Genome Assembly Paradigms

CMSC 423
Carl Kingsford
Shortest Common Superstring

**Def.** Given strings $s_1, \ldots, s_n$, find the shortest string $T$ such that each $s_i$ is a substring of $T$.

- NP-hard (contrast with case when requiring $s_i$ to be subsequences of $T$)
- Approximation algorithms exist with factors: 4, 3, 2.89, 2.75, 2.67, 2.596, 2.5, ...
- Basic greedy method: find pair of strings that overlap the best, merge them, repeat (4 approximation):

  Given match, mismatch, gap costs, how can we compute the score of the best overlap?
Overlap Alignment

Score of an optimal alignment between a suffix of $Y$ and a prefix of $X$

- Initialize first column to 0s
- Answer is maximum score in top row (traceback starts from there until it falls off left side)
Overlap Alignment

Score of an optimal alignment between a suffix of $Y$ and a prefix of $X$

- Initialize first column to 0s
- Answer is maximum score in top row (traceback starts from there until it falls off left side)
K-mer Hashing

Only compute overlap alignment between reads that share a kmer:
The problem with Shortest Common Superstring (SCS): Repeats

**Truth:**

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAA
AAAAAA
AAAAAA
AAAAAA
AAAAAA
AAAAAA
AAAAAA
```

**SCS:**

```
AAAAAA
AAAAAA
AAAAAA
AAAAAA
```

More complex example:

```
ACCGCCT  ACCGCCT  ACCGCCT
```

2 or 3 copies?
Overlap graph:
Nodes = reads
Edges = overlaps

Given overlap graph, how can we find a good candidate assembly?
Overlap graph:
Nodes = reads
Edges = overlaps

Given overlap graph, how can we find a good candidate assembly?
Overlap graph:
Nodes = reads
Edges = overlaps

Given overlap graph, how can we find a good candidate assembly?

Hamiltonian Path (aka Traveling Salesman Path): visit every node in the graph exactly once.
Hamiltonian Path

- Motivation: Every read must be used in exactly one place in the genome.

- Hamiltonian Path is NP-hard.

- Though good solvers exist, they can’t operate on the millions of reads from a sequencing project.

- Solution: greedy walk along the graph.

Assembly via Eulerian Path
**de Bruijn graph**: nodes represent kmers, edges connect k-mers that are known to follow each other based on an observed read.

Can have > 1 edge between nodes.
A directed graph has an Eulerian cycle if and only if:
- All nodes have the same number of edges entering and leaving

Examples

tagacgaacgtacggtagg

tagacgaacgtacggtagg
	agaaccacgacgta
Example bacterial de Bruijn graph

Paths with no branches compressed into a single node

Eulerian path = use every edge exactly once.

With perfect data, the genome can be reconstructed by some Eulerian path through this graph.
Assembly via Eulerian Path

Let $dG(s)$ be the de Bruijn graph of string $s$. Then $s$ corresponds to some Eulerian path in $dG(s)$.

A directed graph has an Eulerian path if and only if:
- One node has one more edge leaving it than entering
- One node has one more edge entering than leaving
- All other nodes have the same number of edges entering and leaving

How can we find such a path?
Connect node with out-degree < in-degree to node with out-degree < in-degree. So that we will have an Eulerian cycle.

Why will you return to $u$?

Walk from some arbitrary node $u$ until you return to $u$, creating a doubly liked list of the path you visit.

Repeat until all edges used:
• Start from some node $w$ on the current tour with unused edges*. 
• Walk along unused edges until you return to $w$, inserting the visited nodes after $w$ into the current tour list.

*How can find such a node quickly?
Eulerian Path Algorithm

Connect node with out-degree < in-degree to node with out-degree < in-degree. So that we will have an Eulerian cycle.

Walk from some arbitrary node $u$ until you return to $u$, creating a doubly liked list of the path you visit.

**Repeat** until all edges used:
- Start from some node $w$ on the current tour with unused edges*.
- Walk along unused edges until you return to $w$, inserting the visited nodes after $w$ into the current tour list.

*How can find such a node quickly?

**Why will you return to $u$?**
The Problem with Eulerian Paths

There are typically an astronomical number of possible Eulerian tours with perfect data.

Adding back constraints to limit # of tours leads to a NP-hard problem.

With imperfect data, there are usually NO Eulerian tours.

Estimating # of parallel edges is usually tricky.

Aside: counting # of Eulerian tours in a directed graph is easy, but in an undirected graph is #P-complete (hard).

(Kingsford, Schatz, Pop, 2010)
Mate Pairs

**Mate pair:** 2 reads, of opposite orientation, separated by an approximately known distance  
⇒ long range information

sequence ≈ 1000 bases from each end

select for a given size

chop up

≈ 1000 bases from each end
Scaffolding

Islands = “contigs”
Scaffolding

Islands = “contigs”
Scaffolding

Islands = “contigs”
Comparative Assembly

Align reads to known genome:

consistent differences = deviation from reference
rare differences = sequencing errors

Can use much lower coverage (e.g. 4X coverage instead of 20-30X for *de novo* assembly).

Aligning a large # of short sequences to one large sequence is an important special case of sequence alignment.
"1000" Genomes Project

find variants that occur in > 1% of the population: sequence ≈2500 genomes at 4X coverage, align them to reference.

<table>
<thead>
<tr>
<th>Population</th>
<th>Status</th>
<th>Available to research community (dates approx)</th>
<th>DNA sequenced from blood</th>
<th>Offspring samples from trios</th>
<th>First set</th>
<th>Second set</th>
<th>Third set</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utah residents (CEPH) with Northern and Western European ancestry (CEU)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>yes</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Toscani in Italia (TSI)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>British from England and Scotland (GBR)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>96</td>
<td>4</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Finnish from Finland (FIN)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Iberian populations in Spain (IBS)</td>
<td>Available to project</td>
<td>Available</td>
<td>no</td>
<td>yes</td>
<td>30</td>
<td>70</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Total European ancestry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>426</td>
<td>74</td>
<td></td>
<td>500</td>
</tr>
<tr>
<td>Han Chinese in Beijing, China (CHB)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Japanese in Toyko, Japan (JPT)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Han Chinese South (CHS)</td>
<td>Available</td>
<td>Available</td>
<td>most</td>
<td>yes</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Chinese Dai in Xishuangbanna (CDX)</td>
<td>Available to project</td>
<td>Oct–Dec 2011</td>
<td>some</td>
<td>no</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Kinh in Ho Chi Minh City, Vietnam (KHV)</td>
<td>Available to project</td>
<td>Oct–Dec 2011</td>
<td>yes</td>
<td>some</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Chinese in Denver, Colorado (CHD) (pilot 3 only)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>TOTAL East Asian ancestry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>300</td>
<td>200</td>
<td></td>
<td>500</td>
</tr>
<tr>
<td>Yoruba in Ibadan, Nigeria (YRI)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>yes</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Luhuya in Webuye, Kenya (LWK)</td>
<td>Available</td>
<td>Available</td>
<td>no</td>
<td>no</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Gambian in Western Division, The Gambia</td>
<td>Collecting samples</td>
<td>Mar–May 2012</td>
<td>no</td>
<td>yes</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>

http://www.1000genomes.org/about#ProjectSamples
Summary

- Sanger sequencing reads DNA via synthesis; 800-1000bp.

- Assembly Paradigms:
  - Shortest Common Superstring (NP-hard; sensitive to repeats)
  - Hamiltonian cycle in overlap graph (NP-hard)
  - Eulerian cycle in de Bruijn graph (polynomial in basic form, but large # of solutions)

- Overlap alignment can be computed with slight variant of sequence alignment DP.
  - K-mer hashing technique avoids all pairs overlap alignment
Hard vs. Easy

- Eulerian path – visit every edge once (easy)
- Hamiltonian path – visit every node once (hard)

- Shortest common supersequence (easy)
- Shortest common superstring (hard)

- Counting Eulerian tours in directed graphs (easy)
- Counting Eulerian tours in undirected graphs (hard)

- Aligning 2 sequences (easy)
- Aligning $k > 2$ sequences (hard)

- Shortest path (easy)
- Longest path (hard)