Week 6 Discussion (Sequence Alignment Part 2)
02-604 Fundamentals of Bioinformatics
Phillip Compeau (TA: Saatvik Shah)

Part 1: Discussion Questions

6.3: Demonstrate what the "free taxi rides" should be in the alignment graph for each of the fitting and overlap alignment problems. How do these compare to the local alignment free taxi rides?

6.4: How would we state and solve a "local" version of the affine alignment problem?

Part 2: Discussion Questions
(Special thanks to Steven Skiena: https://www.youtube.com/watch?v=wkrtXDhVgDI*)*

There are several different ways to manufacture vaccines, but one of the best forms of vaccine is by forming it from a weakened (the technical term is "attenuated") form of the virus. By exposing a host to the attenuated virus, the host's immune system builds up antibodies and is not overwhelmed when encountering the "wild type" virus.

How, then, can we attenuate a virus? One way is to iteratively infect monkeys (or the tissue of some other mammal) with the wild type virus, so that as the virus grows accustomed to the other animal's immune system over a series of generations, it also becomes less adept at infecting humans. Yet this approach can be costly, time-consuming, and inhumane.

At the same time, the cost of synthesizing DNA has dropped to the point that synthesizing a viral genome may cost just a few thousand dollars. Can we hope to design a viral genome from scratch that is already attenuated?

We have already learned that the genetic code is *degenerate*, with as many as six RNA codons encoding the same amino acid. As a result, although there is only one way to produce the amino acid sequence encoding a given RNA strand, the number of RNA strings that can encode a given peptide grows very quickly with the length of the peptide. The typical peptide of length n corresponds to on the order of 3^n different RNA strings that can encode this peptide.

The genetic code's degeneracy leads us to a strange question: if two RNA strings encode the same protein sequence, is it possible that one is more virulent than the other? If so, then we could design an attenuated virus by considering all possible sequences of RNA (or DNA, depending on the virus) encoding the same amino acid strand as the virus, and synthesize the one that produces the weakest virus! When we split this problem up into multiple genes, we obtain the following "biological problem".

Attenuated Virus Problem:

Input: An amino acid string *Protein* (corresponding to a viral gene).
Output: The RNA/DNA string translating into *Protein* that is the "weakest".

What is missing from this problem to make it a well-defined computational problem is a clear metric of what it means for an RNA string to be "weak", or more importantly, whether such a metric even exists – this may all be incoherent rambling.

Scientists have noticed that when we examine the percentage of codons used in real genomes, there is a distinctive **codon bias** in favor of or against certain codons *encoding the same amino acid*. For a simple example of codon bias, there are two DNA codons encoding glutamine, `CAA` and `CAG`. Assuming that `A` and `G` are approximately just as frequent, then we might expect for the codons `CAA` and `CAG` to occur about the same number of times in human genes. Yet in the coding regions of the human genome, `CAG` occurs almost three times more often than `CAA`![1] (And a similar pattern is seen in other species.)

Setting possible reasons for codon bias aside for the moment, we can start to see the workings of a metric for the "affinity" that a cell would have for translating an RNA (or DNA) string into an amino acid string. In the simplest case, for a single reading frame, the score assigned to an RNA string *gene* formed of codons c_1, c_2, ..., c_n is just the product of probabilities of each codon,

$$\Pr(gene) = \Pr(c_1) \Pr(c_2) \ldots \Pr(c_n).$$

Codon Bias Attenuation Problem

Input: An amino acid string *Protein.*

Output: The RNA string *s* maximizing Pr(*s*) over all RNA strings encoding *Protein*.

Exercise: What algorithm would you design to solve this problem?

[1] In our dataset of human coding regions, CAA occurs 93,088 times and CAG occurs 259,851 times.

A more advanced question that we could ask is, "What about the transitions *between* codons?" In addition to the cell's preference rate for individual codons, is it possible that there is a **codon pair bias**, meaning that *pairs* of consecutive codons appear more or less often than their frequency in the genome would indicate? If this is the case, then we could attenuate a virus by choosing consecutive pairs of codons that are relatively rare.

To address this question, we must first answer, "What is the *expected* number of occurrences of a given codon pair in a genome, assuming that there is *no* codon pair bias?" If we let (*A*, *B*) be a codon pair encoding the respective amino acids *x* and *y*, the expected number of occurrences of the pair (*A*, *B*) in a DNA string should be some fraction of the number of occurrences of *x* and *y* in the translated protein string, denoted *occ*(*x*, *y*). If the codons are being selected independently (without bias), then the expected fraction of occurrences of *x* and *y* corresponding to the pair (*A*, *B*) must be the frequency with which we observe *A* in the genome, multiplied by the frequency with which we observe *B* in the genome. In other words, the expected number of occurrences of (*A*, *B*) in a bias-free genome would be approximately

$$\frac{occ(A)}{occ(x)} * \frac{occ(B)}{occ(y)} * occ(x, y) \ .$$

To take an example, say that we would like to compute the expected number of (`CGC`, `GAA`) pairs in a codon pair bias-free world; this codon pair encodes the respective amino acids arginine (`R`) and glutamic acid (`E`). There are 29,700 total pairs of (R, E) amino acid pairs in human proteins, out of 424,891 occurrences of `R` and 529,458 occurrences of `E`. By counting 80,155 total occurrences of `CGC` and 226,499 occurrences of `GAA`, we obtain about 2,400 expected occurrences of (`CGC`, `GAA`) pairs:

$$\frac{occ(\texttt{CGC})}{occ(\texttt{R})} * \frac{occ(\texttt{GAA})}{occ(\texttt{E})} * occ(\texttt{R}, \texttt{E}) = \frac{80{,}155}{424{,}891} * \frac{226{,}499}{529458} * 29{,}700 \approx 2{,}397$$

Yet what do we see when we look at real data? In coding regions of the human genome, there are only 268 (`CGC`, `GAA`) pairs! Such a large discrepancy is far from insignificant, and is not isolated; in the opposite direction, we would expect to see only about 2,600 (`CTC`, `TTC`) pairs. But instead, we see almost 6,500! We don't want to get too far into the statistical weeds here, but we will say that these two codon pairs are not isolated examples of codon pair bias in real genomes.

To make things a bit more precise, we can form a metric of how surprised we are by the number of occurrences of a codon pair (*A*, *B*) if we divide the number of actual

occurrences of the pair by the expected number of occurrences in a bias-free genome, which we denote by $w(A, B)$:

$$w(A,B) = \frac{occ(A,B)}{\frac{occ(A)}{occ(x)} * \frac{occ(B)}{occ(y)} * occ(x,y)} .$$

We are now headed toward a computational problem for attenuating a virus by exploiting code pair bias. We simply need to choose successive pairs of codons with low values of this weight. For an RNA string *gene* formed of the codons c_1, c_2, ..., c_n, we can assign a weight to the entire string by forming the product of its codon pair weights,

$$w(gene) = w(c_1, c_2)\ w(c_2, c_3)\ \ldots\ w(c_{n-1}, c_n) .$$

Codon Pair Bias Attenuation Problem

Input: An amino acid string *Protein*.

Output: The RNA string *s* minimizing $w(s)$ over all RNA strings encoding *Protein*.

Exercise: How could we solve this problem via dynamic programming? Is there a simple way that we can change our set up so that this problem can be solved as a Longest Path in a DAG Problem?