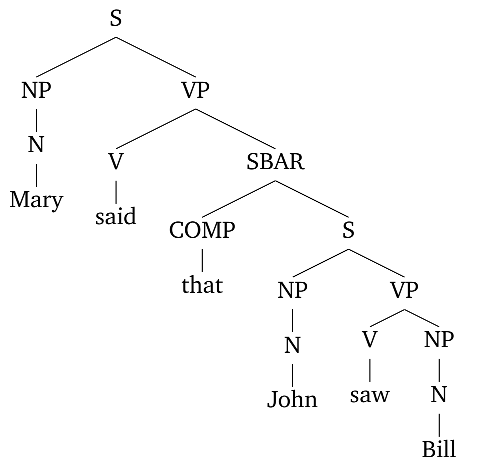
Advanced Human Language Technologies
Exercises on Parsing

Dependency Parsing

Exercise 1.

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



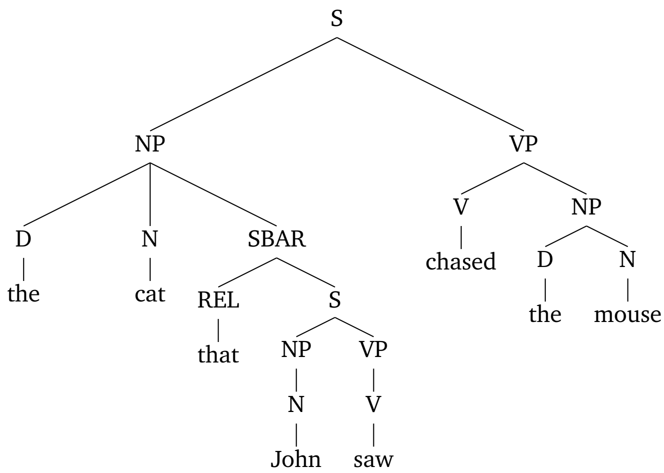
And the following grammar rules (where the superscript + indicates the head):

$S \rightarrow NP\ VP^{+}$
$NP \rightarrow N$
$VP \rightarrow V^{+}\ NP$
$VP \rightarrow V^{+}\ SBAR$
$SBAR \rightarrow COMP^{+}\ S$

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 2.

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):

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

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 3.

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 4.

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:

$$
\begin{aligned}
f_1(x, y) &= \text{num of times a dependency with head } \textit{car} \text{ and modifier } \textit{the} \text{ is seen in } (x, y) \\
f_2(x, y) &= \text{num of times a dependency with head part-of-speech NN, modifier part-of-speech DT,} \\
&\quad \text{and no adjective (JJ) between the DT and the NN is seen in } (x, y)
\end{aligned}
$$

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 5.

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

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

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*
$y = \{(3, 1, \text{det}), (3, 2, \text{nmod}), (4, 3, \text{subj}), (0, 4, \text{root}), (6, 5, \text{nmod}), (4, 6, \text{obj})\}$

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

Exercise 6.

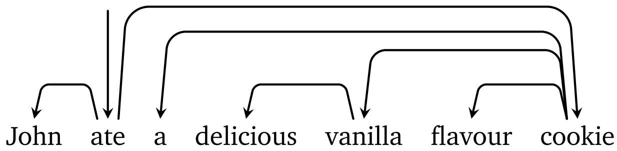
Given the sentence *natural language technology courses are fun*,

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

Exercise 7.

Given the sentence *John ate a delicious vanilla flavour cookie*,

1. Draw unlabeled dependency trees for the following interpretations
   (a) John ate a cookie with flavour of delicious vanilla
   (b) John ate a delicious cookie with vanilla flavour
   (c) John ate a delicious and flavoured cookie made of vanilla
   (d) John ate a cookie with a delicious flavour of vanilla
2. Given the tree



   (a) Explain the interpretation encoded by this tree avoiding any ambiguities.
   (b) Emulate the behaviour that would result in this tree for 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 required action at each step to obtain the final tree.

Exercise 8.

Given the sentence *I had oysters with champagne from France*.

1. Draw unlabeled dependency trees for the following interpretations:
   (a) I had oysters which had champagne on them. The champagne was from France.
   (b) I had oysters which had champagne on them. The oysters were from France.
   (c) I had oysters while having also champagne. The champagne was from France.
   (d) I had oysters while having also champagne. The oysters were from France.
2. Is any of the obtained trees non-projective? Justify your answer.

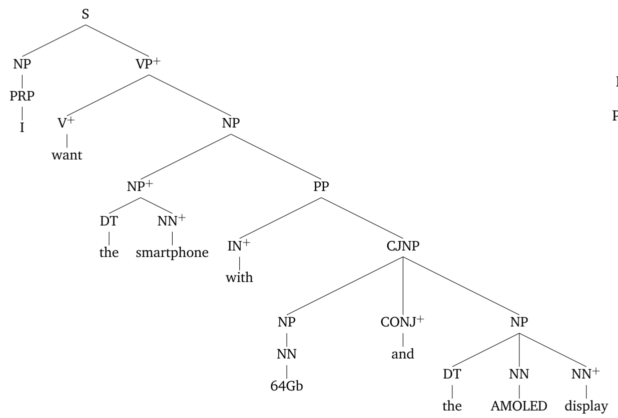
Exercise 9.

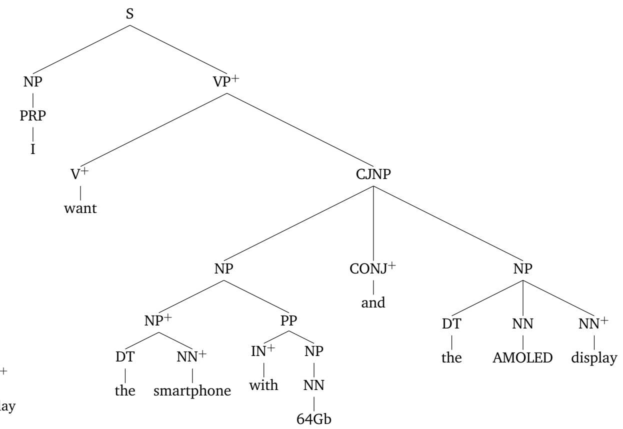
A Papazom.com user wrote the sentence:

I want the smartphone with 64Gb and the AMOLED display

We used the following PCFG grammar (where the $^+$ superscript indicates the head of each rule), and obtained the two possible parse trees below.

r_1 S $\rightarrow$ NP VP$^+$	1.0	r_7 NP $\rightarrow$ PRP	0.1	r_{12} PRP $\rightarrow$ *I*	1.0
r_2 VP $\rightarrow$ V$^+$ NP	0.7	r_8 NP $\rightarrow$ NN	0.2	r_{13} V $\rightarrow$ *want*	1.0
r_3 VP $\rightarrow$ V$^+$ CJNP	0.3	r_9 NP $\rightarrow$ NP$^+$ PP	0.2	r_{14} CONJ $\rightarrow$ *and*	1.0
r_4 CJNP $\rightarrow$ NP CONJ$^+$ NP	1.0	r_{10} NP $\rightarrow$ DT NN$^+$	0.3	r_{15} DT $\rightarrow$ *the*	1.0
r_5 PP $\rightarrow$ IN$^+$ NP	0.6	r_{11} NP $\rightarrow$ DT NN NN$^+$	0.2	r_{16} IN $\rightarrow$ *with*	1.0
r_6 PP $\rightarrow$ IN$^+$ CJNP	0.4			r_{17} NN $\rightarrow$ *display*	0.3
				r_{18} NN $\rightarrow$ *smartphone*	0.4
				r_{19} NN $\rightarrow$ *64Gb*	0.2
				r_{20} NN $\rightarrow$ *AMOLED*	0.1





1. Which parse tree has higher probability according to the PCFG? Reason your answer.
2. Convert both parse trees to dependency trees.
3. Describe in an unambiguous form what is the meaning of the interpretation represented by each tree.

Exercise 10.

One user in out platform wrote the review:

It was a huge bomb explosion movie

1. Draw dependency trees for the following interpretations:
   - (a) The movie was huge and it was about explosions. The movie was also about bombs.
   - (b) The movie was huge and contained the explosion of a bomb.
   - (c) A huge bomb exploded in the movie.
   - (d) The movie was about explosions, and also about a huge bomb.
2. Given the following emulation of the behaviour of a transition dependency parser with an arc-standard model (i.e. with operations *shift*, *left-arc*, and *right-arc* between the two topmost stack elements), answer the questions below:

Stack	Buffer	Edges	Transition
*	It was a huge bomb explosion movie	{}	shift
* It	was a huge bomb explosion movie	{}	shift
* It was	a huge bomb explosion movie	{}	left-arc
* **was**	a huge bomb explosion movie	{**(2,1)**}	shift
* was a	huge bomb explosion movie	{(2,1)}	shift
* was a huge	bomb explosion movie	{(2,1)}	shift
* was a huge bomb	explosion movie	{(2,1)}	left-arc
* was a **bomb**	explosion movie	{(2,1), **(5,4)**}	shift
* was a bomb explosion	movie	{(2,1), (5,4)}	???

(a) If we apply *left-arc* as the next transition

- Which arc would be added to the tree?
- How many of the four interpretations above are still possible, and which ones? Justify your answer.

(b) If we apply *shift* as the next transition instead

- How many of the four interpretations above are still possible, and which ones? Justify your answer.
- Which should be the following transitions to complete the tree and which arcs would each transition add?