Questions & Answers 002 for compiler course

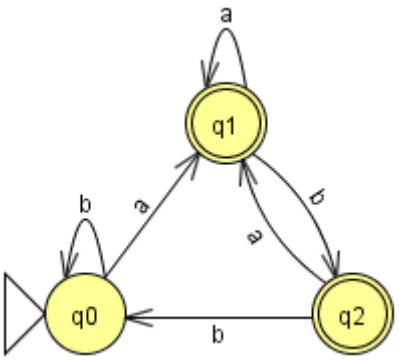
Teacher: Lin Yi , School of Computer Science, Northwestern Polytechnical University
E-mail: ly_cs@nwpu.edu.cn
Mobile Phone: 13572591128

Date: May. 15, 2022

Student Name : ABID ALI
Student ID : 2019380141

【Q10】NFA、DFA、Left/Right Linear Grammar、Regular Expression

【Q10】For the automata in the following figure (q0 is the start state. The triangle on q0 means that it is the start state), answer the following questions.



(1) Judge whether the automata is NFA or DFA? Give the reason for your answer.
(2) Try to draw the left linear grammar and right linear grammar for the automata.
Write the regular expression that could recoganize the same language as the automata.

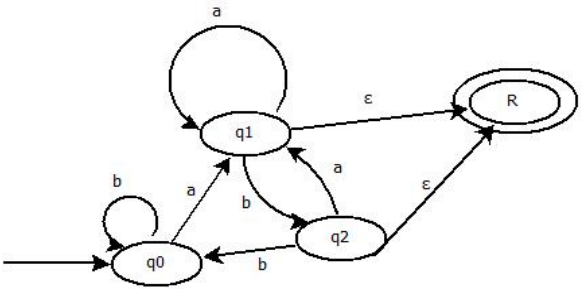
Answer:

(1) It's DFA. Since no labels on transition edges from the same state are the same.
(2) Changing the automata as following (Compute the DFA equal to the NFA firstly and then create the linear grammar is another possible approach）:

①Right Linear grammar G[q0]:

q0->a q1 | b q0
q1->a q1| b q2 | q3
q2->a q1| b q0 | q3

②Left Linear grammar G[R]:



Explaining the modified automata above：

A. Introduce a new state named as R as well as two ε edges from q1 and q2 to R. Then set R as the final state (and relabel q1 and q2 as non-final state).

B. Set q0 as final state (since q0 has a transition pointed to itself.

Left Linear Grammar：

G[q3]

R->q1

R->q2

q1->q0 a | q1 a | q2 a

q2>q1 b

q0->q0 b | q2 b | ε

(3) Regular Expression is as following:

b*aa*(b|baa*)* (bb*aa*(b|baa*)*)*

The following RE is an equivalent RE too:

(b*|aa*b(aa*b)*b)aa*b(aa*b)*(aa*|ε)

【Q11】Regular Expression，NFA，DFA，Deterministic，Minimizing

【Q11】Given regular expression as following:

（(ab)*|b）*（a|(ba)*）a

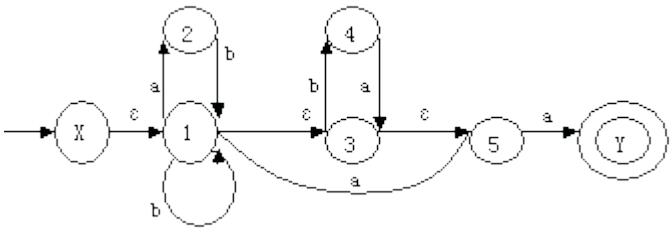
(1) Write the equivalent NFA;

(2) Write the equivalent DFA;

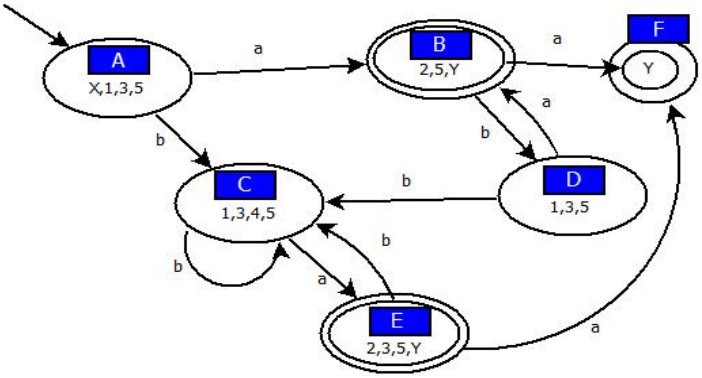
(3) Minimize the DFA;

Answer

(1) NFA is as following



(2) DFA



(3) Minimal DFA

First step, partition the set of states into two set, one only containing final states, the other only containing non-final states： {A，C，E} and {B，E，F}。

Second step, partition these sets again. Suppose we consider a now. then we could calculate the following transitions: {A，C，D}-a->{B，E} are all located in the same set ---- {B，E，F}. Since B and E belong to the same set, {A,C,D} does not need to be partitioned now. But {B，E}-a->{E，F}, and Goto(F,a)=empty. So {B,E,F} could be partitioned into {B、E} and {F}.

Thirdly, re-check input a again: {A，C，D}-a->{B，E} does not to be partitioned. And {B，E}-a->{F} could not be partitioned too.

Fourthly, check input b and then we could find that {A，C，D}-b->{C} does not need to be partitioned. {B,E}-b->{C,D} does not to be partitioned too.

Thus the result in the second step is the minimal DFA.

	a	b
A	B	C
C	E	C
D	B	C
B	F	D
E	F	C
F	Null	Null

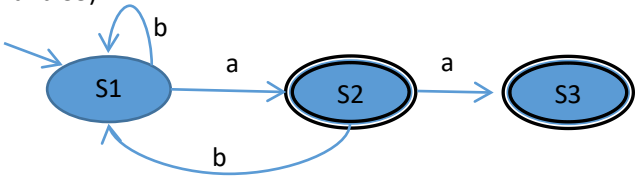
Step 1st, Final state set, non - final state set

	a	b
A	B	C
C	E	C
D	B	C
B	F	D
E	F	C
F	Null	Null

Step 2nd, check input a，separate F from B, E, F. Final result

Minimization (combine A, C, D as new state S1. Combine B, E as new state S2. F as new state S3. The final state are S2 and S3).

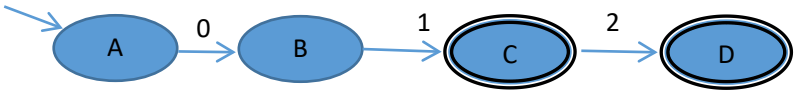
Result:



Explain（4）Proof : when we are minimizing DFA, F has no transition labeled with a. Then F could be separated from any other states.

Ideas to do the proof: counter-evidence

【counter-evidence】



Here, C and D must be separated with each other. Otherwise, the combination operations will introduce 2* which is not required.

【Q12】Given language, write RE or automata

【Q12】Suppose Σ={0，1}，write automata or regular expression or 3-type grammar.

（1）All strings with 0 at the beginning.

Answer

0(0|1)*。

or S->0|S0|S1，or S->0|0S|1S

（2）Strings begin with 0 while end with 1.

Answer

0(0|1)*1

or S->0A1，A->0A|1A|ε

or S->0A，A->1|0A|1A

（3）Strings that have at most one existing of 00 or 11

Answer

Except on 00 or on 11, other 00 and 11 must not appear in the string.

In other words, other part of these strings could only contain (01)* or (10)*.

The result could be:

((01)* | (10)*) (00 | 11 |ε) ((01)* | (10)*)

（4）All strings whose length is an even number

Answer

Here we consider 0 is even too.

Then we could utilize the feature that even number plus even number will get another even number.

(00 | 01 |10 |11)*

BNF grammar could be as following:

S->00 S | 01 S | 10 S | 11 S | ε

But this is not a regular grammar. Make small change will get regular grammar:

S->A S | B S | C S | D S | ε

A->0 A’，A’->0

B->0 B’，B’->0

C->0 C’，C’->0

D->0 D’，D’->0

（5）string as {0，1}+；

Answer

(0|1)*(0|1)，

or (0|1)(0|1)*

（6）{w | w ∈ {0，1}+，w does not include 11}；

Answer

Firstly we calculate w∈ {0，1}*

Without 1：0*

Only have one 1：0*10*

Several 1 that interleaved with at least one 0 between 1s:

0*1 (0 0*1)*0*——length could be 0！

Final result is: (0 | 1) 0*1 (0 0*1)*0*——the green part ensure there are at least one 0 or one 1.

（7）Strings that only contain symbol 0, 1, a in which the count of 0 and 1 is even.

Answer

According to the question, the count of 0 and 1 are even number (0 is allowed).The count of a could be unlimited.

Segment with even count of 0: (A 0 A 0 A)*， where A contains several a and even count of 1.

Segment with even count of 1：(B 1 B 1 B)*，where B contains several a and even count of 0.

B->(0 A 0)* | (a*0a*0a*)*

A->(1 B 1)* | (a*1a*1a*)*

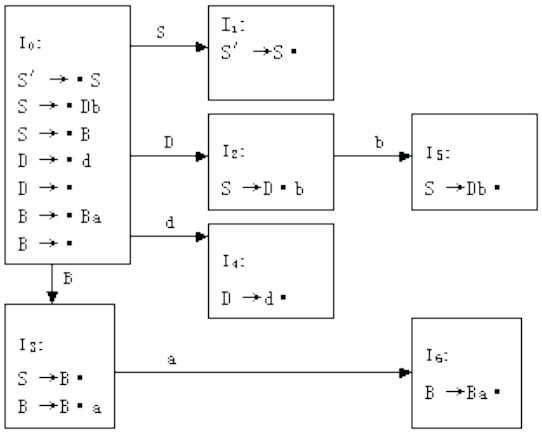
【Q13】Given G[S] to calculate LR(1)

S → Db|B

D → d| ε

B → Ba| ε

LR(0) Automata is as following:



（1）Write LR(0) Table and judge whether this grammar is LR(0)

State	ACTION				GOTO		
	b	d	a	#	S	D	B
0							
1							
2							
3							
4							
5							
6							

（2）Is it SLR(1)?

（3）Write SLR(1) Parsing table， and write the detailed steps to recognize string “aa”.

（4）Write the LR(1) automata as well as LR(1) parsing table.

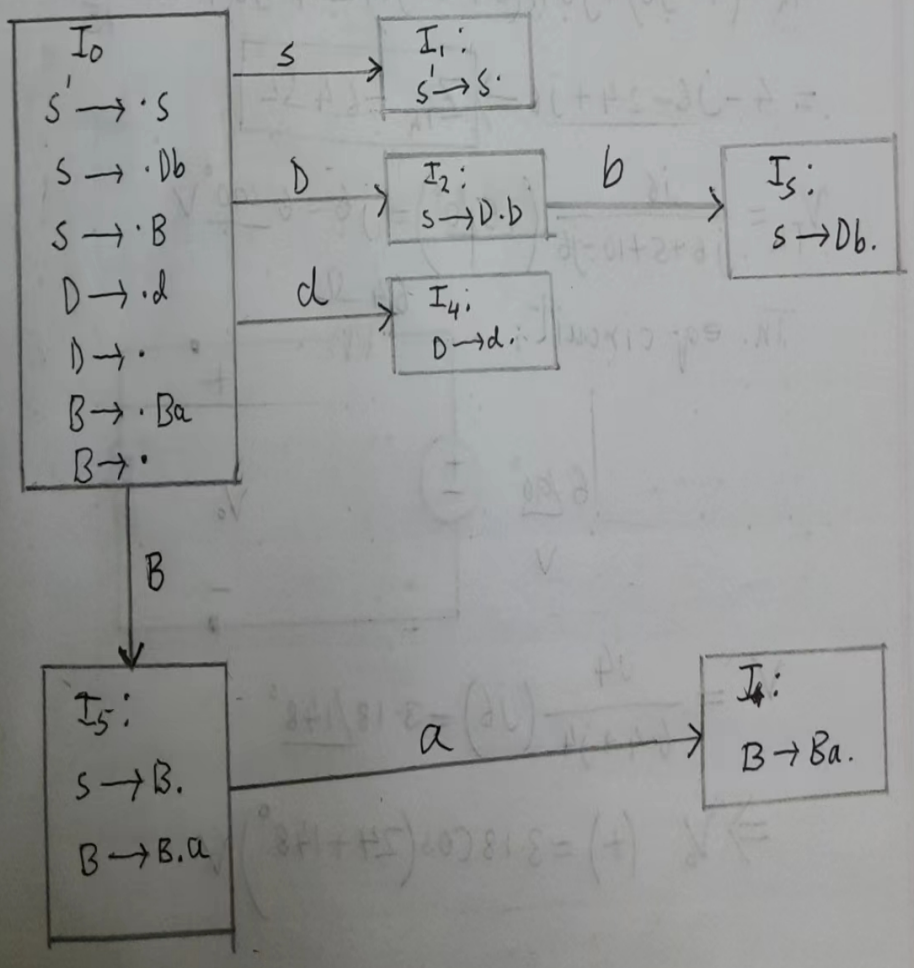
(Q13) Given G[S], To calculate LR(1)

$S \rightarrow Db \mid B$

$D \rightarrow d \mid \varepsilon$

$B \rightarrow Ba \mid \varepsilon$

LR(0) Automata is as following:



① Write LR(0) table and judge whether the grammar is LR(0)

State	ACTION				GOTO		
	b	d	a	#	S	O	B
0		S4			1	2	3
1			accept				
2	S5						
3	R2	R2	S6	R2			
4	R3	R3	R3	R3			
5	R1	R1	R1	R1			
6	R5	R5	R5	R3			

→ Yes, it is a LR(0) grammar

② Is it SLR(1)?

→ Yes.

Q. Write SLR(1) Parsing table and write the detailed steps to recognize string "aa".

State	ACTION				GOTO		
	b	d	a	#	S	D	B
0	R4	S4	R6	R6	1	2	3
1				accept			
2	S5						
3			S6	R2			
4	R3						
5				R1			
6			R5	R5			

State	Parsing Stack	Input	Action
1	#0	aa#	

(4) Write the LR(1) automata as well as LR(1) Parsing table

State 0:
$S' \rightarrow \cdot S, \#$
$S \rightarrow \cdot Db, \#$
$S \rightarrow \cdot B, \#$
$D \rightarrow \cdot d, b$
$B \rightarrow \cdot, b$
$B \rightarrow \cdot Ba, \#/a$
$B \rightarrow \cdot, \#/a$

State 1:
$S' \rightarrow S \cdot, \#$

Q. State 2 : $S \rightarrow D.b, \#$

State 3 : $S \rightarrow B., \#$

$B \rightarrow B.a, \#/a$

State 4 : $D \rightarrow d., b$

State 5 : $S \rightarrow Db., \#$

State 6 : $B \rightarrow Ba., \#/a$

State	ACTION				GOTO		
	b	d	a	#	S	D	B
0	R4	S4	R6	R6	1	2	3
1			accept				
2	S5						
3			S6	R_2			
4	R3						
5				R1			
6			R5	R5			