LOCAS Manual

Juliane D. Klein
kleinj@informatik.uni-tuebingen.de

November 26, 2010

Contents

1 Introduction

LOCAS is a program to assemble short reads of second generation sequencing technologies. It explicitly handles low coverage data by allowing mismatches in the overlap alignment of reads. An extra module, called SUPERLOCAS, provides some additional features for resequencing projects. In a resequencing project reads are mapped onto a closely related reference genome and a consensus from the mapped reads is calculated as an approximation of the new genome sequence. (Highly polymorphic regions and insert sites are not covered with this consensus.) SUPERLOCAS can be used to incorporate unmapped reads into the assembly of mapped regions and elongate this consensus. Further, SUPERLOCAS takes advantage of given mapping positions of reads. Both tools are written in C++.

2 Availability

Binaries and source code can be downloaded from `http://www-ab.informatik.uni-tuebingen.de/software/locas`.

3 Installing

We provide binaries for LOCAS and SUPERLOCAS that are compiled on a LINUX system (Redhat, 64-bit). Additionally, the source code is available.

4 License Details

The source code is distributed under the terms of the GNU General Public License.

5 Author

Juliane D. Klein

6 Running LOCAS

You can run LOCAS from the command line as follows:

```
LOCAS -I input reads.fasta -O output folder -F fasta -L 23 -S 2
```

LOCAS will assemble all reads in the file "inputreads.fasta" by calculating overlap alignments with a minimal length of 23 and a maximum of 2 mismatches. The result is written to folder output-folder. LOCAS will create this folder if it does not exist.

Required options:

$-I\ \langle string \rangle$	defines the full name (including location) of your read file
$-O\ \langle string \rangle$	the name of the new folder which LOCAS will create for its output
$-F\ \ fasta$ or	
$-F\ \ fastq$	chooses between fasta and fastq file formats

Additional options:

$-C\ \langle int \rangle\ \langle int \rangle$	the first number defines the minimal length and the second number the minimal coverage of the contigs that should be reported (default: 0 0)
$-S\ \langle int \rangle$	the maximal number of allowed mismatches in an overlap alignment between two reads (default: 4)
$-L\ \langle int \rangle$	the minimal allowed length of an overlap alignment between two reads (default: 27)
$-P\ \ kmer\ \langle int \rangle$	the length of the sub-sequence (kmer) which has to be equal in two reads before an overlap alignment is calculated (default: 13)

6.1 How to choose the parameters *kmer size* and *overlap length*

First LOCAS searches for pairs of possibly overlapping reads. This is done by looking for equal kmers (sub-sequences of length k) between two reads. After this filtering step, the real overlap alignments are calculated for all reads which share a kmer.
The parameter $-K$ controls the filtering step. The user can define for which length equal sub-sequences are detected between reads. With the parameter $-L$ the minimal length of an overlap alignment is set.

6.2 Example of a LOCAS run

The data for an example can be found in the sub-folder "testset_locas". Open a shell and change into this folder. Then execute the following command:

```
LOCAS -I simulated_reads.fasta -O my_locas_out -F fasta -L 21 -S 2
```

You can compare your results to those in sub-folder "locas_out". Now, you can experiment with the parameter a little. For example, you can discard some contigs by applying:

```
LOCAS -I simulated_reads.fasta -O my_locas_out2 -F fasta
 -L 21 -S 2 -C 3 100.
```

Or you can change the minimal overlap length between reads by executing:

```
LOCAS -I simulated_reads.fasta -O my_locas_out3 -F fasta -L 19 -S 2
```

7 Running SUPERLOCAS

SUPERLOCAS is especially designed for use in resequencing projects. Here all reads are mapped against the genome of a highly related species. Continuously mapped sub-regions can be defined as blocks. All reads can be assigned to one block or to a set of left-over reads.
Using SUPERLOCAS, you can reassemble the blocks and try to incorporate left-over reads. Reads from different blocks should be placed in different files, and the left-over reads should be placed in their own files, too.
Before you can start SUPERLOCAS, you have to create two special input-files. The SUPERLOCAS_Inputfile describes the input for SUPERLOCAS. The SUPERLOCAS_Outputfile describes the output for SUPERLOCAS. The first line in the SUPERLOCAS_Inputfile shows the names of all read files for the first block separated by a space character. In the second line you find all names of the read files of the second block and so on. In each line of the SUPERLOCAS_Outputfile you have to write the name of the output-folder of the corresponding block. For an example file, see the files "super_input_file" and "super_output_file" in the folder "testset_superlocas/myoutput".
SUPERLOCAS is started as follows:

```
SUPERLOCAS -I SUPERLOCAS_Inputfile -O SUPERLOCAS_Outputfile
 -LO left_over_file_1 left_over_file_2 -F fasta
```

Required options:

$-I\ \langle string \rangle$	defines location and name of your SUPERLOCAS_inputfile
$-O\ \langle string \rangle$	defines location and name of your SUPERLOCAS_outputfile
$-F\ \langle fasta \rangle$ or	
$-F\ \langle fastq \rangle$	chooses between these file formats
$-LO\ \langle string \rangle$	defines left-over files

Kmer options (optional):

$-Kmerg\ \langle int \rangle$	chooses kmer size which has to be equal in a read from the block and a read of the left-over set to calculate an overlap alignment (default: 30)
$-P\ \ kmer\ \langle\ int \rangle$	chooses kmer size which has to be equal in two block reads before an overlap alignment is calculated (default: 13)
$-K\ \langle int \rangle$	chooses kmer size which has to be equal in two left-over reads (default: 33)

Overlap length options (optional):

$-Lm\ \langle int\rangle$	chooses minimal length of an overlap alignment between a block read and a left-over read (default: 30)
$-Lt\ \langle int\rangle$	chooses minimal length of an overlap alignment between two block reads (default: 21)
$-Llo\ \langle int\rangle$	chooses minimal length of an overlap alignment between two left-over reads (default: 33)

Substitutions in overlap options (optional):

$-Sm\ \langle int\rangle$	chooses maximal number of allowed mismatches in an overlap alignment between a block read and an left-over read (default: 1)
$-St\ \langle int\rangle$	chooses maximal number of allowed mismatches in an overlap alignment between two block reads (default: 3)
$-Slo\ \langle int\rangle$	chooses maximal number of allowed mismatches in an overlap alignment between two left-over reads (default: 1)

Additional options:

$-C\ \langle int\rangle\ \langle int\rangle$	chooses with the first number the minimal length and with the second number minimal coverage of the contigs which should be reported (default: 0 0)
$-DR\ \langle int\rangle\ \langle int\rangle$	determines that all left-over reads are discarded which have more than a certain number of equal kmers in the set of left-over reads, the kmer size is defined with the first number and the second number defines the number of kmer matches (default: 21 500)

7.1 Running SUPERLOCAS with mapping positions

If mapping positions are also available for reads you can switch on a mode of SUPERLOCAS that will take these information into account:

$-P\ \ pos\langle int\rangle$	defines kmer size as usual, but in addition an overlap alignment is also calculated if two reads are very close to each other with respect to their mapping positions

Further alignment constraints can be defined for very close or distant reads (optional):

$-Ltn\ \langle int\rangle$	defines minimal length of an overlap alignment between two block reads which are close to each other (default: 11)
$-Ltd\ \langle int\rangle$	defines minimal length of an overlap alignment between two block reads which are distant to each other (default: 25)
$-Stn\ \langle int\rangle$	defines maximal number of allowed mismatches in an overlap alignment between two block reads which are close to each other (default: 2)
$-Std\ \langle int\rangle$	defines maximal number of allowed mismatches in an overlap alignment between two block reads which are distant to each other (default: 0)

7.2 Understanding the parameters of SUPERLOCAS

SUPERLOCAS distinguishes between two types of reads: reads assigned to a block (because they mapped to the same region to a reference) and reads of the left-over set. The alignment con-

straints for these two types of reads differ and can be defined independently with the following options:
If two reads belong to a block then parameters $-Lt\ \langle int\rangle$, $-P\ kmer\langle int\rangle$, and $-St\ \langle int\rangle$ define your alignment constraints. If both reads are left-over reads then parameter $-K\ \langle int\rangle$, $-Llo\ \langle int\rangle$, and $-Slo\ \langle int\rangle$ define the overlaps. If a block read is aligned to a left-over read then the constraints are set by $-Kmerg\ \langle int\rangle$, $-Lm\ \langle int\rangle$, and $-Slo\ \langle int\rangle$.
If mapping positions are available for the block reads then SUPERLOCAS can take them into account. This option can be switched on by applying the option $-P\ pos\ \langle int\rangle$ (instead of $-P\ kmer\ \langle int\rangle$) to define the kmer size. In this case SUPERLOCAS will not only look for equal kmers between reads to calculate an overlap alignment, but it will also try to overlap read with a very close mapping position. In this mode you can manipulate the overlap alignment constraints of two reads depending on their mapping positions. Two block reads can be classified as distant or close to each other with respect to their mapping position. In the first case the constraints for the overlap alignment are defined using $-Ltd\ \langle int\rangle$ and $-Std\ \langle int\rangle$, in the case of very close block reads with $-Ltn\ \langle int\rangle$ and $-Stn\ \langle int\rangle$.

7.3 Example of a SUPERLOCAS run

The sub-folder "testset_superlocas" contains the files of an example application. There are 16 blocks to assemble and 29199 left-over reads. A read file of single-end Illumina reads and one of paired-end Illumina reads belongs to each block.
You can start SUPERLOCAS by changing into the folder "testset_superlocas" and execute the following command:

```
superlocas -I myoutput/super_input_file
 -O myoutput/super_output_file -LO left_over_reads.fasta -F fasta
```

This should produce the same output as you can see in the folder output.
The two files "super_input_file" and "super_output_file" are the special input and outputfile you have to generate for use with SUPERLOCAS.