Beverlyn Tucker IST664 Context-Free Grammars-POS HM3

October 28, 2020

```
Professor: Stephen Wallace
IST_664 NLP HM2: Regular Expressions
```

[28]:
```
import nltk

# read the sentences from the file sentences.txt
sentfile = open('sentences.txt', 'r')
# make a list of sentences, separating the tokens by white space.
sentence_list = []
for line in sentfile:
    sentence_list.append(line.split())
```

[29]:
```
# read the grammar file - the nltk data function load will not reload
#    the file unless you set the cache to be False
camg = nltk.data.load('file:camelot_grammar1.cfg', cache=False)

# create a recursive descent parser
cam_parser = nltk.RecursiveDescentParser(camg)

# for each sentence print it and its parse trees
# if the grammar cannot parse a sentence, sometimes it gives an error and
#    sometimes it just goes on to the next sentence with no parse tree
for sent in sentence_list:
    print(sent)
    for tree in cam_parser.parse(sent):
        print (tree)
```

```
['Arthur', 'is', 'the', 'king', '.']
(START
  (S1
    (NP (Proper Arthur))
    (VP (VerbT is) (NP (Det the) (NP (Noun king))))
    (Eos .)))
['Arthur', 'rides', 'the', 'horse', 'near', 'the', 'castle', '.']
(START
  (S1
    (NP (Proper Arthur))
    (VP
      (VerbT rides)
      (NP
        (Det the)
        (NP
          (Noun horse)
          (PP (Prep near) (NP (Det the) (NP (Noun castle)))))))
    (Eos .)))
(START
  (S1
    (NP (Proper Arthur))
    (VP
      (VerbT rides)
      (NP (Det the) (NP (Noun horse)))
      (PP (Prep near) (NP (Det the) (NP (Noun castle)))))
    (Eos .)))
['Arthur', 'rides', 'the', 'plodding', 'horse', 'near', 'the', 'castle', '.']
(START
  (S1
    (NP (Proper Arthur))
    (VP
      (VerbT rides)
      (NP
        (Det the)
        (ADJ plodding)
        (NP
          (Noun horse)
          (PP (Prep near) (NP (Det the) (NP (Noun castle)))))))
    (Eos .)))
(START
  (S1
    (NP (Proper Arthur))
    (VP
      (VerbT rides)
      (NP (Det the) (ADJ plodding) (NP (Noun horse)))
      (PP (Prep near) (NP (Det the) (NP (Noun castle)))))
    (Eos .)))
['the', 'Holy_Grail', 'is', 'a', 'chalice', '.']
(START
  (S1
    (NP (Det the) (NP (PNP Holy_Grail)))
    (VP (VerbT is) (NP (Det a) (NP (Noun chalice))))
    (Eos .)))
['the', 'sensational', 'Holy_Grail', 'is', 'a', 'sacred', 'chalice', '.']
(START
  (S1
    (NP (Det the) (ADJ sensational) (NP (PNP Holy_Grail)))
    (VP (VerbT is) (NP (Det a) (ADJ sacred) (NP (Noun chalice))))
    (Eos .)))
['every', 'coconut', 'was', 'carried', 'to', 'the', 'hottest', 'mountains', '.']
```

```
(START
  (S1
    (NP (Det every) (NP (Noun coconut)))
    (VP
      (VBD was)
      (VBD carried)
      (PP (TO to) (NP (Det the) (JJS hottest) (NP (NNS mountains)))))
    (Eos .)))
['sixty', 'strangers', 'are', 'at', 'the', 'Round_Table', '.']
(START
  (S1
    (NP (CD sixty) (NP (NNS strangers)))
    (VP
      (VBP are)
      (PP (Prep at) (NP (Det the) (NP (PNP Round_Table)))))
    (Eos .)))
['Sir_Lancelot', 'might', 'have', 'spoken', '.']
(START
  (S1
    (NP (Proper Sir_Lancelot))
    (VP (MD might) (VB have) (VBN spoken))
    (Eos .)))
['Guinevere', 'had', 'been', 'riding', 'with', 'Patsy', 'for', 'five', 'weary',
'nights', '.']
(START
  (S1
    (NP (Proper Guinevere))
    (VP
      (VBD had)
      (VBN been)
      (VBG riding)
      (PP
        (Prep with)
        (NP
          (Proper Patsy)
          (PP (Prep for) (NP (CD five)))
          (ADJ weary)
          (NP (NNS nights)))))
    (Eos .)))
['Sir_Bedevere', 'might', 'have', 'been', 'suggesting', 'this', 'quest', '.']
(START
  (S1
    (NP (Proper Sir_Bedevere))
    (VP
      (MD might)
      (VB have)
      (VBN been)
      (VBG suggesting)
      (NP (Det this) (NP (Noun quest))))
    (Eos .)))
['the', 'Britons', 'migrate', 'south', 'frequently', '.']
(START
  (S1
    (NP (Det the) (NP (NNPS Britons)))
    (VP (VB migrate) (RB south) (RB frequently))
    (Eos .)))
['Arthur', 'and', 'Guinevere', 'ride', 'frequently', 'near', 'the', 'castle',
'.']
(START
  (S1
    (NP (Proper Arthur) (CC and) (Proper Guinevere))
    (VP
      (VB ride)
      (RB frequently)
      (PP (Prep near) (NP (Det the) (NP (Noun castle)))))
    (Eos .)))
['he', 'suggests', 'to', 'grow', 'fruit', 'at', 'home', '.']
(START
  (S1
    (NP (PRP he))
    (VP
      (VBZ suggests)
      (TO to)
      (VB grow)
      (NP (Noun fruit) (PP (Prep at) (NP (Noun home)))))
    (Eos .)))
['riding', 'to', 'Camelot', 'is', 'not', 'hard', '.']
(START
  (S1
    (VP (VBG riding) (PP (TO to) (NP (PNP Camelot))))
    (VP (VerbT is) (NOT not) (ADJ hard))
    (Eos .)))
['do', 'coconuts', 'speak', '?']
(START (S1 (NP (DO do) (NP (NNS coconuts))) (VP (VB speak)) (Eos ?)))
['why', 'does', 'England', 'have', 'a', 'king', '?']
(START
  (S1
    (WAdv why)
    (NP (DO does) (NP (PNP England)))
    (VP (VB have) (NP (Det a) (NP (Noun king))))
    (Eos ?)))
['Coconut', 'is', 'the', 'king.']


       ␣
↪-----------------------------------------------------------------------------
```

```
    ValueError                                Traceback (most recent call␣
↪last)

    <ipython-input-29-d5a1c1d8c748> in <module>
     11 for sent in sentence_list:
     12     print(sent)
---> 13     for tree in cam_parser.parse(sent):
     14         print (tree)


    ~\anaconda3\lib\site-packages\nltk\parse\recursivedescent.py in␣
↪parse(self, tokens)
     76
     77         tokens = list(tokens)
---> 78         self._grammar.check_coverage(tokens)
     79
     80         # Start a recursive descent parse, with an initial tree


    ~\anaconda3\lib\site-packages\nltk\grammar.py in check_coverage(self,␣
↪tokens)
    682             missing = ', '.join('%r' % (w,) for w in missing)
    683             raise ValueError(
--> 684                 "Grammar does not cover some of the " "input words:␣
↪%r." % missing
    685             )
    686


    ValueError: Grammar does not cover some of the input words: "'Coconut',␣
↪'king.'".
```

1 [‘Arthur’, ‘is’, ‘the’, ‘king’, ‘.’]

(START (S1 (NP (Proper Arthur)) (VP (VerbT is) (NP (Det the) (NP (Noun king)))) (Eos .)))

The sentence rules are NP VP Eos; the parser will parse Arthur (the noun phrase). Arthur is a proper noun and tagged as Proper. “is the king” is a VP that contains an NP. “is” is a transitive verb with is tagged VerbT. “the king” is a noun phrase that nestled in the verb phrase; therefore, we added VerbT NP to the rules for VP. The last part of the sentence that needed to parse is “the king” the is a determiner (Det), and king is a noun (NN). Therefore, Det NN added to the rules for NP. However, we know that it is likely that Det will have other noun phrases that follow it, and we decided to add two new rules Det NP and NN for NP. When the parser runs, it will find Det NP and then go back to match the correct NP that follows Det.

2 [‘Arthur’, ‘rides’, ‘the’, ‘horse’, ‘near’, ‘the’, ‘castle’, ‘.’]

(START (S1 (NP (Proper Arthur)) (VP (VerbT rides) (NP (Det the) (NP (Noun horse) (PP (Prep near) (NP (Det the) (NP (Noun castle))))))) (Eos .)))

This part only needs to add NN PP to parse “horse near.” The rest of the sentence parsed with the rules previously implemented.

3 [‘Arthur’, ‘rides’, ‘the’, ‘plodding’, ‘horse’, ‘near’, ‘the’, ‘castle’, ‘.’]

(START (S1 (NP (Proper Arthur)) (VP (VerbT rides) (NP (Det the) (ADJ plodding) (NP (Noun horse) (PP (Prep near) (NP (Det the) (NP (Noun castle))))))) (Eos .)))

The only difference in this sentence from sentence 2 is the adjective “plodding.” added the NP rule JJ NN PP to catch “plodding horse near” because we already have the rule Det NP. Det has not required a new rule. We created the JJ tag for adjectives.

4 [‘the’, ‘Holy_Grail’, ‘is’, ‘a’, ‘chalice’, ‘.’]

[‘the’, ‘Holy_Grail’, ‘is’, ‘a’, ‘chalice’, ‘.’] (START (S1 (NP (Det the) (NP (PNP Holy_Grail))) (VP (VerbT is) (NP (Det a) (NP (Noun chalice)))) (Eos .)))

The sentence has a very similar structure to the previous sentences. The only rule that we need is a rule for “Holy_Grail” a proper noun that does not refer to a person. We named this tag NNP. By adding NNP as a NP rule, the parser will catch “the Holy_Grail” as a NP, by first catching Det NP and then going back and looking for a NP that matches “Holy_Grail”.

5 [‘the’, ‘sensational’, ‘Holy_Grail’, ‘is’, ‘a’, ‘sacred’, ‘chalice’, ‘.’]

(START (S1 (NP (Det the) (ADJ sensational) (NP (PNP Holy_Grail))) (VP (VerbT is) (NP (Det a) (ADJ sacred) (NP (Noun chalice)))) (Eos .)))

This sentence additional adjective “sensational,” Therefore, it needs to include a rule that had ADJ PNP.

6 [‘every’, ‘coconut’, ‘was’, ‘carried’, ‘to’, ‘the’, ‘hottest’, ‘mountains’, ‘.’]

(START (S1 (NP (Det every) (NP (Noun coconut))) (VP (VBD was) (VBD carried) (PP (TO to) (NP (Det the) (JJS hottest) (NP (NNS mountains))))) (Eos .)))

The noun phrase “every coconut” will be parsed with the existing rules. However, we need to add new rules for the verb phrase “was carried to the hottest mountains.” “was” is a past tense verb which we labeled as VBD. “carried” we determined was a past participle, which we labeled as VBN. We created the following VP: VBD VBN NP. The noun phrase is “to the hottest mountains.” “to” is tagged as TO, “hottest” is a superlative adjective labeled as JJS and mountains is a plural noun tagged as NNS. Therefore we needed to add two more rules to NP: TO NP and JJS NNS.

7 [‘sixty’, ‘strangers’, ‘are’, ‘at’, ‘the’, ‘Round_Table’, ‘.’]

(START (S1 (NP (CD sixty) (NP (NNS strangers))) (VP (VBP are) (PP (Prep at) (NP (Det the) (NP (PNP Round_Table))))) (Eos .)))

This sentence starts with an NP that begins with a number, which we tagged as a CD. We added two tags for the NP “sixty strangers.” The first tag was CC NP, and the second was NNS. In case there is additional NP that started with CC, CC NP should address this scenario as long as we have an NP rule to match what follows CC. Added NNS because we did not have a rule solely for NNS. The VP “are at the Round_Table” also required a new rule. “are” is a present, plural, third-person verb tagged as VBP, followed by a prepositional phrase “at the Round_Table”. The rest of the sentence already conforms to the existing rules.

8 [‘Sir_Lancelot’, ‘might’, ‘have’, ‘spoken’, ‘.’]

(START (S1 (NP (Proper Sir_Lancelot)) (VP (MD might) (VB have) (VBN spoken)) (Eos .)))

It does not have to add a rule for the NP; however, add a new rule for the VP. “might” is a modal verb that we tagged as MD. We decided to create two rules for this: MD VP, and then we need VBN to parse “have spoken” this way, we might not have to generate as many rules.

9 [‘Guinevere’, ‘had’, ‘been’, ‘riding’, ‘with’, ‘Patsy’, ‘for’, ‘five’, ‘weary’, ‘nights’, ‘.’]

(START (S1 (NP (Proper Guinevere)) (VP (VBD had) (VBN been) (VBG riding) (PP (Prep with) (NP (Proper Patsy) (PP (Prep for) (NP (CD five))) (ADJ weary) (NP (NNS nights))))) (Eos .)))

No new rules required for the initial on NP, a new rule added for the VP. “had been riding with Patsy for five weary nights”. In order to parse “had been riding with,” the following rule is required VBD VBN VBG PP. “with Patsy” will be parsed with the current rules; however, a new rule is needed to be added for a PP to follow a Proper noun. Therefore, Proper PP was included in the rules. The rest of the sentence will be parsed with our current rules.

10 [‘Sir_Bedevere’, ‘might’, ‘have’, ‘been’, ‘suggesting’, ‘this’, ‘quest’, ‘.’]

(START (S1 (NP (Proper Sir_Bedevere)) (VP (MD might) (VB have) (VBN been) (VBG suggesting) (NP (Det this) (NP (Noun quest)))) (Eos .)))

The initial NP “Sir_Bedevere” does not require any additional rules. The new rule that was needed is Noun VBN VBG NP this ensures that “ might have been suggesting this quest” is parsed.

11 [‘the’, ‘Britons’, ‘migrate’, ‘south’, ‘frequently’, ‘.’]

(START (S1 (NP (Det the) (NP (NNPS Britons))) (VP (VB migrate) (RB south) (RB frequently)) (Eos .)))

A new rule was created to capture the first NP "the Britons". "Britons" is a plural proper noun that was given the tag NNPS. The VP also needed a rule generated. "migrate south frequently" is comprised of VB and two adverbs. We gave adverbs the RB tag.

12 ['Arthur', 'and', 'Guinevere', 'ride', 'frequently', 'near', 'the', 'castle', '.']

(START (S1 (NP (Proper Arthur) (CC and) (Proper Guinevere)) (VP (VB ride) (RB frequently) (PP (Prep near) (NP (Det the) (NP (Noun castle))))) (Eos .)))

To parse "Arthur and Guinevere" the first NP, a new rule was generated: Proper NP. A new rule was also necessary to parse the VP " ride frequently near the castle". "ride frequently near" was parsed with VB RB PP. No new rules were needed for "the castle".

13 ['he', 'suggests', 'to', 'grow', 'fruit', 'at', 'home', '.']

(START (S1 (NP (PRP he)) (VP (VBZ suggests) (TO to) (VB grow) (NP (Noun fruit) (PP (Prep at) (NP (Noun home))))) (Eos .)))

A new rule was generated for the initial NP "he." "he" is a personal pronoun was we tagged it as PRP. A new rule was also necessary to parse the VP "suggests to grow fruit at home." "suggests" is a third-person singular verb, and we gave it the tag VBZ. "grow" is also in its base form, and we gave it the tag VB. No new rules were needed to parse the PP "at home."

14 ['riding', 'to', 'Camelot', 'is', 'not', 'hard', '.']

(START (S1 (VP (VBG riding) (PP (TO to) (NP (PNP Camelot)))) (VP (VerbT is) (NOT not) (ADJ hard)) (Eos .)))

A new rule was generated for the NP "riding to Camelot". "riding" is considered a gerund or present participle it given the tag VBG. "to Camelot" is already parsed with our existing rules; therefore, the new rule is VBG NP. The VP also needed a new rule to parse "is not hard" is made of a VerbT NOT and ADJ

15 ['do', 'coconuts', 'speak', '?']

(START (S1 (NP (DO do) (NP (NNS coconuts))) (VP (VB speak)) (Eos ?)))

A new rule was generated for the initial NP "do coconuts". "do" is tagged with DO; this has not been accounted for in the rules yet; therefore, we added the following rule DO NP. A new rule also generated for the VP "speak" no rule previously existed for a singular VB with nothing else following it.

16 ['why', 'does', 'England', 'have', 'a', 'king', '?']

(START (S1 (WAdv why) (NP (DO does) (NP (PNP England))) (VP (VB have) (NP (Det a) (NP (Noun king)))) (Eos ?)))

A new rule was needed for the first NP “why does England”. “why” is a wh-adverb and given the WRB tag. There are already rules written that will parse “does England” therefore the new rule only included WAdv NP. A new rule was also needed for the VP “have a king”. There were no rules for a NP to follow Noun.

17 [‘what’, ‘horse’, ‘does’, ‘Arthur’, ‘ride’, ‘?’]

(START (S1 (NP (WDet what) (NP (NN horse) (NP (DO does) (NP (Proper Arthur))))) (VP (Noun ride)) (Eos ?)))

A new rule was generated to capture “what” in the initial NP. In this situation “what” is acting as a WDet determiner and was given the tag WDet. A second rule was added for the initial NP for “horse does Arthur” we needed a NN NP to parse this.

18 Added following rules to the camelot_grammar.cfg file:

• S1: VP VP Eos – Sentence can have two verb phrases followed by an end of sentence.

• S1: WAdv NP VP Eos – Sentence can start with “how”, “when”, “where”, “why”, followed by a noun phrase and a verb phrase. Reserved for questions.

• S1: NP VP NP Eos – Sentence can have a NP, followed by a VP, and second NP.

• S1: NP VP CC VP Eos – Sentence starts with a NP, followed by two VP joined by a connector (CC).

• VP: VerbT PP – Verb in the present tense followed by PP

• VP: VerbT NOT ADJ – Verb in the present tense followed by negation and adjective.

• VP: VBG NOT ADJ – Present participle verb followed by negation and

adjective. • VP: VB – Verb in present tense.

• VP: VB RB RB – Verb in base form followed by two adverbs.

• VP: VB RB PP – Verb in base form followed by adverb and PP.

• VP: VBD PP – Verb in past tense followed by PP.

• VP: VBD VBD PP – Two verbs in past tense followed by PP.

• VP: VBD VBN VBG PP – Three verbs in past tense, past participle, and present participle, followed by PP.

• VP: VBP PP – Verbs in third person followed by PP.

• VP: VBG PP – Verbs in present participle followed by PP.

• VP: VBZ TO VB NP – Verb in singular third person followed by ‘to’, verb in base form and NP.

• VP: MD VB VBN – Modal followed by verb in base form and past participle.

• VP: MD VB VBN VBG NP – Modal followed by verb in base form, past participle, and present participle, then followed by a NP.

- VP: VBZ NP – Verb in third person singular form, followed by a NP.
- NP: Det ADJ NP – Determiner followed by an adjective and NP.
- NP: Det JJS NP – Determiner followed by a superlative adjective and NP.
- NP: Proper PP – Proper noun followed by a PP.
- NP: PNP – Proper noun, non-people.
- NP: NNPS – Plural proper nouns.
- NP: CD NP – Numbers followed by NP.
- NP: CD – Numbers.
- NP: NNS – Plural nouns
- NP: Proper PP ADJ NP – Proper noun followed by PP, adjective, and second PP.
- NP: Proper CC Proper - Two proper nouns connected by a CC.
- NP: PRP – Personal pronoun.
- NP: DO NP – "Do"/"Does" followed by NP.
- NP: WPro VP – "What"/"Who" followed by VP.
- NP: POS Noun – Possessive personal pronoun followed by noun.
- PP: TO NP – "To" word followed by NP.

19 Challenge Sentences

20 Sentence two not in Original

['Coconut', 'is', 'the', 'king.']

This sentence did not parse; the only difference with this from the "Arthur is the king" is Coconut is NNP and Arthur is an NP. The rest is the same. Coconut is the king; these words had a different meaning. We could say in leman term this is very popular.

21 Part2 Added the sentence below

['frequently', ',', 'Guinevere', 'goes', 'to', 'the', 'castle', 'to', 'speak', 'with', 'the', 'king', '.']

This sentence is unable to be parsed by our current grammar rules. The start of the sentence is a RB (adverb) followed by P (a comma) and then an NP (noun phrase). We have a rule for VP that starts with an RB (adverb) and is followed by a P (pause), which should parse this part of the sentence, and then according to our sentence, rules automatically go to the NP.

The NP "Guinevere goes ..." would be Proper VP, included in our grammar rules, this part of the sentence would be parsed. "goes to the castle" is the next phrase, needs to be parsed. "goes" is a VBZ followed by "to the castle" TO NP. We need to see if we have a rule for VBZ TO NP or VBZ NP, and then a rule in NP that is TO NP, which we do have would parse this part of the sentence VBZ NP and the in NP we have TO NP. parse successfully "goes to the castle."

Though, as far as our current grammar rules will take us in parsing this sentence. Required a rule that will parse “castle to speak with the king,” also we do not have a rule that will connect “castle” and “to speak.”

We have a rule that will connect “castle” and TO NP, but we need the following rule NN TO VP and then the rule VB PP to parse the sentence correctly. Another rule that could be added to allow the sentence to be parsed is the rule NN VP and then the following rule in VP: TO VB PP.

[ ]:
```
```