

Section: LR(0) and NFA Lambda Removal (Solution)

Handout written by David Underhill.

Problem 1 Reverse Polish Notation

Reverse Polish notation is a postfix notation. Here is a simple grammar for recognizing this notation:

$$NT = \{ S, O \} ; T = \{ x, +, * \}$$

$$S \rightarrow SSO | x$$

$$O \rightarrow + | *$$

a) Generate the family of LR(0) configuration sets for this grammar.

I_0 :

$$S' \rightarrow \bullet S \quad I_1$$

$$S \rightarrow \bullet SSO \quad I_1$$

$$S \rightarrow \bullet x \quad I_2$$

I_1 :

$$S' \rightarrow S \bullet \quad \text{accept}$$

$$S \rightarrow S \bullet SO \quad I_3$$

$$S \rightarrow \bullet SSO \quad I_3$$

$$S \rightarrow \bullet x \quad I_2$$

I_2 :

$$S \rightarrow x \bullet \quad \text{reduce 2}$$

I_3 :

$$S \rightarrow SS \bullet O \quad I_4$$

$$S \rightarrow S \bullet SO \quad I_3$$

$$S \rightarrow \bullet SSO \quad I_3$$

$$S \rightarrow \bullet x \quad I_2$$

$$O \rightarrow \bullet + \quad I_5$$

$$O \rightarrow \bullet x \quad I_6$$

I₄:

S → SSO● reduce 1

I₅:

O → +● reduce 3

I₆:

O → *● reduce 4

b) Generate the parse table.

State on Top of the Stack	Action				Goto	
	x	+	*	\$	S	O
0	shift 2				1	
1	shift 2			accept	3	
2	reduce 2	reduce 2	reduce 2	reduce 2		
3	shift 2	shift 5	shift 6		3	4
4	reduce 1	reduce 1	reduce 1	reduce 1		
5	reduce 3	reduce 3	reduce 3	reduce 3		
6	reduce 4	reduce 4	reduce 4	reduce 4		

c) Trace through a parse of the input **xx+xx+***

State Stack	Remaining Input	Parser Action
s0	xx+xx+*\$	shift state 2 (consume x)
s0s2	x+xx+*\$	reduce rule 2 → pop 1 state(s), goto(0,S) → push state 1
s0s1	x+xx+*\$	shift state 2 (consume x)
s0s1s2	+xx+*\$	reduce rule 2 → pop 1 state(s), goto(1,S) → push state 3
s0s1s3	xx+*\$	shift state 5 (consume +)
s0s1s3s5	xx+*\$	reduce rule 3 → pop 1 state(s), goto(3,O) → push state 4
s0s1s3s4	xx+*\$	reduce rule 1 → pop 3 state(s), goto(0,S) → push state 1
s0s1	xx+*\$	shift state 2 (consume x)
s0s1s2	x+*\$	reduce rule 2 → pop 1 state(s), goto(1,S) → push state 3
s0s1s3	x+*\$	shift state 2 (consume x)
s0s1s3s2	+*\$	reduce rule 2 → pop 1 state(s), goto(3,S) → push state 3
s0s1s3s3	+*\$	shift state 5 (consume +)
s0s1s3s3s5	*\$	reduce rule 3 → pop 1 state(s), goto(3,O) → push state 4
s0s1s3s3s4	*\$	reduce rule 1 → pop 3 state(s), goto(1,S) → push state 3
s0s1s3	*\$	shift state 6 (consume *)
s0s1s3s6	\$	reduce rule 4 → pop 1 state(s), goto(3,O) → push state 4
s0s1s3s4	\$	reduce rule 1 → pop 3 state(s), goto(0,S) → push state 1
s0s1	\$	accept!

d) Trace through a parse of the input **x+x**

State Stack	Remaining Input	Parser Action
s0	x+x\$	shift state 2 (consume x)
s0s2	+x\$	reduce rule 2 → pop 1 state(s), goto(0,S) → push state 1
s0s1	+x\$	+ cell is empty - reject!

Problem 2 The Lambda Horror NFA Show

Remove all λ s from the NFA in Figure 1.

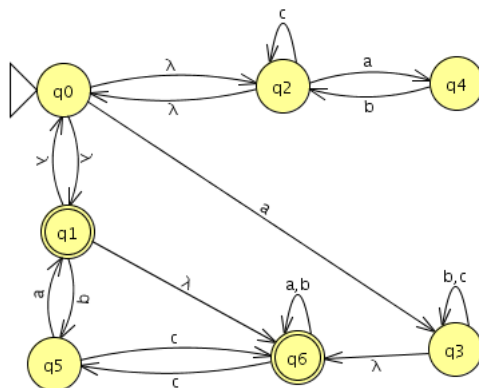


Figure 1: Remove all λ s from the NFA.

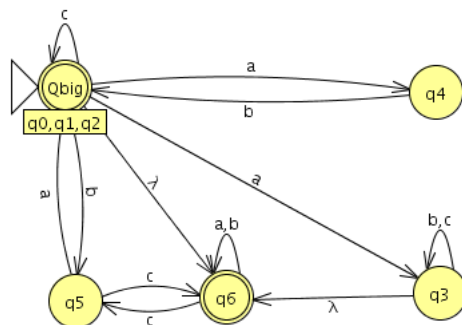


Figure 2: Removed all λ -cycles from the NFA.

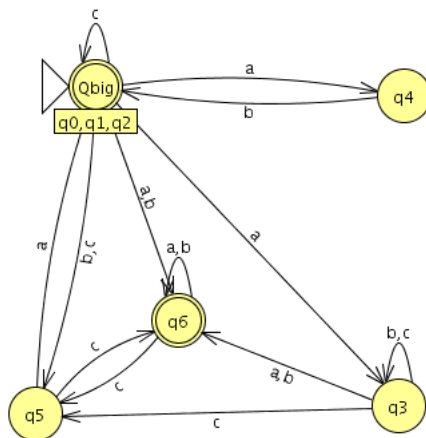


Figure 3: Removed all λ s from the NFA.