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 execution of finite state machines, derivation of sentences and algorithm of Earley's parser animation.

Earley's Parse Animation

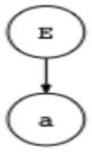
```
g1 = ("S→E", "E→a|b")
x1 = "a"

a1 = Animate(g1,x1,auto_generate=True)
a1.display()
```

next

```
Predict
s[0]: E → •a|b, 0
```


```
Match
s[1]: E → (a)•|b, 0
```



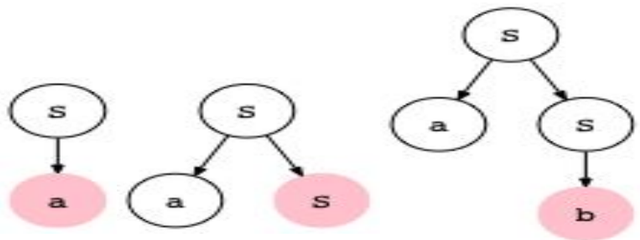

```
===FINISH===
```

Documentation

Documented manually inside Documentation.ipynb, where it contains the description of each class and method.

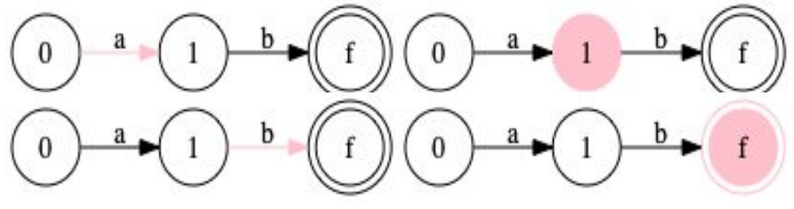
Sentence Derivation Animation

```
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()
```



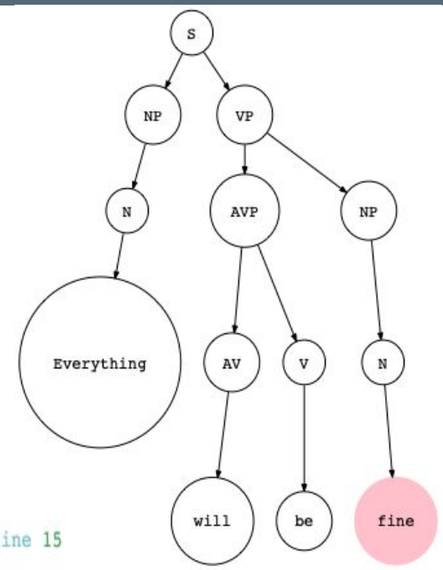
Finite State Machine Animation

```
#user define states
states = ['0', '1', 'f']
#user 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, finals, transitions, check_str)
g.display()
```



Test & 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.



Success test

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

Error test

Statistics

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

Conclusion

- This project is useful for the generation of theoretical algorithm and visualization the step by step execution process.
- In the future, we are hoping to extend the Animation Parsing to be more user-friendly and customizable.

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/.