CS575: Final Project Report

**Project Title: KMP and Boyer-Moore String Matching Algorithms and its comparison using DNA sequencing**

**Team Member(s): Kasturi Vartak**

**/\* Your project will be graded based on the significance of the project and the success and clarity of your project demo, YouTube presentation, and final report. Your final report excluding the attachments and references should not exceed 3 pages. \*/**

/\* **Please organize your report well for clarity, and always check spelling and grammar.** **An excellent style manual for computer science writers is [2]. \*/**

# Problem

KMP(Knuth-Morris-Pratt) Algorithm: KMP is one of the patterns matching algorithm. This algorithm searches ‘pattern’ in a given ‘text’. It compares character by character from left to right and whenever a mismatch occurs, it uses its preprocessed ‘prefix table (LPS)’ to skip character comparison while matching.

# Algorithms

In both algorithms, ‘﻿chr1\_GRCh38.txt’ is a human genome sequence and ‘﻿defected\_chr1\_GRCh38.txt’ is infected human genome sequence. We need to search an occurrence of lamda virus in human genome sequence in ‘﻿lamda\_virus.txt’.

**II.A. KMP**

Preprocessing the pattern: We need to preprocess a pattern string before searching for it in a main string. Preprocessing involves creating lps array corresponding to the pattern string. Length of lps array is always equal to length of pattern string. Lps stands for longest proper prefix which is also a suffix. For example, proper suffixes of “ATG” are “A” and “AT” but not “ATG”.

Searching pattern: The algorithm uses lps array created to search pattern in text and uses the value in lps to decide the next characters to be matched. Basic idea behind KMP is that when we detect a mismatch after some matches in the ‘text’, we know some of the characters in the text for the next window. We take advantage of this information and avoid matching these characters that we know will definitely match unlike brute-force approach.

**II.B. Boyer-Moore**

Similar to KMP, this algorithm also preprocesses the ‘pattern’ to be searched in a “text”. Unlike KMP, it scans matching of the 2 sequences from right to left. This algorithm takes backward approach. It searches from right to left until a mismatch occurs. If there is no mismatch, a pattern has been found in text. Otherwise it calculates the shift, meaning, the number of characters by which pattern is moved to the right.

Whenever mismatch occurs, this means the current location in the text is not the suitable starting place for target pattern.

Good suffix rule for Boyer-Moore algorithm has 2 use cases:

1. Good suffix exists somewhere in the pattern:

Pattern may contain more than one occurrence of sub-pattern, say ‘t’. in this case, we need to shift the pattern to align that occurrence with ‘t’ in ‘text’. But here, we do not need to shift with the same character as new good suffix. Because its already known that there is mismatch left of it.

1. Partial good suffix exists as a prefix of pattern

We cannot find the matched good suffix in the pattern somewhere else. But ‘suffix of suffix’ can be found as prefix of the pattern. In this case, the algorithm shifts the pattern so that the partial suffix aligns with the prefix of the pattern.

# Software Design and Implementation

## Implementation and Tools Used :

## Programming language – Python

## defected\_chr1\_GRCh38.txt : Human genome (Homo sapians)

<http://useast.ensembl.org/Homo_sapiens/Info/Index>

## lamda\_virus.txt – diaease DNA <https://d28rh4a8wq0iu5.cloudfront.net/ads1/data/lambda_virus.fa>

## we search for diseased DNA in Human genome sequence and find the indices at which diseased DNA is present.

## Performance Evaluation (Optional)

The results proposed are described in this section:

We have calculated time required to find the affected by the disease in the human genome with respect to number of occurrences. Following figure shows that KMP algorithm takes longer time than Boyer-Moore algorithm for searching the pattern. Also, time required increases as the number of occurrences of diseased DNA in Human DNA genome increases.

A close up of a map

Description automatically generated

## For KMP algorithm, theory part was referred from this document and programming is done from scratch.

For Boyer-Moore algorithm referred from [2] document.

Attachments

* <https://github.com/kvartak2/pattern_matching>
* Provide the link (URL) to YouTube video here
* Provide the link to your presentation slides here

##### References

[1] <http://www.btechsmartclass.com/data_structures/knuth-morris-pratt-algorithm.html>

[2] <https://www.inf.hs-flensburg.de/lang/algorithmen/pattern/bmen.htm>

[3] <https://d28rh4a8wq0iu5.cloudfront.net/ads1/data/lambda_virus.fa>

[4] <http://useast.ensembl.org/Homo_sapiens/Info/Index>