Final Project

(a)

My implementation of the PCFG class:

My implementation followed the algorithms of Conversions of PCFG to near-CNF. the Probabilistic CKY and Reconstructing parse trees (the last implemented from the last stage to the first to correctly reverse the rules in a parse tree).

I used a generator helper for near-CNF algorithm, that generates permutations of rules after removing variables of epsilon rules.

For the Probabilistic CKY algorithm, I used a helper function that calculates and chooses maximal probabilities for each variable in a cell. I have also used a function that adds unit rules to a cell according to existing variable.

For the Bonus question in Reconstructing parse trees, I used a dictionary of PCFGChange instances depicting rules that should be reversed, a recursive function was used to look up nodes in the parse tree resembling these rules and helper functions that reverse these said nodes into the original rules' form were used as well.

(b) question #1

to_near_cnf probability update:

a. New start variable:
   This step includes adding one new start variable to the grammar.
   The new start variable receives probability in the amount of 1.0, which does not affect multiplication and therefore leaves probability in the same amount as the original grammar.

b. Eliminate ε rules:
   This step includes removing all epsilon rules, updating rules probability for the variable and removing+adding new rules to compensate for the missing epsilon rules.
   For each variable, if an epsilon rule exists with probability [p], it is removed. All other rules of the same variable with probability [q] are then updated to have a $\left[\frac{q}{1-p}\right]$ probability, thus the sum of all such rules reaches a total of 1.0, equal to the original sum of probabilities for all rules of a variable.
   Afterward, all rules with the said variable in their derivation with probability [r] are removed. if the variable appeared in the rule's derivation x times, 2^x new rules are added in stead, each new rule is a permutation of all ways to remove or not remove each occurrence of the variable in the original rule's derivation. each new rule recieves probability of $[r \cdot p^k \cdot (1-p)^l]$, k represents the number of removed occurrences of the variable and l represents the number of kept occurrences of the variable. The sum of all 2^x new rules is exactly [r], equal to the probability of the original rule, and thus not affecting any change of probability to the string.

c. Shorten long rules + Eliminate terminals from binary rules:
   Both these steps include shortening one original rule and adding a new rule.
   For each shortened rule, the probability of the shortened rule remains the same as the original rule, thus not affecting any change in the probability of the string. The added rule receives probability in the amount of 1.0, which does not affect multiplication and therefore leaves probability in the same amount as the original grammar.

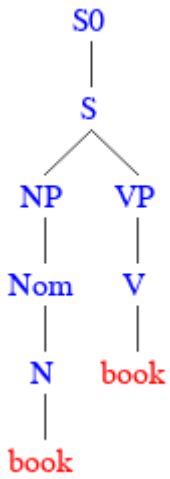
(c) question #4 and bonus question

Printed tree representations:

- book book

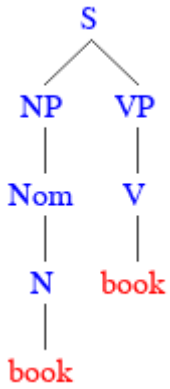
parse tree in near-cnf:

(0.0004950000000000002): [S0 [S [NP [Nom [N book]]] [VP [V book]]]]



parse tree in original grammar:

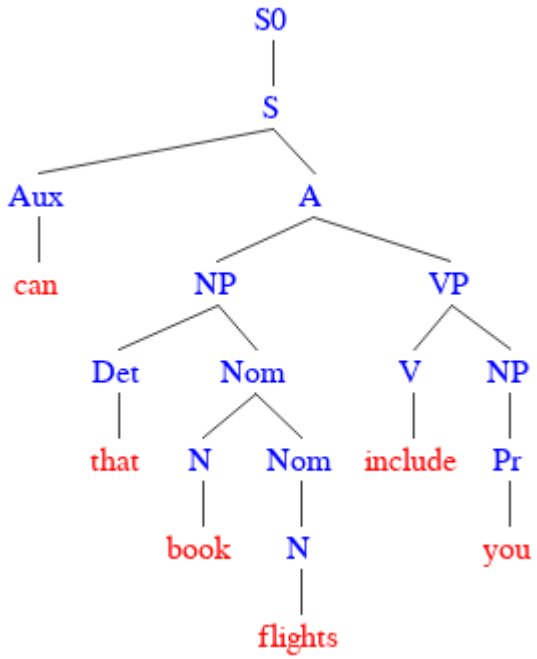
(0.0004950000000000002): [S [NP [Nom [N book]]] [VP [V book]]]



- can that book flights include you

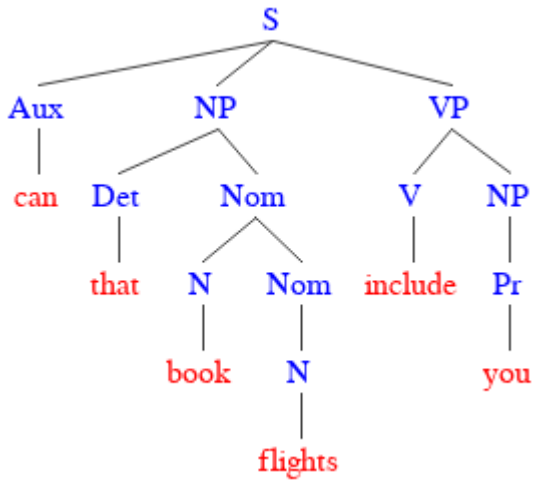
parse tree in near-cnf:

(8.640000000000005e-08): [S0 [S [Aux can] [A [NP [Det that] [Nom [N book] [Nom [N flights]]]] [VP [V include] [NP [Pr you]]]]]]



parse tree in original grammar:

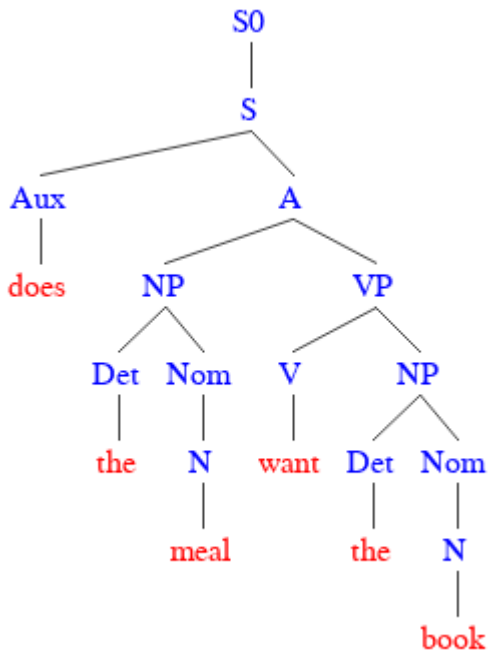
(8.640000000000005e-08): [S [Aux can] [NP [Det that] [Nom [N book] [Nom [N flights]]]] [VP [V include] [NP [Pr you]]]]



- does the meal want the book

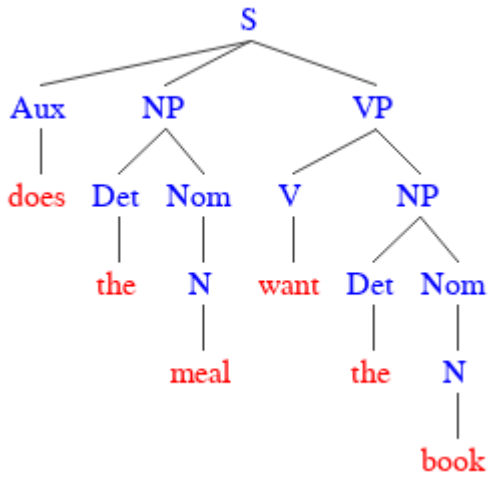
parse tree in near-cnf:

(4.147200000000002e-06): [S0 [S [Aux does] [A [NP [Det the] [Nom [N meal]]] [VP [V want] [NP [Det the] [Nom [N book]]]]]]]



parse tree in original grammar:

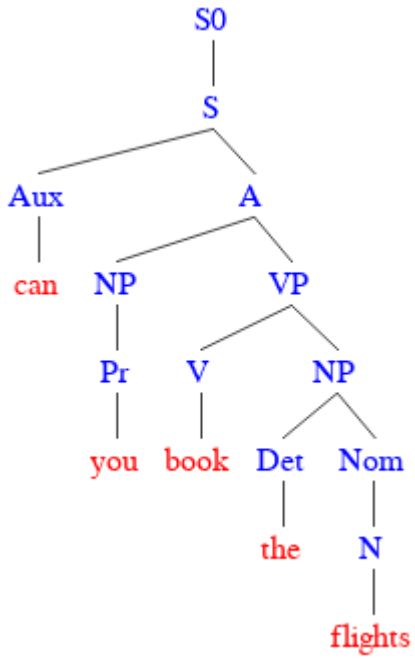
(4.147200000000002e-06): [S [Aux does] [NP [Det the] [Nom [N meal]]] [VP [V want] [NP [Det the] [Nom [N book]]]]]



- can you book the flights

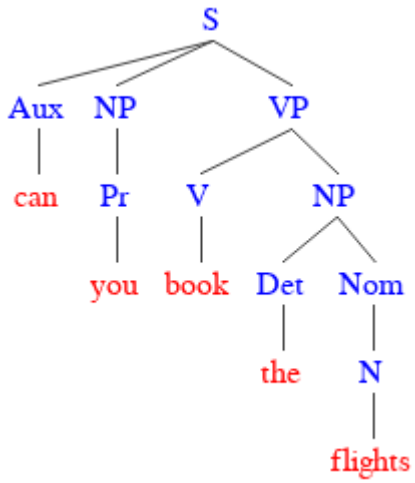
parse tree in near-cnf:

(6.912000000000003e-05): [S0 [S [Aux can] [A [NP [Pr you]] [VP [V book] [NP [Det the] [Nom [N flights]]]]]]]



parse tree in original grammar:

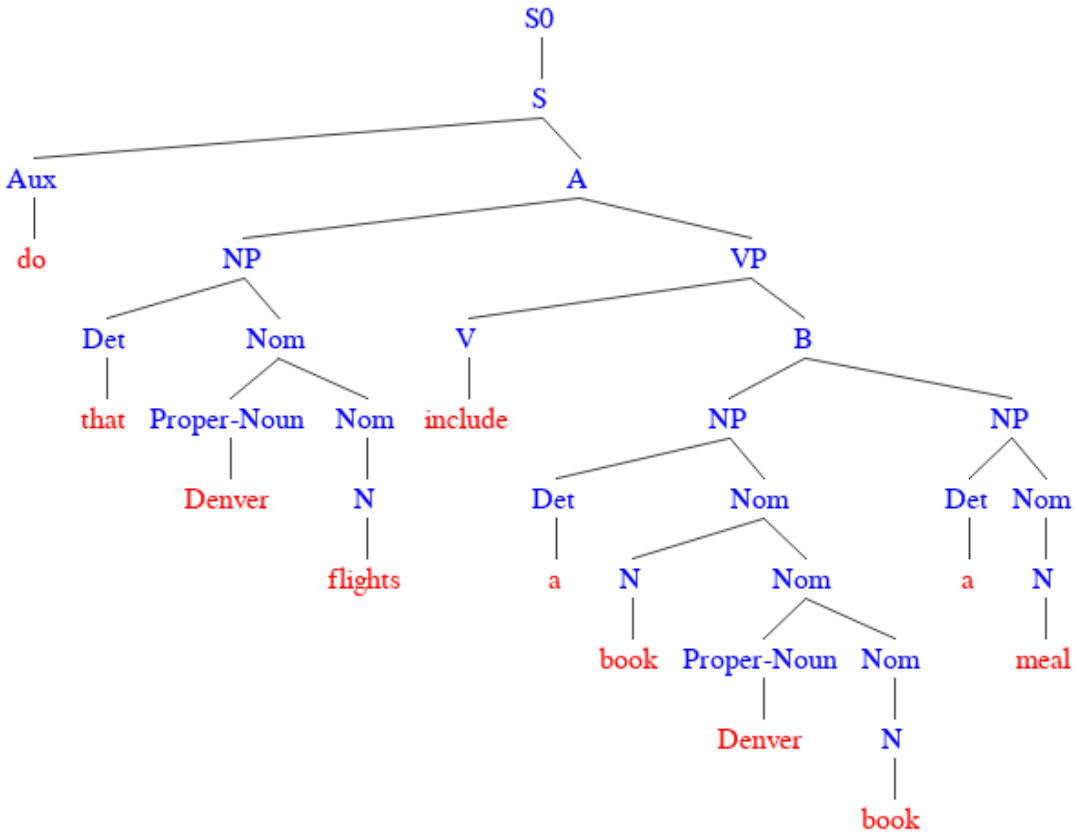
(6.912000000000003e-05): [S [Aux can] [NP [Pr you]] [VP [V book] [NP [Det the] [Nom [N flights]]]]]



- do that Denver flights include a book Denver book a meal

parse tree in near-cnf:

(6.407226562500004e-16): [S0 [S [Aux do] [A [NP [Det that] [Nom [Proper-Noun Denver] [Nom [N flights]]]] [VP [V include] [B [NP [Det a] [Nom [N book] [Nom [Proper-Noun Denver] [Nom [N book]]]]] [NP [Det a] [Nom [N meal]]]]]]]]



parse tree in original grammar:

(6.407226562500004e-16): [S [Aux do] [NP [Det that] [Nom [Proper-Noun Denver] [Nom [N flights]]]] [VP [V include] [NP [Det a] [Nom [N book] [Nom [Proper-Noun Denver] [Nom [N book]]]]] [NP [Det a] [Nom [N meal]]]]]