UPGMA tree inference

Problem 1.

					inferred nodes		
	L2	L3	L4	L5	24	15	3:24
L1	14	14	14	8_2	14	X	14
	L2	10	4_1	14	X	X	X
		L3	10	14	10_3	14	X
			L4	14	X	X	X
				L5	14	X	X
					24	14	X
						15	14
							3:24

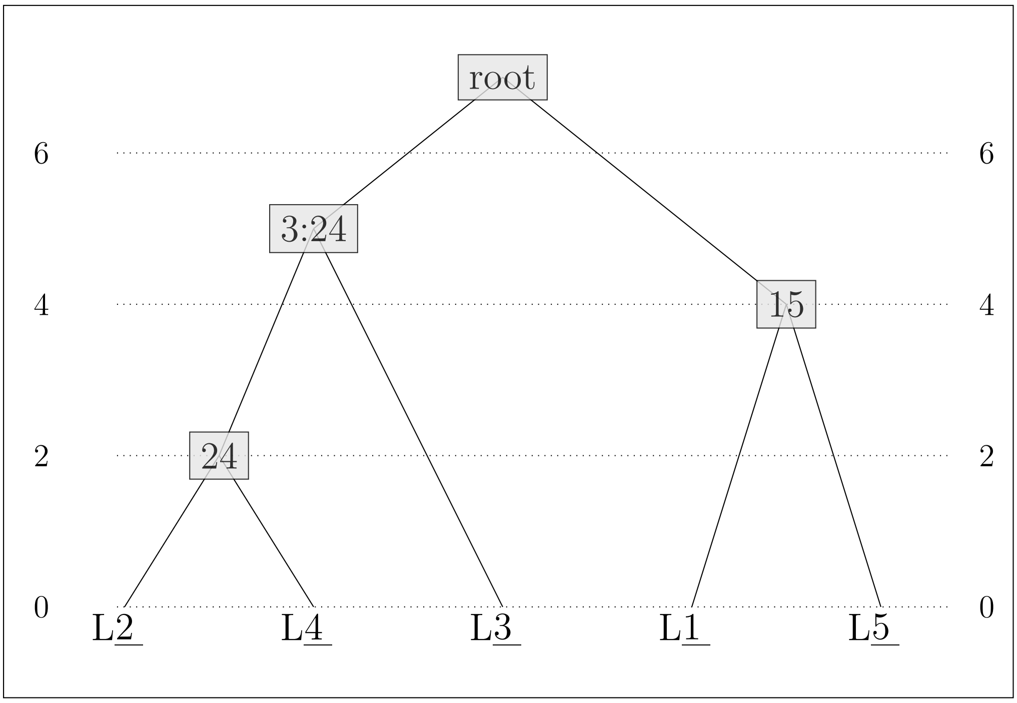
Problem 1a. Fill in all relevant distances computed by UPGMA.
Since this is a closed book test; let me remind you that UPGMA infers rooted trees assuming a constant speed of evolution.

The subscripts on distances indicate the order of merging by UPGMA, for example the 4_1 for the distance between L2 and L4 indicates that L2 and L4 are merged in the first step.

UPGMA tree inference

Problem 1b. Sketch the UPGMA inferred tree in box below.

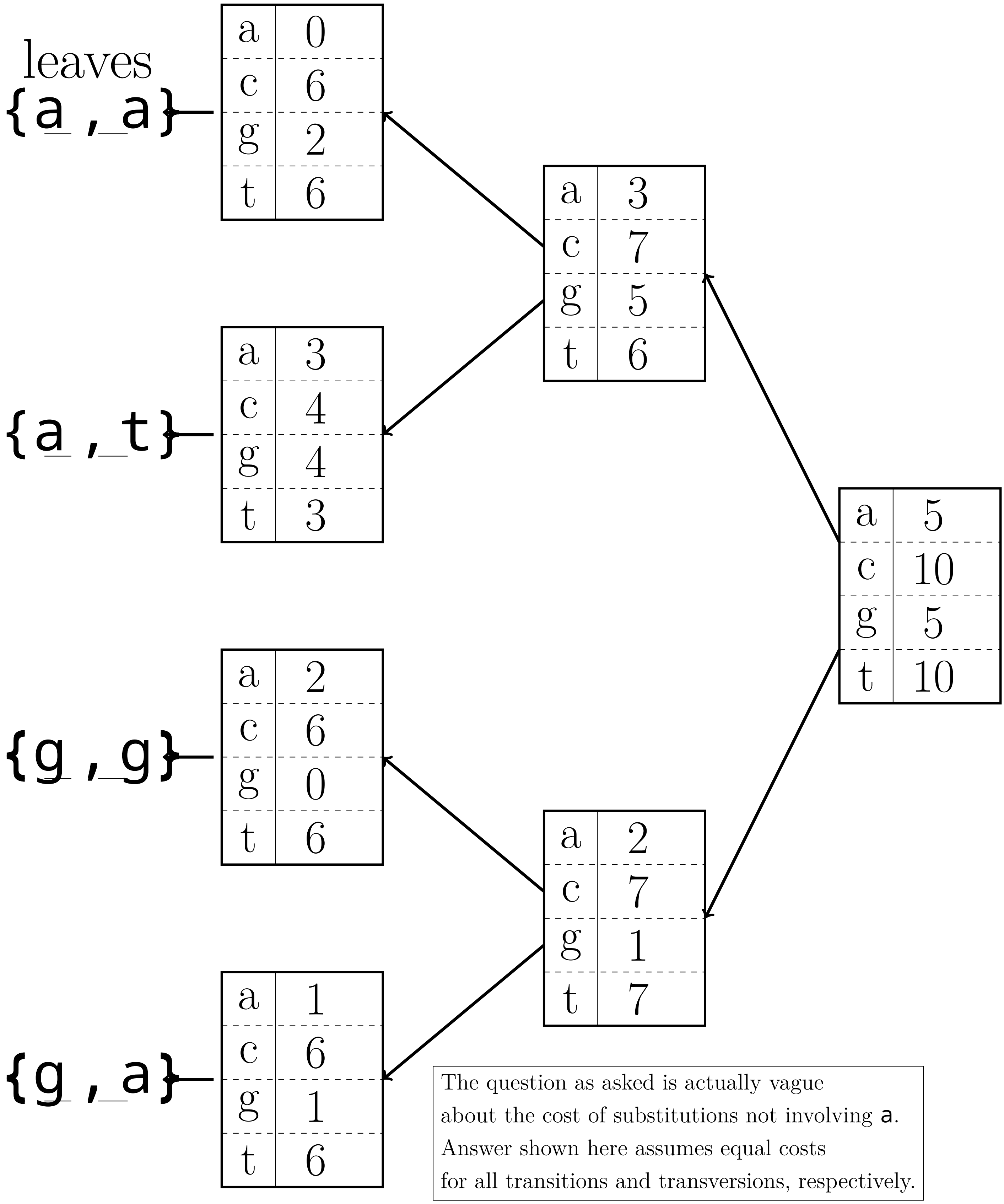
Sketch the tree inferred by UPGMA, with the leaves at height 0 and inferred ancestor nodes at the appropriate height. Indicate leaves by filling in “L1”, “L2” etc. where there is a “L_”. Show the root node as “root”.



Problem 1c. Is this UPGMA tree reliable? State your reasons.

The tree is reliable because the distances given are ultrametric.

Problem 2. Fill in the minimal costs for assuming each nucleotide in the blanks below.



{a_,_a}

a	0
c	6
g	2
t	6

{a_,_t}

a	3
c	4
g	4
t	3

{g_,_g}

a	2
c	6
g	0
t	6

{g_,_a}

a	1
c	6
g	1
t	6

a	3
c	7
g	5
t	6

a	2
c	7
g	1
t	7

a	5
c	10
g	5
t	10

The question as asked is actually vague
about the cost of substitutions not involving a.
Answer shown here assumes equal costs
for all transitions and transversions, respectively.

Affine Gap Alignment

Problem 3.

		C	G	T	A	T	T	G
	0	-3	-4	-5	-6	-7	-8	-9
C	-3	× × +2	-1 × -5	-2 × -5	-3 × -7	-4 × -7	-5 × -8	-6 -12 -10
G	-4	× -1 -5	-8 -8 +4	+1 -8 -3	0 -10 -4	-1 -9 -5	+2 -11 -6	+3 -13 -3
T	-5	× -2 -5	-8 -1 -3	-6 -6 +6	+3 -7 -1	+2 -8 +2	-1 -9 +1	-2 0 0
C	-6	× -3 -3	-6 -2 -4	-7 +3 -2	-5 -4 +4	+1 +1 +2	0 -2 +1	-1 -1 -1
G	-7	× -4 -8	-11 -3 -1	-4 +2 -4	-5 +1 +2	-1 0 +2	0 -2 0	-1 -2 +3

According to instructions on the following page.

Problem 3a. Fill in this dynamic programming table.

Note that some scores could be better (higher) if adjacent alternating gaps were allowed. For example, cell representing aligning `CGT-` with `CG`.

Alternating adjacent gap alignment:

```
CGT-
C--G
*2 1
```

But we are forced to use:

```
CGT-
--CG
2 x1
```

Problem 3**b**. List any globally optimal alignments (here) and show trace-back arrows on the dynamic table on the first page.

Solution:

```
CGTATTG        CGTATTG
CG--TCG        CGT--CG
**12*x*        ***12x*
```

Either of the two alignments above have score of: $(4)(2)-1-3-1=3$.

Affine Gap Alignment

		C	G	T	A	T	T	G
	0	-3	-4	-5	-6	-7	-8	-9
C	-3	-6 -6 +2	-1 -7 -5	-2 -8 -5	-3 -9 -7	-4 -10 -7	-5 -11 -8	-6 -12 -10
G	-4	-7 -1 -5	-8 -4 +4	+1 -5 -3	0 -6 -4	-1 -7 -5	+2 -8 -6	+3 -9 -3
T	-5	-8 -2 -5	-5 -1 -3	-4 -2 +6	+3 -3 -1	+2 -4 +2	-1 -1 +1	-2 0 0
C	-6	-9 -3 -3	-6 -2 -4	-5 +3 -2	0 -1 +4	+1 +1 +2	0 -2 +1	-1 -1 -1
G	-7	-10 -4 -8	-7 -3 -1	-4 +2 -4	-5 +1 +2	-1 0 +2	0 -2 0	-1 -2 +3

Shown above; the dynamic programming table allowing for alternating adjacent gaps. Notice that some gap state scores increase, but the overall maximum remains 3.

As an aside; note that the difference in scores between adjacent cells in the dynamic programming table becomes more homogeneous (often differing by 1 for example). Algorithms (outside the scope of this question) have been designed which take advantage of this regularity.

Scoring Parameters

Alignment scoring parameters: match score +2;
mismatch score transition -1, transversion -2.
Gap open score -3, gap extend score -1.

A transition is a substitution `a` ⟷ `g` or `c` ⟷ `t`, other single nucleotide substitutions are transversions.

For example, the alignment:

```
CGTTTTAAG
CGTCA---G
***xX 3 *
```

With 4 matches, 1 transition, 1 transversion, 1 gap opening, & 2 gap extensions, would score: $(4)(2) - 1 - 2 - 3 - (1)(2) = 0$

Problem 3c. (On the next page) List the recursive relationships needed to efficiently compute a minimal cost global alignment with affine gap costs. you may assume the scoring parameters penalize gaps strongly enough that an optimal alignment never has alternating adjacent gaps.

In other words:

```
x-x     alignment like this never optimal
yy-
```

Notation

Let $x = x_1, \cdots, x_n$, $y = y_1, \cdots, y_m$ denote the sequences.
Note that the indices start from 1; so that 0 can be used as an index to represent the empty string "before" the start of a sequence.
Let $m(\mathtt{a}, \mathtt{b})$ indicate the score of aligning characters `a` and `b` together (a match when `a`=`b`, otherwise a mismatch), Let g_o indicate gap open, and g_e indicate gap extension costs.

Continued next page...

Additional Notation

For integers i, j such that $0 < i \leqq n$, $0 < j \leqq m$

$M(i, j)$ denotes the min. cost alignment of $x_{1\ldots i}$ and $y_{1\ldots j}$; ending with x_i aligned to y_j.

$X(i, j)$ denotes the min. cost alignment of $x_{1\ldots i}$ and $y_{1\ldots j}$; ending with x_i aligned to a gap.

$Y(i, j)$ denotes the min. cost alignment of $x_{1\ldots i}$ and $y_{1\ldots j}$; ending with y_j aligned to a gap.

Base Cases

Let $M(0, 0) = 0$, $\forall k > 0 M(0, k) = M(k, 0) = \infty$

Let $X(0, 0) = Y(0, 0) = \infty$, $\forall k > 0 M(0, k) = M(k, 0) = g_o + (k - 1)(g_e)$

Recurrences

$$M(i, j) = m(x_i, y_j) + \max \begin{cases} M(i-1, j-1) \\ X(i-1, j-1) \\ Y(i-1, j-1) \end{cases}$$

$$X(i, j) = \max \begin{cases} M(i-1, j) + g_o \\ X(i-1, j) + g_e \end{cases} \qquad Y(i, j) = \max \begin{cases} M(i, j-1) + g_o \\ Y(i, j-1) + g_e \end{cases}$$

Global Alignment Score

The global alignment score is: $\max \{M(n, m), X(n, m), Y(n, m)\}$

Problem 4.

Background

This problem is about using stochastic context free grammar formalization to model a RNA sequence motif, including secondary structure information.

We want to model a stem-loop structure like this one:

```
5'  (((NNNN)))Yu   3'
```

Where paired `()` represent bases paired in an RNA stem. Assume the following percent probabilities:

```
Y         c:40      u:60
N    a:20 c:30 g:20 u:30
()   c=g:25 g=c:25 a=u:20 u=a:20 u=g:05 g=u:05
```

Where 40 means 40%, etc; and `c=g` indicates a cytosine base paired with a guanine base in a stem structure.

Continued next page...

Problem 4**a**. Write a stochastic context free grammar to generate sequences consistent with the sequence motif and probabilities described above. Your grammar can use the symbols Y and N in the same way as above; but do not use “`()`”, or “`=`” in your grammar; instead use alphanumerical symbols such as H or H_1 etc.

Solution:

```
N -->  a:20 c:30 g:20 u:30
Y -->       c:40      u:60
L --> N N N N

H₁ --> aLu:20 gLc:25 aLu:20 uLa:20 uLg:5 gLu:5
H₂ --> aH₁u:20 gH₁c:25 aH₁u:20 uH₁a:20 uH₁g:5 gH₁u:05
H₃ --> aH₂u:20 gH₂c:25 aH₂u:20 uH₂a:20 uH₂g:5 gH₂u:05
B  --> H₃ Y
S  --> B u
```

Continued next page...

Problem 4**b**. What is the probability the model would generate the following sequence?

`ggccuaggcuuu`

Show enough calculation to justify your answer.

Solution: Important to note that this grammar produces fixed length strings and the alignment between terminals (RNA bases) to the production rules is trivial. the only possible parse is:

```
 ([{  NNNN  }]) Yu
 ggc  cuag  gcu uu
```

This requires a non-canonical g=u pair in the stem structure, which our grammar allows.

Since there is only one viable parse we just need the product of the mappings:

pattern	base(s)	% probability
N	c	20
N	u	30
N	a	30
N	g	20
()	gu	05
[]	gc	25
{}	cg	25
Y	u	60
u	u	100

```
(* .20 .30 .30 .20 .05 .25 .25 .60) --> 6.75e-06
```

The probability is 6.75×10^{-6}.