Animating Parsing :

Finite State Machine, Parsing Tree, Earley's Parse Animation

Jessica Lei, Mingnan Su

Problem And Motivation

The problem of using Jupyter Notebook is that users have the ability to present algorithms along with the corresponding code segments, but no visualization. Our project aims to add animated graph visualization to the algorithms including finite state machines, derivation of sentences and Earley's Parser .

Earley's Parser Animation

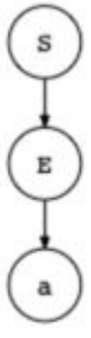
```
g3 = ("S→E", "E→a")
x3 = "a"
a3 = Animate(g3,x3,auto_generate=True)
a3.display()
```

next

Predict
s[0]: E → •a, 0

Match
s[1]: E → (a)•, 0

Complete
s[1]: S → (E(a))•, 0



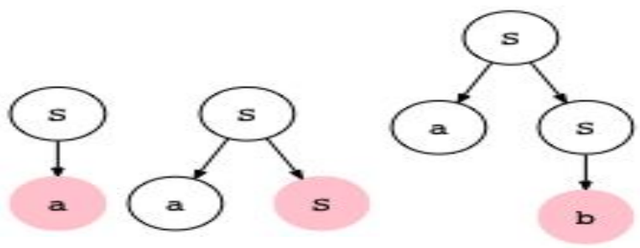
===FINISH===

Statistics

- 660 lines of code
- 15 success test case, 3 error test case, 1 unit test

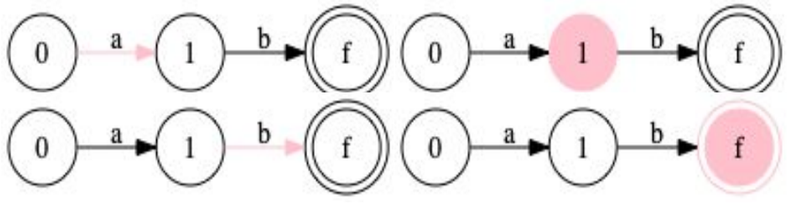
Sentence Derivation Animation

```
grammar0 = nltk.CFG.fromstring("""
S -> 'a' S | 'b'|""")
parser0 = nltk.ChartParser(grammar3)
sentence0 = ['a','b']
t0 = list(parser0.parse(sentence0))[0]
print(t0)
pt0 = PTGraph(t0)
pt0.display()
```



Finite State Machine Animation

```
#user define states
states = ['0', '1', 'f']
#user defined inital state
inital='0'
#user define final states
finals = ['f']
#user define transitions
transitions = [
    { 'trigger': 'a', 'source': '0', 'dest': '1' },
    { 'trigger': 'b', 'source': '1', 'dest': 'f' },]
#Check if the FSM accept string 'abc'
check_str = 'ab'
g = FSMGraph(states,inital, finals, transitions, check_str)
g.display()
```



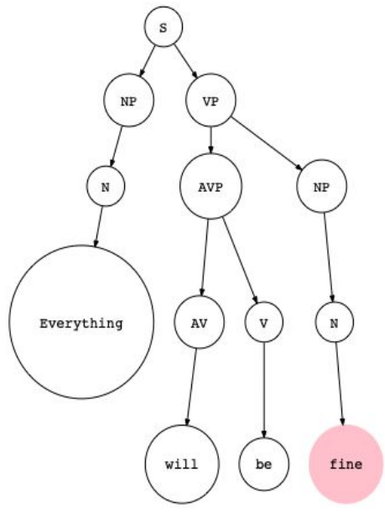
Test And Result

- Unit testing and black-box testing
- Conducted test case for animation by passing in parameters and check the output. The function will output visualization for accepted parameter, and output "Error input" for unaccepted parameter.

```
grammar1 = nltk.CFG.fromstring("""
S -> NP VP
PP -> P NP
NP -> N | Det N
VP -> V NP | VP PP | AVP NP
AVP -> AV V
AV -> 'will'
Det -> 'the'
N -> 'fine' | 'end' | 'Everything'
V -> 'be'
P -> 'in'
""")
parser1 = nltk.ChartParser(grammar1)
```

Success Test

```
sentence2 = ['Everything','will',
    'be','fine','in','the','end']
```



Error Test

```
sentence1 = [will','be','fine']
```

```
  File "<ipython-input-11-413809792563>", line 15
    sentence1 = [will','be','fine']
                 ^
SyntaxError: invalid syntax
```

Conclusion

This project is useful for the visualization of theoretical algorithm in step by step execution process.

References

1.Zuzak, Ivan, and Vedrana Jankovic. "FSM Simulator." FSM Simulator, ivanzuzak.info/noam/webapps/fsm_simulator/.
2.Hasebe, Yoichiro. "RSyntaxTree." Yohasebe.com, yohasebe.com/rsyntaxtree/
3. "Graphviz." PyPI, pypi.org/project/graphviz/.