Genome 373: Genome Assembly

Doug Fowler
What are some of the things we’ve seen we can do with HTS data?
…But We Always Need One Thing!

• After we get the HTS reads, there is a common first step for all these analyses. What is it?

• We’ve just assumed that we were given one critical piece of data. What is it?
Outline

• *De novo* genome assembly introduction

• State-of-the-art assembly with short reads: the De Bruijn graph
Acquiring Data

How would you guys go about acquiring sequencing data for genome assembly?
Reads are the Basic Unit of Assembly

All we start with at the beginning of the assembly process is a read.
Reads are the Basic Unit of Assembly

All we start with at the beginning of the assembly process is a read.

Read length is a key parameter is de novo assembly
Reads are the Basic Unit of Assembly

All we start with at the beginning of the assembly process is a read.

<table>
<thead>
<tr>
<th>Technology</th>
<th>Read Length (nt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sanger</td>
<td>~1,000</td>
</tr>
<tr>
<td>HTS</td>
<td>~100</td>
</tr>
</tbody>
</table>
Assembling a Genome

1. Fragment DNA and sequence

Once we have reads from randomly sheared DNA, what is our next step?
Now we have a problem: gaps between contigs. How can we deal with these? Hint: we have to change our experimental design.
Paired End Reads to Connect Contigs

How would we actually do this?
Contigs are Assembled into Scaffolds

Scaffolds are large units of assembly.
Contigs are Assembled into Scaffolds

Of course, even this strategy won’t be a complete one – what regions are we likely to miss?
Assessing Assembly Quality

How should we assess the quality of our assembly?
Outline

• *De novo* genome assembly introduction

• State-of-the-art assembly with short reads: the De Bruijn graph
Imagine we acquire short reads from a small circular genome
We can represent the traditional assembly process we just talked about as a directed graph where each edge represents the best alignment between two reads.
Using Graphs to Represent Assembly

Walking the graph corresponds to assembling the genome
Breaking Reads into k-Mers

ATGGCGGT

In practice, we break reads into short k-mers to ensure that all k-mers in a genome are represented
Breaking Reads into $k$-Mers

ATGGCGGT

ATG
TGG
GGC
GCG
CGT

$k=3$ mers for the first read in our example
Given a k-mer we define its **suffix** as the string formed by all nucleotides except the first
Breaking Reads into k-Mers

Given a k-mer we define its **prefix** as the string formed by all nucleotides except the last.
A Graph With k-Mers as Nodes

We connect one k-mer to another using a directed edge when the suffix of the first k-mer equals the prefix of the second k-mer.
Using Graphs to Represent Assembly

The assembled genome can be found by visiting each node once and only once.
Using Graphs to Represent Assembly

This is equivalent to the “align all reads to each other and find the optimal assembly” problem
Using Graphs to Represent Assembly

Also known as finding a Hamiltonian cycle, it’s computationally very difficult.
Euler showed that we can find a path that goes through all edges of a graph exactly once, provided that every vertex is equal in in/out-degree.
Euler and the 7 Bridges of Koningsberg

It also turns out that finding a **Eulerian path** through all edges, if it exists, is much less computationally difficult than finding a Hamiltonian path.
Euler and the 7 Bridges of Koningsberg

How can we take advantage of Euler’s observation?
Representing k-Mers as Edges

A prefix and suffix are joined by an edge when they represent an observed k-mer
Representing k-Mers as Edges

Finding an Eulerian path through such a graph gives us the genome assembly

ATG
Finding an Eulerian Path: Hierholzer’s Algorithm

Here is a graph representing the same reads we’ve been working with… we want to find an Eulerian path through the graph
Finding an Eulerian Path: Hierholzer’s Algorithm

1) Start with any node
Finding an Eulerian Path: Hierholzer’s Algorithm

1) Start with any node
2) Walk an arbitrary path of edges back to the start node
Finding an Eulerian Path: Hierholzer’s Algorithm

1) Start with any node
2) Walk an arbitrary path of edges back to the start node
3) If any node has edges not part of the current path, start another walk from that node, following unused edges and returning to the node. Append this second path to the first
Finding an Eulerian Path: Hierholzer’s Algorithm

The algorithm is guaranteed to give an Eulerian path if one exists.
You may have noticed that we could make multiple distinct Eulerian paths for this graph, each of which would correspond to a distinct genome assembly.
This problem arises because, when we converted reads to k-mers we lost linkage information across reads.
Picking the Best Eulerian Path

One of our reads, ATGGCGT, spans the ambiguous region
Picking the Best Eulerian Path

One of our reads, ATGGCGTG, spans the ambiguous region

And we can use it to pick the right path
De novo assembly

Compeau, Pevzner and Tesler, “How to apply de Bruijn graphs to genome assembly.” Nature Biotechnology, 2011


Jones and Pevzner “An introduction to bioinformatics algorithms” Chapter 8
Outline

• *De novo* genome assembly introduction

• State-of-the-art assembly with short reads: the De Bruijn graph