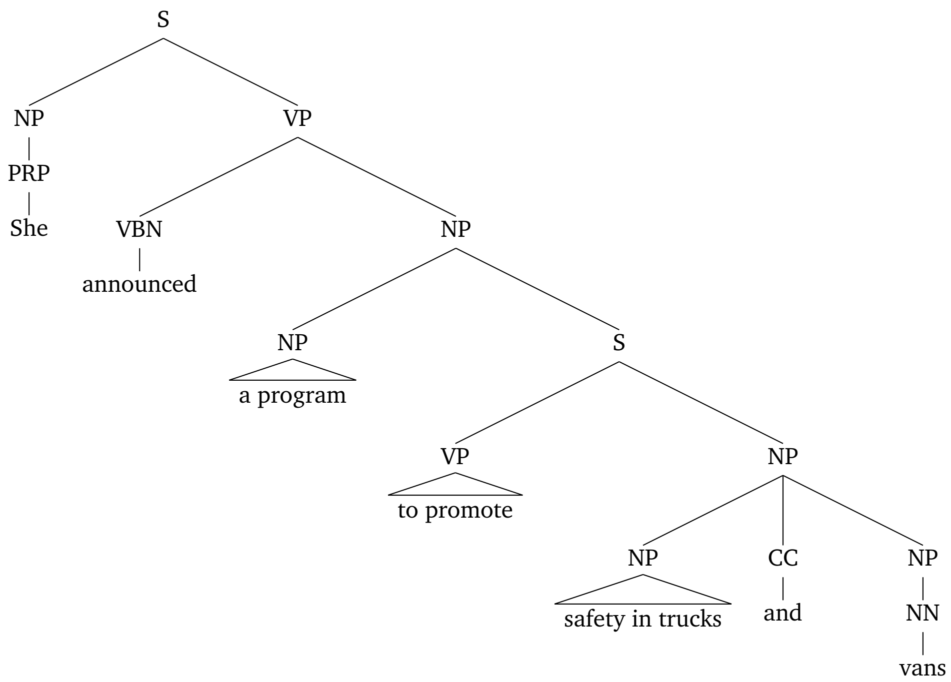
Advanced Human Language Technologies
Exercises on Parsing

Context Free Grammars

Exercise 1.

Exercise 2.

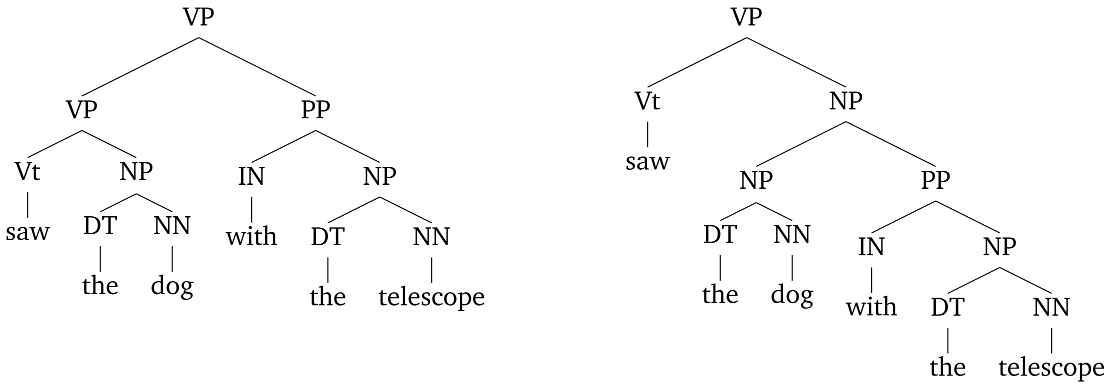
Consider the sentence *She announced a program to promote safety in trucks and vans*
and the following syntactic tree of one of its possible interpretations, in which the program promotes safety in trucks, and also promotes vans:



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*

Exercise 3.

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?

Exercise 4.

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.

Exercise 5.

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 Englihsh 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

Exercise 6.

Consider the following CFG:

S → NP VP	DT → the	VB → saw
NP → DT NN	NN → man	IN → with
PP → IN NP	NN → dog	IN → under
VP → VB NP	NN → telescope	
VP → 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: $\max_{y_1 \cdots y_n} P(x_1 \dots x_n, y_1 \dots y_n) > 0$, 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.

Exercise 7.

Consider the following CFG:

S → NP VP	WH → that
VP → Vt NP	DT → the
VP → Vdt NP NP	NN → man
NP → DT NN	NN → dog
NP → NP RELC	NN → cat
RELC → WH SGAP	NN → park
SGAP → VP	Vt → saw
SGAP → NP VGAP	Vdt → gave
VGAP → Vt	
VGAP → 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:

$$\text{VP} \rightarrow \text{V3 S}$$
$$\text{V3} \rightarrow \text{said}$$

   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?

Probabilistic Context Free Grammars

Exercise 8.

Using the following PCFG in CNF:

S → NP VP	1.0	P → with	1.0
NP → NP PP	0.4	V → saw	1.0
PP → P NP	1.0	NP → astronomers	0.1
VP → V NP	0.7	NP → ears	0.18
VP → VP PP	0.3	NP → saw	0.04
		NP → stars	0.18
		NP → 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.

Exercise 9.

Given the following PCFG:

S → NP VP	1.0	N → time	0.4
NP → N N	0.25	N → flies	0.2
NP → D N	0.4	N → arrow	0.4
NP → N	0.35	D → an	1.0
VP → V NP	0.6	ADV → like	1.0
VP → V ADVP	0.4	V → flies	0.5
ADVP → ADV NP	1.0	V → 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.

Exercise 10.

Consider that you have as a training corpus a treebank containing the following trees:

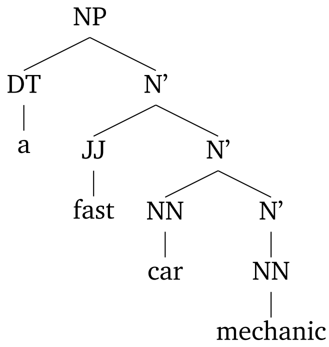
S	S	S	S	S
A A	B B	A A	A A	A A
a a	a a	f g	f a	g f
(1)	(2)	(3)	(4)	(5)

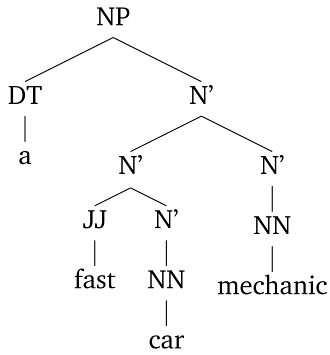
Assume that (1) appears 75 times in the training corpus, (2) occurs 10 times, (3) occurs 325 times, (4) appears 8 times and (5) appears 428 times.

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 *a a*?
   - Is this a reasonable result? Discuss why.

Exercise 11.

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 $P(\mathrm{N}' \to \mathrm{NN}\ \mathrm{N}') > P(\mathrm{N}' \to \mathrm{N}'\ \mathrm{N}')$
3. The first parse tree receives higher probability if $P(\mathrm{N}' \to \mathrm{NN}\ \mathrm{N}') > P(\mathrm{N}' \to \mathrm{N}'\ \mathrm{N}') + P(\mathrm{N}' \to \mathrm{NN})$

Exercise 12.

Consider the following PCFG:

$\mathrm{S} \to \mathrm{V}\ \mathrm{N}$	0.6
$\mathrm{S} \to \mathrm{D}\ \mathrm{N}$	0.4
$\mathrm{D} \to \mathrm{a}$	0.2
$\mathrm{D} \to \mathrm{the}$	0.8
$\mathrm{N} \to \mathrm{president}$	1.0
$\mathrm{V} \to \mathrm{support}$	0.6
$\mathrm{V} \to \mathrm{hate}$	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 $\Sigma = \{\mathrm{a}, \mathrm{the}, \mathrm{president}, \mathrm{support}, \mathrm{hate}\}$. Specify the value for each parameter of the language model.

A bigram language model consists of a finite vocabulary Σ, and a parameter $q(u, v)$ for each bigram (u, v) such that $u \in \Sigma \cup \{\mathrm{START}\}$ and $v \in \Sigma \cup \{\mathrm{STOP}\}$
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 $x_1, \ldots, x_n$ where $x_i \in \Sigma$, the probability of the sentence under the bigram language model is $p(x_1, \ldots, x_n) = \prod_{i=1}^{n+1} q(x_{i-1}, x_i)$, where we define $x_0 = \mathrm{START}$ and $x_{n+1} = \mathrm{STOP}$.

Exercise 13.

Consider the following PCFG

$\mathrm{S} \to \mathrm{NP}\ \mathrm{VP}$	1.0	$\mathrm{VP} \to \mathrm{sleeps}$	1.0
$\mathrm{NP} \to \mathrm{DT}\ \mathrm{NBAR}$	1.0	$\mathrm{DT} \to \mathrm{the}$	1.0
$\mathrm{NBAR} \to \mathrm{NN}$	0.7	$\mathrm{NN} \to \mathrm{mechanic}$	0.1
$\mathrm{NBAR} \to \mathrm{NBAR}\ \mathrm{NBAR}$	0.3	$\mathrm{NN} \to \mathrm{car}$	0.2
		$\mathrm{NN} \to \mathrm{metal}$	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 sentece *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.

Exercise 14.

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 $n = 3$.

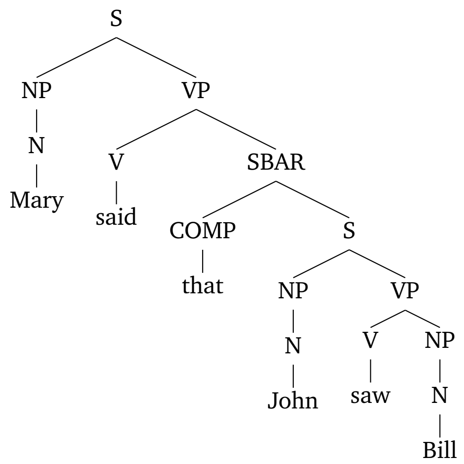
2. **Hidden Markov Models (HMM).** The model represents pos tags in the state sequence and words in the observation sequence. The syntactic tree is ignored. Write the expression of $\P(\mathbf{x})$ for a bigram HMM, where states correspond to single PoS tags, and for a trigram HMM, where states correspond to two adjacent pos tags.

3. **Probabilistic Context-Free Grammars (PCFG).** The model considers the full syntactic tree.

Dependency Parsing

Exercise 15.

Given the sentence *Mary said that John saw Bill*, with the following parse tree:



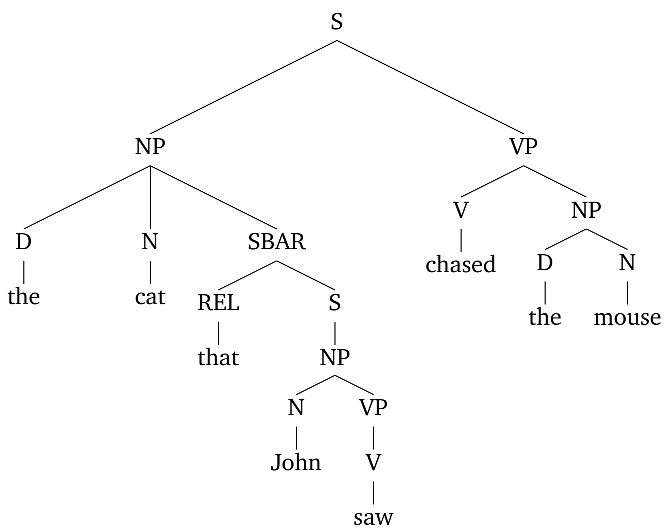
And the following grammar rules (where the superscript + indicates the head):

$$\begin{aligned}
&S \rightarrow NP\ VP^{+} \\
&NP \rightarrow N \\
&VP \rightarrow V^{+}\ NP \\
&VP \rightarrow V^{+}\ SBAR \\
&SBAR \rightarrow COMP^{+}\ S
\end{aligned}$$

1. List the headwords of the following non-terminals:
   - the SBAR
   - the topmost S
   - the VP "*said that John saw Bill*"
2. Draw the dependency tree resulting from the conversion using the given head rules.

Exercise 16.

Given the sentence *The cat that John saw chased the mouse*, with the following parse tree:



And the following grammar rules (where the superscript + indicates the head):

$$\begin{aligned}
&S \rightarrow NP\ VP^{+} \\
&NP \rightarrow N \\
&NP \rightarrow D\ N^{+} \\
&NP \rightarrow D\ N^{+}\ SBAR \\
&VP \rightarrow V^{+}\ NP \\
&VP \rightarrow V \\
&SBAR \rightarrow REL^{+}\ S
\end{aligned}$$

1. List the headwords of the following non-terminals:

- the SBAR
- the NP "*The cat that John saw*"
- the topmost S
- the VP "*chased the mouse*"

2. Draw the dependency tree resulting from the conversion using the given head rules.

Exercise 17.

Consider the sentence: *John quit his job*.
Draw the following dependency parses.

a) (2,1), (0,2), (1,3), (3,4)

b) (2,1), (0,2), (2,3), (3,4)

c) (2,1), (0,2), (2,4), (3,4)

d) (2,1), (0,2), (2,4), (4,3)

e) (0,1), (1,2), (2,3), (3,4)

- Which are invalid parses and why?
- Which are projective parses?

Exercise 18.

In a global linear model for dependency parsing, the feacture vector $f(x, y)$ for any sentence x paired with a dependency tree y is defined as:

$$f(x,y) = \sum_{(h,m)\in y} \mathbf{f}(x,h,m)$$

where $\mathbf{f}(x,h,m)$ is a function that maps a dependency (h,m) and a sentence x to a local feature vector.

We want the vector $f(x,y)$ to have exactly two dimensions, each dimension having the following value:

$f_1(x,y)$ = num of times a dependency with head *car* and modifier *the* is seen in (x,y)
$f_2(x,y)$ = num of times a dependency with head part-of-speech NN, modifier part-of-speech DT, and no adjective (JJ) between the DT and the NN is seen in (x,y)

Assuming that each element in the sentence x_i is a pair $(word, PoS)$, and that the functions $word(x_i)$ and $pos(x_i)$ return the value for each component of the pair:

1. Give a definition of the function $\mathbf{f}(x,h,m) = \langle \mathbf{f}_1(x,h,m), \mathbf{f}_2(x,h,m) \rangle$ that leads to the above definition of $f(x,y)$.

2. Compute the value of $f(x,y)$ for the following pair (x,y):

   $x =$ *The/DT car/NN with/IN the/DT red/JJ hood/NN won/VBD the/DT car/NN race/NN*
   $y = \{(2,1),(7,2),(2,3),(3,6),(6,4),(6,5),(0,7),(7,10),(10,8),(10,9)\}$

Exercise 19.

Recall the factored linear models for labeled dependency parsing. An arc-factored model computes:

$$\begin{aligned} tree(x_{1:n}) &= \operatorname*{argmax}_{y \in \mathcal{Y}(\mathbf{x})} \mathbf{w} \cdot \mathbf{f}(x, y) \\ &= \operatorname*{argmax}_{y \in \mathcal{Y}(\mathbf{x})} \sum_{\langle h,m,l \rangle \in y} \mathbf{w} \cdot \mathbf{f}(x, h, m, l) \end{aligned} \tag{1}$$

In the function, $x_{1:n}$ is an input sentence of n tokens (x_i is the i-th token). $\mathcal{Y}(x)$ is the set of all possible dependency trees for x (each $y \in \mathcal{Y}(x)$ is a dependency tree). The tuple $\langle h, m, l \rangle$ is a labeled dependency: h is is the index of the head word (we have $0 \leq h \leq n$, and $h = 0$ indicates the root token); m is the index of the modifier word (we have $1 \leq m \leq n$), and l is the syntactic label of that dependency (assume $\mathcal{L}$ is the set of possible syntactic relations (e.g. subject, object, modifier, etc.), and that $l \in \mathcal{L}$).

In what follows, assume $pos(x_i)$ and $word(x_i)$ for $i \in \{1 \ldots n\}$ return respectively the part-of-speech and word form in position i in the sentence.

As usual we will define features using feature templates that capture certain syntactic properties. For example, an important property is to consider the compatibility of head-modifier relations with respect to part-of-speech tags. As a particular example, a verb will typically have nouns and adverbs as possible modifiers, but will never have determiners (since these modify nouns). The following feature template will capture this information:

$$\mathbf{f}_{1,a,b}(x_{1:n}, h, m, l) = \begin{cases} 1 & \text{if } pos(x_h) = a \text{ and } pos(x_m) = b \\ 0 & \text{otherwise} \end{cases}$$

In the template above a and b are possible PoS tags. Note that this template ignores the label. We could have another template that looks at PoS compatibility in conjunction with a label $c \in \mathcal{L}$:

$$\mathbf{f}_{2,a,b,c}(x_{1:n}, h, m, l) = \begin{cases} 1 & \text{if } pos(x_h) = a \text{ and } pos(x_m) = b \text{ and } l = c \\ 0 & \text{otherwise} \end{cases}$$

1. Write feature templates that capture the following properties:

   (a) Lexical compatibility. For example, "boy" and "dog" are possible subject modifiers for the verb "eat", but "stone" or "pizza" are not likely subjects; on the other hand, "pizza" is a likely modifier for an object relation with "eat". Write two templates, one ignoring and the other considering the syntactic label:

   - $\mathbf{f}_{3,a,b}(x_{1:n}, h, m, l)$: The head word is a and the modifier is b
   - $\mathbf{f}_{4,a,b,c}(x_{1:n}, h, m, l)$: The head word is a, the modifier is b, and the relation is c.

   (b) Adjectives in English appear before nouns ("small dog"), while for Spanish and Catalan they appear after nouns ("gos petit"). Write templates that capture the relative position of the modifier with respect to the head. Specifically, the features need to capture whether the modifier is to the left or to the right of the head, and whether the two words are adjacent or not. Write templates that only captures the relative position, and others that capture the relative position together with the pos tags or the words.

   - $\mathbf{f}_{5}(x_{1:n}, h, m, l)$: The modifier is to the left of the head word.
   - $\mathbf{f}_{6}(x_{1:n}, h, m, l)$: The modifier is to the right of the head word.
   - $\mathbf{f}_{7}(x_{1:n}, h, m, l)$: The modifier is immediately left of the head word.
   - $\mathbf{f}_{8}(x_{1:n}, h, m, l)$: The modifier is imediately right of the head word.
   - $\mathbf{f}_{9,a,b}(x_{1:n}, h, m, l)$: The head word is a, the modifier is b, and the modifier is to the left of the head word.
   - $\mathbf{f}_{10,a,b}(x_{1:n}, h, m, l)$: The head word is a, the modifier is b, and the modifier is to the right of the head word.

- $\mathbf{f}_{11,a,b}(x_{1:n}, h, m, l)$: The head word is a, the modifier is b, and the modifier is immediately left of the head word.
- $\mathbf{f}_{12,a,b}(x_{1:n}, h, m, l)$: The head word is a, the modifier is b, and the modifier is immediately right of the head word.
- $\mathbf{f}_{13,a,b}(x_{1:n}, h, m, l)$: The head word PoS is a, the modifier PoS is b, and the modifier is to the left of the head word.
- $\mathbf{f}_{14,a,b}(x_{1:n}, h, m, l)$: The head word PoS is a, the modifier PoS is b, and the modifier is to the right of the head word.
- $\mathbf{f}_{15,a,b}(x_{1:n}, h, m, l)$: The head word PoS is a, the modifier PoS is b, and the modifier is immediately left of the head word.
- $\mathbf{f}_{16,a,b}(x_{1:n}, h, m, l)$: The head word PoS is a, the modifier PoS is b, and the modifier is immediately right of the head word.

(c) In a noun phrase such as "many small hungry dogs" we expect to find a sequence of determiners and adjectives before a noun, and don't expect to find verbs in the middle of this sequence. Write feature templates that capture the pos tags of words that appear between the head and the modifier.

- $\mathbf{f}_{17,a}(x_{1:n}, h, m, l)$: The PoS tag a appears between the modifier and the head word.

(d) Write feature templates that capture Subject-Verb-Object phenomena[1] and variations (SOV, SVO, OVS, ...). Try to be general: assume a part of speech of a head word (e.g. verb) and two syntactic relations (e.g. subject and object), and write templates that can capture the relative position of the relations with respect to the head word. Illustrate the type of features that your templates can and can not capture.

- $\mathbf{f}_{18}(x_{1:n}, h, m, l)$: The head is a verb, the modifier is to its left, and it is the subject.
- $\mathbf{f}_{19}(x_{1:n}, h, m, l)$: The head is a verb, the modifier is to its right, and it is the subject.
- $\mathbf{f}_{20}(x_{1:n}, h, m, l)$: The head is a verb, the modifier is to its left, and it is the object.
- $\mathbf{f}_{21}(x_{1:n}, h, m, l)$: The head is a verb, the modifier is to its right, and it is the object.

2. Using the previous templates, compute the value of $f(x, y)$ for the following pair (x, y):

$x =$ *the/DT big/JJ cat/NN eats/VBZ fresh/JJ fish/NN*
$y = \{(3, 1, \mathrm{det}), (3, 2, \mathrm{nmod}), (4, 3, \mathrm{subj}), (0, 4, \mathrm{root}), (6, 5, \mathrm{nmod}), (4, 6, \mathrm{obj})\}$

Exercise 20.

Given the sentence *natural language technology courses are fun*,

1. Draw unlabeled dependency trees for the following interpretations

   (a) technology courses about natural language are fun

   (b) courses about technology on natural language are fun

   (c) natural courses about language technology are fun

   (d) courses about natural technology for language are fun

2. Emulate the behaviour of a transition dependency parser using an arc-standard model (i.e. with operations *shift*, *left-arc*, and *right-arc* between the two topmost stack elements). List the intermediate stack/buffer contents and the selected action at each step to obtain the tree for each of the interpretations above

[1]See `http://en.wikipedia.org/wiki/Subject-verb-object`