Midterm for 6.864
Name:

40	30	30	30

Good luck!

Part #1 — 40 points

Question 1 (10 points)

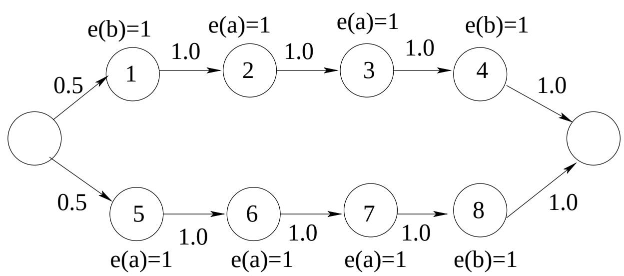
We define a PCFG where non-terminal symbols are $\{S, A, B\}$, the terminal symbols are $\{a, b\}$, and the start non-terminal (the non-terminal always at the root of the tree) is S. The PCFG has the following rules:

Rule	Probability
$S \rightarrow S\ S$	0.3
$S \rightarrow A\ A$	0.2
$S \rightarrow B\ A$	0.5
$A \rightarrow a$	0.3
$A \rightarrow b$	0.7
$B \rightarrow a$	0.4
$B \rightarrow b$	0.6

For the input string *aabb*, show two possible parse trees under this PCFG, and show how to calculate their probability.

Question 2 (10 points)

Consider a model merging algorithm for HMM induction. Given two strings *baab* and *aaab*, the initial model will have the following structure:

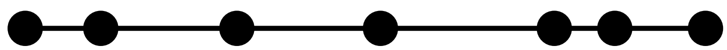


Note that we use the notation $e(b)$ or $e(a)$ to refer to emission probabilities; for example, for state 8 we have $e(b) = 1$, meaning that the symbol b is emitted with probability 1 from state 8 (Note that by implication, $e(a) = 0$ for state 8).

Show the structure of the model after merging two states. You should choose the two states to be merged in a way that the new model preserves the likelihood of the corpus (i.e., $P(baab) = 0.5$ and $P(aaab) = 0.5$).

Question 3 (20 points)

Consider an agglomerative single-link clustering with a Euclidean similarity measure. As shown in class, the complexity of this algorithm in the general case is $O(n^2)$. We wish to apply this clustering method to a set of points in a one dimensional space (i.e., points that on a line), for example the following:



Design an algorithm that identifies the highest two clusters in the agglomerative clustering. Your algorithm should be more efficient than $O(n^2)$. Specify the complexity of your algorithm. (Hint: you may want to consider the connection between single-link clustering and the minimum spanning tree over the n points.)

Part #2 30 points

Nathan L. Pedant generates (x, y) pairs as follows. Take $\mathcal{V}$ to be set of possible words (vocabulary), e.g., $\mathcal{V} = \{$`the, cat, dog, happy, ...`$\}$. Take $\mathcal{V}'$ to be the set of all words in $\mathcal{V}$, **plus** the reveresed string of each word, e.g., $\mathcal{V}' = \{$`the, eht, cat, tac, dog, god, happy, yppah, ...`$\}$.

First, Nathan chooses the symbol x to be some word from the vocabulary $\mathcal{V}$. He then does the following:

- With 0.4 probability, he chooses y to be identical to x.
- With 0.3 probability, he chooses y to be the reversed string of x.
- With 0.3 probability, he chooses y to be some string that is neither x nor the reverse of x. In this case he chooses y from the uniform distribution over words in $\mathcal{V}'$ that are neither x nor the reverse of x.

Question 4 (10 points)

Define the features for a log-linear model that can model this distribution $P(y|x)$ perfectly (Note: you may assume that there are no palindromes in the vocabulary, i.e., no words like `eye` which stay the same when reversed.) Your model should make use of as few features as possible (we will give you 10 points for using 2 features, 8 points for using 3 features, 5 points for using more than 3 features.)

Question 5 (10 points)

Write an expression for each of the probabilities

$$P(\text{the}|\text{the})$$
$$P(\text{eht}|\text{the})$$
$$P(\text{dog}|\text{the})$$

as a function of the parameters in your model.

Question 6 (10 points)

What value do the parameters in your model take to give the distribution used by Nathan, described above?

Part #3 — 30 points

We define a PCFG where the non-terminal symbols are $\{S, A, B\}$, the terminal symbols are $\{a, b\}$, and the start non-terminal (the non-terminal always at the root of the tree) is S. The PCFG has the following rules:

Rule
S → A S
S → A A
S → A B
A → a
A → b
B → a
B → b

We are now going to use EM to estimate the probabilities on rules in the grammar. As training data, we have a single string, $x = \texttt{aab}$. If $\mathcal{Y}$ is the set of parse trees for x under the above grammar, our goal is to maximize

$$L(\bar{\theta}) = \log P(x|\bar{\theta}) = \sum_{y \in \mathcal{Y}} \log P(x, y|\bar{\theta})$$

with respect to $\bar{\theta}$. Here $\bar{\theta}$ is a set of parameters in the model—a vector of values θ_r for each r in the grammar where θ_r is the probability associated with rule r. For example, we write

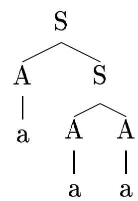
$$\theta_{S \to AB}$$

to denote the conditional probability $P(S \to AB|S)$. $P(x, y|\bar{\theta})$ is the probability given to tree y paired with string x under parameters $\bar{\theta}$.

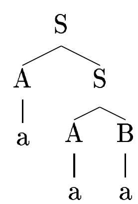
Question 7 (5 points)

For the input string `aab`, there are two parses under the grammar:

$y_1 =$



$y_2 =$



Given that $x = aab$, write expressions for $P(x, y_1|\bar{\theta})$ and $P(x, y_2|\bar{\theta})$ as a function of the parameters in the model.

Question 8 (25 points)

We are now going to derive the EM updates. Assume that we have a current set of parameters $\bar{\theta}^{t-1}$, and we'd like to estimate updated parameters $\bar{\theta}^{t}$. Show how to calculate parameter values $\bar{\theta}^{t}$ from the previous parameters $\bar{\theta}^{t-1}$, using the EM algorithm. Assume again that we have $x = aab$ as the only string in the training data.

Part #4 — 30 points

We define a PCFG where the non-terminal symbols are $\{S, A, B\}$, the terminal symbols are $\{a, b\}$, and the start non-terminal (the non-terminal always at the root of the tree) is S. The PCFG has the following rules:

Rule	Probability
S → A B	0.2
S → A A	0.2
S → A S	0.2
S → B S	0.4
A → a	0.6
A → b	0.4
B → a	0.7
B → b	0.3

Question 9 (5 points)

Draw one of the possible parse trees for the string `aabbb`, and show how to calculate its probability.

Question 10 (25 points)

The following algorithm is the pseudo-code from lecture, which finds the maximum probability for any tree when given as input a sequence of words $w_1, w_2, \ldots, w_n$. We take $K = 3$, and $N_1 = S, N_2 = A, N_3 = B$. The value stored in $\pi[1, n, 1]$ is then the highest probability for any tree spanning the words that is rooted in S.

Initialization:

For i = 1 ... n, k = 1 ... K
    $\pi[i, i, k] = P(N_k \rightarrow w_i | N_k)$

Main Loop:

For $length = 1 \ldots (n-1)$, $i = 1 \ldots (n - 1length)$, $k = 1 \ldots K$
    $j \leftarrow i + length$
    $max \leftarrow 0$
    For $s = i \ldots (j-1)$,
    For N_l, N_m such that $N_k \rightarrow N_l N_m$ is in the grammar
        $prob \leftarrow P(N_k \rightarrow N_l N_m) \times \pi[i, s, l] \times \pi[s+1, j, m]$
        If $prob > max$
            $max \leftarrow prob$
    $\pi[i, j, k] = max$

This algorithm will take $O(n^3)$ time to find the highest scoring tree for a sentence of length n. For our particular PCFG, how would you modify the algorithm to run in $O(n)$ time? (Hint: look closely at the parse tree you drew for question 5(i). You should see that it has a particular form, and that all other parse trees for this or other sentences will have that particular form.)