Genome Sequencing: Introduction to Fragment Assembly

Lecture 5: September 4, 2012
Review from Last Lecture
Sample Preparation

Fragments

Sequencing

Next Generation Sequencing (NGS)

- ACGTAGAATCGACCATG
- GGGACGTAGAATACGAC
- ACGTAGAATACGTAGAA
- GGGACGTAGAATACGAC

Reads
Sample Preparation

Fragments

Sequencing

Reads

Assembly

ACGTAGAATA
ACGTAGAATCGAC
GGGACGTAGAATACGAC

ACGTAGAATAACGTAAGAAACAGATTAGAGAG...

Contigs
“...the ability to determine DNA sequences is starting to outrun the ability of researchers to store, transmit and especially to analyze the data.”

Sample Preparation

Sequencing

Fragments

Reads

Assembly

Contigs

Analysis
Algorithms for Fragment Assembly
Whole Genome Shotgun Sequencing

Genome amplified and sliced into smaller fragments (>=600bp)

Align Contiguous Sequences

Build consensus sequence from overlap
Traditional ("Sanger") Sequencing

- Sequence shotgun fragments of length 600 bp using Sanger sequencing.
- Fragment Assembly is accomplished using "overlap-layout-consensus" approach:
  - **overlap**: matching all possible reads and finding any overlapping.
  - **layout**: finding order of reads along DNA and putting them together.
  - **consensus**: deriving how sequence will appear based on layout.
Overlap-Layout-Consensus Approach

• Build an overlap graph where each node represents a read. An edge exists between two reads if they overlap.

• Traverse the graph to find unambiguous paths which form the contigs.
Problems!

• The main problem with this approach is that it is very, very, very slow and will only work on small genomes or low coverage.

• Not commonly used for complete assembly, however, some software tools still use this approach:
  – **Celera**: genome assembler for 454, PacBio, and Illumina data
  – **LOCAS**: Resequencing genomes.
  – **HapAssembler**: for sequencing highly polymorphic genomes
Problems!

Unfortunately, overlap-layout-consensus approach will **not** work for NGS data or significantly large genomes:

- There is too much data. Calculating the overlap for each pair of reads would take way too much time.
- There has to be a new method for fragment assembly.
De Bruijn Graph Approach to Assembly
De Bruijn Graph for Assembly

• Introduced in 1989.


• Adapted for next generation sequencing data.

De Bruijn Graph Construction

I. Choose a value of $k$. 

II. For each $k$-mer that exists in any sequence create an edge with one vertex labeled as the prefix and one vertex labeled as the suffix. 

III. Glue all vertices that have the same label. 

(Pevzner, Tang & Tesler, 2004)
De Bruijn Graph Construction

(Pevzner, Tang & Tesler, 2004)
De Bruijn Graph Construction

(Pevzner, Tang & Tesler, 2004)
De Bruijn Graph Construction

(Pevzner, Tang & Tesler, 2004)
Challenges in Fragment Assembly

• Repeats in the genome.

```
ACCAGTTGACTGGGATCCTTTTTTAAGACTGGGATTAAAACGCG
CAGTTGACTG
          TGGGATCC
          TGGGATT
```

• Sequencing errors, which vary by platform.

```
TGGGAATT
TGGGACTT
TGGGA--T
TGGGAACTTATT

Substitution
Deletion
Insertion
```

• Size of the data, e.g. 1.5 billion reads.
## De Bruijn Graph of a Genome

**Example Genome:** \( ABCD\ EFG HI \)CDEFGKL

<table>
<thead>
<tr>
<th>( k )-mers</th>
<th>( (k-1) )-mers</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABCD HICD</td>
<td>ABC</td>
</tr>
<tr>
<td>BCDE ICDE</td>
<td>BCD</td>
</tr>
<tr>
<td>CDEF EFGK</td>
<td>CDE</td>
</tr>
<tr>
<td>DEFG FGKL</td>
<td>DEF</td>
</tr>
<tr>
<td>EFGH</td>
<td>EFG</td>
</tr>
<tr>
<td>GHIC</td>
<td>GHI</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
De Bruijn Graph of a Genome

Example Genome: ABCDEFGHICDEFGKLCDEFGHICDEFGK

Graph: ABC → BCD → CDE → DEF → EFG → FGK → GKL
**De Bruijn Graph of a Genome**

*Bulges* (undirected cycles) and *whirls* (directed cycles) occur because of sequencing errors or repeats in the genome.
De Bruijn Graph of a Genome

Example Genome: ABCDEF GHICDEFGKL
Typical De Bruijn Graph

However, this is over a billion vertices (for a very small bacteria genome).
De Bruijn Graph of a Genome

Example Genome: ABCDEFGHICDEFGKLD
De Bruijn Graph of a Genome

Example Genome: \texttt{ABCDEFGHI\texttt{\textup{CDEFGKL}}}

\begin{itemize}
  \item ABC
  \item BCD
  \item CDE
  \item DEF
  \item EFG
  \item FGK
  \item GKL
\end{itemize}
De Bruijn Graph of a Genome

Resulting Erroneous Genome: ABCDEFGKL
Paired-end Reads

- Random fragment with an approximately known size.
- Both ends are sequenced.
- Specified prior to data acquisition.

![Diagram showing paired-end reads with sequences ACTATAAT and ACCGCGAT and an insert size in between.](image)
Standard (Multi-cell) Data

(Chitsaz et al., 2011)
Single-cell Data

Coverage

Genome (Mbp)

(Chitsaz et al., 2011)
Detangling the de Bruijn Graph

Even using mate-pair information, the de Bruijn graph is highly tangled.

There are the following options for detangling the de Bruijn graph:
1. Error correction of reads.
2. Bulge and whirl removal.
Detangling the de Bruijn Graph

Even using mate-pair information, the de Bruijn graph is highly tangled.

There are the following options for detangling the de Bruijn graph:

1. Error correction of reads
2. Bulge and whirl removal

PROBLEM! Both inevitably end-up causing errors rather than correcting them.
Assembly Demonstration
Notes About Demo

• We used Velvet because it’s the simplest to use.
• Write a shell script to run the assembler to keep track of the parameters you used and to avoid writing out the command each time.
• Almost all assemblers require you to specify the following:
  – value of k, whether the data is paired end, insert length, minimum contig length
  – And sometimes, whether it is single-cell data.
Notes About Demo

• How you specify the mate-pair information varies from assembler to assembler. You have to read the manual and write a (perl) script to specify the data in the correct format!
Assemblers can be challenging programs to run. All of them have intricacies even in the installation of the program.

Therefore, running an assembler requires:

1. Some knowledge about Unix/Linux commands.
2. Access to a server with large amounts of memory (64G for small bacteria genomes, 512G for larger genomes).
Notes About Demo

• Be aware that your assembler may not always produce decent results. Can you tell if you did? Yes.
Assembly Evaluation
What has been sequence?

• We’ve sequenced a number of genomes but several genomes remain difficult

• Plant genomes are very hard because they are extremely long, contain huge repeat regions, and are polyploid

• Note: we do not distinguish between genotypes... that is a separate problem
<table>
<thead>
<tr>
<th>Organism</th>
<th>Type</th>
<th>Genome Size</th>
<th>No. of predicted genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homo Sapiens</td>
<td>Human</td>
<td>3.2Gb</td>
<td>20,251</td>
</tr>
<tr>
<td>Takifugu rubripes</td>
<td>Puffer fish</td>
<td>390Mb</td>
<td>22-29,000</td>
</tr>
<tr>
<td>Oryza sativa</td>
<td>Rice</td>
<td>420Mb</td>
<td>32-50,000</td>
</tr>
<tr>
<td>Anopheles gambiae</td>
<td>Mosquito</td>
<td>278Mb</td>
<td>13,700</td>
</tr>
<tr>
<td>Saccharomyces cerevisiae</td>
<td>Baker’s yeast</td>
<td>12.1Mb</td>
<td>6,200</td>
</tr>
<tr>
<td>Cucumis sativus</td>
<td>cucumber</td>
<td>367Mb</td>
<td>27,000</td>
</tr>
</tbody>
</table>

- kb (= kbp) = kilo base pairs = 1,000 bp
- Mb = mega base pairs = 1,000,000 bp
- Gb = giga base pairs = 1,000,000,000 bp.
Assembly Evaluation

• How can we tell the difference between a good assembly and a bad assembly?
  – Answer: **N50 statistic**, which is a metric of the length of a set of sequences, with greater weight given to longer sequences.
  – Given a set of sequences of varying lengths, the N50 length is defined as the length $N$ for which half of all bases in the sequences are in a sequence of length $L < N$.
  – There are some contradictory in the definition(s) of the N50 value.
Calculating N50

Alternative definition: the largest entity E such that at least half of the total size of the entities is contained in entities larger than E.

1. Read Fasta file and calculate sequence length.
2. Sort length on reverse order.
3. Calculate Total size.
Other Evaluations

• Number of insertions, deletions, and substitution errors in an assembly
• misassembly of contigs (chimeric indels)
Other Evaluations

• Number of insertions, deletions, and substitution errors in an assembly
• misassembly of contigs (chimeric indels)
Next Lecture
Detangling the de Bruijn Graph

Even using mate-pair information, the de Bruijn graph is highly tangled.

There are the following options for detangling the de Bruijn graph:

1. Error correction of reads
2. Bulge and whirl removal