60-Minute Discussion Session Plan

Lecture 3: Genome Assembly

Course: BINF301 — Computational Biology
Instructor: Tom Michoel
Date: 21/01/2026
Created with Copilot

0-8 min — Warm-Up

Warm-up prompts:

- "Which part of genome assembly feels the most intuitive to you: OLC, De Bruijn graphs, or error correction?"
- "What slide image from the pre-read stood out as confusing or interesting?"

Instructor note. Use slides 3-4 (overview and "three laws of assembly") to connect warm-up responses to main themes.

8-20 min — Guided Concept Walkthrough

Purpose: Establish a shared understanding before deeper work.

Walkthrough topics:

- What genome assembly is (Slide 3).
- Two algorithms: Overlap-Layout-Consensus (Slides 5-12) vs. De Bruijn Graphs (Slides 13-25).
- Why repeats cause problems (Slide 4).

Guiding prompts:

- "How would you describe the main difference between an OLC graph and a De Bruijn graph?"
- "Why does OLC look for a Hamiltonian path, while De Bruijn graphs use an Eulerian path?"
- "What happens when sequencing is not perfect?" (Slide 18)

Instructor note. Ensure students understand: OLC focuses on reads as nodes (Hamiltonian), DBG focuses on k-mers as edges (Eulerian). Mention Hamiltonian path is NP-hard, Eulerian path is linear-time.

20-40 min — Mini Case / Exercises Block

Purpose: Apply concepts to concrete tasks; deepen intuition.

Exercise 1 — Understanding OLC Graphs (5-7 min)

Use Slides 5-10.

Tasks:

- Identify what nodes and edges represent.
- Locate any transitive edges (Slide 9).
- Discuss how transitive reduction simplifies the graph.

Instructor note. Nodes = reads; edges = overlaps. Transitive edges (A→C when A→B and B→C) are redundant and removed to simplify and avoid false branches in repeats.

Exercise 2 — Hamiltonian vs. Eulerian (5-7 min)

Prompt:

- "In OLC, we try to visit each *read* exactly once. In DBG, we try to traverse each *edge* once. Why does this shift matter computationally?"

Mini-case from Slide 13:

```
ACTTTCTTCTGG
```

Instructor note. Hamiltonian path (NP-hard) vs. Eulerian path (linear-time). Using k-mers avoids explicit all-to-all overlap detection.

Exercise 3 — DBG Error Scenario (8-10 min)

Given genome segment:

```
ATGCTTA
```

Reads (k = 3):

```
ATG, TGC, GCT, CTT, TTA, TGA   <- includes an erroneous k-mer
```

Tasks:

- Build the 3-mer De Bruijn graph (nodes = 2-mers, edges = 3-mers).
- Identify the erroneous k-mer.
- Discuss how high coverage repairs such errors.

Instructor note. TGA does not fit the true sequence path. In the graph it creates an inconsistent branch. High coverage keeps true k-mers high-frequency, while erroneous ones appear once and can be pruned.

Exercise 4 — Choosing OLC vs. DBG (5-7 min)

Prompt:

- "Why do long-read assemblers tend to prefer OLC, while short-read assemblers use DBG?"

Instructor note. Long reads produce fewer, longer nodes → overlaps manageable. Short reads produce millions of reads → OLC infeasible → DBG compresses data into k-mers and is linear in the number of k-mers.

40-55 min — Open Reflection & Deep Dive

Prompts:

- "Which concept was hardest today: overlaps, repeat resolution, Eulerian vs. Hamiltonian paths?"
- "What causes tangles in De Bruijn graphs? Biological repeats? Sequencing errors?"
- "Why are error-correction methods (Slides 27-38) essential before assembly?"

Instructor note. Encourage connections: repeat-induced branching, error-induced tips/bubbles, and how correction tools simplify the graph.

55-60 min — Wrap-Up

Prompts:

- “One key insight from today?”
- “Which algorithm (OLC vs DBG) feels more intuitive to you and why?”
- “What should we revisit next lecture?”

Instructor note. Use answers to adjust pacing and revisit weak spots at the beginning of next session.