

Garret Mook, Jorge Calderon, Katie Pell
Dr. Ravikumar
CS454
22 April 2022

Final Project Documentation

Table Of Contents

[RegEx](#)

[RegEx to \$\epsilon\$ -NFA](#)

[\$\epsilon\$ -NFA to \$\epsilon\$ -free NFA](#)

[Work Cited](#)

RegEx

Note: this assumes RegEx Validation is not needed.

Function that parses a string and puts every char in a list:

```
def Parse_Input(s):  
    parseList = []  
  
    for i in s:  
        parseList.append(i.split())  
  
    print(parseList)  
  
s = "(a+a.b)*(a+ $\epsilon$ )"  
Parse_Input(s)
```

Output:

```
[['('], ['a'], ['+'], ['a'], ['.'], ['b'], [')'], ['*'], ['.'], ['('],  
 ['a'], ['+'], [' $\epsilon$ '], [')']]
```

RegEx to ϵ -NFA

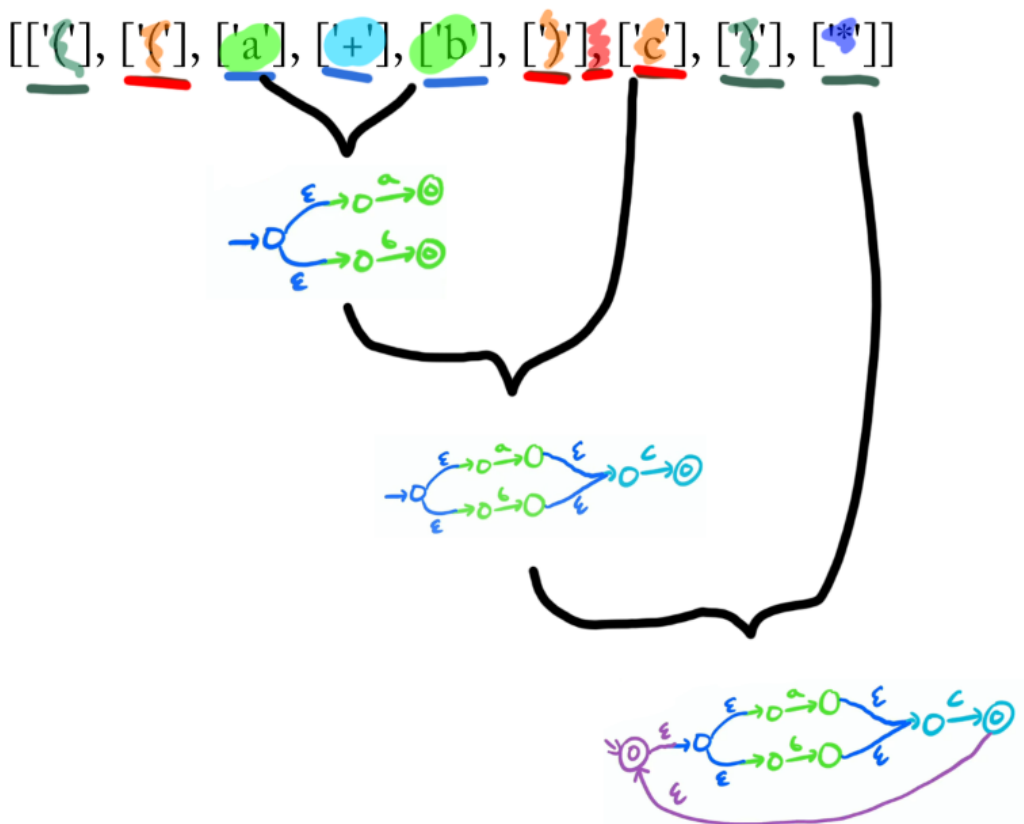
Input for ϵ -NFA

A list of characters

Example:

RegEx = " $((a+b)c)^*$ "

Parse_Input(RegEx) = [['('], ['('], ['a'], ['+'], ['b'], [')'], ['c'], [')'], ['*']]



Info about RegEx of ϵ -NFA

We want to make a NFA out of a given RegEx(1).

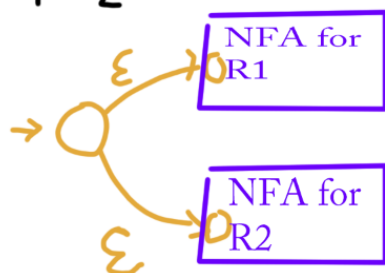
Given: Regex

Want: NFA

RegEx has six possibilities:

1. ϵ
2. \emptyset
3. a (a single character)
4. $R_1 \cup R_2$ (union of 2 RegEx's)
5. $R_1 R_2$ (concatenation of 2 RegEx's)
6. $(R_1)^*$

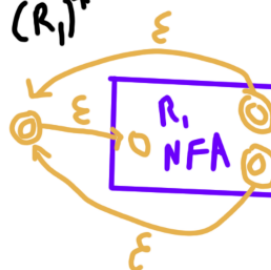
1. $\epsilon \rightarrow \text{NFA with one state and a self-loop}$
2. $\emptyset \rightarrow \text{NFA with no paths}$
3. $a \rightarrow \text{NFA with two states and a transition labeled } a$
4. $R_1 \cup R_2$



5. $R_1 R_2$



6. $(R_1)^*$



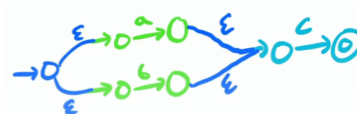
\therefore Given any RegEx we can make a NFA for that particular RegEx. This shows that every language recognized by a regular expression is regular(1).

Let's look at an example of converting a RegEX to a NFA(2)

Ex: $((a \cup b)c)^*$

a : $\rightarrow \text{NFA with two states and a transition labeled } a$
 b : $\rightarrow \text{NFA with two states and a transition labeled } b$
 c : $\rightarrow \text{NFA with two states and a transition labeled } c$

$a \cup b$:
 $\rightarrow \text{NFA with three states and two epsilon transitions from a start state to the start states of the } a \text{ and } b \text{ NFAs. Both } a \text{ and } b \text{ NFAs have their own final states.}$

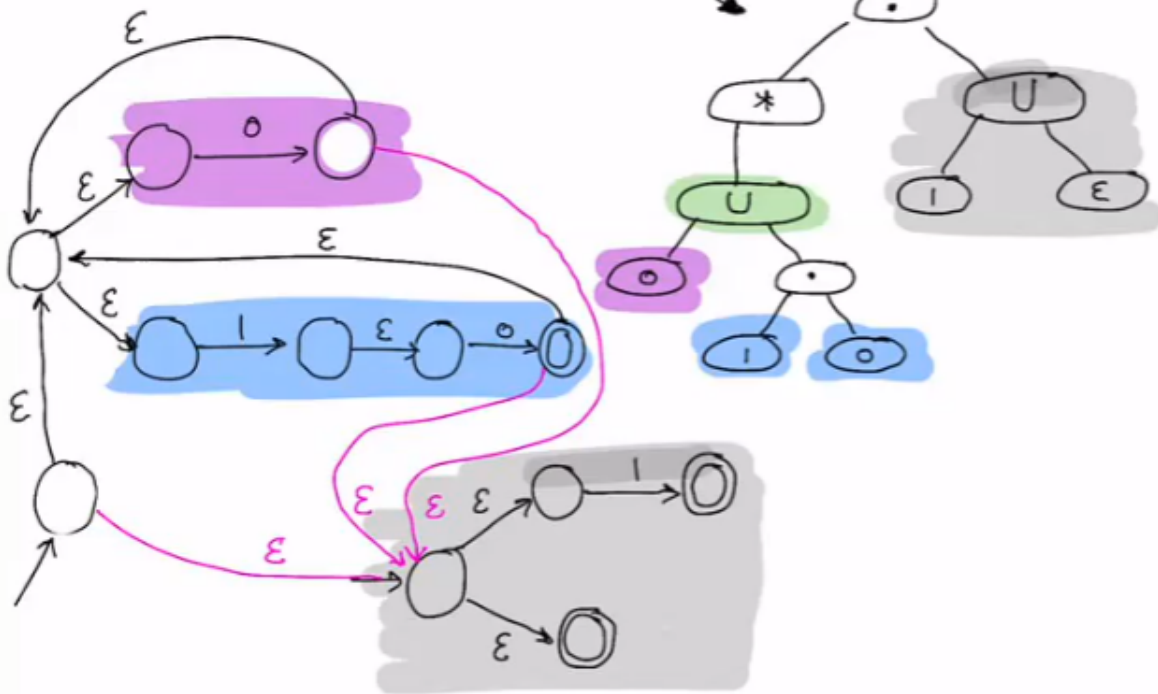


$((a \cup b)c)^*$



Infix to Postfix

Example 2: Convert RE to an equivalent NFA. $R = (0 \cup 10)^* \cdot (1 \cup \epsilon)$



size of a $RE^R = \# \text{ of characters in } R$

Question: If a RE R of size n is converted to an NFA using the above algorithm, what is the size of the resulting NFA?

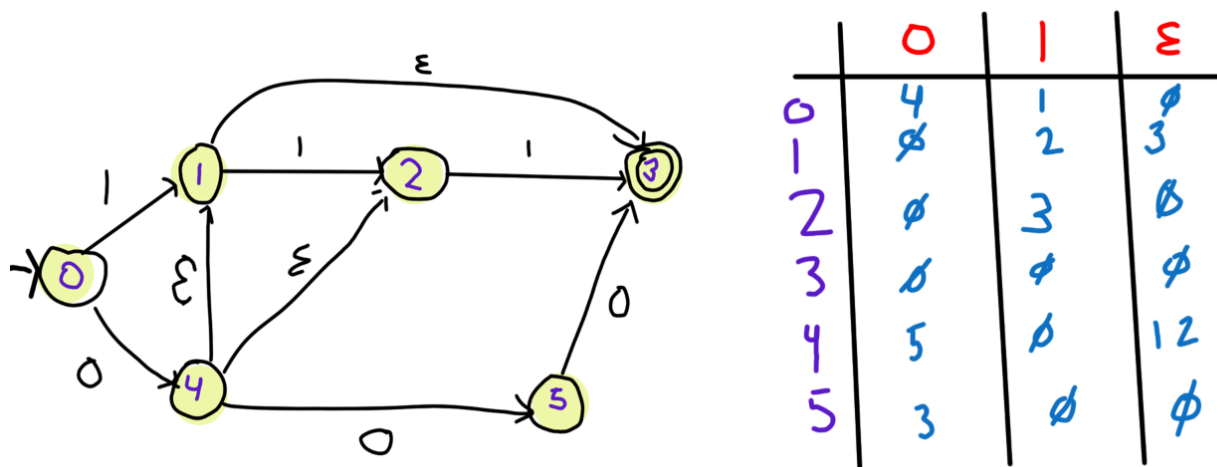
(# of states)

Answer: $\leq cn$ for some constant c .

Output of ϵ -NFA

The output will be a [transition table](#)

Example of a transition table from ϵ -NFA



Example in Python:

```
import pandas as pd
df=pd.DataFrame({'0':[4,{}, {}, {}, {}, 5, 3], '1':[1, 2, 3, {}, {}, {}, {}], 'ε':[{ }, 3, {}, {}, {}, 12, {}]})
print(df)
```

Output:

```

      0  1  ε
0    4  1  {}
1  {}  2  3
2  {}  3  {}
3  {}  {}  {}
4    5  {} 12
5    3  {}  {}
```

ϵ -NFA to ϵ -free NFA

Input for ϵ -free NFA

Transition table

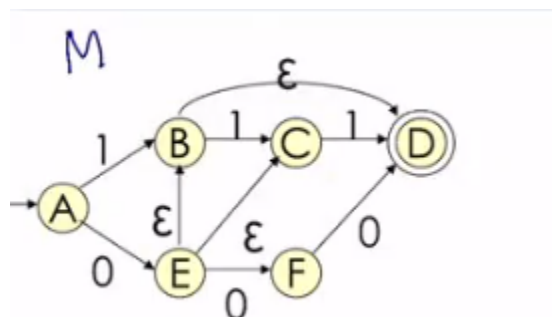
Info about ϵ -free NFA

ϵ -NFA \rightarrow ϵ -free NFA (does not increase the # of states) (4)

Can convert any ϵ -NFA to an equivalent ϵ -free NFA (5)

Example:

Step 1: Compute epsilon closure on ϵ -NFA M



	0	1	ϵ
\rightarrow A	{E}	{B}	\emptyset
B	\emptyset	{C}	{D}
C	\emptyset	{D}	\emptyset
* D	\emptyset	\emptyset	\emptyset
E	{F}	\emptyset	{B, C}
F	{D}	\emptyset	\emptyset

$$\begin{aligned}
 \text{CLO}(A) &= \{A\} \\
 \text{CLO}(B) &= \{B, D\} \\
 \text{CLO}(C) &= \{C\} \\
 \text{CLO}(D) &= \{D\} \\
 \text{CLO}(E) &= \{B, C, D, E\} \\
 \text{CLO}(F) &= \{F\}
 \end{aligned}$$

Step 2: Create the transition table for the ϵ -free NFA M'

M'

	0	1
A	$\{E\}$	$\{B\}$
B	ϕ	$\{C\}$
C	ϕ	$\{D\}$
D	ϕ	ϕ
E	$\{F\}$	$\{C, D\}$
F	$\{D\}$	ϕ

Interesting
closures: $CL(B)$
 $= \{B, D\}$; $CL(E)$
 $= \{B, C, D, E\}$

Example: ϵ -NFA-
to-NFA

	0	1	ϵ
\rightarrow A	$\{E\}$	$\{B\}$	\emptyset
B	\emptyset	$\{C\}$	$\{D\}$
C	\emptyset	$\{D\}$	\emptyset
* D	\emptyset	\emptyset	\emptyset
E	$\{F\}$	\emptyset	$\{B, C\}$
F	$\{D\}$	\emptyset	\emptyset

ϵ -NFA

Since closures of B and E include final state D.

	0	1
\rightarrow A	$\{E\}$	$\{B\}$
* B	\emptyset	$\{C\}$
C	\emptyset	$\{D\}$
* D	\emptyset	\emptyset
E	$\{F\}$	$\{C, D\}$
F	$\{D\}$	\emptyset

Since closure of E includes B and C; which have transitions on 1 to C and D.

39

5

Other Thoughts on implementation

- Have an NFA class that contains a transition table
 - Transition table initializes to be the correct size but empty
 - Make sure that you are updating the fields NOT appending new ones
- Started with ϵ -NFA transition table, found epsilon closures
- See code for detailed comments

Output of ϵ -free NFA

Transition table

Work Cited

- (1) "Regular Expression (Regex) to NFA Conversion - Easy Theory." YouTube, uploaded by Easy Theory, 4 Aug. 2020, www.youtube.com/watch?v=HLOAwCCYVxE.
- (2) "Conversion of Regular Expression (Regex) to NFA Example." YouTube, uploaded by Easy Theory, 5 Aug. 2020, www.youtube.com/watch?v=c-loxIZFeRQ.
- (3) Dr.Ravikumar, Lecture Part 2, Feb 21st 00:25:18
- (4) Dr.Ravikumar, Lecture Par 2, 23 Feb 2022. 00:08:21
- (5) Dr.Ravikumar, Lecture Par 2, 9 Feb 2022. 00:31:23
- (6) (5) Dr.Ravikumar, Lecture Par 1, 14 Feb 2022. 00:43:23