CS195Z: Problem Set 1
Solution Key

Due: Wed, 2/4/09
Questions: email Crystal (clkahn)

February 23, 2009

1. A rooted binary tree is a tree in which every vertex has either 0 or 2 children. Prove that for a binary tree with n leaves, there are $n-1$ internal vertices.

   - There are many possible ways to prove the statement. Here is one. Proof by induction. The statement is true for $n = 1$ (there are 0 internal nodes in a binary tree with only 1 leaf – the entire tree consists of one node that is a leaf). Suppose the statement is true for trees with $k > 1$ leaves. To construct a tree with $k+1$ leaves, we can take a tree with k leaves and add 2 children to one of its existing leaves (thus turning one leaf into an internal node) resulting in a net gain of 1 leaf. In this process, we changed one leaf node into an internal node, increasing the number of internal nodes by 1. So, we now have a tree with $k+1$ leaves and k internal nodes.

2. An unrooted binary tree is one in which every vertex either has degree 1 or 3. Let $u(n)$ be the number of unrooted binary trees with n leaves. Give a formula for $r(n)$, the number of rooted binary trees with n leaves, in terms of $u(n)$.

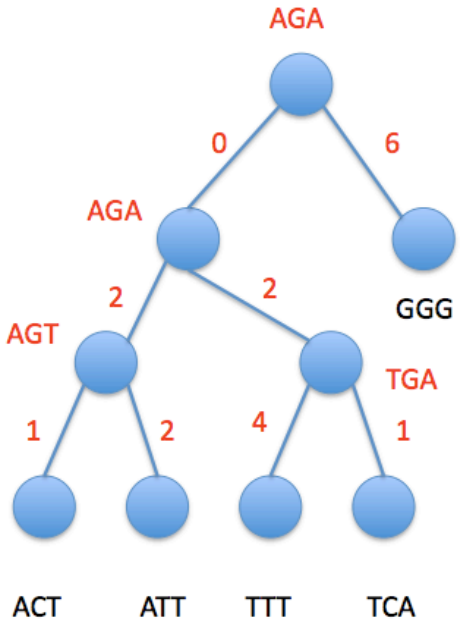
   - We can take an unrooted binary tree and transform it into a rooted binary tree by placing a new root node in the middle of any edge, increasing the number of nodes by 1 and the number of edges by 1. Note that the root of a rooted binary tree has total degree 2. In an unrooted binary tree with n leaves, there are $n-2$ internal nodes, for a total of $2n-2$ nodes and, therefore, $m = 2n-1$ edges. Therefore, we can take any unrooted binary tree and transform it into a rooted binary tree by placing a root node on any of its $m = 2n-1$ edges. Therefore, $r(n) = u(n) * (2n-1)$.

3. An unrooted ternary tree is one in which all vertices have either degree 1 or 4. If there are m internal vertices in an unrooted ternary tree, how many leaves are there and how many edges?

   - leaves $= 2m+2$ and edges $= m +$ leaves -1

4. Using Sankoff's algorithm and the following scoring matrix, find an optimal labeling of the following phylogenetic tree. List the optimal costs on edges.

   - Because I didn't specify, we shall accept the costs on edges to reflect either the cost between the two nodes on the endpoints of that edge (as shown in the figure) or to reflect the entire cost of the subtree rooted at the child node adjacent to that edge.

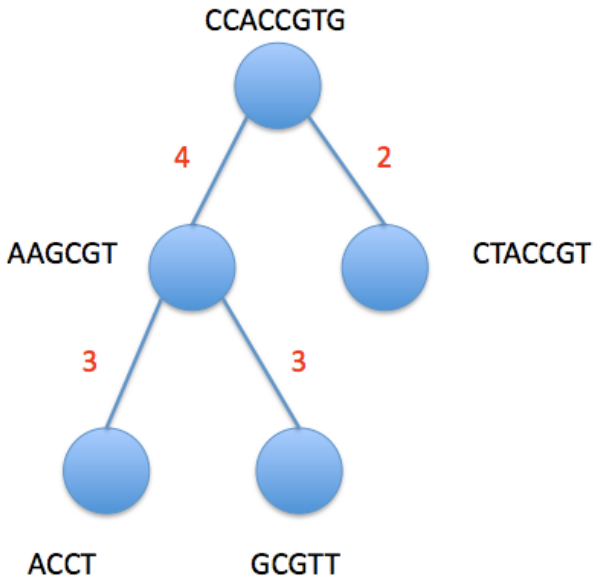


	A	C	G	T
A	0	7	3	2
C	1	0	8	4
G	5	1	0	2
T	6	4	3	0

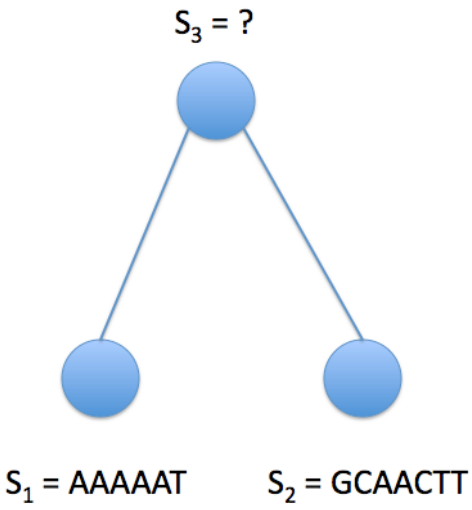
Note: box (row i, col j) is cost for character i to change to character j

5. Edit distance is a metric between strings. The edit distance between two string is the minimum number of individual character insertions, deletions, and changes needed to transform 1 string into the other. For example, the distance between $S_1 = AAATCCGT$ and $S_2 = ACCGAGG$ is 6. ($S_1 = AAATCCGT$. Delete 'A' twice yields $ATCCGT$. Delete 'T' yields $ACCGT$. Change 'T' to 'A' yields $ACCGA$. Insert 'G' twice yields $ACCGAGG = S_2$.) Edit distance can be used as a measure of similarity between genes or any other biological sequences. In class, you learned about character-based parsimony methods for computing phylogenetic tress in which every vertex in the tree represents a string of fixed length and the cost on an edge in the tree corresponds to either the total number of character disagreements between the endpoints of the edge or some weighted version of the number of character disagreements (given by a scoring matrix). Alternatively, we could assign costs to edges using a different distance function between strings.

   (a) Using edit distance as the cost function, what is the parsimony score of the following tree? Label the edges with costs.



(b) If we are given 2 sequences S_1 and S_2, a *median* sequence with respect to edit distance is a third sequence S_3 that minimizes the sum $edit(S_1, S_3) + edit(S_2, S_3)$. List all optimal median strings S_3 for the two strings given below.



- Final list. 32 total.
  AAAAAT
  AAAACT
  AAAATT
  AAACAT
  AAACTT
  ACAAAT
  ACAACT
  ACAATT
  CAAAAT

CAAATT
CAACAT
CAACTT
GAAAAT
GAAACT
GAAATT
GAACAT
GAACTT
GCAAAT
GCAACT
GCAATT
AAAAATT
AAAACTT
ACAAATT
ACAACTT
GAAAAAT
GAAAATT
GAAACAT
GAAACTT
GCAAAAT
GCAAATT
GCAACAT
GCAACTT

6. (BONUS) In class, we made a distinction between 2 phylogeny problems. The first was: given a tree, a set of leaf vertices, and a cost function, find a labeling of the internal vertices that minimizes the total cost of the tree. The second was: given a set of leaf vertices and a cost function, find a tree and a labeling of the internal vertices that minimizes the total cost of the tree. We mentioned that this second problem is "hard." How hard is it (in a complexity sense)? A one-sentence proof is sufficient.

   - It is NP-Hard. Unweighted metric Steiner tree can be reduced to the large parsimony problem in polynomial time.