Part I. Sequence Analysis: 5. Genome Assembly

Sanger Sequencing (chain-termination method)
- Developed by Frederick Sanger and colleagues in 1977
- The most widely used sequencing method for ~25 years (1980 – mid 2000s)
- Remains in wide use for obtaining long contiguous DNA sequence (>500 nucleotides)

Sequencing Methods

Sanger Sequencing (chain-termination method)
1. Start at primer (restriction site)
2. Grow DNA chain
3. Include dideoxynucleoside (modified a, c, g, t)
4. Stops reaction at all possible points
5. Separate products with length, using gel electrophoresis

DNA Length

A
C
G
T

template strand

primer
Sanger sequence method for genome sequencing?

- Can sequence DNA of length 400–900 nucleotides (nts)
- It takes about 20 mins – 3 hours
  - For human genome, 2 hrs x (3 billion / 500)
- Cost per 1 million nts: $2400
  - For human genome, $2400 x ( 3 billion / 1 million )
- Supplanted by “next-generation” sequencing methods for large-scale sequencing

“Next-Gen” Sequencing (NGS)

- High-throughput sequencing
- Parallelized sequencing process – producing thousands or millions of sequences concurrently
- Major techniques
  - 454 pyrosequencing (454 Life Science & Technology)
  - Solexa sequencing (Illumina)
  - SOLiD sequencing (Applied Biosystems)
  - several more

<table>
<thead>
<tr>
<th>Method</th>
<th>Single-molecule real-time sequencing (Pacific Bio)</th>
<th>Ion semiconductor sequencing (Ion Torrent sequencing)</th>
<th>Pyrosequencing (454)</th>
<th>Sequencing by synthesis (Solexa; Illumina)</th>
<th>Sequencing by ligation (SOLID sequencing)</th>
<th>Sanger sequencing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read length</td>
<td>Up to 500 nts; maximum length=30,000 nts</td>
<td>Up to 400 nts</td>
<td>700 nts</td>
<td>50 to 300 nts</td>
<td>100-200 nts</td>
<td>400 to 900 nts</td>
</tr>
<tr>
<td>Accuracy</td>
<td>99.9% consensus accuracy; 87% single-read accuracy</td>
<td>99.9%</td>
<td>98%</td>
<td>98%</td>
<td>96.9%</td>
<td>99.9%</td>
</tr>
<tr>
<td>Reads per run</td>
<td>~50,000</td>
<td>Up to 80M</td>
<td>1 M</td>
<td>Up to 3 billion</td>
<td>1.2 to 1.4 billion</td>
<td>1</td>
</tr>
<tr>
<td>Time per run</td>
<td>30 mins to 2 hrs</td>
<td>2 hrs</td>
<td>24 hrs</td>
<td>1 to 10 days</td>
<td>1 to 2 weeks</td>
<td>20 mins to 3 hrs</td>
</tr>
<tr>
<td>Cost per 1 M nts</td>
<td>$0.33-1</td>
<td>$1</td>
<td>$10</td>
<td>$0.05-0.15</td>
<td>$0.13</td>
<td>$2400</td>
</tr>
</tbody>
</table>

Whole-Genome Shotgun Sequencing

From Wikimedia Commons: http://en.wikipedia.org/wiki/DNA_sequencing
DNA Sequencing – vectors

DNA fragments

Vector
Circular genome (bacterium, plasmid)

Known location (restriction site)

DNA Sequencing - vectors

Shake

DNA

Method to sequence longer regions

Genomic segment

cut many times at random (Shotgun)

Get one or two reads from each segment

Two reads – mate pairs

~900 bp

~900 bp

Reconstructing the Sequence

Reads

Cover region with high redundancy

Overlap & extend reads to reconstruct the original genomic region

Computational problem

Goal:

Given millions or billions of reads, produce a linear (or perhaps circular) genome

Challenges:

Coverage

Errors in reads

Read length varies from very short (35 nts) to quite long (800 nts)

Non-uniqueness of solution

Running time and memory
**Definition of Coverage**

Length of genomic segment: $G$
Number of reads: $N$
Length of each read: $L$

Definition: Coverage $C = \frac{N \times L}{G}$

How much coverage is enough?

Lander-Waterman model: $\text{Prob[ not covered nt ]} = e^{-C}$
Assuming uniform distribution of reads, $C=10$ results in 1 gapped region /1,000,000 nucleotides

**Repeats**

- **Repeat types:**
  - Low-Complexity DNA (e.g., ATATATATACATA…)
  - Microsatellite repeats $(a_1…a_k)^n$ where $k = 3-6$ (e.g., CAGCAGTAGCAGCACCAG)
  - Transposons
    - SINE (Short Interspersed Nuclear Elements)
      - e.g., ALU: ~300-long, $10^6$ copies
    - LINE (Long Interspersed Nuclear Elements)
      - ~4000-long, 200,000 copies
    - LTR retroelements (Long Terminal Repeats (~700 long) at each end)
      - Cousins of HIV
  - Gene Families
    - Genes duplicate & then diverge (paralogs)
  - Recent duplications ~100,000-long, very similar copies

**Repeats can cause problem**

50% of human DNA is composed of repeats

**What can we do about repeats?**

Two main approaches:
- Cluster the reads
- Link the reads
What can we do about repeats?

**Two main approaches:**
- Cluster the reads
- Link the reads

Resolving repeats – link the reads

3x10^9 nucleotides

ARB, CRD

or

ARB, CRD

or

ARD, PRB?
Strategies for whole-genome sequencing

- Hierarchical shotgun – Clone-by-clone
  - Break genome into many long pieces
  - Map each long piece onto the genome
  - Sequence each piece with shotgun
  - Examples: yeast, worm, human, rat

- Online version of Clone-by-clone
  - Break genome into many long pieces
  - Start sequencing each piece with shotgun
  - Construct map as you go
  - Example: rice genome

- Whole genome shotgun
  - One large shotgun pass on the whole genome
  - Examples: fruit fly, human (celera), neurospora, mouse, rat, dog

Shotgun sequencing is applied to each genome segment.
Whole Genome Shotgun Sequencing

- Genome cut many times at random
- Forward-reverse paired reads
- ~800 bp
- Plasmids (2 – 10 Kbp)
- Cosmids (40 Kbp)
- Known dist

Short-read next generation sequence (NGS) technologies have made the computational challenge harder.

To compensate, assemblers need about eight copies of each piece of the genome.

Steps to Assemble a Genome

1. Find overlapping reads
2. Merge some "good" pairs of reads into longer contigs
3. Link contigs to form supercontigs
4. Derive consensus sequence

Two Standard Categories of NGS Assemblers

- "Overlap-layout-consensus" (OLC) approach
  - Overlap graph
- de Bruijn graph (DBG) approach
  - k-mer graph
  - Especially useful for assembly from short reads
Overlap Graph

- Each read is a node
- A directed edge from reads u to v if the two reads have sufficient overlap
- Objective:
  - Find a Hamiltonian Path (for linear genomes) or a Hamiltonian Circuit (for circular genomes)

Paths through a graph and assembly

- Hamiltonian circuit: visit each node (city) exactly once, returning to the start
- Hamiltonian oath: visit each node exactly once

Example

```
TAATACCTAGGCTAGGAATAAGGCAGCTTTATTGAAATTAAGGCAGTTTAGGACCAATTTAATACGGAT
TAATACCTAGG
TAGCCA
GCCAGGAAT
GCCAATT
GAATAGGCAA
AGGCACGTATA
GGAAATTAGGACCCAC
AATTGGAAT
GGAAATTAGGACCCAC
AGGCACGTATA
```

- If minimum overlap is 3, what graph do we get?
- If minimum overlap is 4, what graph do we get?
- If minimum overlap is 5, what graph do we get?

Say that $k = 4$

```
TAATACCTAGG
TAGCCA
```

```
TAATACCTAGG
GGCAATTAGGACCCAC
AATTGGAAT
GGCAATTAGGACCCAC
AATTGGAAT
```

From http://www.cbcb.umd.edu/research/assembly_primer.shtml
Say that \( k = 4 \)

\[
\text{TAATACTTGG} \\
\text{TAGGCCA} \\
\text{GCCAGGAAT} \\
\text{TAATACTTAGG} \\
\text{GGCCGACCATT} \\
\text{GCCAGGAAT} \\
\text{AGGCAGTITA} \\
\text{GAATAAGCCAA} \\
\text{GCCAATTT} \\
\text{AATTTTGGAAT} \\
\text{GGAATTTAGGCAC} \\
\text{AGGCACGTTA} \\
\text{CACGTTATGGACCATT} \\
\text{GGACCATTTAATACGGAT}
\]

Let’s find the Hamiltonian path

Techniques developed in the field of graph theory (NP-hard but good heuristics exist)

Overlap-Layout-Consensus

- Overlap stage
  - All overlaps between the reads are computed and the graph structure is computed.
Overlap-Layout-Consensus

- **Overlap stage**
  - All overlaps between the reads are computed and the graph structure is computed.

- **Layout stage**
  - Simplify the graph by removing redundant information.

\[
\begin{array}{c}
1 \rightarrow 2 \rightarrow 3 \\
1 \rightarrow 3 \rightarrow 2
\end{array}
\]

- **Consensus stage**
  - Build an alignment of all the reads covering the genome and infers, as a consensus of the aligned reads.

Overlap-Layout-Consensus approach

- The first microbial genome in 1995

- The human genome project in 2001, as well as for all other projects based on Sanger sequencing (longer reads)

- Disadvantages
  - Repeats cause challenges
  - Non-uniqueness
  - Tends to produce fragmented contigs

Challenges

- Computing all-pairs overlap is computationally expensive
  - Especially for NGS datasets, which can have millions of short reads
  - Building an overlap graph for a single run of an Illumina sequencer that generates $10^6$ reads requires a trillion ($10^{12}$) pairwise alignments

- There is no known efficient algorithm for finding a Hamiltonian cycle (path) in a large graph with millions of nodes.

- Need something faster
Two Standard Categories of NGS Assemblers

- “Overlap-layout-consensus” (OLC) approach
  - Overlap graph

- de Bruijn graph (DBG) approach
  - k-mer graph
  - Especially useful for assembly from short reads

k-mer graph
(aka de Bruijn graph)

- Each k-mer is a node

- Each edge is an overlap of (k-1) between k-mers
  - Small values of k produce small graphs

- Does not require all-pairs overlap calculation!
  - Especially effective for next-gen sequencing data producing millions of short reads

- Also produces fragmented assemblies

Example: k-mer graph

<table>
<thead>
<tr>
<th>Sequence</th>
<th>ATGGAATCGCGGAATC</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-mers</td>
<td></td>
</tr>
<tr>
<td>ATGGAAG</td>
<td>ATGGAAG</td>
</tr>
<tr>
<td>TGGAAAG</td>
<td>TGGAAAG</td>
</tr>
<tr>
<td>GGAAGTC</td>
<td>GGAAGTC</td>
</tr>
<tr>
<td>GAAATCG</td>
<td>GAAATCG</td>
</tr>
<tr>
<td>AAGTGGG</td>
<td>AAGTGGG</td>
</tr>
<tr>
<td>ATGTGCCG</td>
<td>ATGTGCCG</td>
</tr>
<tr>
<td>GTCCGGG</td>
<td>GTCCGGG</td>
</tr>
<tr>
<td>TCGCCGA</td>
<td>TCCCGG</td>
</tr>
<tr>
<td>CGCCGAA</td>
<td>CGCCGAA</td>
</tr>
<tr>
<td>GCAGAAT</td>
<td>GCAGAAT</td>
</tr>
<tr>
<td>CGGAATC</td>
<td>CGGAATC</td>
</tr>
</tbody>
</table>

de Bruijn Graphs are Eulerian

- If the k-mer set comes from a sequence and every k-mer appears exactly once in the sequence,

- Then the de Bruijn graph has an Eulerian path!
Find an Eulerian path

- **Eulerian path**: a path that goes through every edge exactly once
- If a graph has an Eulerian path, then all but 2 nodes have even degree. The converse is also true, but a bit harder to prove.

- For directed graphs, the cycle will need to follow the direction of the edges.
  - In this case, a graph has an Eulerian path if and only if the indegree(v)=outdegree(v) for all but 2 nodes (x and y), where indegree(x)=outdegree(x)+1, and indegree(y)=outdegree(y)-1.