LING/C SC 581:
Advanced Computational Linguistics

Lecture 20

Today's Topic

- Homework 9 Review
- We should do some *live* programming

Homework 9: Question 1 Review

- Tregex code:
  - `/^WH.*-([0-9]+)/#1%index > (__ << (/^-NONE-/ <: /^\*T\*-([0-9]+$)/#1%index))`
- Limit *y* in *y* c-commands *w* to the above regex.
- How many matching c-commands relations did you get?
- Show your output and code

ccommand2f.py

```python
# (c) Sandiway Fong, University of Arizona, 2022
from itertools import permutations
from nltk.tree import Tree
import sys

def dom(x):
    yield x
    if not isinstance(x, str):
        for y in x:
            yield from dom(y)

def cc(x):
    if not isinstance(x, str):
        if len(x) > 1:
            for y,z in permutations(x, 2):
                for w in dom(z):
                    print(y, 'c-commands', w)
            for u in x:
                cc(u)
        else:
            cc(x[0])

if len(sys.argv) == 2:
    with open(sys.argv[1]) as f:
        t  = Tree.fromstring(f.read())
        cc(t)
```

Homework 9: Question 1 Review

How many c-commands relations did you get?

```
python ccommand2fq1.py tregex-whtrace-ex1.mrg
```

1. (WHNP-1 (WDT which)) c-commands (S (NP-SBJ (-NONE- *T*-1)) (VP (MD may) (VP (VB end) (NP (PRP$ our) (NN civilization)))))
2. (WHNP-1 (WDT which)) c-commands (NP-SBJ (-NONE- *T*-1))
3. (WHNP-1 (WDT which)) c-commands (-NONE- *T*-1)
4. (WHNP-1 (WDT which)) c-commands *T*-1
5. (WHNP-1 (WDT which)) c-commands (VP (MD may) (VP (VB end) (NP (PRP$ our) (NN civilization))))
6. (WHNP-1 (WDT which)) c-commands (MD may)
7. (WHNP-1 (WDT which)) c-commands may
8. (WHNP-1 (WDT which)) c-commands (VP (VB end) (NP (PRP$ our) (NN civilization)))
9. (WHNP-1 (WDT which)) c-commands (VB end)
10. (WHNP-1 (WDT which)) c-commands end
11. (WHNP-1 (WDT which)) c-commands (NP (PRP$ our) (NN civilization))
12. (WHNP-1 (WDT which)) c-commands (PRP$ our)
13. (WHNP-1 (WDT which)) c-commands our
14. (WHNP-1 (WDT which)) c-commands (NN civilization)
15. (WHNP-1 (WDT which)) c-commands civilization

should glance at the output to see if all y have form ^WH.*[0-9]+

Aside: Counting in Python

- Basics:
  - `total = 0` and somewhere in your code `total += 1`
- However:
  - https://docs.python.org/3/faq/programming.html#what-are-the-rules-for-local-and-global-variables-in-python

In Python, variables that are only referenced inside a function are implicitly global. If a variable is assigned a value anywhere within the function's body, it's assumed to be a local unless explicitly declared as global.

Though a bit surprising at first, a moment's consideration explains this. On one hand, requiring `global` for assigned variables provides a bar against unintended side-effects. On the other hand, if `global` was required for all global references, you'd be using `global` all the time. You'd have to declare as global every reference to a built-in function or to a component of an imported module. This clutter would defeat the usefulness of the `global` declaration for identifying side-effects.

Aside: Counting in Bash (Terminal)

- Command (using Python global variable):

```
python ccommand2fq1b.py tregex-whtrace-ex1.mrg
15
```

- Command (using `wc -l`):

```
python ccommand2fq1.py tregex-whtrace-ex1.mrg | wc -l
17
```

- Command (using `grep -Ec`):

```
python ccommand2fq1.py tregex-whtrace-ex1.mrg | grep
-Ec '^\(WH'
15
```

Homework 9: Question 2 Review

- regex code:
  - `/^WH.*-([0-9]+)/#1%index > (__ << (/^-NONE-/ <: /^\*T\*-([0-9]+$)/#1%index))`
- Add: limit *w* in *y* c-commands *w* to the above regex.
- Example:

```
python ccommand2fq2.py tregex-whtrace-ex1.mrg
(WHNP-1 (WDT which)) c-commands *T*-1
```

Homework 9: Question 3 Review

- Example:

```
python ccommand2fq2.py tregex-whtrace-ex2.mrg
(WHNP-1 (WDT which)) c-commands *T*-1
(WHNP-1 (WDT which)) c-commands *T*-2
(WHNP-2 (-NONE- 0)) c-commands *T*-2
```

- Example:

```
python ccommand2fq3.py tregex-whtrace-ex2.mrg
(WHNP-1 (WDT which)) c-commands *T*-1
(WHNP-2 (-NONE- 0)) c-commands *T*-2
```

Homework 9: Question 3 Review

Impose the constraint that the indices have to be the same

- Example:
  - `(WHNP-1 (WDT which)) c-commands *T*-1`
- Tregex search expression
  - `/^WH.*-([0-9]+)/#1%index > (__ << (/^-NONE-/ <: /^\*T\*-([0-9]+$)/#1%index))`
- `Suppose:`
  - `m1 = re.search(regex1, string1)`
  - `m1.group(1)` gets group 1 string from regex match
  - `m2 = re.search(regex2, string2)`
  - `m2.group(1)` gets group 1 string from regex match
- Then:
  - `m1.group(1) == m2.group(1)` is what we want

Homework 9: Question 4 Review

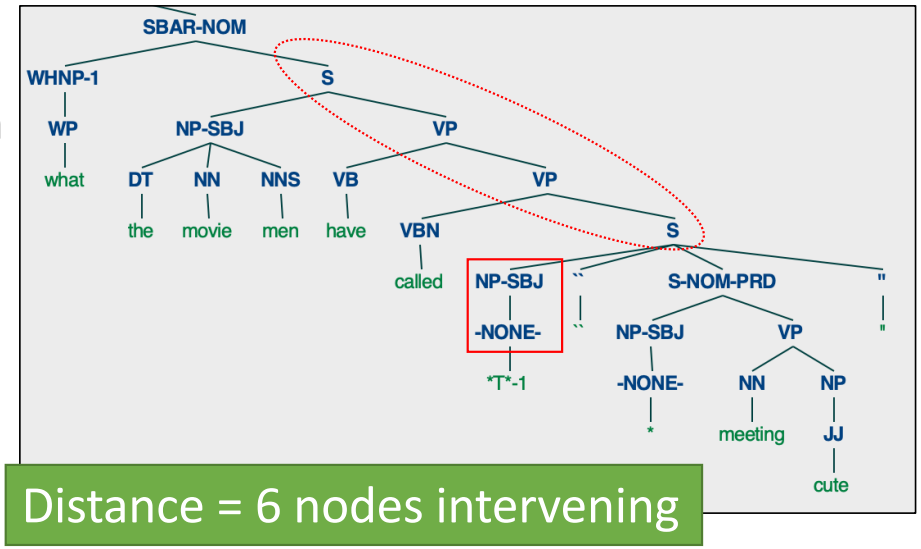
Use the counting idea (*see previous slides*)

```
python -i ccommand2fq4.py
>>> from nltk.corpus import ptb
>>> for t in ptb.parsed_sents():
...     cc(t)
...
>>> total
21603
```

Homework 9: Question 5 Review

Modify your code to compute the distance between the antecedent and its trace.

- Find the tree with the biggest distance in the ptb corpus.
- **Hint:**
  - modify the function `dom` to count depth
  - Note: `dom` is recursive
  - use an extra parameter in the function call to increment the depth each time around



Distance = 6 nodes intervening

Homework 9: Question 5 Review

Answers in increasing order:

- 0 : (WHNP-1 (WP what)) c-commands *T*-1 at 6
- 53 : (WHPP-4 (IN in) (WHNP (WDT which))) c-commands *T*-4 at 7
- 199 : (WHNP-6 (NP (DT the) (NN source)) (WHPP (IN of) (WHNP (WDT which)))) c-commands *T*-6 at 9
- 2738 : (WHPP-1 (IN in) (WHNP (WDT which))) c-commands *T*-1 at 13

- (blue: tree number)

Homework 9: Question 5 Review

- Idea: increment the count with recursion

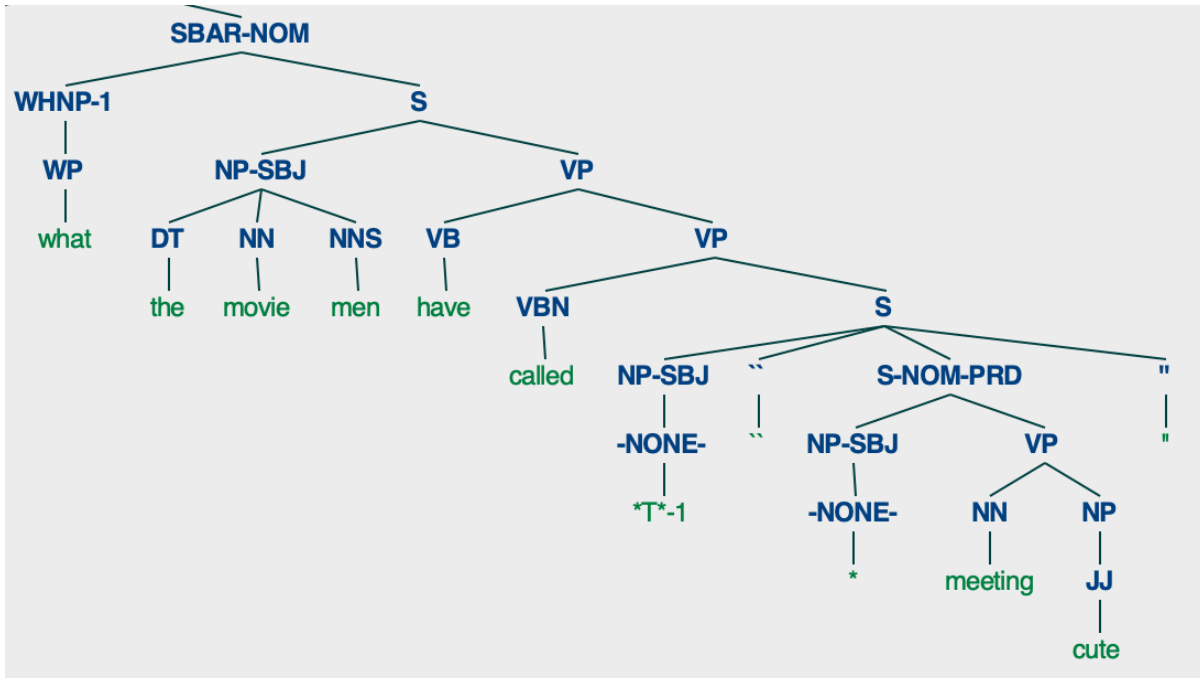
```
def dom(x):
    yield x
    if not isinstance(x, str):
        for y in x:
            yield from dom(y)
```

- Call:
  - `dom(x, 0)`
- Definition:
  - `dom(x, n)`
- Recursive call:
  - `dom(x, n+1)`
- `Yield` (normally `return`):
  - `x, n`
- Call return:
  - `w, i = dom(z, 0)`

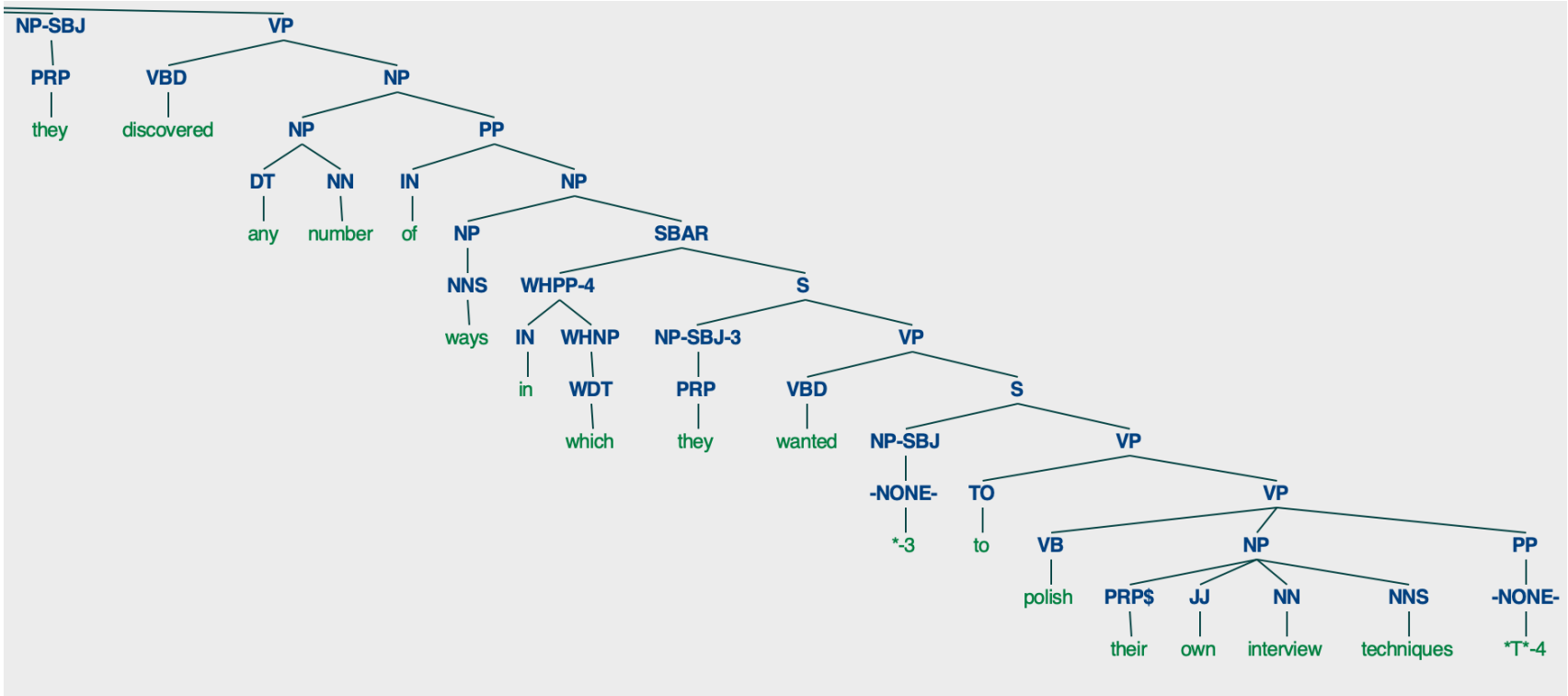
Aside: Max in Python

- Basics:
  - `mx = 0` and somewhere in your code
  - `if depth > mx:`
  - `    mx = depth`

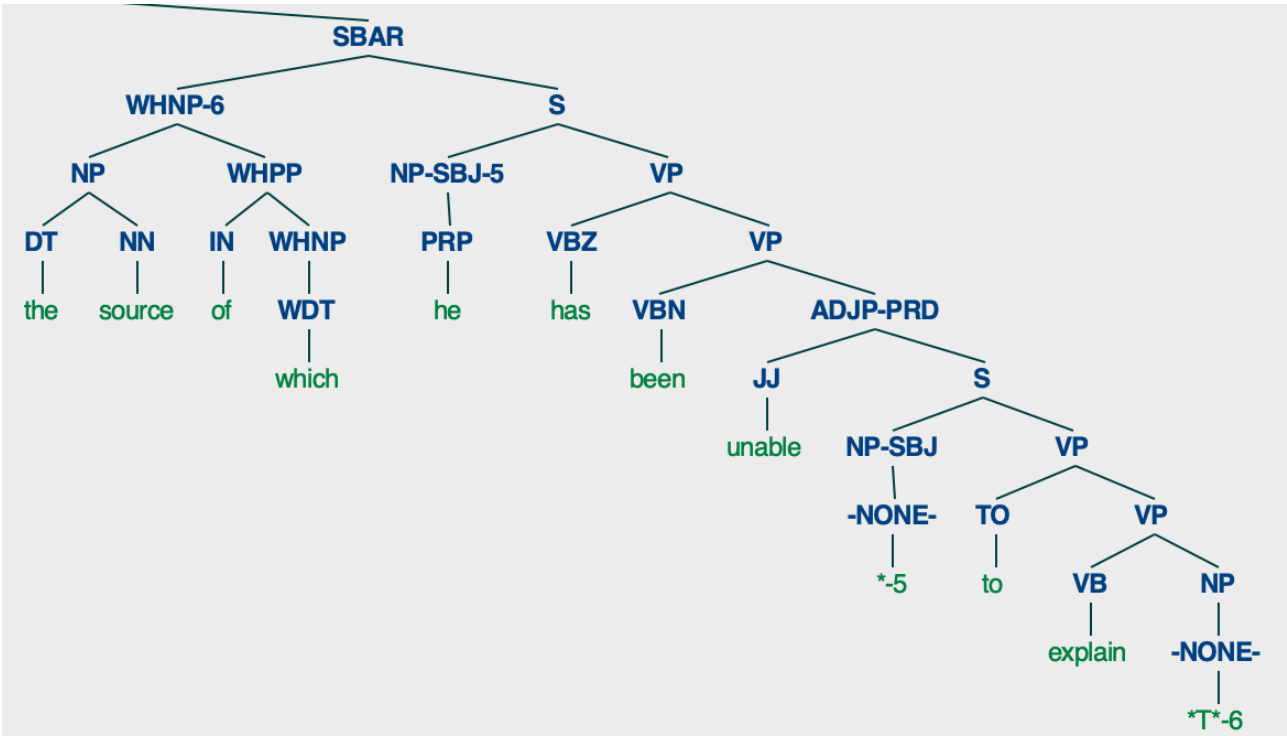
0 : (WHNP-1 (WP what)) c-commands *T*-1 at 6



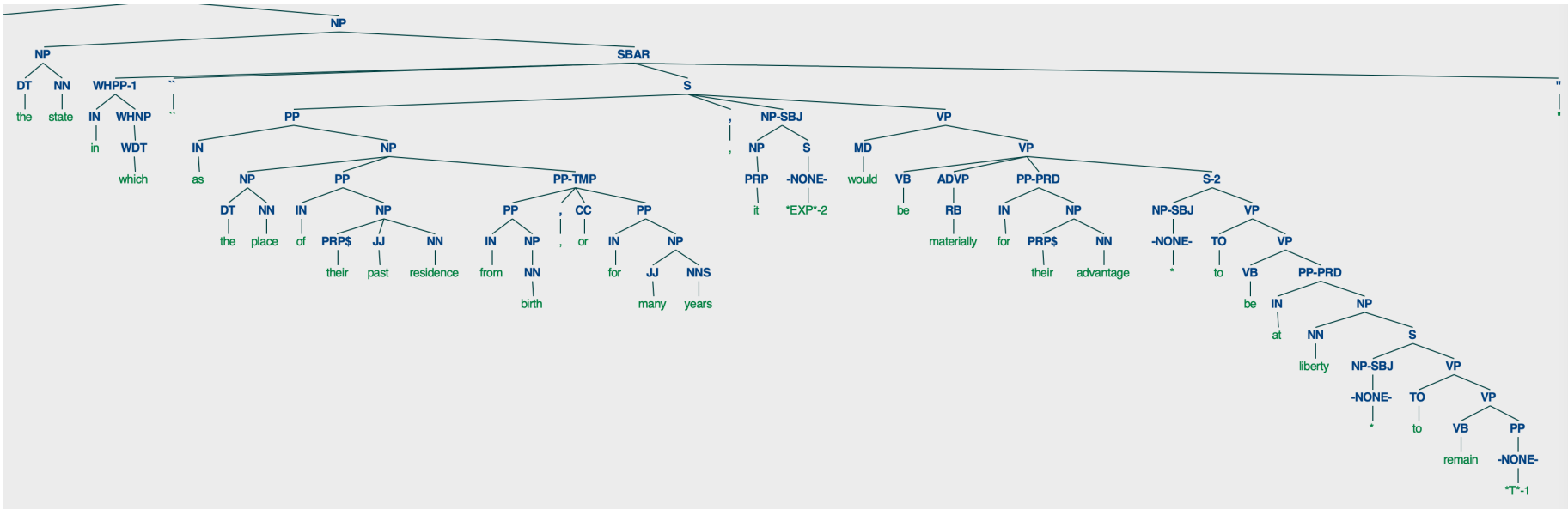
53 : (WHPP-4 (IN in) (WHNP (WDT which))) c-commands *T*-4 at 7



199 : (WHNP-6 (NP (DT the) (NN source)) (WHPP (IN of) (WHNP (WDT which)))) c-commands *T*-6 at 9



2738 : (WHPP-1 (IN in) (WHNP (WDT which))) c-commands *T*-1 at 13



Homework 9: Question 5 Review

- How to get the tree (and its number)?
  - iterate over `ptb.parsed_sents()`
  - use built-in `enumerate()`
  - https://docs.python.org/3/library/functions.html#enumerate

```
>>> seasons = ['Spring', 'Summer', 'Fall', 'Winter']
>>> list(enumerate(seasons))
[(0, 'Spring'), (1, 'Summer'), (2, 'Fall'), (3, 'Winter')]
>>> list(enumerate(seasons, start=1))
[(1, 'Spring'), (2, 'Summer'), (3, 'Fall'), (4, 'Winter')]
```

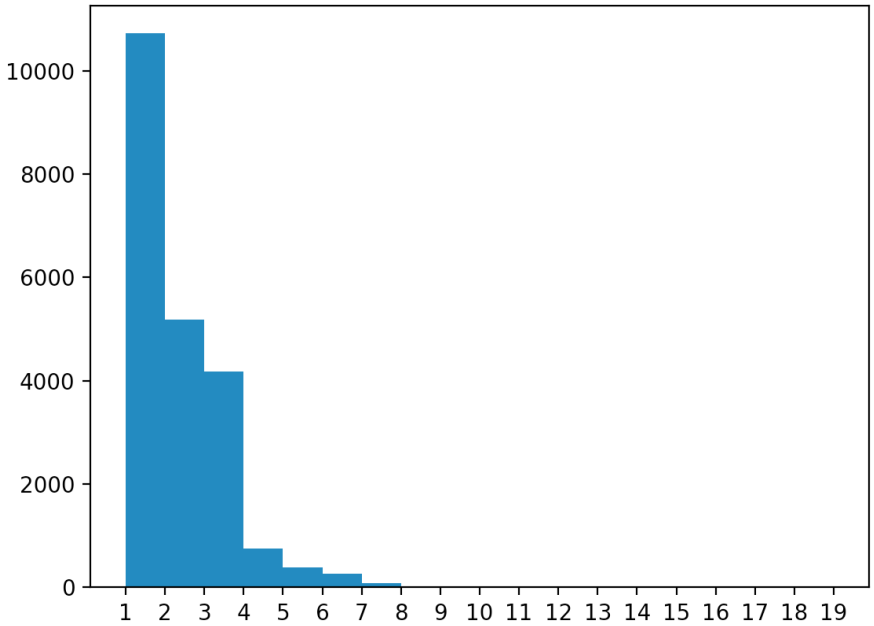
  - pass the tree number down to `cc()`
  - i.e. `cc(x, num)`

Homework 9: Question 6 Review

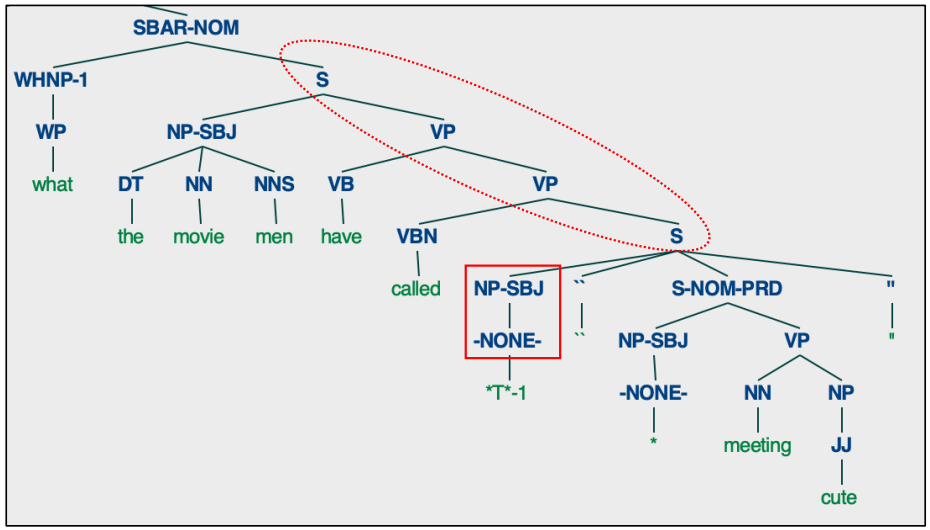
Extra Credit

- Modify your code to plot the histogram for the WH-antecedent to trace depth over the entire ptb corpus
- **Hint**:
  - `import matplotlib.pyplot as plt`
  - `plt.hist(dist, bins=range(1,mx))`
  - `plt.xticks(range(1,mx))`
  - `plt.show()`

Distance between trace and antecedent



- counting internal nodes only
- `count – 2` used to exclude POS tag –NONE– and parent label



Aside: List of values in Python

- Basics:
  - `dist = []` and somewhere in your code `dist.append(value)`
- Then:
  - `plt.hist(dist, bins=range(1,mx))`