NLP Lab 5

Dixitha Kasturi
dkasturi@syr.edu

Topic: Context Free Grammars(CFG)

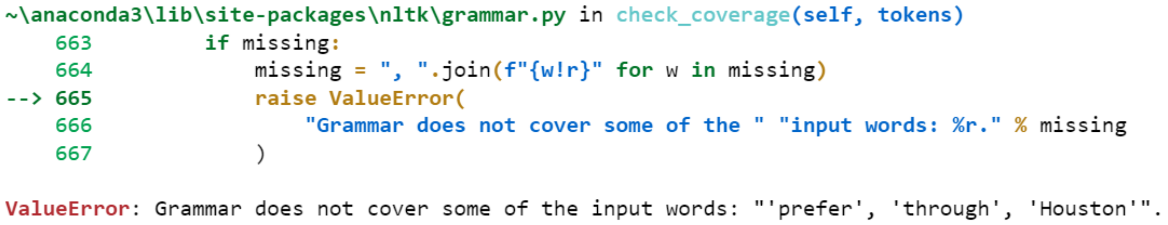
For this assignment I chose 4 sentences. 3 from the given examples, 1 of my own.

```
text1 = "I prefer a flight through Houston".split()
text2 = "John gave the dog a bone".split()
text3 = "I want to book that flight".split()
text4 = "I am a dog lover and an ice-cream connoisseur".split() #my own example
```

When i considered the default flights cfg to parse the first text, it gave me error,

Default cfg :

```
        flight_grammar = nltk.CFG.fromstring("""
 S -> NP VP | VP
 VP -> V NP | V NP PP
 PP -> P NP
 V -> "saw" | "ate" | "walked" | "shot" | "book"
 NP -> Prop | Det N | Det N PP
 Prop -> "John" | "Mary" | "Bob" | "I"
 Det -> "a" | "an" | "the" | "my" | "that"
 N -> "man" | "dog" | "cat" | "telescope" | "park" | "elephant" | "pajamas" | "flight"
 P -> "in" | "on" | "by" | "with"
 """)
```

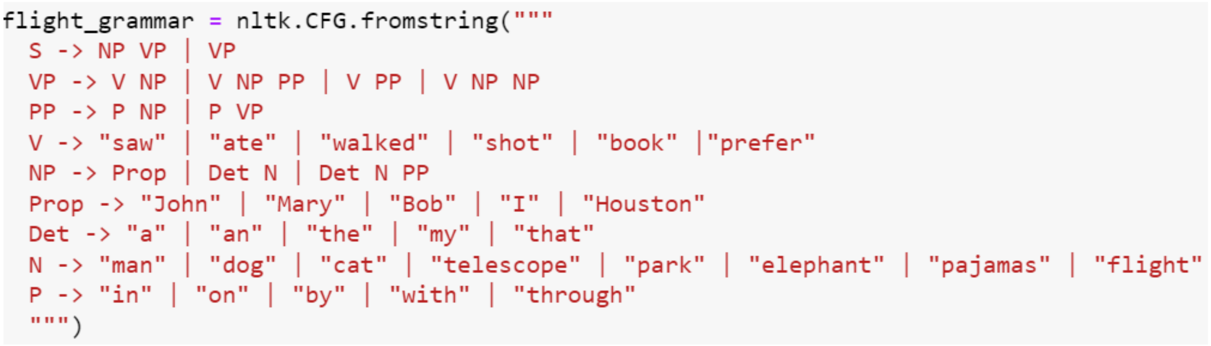


```
~\anaconda3\lib\site-packages\nltk\grammar.py in check_coverage(self, tokens)
    663         if missing:
    664             missing = ", ".join(f"{w!r}" for w in missing)
--> 665             raise ValueError(
    666                 "Grammar does not cover some of the " "input words: %r." % missing
    667             )

ValueError: Grammar does not cover some of the input words: "'prefer', 'through', 'Houston'".
```

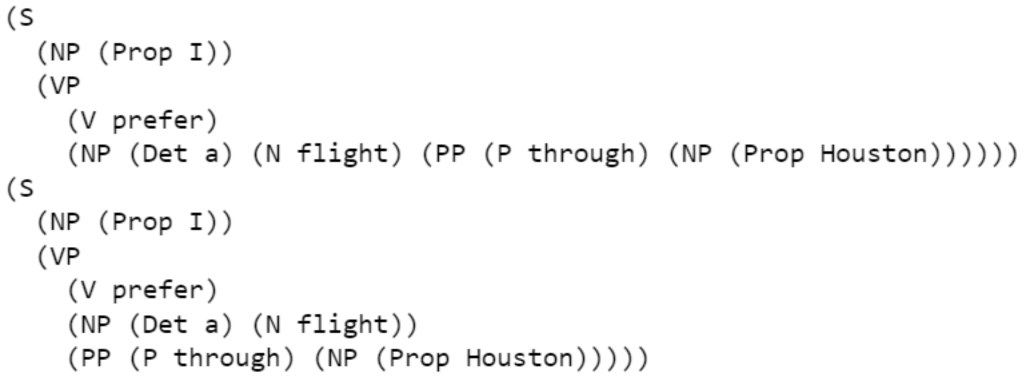
From here on I understood that the CFG has to be modified according to the requirements(here sentences) to accommodate all possible cases of sentences.

1) For sentence 1 the CFG and tree were : *"I prefer a flight through Houston"*



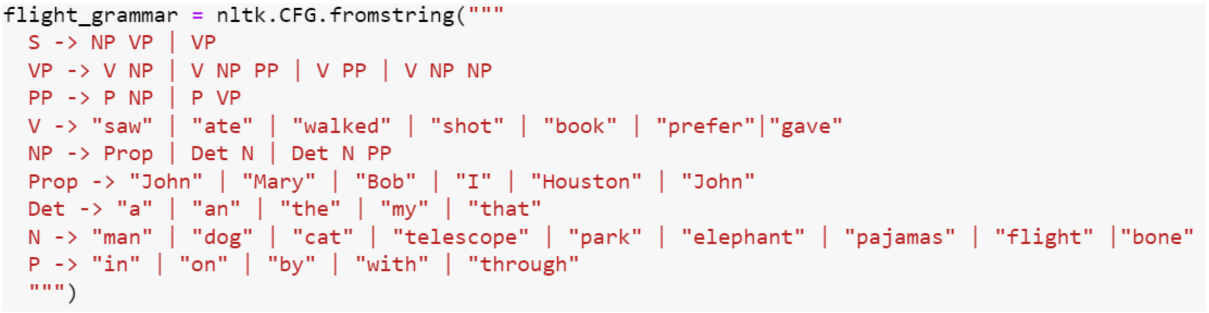
```
flight_grammar = nltk.CFG.fromstring("""
  S -> NP VP | VP
  VP -> V NP | V NP PP | V PP | V NP NP
  PP -> P NP | P VP
  V -> "saw" | "ate" | "walked" | "shot" | "book" |"prefer"
  NP -> Prop | Det N | Det N PP
  Prop -> "John" | "Mary" | "Bob" | "I" | "Houston"
  Det -> "a" | "an" | "the" | "my" | "that"
  N -> "man" | "dog" | "cat" | "telescope" | "park" | "elephant" | "pajamas" | "flight"
  P -> "in" | "on" | "by" | "with" | "through"
  """)
```

tree :



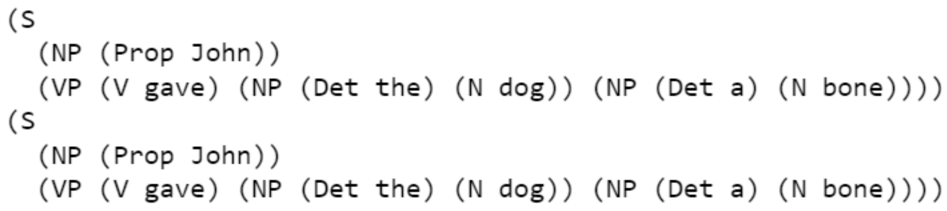
```
(S
  (NP (Prop I))
  (VP
    (V prefer)
    (NP (Det a) (N flight) (PP (P through) (NP (Prop Houston))))))
(S
  (NP (Prop I))
  (VP
    (V prefer)
    (NP (Det a) (N flight))
    (PP (P through) (NP (Prop Houston)))))
```

2) For sentence 2 : *"John gave the dog a bone".*



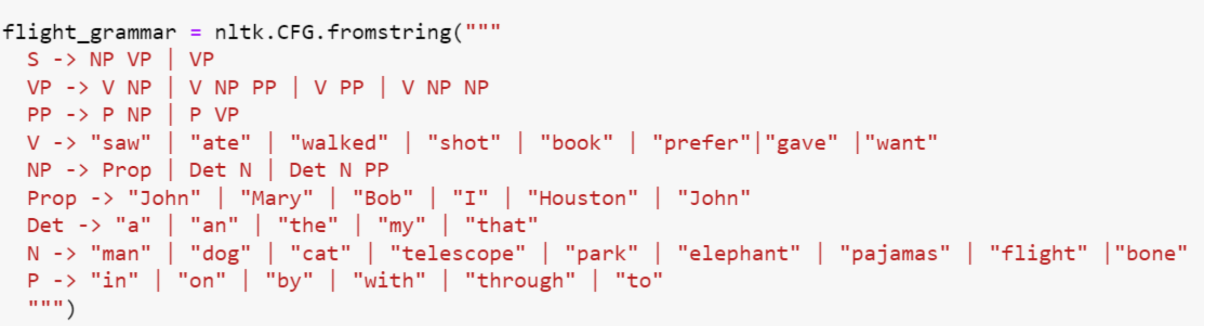
```
flight_grammar = nltk.CFG.fromstring("""
  S -> NP VP | VP
  VP -> V NP | V NP PP | V PP | V NP NP
  PP -> P NP | P VP
  V -> "saw" | "ate" | "walked" | "shot" | "book" | "prefer"|"gave"
  NP -> Prop | Det N | Det N PP
  Prop -> "John" | "Mary" | "Bob" | "I" | "Houston" | "John"
  Det -> "a" | "an" | "the" | "my" | "that"
  N -> "man" | "dog" | "cat" | "telescope" | "park" | "elephant" | "pajamas" | "flight" |"bone"
  P -> "in" | "on" | "by" | "with" | "through"
  """)
```

tree :



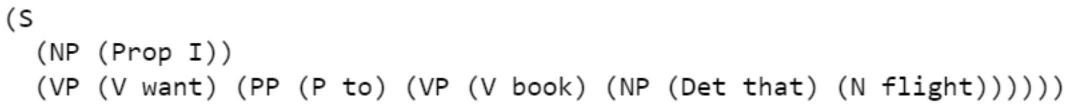
```
(S
  (NP (Prop John))
  (VP (V gave) (NP (Det the) (N dog)) (NP (Det a) (N bone))))
(S
  (NP (Prop John))
  (VP (V gave) (NP (Det the) (N dog)) (NP (Det a) (N bone))))
```

3) For sentence 3 : *"I want to book that flight"*



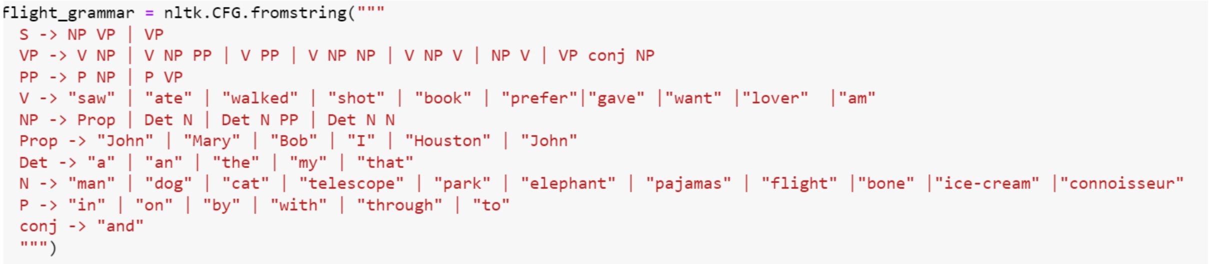
```
flight_grammar = nltk.CFG.fromstring("""
  S -> NP VP | VP
  VP -> V NP | V NP PP | V PP | V NP NP
  PP -> P NP | P VP
  V -> "saw" | "ate" | "walked" | "shot" | "book" | "prefer"|"gave" |"want"
  NP -> Prop | Det N | Det N PP
  Prop -> "John" | "Mary" | "Bob" | "I" | "Houston" | "John"
  Det -> "a" | "an" | "the" | "my" | "that"
  N -> "man" | "dog" | "cat" | "telescope" | "park" | "elephant" | "pajamas" | "flight" |"bone"
  P -> "in" | "on" | "by" | "with" | "through" | "to"
  """)
```

tree :



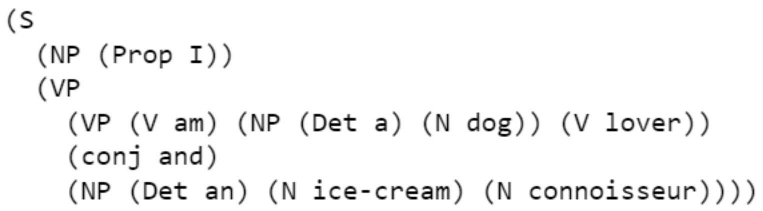
```
(S
  (NP (Prop I))
  (VP (V want) (PP (P to) (VP (V book) (NP (Det that) (N flight))))))
```

4) For sentence 4 : *"I am a dog lover and an ice-cream connoisseur".*
My own sentence, forming the CFG was a little tricky, as the sentence structure was different and I used a conjunction "and". To parse the sentence, I had to include the condition VP conj NP under VP , and Det N N under NP in the cfg. This is because of the sentence structure of the sentence I chose.



```
flight_grammar = nltk.CFG.fromstring("""
  S -> NP VP | VP
  VP -> V NP | V NP PP | V PP | V NP NP | V NP V | NP V | VP conj NP
  PP -> P NP | P VP
  V -> "saw" | "ate" | "walked" | "shot" | "book" | "prefer"|"gave" |"want" |"lover"  |"am"
  NP -> Prop | Det N | Det N PP | Det N N
  Prop -> "John" | "Mary" | "Bob" | "I" | "Houston" | "John"
  Det -> "a" | "an" | "the" | "my" | "that"
  N -> "man" | "dog" | "cat" | "telescope" | "park" | "elephant" | "pajamas" | "flight" |"bone" |"ice-cream" |"connoisseur"
  P -> "in" | "on" | "by" | "with" | "through" | "to"
  conj -> "and"
  """)
```

Tree :



```
(S
  (NP (Prop I))
  (VP
    (VP (V am) (NP (Det a) (N dog)) (V lover))
    (conj and)
    (NP (Det an) (N ice-cream) (N connoisseur))))
```

Inference:

After trial and error I was able to parse the sentences using the CFG. I understood that including all conditions is important. If there was a case discrepancy, converting the sentence into lowercase (if our grammar only has lowercase tokens) is a much better approach, this would avoid getting errors for upper case tokens. Editing/accommodating different scenarios in the CFG according to the requirement is important. Understanding sentence structure through Parts of speech is important in deciding which CFG elements/parsing structure should be included in the CFG(eg det N PP). I would further like to experiment with complex and inclusive cfgs, which are not limited.