NLP Assignment - 3

Ujjwal Sharma (11392010)

02,December 2016

1 Context Free Grammar

(a) For the given grammar, conversion to Chomsky Normal Form yields:

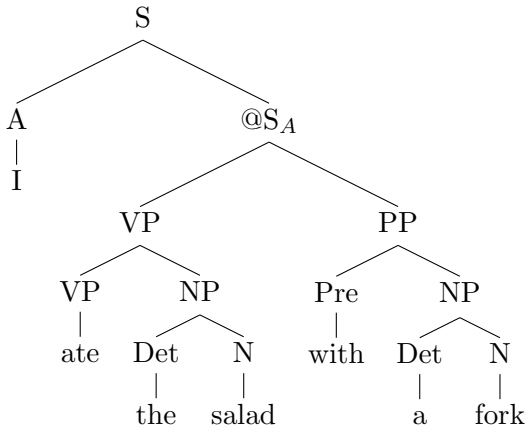
- S → NP VP
- S → A @SA
- A → I
- @SA → VP PP
- NP → Det N
- VP → V NP
- PP → Pre P
- Det → the | a
- N → fork | salad
- Pre → with
- VP → ate
- V → ate

(b) The CKY Parse for the given sentence is as below:

I	ate	the	salad	with	a	fork
A	VP,V	Det	N	Pre	Det	N
Φ	Φ	NP	Φ	Φ	NP	
Φ	VP	Φ	Φ	Φ		
Φ	Φ	Φ	Φ			
Φ	Φ	Φ				
Φ	Φ					
Φ						

Φ – Denotes NULL or no element in that box.

(c) Using the most probable sentence structure from above CKY parse (although it has no explicit properties, we observe only a single parse is possible)



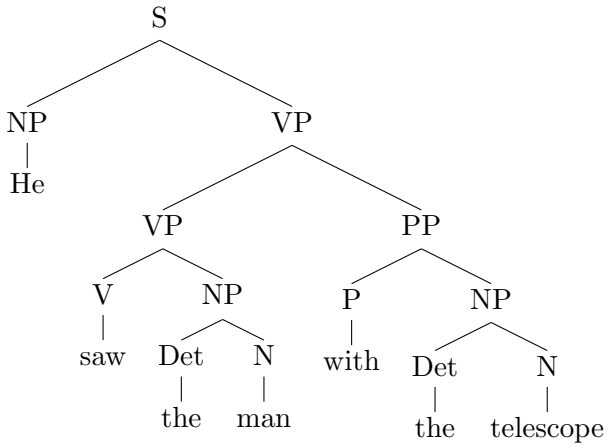
2 Probabilistic Context Free Grammar

(a) For the given structures, the corpus is as below:

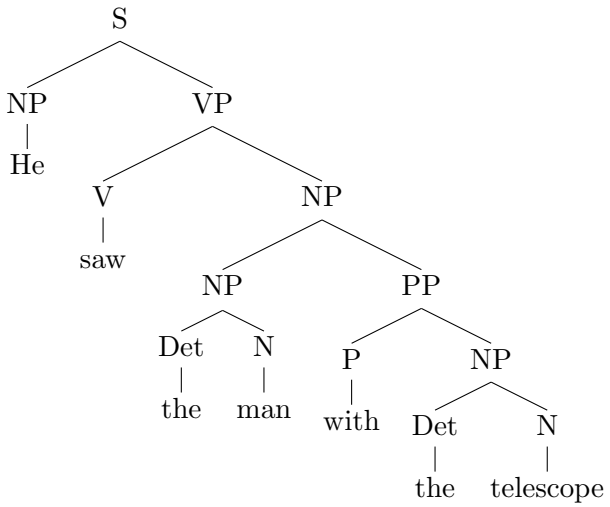
- S → NP VP ($\frac{4}{4}$)
- VP → VP PP ($\frac{1}{5}$)
- VP → V NP ($\frac{4}{5}$)
- PP → P NP ($\frac{3}{3}$)
- NP → Det N($\frac{9}{14}$)
- P → with ($\frac{2}{3}$) | from ($\frac{1}{3}$)
- N → man ($\frac{3}{9}$) | distance ($\frac{1}{9}$) | guitar ($\frac{1}{9}$) | telescope ($\frac{1}{9}$) | girl ($\frac{2}{9}$) | flowers ($\frac{1}{9}$)
- V → saw ($\frac{3}{4}$) | is ($\frac{1}{-}$)
- det → the ($\frac{7}{9}$) | a ($\frac{2}{9}$)
- NP → NP PP ($\frac{2}{14}$) | she ($\frac{1}{14}$) | Here ($\frac{1}{14}$) | He ($\frac{1}{14}$)

The two different Trees are generated by selecting a different rule for splitting up VP at level one wherein the first example splits **VP → VP PP**, the second one splits according to the rule **VP → V NP**

(b)

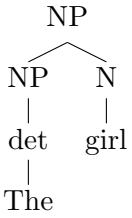


The Tree has the following transitions : (S → NP VP), (VP → VP PP), (VP → V NP), (PP → P NP),(NP → Det N), (P → with) , (V → saw), (Det → the) , (N → man), (P → with), (Det → the), (N → telescope) = 3.7792e-5



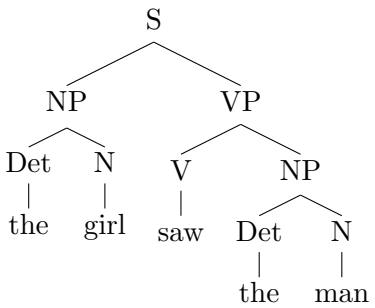
The Tree has the following transitions :(S→NP VP), (VP→VP PP),(VP→V NP),(NP→ Det N),(PP→P NP),(NP→Det N) = 5.29100e-5

(c) The girl saw has the tag structure : The (det) Girl (N) saw (V) which forms (det + N) and V which is a VP.



This follows that according to grammar rules it has the most probability of combining with a PP type sentence which is further a P and a NP so the next phrase could be a P and probabilistically, the most likely P is 'with' hence the next word can be 'with' followed by an NP form according to the derived grammar.

Hence the tree can be further expanded by taking saw as V which makes it more probable for the other part to be of the form VP → V NP and hence NP and hence a maximal likely NP can be formed by using the (det with high probability) and man (N with high probability) to form the sentence "The girl saw the man" with an extremely high probability of occurrence.

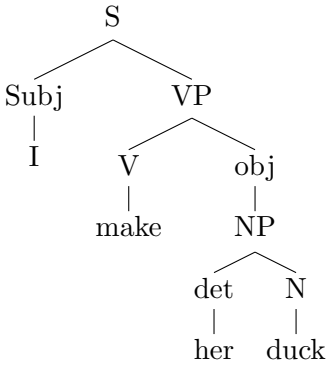


3 CKY Algorithm

(0) I	(1) make	(2) her	(3) duck (4)
Subj -> I (0.3)	-	S -> Subj VP (0.018)	S -> Subj VP (0.018)
	V -> make (0.6)	VP -> V Obj (0.06)	VP* -> V Obj. (0.06) VP -> V Small (0.012)
		Obj -> her (0.2) Det -> her (1.0)	NP -> Det N (0.25) Subj -> NP (0.175) Obj -> NP (0.2) Small -> Obj V (0.1)
			N -> duck (0.5) V -> duck (0.5) NP -> N (0.25) Subj -> NP (0.175) Obj -> NP (0.2)

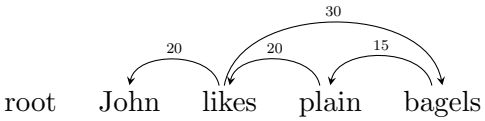
An asterisk (*) denotes the dominant value of that parameter (the greater value) that was used in subsequent calculations.

The back-trace pointers yields:



4 MST Parsing

The initial step will involved a **Greedy Parsing** routine which yields.



This however causes a Cycle which is not permissible in a MST (Tree structure does not permit cyclic chains). Hence we contract cycle into a single loop and recalculate incoming and outgoing weights.

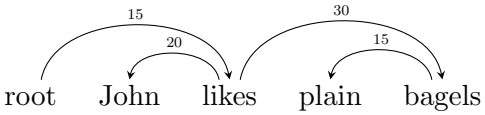
Incoming Weights

- root (15) → likes (30) → bagels(15) → plain = 60
- root (15) → likes (30) → plain(15) → bagels = 25
- root (15) → plain (0) → likes(15) → bagels = 50
- root (15) → plain (0) → bagels(15) → likes = 15
- root (15) → plain (0) → plain(15) → plain = 15
- root (15) → plain (0) → likes(15) → likes = 35

Hence the best path is :
root (15) → likes (30) → bagels(15) → plain = 60

Outgoing Weights
The weight likes → John is the highest scoring and is consequently retained.

Hence the graph is given by,:



5 Dependency Parsing

Transition	Stack	Buffer	A
	[ROOT]	[A koala eats leafs and barks]	ϕ
SHIFT	[ROOT A]	[koala eats leafs and barks]	-
SHIFT	[ROOT A koala]	[eats leafs and barks]	-
LEFT-ARC(det)	[ROOT koala]	[eats leafs and barks]	A∪ det(koala, A)
SHIFT	[ROOT koala eats]	[leafs and barks]	-
LEFT-ARC(nsubj)	[ROOT eats]	[leafs and barks]	A∪ nsubj(eats, koala)
SHIFT	[ROOT eats leafs]	[and barks]	-
RIGHT-ARC(dobj)	[ROOT eats]	[and barks]	A∪ dobj(eats, leafs)
SHIFT	[ROOT eats and]	[barks]	-
RIGHT-ARC(cc)	[ROOT eats]	[barks]	A∪ cc(eats, and)
SHIFT	[ROOT eats barks]	[NULL]	-
RIGHT-ARC(conj)	[ROOT eats]	[NULL]	A∪ conj(eats, barks)
RIGHT-ARC(root)	[ROOT]	[NULL]	A∪ root(ROOT, eats)

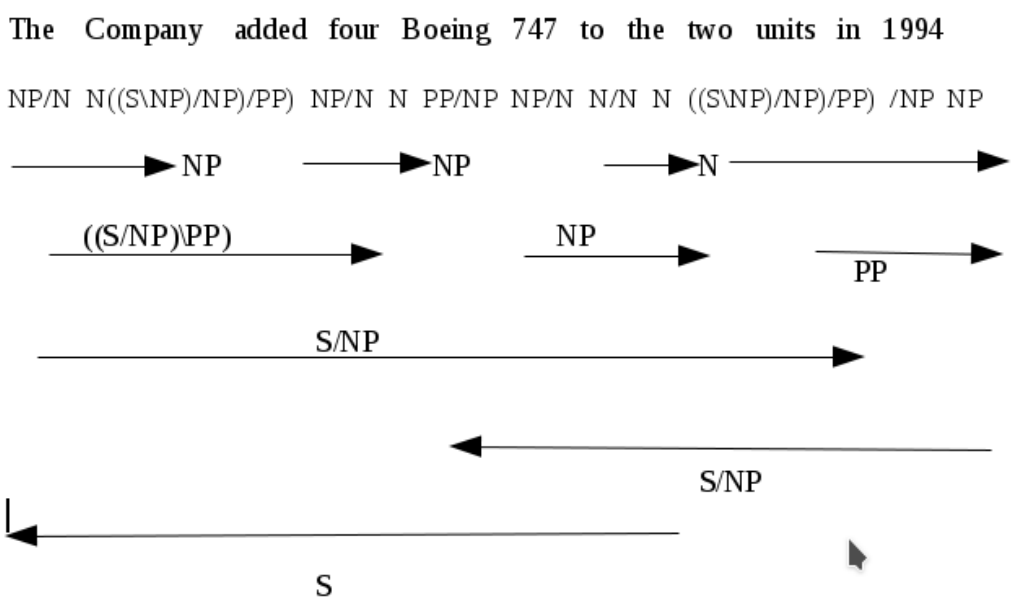
In the above Arc-Standard parsing scheme, we see a wrong association created between (eats,and) and (eats,barks) which is primarily casued by the stack pushing out the dependent word which leaves the head word without a suitable context and causes it to further link up with an unrelated word due to the process of contracting the chain once a word leaves. This link is sometimes not relevant to the case at hand and the contraction of the chain caused eats,and to get related.

NULL denotes a buffer empty status.

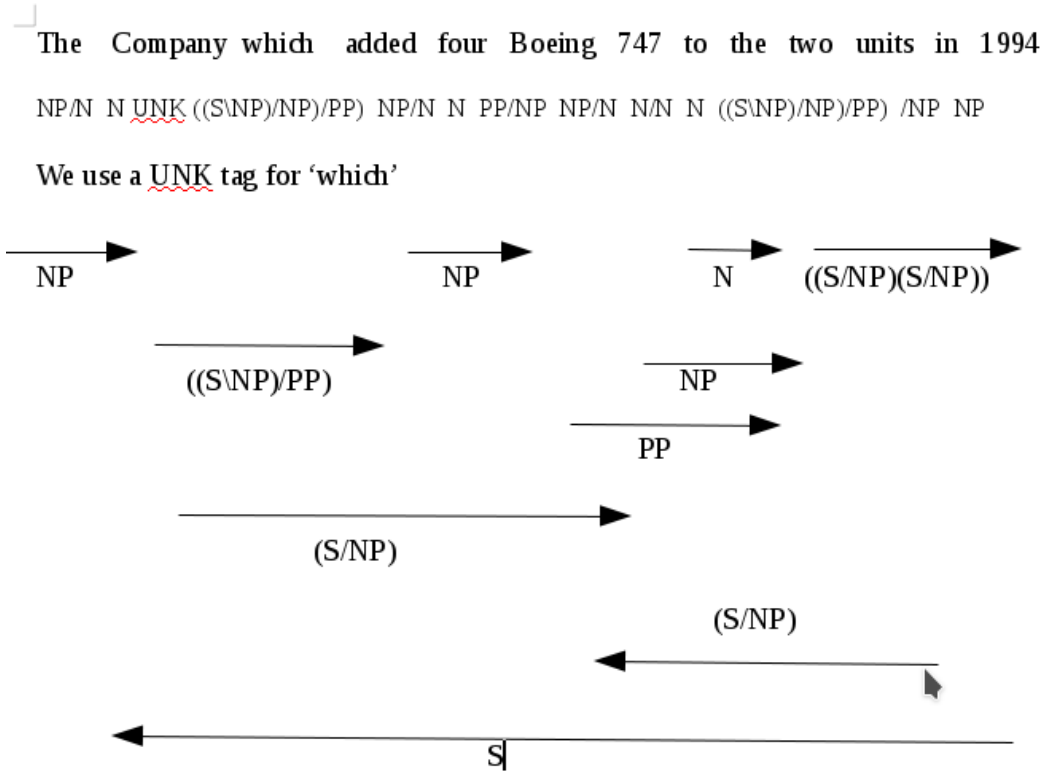
Transition	Stack	Buffer	A
	[ROOT]	[A koala eats leafs and barks]	ϕ
SHIFT	[ROOT A]	[koala eats leafs and barks]	-
LEFT-ARC(det)	[ROOT]	[koala eats leafs and barks]	A∪det (koala, A)
SHIFT	[ROOT koala]	[eats leafs and barks]	-
LEFT-ARC(nsubj)	[ROOT]	[eats leafs and barks]	A∪nsubj (koala,eats)
RIGHT-ARC(root)	[ROOT eats]	[leafs and barks]	A∪root (ROOT,eats)
RIGHT-ARC(dobj)	[ROOT eats leafs]	[and barks]	A∪dobj (eats leafs)
RIGHT-ARC(cc)	[ROOT eats leafs and]	[barks]	A∪cc (leafs,and)
RIGHT-ARC(conj)	[ROOT eats leafs and barks]	[NULL]	A∪ (leafs,barks)
REDUCE	[ROOT eats leafs and barks]	[NULL]	-
REDUCE	[ROOT eats leafs]	[NULL]	-
REDUCE	[ROOT eats]	[NULL]	-
REDUCE	[ROOT]	[NULL]	-

6 CCG Parsing

(a) We apply multiple forward composition rules to obtain the following:



(b) In this question 'which' is taken as an example of Unknown (UNK) type.



(c) We hebben Hans geholpen het huis te schilderen is a sentence with crossing dependency projections and literally translated to English would imply that We have Hans helped to paint the house.

This sentence has a crossing dependency projector and hence the literal English meaning is grammatically incorrect.