NextDenovo Documentation

Release stable

Mar 13, 2023

Getting started

NextDenovo is a string graph-based *de novo* assembler for long reads (CLR, HiFi and ONT). It uses a "correct-then-assemble" strategy similar to canu (no correction step for PacBio HiFi reads), but requires significantly less computing resources and storages. After assembly, the per-base accuracy is about 98-99.8%, to further improve single base accuracy, try NextPolish.

NextDenovo contains two core modules: NextCorrect and NextGraph. NextCorrect can be used to correct long noisy reads with approximately 15% sequencing errors, and NextGraph can be used to construct a string graph with corrected reads. It also contains a modified version of minimap2 and some useful utilities (see *utilities* for more details).

We benchmarked NextDenovo against other assemblers using Oxford Nanopore long reads from *human* and *Drosophila melanogaster*, and PacBio continuous long reads (CLR) from *Arabidopsis thaliana*. NextDenovo produces more contiguous assemblies with fewer contigs compared to the other tools. NextDenovo also shows a high assembly accurate level in terms of assembly consistency and single-base accuracy.

CHAPTER 1

Installation

- **REQUIREMENT**
  - Python (Support python 2 and 3):
    - Paralleltask

```
pip install paralleltask
```

- **INSTALL**

  click here or use the following command:

```
wget https://github.com/Nextomics/NextDenovo/releases/latest/download/NextDenovo.
↪tgz
tar -vxzf NextDenovo.tgz && cd NextDenovo
```

  If you want to compile from the source, run:

```
git clone git@github.com:Nextomics/NextDenovo.git
cd NextDenovo && make
```

- **TEST**

```
nextDenovo test_data/run.cfg
```

CHAPTER 2

Quick Start

1. Prepare input.fofn

```
ls reads1.fasta reads2.fastq reads3.fasta.gz reads4.fastq.gz ... > input.fofn
```

2. Create run.cfg

```
cp doc/run.cfg ./
```

Note: Please set *read_type* and *genome_size*, and refer to *doc/FAQ* and *doc/OPTION* to optimize parallel computing parameters.

3. Run

```
nextDenovo run.cfg
```

4. Result

   - Sequence: `01_rundir/03.ctg_graph/nd.asm.fasta`
   - Statistics: `01_rundir/03.ctg_graph/nd.asm.fasta.stat`

CHAPTER 3

Getting Help

- **HELP**

  Feel free to raise an issue at the issue page.

  Important: Please ask questions on the issue page first. They are also helpful to other users.

- **CONTACT**

  For additional help, please send an email to huj_at_grandomics_dot_com.

CHAPTER 4

Copyright

NextDenovo is only freely available for academic use and other non-commercial use. For commercial use, please contact GrandOmics.

CHAPTER 5

Cite

We are now preparing the manuscript of NextDenovo, so if you use NextDenovo now, please cite the official website (https://github.com/Nextomics/NextDenovo)

CHAPTER 6

Limitations

1. NextDenovo is optimized for assembly with seed_cutoff >= 10kb. This should not be a big problem because it only requires the longest 30x-45x seeds length >= 10kb. For shorter seeds, it may produce unexpected results for some complex genomes and need be careful to check the quality.

CHAPTER 7

Star

You can track updates by tab the `Star` button on the upper-right corner at the github page.

CHAPTER 8

Assemble the genome of HG002_NA24385_son using NextDenovo

1. **Download reads**

```
wget ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/HG002_
↪NA24385_son/Ultralong_OxfordNanopore/guppy-V2.3.4_2019-06-26/ultra-long-
↪ont.fastq.gz
```

2. **Prepare input file (input.fofn)**

```
ls ultra-long-ont.fastq.gz > input.fofn
```

3. **Prepare config file (run.cfg)**

```
[General]
job_type = sge # here we use SGE to manage jobs
job_prefix = nextDenovo
task = all
rewrite = yes
deltmp = yes
parallel_jobs = 22
input_type = raw
read_type = ont # clr, ont, hifi
input_fofn = input.fofn
workdir = HG002_NA24385_son_assemble

[correct_option]
read_cutoff = 1k
genome_size = 3g # estimated genome size
sort_options = -m 50g -t 30
minimap2_options_raw = -t 8
pa_correction = 5
correction_options = -p 30

[assemble_option]
minimap2_options_cns = -t 8
nextgraph_options = -a 1
```

4. **Run**

```
nohup nextDenovo run.cfg &
```

5. **Get result**

   - Final corrected reads file (use the `-b` parameter to get more corrected reads):

```
HG002_NA24385_son_assemble/02.cns_align/01.seed_cns.sh.work/seed_cns*/
↪cns.fasta
```

   - Final assembly result:

```
HG002_NA24385_son_assemble/03.ctg_graph/nd.asm.fasta
```

   you can get some basic statistical information from file `HG002_NA24385_son_assemble/03.ctg_graph/nd.asm.fasta.stat`, the folowing is the assembly statistics with default parameters:

```
Type            Length (bp)             Count (#)
N10              168924870                    2
N20              127260938                    4
N30               94622851                    7
N40               85456034                   10
N50               79737202                   13
N60               69943198                   17
N70               58504138                   21
N80               40548231                   27
N90               19732879                   36

Min.                 82439                    -
Max.             220056807                    -
Ave.              24389616                    -
Total          2877974703                  118
```

Note: This result will have some minor changes with the version upgrade.

CHAPTER 9

NextDenovo Parameter Reference

NextDenovo requires at least one read file (option: `input_fofn`) as input, it works with gzip'd FASTA and FASTQ formats and uses a `config file` to pass options.

9.1 Input

- `input_fofn` (one file one line)

```
ls reads1.fasta reads2.fastq reads3.fasta.gz reads4.fastq.gz ... > input.fofn
```

- `config file`

  A config file is a text file that contains a set of parameters (key=value pairs) to set runtime parameters for NextDenovo. The following is a typical config file, which is also located in `doc/run.cfg`.

```
[General]
job_type = local
job_prefix = nextDenovo
task = all
rewrite = yes
deltmp = yes
parallel_jobs = 20
input_type = raw
read_type = clr # clr, ont, hifi
input_fofn = input.fofn
workdir = 01_rundir

[correct_option]
read_cutoff = 1k
genome_size = 1g # estimated genome size
sort_options = -m 20g -t 15
minimap2_options_raw = -t 8
pa_correction = 3
```

(continues on next page)

(continued from previous page)

```
correction_options = -p 15

[assemble_option]
minimap2_options_cns = -t 8
nextgraph_options = -a 1
```

9.2 Output

- `workdir/03.ctg_graph/nd.asm.fasta`

  Contigs with fasta format, the fasta header includes ID, type, length, node count, a consecutive lowercase region in the sequence implies a weak connection, and a low quality base is marked with a single lowercase base.

- `workdir/03.ctg_graph/nd.asm.fasta.stat`

  Some basic statistical information (N10-N90, Total size et al.).

9.3 Options

9.3.1 Global options

job_type = sge
: local, sge, pbs, lsf, slurm... (default: sge)

job_prefix = nextDenovo
: prefix tag for jobs. (default: nextDenovo)

task = <all, **correct, assemble>**
: task need to run, correct = only do the correction step, assemble = only do the assembly step (only work if `input_type` = corrected or `read_type` = hifi), all = correct + assemble. (default: all)

rewrite = no
: overwrite existed directory [yes, no]. (default: no)

deltmp = yes
: delete intermediate results. (default: yes)

rerun = 3
: re-run unfinished jobs untill finished or reached `rerun` loops, 0=no. (default: 3)

parallel_jobs = 10
: number of tasks used to run in parallel. (default: 10)

input_type = raw
: input reads type [raw, corrected]. (default: raw)

input_fofn = input.fofn
: input file, one line one file. (**required**)

read_type = {clr, **hifi, ont}**
: reads type, clr=PacBio continuous long read, hifi=PacBio highly accurate long reads, ont=NanoPore 1D reads. (**required**)

workdir = 01.workdir
: work directory. (default: ./)

usetempdir = /tmp/test
temporary directory in compute nodes to avoid high IO wait. (default: None)

nodelist = avanode.list.fofn
a list of hostnames of available nodes, one node one line, used with usetempdir for non-sge job_type.

submit = auto
command to submit a job, auto = automatically set by Paralleltask.

kill = auto
command to kill a job, auto = automatically set by Paralleltask.

check_alive = auto
command to check a job status, auto = automatically set by Paralleltask.

job_id_regex = auto
the job-id-regex to parse the job id from the out of submit, auto = automatically set by Paralleltask.

use_drmaa = no
use drmaa to submit and control jobs.

9.3.2 Correction options

read_cutoff = 1k
filter reads with length < read_cutoff. (default: 1k)

genome_size = 1g
estimated genome size, suffix K/M/G recognized, used to calculate seed_cutoff/seed_cutfiles/blocksize and average depth, it can be omitted when manually setting seed_cutoff.

seed_depth = 45
expected seed depth, used to calculate seed_cutoff, co-use with genome_size, you can try to set it 30-45 to get a better assembly result. (default: 45)

seed_cutoff = 0
minimum seed length, <=0 means calculate it automatically using *bin/seq_stat*.

seed_cutfiles = 5
split seed reads into seed_cutfiles subfiles. (default: pa_correction)

blocksize = 10g
block size for parallel running, split non-seed reads into small files, the maximum size of each file is blocksize. (default: 10g)

pa_correction = 3
number of corrected tasks used to run in parallel, each corrected task requires ~TOTAL_INPUT_BASES/4 bytes of memory usage, overwrite parallel_jobs only for this step. (default: 3)

minimap2_options_raw = -t 10
minimap2 options, used to find overlaps between raw reads, see *minimap2-nd* for details.

sort_options = -m 40g -t 10
sort options, see *ovl_sort* for details.

correction_options = -p 10
correction options, see following:

```
-p, --process, set the number of processes used for correcting.␣
↪(default: 10)
-b, --blacklist, disable the filter step and increase more corrected␣
↪data.
-s, --split, split the corrected seed with un-corrected regions.␣
↪(default: False)
-fast, 0.5-1 times faster mode with a little lower accuracy. (default:␣
↪False)
-dbuf, disable caching 2bit files and reduce ~TOTAL_INPUT_BASES/4 bytes␣
↪of memory usage. (default:False)
-max_lq_length, maximum length of a continuous low quality region in a␣
↪corrected seed, larger max_lq_length will produce more corrected data␣
↪with lower accuracy. (default: auto [pb/1k, ont/10k])
```

9.3.3 Assembly options

minimap2_options_cns = -t 8 -k17 -w17
: minimap2 options, used to find overlaps between corrected reads.

minimap2_options_map = -t 10
: minimap2 options, used to map reads back to the assembly.

nextgraph_options = -a 1
: nextgraph options, see *nextgraph* for details.

CHAPTER 10

Utilities

10.1 seq_stat

seq_stat can be used to perform some simple statistics (such as length distribution, total amount of data and sequencing depth) on the input data, and give the recommended minimum seed length.

INPUT

- read files list, one line one file

OUTPUT (stdout)

- Read length histogram
- Read length info.
- Total Bases info.
- Recommended minimum seed length

OPTIONS

-f	skip reads with length shorter than this value [1kb].
-g	estimated genome size [5Mb].
-d	expected seed depth (30-45), used to be corrected [45].
-a	disable automatic adjustment.
-o	output file [stdout].

10.2 seq_dump

sql_dump is used to classify reads based on a given seed length threshold, and split and compress different categories to subfiles (bit format).

INPUT

- read files list, one line one file

OUTPUT

The output consists of four parts:

```
- input.part*2bit (non-seed reads)
- .input.part*idx (index of non-seed reads)
- input.seed*2bit (seed reads)
- .input.seed*idx (index of seed reads)
```

OPTIONS

-f	minimum read length.
-s	minimum seed length.
-b	block size (Mb or Gb, < 16Gb).
-n	number of seed subfiles in total.
-d	output directory.

10.3 seq_bit

seq_bit can be used to compress fasta files to bit files or uncompress bit files to fasta files.

INPUT

- one seq file.

OUTPUT (stdout)

- sequences with fasta or bit format.

10.4 minimap2-nd

minimap2-nd is a modified version of minimap2, which is used to find all overlaps between raw reads and dovetail overlaps between corrected seeds. Compared to minimap2, minimap-nd has five minor modifications:

```
1. Add support for input files in bit format.
2. Add a filter step for output.
3. Compress output when output to a file.
4. Add a re-align step for potential dovetail overlaps.
5. Optimize overlapping for PacBio Hifi reads.
```

EXTRA OPTIONS

--step <1,2,3>	preset options for NextDenovo, **[required]**.
--minlen INT	min overlap length [500]
--minmatch INT	min match length [100]
--minide FLOAT	min identity [0.05]

--mode <0,1,2>	re-align mode, 0:disable 1:fast mode, low accuracy 2:slow mode, high accuracy [2]
--kn INT	k-mer size (no larger than 28), used to re-align [17]
--wn INT	minizer window size, used to re-align [10]
--cn INT	do re-align for every INT reads, larger is faster [20]
--maxhan1 INT	max over hang length, used to re-align [5000]
--maxhan2 INT	max over hang length, used to filter contained reads [500]
-x ava-hifi	Hifi read overlap

10.5 ovl_sort

ovl_sort is used to sort and remove redundancy overlaps by number of matches for a given seed.

INPUT

- overlap files, one line one file.
- index file of seeds need to be sorted.

OUTPUT

- sorted overlap file.

OPTIONS

-i	index file of seeds need to be sorted **[required]**
-m	set max total available buffer size, suffix K/M/G [40G]
-t	number of threads to use [8]
-k	max depth of each overlap, should <= average sequencing depth [40]
-l	max over hang length to filter [300]
-o	output file name **[required]**
-d	temporary directory [$CWD]

10.6 ovl_cvt

ovl_cvt can be used to compress or uncompress overlap files.

INPUT

- one overlap file

OUTPUT (stdout)

- compressed or uncompressed overlaps

OPTIONS

-m INT	conversion mode (0 for compress, 1 for uncompress)

10.7 nextgraph

NextGraph is used to construct a string graph with corrected reads. The main algorithms are similar to other mainstream assemblers except using a graph-based algorithm to identify chimeric nodes and a scoring-based strategy to identify incorrect edges. It can output an assembly in Fasta, GFA2, GraphML, Path formats, or only statistical information (for quickly optimize parameters).

INPUT

- read files list, one line one file.
- overlap files list, one line one file.

OUTPUT

- assembly statistical information.
- assembly sequences.

OPTIONS

-f FILE input seq list [required]

-o FILE output file [stdout]

-c disable pre-filter chimeric reads

-G retain potential chimeric edges

-k delete complex bubble paths

-A output alternative contigs, for highly heterozygous genomes, it will increase assembly size.

-a, --out_format INT output format, 0=None, 1=fasta, 2=graphml, 3=gfa2, 4=path [1]

-E, --out_ctg_len INT min contig length for output [1000]

-q, --out_spath_len INT min short branch len for output, 0=disable, set 5-16 to adjust the assembly size [0]

-i, --min_ide FLOAT min identity of alignments [0.10]

-I, --min_ide_ratio FLOAT min test-to-best identity ratio [0.70]

-R, --max_ide_ratio FLOAT min test-to-best identity ratio of a low quality edge [0.00]

-S, --min_sco_ratio FLOAT min test-to-best aligned length ratio [0.40]

-r, --max_sco_ratio FLOAT max test-to-best score ratio of a low quality edge [0.50]

-M, --min_mat_ratio FLOAT min test-to-best aligned matches ratio [0.90]

-T, --min_depth_ratio FLOAT min test-to-best depth ratio of an edge [0.60]

-N, --min_node_count <1,2> min valid nodes of a read [2]

-u, --min_con_count <1,2> min contained number to filter contained reads [2]

-w, --min_edge_cov INT min depth of an edge [3]

-D, --bfs_depth INT depth of BFS to identify chimeric nodes [2]

-P, --bfs_depth_multi INT max depth multiple of a node for BFS [2]

-m, --min_depth_multi FLOAT min depth multiple of a repeat node [1.50]

-n, --max_depth_multi FLOAT max depth multiple of a node [2000.00]

-B, --bubble_len INT max len of a bubble [500]

-C, --cpath_len INT max len of a compound path [20]

-z, --zbranch_len INT max len of a z branch [8]

-l, --sbranch_len INT max len of a short branch [15]

-L, --sloop_len INT max len of a short loop [5]

-t, --max_hang_len INT max over hang length of dovetails [500]

-F, --fuzz_len INT fuzz len for trans-reduction [1000]

10.8 bam_sort

bam_sort is used to sort bam files.

INPUT

- bam file need to be sorted.

OUTPUT

- sorted bam file.
- index file.

OPTIONS

-i	Write index file.
-m INT	Set maximum memory per thread; suffix K/M/G recognized [1024M]
-o FILE	Write final output to FILE rather than standard output.
-T PREFIX	Write temporary files to PREFIX.nnnn.bam.

-@ INT Number of additional threads to use [0]

CHAPTER 11

Frequently Asked Questions

11.1 How to optimize parallel computing parameters?

The main parallel computing parameters include `parallel_jobs`, `pa_correction`, `-t` in *minimap2_options_raw*, *minimap2_options_cns* and `-p` in *correction_options*. `parallel_jobs` and `pa_correction` are used to control the number of subtasks running at the same time, `-t` and `-p` are used to control the number of threads/processes used in a single subtask. Each `parallel_jobs` subtask (including *minimap2_options_raw* and *minimap2_options_cns*) requires 32~64 gb RAM depending on the max read length, each `pa_correction` subtask (including *correction_options*) requires ~TOTAL_INPUT_BASES/4 bytes RAM.

1. For an assembly on a local computer with `P` cores and `M` gb memory. A typical configuration file can be set like this (not the best, but better than the default):

```
[General]
job_type = local
parallel_jobs = M/64 #here, 64 can optimize to 32~64
...
```

(continues on next page)

(continued from previous page)

```
[correct_option]
pa_correction = M/(TOTAL_INPUT_BASES * 1.2/4)
sort_options = -m TOTAL_INPUT_BASES * 1.2/4g -t P/pa_correction
correction_options = -p P/pa_correction
minimap2_options_raw = -t P/parallel_jobs
...

[assemble_option]
minimap2_options_cns = -t P/parallel_jobs
...
```

2. For an assembly on a computer cluster with `N` computer nodes and each computer node has `P` cores and `M` gb memory. A typical configuration file can be set like this (not the best, but better than the default):

```
let parallel_jobs_local = M/64 #here, 64 can optimize to 32~64
let pa_correction_local = M/(TOTAL_INPUT_BASES * 1.2/4)
```

```
[General]
job_type = sge
parallel_jobs = parallel_jobs_local * N
...

[correct_option]
pa_correction = pa_correction_local * N
sort_options = -m TOTAL_INPUT_BASES * 1.2/4g -t P/pa_correction_local
correction_options = -p P/pa_correction_local
minimap2_options_raw = -t P/parallel_jobs_local
...

[assemble_option]
minimap2_options_cns = -t P/parallel_jobs_local
...
```

11.2 What's the difference between `nd.asm.p.fasta` and the final assembly result `nd.asm.fasta`?

In theroy, `nd.asm.p.fasta` contains more structural & base errors than `nd.asm.fasta`, you can chose `nd.asm.p.fasta` as the final assembly result, but validate the assembly quality first.

11.3 How to adjust parameters if the assembly size is smaller than the expected genome size?

For highly heterozygous genomes, try to set `nextgraph_options = -a 1 -A`, otherwise you can set `-q` from 5 to 16 in `nextgraph_options`, our tests show that setting `nextgraph_options = -a 1 -q 10` can usually get the best result.

11.4 Which job scheduling systems are supported by NextDenovo?

NextDenovo uses Paralleltask to submit, control, and monitor jobs, so theoretically it supports all Paralleltask-compliant systems, such as LOCAL, SGE, PBS, SLURM.

11.5 How to continue running unfinished tasks?

No need to make any changes, simply run the same command again.

11.6 How to reduce the total number of subtasks?

Please increase blocksize and reduce seed_cutfiles.

11.7 How to speed up NextDenovo?

Currently, the bottlenecks of NextDenovo are minimap2 and IO. For minimap2, please see here to accelerate minimap2, besides, you can increase `-l` to reduce result size and disk consumption. For IO, you can check how many activated subtasks using top/htop, in theory, it should be equal to the `-p` parameter defined in correction_options. Use usetempdir will reduce IO wait, especially if usetempdir is on a SSD driver.

11.8 How to specify the queue/cpu/memory/bash to submit jobs?

See here to edit the Paralleltask configure template file `cluster.cfg`, or use the `submit` parameter.

CHAPTER 12

Assessment of the CHM13 genome (120X NanoPore data) assemblies using NextDenovo, Canu, Flye, Shasta

1. **Download reads**

```
wget https://s3.amazonaws.com/nanopore-human-wgs/chm13/nanopore/rel3/rel3.
↪fastq.gz
```

2. **Prepare input file (input.fofn)**

```
ls rel3.fastq.gz > input.fofn
```

3. **Prepare config file (run.cfg)**

```
[General]
job_type = sge # here we use SGE to manage jobs
job_prefix = nextDenovo
task = all
rewrite = yes
deltmp = yes
parallel_jobs = 25
input_type = raw
read_type = ont # clr, ont, hifi
input_fofn = input.fofn
workdir = chm13_asm

[correct_option]
read_cutoff = 20k
genome_size = 3.1g # estimated genome size
sort_options = -m 150g -t 30
minimap2_options_raw = -t 8
pa_correction = 5
correction_options = -p 30

[assemble_option]
```

(continues on next page)

(continued from previous page)

```
minimap2_options_cns = -t 8
nextgraph_options = -a 1
```

4. **Run**

```
nohup nextDenovo run.cfg &
```

5. **Get result**

   - Final corrected reads file (use the `-b` parameter to get more corrected reads):

```
chm13_asm/02.cns_align/01.seed_cns.sh.work/seed_cns*/cns.fasta
```

   - Final assembly result:

```
chm13_asm/03.ctg_graph/nd.asm.fasta
```

   The folowing is the assembly statistics:

```
Type            Length (bp)             Count (#)
N10             179297054                    2
N20             169128386                    3
N30             131652719                    6
N40             120761272                    8
N50             106090521                   10
N60              95206689                   13
N70              80513393                   16
N80              59725892                   21
N90              39058727                   27

Min.                84432                    -
Max.            237405279                    -
Ave.             35344197                    -
Total          2898224197                   82
```

6. **Download reference**

```
wget https://s3.amazonaws.com/nanopore-human-wgs/chm13/assemblies/chm13.
↪draft_v0.7.fasta.gz
gzip -d chm13.draft_v0.7.fasta.gz
```

7. **Run Quast v5.0.2**

```
quast.py --eukaryote --large --min-identity 80 --threads 30 -r ./chm13.draft_
↪v0.7.fasta --fragmented nd.asm.fasta
```

Quast result

	NextDenovo	Canu	Flye	Shasta
# contigs	82	1223	472	297
Largest contig	237405279	139909728	132009996	130803838
Total length	2898224197	2991947723	2920201070	2823384269
# **misassemblies**	1227	6396	3230	187
# misassembled contigs	61	875	193	78
Misassembled contigs length	2740877545	2458710426	2440399207	1351075153
# **local misassemblies**	433	1164	981	129
# possible TEs	42	160	96	14
# unaligned mis. contigs	11	73	17	0
# unaligned contigs	0 + 64 part	168 + 248 part	8 + 135 part	0 + 37 part
Unaligned length	22021119	30076945	14583673	393547
Genome fraction (%)	97.421	98.391	97.392	96.149
Duplication ratio	1.007	1.027	1.018	1.002
# **mismatches per 100 kbp**	29.43	77.26	74.04	15.56
# **indels per 100 kbp**	170.98	327.08	447.97	141.25
Largest alignment	111497488	104447985	111814657	111679369
Total aligned length	2865321418	2943726417	2894073152	2821352191
N50	106090521	77964612	70319350	58111632
NG50	106090521	77964612	70319350	58088067
L50	10	15	16	17
LG50	10	15	16	18
NA50	57779597	47440498	46858921	47392260
NGA50	57779597	47440498	46546094	44539326
LA50	18	21	19	19
LGA50	18	21	20	20

Note: The results of Canu, Flye and Shasta are copied from here, the complete result of NextDenovo can be seen from `here`.

CHAPTER 13

Assessment of the Arabidopsis thaliana F1 generation of Col-0 and Cvi-0 strains genome (~1% heterozygosity, 192X PacBio CLR reads) assemblies using NextDenovo, Canu, Falcon, Flye, Shasta, Mecat and Wtdbg

1. **Download reads**

```
SRA Accession: SRX1715706, SRX1715705, SRX1715704, SRX1715703
```

2. **Prepare input file (input.fofn)**

```
ls f1.fasta.gz > input.fofn
```

3. **Prepare config file (run.cfg)**

```
[General]
job_type = sge # here we use SGE to manage jobs
job_prefix = nextDenovo
task = all
rewrite = yes
deltmp = yes
parallel_jobs = 12
input_type = raw
read_type = clr # clr, ont, hifi
input_fofn = input.fofn
workdir = 01_rundir

[correct_option]
read_cutoff = 1k
genome_size = 120m # estimated genome size
sort_options = -m 50g -t 35
minimap2_options_raw = -t 20
pa_correction = 6
correction_options = -p 35
```

(continues on next page)

(continued from previous page)

```
[assemble_option]
minimap2_options_cns = -t 20
nextgraph_options = -a 1
```

4. **Run**

```
nohup nextDenovo run.cfg &
```

5. **Get result**

- Final corrected reads file (use the `-b` parameter to get more corrected reads):

```
01_rundir/02.cns_align/01.seed_cns.sh.work/seed_cns*/cns.fasta
```

- Final assembly result:

```
01_rundir/03.ctg_graph/nd.asm.fasta
```

The folowing is the assembly statistics:

Type	Length (bp)	Count (#)
N10	13144176	1
N20	13090493	2
N30	9367478	4
N40	9212899	5
N50	8798661	6
N60	5544810	8
N70	3588034	11
N80	2192782	16
N90	688550	25
Min.	26566	-
Max.	13144176	-
Ave.	1434812	-
Total	126263508	88

6. **Assemble with shasta**

```
shasta-Linux-0.5.1 --input f1.fasta --threads 30
```

7. **Download reference**

```
wget ftp://ftp.arabidopsis.org/home/tair/Genes/TAIR10_genome_release/TAIR10_
↪chromosome_files/TAIR10_chr_all.fas
```

8. **Run Quast v5.0.2**

```
quast.py --large --eukaryote --min-identity 80 --threads 30 -r TAIR10_chr_
↪all.fa nextDenovo.asm.fa Canu.asm.fa Falcon.asm.fa Flye.asm.fa Shasta.asm.
↪fa Mecat.asm.fa Wtdbg.asm.fa
```

Quast result

	NextDenovo	Canu	Falcon	Flye	Shasta	Mecat	Wtdbg
# contigs	88	2107	171	1097	1468	1243	703
Largest contig	13144176	3980575	13319401	4836132	4378421	12631656	14128365
Total length	126263508	229056851	140024465	131553479	143148140	202215921	132890796
N50	8798661	231924	7960654	325940	357597	688687	5479602
NG50	8798661	873036	7979657	370306	560105	3525236	8707235
N75	2323231	69274	1507122	137772	93305	85155	1095469
NG75	3588034	460325	4810976	180227	185928	1096121	2182254
LG50	6	40	6	71	50	8	6
LG75	11	86	10	190	149	22	13
# **misassemblies**	1314	2314	1607	1570	1631	1783	1529
# misassembled contigs	63	383	89	362	357	250	156
# **local misassemblies**	1128	2571	1437	1189	1077	2196	1086
# unaligned mis. contigs	0	8	0	39	79	0	25
# unaligned contigs	13 + 57 part	278 + 511 part	48 + 63 part	27 + 494 part	81 + 528 part	1 + 355 part	253 + 256 part
Unaligned length	5577991	13404835	6336453	4365056	11810280	5760459	12620722
Genome fraction (%)	96.006	99.528	96.938	96.517	97.774	98.166	93.695
Duplication ratio	1.052	1.813	1.154	1.103	1.124	1.675	1.074
# **mismatches per 100 kbp**	668.46	1299.53	822.92	753.04	763.33	1052.95	722.82
# **indels per 100 kbp**	193.40	281.21	127.09	212.74	727.64	338.60	303.37
Largest alignment	5887963	3963652	10477942	4820655	3059195	5451806	7529822
Total aligned length	120235666	214635623	133317043	126764931	131090282	196116682	120017897
NA50	1136416	115341	1459104	280334	255952	202014	756810
NGA50	1504454	539509	1909294	328298	384761	901832	945708
NA75	354228	48301	270481	93990	41634	62905	192079
NGA75	472949	246039	676191	128725	118594	339389	316618
LGA50	21	60	15	82	60	27	27
LGA75	61	140	41	230	202	80	82

Note: the results of Canu, Falcon, Flye, Mecat and Wtdbg are copied from ftp://ftp.dfci.harvard.edu/pub/hli/wtdbg/at-f1, published by wtdbg2 paper, the complete result of Quast can be seen from `here`.

CHAPTER 14

Assessment of the Drosophila melanogaster ISO1 ref. strain genome (69X NanoPore data) assemblies using NextDenovo, Canu, Flye, Shasta and Wtdbg

1. **Download reads**

```
SRA Accession: SRR6702603, SRR6821890
```

2. **Prepare input file (input.fofn)**

```
ls SRR6702603.fasta.gz SRR6821890.fasta.gz > input.fofn
```

3. **Prepare config file (run.cfg)**

```
[General]
job_type = sge # here we use SGE to manage jobs
job_prefix = nextDenovo
task = all
rewrite = yes
deltmp = yes
parallel_jobs = 12
input_type = raw
read_type = ont # clr, ont, hifi
input_fofn = input.fofn
workdir = 01_rundir

[correct_option]
read_cutoff = 1k
genome_size = 130m # estimated genome size
sort_options = -m 30g -t 35
minimap2_options_raw = -t 20
pa_correction = 6
correction_options = -p 35

[assemble_option]
```

(continues on next page)

(continued from previous page)

```
minimap2_options_cns = -t 20
nextgraph_options = -a 1
```

4. **Run**

```
nohup nextDenovo run.cfg &
```

5. **Get result**

- Final corrected reads file (use the `-b` parameter to get more corrected reads):

```
01_rundir/02.cns_align/01.seed_cns.sh.work/seed_cns*/cns.fasta
```

- Final assembly result:

```
01_rundir/03.ctg_graph/nd.asm.fasta
```

The folowing is the assembly statistics:

```
Type            Length (bp)             Count (#)
N10              25701192                    1
N20              22251987                    2
N30              22251987                    2
N40              21195733                    3
N50              21195733                    3
N60              18110856                    4
N70              13648743                    5
N80               6408543                    6
N90               1033518                   12

Min.                18454                    -
Max.             25701192                    -
Ave.              1826448                    -
Total           133330776                   73
```

6. **Assemble with shasta**

```
shasta-Linux-0.5.1  --input SRR6702603.fasta --input SRR6821890.fasta --
↪threads 30
```

7. **Download reference**

```
wget ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/215/GCF_000001215.4_
↪Release_6_plus_ISO1_MT/GCF_000001215.4_Release_6_plus_ISO1_MT_genomic.fna.
↪gz
gzip -d GCF_000001215.4_Release_6_plus_ISO1_MT_genomic.fna.gz
```

8. **Run Quast v5.0.2**

```
quast.py --large --eukaryote --min-identity 80 --threads 30 -r GCF_000001215.
↪4_Release_6_plus_ISO1_MT_genomic.fna nextDenovo.asm.fa Canu.asm.fa Flye.
↪asm.fa Shasta.asm.fa Wtdbg.asm.fa
```

Quast result

	NextDenovo	Canu	Flye	Shasta	Wtdbg
# contigs	73	424	461	872	510
Largest contig	25701192	14715425	12613153	1801407	23221757
Total length	133330776	140540470	135880693	129225244	132926651
N50	21195733	4298595	6016667	535885	12028162
NG50	18110856	4298595	6016667	440773	10631323
N75	13648743	777595	2182645	244480	3308195
NG75	3925274	714013	1367004	182722	1752322
LG50	4	11	9	92	5
LG75	7	36	20	218	13
# **misassemblies**	345	971	724	262	616
# misassembled contigs	48	226	217	78	191
# **local misassemblies**	137	433	670	123	185
# unaligned mis. contigs	1	3	5	7	36
# unaligned contigs	1 + 36 part	8 + 122 part	11 + 118 part	191 + 76 part	89 + 291 part
Unaligned length	603053	769264	811595	1660668	2264882
Genome fraction (%)	92.109	93.614	91.799	88.085	91.504
Duplication ratio	1.011	1.047	1.032	1.016	1.002
# **mismatches per 100 kbp**	90.86	183.12	220.48	609.69	179.86
# **indels per 100 kbp**	567.78	831.54	1334.52	1428.10	1081.15
Largest alignment	25696021	11699048	11981267	1799773	18844039
Total aligned length	132416893	139189216	134650393	127438699	130313270
NA50	6618721	3863099	5596752	527231	4309906
NGA50	6618721	3863099	5143715	434179	4174617
NA75	3269191	670044	1955654	230034	1573933
NGA75	2125978	611559	1267543	168924	928918
LGA50	5	13	11	94	10
LGA75	14	42	24	227	27

Note: The results of Canu, Flye and Wtdbg are copied from ftp://ftp.dfci.harvard.edu/pub/hli/wtdbg/dm-ISO1, published by wtdbg2 paper, the complete result of Quast can be seen from `here`.

CHAPTER 15

Benchmarking data used in NextDenovo Paper

1. Data used in error correction

- **simulated data**
  - ont.raw.fa.gz
  - nextdenovo.cns.fa.gz
  - necat.cns.fa.gz
  - canu.cns.fa.gz
- **actual biological data**
  - canu.cns.fa.gz
  - necat.cns.fa.gz
  - nextdenovo.cns.fa.gz
  - ont.raw.fa.gz

2. Data used in assembly

- **Arabidopsis thaliana**
  - canu.polished.fa
  - canu.raw.fa
  - flye.polished.fa
  - flye.raw.fa
  - necat.polished.fa
  - necat.raw.fa
  - nextdenovo.polished.fa
  - nextdenovo.raw.fa
  - wtdbg.polished.fa

  - wtdbg.raw.fa

- **Drosophila melanogaster**
  - canu.polished.fa
  - canu.raw.fa
  - flye.polished.fa
  - flye.raw.fa
  - necat.polished.fa
  - necat.raw.fa
  - nextdenovo.polished.fa
  - nextdenovo.raw.fa
  - wtdbg.polished.fa
  - wtdbg.raw.fa
- **Oryza sativa**
  - canu.polished.fa
  - canu.raw.fa
  - flye.polished.fa
  - flye.raw.fa
  - necat.polished.fa
  - necat.raw.fa
  - nextdenovo.polished.fa
  - nextdenovo.raw.fa
  - wtdbg.polished.fa
  - wtdbg.raw.fa
- **Zea mays**
  - canu.polished.fa
  - canu.raw.fa
  - flye.polished.fa
  - flye.raw.fa
  - necat.polished.fa
  - necat.raw.fa
  - nextdenovo.polished.fa
  - nextdenovo.raw.fa
  - wtdbg.polished.fa
  - wtdbg.raw.fa

CHAPTER 16

NextDenovo

NextDenovo is a string graph-based *de novo* assembler for long reads (CLR, HiFi and ONT). It uses a "correct-then-assemble" strategy similar to canu (no correction step for PacBio HiFi reads), but requires significantly less computing resources and storages. After assembly, the per-base accuracy is about 98-99.8%, to further improve single base accuracy, try NextPolish.

NextDenovo contains two core modules: NextCorrect and NextGraph. NextCorrect can be used to correct long noisy reads with approximately 15% sequencing errors, and NextGraph can be used to construct a string graph with corrected reads. It also contains a modified version of minimap2 and some useful utilities (see *utilities* for more details).

We benchmarked NextDenovo against other assemblers using Oxford Nanopore long reads from *human* and *Drosophila melanogaster*, and PacBio continuous long reads (CLR) from *Arabidopsis thaliana*. NextDenovo produces more contiguous assemblies with fewer contigs compared to the other tools. NextDenovo also shows a high assembly accurate level in terms of assembly consistency and single-base accuracy.

16.1 Installation

- **REQUIREMENT**
  - Python (Support python 2 and 3):
    - Paralleltask

```
pip install paralleltask
```

- **INSTALL**

  click here or use the following command:

```
wget https://github.com/Nextomics/NextDenovo/releases/latest/download/NextDenovo.
↪tgz
tar -vxzf NextDenovo.tgz && cd NextDenovo
```

  If you want to compile from the source, run:

```
git clone git@github.com:Nextomics/NextDenovo.git
cd NextDenovo && make
```

- **TEST**

```
nextDenovo test_data/run.cfg
```

16.2 Quick Start

1. Prepare input.fofn

```
ls reads1.fasta reads2.fastq reads3.fasta.gz reads4.fastq.gz ... > input.fofn
```

2. Create run.cfg

```
cp doc/run.cfg ./
```

Note: Please set *read_type* and *genome_size*, and refer to *doc/FAQ* and *doc/OPTION* to optimize parallel computing parameters.

---

3. Run

```
nextDenovo run.cfg
```

4. Result
   - Sequence: `01_rundir/03.ctg_graph/nd.asm.fasta`
   - Statistics: `01_rundir/03.ctg_graph/nd.asm.fasta.stat`

16.3 Getting Help

- **HELP**

  Feel free to raise an issue at the issue page.

Important: Please ask questions on the issue page first. They are also helpful to other users.

- **CONTACT**

  For additional help, please send an email to huj_at_grandomics_dot_com.

16.4 Copyright

NextDenovo is only freely available for academic use and other non-commercial use. For commercial use, please contact GrandOmics.

16.5 Cite

We are now preparing the manuscript of NextDenovo, so if you use NextDenovo now, please cite the official website (https://github.com/Nextomics/NextDenovo)

16.6 Limitations

1. NextDenovo is optimized for assembly with seed_cutoff >= 10kb. This should not be a big problem because it only requires the longest 30x-45x seeds length >= 10kb. For shorter seeds, it may produce unexpected results for some complex genomes and need be careful to check the quality.

16.7 Star

You can track updates by tab the `Star` button on the upper-right corner at the github page.

Index

P

R

S

T

U

W