Long read mapping improvements for Flye assembler

Evgeny Polevikov

Scientific advisor: Mikhail Kolmogorov
University of California San Diego
Sequencing technologies

- **Short-read** sequencing technologies (read length is \(\approx 25 - 1100\), error rate is \(\approx 1 - 2\%\)):
  - Illumina
  - Roche 454
  - Ion Torrent

- **Long-read** sequencing technologies (read length is \(\approx 1000 - 200,000\), error rate is \(\approx 10 - 30\%\)):
  - Pacific Biosciences
  - Oxford Nanopore
The primary challenge to all assembly algorithms is repeats. Long reads allow to resolve them more efficiently comparing with shorts reads.

Long-reads assemblers:

- Canu
- Falcon
- Flye
- HINGE
- Miniasm
- ...
Flye (Kolmogorov M, et al. bioRxiv 2018)

minimap2 (Li H, arXiv 2017)

The goal of the project is to incorporate minimap2 into Flye in order to reduce memory usage bottleneck and improve assembly accuracy.
The Flye assembly pipeline

Long reads

Solid kmer selection

Finding read overlaps

Contig assembly

Reconstruction of repeat graph

Repeat resolving

Polishing

Contigs/scaffolds, assembly graph
Finding read overlaps using solid kmers

**Shortcomings** of the current approach:

- Positions of overlaps are less accurate comparing with those we can find using alignment
- An index for the solid kmers takes a lot of memory
- Low flexibility of the parameters choice
Minimap2 approach for finding overlaps

- Minimizer of a sequence is a minimal k-mer in a window of size $w$.
- In order to find overlaps minimap2 constructs a set of minimizers for target and query sequences and finds hits between them.
E. coli dataset: input reads – 203Mb, genome size – 5Mb, coverage – 40x.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Memory (assembly module), Gb</th>
<th>Memory (total), Gb</th>
<th># contigs</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimap2</td>
<td>1.9</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>flye-vertex</td>
<td>2.1</td>
<td>5.6</td>
<td>1</td>
</tr>
<tr>
<td>flye-minimap</td>
<td>2.2</td>
<td>5.6</td>
<td>1</td>
</tr>
</tbody>
</table>
The results obtained

S. cerevisiae dataset: input reads – 367Mb, genome size – 12Mb, coverage – 31x.

<table>
<thead>
<tr>
<th></th>
<th>Memory (assembly module), Gb</th>
<th>Memory (total), Gb</th>
<th># contigs</th>
<th>N50</th>
<th># misassembled contigs</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimap2</td>
<td>3.1</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>flye-vertex</td>
<td>4.9</td>
<td>5.0</td>
<td>33</td>
<td>605k</td>
<td>11</td>
</tr>
<tr>
<td>flye-minimap</td>
<td>3.4</td>
<td>4.7</td>
<td>26</td>
<td>748k</td>
<td>8</td>
</tr>
</tbody>
</table>
The results obtained

X. oryzae dataset: input reads – 1.1Gb, genome size – 4.8Mb, coverage – 220x.

<table>
<thead>
<tr>
<th></th>
<th>Memory (assembly module), Gb</th>
<th>Memory (total), Gb</th>
<th># contigs</th>
<th>N50</th>
<th># misassembled contigs</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimap2</td>
<td>8.0</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>flye-vertex</td>
<td>5.2</td>
<td>10.7</td>
<td>5</td>
<td>4.8M</td>
<td>0</td>
</tr>
<tr>
<td>flye-minimap</td>
<td>8.0</td>
<td>11.8</td>
<td>4</td>
<td>2.8M</td>
<td>2</td>
</tr>
</tbody>
</table>
The algorithm of finding overlaps in Flye was substituted by seed-chain-align procedure using minimap2 API.

But it did not decrease memory usage significantly.

Nonetheless, another ways of improvements are possible: e.g saving of an index to a disk and finding overlaps using the constructed index by parts.