Sequence Alignment Games

written by Lisa Wang

Ideal section duration: 70 minutes
Students should bring their computers to section.

Section Breakdown:

1. *Explain DNA strands and purpose of sequence alignment in genetics. (5 min.)*
2. *Let students play Phylo (http://phylo.cs.mcgill.ca/), an online game for multiple sequence alignment. They should answer the questions on the handout. (10 min.)*
3. *Lab. Explain the section problem, which is writing a program that aligns two sequences with the goal to achieve the best alignment. First, discuss with the students how alignments are scored. Students work on their own for 10 minutes. Then, discuss the recursive insight, let students work in pairs for 15 minutes. Students who finish parts a) and b) early, are encouraged to do parts c) and d). (Hints see below)*
4. *Discuss solution with the students (15 minutes). If there is time left, feel free to talk about faster algorithms or bioinformatics.*

Crucial concepts that every student should understand after this section:

- Recursive backtracking: Try out all possibilities
- Basic concept of Big-Oh. Recursion is elegant, but often not the most efficient solution, since some computations are done over and over again!

Why Phylo?

This game aims at improving "multiple sequence alignment," which is one of the most used tools in genomics. Unfortunately, this is a very hard problem and it would take a computer very long to find the best alignment, since it has to try out all possibilities. To speed up computation, current multiple sequence alignment programs use statistical heuristics based on previous matches to avoid testing all possible alignments. However, these heuristics do not guarantee that the program finds the best alignment. Since multiple sequence alignment is so crucial in molecular biology, the online game platform Phylo allows researchers from all over the world to submit their gene sequences to be matched, and utilizes the power of gamers to find better matches than the computer.

Comments on the section problem:

Let matchPts denote the points you get for a single match.
Let mismatchPnlt denote the points deducted for a single mismatch.
Let gapPnlt denote the the points deducted for a single gap. This is also referred to as gap penalty.

Discuss with the students what scorings make sense. For example, the gap penalty has to be higher than the mismatch penalty, because mismatches (= point mutations) are more likely than gaps for biological reasons. We want to avoid gaps! To keep the problem simple, we will use linear gap penalty, meaning that a consecutive series of four gaps decreases the score by 4*gapPnlt.

A plausible scoring model would be:
matchPoints = 1
mismatchPenalty = 1
gapPnlt = 3

Recursive insight: Only look at the first letters of each string. There are three possibilities:
1. Align them. If they match, score increases by matchPts, if they don't, score decreases by mismatchPnlt.
2. Insert a gap in strand1, score decreases by gapPnlt.
3. Insert a gap in strand2, score decreases by gapPnlt.

Use recursion to solve the rest by passing in the remaining strands. Since there are three possibilities, there are three recursive calls.

Base case: If one of the strings is empty, the score is deducted by the length of the other remaining string

Hints for part d):
One way to do this is memoization, which means memorizing intermediate results in a table to avoid doing work (recursive calls) we have already done before.
We could even go one step further and use dynamic programming. (If you are familiar with dynamic programming, feel free to mention it to students who want to be challenged)
Yo can also pitch CS161 Analysis and Design of Algorithms, if students want to learn more about clever algorithms that can solve the same problem a lot faster!