COMP 462 / 561 – Fall 2019
Homework #2

Question 1. (30 points)
To answer this question, you will need to use some online tools implementing some of the algorithms seen in class. These might include:
Blast (http://www.ncbi.nlm.nih.gov/BLAST)
and
the LIRMM Phylogenetic inference package
http://phylogeny.lirmm.fr/phylo_cgi/index.cgi

If you have never used Blast before, you may find this tutorial useful: http://www.ncbi.nlm.nih.gov/books/NBK1734/ . In particular, learn what an E-value is, and what the different Blast versions do (Blastn vs MegaBlast vs Blastp vs TBlastN, etc.)

Context: You are doctor and you have a patient suffering from a mysterious infection. You've extracted DNA from the infected area, sequenced it, and obtained the DNA sequence at http://www.cs.mcgill.ca/~blanchem/561/mystery.fa .

a) (15 points) What do you think is the cause of the infection? What is the name of the disease?
Hint: Default parameters for blastn may not result in the identification of very useful hits. Consider changing some of these parameters in order to try to identify hits with good E-values.

Using the default Blastn parameters, we get no hit with a good E-value. To increase the sensitivity of the search, we can reduce the word size from 11 to 7, we get hits for the Zika virus, with a very good E-value (E=9e-26).

b) (15 points) Suppose there exist treatments for various strains of the pathogen. Five known strains exist, with sequences given at:
http://www.cs.mcgill.ca/~blanchem/561/strains.fa
Which treatment (i.e. that for which strain) is the most likely to be appropriate for your patient? Use a phylogenetic inference tool such as http://phylogeny.lirmm.fr/phylo_cgi/index.cgi to figure it out. Explain your answer.

Paste your sequences in the web interface, get the tree, se what species is most closely related to mystery.

Question 2. (30 points)
The first algorithm to calculate the parsimony score of a given set of nucleotides at the leaves of a given rooted binary tree was invented by Fitch and Wagner in 1971. The algorithm works as follows. For each node u of the tree T, define a set X_u as follows:

If u is a leaf then
$X_u = \{x\}$, where x is the nucleotide at leaf u.
Score(u) = 0

If u is an internal node with children v and w, then

If ($X_v \cap X_w = \emptyset$) then
$X_u = X_v \cup X_w$
Score(u) = Score(v) + Score(w) + 1
Else
$X_u = X_v \cap X_w$
Score(u) = Score(v) + Score(w)

After all the X sets have been computed, starting from the leaves back to the root, Score(root) is the desired parsimony score.

Question: Prove that the algorithm always yields the correct (minimal) parsimony score. Perhaps the easiest (but not the only) way to do this is to show that the Fitch algorithm will always produce the same answer as Sankoff's algorithm, which we can assume is a correct algorithm. Try to be as formal as you can in your proof, but if you don't have experience writing mathematical/algorithmic proofs, just explain your reasoning as clearly as possible.

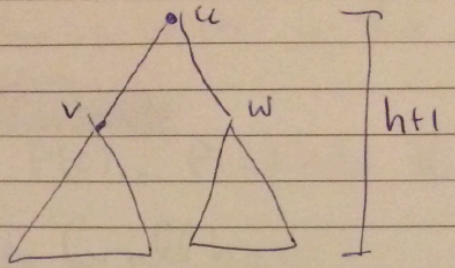
2. We prove the correctness of the algorithm by induction on the height of the tree

Let $P(h)$ be the proposition: In the Fitch algorithm, if node u is the root of a tree of height h, then $score(u) = Parsimony\ Score(u)$ and X_u = set of nucleotides that can be used to label node u and obtain an optimal score.

① Base case: If $h=0$, then the tree consists of a single node u. The algorithm yields $score(u)=0$, and $X_u = \{x\}$, where x is the nucleotide at node u. Both are correct. Thus $P(0)$ holds.

Induction hypothesis: $P(a)$ holds for $0 \le a \le h$

Consider a tree rooted at node u, where the tree has height $h+1$.

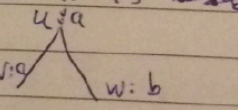


Then the subtrees rooted at nodes v and w both have height $\le h$, so the induction hypothesis applies to them.

If $X_v \cap X_w \neq \emptyset$, then the optimal labeling is obtained by labeling node u with any nucleotide $a \in X_v \cap X_w$, and labeling v and w with the same nucleotide.
This yields a parsimony score of score(v) + score(w)
Thus, the algorithm is correct in that case

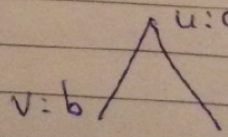
If $X_v \cap X_w = \emptyset$, then the optimal solution is either

① {label u with $a \in X_v$:
label v with a
label w with $b \neq a \in X_w$
→ This yields a solution with score: score(v)+score(w)+1



OR

② {label u with $a \in X_w$
label v with $b \neq a \in X_v$
label w with a
→ This yields a solution with score: score(v)+1+score(w)



Thus, node u can be labeled with any nucleotide from $X_v \cup X_w$, and the score is score(v)+score(w)+1

Thus the algorithm is correct in this case.

Conclusion:
If P(0)... P(h) hold, then P(h+1) holds

$\Rightarrow$ P(h) holds for all $h \geq 0$

Question 3. (40 points)
Phylogenetic trees can be built from non-genetic data, and in fact this was the only type of information available prior to the 1970's, when DNA sequence became possible. Here, you will design and implement an algorithm similar to Sankoff's algorithm, but that will work on quantitative traits (things about a species that can be measured with integers) rather than genetic data. Suppose that you have information about *k* traits for each species. These traits are measured as non-negative integers. For example, for mammals, trait1 might be the length of the forearm (in cm), trait2 might the volume of the skull (in cm^3), trait3 might be the lifespan (in years), etc. As species evolve, their traits change but those changes are generally slow (although there are exceptions). Thus, a parsimony-based phylogenetic inference approach makes sense. The problem we want to solve is the following:

Input:

- A set of *n* species, with, for each species *i*, a vector of *k* integers $D_i = (D_{i,1}, D_{i,2}, \ldots, D_{i,k})$ representing the measurements made for the *k* traits
- A phylogenetic tree *T* with leaves labeled with the *n* species.

Goal:
Assign a vector D_u to each internal node $u = n+1, \ldots, 2n-1$ such that $\sum_{(u,v)\in E(T)} |D_u - D_v|_1$ is minimized, where $|D_u - D_v|_1 = \sum_{i=1}^{k} |D_u(i) - D_v(i)|$.

a) (25 points) Write the pseudocode of algorithm to solve this problem. You can assume that no trait value will ever exceed a given maximum value *M*.
b) (5 points) What is the running time of your algorithm (using big-O notation)?
c) (10 points) The problem formulation above is not ideal in cases where the range of values of different traits differs significantly (e.g. the lifespan ranges between 0 and 50 years, whereas the volume of the brain ranges from 0 to 1500 cm^3). **Explain why and propose a modification to the problem formulation that would make it more biologically relevant. You do not need to propose a revised algorithm to go with your revised problem formulation.**

(3) The objective is to minimize

$$Obj = \sum_{(u,v)\in T} \|D_u - D_v\|_1 = \sum_{(u,v)\in T} \sum_{k=1}^{K} |D_u(i) - D_v(i)|$$

Note that $Obj = \sum_{i=1}^{K} \sum_{(u,v)\in T} |D_u(i) - D_v(i)|$

$= \sum_{i=1}^{K} \text{ParsimonyScore}(i)$

Thus we only need to figure out how calculate the ParsimonyScore of a ~~given position~~ single position of D at a time.

To calculate the parsimony score for position i Mimicking the Sankoff algorithm, let

$S_u[n]$ be the parsimony score of the subtree rooted at u, if u is labeled with integer $n \leq M$

Then

$$S_u[n] = \begin{cases} 0 & \text{if } u \text{ is a leaf and } D_u[i] = n \\ +\infty & \text{if } u \text{ is a leaf and } D_u[i] \neq n \\ \min_{n' \leq M} \{S_v[n'] + |n' - n|\} + \min_{n' \leq M} \{S_w[n'] + |n' - n|\} & \text{if } u \text{ is not a leaf} \end{cases}$$

The rest of the algorithm proceeds exactly like the Sankoff algorithm. The score of the optimal solution is $\min_{n \le M} \{X_{root}[n]\}$

and the optimal labeling is obtained by tracing back the solution from there.

b) The running time for a tree with n leaves is

$O(K \cdot M^2 \cdot n)$

c) The formulation of the problem is not ideal because different traits have different ranges.
For example two species with life span of 5 and 50 years respectively are very different from each other, whereas two species with brain volume 1005 and 1050 cm^3 are actually quite similar. The current formulation will give too much weight to traits with large values.

One solution would be to assign different weights to each trait, to optimize

$$Obj = \sum_{i=1}^{n} \sum_{(u,v) \in T} |D_u(i) - D_v(i)| \cdot w(i)$$

where $w(i)$ would be inversely proportional to the variance of trait i.

Good luck!