Introduction to Genome Assembly

Tandy Warnow
DNA target sample

SHEAR & SIZE

End Reads / Mate Pairs

550bp

10,000bp

Not all sequencing technologies produce mate-pairs.
Different error models
Different read lengths
Basic Challenge

- Given many (millions or billions) of reads, produce a linear (or perhaps circular) genome
- Issues:
  - Coverage
  - Errors in reads
  - Reads vary from very short (35bp) to quite long (800bp), and genomes are double stranded
  - Non-uniqueness of solution
  - Running time and memory
Simplest scenario

• Reads have no error
• Read are long enough that each appears exactly once in the genome
• Each read given in the same orientation (all 5’ to 3’, for example)
De novo vs. comparative assembly

• *De novo* assembly means you do everything from scratch

• Comparative assembly means you have a “reference” genome. For example, you want to sequence your own genome, and you have Craig Venter’s genome already sequenced. Or you want to sequence a chimp genome and you have a human already sequenced.
Comparative Assembly

Much easier than *de novo*!

Basic idea:

- Take the reads and map them onto the reference genome (allowing for some small mismatch)
- Collect all overlapping reads, produce a multiple sequence alignment, and produce consensus sequence
Comparative Assembly

Fast
Short reads can map to several places (especially if they have errors)
Needs close reference genome
Repeats are problematic
Can be highly accurate even when reads have errors
De Novo assembly

• Much easier to do with long reads
• Need very good coverage
• Generally produces fragmented assemblies
• Necessary when you don’t have a closely related (and correctly assembled) reference genome
De Novo Assembly paradigms

• Overlap Graph:
  overlap-layout-consensus methods
  – greedy (TIGR Assembler, phrap, CAP3...)
  – graph-based (Celera Assembler, Arachne)

• k-mer graph (especially useful for assembly from short reads)
Overlap Graph

• Each read is a node
• There is a directed edge from 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 graphs and assembly

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

A B C D

E F G H

I

Genome
Hamiltonian Path Approach

- Hamiltonian Path is NP-hard (but good heuristics exist), and can have multiple solutions
- Dependency on detecting overlap (errors in reads, overlap length)
- Running time (all-pairs overlap calculation)
- Repeats
- Tends to produce fragmented assemblies (contigs)
Example

Reads:
- TAATACTTTAGG
- TAGGCCA
- GCCAGGAAT
- GAATAAGCCAA
- GCCAATTT
- AATTTGGAAT
- GGAATTAAGGCAC
- AGGCACGTTTA
- CACGTAGGACCATT
- GGACCATTATAATACGGAT

If minimum overlap is 3, what overlap graph do we get?
If minimum overlap is 4, what overlap graph do we get?
If minimum overlap is 5, what overlap graph do we get?
Computing and using Overlap Graphs

To compute the Overlap Graph:
   For all pairs \( x, y \) of reads
   Determine if there is sufficient overlap.
This is expensive, since there are millions (or billions) of reads

Using overlap graphs: Hamiltonian Path, Hamiltonian Cycle, and related problems, are NP-hard.
de Bruijn graph

• Vertices are the prefixes or suffixes (of length k-1) that appear in some k-mer in some read, and directed edges are defined by overlap of k-2 nucleotides.

• Note that every edge v->w implies a k-mer!

• Example: The read ACTAG, with k=4, gives vertices ACT, CTA, and TAG, and edges ACT->CTA->TAG. Note that ACT->CTA implies the 4-mer ACTA.
de Bruijn graph

- Example: The read ACTAG, with k=4, gives vertices ACT, CTA, and TAG, and edges ACT->CTA->TAG. Note that ACT->CTA implies the 4-mer ACTA, and CTA->TAG implies the 4-mer CTAG.

- The read ACTAG has only two 4-mers, ACTA and CTAG.
de Bruijn graph

- Small values of k produce small graphs
- *Does not require all-pairs overlap calculation!*
- But: loss of information about reads can lead to “chimeric” contigs, and incorrect assemblies
- Also produces fragmented assemblies (even shorter contigs)
Eulerian Paths

• An **Eulerian path** is one that goes through every edge exactly once.

• It is easy to see that 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 (also called “arcs”). 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.
de Bruijn Graphs are Eulerian

Theorem: 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!
de Bruijn Graph

• Create the de Bruijn graph for the following string, using \( k=5 \)
  – ACATAGGATTTCAC

• Find the Eulerian path
• Is the Eulerian path unique?
• Reconstruct the sequence from this path
Constructing the de Bruijn Graph

• Create the de Bruijn graph for the following string, using $k=5$
  – ACATAGGATTTCAC

• De Bruijn graph for $k=5$:
  – vertices are 4-mers
  – edges are 5-mers
Constructing the de Bruijn Graph

• Create the de Bruijn graph for the following string, using k=5
  – ACATAGGATTCAC

• Edges (5-mers): ACATA, CATAG, ATAGG, TAGGA, AGGAT, GGATT, GATTC, ATTCA, TTCAC

• Vertices (4-mers, two for each 5-mer): ACAT, CATA, ATAG, TAGG, AGGA, GGAT, GATT, ATTCA, TCAC
Using de Bruijn Graphs

Given: set of k-mers from a DNA sequence

Algorithm:
• Construct the de Bruijn graph
• Find an Eulerian path in the graph
• The path defines a sequence with the same set of k-mers as the original
Using de Bruijn Graphs

Given: set of k-mers from a set of reads for a genome

Algorithm:
• Construct the de Bruijn graph
• Try to find an Eulerian path in the graph
• Use the path to assemble the genome
Using de Bruijn Graphs

Given: set of k-mers from a set of reads for a genome

Algorithm:
• Construct the de Bruijn graph
• Try to find an Eulerian path in the graph
• Use the path to assemble the genome
No matter what

- Because of
  - Errors in reads
  - Repeats
  - Insufficient coverage

  the overlap graphs and de Bruijn graphs generally
  don’t have Hamiltonian paths/circuits or Eulerian
  paths/circuits

- This means the first step doesn’t completely
  assemble the genome
Using de Bruijn Graphs

Genome assembly is not as simple as we’ve described…

…but finding Eulerian paths is still the basis of most genome assembly methods!