

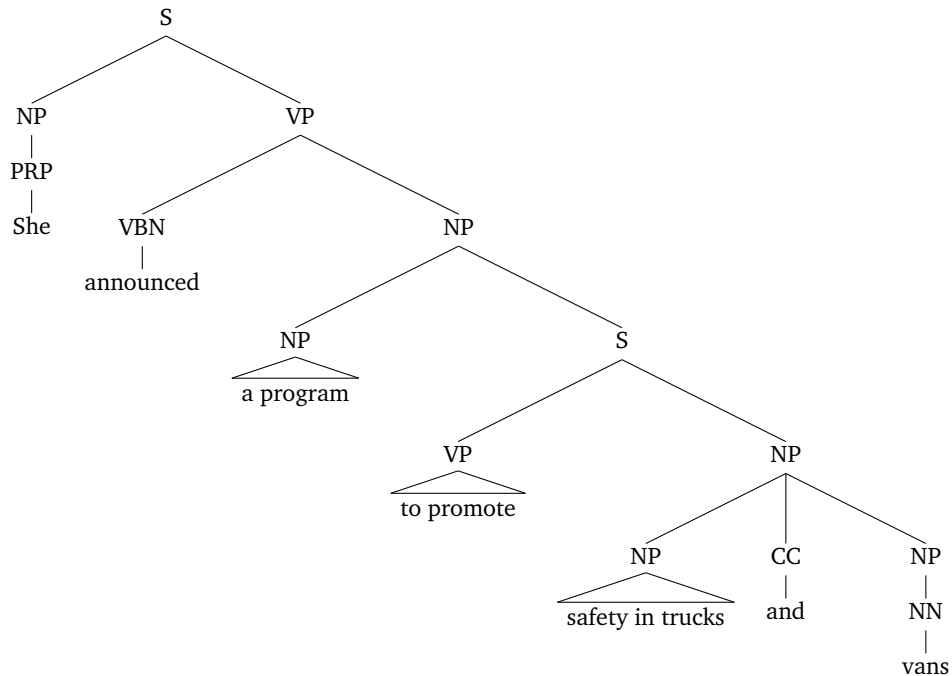
# Advanced Human Language Technologies

## Exercises on Parsing

### Context Free Grammars

#### Exercise 1.

Consider the sentence *She announced a program to promote safety in trucks and vans* and the following syntactic tree of one of its possible interpretations, in which the program promotes safety in trucks, and also promotes vans:

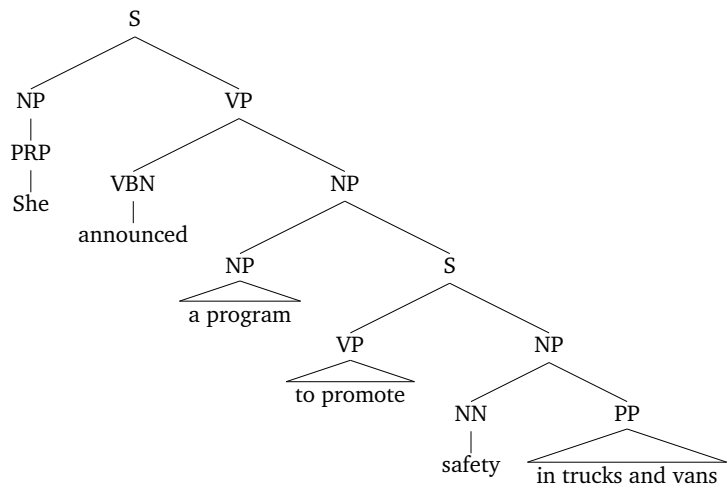


1. Draw the trees for at least three other interpretations for this sentence
2. Draw the trees for at least two interpretations for each of the following sentences
  - *The post office will hold out discounts and service concessions as incentives*
  - *They are hunting lions and tigers*
  - *Monty flies like mosquitoes*

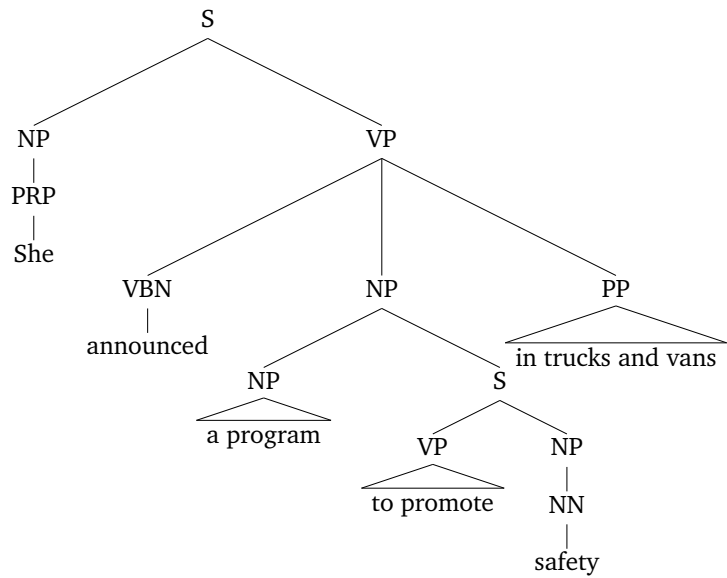
## SOLUTION

1. Find three interpretations:

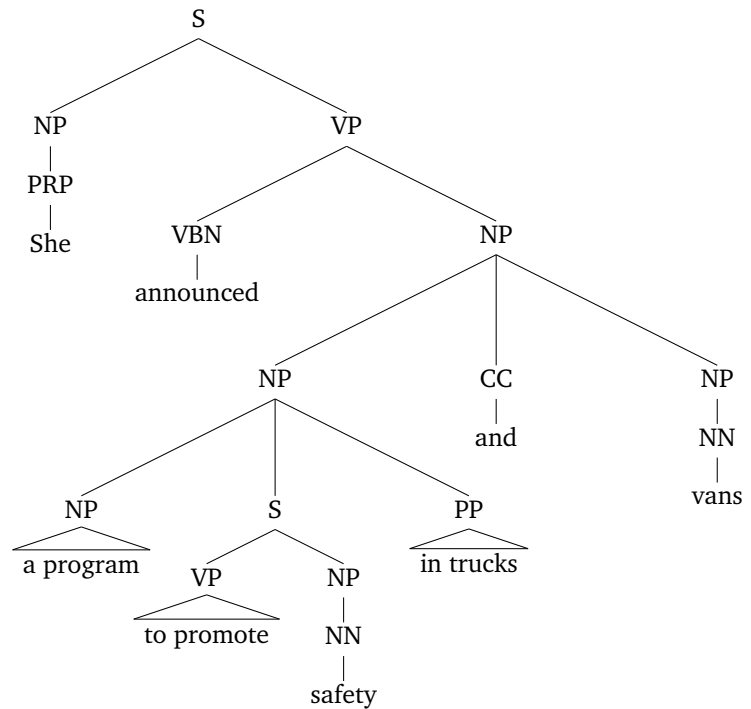
Interpretation 1: The announced program promotes safety in both trucks and vans.



Interpretation 2: The program is announced in trucks and vans.

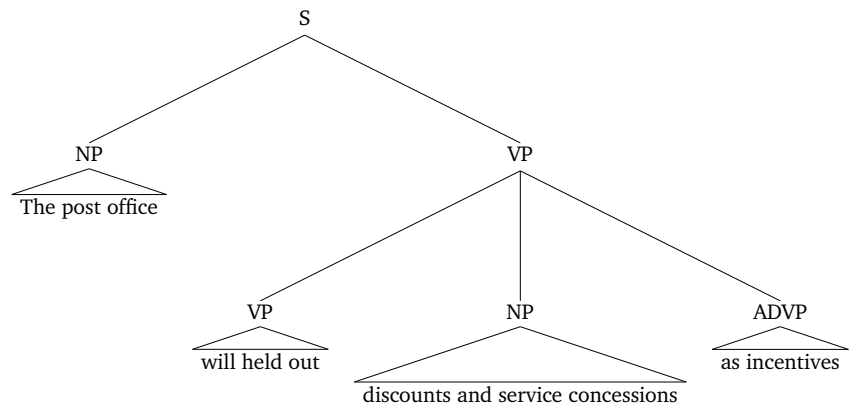


Interpretation 3: The announced program promotes safety in trucks. Vans are also announced.

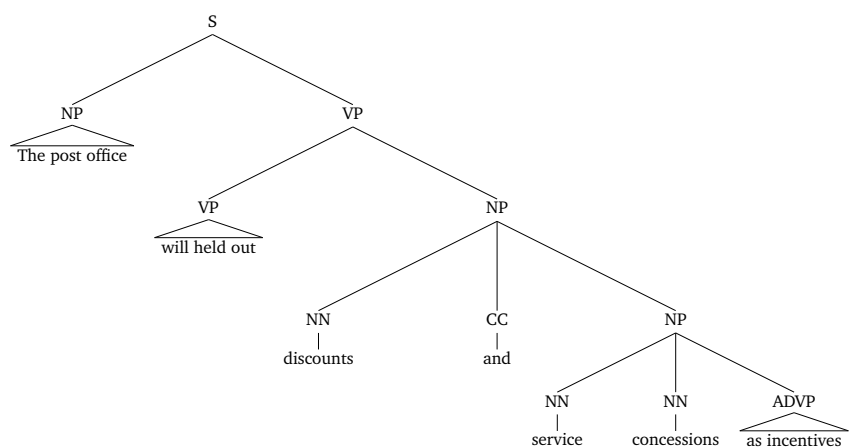


2. Find two interpretations for each sentence

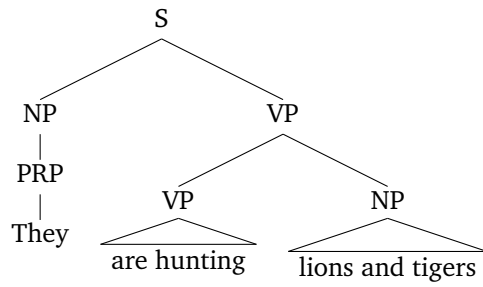
Sentence 1, Interpretation 1:  
Discounts and concessions are held out as incentives



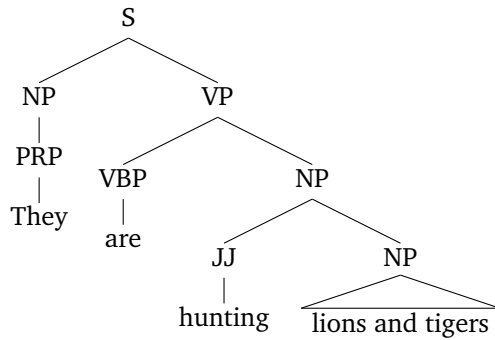
Sentence 1, Interpretation 2:  
Discounts and concessions are held out. Concessions look like incentives



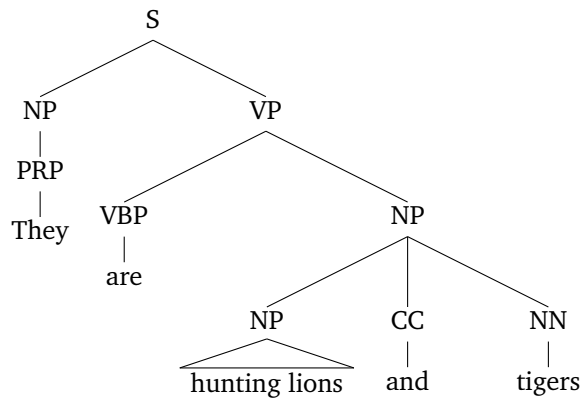
Sentence 2, Interpretation 1:  
Someone is hunting big felines.



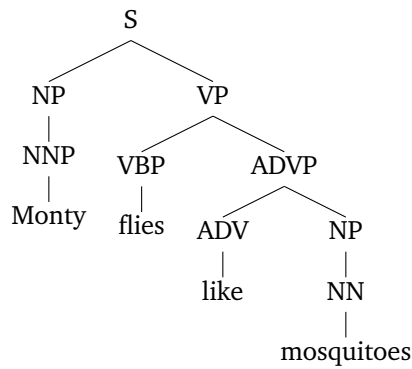
Sentence 2, Interpretation 2:  
Those animals are big felines that hunt.



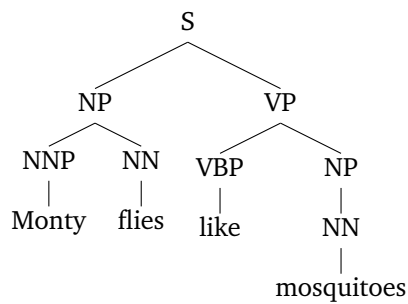
Sentence 2, Interpretation 3:  
Those animals are lions that hunt, and also tigers



Sentence 3, Interpretation 1:  
Someone named Monty moves through the air in the same way than mosquitoes do

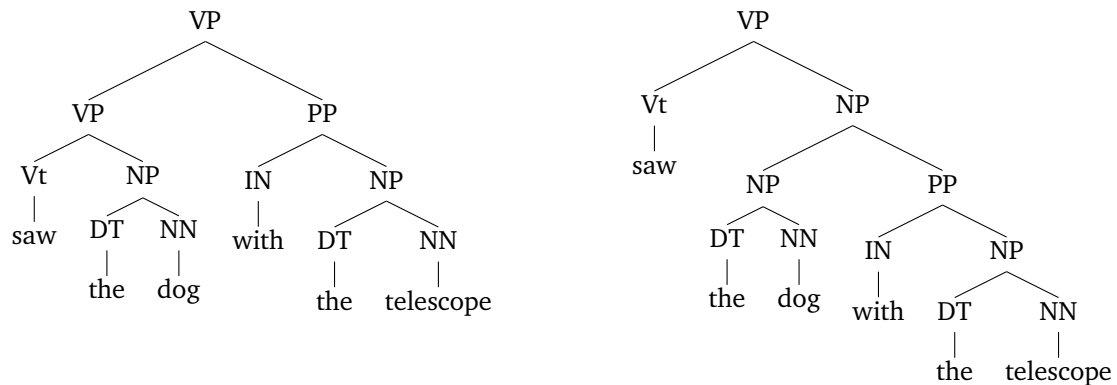


Sentence 3, Interpretation 2:  
Flies from a place named Monty are fond of mosquitoes



## Exercise 2.

Say we have the phrase *saw the dog with the telescope* and we are given the gold parse tree (left) and the predicted parse tree (right):



What are the precision and recall of this predicted parse tree?

## SOLUTION

The gold tree has 11 nonterminal nodes, each expanded using certain rule.

The predicted tree has 11 nonterminal nodes, each expanded using a rule that may be or not the right one. In particular, the predicted rules  $VP \rightarrow Vt NP$  and  $NP \rightarrow NP PP$  are not in the gold tree. All the other rules are found in the gold tree, so they are right.

Precision is the number of right rules divided by the number of predicted rules (i.e. size of the predicted tree). So in this case, it is  $9/11 = 81.8\%$ .

Recall is the number of right rules divided by the number of expected rules (i.e. size of the gold tree). So in this case, it is also  $9/11 = 81.8\%$  (since the size of both trees was the same, which shouldn't necessarily happen)

### Exercise 3.

Consider the following CFG:

$S \rightarrow NP VP$	$DT \rightarrow the$	$NN \rightarrow park$
$NP \rightarrow DT NN$	$NN \rightarrow man$	$VB \rightarrow saw$
$NP \rightarrow NP PP$	$NN \rightarrow dog$	$IN \rightarrow with$
$PP \rightarrow IN NP$	$NN \rightarrow cat$	$IN \rightarrow under$
$VP \rightarrow VB NP$		

1. How many parse trees are there under this grammar for the sentence *the man saw the dog in the park* ?
2. How many parse trees are there under this grammar for the sentence *the man saw the dog in the park with the cat* ?
3. Consider a sentence that is grammatical under the above context-free grammar, and has exactly  $k$  prepositions following the verb, and 0 prepositions before the verb (a preposition is any word with the tag IN). How many parse trees will this sentence have? Reason why.

The  $n^{th}$  Catalan number is defined as  $C_n = \frac{(2n)!}{(n+1)!n!}$  (see Wikipedia for a full description).  
It can be shown that  $C_n$  is the number possible different binary trees with  $n + 1$  leaves.

### SOLUTION

1. The sentence *the man saw the dog in the park* has a unique analysis in this grammar, where the dog is in the park. This is because the grammar does not have a rule such as e.g.  $VP \rightarrow VP PP$  that allows the PP *in the park* to be attached to the verb *saw*.
2. The sentence *the man saw the dog in the park with the cat* has two analysis under this grammar: One where the cat is with the dog, and another where the cat is with the park.
3. If we have  $k$  prepositions after the verb, it means there are  $k + 1$  nouns: the first noun after the verb (e.g. *dog* in the previous sentences) and then one noun after each preposition (*park*, *cat*, etc.). Each new preposition we add, can be attached to any of the previous nouns. Thus, if we have one preposition, it can only be attached to the first noun after the verb (*dog* in the example). The second preposition we add can be attached to the first noun, or to the second, etc. Thus, potentially we could build  $k!$  combinations. However, not all combinations are valid because many of them have crossing arcs and do not correspond to correct constituency trees. Thus, the number of possible combinations is the number of valid binary trees that can be build with  $k + 1$  leaves, that is, the  $k^{th}$  Catalan number,  $C_k$ .

#### Exercise 4.

Consider the following CFG:

$S \rightarrow NP VP$	$DT \rightarrow the$	$NNS \rightarrow cats$
$NP \rightarrow DT NN$	$NN \rightarrow man$	$NNS \rightarrow parks$
$NP \rightarrow DT NNS$	$NN \rightarrow dog$	$VB \rightarrow see$
$NP \rightarrow NP PP$	$NN \rightarrow cat$	$VB \rightarrow sees$
$PP \rightarrow IN NP$	$NN \rightarrow park$	$IN \rightarrow in$
$VP \rightarrow VB NP$	$NNS \rightarrow dogs$	$IN \rightarrow with$
$VP \rightarrow VP PP$		

This grammar overgenerates incorrect English sentences, such as:

*the dog see the cat*  
*the dog in the park see the cat*  
*the dog in the park see the cat in the park*  
*the dogs sees the cat*  
*the dogs in the park sees the cat*  
*the dogs in the park sees the cat in the park*

1. Modify the grammar so that all generated sentences respect third-person subject-verb agreement rules for English

#### SOLUTION

The rule joining the subject and the verb of the sentences is  $S \rightarrow NP VP$ , so we need to alter this rule to allow only the combination of singular NP with third person VP, and plural NP with non-third person VP. For this, we need different rules for singular/plural NP and for third/non-third person VP.

Thus, the top rule  $S \rightarrow NP VP$  needs to be replaced with:

$S \rightarrow NP_s VP_s$   
 $S \rightarrow NP_p VP_p$

All the NP rules must distinguish both kinds of noun phrases, replacing them with:

$NP_s \rightarrow DT NN$   
 $NP_p \rightarrow DT NNS$   
 $NP_s \rightarrow NP_s PP$   
 $NP_p \rightarrow NP_p PP$

Finally, the rules for verb phrases must also distinguish both cases:

$VB_s \rightarrow sees$   
 $VB_p \rightarrow see$   
 $VP_s \rightarrow VB_s NP$   
 $VP_p \rightarrow VB_p NP$   
 $VP_s \rightarrow VP_s PP$   
 $VP_p \rightarrow VP_p PP$

To avoid an explosion of rules, we can keep a generic NP to be used for noun phrases after the verb or inside a PP:

$NP \rightarrow NP_s$   
 $NP \rightarrow NP_p$

## Exercise 5.

Consider the following CFG:

$S \rightarrow NP VP$	$DT \rightarrow \text{the}$	$VB \rightarrow \text{saw}$
$NP \rightarrow DT NN$	$NN \rightarrow \text{man}$	$IN \rightarrow \text{with}$
$PP \rightarrow IN NP$	$NN \rightarrow \text{dog}$	$IN \rightarrow \text{under}$
$VP \rightarrow VB NP$	$NN \rightarrow \text{telescope}$	
$VP \rightarrow VP PP$		

An infinite number of sentences can be generated by this grammar, for example:

*the man saw the dog*

*the man saw the dog with the telescope*

*the man saw the dog with the telescope under the dog*

*the man saw the dog under the telescope with the dog under the telescope*

etc.

The language  $\mathcal{L}(G)$  generated by a context-free grammar  $G$  is defined as the set of sentences that can be derived with a sequence of grammar rule applications.

A hidden Markov model (HMM), defines a distribution  $P(x_1 \dots x_n, y_1 \dots y_n)$  over sentences  $x_1 \dots x_n$  paired with PoS tag sequences  $y_1 \dots y_n$ .  
The language generated by a HMM is defined as the set of sentences  $x_1 \dots x_n$  such that:  
 $\max_{y_1 \dots y_n} P(x_1 \dots x_n, y_1 \dots y_n) > 0$ , that is, sentences with at least one possible PoS-tag sequence  $y_1 \dots y_n$  that gives a non-zero value for the probability  $P(x_1 \dots x_n, y_1 \dots y_n)$ .

1. Write a bigram HMM that generates the same language than the context-free grammar given above.

## SOLUTION

This grammar is very restrictive in the order of words, and we can observe that:

- DT is always followed by NN
- IN is always followed by DT
- VB is always followed by DT
- NN is followed by a VB the first time, or by IN if after the verb, or by the end of the sentence.

Since each token may have a small amount of possible tokens after it, it is relatively straightforward to model this as a bigram model. The only tricky part is that we need to distinguish the first NN which is followed by VB, since the sentence can have only one verb. For this, we will add two special PoS: DTs and NNs for the subject.

Then, the transition probabilities of the bigram HMM will be:

$P(\text{NNs} \text{DTs}) = a_{\text{DTs.NNs}} = 1$	(after the subject determiner always comes the subject noun)
$P(\text{VB} \text{NNs}) = a_{\text{NNs.VB}} = 1$	(after the subject noun always comes the verb)
$P(\text{DT} \text{VB}) = a_{\text{VB.DT}} = 1$	(after the verb always comes a regular determiner)
$P(\text{NN} \text{DT}) = a_{\text{DT.NN}} = 1$	(after a regular determiner always comes a regular noun))
$P(\text{IN} \text{NN}) = a_{\text{NN.IN}} = 0.5$	(after a regular noun may come a preposition))
$P(\text{EoS} \text{NN}) = a_{\text{NN.EoS}} = 0.5$	(after a regular noun the sentence may end)
$P(\text{DT} \text{IN}) = a_{\text{IN.DT}} = 1$	(after a preposition always comes a regular determiner)

The initial probabilities are:

$$\pi(\text{DTs}) = 1 \quad (\text{the sentence always starts with a subject determiner})$$



And the emission probabilities are:

$$P(\text{the}|\text{DTs}) = b_{\text{the.DTs}} = 1$$

$$P(\text{the}|\text{DT}) = b_{\text{the.DT}} = 1$$

(The only determiner in the grammar is *the*, so it will be emitted both for subject and regular determiners)

$$P(\text{man}|\text{NNs}) = b_{\text{man.NNs}} = 0.33$$

$$P(\text{dog}|\text{NNs}) = b_{\text{dog.NNs}} = 0.33$$

$$P(\text{telescope}|\text{NNs}) = b_{\text{telescope.NNs}} = 0.33$$

(The subject may be any of the three nouns existing in the grammar)

$$P(\text{man}|\text{NN}) = b_{\text{man.NN}} = 0.33$$

$$P(\text{dog}|\text{NN}) = b_{\text{dog.NN}} = 0.33$$

$$P(\text{telescope}|\text{NN}) = b_{\text{telescope.NN}} = 0.33$$

(Any regular noun may be any of the three nouns existing in the grammar)

$$P(\text{in}|\text{IN}) = b_{\text{in.IN}} = 0.5$$

$$P(\text{with}|\text{IN}) = b_{\text{with.IN}} = 0.5$$

(There are two possible prepositions in the grammar)

$$P(\text{saw}|\text{VB}) = b_{\text{saw.VB}} = 1$$

(The verb is *saw* in all sentences of this grammar)

### Exercise 6.

Consider the following CFG:

- |                            |                        |
|----------------------------|------------------------|
| $S \rightarrow NP VP$      | $WH \rightarrow that$  |
| $VP \rightarrow Vt NP$     | $DT \rightarrow the$   |
| $VP \rightarrow Vdt NP NP$ | $NN \rightarrow man$   |
| $NP \rightarrow DT NN$     | $NN \rightarrow dog$   |
| $NP \rightarrow NP RELC$   | $NN \rightarrow cat$   |
| $RELC \rightarrow WH SGAP$ | $NN \rightarrow park$  |
| $SGAP \rightarrow VP$      | $Vt \rightarrow saw$   |
| $SGAP \rightarrow NP VGAP$ | $Vdt \rightarrow gave$ |
| $VGAP \rightarrow Vt$      |                        |
| $VGAP \rightarrow Vdt NP$  |                        |

1. Draw parse trees for the sentences:

the man that saw the dog saw the cat  
the man that the cat saw saw the dog

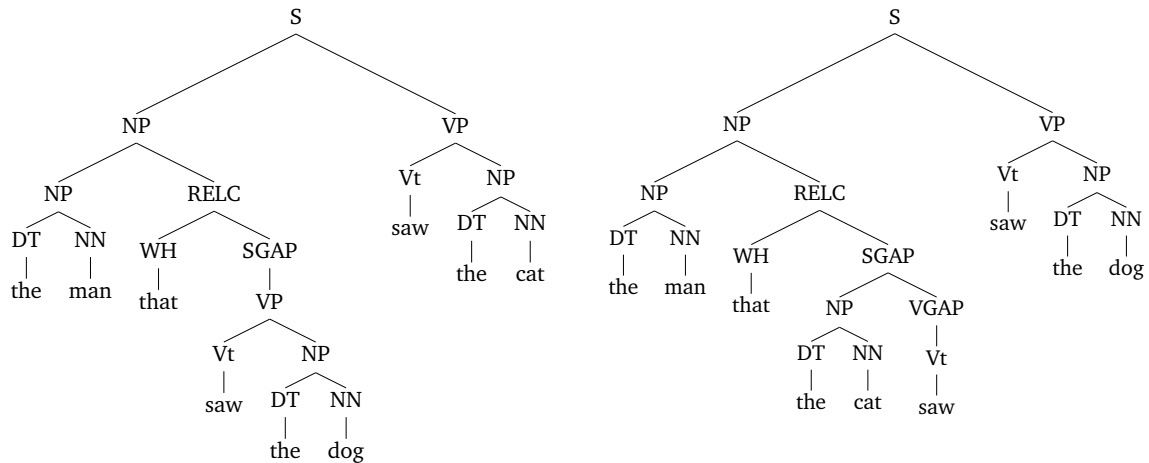
2. Write a sentence that is grammatical under the above grammar, and contains the trigram: *saw saw saw*. Draw the parse tree for the sentence.
3. Assume that we add the following rules to the grammar, so that the sentence *the man said the cat saw the dog* can be parsed correctly:

- $VP \rightarrow V3 S$   
 $V3 \rightarrow said$

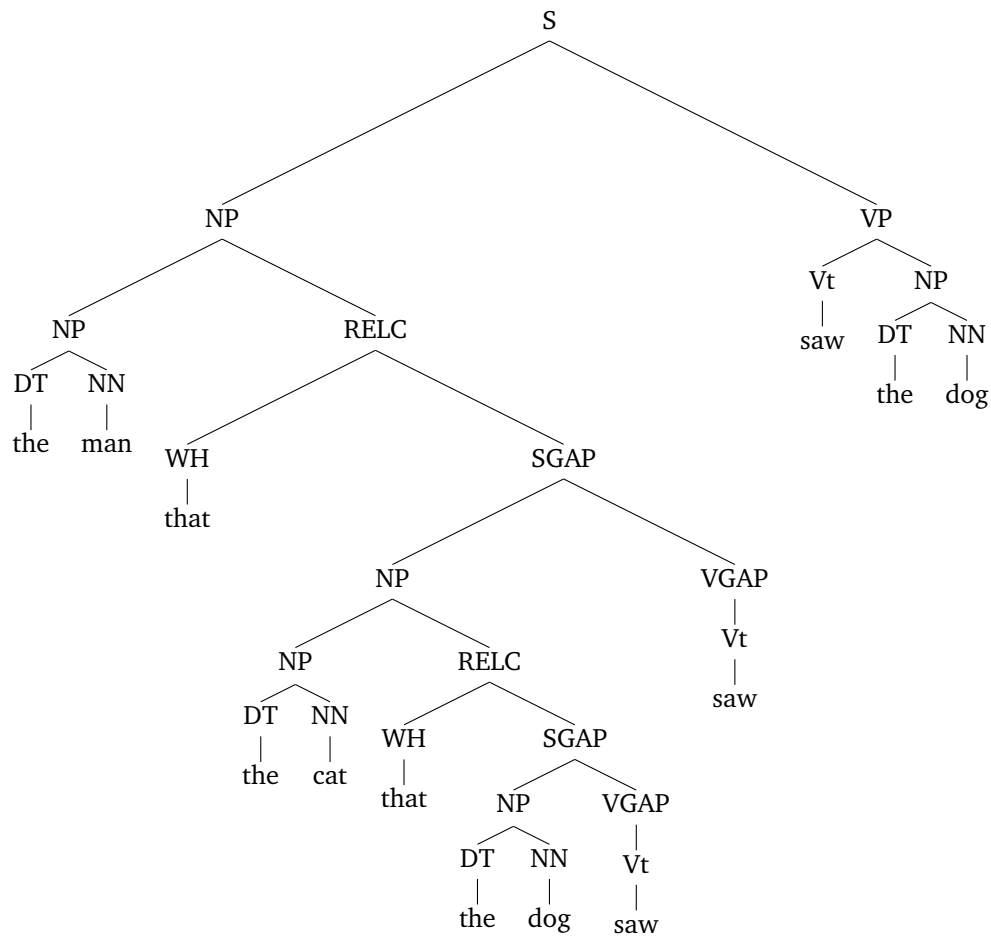
What additional rules should be added to the grammar so that the sentence *the dog that the man said the cat saw saw the park* can be parsed?

### SOLUTION

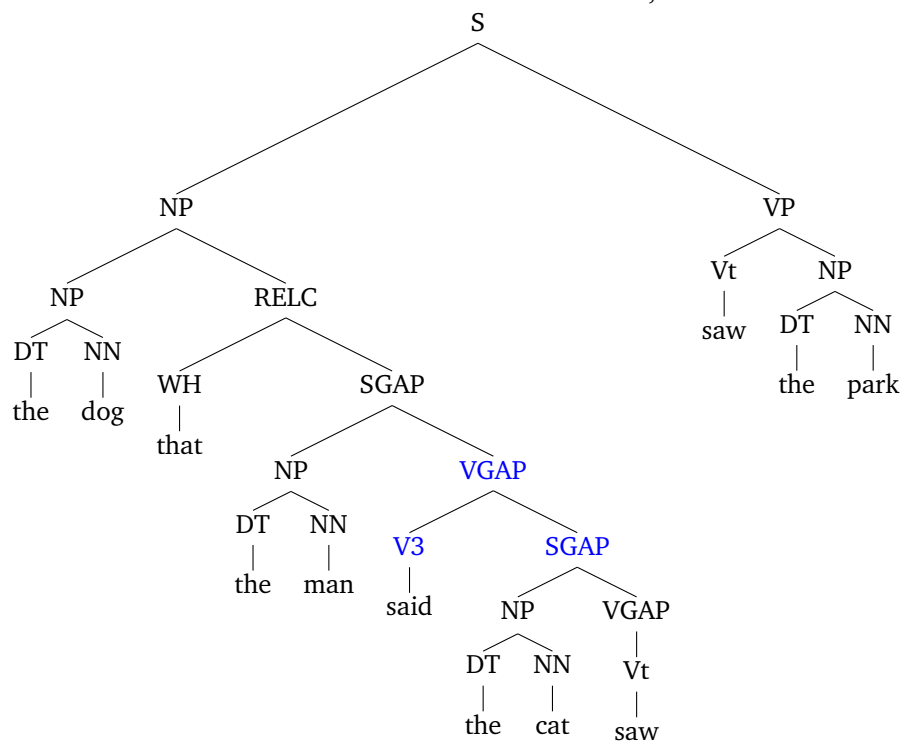
1.



2. the man that the cat that the dog saw saw saw the dog



3. We need to add an additional rule  $VGAP \rightarrow V3 SGAP$ , so we can obtain the tree:



the dog that the man said the cat saw saw the park

## Probabilistic Context Free Grammars

### Exercise 7.

Using the following PCFG in CNF:

$S \rightarrow NP VP$	1.0	$P \rightarrow with$	1.0
$NP \rightarrow NP PP$	0.4	$V \rightarrow saw$	1.0
$PP \rightarrow P NP$	1.0	$NP \rightarrow astronomers$	0.1
$VP \rightarrow V NP$	0.7	$NP \rightarrow ears$	0.18
$VP \rightarrow VP PP$	0.3	$NP \rightarrow saw$	0.04
		$NP \rightarrow stars$	0.18
		$NP \rightarrow telescopes$	0.1

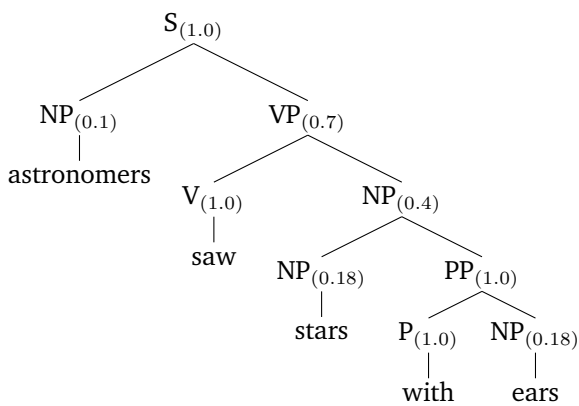
Work with the sentence: *astronomers saw stars with ears*

- How many correct parses are there for this sentence?
- Write them, along with their probabilities.

### SOLUTION

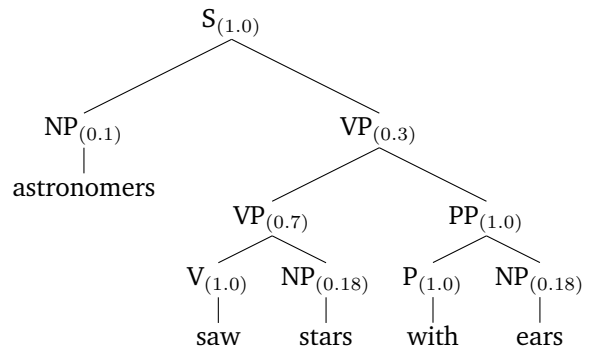
There are two possible parsers for this sentence according to the given grammar:

Option 1: (the stars had ears)



Probability:  $1.0 \times 0.1 \times 0.7 \times 1.0 \times 0.4 \times 0.18 \times 1.0 \times 1.0 \times 0.18 = 0.00091$

Option 2: (Astronomers had their ears while watching the stars –or used ears to watch them)



Probability:  $1.0 \times 0.1 \times 0.3 \times 0.7 \times 1.0 \times 0.18 \times 1.0 \times 1.0 \times 0.18 = 0.00068$

### Exercise 8.

Given the following PCFG:

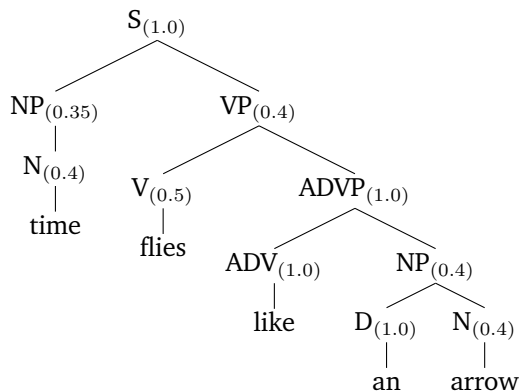
$S \rightarrow NP VP$	1.0	$N \rightarrow \text{time}$	0.4
$NP \rightarrow N N$	0.25	$N \rightarrow \text{flies}$	0.2
$NP \rightarrow D N$	0.4	$N \rightarrow \text{arrow}$	0.4
$NP \rightarrow N$	0.35	$D \rightarrow \text{an}$	1.0
$VP \rightarrow V NP$	0.6	$ADV \rightarrow \text{like}$	1.0
$VP \rightarrow V ADVP$	0.4	$V \rightarrow \text{flies}$	0.5
$ADVP \rightarrow ADV NP$	1.0	$V \rightarrow \text{like}$	0.5

and the sentence *time flies like an arrow*

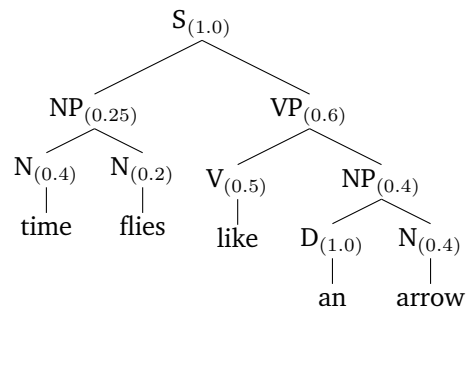
1. Write two parse trees that this grammar generates for this sentence
2. Compute the probability of each tree.
3. Convert the grammar to CNF and emulate the behaviour of the CKY algorithm on this sentence. Provide the final chart.
4. Emulate the behaviour of the Earley algorithm on this sentence (ignoring rule probabilities). Provide the final chart.

### SOLUTION

1. Option 1: (time goes by so fast that reminds of an arrow)



- Option 2: (Alien 4-dimensional flies are fond of arrows)



2. Option 1 probability:  $1.0 \times 0.35 \times 0.4 \times 0.4 \times 0.5 \times 1.0 \times 1.0 \times 0.4 \times 1.0 \times 0.4 = 0.00448$   
Option 2 Probability:  $1.0 \times 0.25 \times 0.4 \times 0.2 \times 0.6 \times 0.5 \times 0.4 \times 1.0 \times 0.4 = 0.00096$

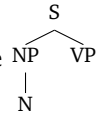

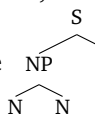
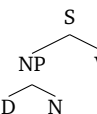
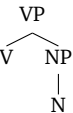

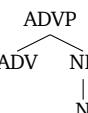
3. (a) Conversion of the grammar to CNF:

Chomsky Normal Form requires that all rules have a right hand side with exactly two non-terminals, or exactly one terminal. The rule  $NP \rightarrow N$  violates this condition (N is a non-terminal), so we need to remove it.

Once removed, we need to compensate for the lost combinations duplicating rules where NP appears on the right hand side, using N instead. Thus, the grammar is the following (blue rules have been added)

$S \rightarrow NP VP$	0.65	} 1.0	$N \rightarrow \text{time}$	0.4
$S \rightarrow N VP$	0.35		$N \rightarrow \text{flies}$	0.2
$NP \rightarrow N N$	0.385		$N \rightarrow \text{flies}$	0.2
$NP \rightarrow D N$	0.615		$N \rightarrow \text{arrow}$	0.4
$VP \rightarrow V NP$	0.39	} 0.6	$D \rightarrow \text{an}$	1.0
$VP \rightarrow V N$	0.21		$ADV \rightarrow \text{like}$	1.0
$VP \rightarrow V ADVP$	0.4		$V \rightarrow \text{flies}$	0.5
$ADVP \rightarrow ADV NP$	0.65	} 1.0	$V \rightarrow \text{like}$	0.5
$ADVP \rightarrow ADV N$	0.35			

We also need to redistribute rule probabilities. Recomputed probabilities are shown in red in the grammar above.

- Rule  $S \rightarrow NP VP$  had probability 1.0 in the original grammar. The partial tree  had probability  $1.0 \times 0.35$  in the original grammar. Thus, the rest of possible trees starting with  added up to 0.65. We can get the same distribution setting probability 0.35 for the new rule  $S \rightarrow N VP$  and probability 0.65 for the rest of cases.
- Non terminal NP had three rules in the original grammar, whose probabilities added up to 1.0. Now this non-terminal has one rule less, so we need to redistribute the probability mass among the remaining rules, but keeping the probabilities of the generated trees. In the original grammar, the tree  had probability  $1.0 \times 0.25$ , and the tree  had  $1.0 \times 0.4$ . Since we changed the probability of the top rule to 0.65, to keep the same distribution for the same trees, we need to set  $P(NP \rightarrow N N) = 0.25/0.65 = 0.385$  and  $P(NP \rightarrow D N) = 0.4/0.65 = 0.615$ .
- We splitted rule  $VP \rightarrow V NP$  (with original probability 0.6) in two. The partial tree  had probability  $0.6 \times 0.35 = 0.21$ , so the new rule  $VP \rightarrow V N$  must have this probability, and the rest of tree starting with  will have the remaining mass up to 0.6, i.e. 0.39.
- Similarly, rule  $ADVP \rightarrow ADV NP$  was splitted in two. The partial tree  had probability  $1.0 \times 0.35$ , so this will be the probability of the new rule for ADVP. The original rule will retain the rest of the mass (0.65).

(b) CKY chart:

					15 0.00448 $S \rightarrow N_{11}VP_{25}$ ( $0.35 \times 0.4 \times 0.032$ )
				14	25 0.032 $VP \rightarrow V_{22}ADVP_{35}$ ( $0.4 \times 0.5 \times 0.16$ )
			13	24	35 0.16 $ADVP \rightarrow ADV_{33}NP_{45}$ ( $0.65 \times 1.0 \times 0.246$ ) 0.048 $VP \rightarrow V_{33}NP_{45}$ ( $0.39 \times 0.5 \times 0.246$ )
	12		23	34	45 0.246 $NP \rightarrow D_{44}N_{55}$ ( $0.615 \times 1.0 \times 0.4$ )
11	22		33	44	55
0.4 $N \rightarrow \text{time}$	0.2 $N \rightarrow \text{flies}$ 0.5 $V \rightarrow \text{flies}$	1.0 $ADV \rightarrow \text{like}$ 0.5 $V \rightarrow \text{like}$	1.0 $D \rightarrow \text{an}$	0.4 $N \rightarrow \text{arrow}$	
time	flies	like	an	arrow	

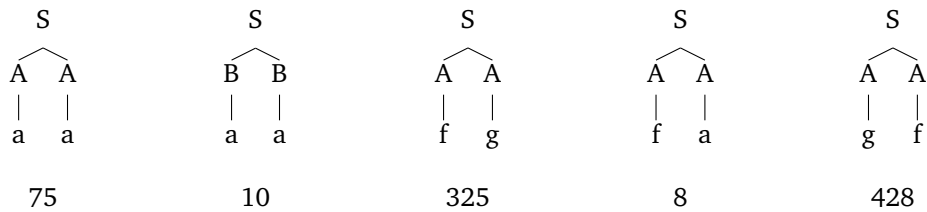
(Blue rule in cell 35 indicates the most likely subtree selected in that cell)

4. Earley chart:

					<b>chart[5]</b> [0,5] $S \rightarrow NP_{01}VP_{15} \bullet$ $S \rightarrow NP_{02}VP_{25} \bullet$
				<b>chart[4]</b>	[0,4] [1,5] $VP \rightarrow V_{12}ADVP_{25} \bullet$
			<b>chart[3]</b>	[0,3]	[1,4] [2,5] $ADVP \rightarrow ADV_{23}NP_{35} \bullet$ $VP \rightarrow V_{23}NP_{35} \bullet$
		<b>chart[2]</b>		[1,3]	[2,4] [3,5] $NP \rightarrow D_{34}N_{45} \bullet$
	<b>chart[1]</b>			[0,2]	[1,2] $NP \rightarrow N_{01}N_{12} \bullet$ $S \rightarrow NP_{02} \bullet VP$
	[0,1]	[1,2]	[2,3]	[3,4]	[4,5]
	$N \rightarrow \text{time} \bullet$ $NP \rightarrow N_{01} \bullet N$ $NP \rightarrow N_{01} \bullet$ $S \rightarrow NP_{01} \bullet VP$	$N \rightarrow \text{flies} \bullet$ $V \rightarrow \text{flies} \bullet$ $VP \rightarrow V_{12} \bullet NP$ $VP \rightarrow V_{12} \bullet ADVP$	$ADV \rightarrow \text{like} \bullet$ $V \rightarrow \text{like} \bullet$ $ADVP \rightarrow ADV_{23} \bullet NP$ $VP \rightarrow V_{23} \bullet NP$ $VP \rightarrow V_{23} \bullet ADVP$	$D \rightarrow \text{an} \bullet$ $NP \rightarrow D \bullet N$	$N \rightarrow \text{arrow} \bullet$
<b>chart[0]</b>	[1,1]	[2,2]	[3,3]	[4,4]	[5,5]
$\gamma \rightarrow \bullet S$ $S \rightarrow \bullet NP VP$ $NP \rightarrow \bullet N N$ $NP \rightarrow \bullet D N$ $NP \rightarrow \bullet N$	$VP \rightarrow \bullet V NP$ $VP \rightarrow \bullet V ADVP$	$NP \rightarrow \bullet N N$ $NP \rightarrow \bullet D N$ $NP \rightarrow \bullet N$ $ADVP \rightarrow \bullet ADV NP$ $VP \rightarrow \bullet V NP$ $VP \rightarrow \bullet V ADVP$	$NP \rightarrow \bullet N N$ $NP \rightarrow \bullet D N$ $NP \rightarrow \bullet N$ $ADVP \rightarrow \bullet ADV NP$		
0	1	2	3	4	5
time	flies	like	an	arrow	

### Exercise 9.

Consider that you have as a training corpus a treebank containing the following trees. Each tree was observed the number of times indicated below it.



1. What PCFG would one get from this treebank (using MLE)?
2. Given the obtained grammar:
  - What is the most likely parse of the string  $aa$ ?
  - Is this a reasonable result? Discuss why.

### SOLUTION

1. The given collection of training trees, taking into account the number of repetitions of each, will produce the following counts of rule applications:

$S \rightarrow AA$	$1 \times 75 + 0 \times 10 + 1 \times 325 + 1 \times 8 + 1 \times 428 = 836$
$S \rightarrow BB$	$0 \times 75 + 1 \times 10 + 0 \times 325 + 0 \times 8 + 0 \times 428 = 10$
$S \rightarrow \text{anything}$	$1 \times 75 + 1 \times 10 + 1 \times 325 + 1 \times 8 + 1 \times 428 = 846$
$B \rightarrow a$	$0 \times 75 + 2 \times 10 + 0 \times 325 + 0 \times 8 + 0 \times 428 = 20$
$B \rightarrow \text{anything}$	$0 \times 75 + 2 \times 10 + 0 \times 325 + 0 \times 8 + 0 \times 428 = 20$
$A \rightarrow a$	$2 \times 75 + 0 \times 10 + 0 \times 325 + 1 \times 8 + 0 \times 428 = 158$
$A \rightarrow f$	$0 \times 75 + 0 \times 10 + 1 \times 325 + 1 \times 8 + 1 \times 428 = 761$
$A \rightarrow g$	$0 \times 75 + 0 \times 10 + 1 \times 325 + 0 \times 8 + 1 \times 428 = 753$
$A \rightarrow \text{anything}$	$2 \times 75 + 0 \times 10 + 2 \times 325 + 2 \times 8 + 2 \times 428 = 1672$

Thus, the MLE probability for each rule would be:

$P(S \rightarrow AA) = P(AA S) = \#(S \rightarrow AA)/\#(S \rightarrow \text{anything}) = 836/846 = 0.988$
$P(S \rightarrow BB) = P(BB S) = \#(S \rightarrow BB)/\#(S \rightarrow \text{anything}) = 10/846 = 0.012$
$P(B \rightarrow a) = P(a B) = \#(B \rightarrow a)/\#(B \rightarrow \text{anything}) = 20/20 = 1.000$
$P(A \rightarrow a) = P(a A) = \#(A \rightarrow a)/\#(A \rightarrow \text{anything}) = 158/1672 = 0.095$
$P(A \rightarrow f) = P(f A) = \#(A \rightarrow f)/\#(A \rightarrow \text{anything}) = 761/1672 = 0.455$
$P(A \rightarrow g) = P(g A) = \#(A \rightarrow g)/\#(A \rightarrow \text{anything}) = 753/1672 = 0.450$

2. The input sequence  $aa$  can be derived by the obtained grammar in only two ways, which correspond to the first two trees in the training data.

The first tree has probability:  $P(S \rightarrow AA) \times P(A \rightarrow a) \times P(A \rightarrow a) = 0.988 \times 0.095 \times 0.095 = 0.009$

The second tree has probability:  $P(S \rightarrow BB) \times P(B \rightarrow a) \times P(B \rightarrow a) = 0.012 \times 1.000 \times 1.000 = 0.012$

So, the most likely parse tree is the second one.

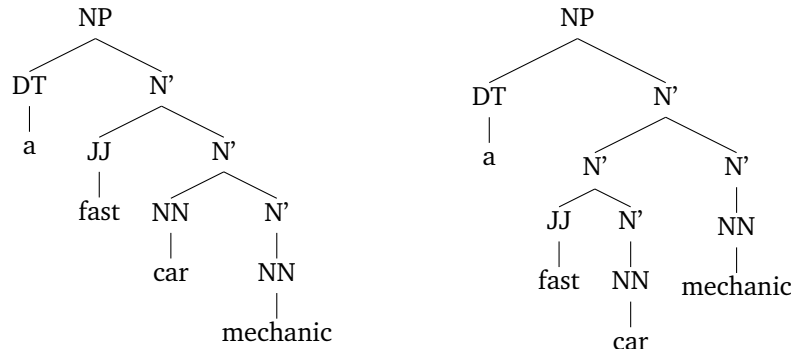
The first tree appears 75 times in the training data, while the second one occurs only 10 times, so one would expect the probability for the former to be higher. However, we do not compute the tree probability by counting how many times the whole tree occurs, but we just approximate it by multiplying the individual rule probabilities. Thus, the fact that B produces a with much higher probability than A is biasing the result.

This is due to the reduced amount of training data combined with the use of MLE: Since we do not perform any smoothing to consider the possibility of B producing other symbols, we are overestimating the probability of the rule  $B \rightarrow a$ .



### Exercise 10.

Consider the two following parse trees:



Discuss whether the following statements are true or false and why:

1. The two parse trees receive the same probability under any PCFG
2. The first parse tree receives higher probability if  $P(N' \rightarrow NN N') > P(N' \rightarrow N' N')$
3. The first parse tree receives higher probability if  $P(N' \rightarrow NN N') > P(N' \rightarrow N' N') + P(N' \rightarrow NN)$

### SOLUTION

1. False, since the set of rules used in each tree differ: left tree uses rule  $N' \rightarrow NN N'$  while the tree on the right uses the rule  $N' \rightarrow N' N'$ . Also, the former uses rule  $N' \rightarrow NN$  once but the latter uses it twice. So, assuming  $Q$  is the product of probabilities of rules shared by both trees, the first tree has probability  $Q \times P(N' \rightarrow NN N')$ , and the second has probability  $Q \times P(N' \rightarrow N' N') \times P(N' \rightarrow NN)$ , which are not necessarily equal.
2. True, since under this condition  $Q \times P(N' \rightarrow NN N') > Q \times P(N' \rightarrow N' N')$ . If we multiply the right hand side term by  $P(N' \rightarrow NN)$  which is smaller than 1, the difference will increase.
3. False, since probabilities are multiplied, not added. The first tree would have higher probability if  $P(N' \rightarrow NN N') > P(N' \rightarrow N' N') \times P(N' \rightarrow NN)$

### Exercise 11.

Consider the following PCFG:

$S \rightarrow V N$	0.6
$S \rightarrow D N$	0.4
$D \rightarrow a$	0.2
$D \rightarrow the$	0.8
$N \rightarrow president$	1.0
$V \rightarrow support$	0.6
$V \rightarrow hate$	0.4

1. List all sentences generated by this grammar, along with their probability
2. Define a bigram language model that gives the same probability distribution  $p(x)$  than the PCFG shown above. The vocabulary of the language model should be  $\Sigma = \{a, the, president, support, hate\}$ . Specify the value for each parameter of the language model.

A bigram language model consists of a finite vocabulary  $\Sigma$ , and a parameter  $q(u, v)$  for each bigram  $(u, v)$  such that  $u \in \Sigma \cup \{\text{START}\}$  and  $v \in \Sigma \cup \{\text{STOP}\}$ . The value for  $q(u, v)$  can be interpreted as the probability of seeing word  $v$  immediately after word  $u$ , i.e.  $P(v|u)$ . For any sentence  $x_1, \dots, x_n$  where  $x_i \in \Sigma$ , the probability of the sentence under the bigram language model is  $p(x_1, \dots, x_n) = \prod_{i=1}^{n+1} q(x_{i-1}, x_i)$ , where we define  $x_0 = \text{START}$  and  $x_{n+1} = \text{STOP}$ .

### SOLUTION

1. All possible sequences generated by this grammar are:

support president	$0.6 \times 0.6 \times 1.0 = 0.36$
hate president	$0.6 \times 0.4 \times 1.0 = 0.24$
a president	$0.4 \times 0.2 \times 1.0 = 0.08$
the president	$0.4 \times 0.8 \times 1.0 = 0.32$

2. The bigram model that assigns the same probabilities to the above sequences can be derived as follows:

$q(\text{START}, \text{support}) = 0.36$	(the only sentence starting with <i>support</i> has probability 0.36)
$q(\text{START}, \text{hate}) = 0.24$	(the only sentence starting with <i>hate</i> has probability 0.24)
$q(\text{START}, a) = 0.08$	(the only sentence starting with <i>a</i> has probability 0.08)
$q(\text{START}, \text{the}) = 0.32$	(the only sentence starting with <i>the</i> has probability 0.32)
$q(\text{support}, \text{president}) = 1.0$	(The only possible word after <i>support</i> is <i>president</i> )
$q(\text{hate}, \text{president}) = 1.0$	(The only possible word after <i>hate</i> is <i>president</i> )
$q(a, \text{president}) = 1.0$	(The only possible word after <i>a</i> is <i>president</i> )
$q(\text{the}, \text{president}) = 1.0$	(The only possible word after <i>the</i> is <i>president</i> )
$q(\text{president}, \text{STOP}) = 1.0$	(after <i>president</i> , the sentence ends)

This bigram model generates the same four sequences with the same probabilities than the given grammar.

## Exercise 12.

Consider the following PCFG

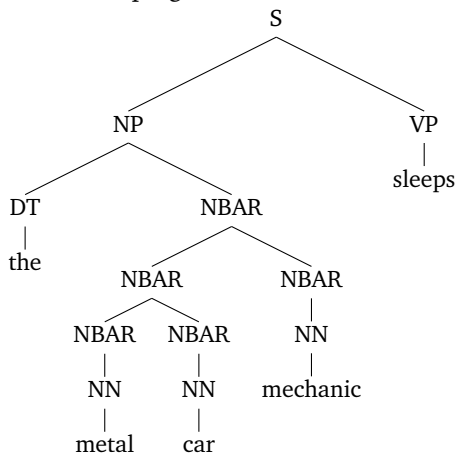
$S \rightarrow NP VP$	1.0	$VP \rightarrow sleeps$	1.0
$NP \rightarrow DT NBAR$	1.0	$DT \rightarrow the$	1.0
$NBAR \rightarrow NN$	0.7	$NN \rightarrow mechanic$	0.1
$NBAR \rightarrow NBAR NBAR$	0.3	$NN \rightarrow car$	0.2
		$NN \rightarrow metal$	0.7

1. What is the parse tree with highest probability for the sentence *the metal car mechanic sleeps* ?
2. Modify the grammar above so that the sentence *the human language technology rules* has two interpretations (one about *human language* and another about *human technology*). Draw the trees for both interpretations, and point out which is the most likely.

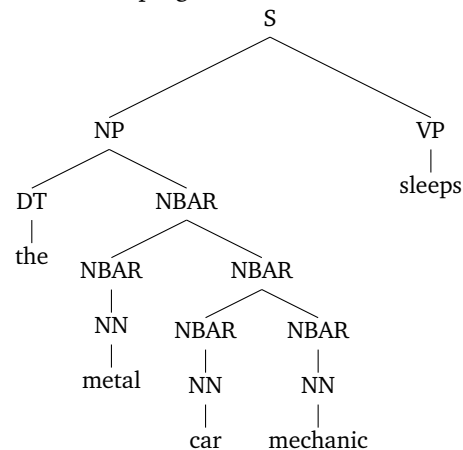
## SOLUTION

1. This grammar produces two possible trees for the given sentence:

Option 1: (the mechanic that works on metal cars is sleeping)



Option 2: (the metal mechanic that works on cars is sleeping)



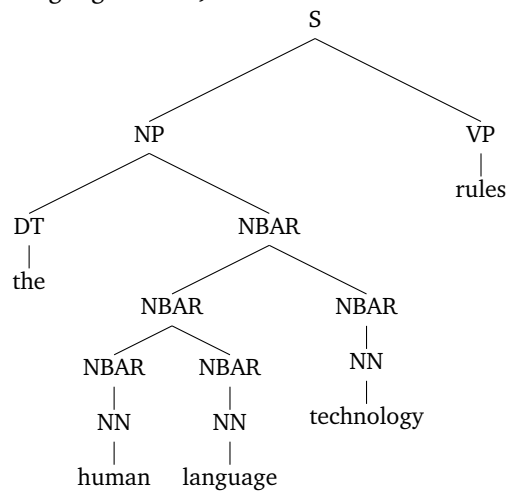
Both trees use once the rules  $S \rightarrow NP VP$  and  $NP \rightarrow DT NBAR$ , twice the rule  $NBAR \rightarrow NBAR NBAR$ , and three times the rule  $NBAR \rightarrow NN$ . Also, the rules producing the leaves are also the same. The only difference is the order in which the rules are applied. Thus, the probabilities of both trees are identical, and there is not one single best tree, but two.

2. The grammar already allows the ambiguous structure for the sequence NN NN NN. We only need to add the new words to the grammar, and fix the probabilities. Rule probabilities are invented, since we do not have training data, but we need to ensure that the rules for the same non-terminal symbol add up to 1. New rules are highlighted in red. Redistributed probabilities are shown in blue.

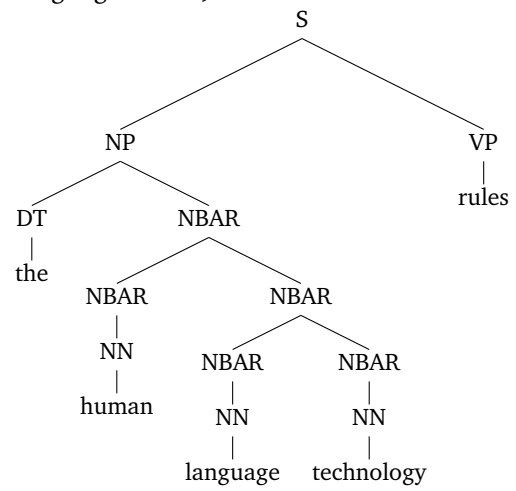
$S \rightarrow NP VP$	1.0	$VP \rightarrow sleeps$	0.5	$VP \rightarrow rules$	0.5
$NP \rightarrow DT NBAR$	1.0	$DT \rightarrow the$	1.0		
$NBAR \rightarrow NN$	0.7	$NN \rightarrow mechanic$	0.1	$NN \rightarrow language$	0.2
$NBAR \rightarrow NBAR NBAR$	0.3	$NN \rightarrow car$	0.2	$NN \rightarrow technology$	0.2
		$NN \rightarrow metal$	0.2	$NN \rightarrow human$	0.1

Possible trees for the new sentence are :

Option 1: (technology that deals with human language is cool)



Option 2: (human technology that deals with language is cool)



Again, since both trees have the same rules, they have exactly the same probability.

### Exercise 13.

This exercise considers several forms of language models that compute the probability of sentences  $P(\mathbf{x})$ . In each of the following cases you need to write an expression that indicates how the particular language model computes the probability  $P(\mathbf{x})$ , making clear what parameters of the model are used to compute the probability for the example sentence.

1.  **$n$ -gram language models.** The model considers only the words of  $\mathbf{x}$ , the rest of the linguistic structure is ignored. Write the expression for  $n = 2$  and  $n = 3$ .
2. **Hidden Markov Models (HMM).** The model represents pos tags in the state sequence and words in the observation sequence. The syntactic tree is ignored. Write the expression of  $P(\mathbf{x})$  for a bigram HMM, where states correspond to single PoS tags, and for a trigram HMM, where states correspond to two adjacent pos tags.
3. **Probabilistic Context-Free Grammars (PCFG).** The model considers the full syntactic tree.

### SOLUTION

1. The probability of a sequence  $\mathbf{x}$  according to a bigram language model is computed as:

$$P(\mathbf{x}) = \prod_{i=1}^n q(x_{i-1}, x_i) \quad \text{where we define } x_0 = \text{START for any sequence } \mathbf{x}_{[1:n]}. \\ q(v, w) \text{ is the parameter model corresponding to the probability} \\ \text{of word } w \text{ occurring just after word } v, q(v, w) = P(w|v)$$

The probability of a sequence  $\mathbf{x}$  according to a trigram language model is computed as:

$$P(\mathbf{x}) = \prod_{i=1}^n q(x_{i-2}, x_{i-1}, x_i) \quad \text{where we define } x_{-1} = x_0 = \text{START for any sequence } \mathbf{x}_{[1:n]}. \\ q(u, v, w) \text{ is the parameter model corresponding to the prob-} \\ \text{ability of word } w \text{ occurring just after words } uv, q(u, v, w) = \\ P(w|uv)$$

2. The probability of a sequence  $\mathbf{x}$  according to a bigram HMM is computed as:

$$P(\mathbf{x}) = \sum_{t_{[1:n]} \in T^n} \pi_{t_1} b_{t_1 x_1} \prod_{i=2}^n a_{t_{i-1} t_i} b_{t_i x_i}$$

where  $\pi_t$  is the probability of starting a sentence with tag  $t$ ,  $b_{tx}$  is the probability that tag  $t$  produces word  $x$ , and  $a_{tr}$  is the probability that after tag  $t$  comes tag  $r$  (i.e.  $P(r|t)$ ). The probability of the sequence is the sum of the probabilities of  $\mathbf{x}$  being produced by any possible PoS tag sequence of length  $n$ .

The probability of a sequence  $\mathbf{x}$  according to a trigram language model is computed as:

$$P(\mathbf{x}) = \sum_{t_{[1:n]} \in T^n} \pi_{t_0 t_1} b_{t_0 t_1 x_1} \prod_{i=2}^n a_{t_{i-2} t_{i-1} t_i} b_{t_{i-1} t_i x_i}$$

where we define  $t_0 = \text{START}$  for any sequence  $\mathbf{x}_{[1:n]}$ .  $\pi_{t_0 t}$  is the probability of starting a sentence with tag  $t$ ,  $b_{stx}$  is the probability word  $x$  is produced when the last two tags are  $s$  and  $t$ , and  $a_{str}$  is the probability that after tag  $r$  comes after tags  $s$  and  $t$  (i.e.  $P(r|st)$ ).

The probability of the sequence is the sum of the probabilities of  $\mathbf{x}$  being produced by any possible PoS tag sequence of length  $n$ .

3. The probability of a sequence  $\mathbf{x}$  according to a PCFG is computed as:

$$P(\mathbf{x}) = \sum_{t \in \mathcal{T}(\mathbf{x})} \prod_{r \in t} q(r)$$

That is, the probability of  $\mathbf{x}$  is the sum of the probabilities of all trees that generate  $\mathbf{x}$  under the PCFG,  $\mathcal{T}(\mathbf{x})$ . The probability of each tree  $t$  is the product of the probabilities of the rules used to build it,  $q(r)$ .