Introduction to Bioinformatics - Project 1

Władysław Olejnik

November 9, 2022

1 Introduction

One of the most important scientific discoveries was DNA and subsequent research that allowed us to understand what a human being is on the molecular level. We learned how the ancestors of every organism impact its next generations. If two organisms share a common ancestor, their DNA should be similar, with differences increasing the former common ancestor's life. Mutations - mismatch, insertion, and deletion - are causing these differences during the process of evolution. Taking the DNA of two organisms, we can try to find regions of similarity and better understand the heritage of living and extinct species. This task is called sequence alignment and lies at the heart of bioinformatics.

2 Methods

This project focuses on implementing the Needleman-Wunsch algorithm for the global alignment problem. It is a dynamic programming algorithm using scoring matrices to find the best alignment by dividing big problems into smaller ones.

To use the algorithm, we need two sequences of length m and n. We initialize score matrix M with $m+1$ columns and $n+1$ rows and a similar direction matrix to find the path corresponding to optimal alignment. This matrix represents all possibilities of aligning those two sequences. We also need a scoring function $S(i,j)$ with score values: if two nucleotides i, j are matching, the score is x; if they are mismatched penalty equals y, and if there is a gap penalty is z. We need to fill the score matrix using formula

$$M_{i,j} = \max\left(M_{i-1,j-1} + S(X_i, Y_j), M_{i,j-1} + S(-, Y_j), M_{i-1,j} + S(X_i, -)\right)$$

We also must remember directions from where the maximum is coming.

After filling the whole matrix, we can trace the path that leads to optimal alignment using those directions. We are starting from the bottom right corner. If the direction is diagonal, we write down corresponding nucleotides from both sequences. We insert a gap in the sequence if the direction is left or top.

As we see algorithm consists of three main steps:

1. Initialization of score matrix and direction matrix
2. Calculating score
3. Tracing back the optimal alignment paths

I use a separate class to handle all the needed methods in my implementation. The first step is achieved using the *init_s_matrix* and *init_dir_matrix* methods in the class constructor. In the second step, I use the *calculate* method, which fills the score matrix using the *s_function* method and default or user-provided score values (*s_dict*). The third step is a recursion-based approach which allows for finding many optimal alignments. It is done using the *find_alignments* method, which uses the *find_path* method.

3 Results and discussion

The obtained results are consistent with the intuition following the input data. Findings are vulnerable to match, mismatch, and gap score values - choosing the correct values is crucial to getting good results.

The implemented algorithm allows us to solve global alignment problems for given sequences. It calculates the score matrix in quite a significant time for long sequences (complexity of mxn matrix). However, returning one (or any other small number) optimal alignment is high-speed. Optimal alignment finding rapidly slows when we try to find every optimal alignment in the colossal matrix, primarily based on sequences with considerable differentiation. The implementation contains calculated scores and methods to print nicely formatted sequences and score matrices as gradient-colored plots.

Major issues of this algorithm are time and memory complexity. There is much room for improvement, but the Needleman-Wunsch algorithm is a powerful and straightforward tool for solving global alignment problems.