Lecture 7: Searching Genomes and Genome Indexing

Student Handout & In-Class Exercises

Course: BINF301 — Computational Biology
Instructor: Tom Michoel
Date: 4/2/2026
Created with Copilot

1 Text Search Basics (Slides 3–6)

Genome search is defined as: **Given a pattern P of length n and a text T of length m, find all occurrences of P in T.**

Examples:

- grep-like text search
- text editors (Ctrl+F)
- mapping reads to a reference genome

Naïve Approach (Slides 4–6): Try every alignment of P in T.

- Worst case: $O(mn)$
- Best case: $O(m)$

2 Boyer–Moore Algorithm (Slides 8–15)

A fast exact matching algorithm using:

- **Bad character rule:** After mismatch, shift pattern so mismatch aligns with rightmost matching character.
- **Good suffix rule:** Reuse matched suffix to shift pattern optimally.

3 Genome Indexing (Slides 17–18)

Indexing preprocesses the text T to support fast lookups for many patterns. Essential for:

- read mapping
- repeated queries on a reference genome

4 k-mer Tables (Slides 19–23)

A basic genome index storing each k-mer and the positions where it appears.

Advantages:

- fast lookup (especially via hashing)

Limitations:

- best when pattern length equals k
- not suitable for all substring lengths

5 Hash Tables (Slides 28–29)

Hash tables store key-value pairs with near-constant lookup time.

Used for:

- k-mer count tables
- presence/absence queries

6 Suffix Trees and Suffix Arrays (Slides 32–38)

6.1 Suffix Trees

Compressed tries of all suffixes. Fast but extremely memory-heavy (on the order of 15 bytes per base).

6.2 Suffix Arrays

A space-efficient alternative:

- store sorted suffix positions
- support fast binary search
- require much less space

7 Burrows–Wheeler Transform (Slides 39–41)

A reversible transform that groups identical characters together, making the text more compressible.

8 FM-index (Slides 44–52)

FM-index = BWT + rank/select + partial suffix array sampling.

Properties:

- extremely low memory usage (about 0.5 bytes/character)
- supports fast backward search

9 Hands-On Exercises with Solutions

9.1 Exercise 1 — Naïve Search Simulation (Slides 3–6)

Text: `ATGATCATGAC` Pattern: `ATG`

Task: List match positions.

9.2 Exercise 2 — Boyer–Moore Skip (Slides 9–12)

Pattern: `ATCGA` Mismatch char: `T`

Task: Compute skip distance using the bad character rule.

9.3 Exercise 3 — Build a 3-mer Table (Slides 19–21)

Text: `GATATAGA`

Task: List all 3-mers and positions.

9.4 Exercise 4 — Suffix Array Binary Search (Slides 36–38)

Suffixes of `GCTA$` in sorted order:

0	`$`
1	`A$`
2	`CTA$`
3	`GCTA$`
4	`TA$`

Task: Does “TA” occur?

9.5 Exercise 5 — BWT Construction (Slides 39–41)

Text: `GAT$`

Task: Build BWT.

9.6 Exercise 6 — FM-index Backward Step (Slides 45–48)

BWT L = `T G $ A A` Pattern = `GA`

Task: Compute initial range for final character `A`.