Faculty of Arts & Sciences
Department of Computer Science
CMPS 297A/396AG – Bioinformatics
Fall 2023 – Course Project
Due 7am November 20th, 2023

Guidelines and Grading Policy

A program that does not compile will receive a **zero**. Make sure you upload your solution to your GitHub repository and maintain proper version control practices (commit your work regularly and meaningfully). Please leave meaningful comments and apply good code readability practices. You are also expected to submit a report explaining your strategy, and to present your project orally. The final project grade will include the report, presentation, and program's correctness. Not every program that runs will be considered correct. Programs should generate better than random results. Your report should document any assumptions made.

Problem Statement

Genome assembly is the precursor to many bioinformatics applications. Genomes are sequenced as short reads, typically a few hundred base pairs long, and then assembled into contigs and full genomes. In some instances, the species of the genome is known and a reference genome can be used as a template in the genome assembly process, which could be more efficient, especially for larger genomes. In this project, you will be given a list of reads of length 100. The list may contain overlapping and repeating reads. You will also be given a reference genome. Your goal is to assemble the reads into a genome that is as close as possible to the reference genome provided. The similarity between the assembled and the reference genomes will be measured using Blast. Your solution should be resource-conscious. It will be graded first on the correctness of the assembled genome and how close it is to the reference, then on how much time and memory it needed to perform the assembly.

Note that:

1. A read may only be chosen from the provided list of reads.
2. A read may exist multiple times in the provided list.
3. A read may not be used more than the number of times it appears in the list.

Write a program that accepts two files (reads.txt, and reference.txt) as input, and produces a file assembled_reads.txt as output. The input file (reads.txt) will contain one read (of length 100) per line. The output file (assembled_reads.txt) should have one assembled contig per line (in other words, if the reads are assembled into a full genome, the output file would contain a single line with that genome sequence).

A sample file is provided. Your solution will be scored on different test cases with varying level of difficulty (different genome sizes, number of repeats, gaps, and overlaps).

Your program should be named $< groupname >$_assembler

Competition

Towards the end of the semester, a competition will be held (participation is optional but strongly encouraged). You will be asked to submit your software package to be tested on data it has not

seen before. A leader-board will be generated and three awards in the form of two bonus points on the course's final grades will be given for:

1. The solution whose assembled genome is most similar to the reference (measured using BLAST)
2. The fastest solution
3. The most memory efficient solution

Only solutions that pass a certain distance score will be considered for the efficiency awards. One team may win more than one award.

Implementation

You may use any language to implement your solution (as long as the Teaching Assistants are familiar with it), and any resources, as long as you cite them, and are able to explain them in your oral presentation at the end of the semester.

In addition to your GitHub submission, add your programs to one folder named as: groupname.genome_assembler.zip (or .rar) and submit it to moodle.