UNIVERSITÄT DES SAARLANDES

ZBI ZENTRUM FÜR BIOINFORMATIK

Assignment 01

Algorithms for Sequence Analysis

Sven Rahmann, Jens Zentgraf and Johanna Schmitz

14.04.2025, due 21.04.2025 (23:59)

01.1: Naive Pattern Search Algorithm (4 Theory)

Consider

- an alphabet of size 4 $\Sigma = \{A, C, G, T\}$
- with the probabilities ($p_A = 0.22, p_C = 0.28, p_G = 0.27, p_T = 0.23$),
- a random pattern
- of length $m = 7$.

a) What is the expected number of comparisons against a text window for the naive algorithm?
Round at least to the 6th digit.

b) What is the number for $m \to \infty$?
Round at least to the 6th digit.

01.2: Canonical Codes (4 Theory)

Reminder

The canonical code is the **minimum** of the encoding of the DNA k-mer and its reverse complement.

Calculate the canonical codes for the given k-mers.

a) ATTCGCG

b) TAAC

Is the given value a canonical code?
Check for $k = 4$ and $k = 5$.
If yes, what is the corresponding 4-mer/5-mer?

c) 15

d) 109

01.3: Number of distinct DNA molecules (Theory 4P)

1. How many different DNA sequences of length k exist?
2. Prove that on DNA sequences of length k, the following relation $\sim$ is an equivalence relation (symmetry, reflexivity, transitivity):
   - $s \sim t$ iff $s = t$ or $s = \text{revcomp}(t)$
   - revcomp is the function that maps a DNA sequence to its reverse complement.
3. What is the answer to question 1 modulo $\sim$,
   i.e., how many different DNA molecules of length k exist?
   (The answer is more complex than you think. It is best to write down all DNA sequences and their reverse complements once for small $k = 1, 2, 3, 4$)

01.4: Horspool Implementation (Programming 4P)

The Horspool algorithm was discussed in the lecture. If we have a small alphabet $\Sigma = \{A, B\}$ and a long sequence and pattern, it can be useful two compute the shift table based on more than one character.

Example

$p = BAAAAB$

```
shifts
length l: 1       2               3
          A B     AA AB BA BB     AAA AAB ABA ABB BAA BAB BBA BBB
          1 5      1  5  4  5       1   5   4   5   3   5   4   5
```

You can use and adapt the code provided on the lecture slides.

- Implement the support of shift pattern length $l = \{1, 2, \ldots\}$.
  - Add l as a parameter to the `preprocessing` and `search` functions.
  - Adapt the functions to support l.

For the text
$T = ABAABABABABBABABAABBBABACABBBABABBABABAABABCBABC$
and the pattern $P = ABABBABABA$:

- Print the matching positions.
- Count how often is the comparison between pattern and text done for $l \in \{1, 2, 3\}$
- How often is a shift of length n done for $l \in \{1, 2, 3\}$