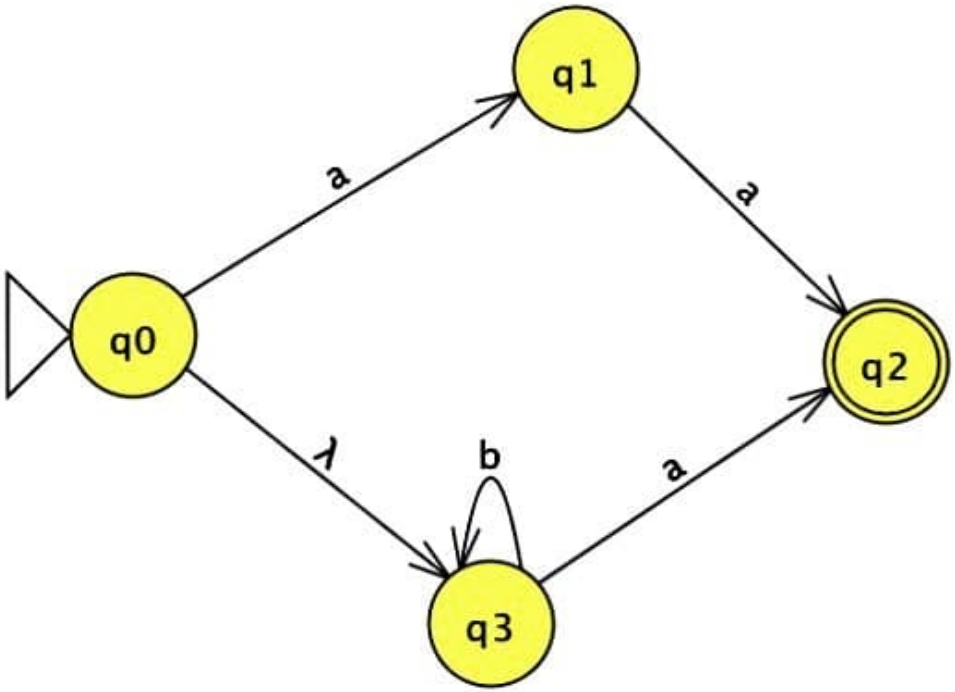


Table Text Size

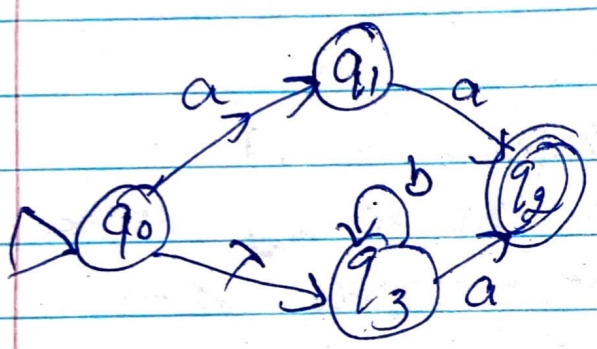
Input	Result
aa	Accept
aaa	Reject
aba	Reject
a	Accept
ab	Reject
baaaa	Reject

a) ~~[illegible]~~

Section 3.1

1) Create an nfa for $\Sigma = \{a, b\}$ that accepts the language L (aa+bb)

Here top part is aa
Bottom part is for b*a



States : $\{q_0, q_1, q_2, q_3\}$

input alphabet = $\{a, b\}$
initial state = q_0
final state $\{q_2\}$

transitions

$\delta(q_0, a) = \{q_1\}$
$\delta(q_0, \lambda) = \{q_3\}$
$\delta(q_1, a) = \{q_2\}$
$\delta(q_3, a) = \{q_2\}$
$\delta(q_3, b) = \{q_3\}$

2 Create regular expression for the set of all strings that consist of an odd number of 'a' followed by 'bb'.

Regular expression for the set of all strings that consist of an odd numbers of a's followed by bb is

$R = (aa)^* abb$

if 0, then $R = (aa)^0 abb$ i.e string abb (here $a = 1$ which is odd followed by bb)

if 1 then $R = (aa)^1 abb$

i.e string aaabb

(here $a = 3$ which is odd followed by b

if 2 then $R = (aa)^2 abb$

i.e : string aaaaabb

(here $a = 5$ which is odd followed by b.

Similarly it accepts all strings which consists of an odd numbers of a's followed by bb.

JFLAP : (Q3.jff)

ile Input Test View Convert Help

Editor | Multiple Run

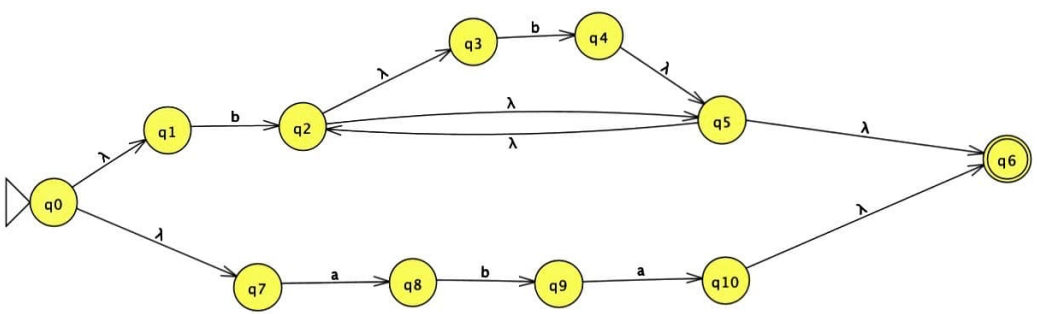


Table Text Size

Input	Result
aa	Reject
aaa	Reject
aba	Accept
a	Reject
ab	Reject
baaaa	Reject

Load Inputs | Run Inputs | Clear | Enter Lambda | View Trace

Section 3.2

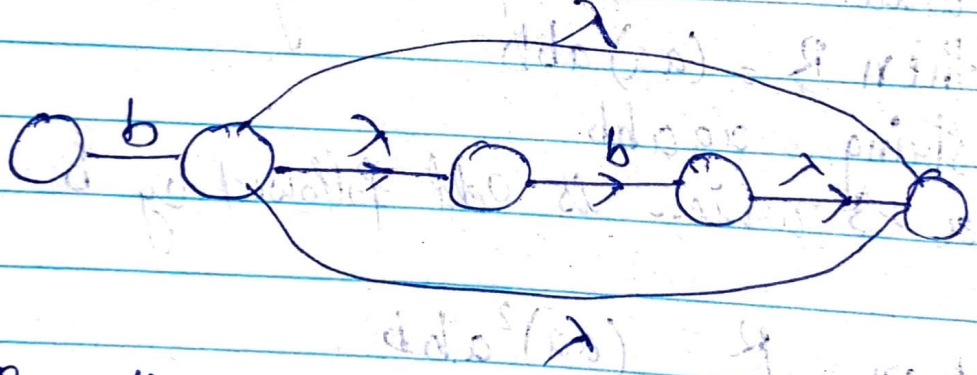
3. Given language is

$L = (bb^* + aba)$

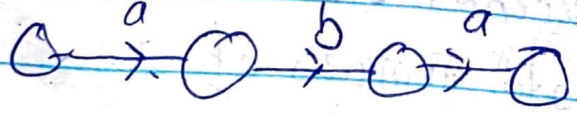
First we have to devide the language into 2 portions

$L(\underbrace{bb^*}_{1} + \underbrace{aba}_{2})$

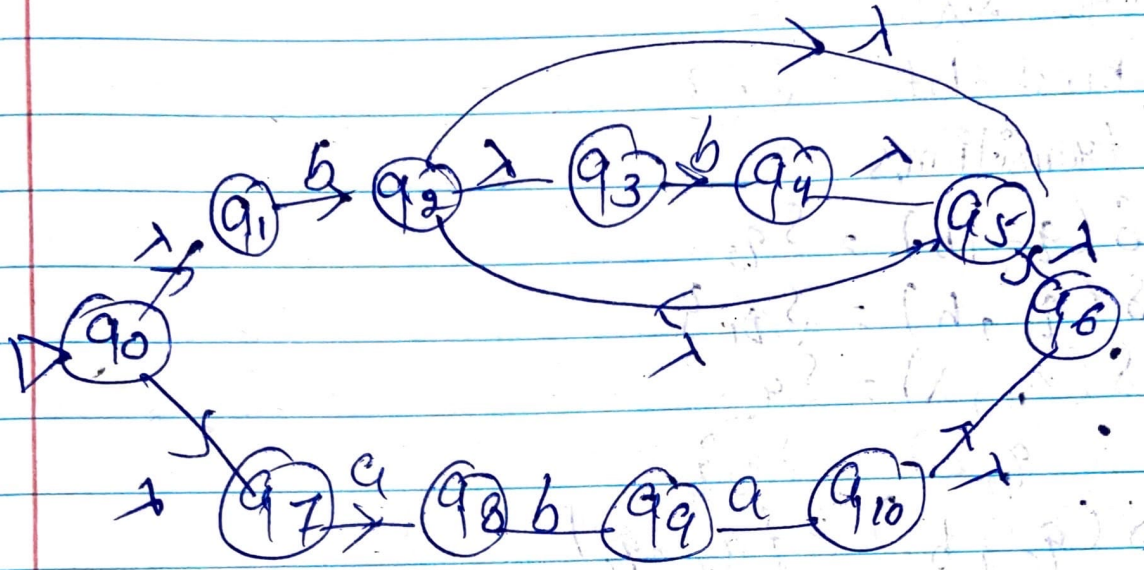
By theorem 3.1 the aytomot for $L(bb^*)$ i.e portion is:



By theorem 3.1 the automata for $L(aba)$ is



Thus, by theorem 3.1 the automata for $L(bb^* + aba)$ is



Here q_0 is initial state

q_6 is final state

states $\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}\}$

transitions

$\delta(q_0, \lambda) = \{q_1\}$

$\delta(q_1, b) = \{q_2\}$

$\delta(q_2, \lambda) = \{q_3, q_5\}$

$\delta(q_3, b) = \{q_4\}$

$\delta(q_4, \lambda) = \{q_5\}$

$\delta(q_5, \lambda) = \{q_2, q_6\}$

$\delta(q_0, \lambda) = \{q_7\}$

$\delta(q_7, a) = \{q_8\}$

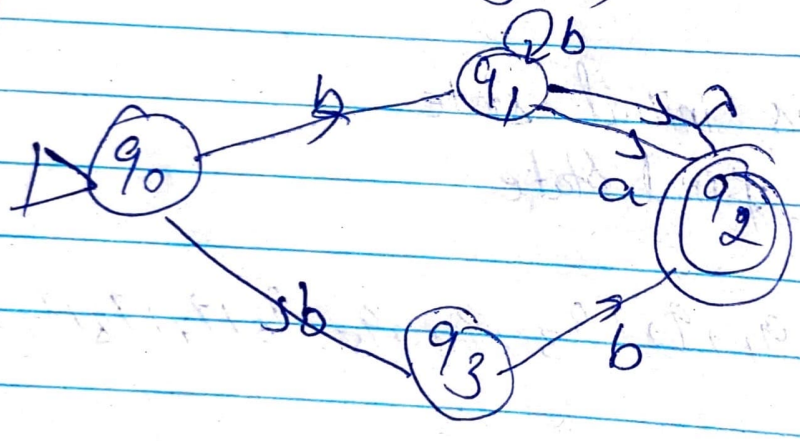
$\delta(q_8, b) = \{q_9\}$

$\delta(q_9, a) = \{q_{10}\}$

$\delta(q_{10}, \lambda) = \{q_6\}$

Section 3.2

(4) States $\{q_0, q_1, q_2, q_3\}$
input: $\{a, b\}$
initial state : q_0
final state : $\{q_2\}$
transitions
$\delta(q_3, b) = \{q_2\}$
$\delta(q_1, b) = \{q_1\}$
$\delta(q_1, \lambda) = \{q_2\}$
$\delta(q_1, a) = \{q_2\}$
$\delta(q_0, b) = \{q_3, q_1\}$



so, regular expression is $(bb^*a + bb)$

(5) Create nfa for $\Sigma = \{a, b\}$

$S \rightarrow abbA \mid bbB$

JFLAP : (Q5.jff)

ile Input Test View Convert Help

Editor Multiple Run

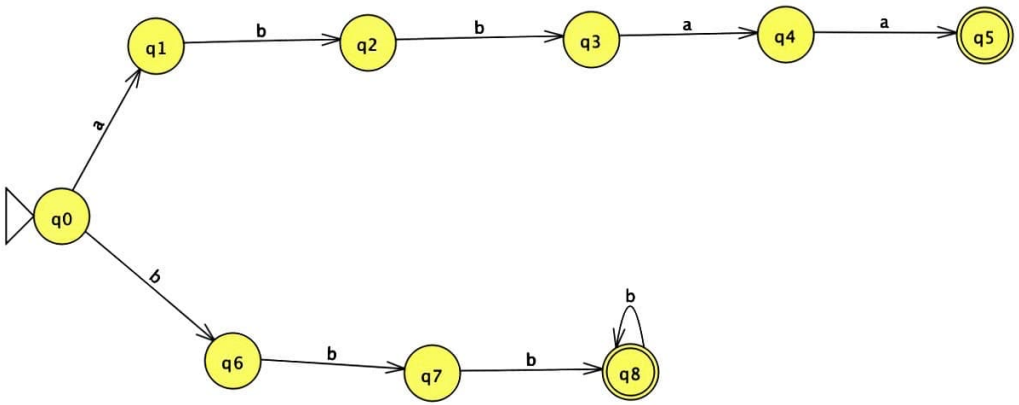


Table Text Size

Input	Result
aa	Reject
ba	Reject
aaba	Reject
babaa	Reject
baaaa	Reject
baabbaaa	Reject
abbabb	Reject

$S \rightarrow abbA \mid bbB$

$A \rightarrow aa \mid a$

$B \rightarrow bB \mid b$

Given grammer is

$S \rightarrow abbA \mid bbB$

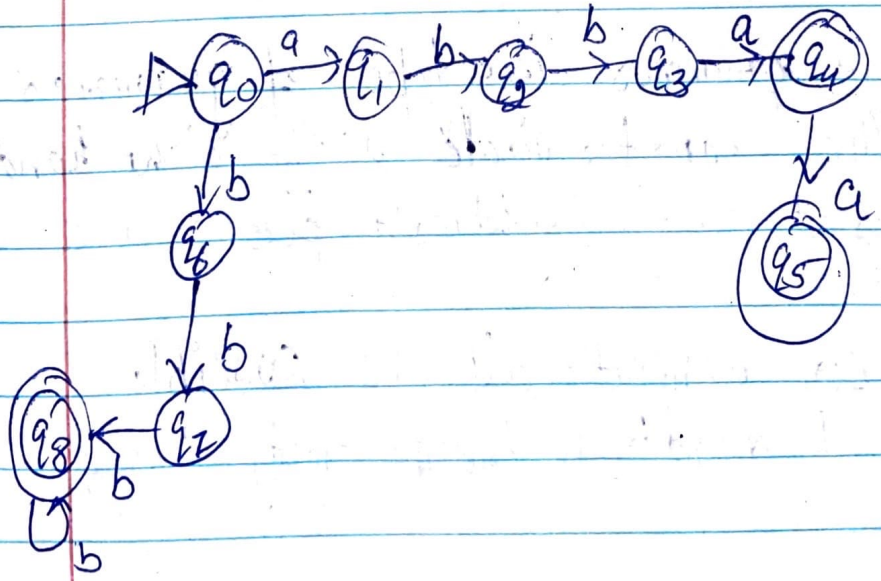
$A \rightarrow aa \mid a$

$B \rightarrow bB \mid b$

The language which is accepted by this grammer is

$L = \{abba, bbb, abbaa, bbbb, \ldots\}$

for this language nfa is



States $\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8\}$

initial state: q_0

final state: $\{q_8, q_4, q_5\}$

Transitions.

$\delta(q_0, a) = \{q_1\}$
$\delta(q_1, b) = \{q_2\}$
$\delta(q_2, b) = \{q_3\}$
$\delta(q_3, a) = \{q_4\}$
$\delta(q_4, a) = \{q_5\}$
$\delta(q_0, b) = \{q_6\}$
$\delta(q_6, b) = \{q_7\}$
$\delta(q_7, b) = \{q_8\}$
$\delta(q_8, b) = \{q_8\}$

(6) Right linear grammer is a type of grammer where all the non-terminals on the right hand side exists at the rightmost place, i.e right side.

$(a+b)^*$ can generate all the possible combination of a and b including empty string

we can start with a starting symbol S

$S \rightarrow aS$

$S \rightarrow bS$

$S \rightarrow \epsilon$