Problems for 394C – Algorithms for Computational Biology

Problems are numbered by the section and subsection from which they are derived, followed by an integer in parentheses. For example, the sixth problem from section 3.2 would be numbered 3.2(6).

1 Problems from Section 1: Introduction

Problem 1.1(1) Suppose T is a rooted binary tree on n leaves. Give a linear-time algorithm that returns a matrix indexed by the nodes of v the number of leaves in the subtree of T rooted at v. (Hint: think dynamic programming or recursion.) Prove that your algorithm uses linear time.

Problem 1.1(2) Let T be an unrooted tree on n leaves that are labelled $s_1, s_2, \ldots, s_n$, and let the edges of T all have unit length. Give an algorithm to compute the matrix of pairwise distances in T. Analyze the running time.

Problem 1.1(3) Let A and B be two binary sequences. Give an algorithm to determine if A is a substring of B. Analyze the running time.

Problem 1.1(4) Let A and B be two binary sequences. Give an algorithm to determine if A is a subsequence of B. Analyze the running time.

Problem 1.1(5) Let $G = (V, E)$ be a graph. Give an algorithm to determine if G has a clique of size 3. Analyze the running time.

Problem 1.1(6) Let $G = (V, E)$ be a graph and k be a positive integer. Give an algorithm to determine if G has a clique of size k. Analyze the running time.

Problem 1.1(7) Give an example of a simple undirected graph with at most 10 vertices and exactly two components.

Problem 1.1(8) Give an example of a simple undirected graph with the following degree sequence: 0,0,1,1,2,2,2.

Problem 1.1(9) Prove that the number of vertices of odd degree in any graph is always even.

Problem 1.1(10) Prove or disprove: every connected graph with maximum degree two has an Eulerian path or cycle.

Problem 1.1(11) Prove or disprove: every connected graph with maximum degree three has an Eulerian path or cycle.

Problem 1.1(12) Prove or disprove: every connected graph with maximum degree two has a Hamiltonian path.

Problem 1.1(13) Suppose I were to tell you that I had discovered an algorithm that could correctly tell whether a graph G has a 5-clique, and could answer the problem in time that is polynomial in the number of vertices for the graph. What would that mean for the question "Does P = NP?"

Problem 1.1(14) Suppose I were to tell you that I had discovered an algorithm that could correctly tell you if a graph had an Eulerian cycle, and run in time that is polynomial in the number of vertices for the graph. What would that mean for the question "Does P = NP?"

Problem 1.1(15) Suppose I were to tell you that I had discovered an algorithm that could correctly tell you if a graph had a Hamiltonian path, and run in time that is polynomial in the number of vertices for the graph. What would that mean for the question "Does P = NP?"

2 Problems from Section 2: Trees

Problem 2.1(1) Draw the rooted tree that is given by $(f, ((a, b), (c, (d, e)))$.

Problem 2.1(2) Draw a rooted tree and give its Newick format representation.

Problem 2.1(3) Draw the rooted tree given by $(1, (2, (3, (4, (5, 6)))))$, and write down the set of clades of that tree.

Problem 2.1(4) Draw the same rooted tree in at least two different ways, with at least one of them using the style as in Figure 1(b) in the text.

Problem 2.1(5) For the rooted tree T given by $(a, ((b, c), (d, (e, f))))$,

- write down at least three other Newick representations.
- write down the the set of clades, and indicate which of the clades is non-trivial.

Problem 2.1(6) Compute the Hasse Diagram on the posets defined for the following sets of clades, and then draw the rooted tree for each set.

- $\{\{a, b\}, \{a, b, c\}, \{a, b, c, d\}, \{e, f\}, \{e, f, g\}\}$
- $\{\{a, b, c\}, \{a, b, c, d\}, \{e, f\}, \{e, f, g\}\}$

Which one of these trees is *not binary*?

Problem 2.1(7) Draw all rooted binary trees on leaf set $\{a, b, c, d\}$. (Note that trees that can be obtained by swapping siblings are the same.)

Problem 2.1(8) Draw all rooted trees (not necessarily binary) on leaf set $\{a, b, c, d\}$.

Problem 2.1(9) Give a polynomial time problem to determine if two Newick strings represent the same rooted tree. For example, your algorithm should return "YES" on the following pair of strings:

- $(a, (b, c))$ and $((c, b), a)$

and should return "NO" on

- $(a, (b, c))$ and $(b, (a, c))$

Problem 2.2(1) Draw the rooted and unrooted versions of the unrooted tree given by the following Newick string: $((a, b), (c, (d, e))$.

Problem 2.2(2): Draw all the rooted versions of the unrooted tree $(x, (y, (z, w)))$, and give their Newick formats.

Problem 2.2(3) Draw the unrooted version of the trees given below, and write down the set $C(T)$ of each tree T below. Are the two trees the same as unrooted trees?

1. $(a, (b, (c, ((d, e), (f, g)))))$.
2. $(((a, b), c), ((d, e), (f, g)))$

Problem 2.2(4) Consider the two unrooted trees given below by their bipartition encodings. Draw them. Do you see how one tree can be derived from the other by contracting a single edge? Which one refines the other?

- T_1 is given by $C(T_1) = \{(ab|cdef), (abcd|ef)\}$
- T_2 is given by $C(T_2) = \{(ab|cdef)\}$.

Problem 2.2(5) Draw two unrooted trees, so that neither can be derived from the other by contracting a set of edges.

Problem 2.2(6) Draw three different unrooted trees, T_1, T_2, and T_3, on no more than 8 leaves, so that T_1 is a contraction of T_2, and T_2 is a contraction of T_3 (identically, T_3 is a refinement of T_2, and T_2 is a refinement of T_1). Write down the bipartition encodings of each tree.

Problem 2.2(7) Apply the technique for computing unrooted trees from compatible bipartitions to the input given below, using leaf 3 as the root. After you are done, do it again but use a different leaf as the root. Compare the rooted trees you obtained using the different leaves as roots: are they different? Unroot the trees, and compare the two unrooted trees. Are they the same?

Input: $\{(123|456789), (12345|6789), (12|3456789), (89|1234567)\}$.

Problem 2.2(8) Compute the unrooted trees compatible with the following sets of bipartitions (use the algorithm that operates on clades, using the specified roots):

- $\{(ab|cdef), (abc|def), (abcd|ef)\}$, with root "b". Then do this again using root c. Are the unrooted trees you get different or the same?
- $\{(ab|cdef), (abc|def), (abcd|ef)\}$, with root "d".
- $\{(abcdef|ghij), (abc|defghij), (abcdefg|hij)\}$, using any root you wish.

Problem 2.2(9) Give a polynomial time algorithm to determine if the *unrooted* trees defined by two Newick strings are the same. Your algorithm should return "YES" for the following pairs of strings:

- $(a,(b,(c,d)))$ and $((a,b),(d,c))$
- $(a,(b,(c,d)))$ and $(c,(d,(b,a)))$

Your algorithm should return "NO" for

- $(a,(b,(c,d)))$ and $((b,d),(a,c))$

Problem 2.2(10) Implement the algorithm you designed for problem 2.2(9). Record its running time and compare against the theoretical running time analysis you provided for problem 2.2(9).

Problem 2.3(1) The following set of unrooted trees was discussed in Section 2.3:

- T_1 given by $C(T_1) = \{(12|3456),(123|456),(1234|56)\}$
- T_2 given by $C(T_2) = \{(12|3456),(123|456),(1235|46)\}$
- T_3 given by $C(T_3) = \{(12|3456),(126|345),(1236|45)\}$

Is it possible to order the bipartitions of this set so as to produce T_2 as a greedy consensus? If so, provide one such ordering. If not, explain why not.

Problem 2.3(2) Suppose you have an arbitrary set $\mathcal{T}$ of trees on the same leaf set, and you compute the strict, majority, and greedy consensus trees. Suppose that the strict and majority consensus trees are different. Must one of them refine the other? If so, which one, and why? Same question for the greedy consensus and the majority consensus. Finally, what about the strict consensus tree and an arbitrary tree in $\mathcal{T}$? What about the majority consensus and an arbitrary tree in $\mathcal{T}$?

Problem 2.4(1) Give two different compatible unrooted trees on the same leaf set, and present their minimal common refinement.

Problem 2.4(2) Give two different trees on the same leaf set, neither of which is fully resolved, and which are *not* compatible.

Problem 2.5(1) Let unrooted T_0 given by $(a,(b,(c,((d,e),(f,g)))))$ denote the true tree.

1. For each unrooted tree below, draw the tree, and write down the bipartitions that are false positives and false negatives with respect to T_0.

- $T_1 = (f, (g, (a, (b, (c, (d, e))))))$.
- $T_2 = (g, (f, (c, (d, (e, (a, b))))))$.
- $T_3 = (g, (f, (a, (b, (c, (d, e))))))$.

2. Draw the strict, majority, and greedy consensus trees for these three trees T_1, T_2, and T_3. Compute the false negatives and false positives (with respect to T_0) for these consensus trees.

Problem 2.5(2) Consider an arbitrary unrooted true tree that is binary, and let $\mathcal{T}$ be a set of estimated unrooted trees. Suppose you compute the strict consensus, majority consensus, and greedy consensus of these trees. Now compute the false negative error rates of these three consensus trees, and compare them to each other and also to the false negative error rate of the trees in the set $\mathcal{T}$. What can you deduce? Do the same thing for the false positive error rates.

Problem 2.5(3) Give two unrooted trees, T_1 and T_2, which are compatible, and their unrooted compatibility tree T_3. Treat T_3 as the true tree, and compute the False Negative and False Positive rates of T_1 and T_2 with respect to T_3. What do you see?

Problem 2.5(4) Let T_0 be the unrooted tree given by splits $\{123|456, 12|3456, 1234|56\}$, and let T_1 be an estimated tree. Suppose T_1 is missing split $123|456$, but has a false positive $124|356$. Draw T_1.

Problem 2.5(5) Give an algorithm for the following problem:

- Input: unrooted tree T_0 and two sets of bipartitions, C_1 and C_2, where $C_1 \subseteq C(T_0)$ and $C_2 \cap C(T_0) = \emptyset$.
- Output: tree T_1 (if it exists) such that T_1 has false negative set C_1 and false positive set C_2, when T_0 is treated as the true tree. (Equivalently, $C(T_1) = [C(T_0) - C_1] \cup C_2$.)

Problem 2.5(6) Describe a polynomial time algorithm to compute the compatibility tree of two unrooted trees, and implement it.

Problem 2.7(1) For each of the given unrooted trees, draw the subtree induced on $\{a, b, c, d\}$.

- T has Newick format $(b, (a, (f, (c, (g, (d, e))))))$ (i.e., it is the caterpillar b, a, f, c, g, d, e).
- T has the Newick format $(f, (a, (c, (g, (d, (b, e))))))$ (i.e., it is the caterpillar f, a, c, g, d, b, e).

Problem 2.7(2)

- Give two unrooted trees on a, b, c, d, e, f, g that induce the same subtree on a, b, c, d but that are different trees.
- Give two unrooted trees on a, b, c, d, e, f, g that are identical on $\{a, b, c, d\}$ and different on $\{d, e, f, g\}$.
- Give two rooted trees on a, b, c, d, e that are identical on a, b, c but different on d, e, f.

3 Problems from Section 3: Computing trees from subtrees

Problem 3.1(1) Make up a rooted tree on 6 leaves, and write down all its rooted triples. Then make up another rooted tree on the same 6 leaves, and write down all its rooted triples. How many rooted triples do your trees disagree on?

Problem 3.1(2) Make up two rooted trees on at least 5 leaves that differ in exactly one rooted triple.

Problem 3.1(3)

- Write down the set X of rooted triples for the caterpillar tree given by $(1,(2,(3,(4,5))))$.
- Apply the two algorithms for constructing trees from rooted triples to the set X. What do you find? Do they produce the same output?

Problem 3.1(4) Is it possible to have a set of rooted triplets that is compatible but where some pair of leaves i, j is not separated in any pair in which they both appear, but are not siblings in *any* tree that realizes the set of rooted triplets? If so, provide the example, and otherwise prove it is impossible.

Problem 3.1(5) Suppose we modify the Aho, Sagiv, Szymanski, and Ullman algorithm (see Section 3.1 in the text) as follows. We compute the equivalence relation, and if there is more than two equivalence classes, $C_1, C_2, \ldots, C_k$ (with $k > 2$) we make *two* subproblems, C_1 and $C_2 \cup C_3 \cup \ldots, \cup C_k$. Otherwise, we don't change the algorithm. Does this also solve rooted triplet compatibility? (Prove or disprove.)

Problem 3.2(1) Make up an unrooted tree on at least 5 leaves, and write down all its unrooted quartet trees.

Problem 3.2(2) Make up two different unrooted trees on the same leaf set, but try to make them disagree on as few unrooted quartet trees as possible. How many do they disagree on?

Problem 3.2(3) Construct a tree on leaf set $\{a, b, c, d, e, f\}$ that induces each of the following quartet trees:

- $(ab|cd)$,
- $(ab|ce)$,
- $(ac|de)$,

- $(bc|de)$,
- $(ab|de)$,
- $(ab|cf)$,
- $(ab|df)$,
- $(ab|ef)$,
- $(ac|df)$,
- $(ac|ef)$,
- $(ad|ef)$

4 Problems from Section 4: Constructing trees from qualitative characters

Problem 4.2(1) Suppose we are given the following input of binary characters, in which 0 denotes the ancestral state and 1 denotes the derived state. Construct the rooted tree that is consistent with these characters evolving without homoplasy.

- $a = (1, 1, 0, 0, 1, 0)$
- $b = (1, 0, 1, 0, 1, 0)$
- $c = (0, 0, 0, 1, 0, 0)$
- $d = (0, 0, 0, 0, 1, 1)$

Problem 4.3(1) Construct an unrooted tree that is consistent with the following input of binary characters, under the assumption that all characters evolve without homoplasy. (You may not assume that any particular state is ancestral on any character.)

- $a = (0, 0, 1, 1)$
- $b = (1, 0, 0, 1)$
- $c = (1, 1, 0, 1)$
- $d = (1, 0, 1, 0)$

Problem 4.3(2) Take the data matrix from Problem 4.2(1) and add in the root sequence, r, given by $r = (0, 0, 0, 0, 0, 0)$. Thus, you now have a matrix with five taxa, a, b, c, d, r, defined by six characters. Divide this matrix into two pieces: the first three characters, and the last three characters. Construct the minimally resolved unrooted tree that is compatible with each submatrix. How are these trees different? Are they fully resolved, or do they have polytomies? Compare them to the tree you obtained on the full matrix. Now, treat the tree on the full matrix as the "true tree", and compute the False Negative and False Positive rates for these two trees. What do you find? Finally, are these two trees compatible?

Problem 4.4(1) For the tree T given by ((a,(b,(c,(d,(e,f))))));), determine for each of the following characters (columns in the following tuple representation) whether it could have evolved on the tree T without any homoplasy:

- $a = (0, 0, 0, 0, 1)$
- $b = (0, 1, 1, 0, 0)$
- $c = (1, 0, 0, 1, 1)$

- $d = (1, 2, 0, 1, 0)$
- $e = (2, 0, 2, 0, 1)$
- $f = (2, 3, 2, 0, 1)$

Problem 4.4(2) For the following input, show how to set the entries given with "?" so as to produce a compatible matrix:

- $A = (0, 1, 0, ?)$
- $B = (0, 1, 1, 0)$
- $C = (0, 0, 1, 0)$
- $D = (1, 0, 1, 1)$
- $E = (1, 0, ?, 1)$

Explain how you derived your solution.

Problem 4.4(3) In the text, we said that there was no way to set the values for the missing entries in the following matrix, in order to produce a tree on which all the characters are compatible:

- $A = (0, 0, ?)$
- $B = (0, 1, 0)$
- $C = (1, 0, 0)$
- $D = (1, ?, 1)$
- $E = (?, 1, 1)$

Prove this assertion.

Problem 4.4(4) Suppose T and T' are two trees on the same leaf set, and T' refines T.

- Prove or disprove: if character c is compatible on T then it is compatible on T'.
- Prove or disprove: if character c is compatible on T' then it is compatible on T.

Problem 4.4(5) The maximum parsimony problem asks us to find a tree that has the best maximum parsimony score with respect to a matrix M. Suppose we consider the following problem, "binary tree maximum parsimony": Given a matrix M, find a *binary tree* that optimizes maximum parsimony.

1. Is it possible for a solution to the "binary tree maximum parsimony" problem to not be optimal for the standard maximum parsimony problem?
2. Consider the same question but restated in terms of maximum compatibility and "binary tree maximum compatibility". Does your answer change?

Problem 4.6(1) Consider the set of six taxa described by two multi-state characters, $A = (0,0), B = (1,2), C = (0,2), D = (2,1), E = (1,1)$, and $F = (1,0)$, and the tree on the taxa given by: $(((A,B),C),(D,(E,F)))$.

- Apply the parsimony algorithm to assign states to each node for each of the two characters. What is the parsimony score of this tree?
- For which nodes of the tree is the character state of either character determined, and for which nodes is it optional?
- Give two different character state assignments to the nodes to produce the minimum number of changes.

Problem 4.6(2) Find an optimal MP tree T for the input given in Problem 4.6(1). Are either of the characters compatible on T? If not, find an optimal MP tree for this input for which at least one character is compatible.

Problem 4.6(3) Suppose T and T' are two trees on the same leaf set, and T' refines T. Prove that the parsimony score of T' is at most that of T.

Problem 4.7(1) Consider the following multi-state characters.

- $L_1 = (0,0,0)$
- $L_2 = (0,1,1)$
- $L_3 = (1,1,2)$
- $L_4 = (1,2,0)$

1. Does a perfect phylogeny (tree on which all characters evolve without homoplasy) exist for this dataset? If so, prove this by presenting the perfect phylogeny. Otherwise, prove that it does not.
2. Write down the binary encoded version of this input. Does a perfect phylogeny exist for the binary encoded version of the matrix? If so, prove this by presenting the perfect phylogeny. Otherwise, prove that it does not.

Problem 4.8(1) Suppose M is an input matrix for maximum parsimony, so M assigns states for each character to all the taxa in a set S. Suppose M' is the result of removing all characters from M that are identical on all taxa (i.e., characters c such that $c(s) = c(s')$ for all s, s' in S). Prove or disprove: M and M' have the same set of optimal trees under maximum parsimony.

Problem 4.8(2) Suppose M is an input matrix for maximum parsimony and M' the result of removing all characters from M that have different states on every taxon (i.e., i.e., characters c such that $c(s) \neq c(s')$ for all $s \neq s'$ in S). Prove or disprove: M and M' have the same set of optimal trees under maximum parsimony.

Problem 4.8(3) Do problems 4.8(1) and 4.8(2) but with respect to maximum compatibility.

Problem 4.8(4) Let M be an input matrix to maximum parsimony, and let M' be the result of removing all parsimony uninformative characters from M. Thus, M' has a subset of the columns of M. By Lemma 3, the trees that are returned by an exact MP solution on M' will be the same as the maximum parsimony trees returned for M. However, suppose you use the characters to define "branch lengths" in some output tree (as there can be many), as follows. You use maximum parsimony to calculate ancestral sequences, and then you use Hamming distances to define the branch lengths on the tree.

1. Is it the case that branch lengths you estimate on a given tree T must be the same for M as for M'? (In other words, can branch length estimations change?)

2. If you use normalized Hamming distances instead of Hamming distances, does your answer change?

Problem 4.8(5) Consider the following input matrix to maximum parsimony:

- $a = (0, 1, 0, 0, 0)$
- $b = (0, 0, 1, 1, 1)$
- $c = (0, 0, 2, 3, 2)$
- $d = (0, 2, 0, 1, 1)$
- $e = (1, 2, 0, 1, 1)$
- $f = (0, 0, 3, 2, 1)$

Write down all the optimal solutions to maximum parsimony on this input, and explain how you obtain your answer. Do *not* solve this by looking at all possible trees on $\{a, b, c, d, e, f\}$. (Hint: Read Section 4.8 in the text.)

Problem 4.8(6) Is it the case that maximum compatibility and maximum parsimony always return the same set of optimal trees? If so prove it, and otherwise find a counterexample.

5 Problems from Section 5: Distance-based methods

Problem 5.2(1): Draw an edge weighted tree T with all branches having positive weight, and derive its additive matrix. Check that the four point condition applies for at least two different quartets of leaves.

Problem 5.2(2) For the additive matrix you produced in Problem 5.1, compute the tree for every quartet of taxa, by applying the four-point method. Then apply the Naive Quartet method to the set of quartets. Verify that you produce the same tree.

Problem 5.2(3) Consider the matrix in Figure 19 from the text. Apply UPGMA to the matrix. What is the *unrooted* tree that you obtain? Does it equal the tree given in Figure 20?

Problem 5.2(4) Take the matrix you had produced in Problem 5.2(1), and change one entry. Determine if the new matrix is additive. If not, prove it is not by producing the four leaves for which the four-point condition fails. If yes, prove that is by producing the edge weighted tree that realizes the new matrix.

Problem 5.2(5) Compute the Hamming distance matrix for the set of four taxa, $\mathcal{L} = \{L_1, L_2, \ldots, L_4\}$, given below (each described by four binary characters). Is the distance matrix additive? If you apply the UPGMA method to this distance matrix, what do you get? If you apply the Four-Point Method to the matrix, what do you get? What is the solution to maximum parsimony on this input of four taxa? What is the solution to maximum compatibility? Are these characters compatible?

- $L_1 = (0, 1, 0, 1, 0)$
- $L_2 = (0, 0, 0, 0, 0)$
- $L_3 = (1, 0, 0, 0, 0)$
- $L_4 = (1, 0, 1, 0, 1)$

Problem 5.2(6) Prove the following: If C is a set of binary characters that evolve without homoplasy on a tree T, then the Hamming distance matrix $H(i, j)$ is additive. Furthermore, the Naive Quartet Method applied to H would yield the tree T', defined to be T with all zero-event edges contracted.

Problem 5.2(7) Prove or disprove: If C is a set of characters (not necessarily binary) that evolve without any homoplasy on a tree T, then the Hamming distance matrix is additive.

Problem 5.2(8) Consider the Naive Quartet Method applied to pairwise Hamming distances; call this the NQM(Hamming) method. For binary characters, what characters are uninformative for the NQM(Hamming) method?

6 Problems from Section 6: Statistical phylogeny estimation methods

Problem 6.1(1) Suppose you have the CF tree T with topology $((A, B), (C, D))$ with every edge having $p(e) = 0.1$, and rooted at A. Compute the probability of $A = B = C = D = 0$.

Problem 6.1(2) Consider a Cavender-Farris model tree T given by $((A, B), (C, D))$. Treat this as a rooted tree, with A being the root, and thus having five edges. Suppose the internal edge is labelled e_I, and we set $p(e_I) = .4$, and $p(e) = 0.001$ for all the other edges.

1. Compute the probability of the following events:
   - $A = B = 0$ and $C = D = 1$
   - $A = C = 0$ and $B = D = 1$
   - $A = D = 0$ and $B = C = 1$
2. Would maximum parsimony be statistically consistent on this model tree? Why?

Problem 6.1(3) In this problem we will define a set of different CF model trees on the same tree topology, $((A, B), (C, D))$ but with different edge parameters. We let e_I be the internal edge separating A, B from C, D, and let e_x be the edge incident with leaf x (for x=A,B,C,D). The trees are then defined by the edge parameters $p(e)$ for each of these edges, with these $p(e)$ given as follows:

- For T_1, we have $p(e_A) = p(e_C) = .499$, and $p(e) = 0.0001$ for the other edges e.
- For T_2, we have $p(e) = .499$ for all edges e.
- For T_3, we have $p(e) = .0001$ for all edges e.
- For T_4, we have $p(e_I) = .499$ and $p(e) = .01$ for the other edges e.

Think about what kinds of character patterns you would see at the leaves of the trees, and answer the following questions;

1. Of the three parsimony-informative character patterns, identify which one(s) would appear most frequently for tree T_1
2. Of the three parsimony-informative character patterns, identify which one(s) would appear most frequently for tree T_4
3. Now suppose one of these CF trees generated a dataset of four sequences, and you had to guess which one generated the data. Suppose the dataset consisted of four sequences A, B, C, D of length 100 that were all identical, which would you choose?

4. Same question as above, but suppose the dataset consisted of four sequences A, B, C, D of length 10, where

   - $A = 0100100111$
   - $B = 0000000000$
   - $C = 0010101001$
   - $D = 0000000000$

Problem 6.1(4) Suppose you have performed the binary-encoding of a multistate character c defined on six taxa, A, B, C, D, E, F, with $c(A) = c(B) = 1, c(C) = c(D) = 2$, and $c(E) = c(F) = 3$. The three binary characters you obtain are c_i, for $i = 1, 2, 3$. Thus, $c_i(x) = 1$ if $c(x) = i$, and otherwise $c_i(x) = 0$. Assume that there is no polymorphism in the multistate character c, so that each of the taxa can have only one state of c.

1. Write down the binary encoding of each character.
2. Analyze the dataset under maximum parsimony; what trees do you find?

Problem 6.1(5) Let c be a character with three states, $1, 2$ and 3; consider the binary encoding of c, and let A be one of the taxa on which c is defined.

1. Suppose you know that $c_1(A) = 1$. What can you say about $c_2(A)$?
2. Suppose you know that $c_1(A) = 0$. What can you say about $c_2(A)$?
3. Suppose you know that $c_1(A) = c_2(A) = 0$. What can you say about $c_3(A)$?
4. Is it possible for $c_1(A) = c_2(A) = c_3(A)$? Why or why not?
5. Under the assumption that the character c is not polymorphic, can the characters c_1, c_2 and c_3 be *statistically independent*?

7 Problems from Section 7: Other phylogeny estimation issues

Problem 7.1(1) Suppose the rooted tree T is given, as $(A,(B,(C,(D,E))))$. Suppose the character dataset is

- $A = (0,0,0,0)$
- $B = (0,1,0,1)$
- $C = (1,1,2,1)$
- $D = (1,2,3,0)$
- $E = (1,2,4,0)$

First, determine which characters are compatible on the tree. For these character(s), determine which nodes of the tree have uniquely determined states for the character. Finally, for the character(s) that are not compatible on the tree, which nodes have uniquely determined states for these characters? To answer this, apply the maximum parsimony algorithm, and determine the character state assignments which optimize the parsimony score.

Problem 7.2(1) Suppose the tree is given by $(A,(B,(C,(D,E))))$, and that we have three homoplasy-free characters on these taxa given by:

- $A = (0,0,1)$
- $B = (0,1,1)$
- $C = (0,0,0)$
- $D = (1,0,0)$
- $E = (1,0,0)$

Assume that 0 is the ancestral state and 1 the derived state for each of these characters. Determine the edges in the tree that could contain the root.

8 Problems from Section 8: Multiple sequence alignment

9 Problems from Section 9: Constructing species trees from multiple genes

10 Problems from Section 10: Detecting and representing reticulation

Problem 10.2(1): Write down a rooted phylogenetic network with two contact edges, and all the trees that are contained within the network.

Problem 10.2(2): Consider the following taxa defined by qualitative characters. A perfect phylogenetic network with exactly one contact edge exists for these taxa – find it. Produce the trees contained inside the network, and demonstrate that for every character in this set, there is at least one tree in the network on which it is compatible.

- $A = (0, 1, 0)$
- $B = (0, 0, 1)$
- $C = (1, 0, 1)$
- $D = (1, 1, 2)$

Problem 10.2(3): Consider the following set of rooted trees:

- $((A, B), (C, D))$
- $((A, D), (B, C))$
- $((A, C), (B, D))$
- $(A, (B, (C, D)))$.

For each tree, see if you can add a single contact edge so as to make a perfect phylogenetic network for the input given in Problem 8.2(2).

11 Problems from Section 11: Genome Rearrangement Phylogeny

12 Problems from Section 12: Historical Linguistics