Short Metagenomic Assembly Tutorial

## Pengantar

Tutorial ini diadaptasi dari <https://www.hadriengourle.com/tutorials/meta_assembly/>

## Tentang Dataset

Pada tutorial ini, kita akan mencoba menyusun MAGs dari 20 bakteri yang di sequence dengan teknologi Illumina HiSeq (yang disimulasikan dengan [InSilicoSeq](http://insilicoseq.readthedocs.io/). Dataset ini diperoleh dari ekspedisi [Tara Ocean](http://ocean-microbiome.embl.de/companion.html) (selengkapnya di [figshare](https://figshare.com/articles/TARA-NON-REDUNDANT-MAGs/4902923/1))

## Setup Conda

mamba env create -f environment.yml

## Quality Control

mkdir -p data  
(cd data && curl -O -J -L https://osf.io/th9z6/download)  
(cd data && curl -O -J -L https://osf.io/k6vme/download)  
(cd data && chmod -w tara\_reads\_R\*)

mkdir -p results  
(cd results && ln -s ../data/tara\_reads\_\* .)  
(cd results && fastqc tara\_reads\_\*.fastq.gz)

#### Forward

Full preview here

#### Reverse

Full preview here

(cd results && sickle pe -f tara\_reads\_R1.fastq.gz -r tara\_reads\_R2.fastq.gz -t sanger \  
 -o tara\_trimmed\_R1.fastq -p tara\_trimmed\_R2.fastq -s /dev/null)

Output

FastQ paired records kept: 2995072 (1497536 pairs)  
FastQ single records kept: 2460 (from PE1: 2366, from PE2: 94)  
FastQ paired records discarded: 0 (0 pairs)  
FastQ single records discarded: 2460 (from PE1: 94, from PE2: 2366)

## Assembly

(cd results && megahit -1 tara\_trimmed\_R1.fastq -2 tara\_trimmed\_R2.fastq -o tara\_assembly)

Output

2025-02-12 22:45:50 - MEGAHIT v1.2.9  
2025-02-12 22:45:50 - Using megahit\_core with POPCNT and BMI2 support  
2025-02-12 22:45:50 - Convert reads to binary library  
2025-02-12 22:45:51 - b'INFO sequence/io/sequence\_lib.cpp : 75 - Lib 0 (/home/matinnu/courses/short\_read\_metagenomics\_tutorial/results/tara\_trimmed\_R1.fastq,/home/matinnu/courses/short\_read\_metagenomics\_tutorial/results/tara\_trimmed\_R2.fastq): pe, 2995072 reads, 126 max length'  
2025-02-12 22:45:51 - b'INFO utils/utils.h : 152 - Real: 1.0162\tuser: 0.7983\tsys: 0.3193\tmaxrss: 184692'  
2025-02-12 22:45:51 - k-max reset to: 141   
2025-02-12 22:45:51 - Start assembly. Number of CPU threads 22   
2025-02-12 22:45:51 - k list: 21,29,39,59,79,99,119,141   
2025-02-12 22:45:51 - Memory used: 15093769420  
2025-02-12 22:45:51 - Extract solid (k+1)-mers for k = 21   
2025-02-12 22:46:11 - Build graph for k = 21   
2025-02-12 22:46:17 - Assemble contigs from SdBG for k = 21  
2025-02-12 22:46:35 - Local assembly for k = 21  
2025-02-12 22:46:53 - Extract iterative edges from k = 21 to 29   
2025-02-12 22:46:55 - Build graph for k = 29   
2025-02-12 22:46:59 - Assemble contigs from SdBG for k = 29  
2025-02-12 22:47:17 - Local assembly for k = 29  
2025-02-12 22:47:30 - Extract iterative edges from k = 29 to 39   
2025-02-12 22:47:32 - Build graph for k = 39   
2025-02-12 22:47:36 - Assemble contigs from SdBG for k = 39  
2025-02-12 22:47:55 - Local assembly for k = 39  
2025-02-12 22:48:14 - Extract iterative edges from k = 39 to 59   
2025-02-12 22:48:17 - Build graph for k = 59   
2025-02-12 22:48:22 - Assemble contigs from SdBG for k = 59  
2025-02-12 22:48:40 - Local assembly for k = 59  
2025-02-12 22:49:03 - Extract iterative edges from k = 59 to 79   
2025-02-12 22:49:05 - Build graph for k = 79   
2025-02-12 22:49:09 - Assemble contigs from SdBG for k = 79  
2025-02-12 22:49:28 - Local assembly for k = 79  
2025-02-12 22:49:50 - Extract iterative edges from k = 79 to 99   
2025-02-12 22:49:52 - Build graph for k = 99   
2025-02-12 22:49:56 - Assemble contigs from SdBG for k = 99  
2025-02-12 22:50:15 - Local assembly for k = 99  
2025-02-12 22:50:42 - Extract iterative edges from k = 99 to 119   
2025-02-12 22:50:44 - Build graph for k = 119   
2025-02-12 22:50:48 - Assemble contigs from SdBG for k = 119  
2025-02-12 22:51:09 - Local assembly for k = 119  
2025-02-12 22:51:30 - Extract iterative edges from k = 119 to 141   
2025-02-12 22:51:30 - Build graph for k = 141   
2025-02-12 22:51:34 - Assemble contigs from SdBG for k = 141  
2025-02-12 22:51:49 - Merging to output final contigs   
2025-02-12 22:51:49 - 5826 contigs, total 23054620 bp, min 215 bp, max 2448145 bp, avg 3957 bp, N50 21953 bp  
2025-02-12 22:51:49 - ALL DONE. Time elapsed: 35

(cd results && ln -s tara\_assembly/final.contigs.fa .)  
(cd results && bowtie2-build final.contigs.fa final.contigs)  
(cd results && bowtie2 -x final.contigs -1 tara\_reads\_R1.fastq.gz -2 tara\_reads\_R2.fastq.gz | \  
 samtools view -bS -o tara\_to\_sort.bam)  
(cd results && samtools sort tara\_to\_sort.bam -o tara.bam)  
(cd results && samtools index tara.bam)

Output

Settings:  
 Output files: "final.contigs.\*.bt2"  
 Line rate: 6 (line is 64 bytes)  
 Lines per side: 1 (side is 64 bytes)  
 Offset rate: 4 (one in 16)  
 FTable chars: 10  
 Strings: unpacked  
 Max bucket size: default  
 Max bucket size, sqrt multiplier: default  
 Max bucket size, len divisor: 4  
 Difference-cover sample period: 1024  
 Endianness: little  
 Actual local endianness: little  
 Sanity checking: disabled  
 Assertions: disabled  
 Random seed: 0  
 Sizeofs: void\*:8, int:4, long:8, size\_t:8  
Input files DNA, FASTA:  
 final.contigs.fa  
Building a SMALL index  
Reading reference sizes  
 Time reading reference sizes: 00:00:00  
Calculating joined length  
Writing header  
Reserving space for joined string  
Joining reference sequences  
 Time to join reference sequences: 00:00:00  
bmax according to bmaxDivN setting: 5763655  
Using parameters --bmax 4322742 --dcv 1024  
 Doing ahead-of-time memory usage test  
 Passed! Constructing with these parameters: --bmax 4322742 --dcv 1024  
Constructing suffix-array element generator  
Building DifferenceCoverSample  
 Building sPrime  
 Building sPrimeOrder  
 V-Sorting samples  
 V-Sorting samples time: 00:00:00  
 Allocating rank array  
 Ranking v-sort output  
 Ranking v-sort output time: 00:00:00  
 Invoking Larsson-Sadakane on ranks  
 Invoking Larsson-Sadakane on ranks time: 00:00:01  
 Sanity-checking and returning  
Building samples  
Reserving space for 12 sample suffixes  
Generating random suffixes  
QSorting 12 sample offsets, eliminating duplicates  
QSorting sample offsets, eliminating duplicates time: 00:00:00  
Multikey QSorting 12 samples  
 (Using difference cover)  
 Multikey QSorting samples time: 00:00:00  
Calculating bucket sizes  
Splitting and merging  
 Splitting and merging time: 00:00:00  
Split 2, merged 7; iterating...  
Splitting and merging  
 Splitting and merging time: 00:00:00  
Split 1, merged 0; iterating...  
Splitting and merging  
 Splitting and merging time: 00:00:00  
Split 1, merged 1; iterating...  
Splitting and merging  
 Splitting and merging time: 00:00:00  
Avg bucket size: 2.88183e+06 (target: 4322741)  
Converting suffix-array elements to index image  
Allocating ftab, absorbFtab  
Entering Ebwt loop  
Getting block 1 of 8  
 Reserving size (4322742) for bucket 1  
 Calculating Z arrays for bucket 1  
 Entering block accumulator loop for bucket 1:  
 bucket 1: 10%  
 bucket 1: 20%  
 bucket 1: 30%  
 bucket 1: 40%  
 bucket 1: 50%  
 bucket 1: 60%  
 bucket 1: 70%  
 bucket 1: 80%  
 bucket 1: 90%  
 bucket 1: 100%  
 Sorting block of length 4137201 for bucket 1  
 (Using difference cover)  
 Sorting block time: 00:00:01  
Returning block of 4137202 for bucket 1  
Getting block 2 of 8  
 Reserving size (4322742) for bucket 2  
 Calculating Z arrays for bucket 2  
 Entering block accumulator loop for bucket 2:  
 bucket 2: 10%  
 bucket 2: 20%  
 bucket 2: 30%  
 bucket 2: 40%  
 bucket 2: 50%  
 bucket 2: 60%  
 bucket 2: 70%  
 bucket 2: 80%  
 bucket 2: 90%  
 bucket 2: 100%  
 Sorting block of length 3179054 for bucket 2  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 3179055 for bucket 2  
Getting block 3 of 8  
 Reserving size (4322742) for bucket 3  
 Calculating Z arrays for bucket 3  
 Entering block accumulator loop for bucket 3:  
 bucket 3: 10%  
 bucket 3: 20%  
 bucket 3: 30%  
 bucket 3: 40%  
 bucket 3: 50%  
 bucket 3: 60%  
 bucket 3: 70%  
 bucket 3: 80%  
 bucket 3: 90%  
 bucket 3: 100%  
 Sorting block of length 2213233 for bucket 3  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 2213234 for bucket 3  
Getting block 4 of 8  
 Reserving size (4322742) for bucket 4  
 Calculating Z arrays for bucket 4  
 Entering block accumulator loop for bucket 4:  
 bucket 4: 10%  
 bucket 4: 20%  
 bucket 4: 30%  
 bucket 4: 40%  
 bucket 4: 50%  
 bucket 4: 60%  
 bucket 4: 70%  
 bucket 4: 80%  
 bucket 4: 90%  
 bucket 4: 100%  
 Sorting block of length 2638513 for bucket 4  
 (Using difference cover)  
 Sorting block time: 00:00:01  
Returning block of 2638514 for bucket 4  
Getting block 5 of 8  
 Reserving size (4322742) for bucket 5  
 Calculating Z arrays for bucket 5  
 Entering block accumulator loop for bucket 5:  
 bucket 5: 10%  
 bucket 5: 20%  
 bucket 5: 30%  
 bucket 5: 40%  
 bucket 5: 50%  
 bucket 5: 60%  
 bucket 5: 70%  
 bucket 5: 80%  
 bucket 5: 90%  
 bucket 5: 100%  
 Sorting block of length 2630120 for bucket 5  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 2630121 for bucket 5  
Getting block 6 of 8  
 Reserving size (4322742) for bucket 6  
 Calculating Z arrays for bucket 6  
 Entering block accumulator loop for bucket 6:  
 bucket 6: 10%  
 bucket 6: 20%  
 bucket 6: 30%  
 bucket 6: 40%  
 bucket 6: 50%  
 bucket 6: 60%  
 bucket 6: 70%  
 bucket 6: 80%  
 bucket 6: 90%  
 bucket 6: 100%  
 Sorting block of length 3107963 for bucket 6  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 3107964 for bucket 6  
Getting block 7 of 8  
 Reserving size (4322742) for bucket 7  
 Calculating Z arrays for bucket 7  
 Entering block accumulator loop for bucket 7:  
 bucket 7: 10%  
 bucket 7: 20%  
 bucket 7: 30%  
 bucket 7: 40%  
 bucket 7: 50%  
 bucket 7: 60%  
 bucket 7: 70%  
 bucket 7: 80%  
 bucket 7: 90%  
 bucket 7: 100%  
 Sorting block of length 2715977 for bucket 7  
 (Using difference cover)  
 Sorting block time: 00:00:01  
Returning block of 2715978 for bucket 7  
Getting block 8 of 8  
 Reserving size (4322742) for bucket 8  
 Calculating Z arrays for bucket 8  
 Entering block accumulator loop for bucket 8:  
 bucket 8: 10%  
 bucket 8: 20%  
 bucket 8: 30%  
 bucket 8: 40%  
 bucket 8: 50%  
 bucket 8: 60%  
 bucket 8: 70%  
 bucket 8: 80%  
 bucket 8: 90%  
 bucket 8: 100%  
 Sorting block of length 2432552 for bucket 8  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 2432553 for bucket 8  
Exited Ebwt loop  
fchr[A]: 0  
fchr[C]: 5752548  
fchr[G]: 11542726  
fchr[T]: 17335916  
fchr[$]: 23054620  
Exiting Ebwt::buildToDisk()  
Returning from initFromVector  
Wrote 12195856 bytes to primary EBWT file: final.contigs.1.bt2.tmp  
Wrote 5763660 bytes to secondary EBWT file: final.contigs.2.bt2.tmp  
Re-opening \_in1 and \_in2 as input streams  
Returning from Ebwt constructor  
Headers:  
 len: 23054620  
 bwtLen: 23054621  
 sz: 5763655  
 bwtSz: 5763656  
 lineRate: 6  
 offRate: 4  
 offMask: 0xfffffff0  
 ftabChars: 10  
 eftabLen: 20  
 eftabSz: 80  
 ftabLen: 1048577  
 ftabSz: 4194308  
 offsLen: 1440914  
 offsSz: 5763656  
 lineSz: 64  
 sideSz: 64  
 sideBwtSz: 48  
 sideBwtLen: 192  
 numSides: 120077  
 numLines: 120077  
 ebwtTotLen: 7684928  
 ebwtTotSz: 7684928  
 color: 0  
 reverse: 0  
Total time for call to driver() for forward index: 00:00:08  
Reading reference sizes  
 Time reading reference sizes: 00:00:00  
Calculating joined length  
Writing header  
Reserving space for joined string  
Joining reference sequences  
 Time to join reference sequences: 00:00:00  
 Time to reverse reference sequence: 00:00:00  
bmax according to bmaxDivN setting: 5763655  
Using parameters --bmax 4322742 --dcv 1024  
 Doing ahead-of-time memory usage test  
 Passed! Constructing with these parameters: --bmax 4322742 --dcv 1024  
Constructing suffix-array element generator  
Building DifferenceCoverSample  
 Building sPrime  
 Building sPrimeOrder  
 V-Sorting samples  
 V-Sorting samples time: 00:00:00  
 Allocating rank array  
 Ranking v-sort output  
 Ranking v-sort output time: 00:00:00  
 Invoking Larsson-Sadakane on ranks  
 Invoking Larsson-Sadakane on ranks time: 00:00:00  
 Sanity-checking and returning  
Building samples  
Reserving space for 12 sample suffixes  
Generating random suffixes  
QSorting 12 sample offsets, eliminating duplicates  
QSorting sample offsets, eliminating duplicates time: 00:00:00  
Multikey QSorting 12 samples  
 (Using difference cover)  
 Multikey QSorting samples time: 00:00:00  
Calculating bucket sizes  
Splitting and merging  
 Splitting and merging time: 00:00:00  
Split 2, merged 6; iterating...  
Splitting and merging  
 Splitting and merging time: 00:00:00  
Avg bucket size: 2.88183e+06 (target: 4322741)  
Converting suffix-array elements to index image  
Allocating ftab, absorbFtab  
Entering Ebwt loop  
Getting block 1 of 8  
 Reserving size (4322742) for bucket 1  
 Calculating Z arrays for bucket 1  
 Entering block accumulator loop for bucket 1:  
 bucket 1: 10%  
 bucket 1: 20%  
 bucket 1: 30%  
 bucket 1: 40%  
 bucket 1: 50%  
 bucket 1: 60%  
 bucket 1: 70%  
 bucket 1: 80%  
 bucket 1: 90%  
 bucket 1: 100%  
 Sorting block of length 3053661 for bucket 1  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 3053662 for bucket 1  
Getting block 2 of 8  
 Reserving size (4322742) for bucket 2  
 Calculating Z arrays for bucket 2  
 Entering block accumulator loop for bucket 2:  
 bucket 2: 10%  
 bucket 2: 20%  
 bucket 2: 30%  
 bucket 2: 40%  
 bucket 2: 50%  
 bucket 2: 60%  
 bucket 2: 70%  
 bucket 2: 80%  
 bucket 2: 90%  
 bucket 2: 100%  
 Sorting block of length 2698885 for bucket 2  
 (Using difference cover)  
 Sorting block time: 00:00:01  
Returning block of 2698886 for bucket 2  
Getting block 3 of 8  
 Reserving size (4322742) for bucket 3  
 Calculating Z arrays for bucket 3  
 Entering block accumulator loop for bucket 3:  
 bucket 3: 10%  
 bucket 3: 20%  
 bucket 3: 30%  
 bucket 3: 40%  
 bucket 3: 50%  
 bucket 3: 60%  
 bucket 3: 70%  
 bucket 3: 80%  
 bucket 3: 90%  
 bucket 3: 100%  
 Sorting block of length 2218194 for bucket 3  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 2218195 for bucket 3  
Getting block 4 of 8  
 Reserving size (4322742) for bucket 4  
 Calculating Z arrays for bucket 4  
 Entering block accumulator loop for bucket 4:  
 bucket 4: 10%  
 bucket 4: 20%  
 bucket 4: 30%  
 bucket 4: 40%  
 bucket 4: 50%  
 bucket 4: 60%  
 bucket 4: 70%  
 bucket 4: 80%  
 bucket 4: 90%  
 bucket 4: 100%  
 Sorting block of length 2795622 for bucket 4  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 2795623 for bucket 4  
Getting block 5 of 8  
 Reserving size (4322742) for bucket 5  
 Calculating Z arrays for bucket 5  
 Entering block accumulator loop for bucket 5:  
 bucket 5: 10%  
 bucket 5: 20%  
 bucket 5: 30%  
 bucket 5: 40%  
 bucket 5: 50%  
 bucket 5: 60%  
 bucket 5: 70%  
 bucket 5: 80%  
 bucket 5: 90%  
 bucket 5: 100%  
 Sorting block of length 2214292 for bucket 5  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 2214293 for bucket 5  
Getting block 6 of 8  
 Reserving size (4322742) for bucket 6  
 Calculating Z arrays for bucket 6  
 Entering block accumulator loop for bucket 6:  
 bucket 6: 10%  
 bucket 6: 20%  
 bucket 6: 30%  
 bucket 6: 40%  
 bucket 6: 50%  
 bucket 6: 60%  
 bucket 6: 70%  
 bucket 6: 80%  
 bucket 6: 90%  
 bucket 6: 100%  
 Sorting block of length 2133259 for bucket 6  
 (Using difference cover)  
 Sorting block time: 00:00:01  
Returning block of 2133260 for bucket 6  
Getting block 7 of 8  
 Reserving size (4322742) for bucket 7  
 Calculating Z arrays for bucket 7  
 Entering block accumulator loop for bucket 7:  
 bucket 7: 10%  
 bucket 7: 20%  
 bucket 7: 30%  
 bucket 7: 40%  
 bucket 7: 50%  
 bucket 7: 60%  
 bucket 7: 70%  
 bucket 7: 80%  
 bucket 7: 90%  
 bucket 7: 100%  
 Sorting block of length 3879811 for bucket 7  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 3879812 for bucket 7  
Getting block 8 of 8  
 Reserving size (4322742) for bucket 8  
 Calculating Z arrays for bucket 8  
 Entering block accumulator loop for bucket 8:  
 bucket 8: 10%  
 bucket 8: 20%  
 bucket 8: 30%  
 bucket 8: 40%  
 bucket 8: 50%  
 bucket 8: 60%  
 bucket 8: 70%  
 bucket 8: 80%  
 bucket 8: 90%  
 bucket 8: 100%  
 Sorting block of length 4060889 for bucket 8  
 (Using difference cover)  
 Sorting block time: 00:00:00  
Returning block of 4060890 for bucket 8  
Exited Ebwt loop  
fchr[A]: 0  
fchr[C]: 5752548  
fchr[G]: 11542726  
fchr[T]: 17335916  
fchr[$]: 23054620  
Exiting Ebwt::buildToDisk()  
Returning from initFromVector  
Wrote 12195856 bytes to primary EBWT file: final.contigs.rev.1.bt2.tmp  
Wrote 5763660 bytes to secondary EBWT file: final.contigs.rev.2.bt2.tmp  
Re-opening \_in1 and \_in2 as input streams  
Returning from Ebwt constructor  
Headers:  
 len: 23054620  
 bwtLen: 23054621  
 sz: 5763655  
 bwtSz: 5763656  
 lineRate: 6  
 offRate: 4  
 offMask: 0xfffffff0  
 ftabChars: 10  
 eftabLen: 20  
 eftabSz: 80  
 ftabLen: 1048577  
 ftabSz: 4194308  
 offsLen: 1440914  
 offsSz: 5763656  
 lineSz: 64  
 sideSz: 64  
 sideBwtSz: 48  
 sideBwtLen: 192  
 numSides: 120077  
 numLines: 120077  
 ebwtTotLen: 7684928  
 ebwtTotSz: 7684928  
 color: 0  
 reverse: 1  
Total time for backward call to driver() for mirror index: 00:00:08  
Renaming final.contigs.3.bt2.tmp to final.contigs.3.bt2  
Renaming final.contigs.4.bt2.tmp to final.contigs.4.bt2  
Renaming final.contigs.1.bt2.tmp to final.contigs.1.bt2  
Renaming final.contigs.2.bt2.tmp to final.contigs.2.bt2  
Renaming final.contigs.rev.1.bt2.tmp to final.contigs.rev.1.bt2  
Renaming final.contigs.rev.2.bt2.tmp to final.contigs.rev.2.bt2  
1499996 reads; of these:  
 1499996 (100.00%) were paired; of these:  
 1067143 (71.14%) aligned concordantly 0 times  
 432632 (28.84%) aligned concordantly exactly 1 time  
 221 (0.01%) aligned concordantly >1 times  
 ----  
 1067143 pairs aligned concordantly 0 times; of these:  
 1048579 (98.26%) aligned discordantly 1 time  
 ----  
 18564 pairs aligned 0 times concordantly or discordantly; of these:  
 37128 mates make up the pairs; of these:  
 13221 (35.61%) aligned 0 times  
 16950 (45.65%) aligned exactly 1 time  
 6957 (18.74%) aligned >1 times  
99.56% overall alignment rate  
[bam\_sort\_core] merging from 1 files and 1 in-memory blocks...

## Binning

(cd results && runMetaBat.sh -m 1500 final.contigs.fa tara.bam)  
(cd results && mv final.contigs.fa.metabat-bins1500\* metabat)