Lectures 20, 21: Single-cell Sequencing and Assembly

Fall 2018
Nov 27, 29, 2018
SINGLE-CELL SEQUENCING AND ASSEMBLY
Single-cell Sequencing

- **Motivation:**
  - Vast majority of environmental bacteria are unculturable outside of their natural habitat.
  - Cell culture may distort genomic information, e.g. cancerous cells.

- **Metagenomics:**
  - Superimposed sequencing of mixed cells of different species in one pool.

- **Single-cell genomics:** sequencing of one DNA molecule from one cell.
Single Cell Genome Sequencing

Start with a single copy of genome.

Amplify (copy only) the genome using multiple displacement amplification (MDA) technique.


Fragment the amplified DNA and sequence reads at both ends.
Multiple Displacement Amplification Video
https://www.youtube.com/watch?v=CaFq9cnfTZI
Sequencing Coverage
Normal multicell vs. single cell

E. coli, Lane normal

E. coli, Lane 1

Green regions are blackout

Number of reads: ~28 million, read length: 100 bp, genome size: 4.6 Mbp, coverage: ~600x

A cutoff threshold will eliminate about 25% of valid data in the single cell case, whereas it eliminates noise in the normal multicell case.

Rescuing Low Coverage Contigs
A quick example

We remove the lowest coverage contig, in blue.
Rescuing Low Coverage Contigs
After error removal

Merged Contig. Coverage = 9
**Velvet vs. Velvet-SC**

**Velvet assembly algorithm**
1: Build a roadmap \(r_dmap\) from \(R\) by indexing all \(k\)-mers.
2: Build a de Bruijn pregraph \(pg\) from \(r_dmap\).
3: Clip tips of \(pg\).
4: Build a graph \(g\) from \(pg\) by threading \(R\).
5: Condense graph \(g\) by merging 1-in 1-out vertices.
6: Clip tips of graph \(g\).
7: Correct graph \(g\) by the Tour Bus algorithm.
8: Remove vertices with average coverage < \(cutoff\).
9: Clip tips of graph \(g\).
10: Correct graph \(g\) by the Tour Bus algorithm.
11: Resolve repeats using read pairing.
12: Condense graph \(g\) by merging 1-in 1-out vertices.
13: Return vertices of graph \(g\) as contigs.

**Our assembly algorithm**

(a) **EULER-SR error correction**

(b) **Velvet-SC assembly algorithm**
1-7: Same as Velvet assembly algorithm.
8: \(\text{for } i = 2 \text{ to } \text{cutoff } \text{do}\)
9: Remove vertices with average coverage < \(i\)
10: Clip tips of graph \(g\).
11: Correct graph \(g\) by the Tour Bus algorithm.
12: Resolve repeats using read pairing.
13: Condense graph \(g\) by merging 1-in1-out vertices.
14: \(\text{end for}\)
15: Return vertices of graph \(g\) as contigs.


E. coli Assembly Results

<table>
<thead>
<tr>
<th>Assembler</th>
<th># contigs</th>
<th>NG50 (bp)</th>
<th>Known genes</th>
<th>Complete genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EULER-SR</td>
<td>1344</td>
<td>26662</td>
<td>4324</td>
<td>3178</td>
</tr>
<tr>
<td>Edena</td>
<td>1592</td>
<td>3919</td>
<td></td>
<td>2425</td>
</tr>
<tr>
<td>SOAPdenovo</td>
<td>1240</td>
<td>18468</td>
<td></td>
<td>3021</td>
</tr>
<tr>
<td>Velvet</td>
<td>428</td>
<td>22648</td>
<td></td>
<td>3055</td>
</tr>
<tr>
<td>E+V-SC</td>
<td>501</td>
<td><strong>32051</strong></td>
<td></td>
<td><strong>3753</strong></td>
</tr>
</tbody>
</table>

NG50 = the contig length at which longer contigs represent half of the total genome length.

### New Genome

*Deltaproteobacteria* single cell assembly results

<table>
<thead>
<tr>
<th>Assembler</th>
<th># of contigs</th>
<th>N50 (bp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Velvet</td>
<td>1856</td>
<td>11531</td>
</tr>
<tr>
<td>E+V-SC</td>
<td>823</td>
<td>30293</td>
</tr>
</tbody>
</table>

N50 = the contig length at which longer contigs represent half of the total *assembly* length.
Efficient *de novo* assembly of single-cell bacterial genomes from short-read data sets

Hamidreza Chitsaz\(^1\), Joyclyn L Yee-Greenbaum\(^2\,\(^6\), Glenn Tesler\(^3\), Mary-Jane Lombardo\(^2\), Christopher I Dupont\(^2\), Jonathan H Badger\(^2\), Mark Novotny\(^2\), Douglas B Rusch\(^4\), Louise J Fraser\(^5\), Niall A Gormley\(^5\), Ole Schulz-Trieglaff\(^5\), Geoffrey P Smith\(^5\), Dirk J Evers\(^5\), Pavel A Pevzner\(^1\) & Roger S Lasken\(^2\)

Whole genome amplification by the multiple displacement amplification (MDA) method allows sequencing of DNA from single cells of bacteria that cannot be cultured. Assembling a genome is challenging, however, because MDA generates highly nonuniform coverage of the genome. Here we describe an algorithm tailored for short-read data from single cells that improves assembly through the use of a progressively increasing coverage cutoff. Assembly of reads from single *Escherichia coli* and *Staphylococcus aureus* cells captures >91% of genes within contigs, approaching the 95% captured from an assembly based
Single-cell Assemblers

- **E+Velvet-SC**

- **SPAdes**

- **IDBA-UD**
Coverage Bias Not Sequence Specific
Combining Multiple MDAs

- Combining DNA from multiple identical single cells, before or after amplification, reduces non-uniformity.

- In practice, combining MDA from 6-12 identical cells gives very high quality assemblies.

- It is hard to pick identical cells before sequencing. Chicken and egg problem.
Synergistic Co-assembly

• **Our solution: co-assembly**
  ◦ N. Movahedi, *et al.*, *BMC Genomics, under review*.

• **Another application of co-assembly: variation detection**
SYNERGISTIC CO-ASSEMBLY
HyDA Single Cell Co-Assembler

- Isolate a number of single cells that are likely to be of the same species. But don’t worry, if they are not, our algorithm will tell you in the end.

- Amplify and sequence each of them individually.

- Assign a unique color to each read dataset.

- Build a colored de Bruijn graph from the colored datasets.  
  J. Simpson, Genome Informatics 2011.

- Iteratively remove errors, condense, and finally output contigs.
Small Toy Example
Shred reads into k-mers (k = 3)

Green Read
G G A C T A A A
G G A
G A C
A C T
C T A
T A A
A A A

Red Read
G A C C A A A T
G A C
A C C
C C A
C A A
A A A
A A T

GGA GAC ACT CTA TAA AAA
(1x) (1x) (1x) (1x) (1x) (1x)

GAC ACC CCA CAA AAA AAT
(1x) (1x) (1x) (1x) (1x) (1x) (1x)

Small Toy Example
Merge vertices labeled by identical k-mers

Green Read:

```
  GGA (1x) → GAC (1x) → ACT (1x) → CTA (1x) → TAA (1x) → AAA (1x)
```

Red Read:

```
  GAC (1x) → ACC (1x) → CCA (1x) → CAA (1x) → AAA (1x) → AAT (1x)
```

Resulting Graph:

```
  GGA (1x) → GAC (1x) → ACT (1x) → CTA (1x) → TAA (1x) → AAA (1x) → AAT (1x)
  ↓                        ↓                        ↓                        ↓
  ACC (1x) → CCA (1x) → CAA (1x) (1x) → AAA (1x) (1x) → AAT (1x) (1x)
```
Co-assembly

- Condensation is done solely based on graph structure, ignoring colorings.

- Maximum colored coverage is used to determine erroneous sequences.
Relationships between Co-assembled Sequences

Exclusive portion: \[ A_i \setminus A_j = \{ a \in A_i | a \notin A_j \} \]

Exclusivity ratio: \[ \frac{|A_i \setminus A_j|}{|A_i|} \]

Assembly size: \[ | \cdot | \]
Alkane-Degrading Bacterial Community

- An alkane-degrading community enriched from sediment from a hydrocarbon-contaminated ditch in Bremen, Germany.

- Consists of 3 species: *Anaerolinea* (2 cells), *Smithella* (6 cells), and *Syntrophus* (2 cells), that have sophisticated metabolic interactions. They cannot be cultured.

- Finished reference genome for a member of *Anaerolinea* and a member of *Syntrophus* is available.

In collaboration with Karsten Zengler and Mallory Embree at UCSD.

M. Embree, *et al.*, *The ISME J.*, 2013

N. Movahedi, *et al.*, BMC Genomics, *under review*
## Co-assembly Results

QUAST results, comparison with the state-of-the-art

<table>
<thead>
<tr>
<th></th>
<th>HyDA</th>
<th>SPAdes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total (bp)</td>
<td>N50</td>
</tr>
<tr>
<td><strong>Syntrophus</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K05</td>
<td>1,265,548</td>
<td>3,782</td>
</tr>
<tr>
<td>C04</td>
<td>465,091</td>
<td>1,928</td>
</tr>
<tr>
<td><strong>Smithella</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEL13</td>
<td>1,590,259</td>
<td>6,977</td>
</tr>
<tr>
<td>MEK03</td>
<td>1,945,701</td>
<td>5,952</td>
</tr>
<tr>
<td>MEB10</td>
<td>1,569,709</td>
<td>5,887</td>
</tr>
<tr>
<td>K19</td>
<td>840,236</td>
<td>7,295</td>
</tr>
<tr>
<td>K04</td>
<td>720,188</td>
<td>5,239</td>
</tr>
<tr>
<td>F16</td>
<td>1,323,536</td>
<td>6,088</td>
</tr>
<tr>
<td><strong>Anaerolinea</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F02</td>
<td>1,352,341</td>
<td>8,201</td>
</tr>
<tr>
<td>A17</td>
<td>260,386</td>
<td>850</td>
</tr>
</tbody>
</table>
## Co-assembly Results

### RAST functional elements

<table>
<thead>
<tr>
<th></th>
<th>HyDA</th>
<th>SPAdes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coding</td>
<td>Coding</td>
</tr>
<tr>
<td></td>
<td>sequence</td>
<td>sequence</td>
</tr>
<tr>
<td></td>
<td>subsystem</td>
<td>subsystem</td>
</tr>
<tr>
<td>Anaerolinea</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A17</td>
<td>212</td>
<td>146</td>
</tr>
<tr>
<td>F02</td>
<td>1,283</td>
<td>1,653</td>
</tr>
<tr>
<td>Smithella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F16</td>
<td>1,197</td>
<td>899</td>
</tr>
<tr>
<td>K04</td>
<td>659</td>
<td>559</td>
</tr>
<tr>
<td>K19</td>
<td>757</td>
<td>581</td>
</tr>
<tr>
<td>MEB10</td>
<td>1,491</td>
<td>1,504</td>
</tr>
<tr>
<td>MEK03</td>
<td>1,856</td>
<td>1,955</td>
</tr>
<tr>
<td>MEL13</td>
<td>1,535</td>
<td>1,435</td>
</tr>
<tr>
<td>Syntrophus</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C04</td>
<td>416</td>
<td>375</td>
</tr>
<tr>
<td>K05</td>
<td>1,216</td>
<td>873</td>
</tr>
</tbody>
</table>
## Co-assembly Results

### Exclusivity ratios (%)

<table>
<thead>
<tr>
<th></th>
<th>Anaerolinea</th>
<th>Smithella</th>
<th>Syntrophus</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A17</td>
<td>F02</td>
<td>F16</td>
</tr>
<tr>
<td>Ana.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A17</td>
<td>0</td>
<td>24</td>
<td>87</td>
</tr>
<tr>
<td>F02</td>
<td>77</td>
<td>0</td>
<td>96</td>
</tr>
<tr>
<td>Smi.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F16</td>
<td>96</td>
<td>96</td>
<td>0</td>
</tr>
<tr>
<td>K04</td>
<td>97</td>
<td>97</td>
<td>49</td>
</tr>
<tr>
<td>K19</td>
<td>98</td>
<td>98</td>
<td>54</td>
</tr>
<tr>
<td>MEB10</td>
<td>96</td>
<td>96</td>
<td>48</td>
</tr>
<tr>
<td>MEK03</td>
<td>97</td>
<td>97</td>
<td>49</td>
</tr>
<tr>
<td>MEL13</td>
<td>97</td>
<td>97</td>
<td>50</td>
</tr>
<tr>
<td>Syn.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C04</td>
<td>44</td>
<td>39</td>
<td>89</td>
</tr>
<tr>
<td>K05</td>
<td>77</td>
<td>75</td>
<td>54</td>
</tr>
</tbody>
</table>
HyDA
Outline

1. Index all distinct k-mers, storing their multiplicities and connections, in a hash table. Each hash node is a self-balancing tree.

2. Construct the condensed de Bruijn graph.

3. Iteratively remove low coverage vertices and recondense.

4. Under development and future work: resolve repeats using long reads and mate pairs.
HyDA

Need for parallelization

- Up to 2 billion reads for a vertebrate genome. Up to several billion vertices in the graph.

- Highly entangled and complex graph.

- Current tools require special large shared memory hardware.

- **Our goal:** reduce the memory footprint so that assembly becomes accessible on publicly available cloud computing environments, e.g. Amazon EC2, etc.
HyDA
Distributing k-mers among processing nodes

- Design a hash function that computes a processing node for every k-mer. More precisely, design

\[ h : K \rightarrow \{1, 2, \ldots, n\} \]

in which \( K \) is the set of k-mers and \( n \) is the number of processors.

- It should provide balance between processing nodes.

- It should not impose communication or memory overhead.
Naïve Approach
A quick example

Let \( h : K \rightarrow \{1, 2, \ldots, n\} \) be \( 1 + (k\text{-mer mod } n) \).
HyDA Approach

- Try to assign adjacent k-mers to the same processing node, as much as possible.

- It is impossible to expect all adjacent k-mers to be assigned to the same processing node.

- Try to keep the balance between processing nodes, i.e. keep memory usage almost equal.
Minimizers
A quick example \((k = 6, m = 2)\)

Minimizer = lexicographically minimum m-mer in the k-mer

\[
\begin{align*}
\text{GGATCC} & \rightarrow \text{GATCCT} & \text{ATCCTC} & \rightarrow \text{TCCTCA} \\
\text{Minimizer:} & \quad \text{AT} \quad \text{AT} \quad \text{AT} & \quad \text{CA} \\
\text{Minimizer hash:} & \quad 1 \quad 1 \quad 1 & \quad 2
\end{align*}
\]

Communication

One single cell ➞ few single cells ➞ the entire sample

- **Goal**: capture every distinct genome present in a microbial sample, even if represented by only one (few) single cell(s).

- **Exploit sparsity**: there are billions of cells, but only thousands distinct genomes/species. A lot of duplicate information.
COMPRESSIVE SINGLE-CELL GENOMICS
Assumptions

Automated microfluidic devices are envisioned that will be capable of high throughput:

- isolation of every individual cell in the sample,
- DNA extraction,
- DNA amplification,
- selective sampling and grouping (and potential barcoding) of amplicons.
Cell Isolation

Input microfluidics channel containing separated single cells

Microfluidics channel containing water droplets in oil
Capture One Cell per Microdroplet
Assumptions

Automated microfluidic devices are envisioned that will be capable of high throughput:

- isolation of every individual cell in the sample,
- DNA extraction,
- DNA amplification,
- selective sampling and grouping (and potential barcoding) of amplicons.
Lysis and DNA Extraction
Assumptions

Automated microfluidic devices are envisioned that will be capable of high throughput:

- isolation of every individual cell in the sample,
- DNA extraction,
- DNA amplification,
- selective sampling and grouping (and potential barcoding) of amplicons.
DNA Amplification

Amplification kit
Primers + Φ29

DNA template

Amplicons
Assumptions

Automated microfluidic devices are envisioned that will be capable of high throughput:

- isolation of every individual cell in the sample,

- DNA extraction,

- DNA amplification,

- selective sampling and grouping (and potential barcoding) of amplicons.
Selective Sampling and Grouping

Sample from droplets 2 and 5 not the entire droplets

Prepare and send for DNA sequencing, e.g. Illumina HiSeq, etc.
Naïve Method

- Recall the goal: capture every distinct genome.
- Exhaustive sequencing and co-assembly of every single cell in the sample: amplify the genome of each cell, sequence, and co-assemble.

Not tractable, because of the cost and duration of sequencing.
Adaptive Divide-and-Conquer

- **Objective:** capture all the genomes while minimizing the cost which depends on
  - the total number of bases required to be sequenced,
  - the number of pools (sampling-and-sequencing runs) needed.

- **Method:** iteratively pool samples of amplicons from different cells, sequence, co-assemble, and compare.
Adaptive Divide-and-Conquer (Example)
Relationships between Co-assembled Sequences (reminder)

\[ A_i \setminus A_j = \{ a \in A_i | a \notin A_j \} \]

Exclusivity ratio: \[ \frac{|A_i \setminus A_j|}{|A_i|} \]

Subsumption: \[ A_i \preceq_\tau A_j \text{ iff } 0 \leq \tau - \frac{|A_i \setminus A_j|}{|A_i|} \]

| \cdot | : assembly size

\( \tau \) : an input parameter
Computational Challenges

- Resource allocation – choosing the required sampling amount from each cell in each round.

- Assembly-assembly comparison – hard to choose a good genome distinction sensitivity parameter $\tau$. Parameters are interdependent.
Squeezambler

- A software to implement both naïve and adaptive divide-and-conquer methods.

- Squeezambler is available at
  http://chitsazlab.org/software/squeezambler/

Simulation Results

- We selected 9 distinct genomes (species) from human gut microbiome and created three scenarios for our simulation. DNA amplification (MDA) and sequencing were simulated by MDAsim and ART software packages.

- MDAsim: A software to simulate MDA process

Simulation Results (cont.)

<table>
<thead>
<tr>
<th>NCBI ID</th>
<th>Name</th>
<th>Ref. Status</th>
<th>Size (bps)</th>
<th>No. of Cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC_004663.1</td>
<td>Bacteroides thetaiotaomicron VPI-5482 chromosome</td>
<td>complete</td>
<td>6.29 M</td>
<td>23</td>
</tr>
<tr>
<td>NC_009614.1</td>
<td>Bacteroides vulgatus ATCC 8482 chromosome</td>
<td>complete</td>
<td>5.16 M</td>
<td>7</td>
</tr>
<tr>
<td>NC_009615.1</td>
<td>Parabacteroides distasonis ATCC 8503 chromosome</td>
<td>complete</td>
<td>4.81 M</td>
<td>8</td>
</tr>
<tr>
<td>NC_008532.1</td>
<td>Streptococcus thermophilus LMD-9</td>
<td>complete</td>
<td>1.86 M</td>
<td>2</td>
</tr>
<tr>
<td>NC_016776.1</td>
<td>Bacteroides fragilis 638R</td>
<td>complete</td>
<td>5.37 M</td>
<td>1</td>
</tr>
<tr>
<td>FP929046.1</td>
<td>Faecalibacterium prausnitzii SL3/3</td>
<td>draft</td>
<td>3.21 M</td>
<td>12</td>
</tr>
<tr>
<td>FP929051.1</td>
<td>Ruminococcus bromii L2-63</td>
<td>draft</td>
<td>2.25 M</td>
<td>35</td>
</tr>
<tr>
<td>FP929053.1</td>
<td>Ruminococcus sp. SR1/5</td>
<td>draft</td>
<td>3.55 M</td>
<td>12</td>
</tr>
<tr>
<td>FP929055.1</td>
<td>Ruminococcus torques L2-14</td>
<td>draft</td>
<td>3.34 M</td>
<td>15</td>
</tr>
</tbody>
</table>
Simulation Results (cont.)

<table>
<thead>
<tr>
<th>NCBI ID</th>
<th>62 cells; 6 distinct genomes</th>
<th>97 cells; 5 distinct genomes</th>
<th>112 cells; 7 distinct genomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC_004663.1</td>
<td>22</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>NC_009614.1</td>
<td>7</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>NC_009615.1</td>
<td>8</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>NC_008532.1</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NC_016776.1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>FP929046.1</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>FP929051.1</td>
<td>22</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>FP929053.1</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>FP929055.1</td>
<td>0</td>
<td>15</td>
<td>15</td>
</tr>
</tbody>
</table>
Conclusions:

- The number of required barcodes with our adaptive divide-and-conquer algorithm is less than that required by the naïve approach,

- The amount of sequencing needed remains the same or decreases.
Assembly Scaffolding and Verification
Using optical mapping

1. Cells are lysed to retrieve genomic DNA

2. Single genomic DNA molecules are placed onto a microfluidic device

3. Restriction enzymes are added to cut the DNA molecules at specific positions

4. Each DNA molecule is stained with a fluorescent dye. An optical map of single-molecules are derived by measuring the fluorescent intensity.

5. Overlapping of the multiple single-molecule maps gives us the consensus genomic optical map
