines with multiple processors), allows the subcontig connection process to be performed in parallel. When the memory consumption at the subcontig connection step is large, you can reduce it by decreasing the threads with this option. [default: the number specified with --thread]

-split only splitting the target scaffolds into subcontigs (required options: -t, -ms -mg, and -p, generating $out_prefix-subcon.fa)

--help or -h output help message

[Options when a long-read set is used as query (long-read mode)]

--long_read or -LR	query sequence file is a fasta file of long reads [default: false] (this option must be specified when a long-read set is used. If PacBio reads are used, they must be error-corrected)

--lr_cov or -lc <INT>	fold coverage of long reads for target scaffolds [default: auto; automatically calculated by dividing the total length of the query by the total length of the target]

--min_qalign or -mq <INT>	minimum number of queries that are aligned to a target subcontig (long reads with < INT are not used for gap closure) [default: 1]

--iterate or -it <INT>	number of iterations [default: 3]

--alignq or -aq <STR>	BLAST alignment file for query against query [optional]

--query_index or -qi <STR>	base name of bowtie2 index files for query long reads [optional]

【Caution】

GMcloser can be run with the alignment files that were generated in the previous GMcloser execution using the options, --sam_dir, --sam_talign, --sam_qalign, or –align_file. It should be taken care that the previous alignment data to be used must be the same as the datasets of the target scaffolds, query contigs, and PE reads, and the optional values to split target scaffolds (which are specified with --min_subcon and –min_gap_size) must be the same as those specified when obtaining the previous alignment data.

4. Output files

(1) {$prefix_out}.gapclosed.fa

Fasta file of gap-closed scaffolds.

(2) {$prefix_out}.query-singletone.fa

Fasta file of query contigs that show no or little overlap (≤ % coverage specified with --max_qsc) with the target subcontigs. (This file is not produced when the --lr option is specified.)

(3) {$target_scaf}-subcon.fa

Fasta file of subcontigs encompassing gaps (≥ bp specified with --min_gap_size) in the target scaffolds.

(4) {$prefix_out}-1m.nucmer.coords ({$prefix_out}-1m.blast.out)

Nucmer (or BLASTn) alignment output file.

<Nucmer output>

1st column: start position of the target subcontig alignment
2nd: end position of the target subcontig alignment
3rd: start position of the query alignment
4th: end position of the query alignment
5th: alignment overlap length of the target subcontig
6th: alignment overlap length of the query sequence
7th: alignment overlap identity (%)
8th: length of the target subcontig
9th: length of the query sequence
10th: coverage (%) of the target subcontig
11th: coverage (%) of the query sequence
12th: name of the target subcontig
13th: name of the query sequence

<BLAST output>

1st column: name of the query sequence
2nd: name of the target subcontig

3rd: alignment overlap identity (%)
4th: alignment overlap length
5th: number of mismatches
6th: number of gap opens
7th: start position of the query alignment
8th: end position of the query alignment
9th: start position of the target subcontig alignment
10th: end position of the target subcontig alignment
11th: expect value (Evalue)
12th: bit score

(5) {$prefix_out}-2m.nucmer-filt.txt　({$prefix_out}-2m.blast-filt.txt)
Filtered Nucmer (or BLASTn) alignment result, which is a list of end-to-end alignments or alignments in which either the query or the target subcontig is completely covered.

1st column: name of the query sequence
2nd: start position of the query alignment
3rd: end position of the query alignment
4th: length of the query sequence
5th: name of the target subcontig
6th: start position of the target subcontig alignment
7th: end position of the target subcontig alignment
8th: length of the target subcontig
9th: strand of the aligned (sub)contig
10th: alignment overlap identity (%)

(6) {$prefix_out}-3m. selected-align-list.txt
List of alignments selected based on a combined likelihood score calculated from the likelihood ratios for the alignment overlap length, the overlap identity, and the mapping rate of the PE reads.

1st column: name of the query sequence

2nd: start position of the query alignment
3rd: end position of the query alignment
4th: name of the target subcontig
5th: start position of the target subcontig alignment
6th: end position of the target subcontig alignment
7th: strand of the aligned (sub)contig
8th: alignment overlap length
9th: alignment overlap identity (%)

(7) {$prefix_out}-4m.contigs/longreads-used-forgapfill.txt

List of query contigs (or long reads) used for gap closure.

1st column: name of the target subcontig
2nd: assigned region of the subcontig
(5-term: query is assigned to the 5'-terminal region of the subcontig; 3-term: assigned to the 3'-terminal region of the subcontig; gap-all: assigned to regions completely covering the gap that is located at the 3' terminus of the subcontig)
3rd: query name of contig or long read assigned to the subcontig/gap

(8) {$prefix_out}-score.txt

List of alignments before the likelihood-based selection, showing the mapping rate of the PE reads and the combined likelihood scores.

1st–9th columns: same as those in (6)
10th: mapping rate of the PE reads
11th: combined likelihood score

(9) {$prefix_out}-bowtie_align/{$prefix_out}-target-align-se-[1..n].sam

SAM-related file of the filtered alignments of a single read set of PE reads to the target subcontigs.
Read files are split into several subfiles according to the number of threads to be used and the corresponding sam files are generated.

The first six columns from the alignment SAM output are printed in the files, and the read names in the first column are simplified.

(10) {$prefix_out}-bowtie_align/{$prefix_out}-query-align-se-[1..n].sam

SAM-related file of the filtered alignments of a single read set of PE reads to the query sequences.

The file format is the same as that of (8).

(11) {$prefix_out}.submit.log

A log file in which the submitted command is recorded.

<<Other output files when specifying the --long_read option>>

When the number specified with the --iterate option is larger than 1, iteration-directories corresponding iteration-cysles are created and output files are produced in each of the iteration-directories. An blast alignment output file for the long-read query data against the long-read query database is produced as '{$prefix_out}-QQ.blast.out'. Other alignment data for the query dataset are stored in a 'Query-Query-align' directory for the subsequent iteration.

5. Command examples

(1) Gap closing in rice scaffolds (scaf.fa) with a contig set (contig.fa):

```
gmcloser -t scaf.fa -q contig.fa -r pe_read_1.fq pe_read_2.fq -p out1 -l 100 -i 500 -d 50 -c -n 20
```

(2) Gap closing with pre-existing alignment files:

```
gmcloser -t scaf.fa -q contig.fa -p out2 -a align1.coords -st align.t1.sam align.t2.sam -sq align.q1.sam align.q2.sam -l 100 -i 500 -d 50 -c -n 12
```

(3) Gap closing with error-corrected PacBio reads (e.g., 6x coverage)

```
gmcloser -t scaf.fa -q read.fa -lr -it 3 -mq 2 -p out3 -l 100 -i 500 -d 50 -c
-n 12
```

C. GMvalue

1. About GMvalue

GMvalue is a utility program to evaluate the assembly accuracy and to create error-free (error-corrected) assemblies from an input assembly set. This program finds misassemblies in an input contig set or scaffold set by aligning the assemblies to a reference sequences with Nucmer, with a method related to that used for the GAGE (Salzberg et al., 2012, *Genome Res*. 22:557-567) and Quast (Gurevich et al., 2013, *Bioinformatics* 29:1072-1075) assembly evaluation methods. Unlike GAGE and Quast, GMvalue can specifies a minimum overlap identity, a minimum coverage of query, and a maximum allowable size of indels for alignments between the reference and query sequences, and can be applied to assemblies of large (> 300 Mb) genomes. When a contig set is used to evaluate its misassemblies (the corresponding command: **"gmvalue contig"**), the program counts misassembly events according to the following criteria.

(1) Multiple segments from a contig are aligned to a different chromosome of the reference or a different strand of the same chromosome.

(2) When multiple segments from a contig are aligned to an identical strand of the same chromosome, the break points of the aligned contig segments are at a distance larger than a value specified with the maximum indel size (default: 100). In this case, the break point indicates the position of the reference-alignment site corresponding to the broken side of a contig segment.

(3) Overall coverage of a contig (or multiple contig segments aligned to the reference) aligned to the reference is smaller than a minimum coverage (default: 99%) and a minimum identity (default: 97%).

(4) Misassembled contig segments with alignment lengths of < 200 bp to the reference was counted as one misassemble event, even if multiple < 200-bp segments in the contig were aligned with the reference.

- Unlike GAGE and Quast, GMvalue can specify “minimum identity of alignment overlap”, “minimum coverage of aligned query”, and “maximum indel size (maximum break point distance)” for defining misassemblies, as explained above.

- If you wish to evaluate the misassembly events contained in subcontigs in a scaffold set, use the command **“gmvalue subcon”**, which is most suitable for evaluating misassemblies introduced into scaffolds through a gap-closing process.

- If you wish to evaluate for the connectivity and integrity of subcontigs in scaffolds, use the command **“gmvalue scaf”**, which is suitable for evaluating misconnections introduced into scaffolds through a scaffolding process.

- In each command, an error-free assembly set can be created by specifying the --error_correct option, which split, correct, and/or remove misassemblies, although SNPs and short indels are not corrected. Complete correction of misassemblies is not often observed in a single run. In such case, repeat the run with the -e option until the number of misassemblies reaches a desired number.

- Alignment data (i.e., coords files) with Nucmer are obtained with the following commands:

```
nucmer -maxmatch -banded -D 5 -l <min_match>
delta-filter -o 95 -i <min_identity>
```

```
  show-coords –THrcl
```

where <min_match> is 30, 40, or 50, depending on assembly size, and <min_identity> is set with the value specified with the -mi option.

2. Command line options

[Usage]

gmvalue [contig|subcon|scaf] [options]

[Options for gmvalue contig]

--ref or -r <STR>	input fasta file of reference [mandatory]
--query or -q <STR>	input fasta file of a query contig (scaffold) set [mandatory]
--prefix or -p <INT>	prefix name of output files
--min_id or -mi <INT>	minimum alignment identity (%) [default: 97]
--min_cov or -mc <INT>	minimum coverage (%) of query (contig) aligned to a reference [default: 99]
--min_align or -ma <INT>	minimum alignment overlap length. The overlap region may contain indels with a maximum size, specified with the --max_indel option [default: 200]
--min_len or -ml <INT>	minimum contig length to be considered [default: 200]
--max_indel or -is <INT>	maximum allowable size of indels in subcontigs (or distance between break points of a local misassembly) [default: 100]
--nuc_len or -l <INT>	minimum exact match length for specifying nucmer option -l [default: 30]
--error_correct or -e	output an error-corrected contig set [default: false]

--thread, -n or –t	number of threads to run [default: 1]
--help or –h	output help message

[Options specific to gmvalue subcon]

--min_gap or -g <INT>	minimum gap size in query scaffolds to split into subcontigs [default: 1]

[Options specific to gmvalue scaf]

--min_gap or -g <INT>	minimum gap size in query scaffolds to split into subcontigs [default: 1]
--max_gap or -mg <INT>	maximum length of gaps contained in the scaffolds [default: 50000]

3. Output files

(1) {$prefix_out}.stat.txt

Sequence statistics and misassembly evaluation results for query assemblies.

(2) {$prefix_out}.misassemble.list.txt

List of query sequences containing misassemblies.

<Output from the “gmvalue contig” and “gmvalue subcon” commands>

1st column: Name of the query contig or a subcontig in the query scaffold
2nd: Alignment start position of the query contig
3rd: Alignment end position of the query contig
4th: Length of the query contig
5th: Reference name aligned with the contig
6th: Category of misassembly

<Output from the "gmvalue scaf" command>

1st column: Name of the query scaffold

2nd: Subcontig name (no.) with misassemblies or subcontig junction (e.g., 5-6: the link between subcontigs no.5 and no.6) with mislinks in the scaffold

3rd: Tags to indicate 'mislink' or 'contig-misassembly'; mislink: misassembly of contig linking generated through scaffolding, contig-misassembly: misassembly within a (sub)contig in a scaffold generated through contig-assembly

(3) {$prefix_out}.corrected.fa

Fasta file of error-corrected sequences from query assemblies, generated when the -e option is specified.

(4) {$prefix_out}.coords, {$prefix_out}..delta, {$prefix_out}.fdelta

Output files of MUMmer/Nucmer alignment runs.

D. Tutorial with sample data

1. Download GMcloser and Sample data

Download the GMcloser package and the sample data from https://sourceforge.net/projects/gmcloser/.

Decompress the downloaded compressed files by the following commands.

```
gzip -dc GMcloser-1.2.tar.gz | tar xvf - [Return]
gzip -dc Sample_data.tar.gz | tar xvf - [Return]
```

2. Installation

GMcloser requires the following outside alignment software. The PATH environmental variables for each tool must be set.

(1) Bowtie 2 (http://bowtie-bio.sourceforge.net/bowtie2/index.shtml)

(2) MUMmer 3.23 (http://mummer.sourceforge.net) (MUMer 3.22 is not allowed)

(3) blast+ 2.2.18 or later
(ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+)

(4) YASS (http://bioinfo.lifl.fr/yass/)
If the installed executable is "yass.bin", that should be linked to "yass" to allow the implementation with a 'yass' command (e.g., `ln -s yass.bin yass`).

By entering the following commands, you can confirm whether the software is installed and the PATH environmental variables are set on your system. The commands should give explanations for executable options for each tool on the console.

<For Bowtie2>

`bowtie2` [Return]

<For MUMmer>

`nucmer -h` [Return]

<For blast+>

`blastn -h` [Return]

<For YASS>

`yass` [Return]

If the commands return a message of "command not found", the corresponding tools must be installed and the PATH environmental variables must be set to your system as follows:

<For tcsh or csh shell>

Add the following lines in your .cshrc.

setenv PATH {$PATH}:path_to_MUMmer3.23

(e.g., setenv PATH {$PATH}:/home/myname/tools/MUMmer3.23)

setenv PATH {$PATH}:path_to_ncbi-blast-2.2.??+/bin

```
setenv PATH {$PATH}:path_to_bowtie2-2.?.?
setenv PATH {$PATH}:path_to_yass
```

<For bash shell>

Add the following lines in your ./bashrc or .bash_profile.

```
PATH=$PATH:path_to_MUMmer3.23
PATH=$PATH:path_to_ncbi-blast-2.2.??+/bin
PATH=$PATH:path_to_bowtie2-2.?.?
PATH=$PATH:path_to_yass
export PATH
```

Also, the PATH of GMcloser may be set to execute “gmcloser” command. Alternatively, GMcloser can be conduct by executing with an absolute path of gmcloser (for example, if the GMcloser-1.2 directory is installed at /home/myname/tools/, the executable command of gmcloser is /home/myname/tools/GMcloser-1.2/gmcloser).

3. Run GMcloser

Now you finished to prepare for executing GMcloser, try to run GMcloser with the downloaded sample data.

The sample data contains the sequencing data of the *Rhodobacter sphaeroides* genome and the assembly sets. The file components are follows:

(1) An error-free scaffold set to be gap closed (scaffolds.fasta). The scaffolds are composed of 239 sequences (the total length: 4,716,760 bp) containing 659,949 bp of gaps.

(2) A preassembled real contig set generated with FERMI (contigs.fasta).

(3) An illumina paired-end read set from SRR363373 (pe300-40x_1.fastq and pe300-40x_2.fastq, read length is 100 bp and the fragment size is approx. 300 bp).

(4) An error-corrected PacBio read set from SRP001079 (pacbio-error-corrected-7x.fasta).

(5) The *R. sphaeroides* reference genome from the GAGE website (Rhodobacter.genome.fasta).

Change the current directory to the Sample_data directory (e.g., `cd Sample_data`).

For the run with the preassembled contig set, use the following command:

```
gmcloser -t scaffolds.fasta -q contigs.fasta -r pe300-40x_1.fastq pe300-40x_2.fastq -l 100 -i 300 -d 40 -c -n 4 -p sample
```

[Return]

(or **`absolute_path_to_GMcloser-1.2_folder/gmcloser`** ---- if PATH of GMcloser is not set)

For the run with the PacBio read set, use the following command:

```
gmcloser -LR -t scaffolds.fasta -q pacbio-error-corrected-7x.fasta -r pe300-40x_1.fastq pe300-40x_2.fastq -l 100 -i 300 -d 40 -it 3 -c -n 4 -p sample
```

[Return]

When the run was successfully completed, several output files including a gap-closed assembly set (sample.gapclosed.fa) should be created in the sample directory.

4. Run GMcloser step by step using pre-generated alignment data

(This may be helpful for speeding up large data processing with separate computational resources.)

(1) Align short reads to the target subcontigs with bowtie2

(a) Split target scaffolds into subcontigs:

```
gmcloser -t scaffolds.fasta -p sample
```

[Return]

(b) Index subcontigs

```
mkdir sample-bowtie_align
```

[Return]

```
mkdir sample-bowtie_align/index [Return]
bowtie2-build -q -f sample-subcon.fa
sample-bowtie_align/index/sample-subcon.fa [Return]
```

(c) Align short reads to subcontigs

```
bowtie2 --sanger -p 10 -x
sample-bowtie_align/index/sample-subcon.fa
pe300-40x_1.fastq | coval-filter-short.pl -n
$max_mismatch -r sample-subcon.fa - >
sample-bowtie_align/sample-target-align-se-1.sam
[Return]
bowtie2 --sanger -p 10 -x
sample-bowtie_align/index/sample-subcon.fa
pe300-40x_2.fastq | coval-filter-short.pl -n
$max_mismatch -r sample-subcon.fa - >
sample-bowtie_align/sample-target-align-se-2.sam
[Return]
```

($max_mismatch: 1 for read_length < 100 bp, int (read_length * 0.015) for read_length >= 100 bp)

(2) Align short reads to the query contigs (or long reads) with bowtie2

(a) Index query sequences

```
bowtie2-build -q -f pacbio-error-corrected-7x.fasta
sample-bowtie_align/index/
pacbio-error-corrected-7x.fasta [Return]
```

(b) Align short reads to query sequences

```
bowtie2 --sanger -k $LR_coverage -p 10 -x
sample-bowtie_align/index/
pacbio-error-corrected-7x.fasta pe300-40x_1.fastq |
coval-filter-short.pl -n $max_mismatch -r
sample-subcon.fa - >
sample-bowtie_align/sample-query-align-se-1.sam [Return]

bowtie2 --sanger -k $LR_coverage -p 10 -x
```

```
sample-bowtie_align/index/
pacbio-error-corrected-7x.fasta pe300-40x_2.fastq |
coval-filter-short.pl -n $max_mismatch -r
sample-subcon.fa - >
sample-bowtie_align/sample-query-align-se-2.sam [Return]
```

($LR_coverage: fold-coverage of long reads, the -k option can be omitted when ~1x contig sequences are used as query.)

(3) Align the query contigs (or long reads) to the target subcontigs with blastn

(a) Index target subcontigs

```
mkdir blast_db [Return]
cp sample-subcon.fa blast_db/ [Return]
makeblastdb -in blast_db/sample-subcon.fa -dbtype nucl
[Return]
```

(b) Align query sequences to target subcontigs

```
blastn -db blast_db/sample-subcon.fa -query
pacbio-error-corrected-7x.fasta -out Sample.blast.out
-outfmt 6 -num_alignments 100 -perc_identity 95
-num_threads 5 [Return]
```

(4) Align the query contigs (or long reads) to the query contigs with blastn (only for the long read mode)

(a) Index query sequences

```
cp pacbio-error-corrected-7x.fasta blast_db/ [Return]
makeblastdb -in blast_db/
pacbio-error-corrected-7x.fasta -dbtype nucl [Return]
```

(b) Align query sequences to query sequences

```
blastn -db blast_db/pacbio-error-corrected-7x.fasta
-query pacbio-error-corrected-7x.fasta -out
Sample-QQ.blast.out -outfmt 6 -num_alignments 100
-perc_identity 95 -num_threads 5 [Return]
```

(5) For the 1st iteration in the long read mode using the above alignment data, use the following command:

```
gmcloser -t scaffolds.fasta -q contigs.fasta -l 100 -i 300
-d 40 -c -n 4 -p sample2 -ad sample-bowtie_align -a
Sample.blast.out -aq Sample-QQ.blast.out [Return]
```

(The directory specified with the -ad option should contain sam files with names ‘*-target-align-se-*‘ and ‘*-query-align-se-*‘.)

For further rounds of iteration in the long read mode, repeat the steps (1), (3), and (5) [the alignment data generated at the steps (2) and (4) can be reused for > 2nd iterations].

5. Evaluate the gap-closed assemblies with GMvalue

Using the GMvalue tool contained in the GMcloser package, you can determine the total length of the gap-closed sequences and the number of misassemblies introduced into the gap-closed assemblies through the gap-closing process.
To evaluate the gap-closed sequences from the step 3, run the following command.

```
mkdir eval [Return]
cd eval [Return]
gmvalue subcon -r ../Rhodobacter.genome.fasta
-q ../sample.gapclosed.fa -p sample.gapclosed [Return]
```

(or **absolute_path_to_GMcloser-1.2_folder/gmvalue subcon** ---- if PATH of GMcloser is not set)

Within a few minutes, the output statistics will appear on the console and in a stat file (sample.gapclosed.stat.txt).

Change logs

Ver. 1.6.2:

(1) The setting of Getopt::Long module was modified to fix wrong recognition of specified options.

(2) An option (--split) to split the target scaffolds into subcontigs was added.

(3) A protocol for running GMcloser with pre-generated alignment data was added to the tutorial section of the manual (pages 23-26).

Ver. 1.6.1:

(1) Revised to be able to use gzip compressed short read fastq files.

(2) An option (--query_index or –qi) to specify bowtie2 index files of the query long reads for the long-read mode was added. This option is useful for the re-use of the index files of a large size of the query long read file.

Ver. 1.6:

(1) An option (--thread_connect or -nc) to specify the number of threads at the subcontig connection step of GMcloser was added. When the memory usage at the subcontig connection step is large, decreasing in the number of threads with this option may help decrease the memory consumption.

(2) The codes were revised to reduce the memory consumption at the subcontig connection step of GMcloser.

Ver. 1.5.1:

(1) The scaffold evaluation of GMvalue is modified to show separately mislinking and local-misassembly counts of subcontigs.

(2) The contig (subcontig) evaluation of GMvalue is modified to show

local-misassembies to be classified into two categories: one with break point distance >= 1000 bp and the other with break point distance > 100 bp and < 1000 bp.

(3) An additional option (--min_align_len/-ma) to specify a minimum overlap of Nucmer alignments for GMvalue is added. This had been previously shared with the value specified with the --min_len option.

(4) An option (--thread_connect or -nc) to specify the number of threads at the subcontig connection step of GMcloser was added. When the memory usage at the subcontig connection step is large, decreasing in the number of threads with this option may help decrease the memory consumption.

(5) The codes were revised to reduce the memory consumption at the subcontig connection step of GMcloser.

Ver. 1.5:

(1) The codes were revised to reduce the memory consumption of both GMcloser and GMvalue.

(2) The output files of GMvalue was changed to describe the category of misassemblies (e.g., local-misassembly, translocation, inversion, etc.)

Ver. 1.4:

(1) A bug found in the gmcloser codes with a long read set was fixed. The command gmval was renamed to gmvalue.

Ver.1.3:

(1) The efficiency and accuracy to close gaps (i.e., connect neighboring connectable subcontigs that are present in input scaffolds) have been improved by adjusting the criteria for the selection of correct alignments.

(2) A bug in the code to connect subcontigs has been fixed. In the previous version, when there are a contig that spans a ‘negative gap’, whose neighboring subcontigs are connectable, in the scaffold, GMcloser did not use this contig-alignment information and left the

gap unclosed.

(3) The codes to select a single alignment from multiple contig alignments have been modified.

(4) The default values of the options --min_len_local and --min_gap_size were changed to 20 and 1, respectively.

(5) The default values of the option --min_subcon was changed to 100 and 150 when using Nucmer and Blast, respectively.

(6) The codes in GMvalue to treat split contig segments that are aligned to the reference have been modified (i.e., out of split contig segments overlapped with ≥ 80% coverage for either contig segment, shorter contig segments are discarded).

Ver.1.2:

(1) A program optimized to use a long-read set as the input query is included in the package. This program can be used by specifying the --long_read option.

(2) The codes to connect a subcontig pair surrounding a gap by local alignment have been modified to increase the accuracy and effectiveness.

(3) A bug present in a code that assigns contigs to subcontigs was fixed.

(4) In the previous version, the termini of the target assemblies were extended with aligned query contigs by default. In this version, this extension is selectable with the --extend option, which is ‘off’ by default.

License

GMcloser

Contact

Shunichi Kosugi　Center for Integrative Medical Sciences, RIKEN
Email: shunichi.kosugi@riken.jp