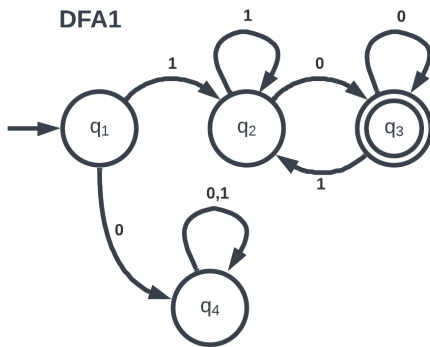


Homework2

Antonio Zea Jr

September 28, 2022

- 1 Design the following finite automata (DFA denotes deterministic and NFA - nondeterministic). For each one, draw the state diagram, implement it for the FASimulator and show its output (copy of the simulator window) for two inputs, one ACCEPTED and one REJECTED
- 1.1 DFA1 that recognizes the language $A_1 = \{w | w \in \{0,1\}^*, w \text{ begins with a 1 and end with a 0}\}$



```
DFA1.dfa x
homework2 > DFA1.dfa
1 states: q1, q2, q3, q4
2
3 input alphabet: 0, 1
4
5 start state: q1
6
7 accept states: q3
8
9 delta: q1, 0 -> q4
10 q1, 1 -> q2
11 q2, 0 -> q3
12 q2, 1 -> q2
13 q3, 0 -> q3
14 q3, 1 -> q2
15 q4, 0 -> q4
16 q4, 1 -> q4
```

DFA/NFA Simulator (copyright ...)

File

type input here: 10110

Back Forward

(* is before symbol about to be read)

input string: *10110

ACCEPTED

DFA1

State	Transitions	Accepting?
q1	0 -> q4, 1 -> q2	
q2	0 -> q3, 1 -> q2	
q3	0 -> q3, 1 -> q2	yes
q4	0 -> q4, 1 -> q4	

DFA/NFA Simulator (copyright ...)

File

type input here: 101101

Back Forward

(* is before symbol about to be read)

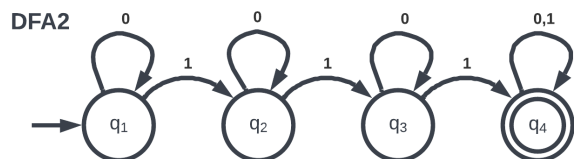
input string: *101101

REJECTED

DFA1

State	Transitions	Accepting?
q1	0 -> q4, 1 -> q2	
q2	0 -> q3, 1 -> q2	
q3	0 -> q3, 1 -> q2	yes
q4	0 -> q4, 1 -> q4	

1.2 DFA2 that recognizes the language $A_2 = \{w | w \in \{0,1\}^*, w \text{ contains at least three 1s}\}$



```

DFA2.dfa
homework2 > dfa2 > DFA2.dfa
1  states: q1, q2, q3, q4
2
3  input alphabet: 0, 1
4
5  start state: q1
6
7  accept states: q4
8
9  delta: q1, 0 -> q1
10     q1, 1 -> q2
11     q2, 0 -> q2
12     q2, 1 -> q3
13     q3, 0 -> q3
14     q3, 1 -> q4
15     q4, 0 -> q4
16     q4, 1 -> q4
  
```

DFA/NFA Simulator (copyright)

File

type input here: 1000101000

Back Forward

(* is before symbol about to be read)

input string: *1000101000

ACCEPTED

DFA2

State	Transitions	Accepting?
q1	0 -> q1, 1 -> q2	
q2	0 -> q2, 1 -> q3	
q3	0 -> q3, 1 -> q4	
q4	0 -> q4, 1 -> q4	yes

DFA/NFA Simulator (copyright)

File

type input here: 000101000

Back Forward

(* is before symbol about to be read)

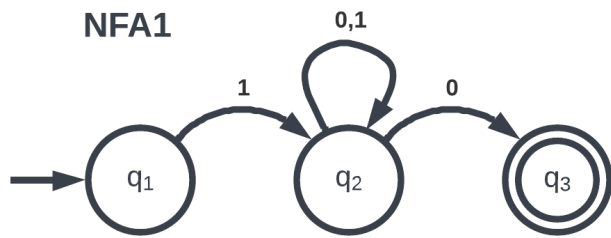
input string: *000101000

REJECTED

DFA2

State	Transitions	Accepting?
q1	0 -> q1, 1 -> q2	
q2	0 -> q2, 1 -> q3	
q3	0 -> q3, 1 -> q4	
q4	0 -> q4, 1 -> q4	yes

1.3 NFA1 for language A_1 with three states



```

NFA1.nfa
homework2 > nfa1 > NFA1.nfa
1  states: q1, q2, q3
2
3  input alphabet: 0,1
4
5  start state: q1
6
7  accept states: q3
8
9  delta:
10  | q1,1 -> q2
11  | q2,0 -> q2, q3
12  | q2,1 -> q2
  
```

DFA/NFA Simulator (copyright)

File

type input here: 10110

Back Forward

(* is before symbol about to be read)

input string: *10110

ACCEPTED

NFA1

State	Transitions	Accepting?
q1	1 -> {q2}	
q2	0 -> {q2,q3}, 1 -> {q2}	
q3		yes

DFA/NFA Simulator (copyright)

File

type input here: 101101

Back Forward

(* is before symbol about to be read)

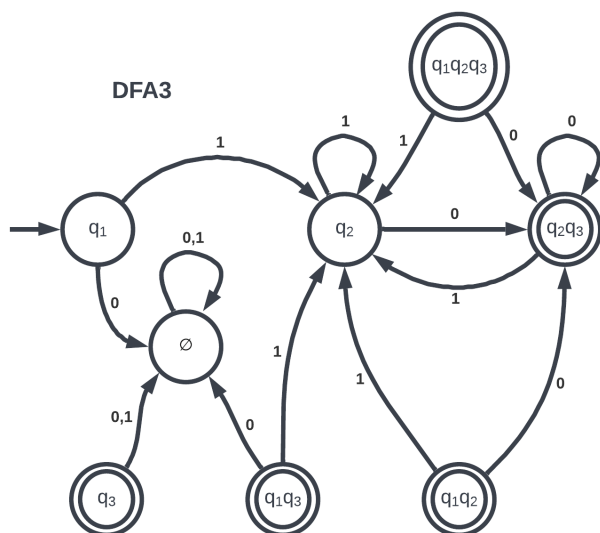
input string: *101101

REJECTED

NFA1

State	Transitions	Accepting?
q1	1 -> {q2}	
q2	0 -> {q2,q3}, 1 -> {q2}	
q3		yes

1.4 DFA3 converted from NFA1 using the proof of Theorem 1.39



```

DFA3.dfa
homework2 > dfa3 > DFA3.dfa
You, 4 minutes ago | 1 author (You)
1 states: ·q1, ·q2, ·q2q3, ·null, ·q3, ·q1q3, ·q1q2, ·q1q2q3
2
3 input·alphabet: ·0, ·1
4
5 start·state: ·q1
6
7 accept·states: ·q2q3, ·q3, ·q1q3, ·q1q2, ·q1q2q3
8
9 delta:·
10   q1, 0 -> ·null
11   q1, 1 -> ·q2
12   q2, 0 -> ·q2q3
13   q2, 1 -> ·q2
14   q2q3, 0 -> ·q2q3
15   q2q3, 1 -> ·q2
16   null, 0 -> ·null
17   null, 1 -> ·null
18   q3, 0 -> ·null
19   q3, 1 -> ·null
20   q1q3, 0 -> ·null
21   q1q3, 1 -> ·q2
22   q1q2, 0 -> ·q2q3
23   q1q2, 1 -> ·q2
24   q1q2q3, 0 -> ·q2q3
25   q1q2q3, 1 -> ·q2
26

```

DFA/NFA Simulator (copyright)

File

type input here: 10110

Back Forward

(* is before symbol about to be read)

input string: *10110

ACCEPTED

DFA3

State	Transitions	Accepting?
q1	0 -> null, 1 -> q2	
q2	0 -> q2q3, 1 -> q2	
q2q3	0 -> q2q3, 1 -> q2	yes
null	0 -> null, 1 -> null	
q3	0 -> null, 1 -> null	yes
q1q3	0 -> null, 1 -> q2	yes
q1q2	0 -> q2q3, 1 -> q2	yes
q1q2q3	0 -> q2q3, 1 -> q2	yes

DFA/NFA Simulator (copyright)

File

type input here: 101101

Back Forward

(* is before symbol about to be read)

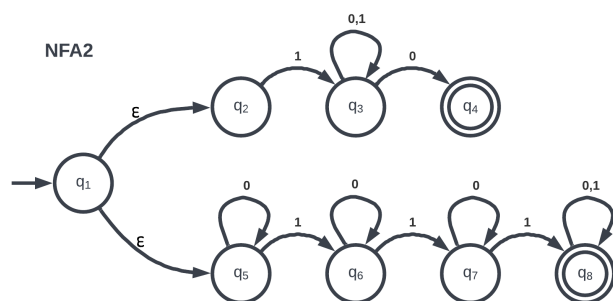
input string: *101101

REJECTED

DFA3

State	Transitions	Accepting?
q1	0 -> null, 1 -> q2	
q2	0 -> q2q3, 1 -> q2	
q2q3	0 -> q2q3, 1 -> q2	yes
null	0 -> null, 1 -> null	
q3	0 -> null, 1 -> null	yes
q1q3	0 -> null, 1 -> q2	yes
q1q2	0 -> q2q3, 1 -> q2	yes
q1q2q3	0 -> q2q3, 1 -> q2	yes

1.5 NFA2 that recognizes the language $A_1 \cup A_2$. Use the construction in the proof of Theorem 1.45



```

NFA2.nfa
homework2 > nfa2 > NFA2.nfa
1  states: q1,q2,q3,q4,q5,q6,q7,q8
2
3  input.alphabet: 0,1
4
5  start.state: q1
6
7  accept.states: q4,q8
8
9  delta: q1, - - -> q2,q5
10 q2,1 -> q3
11 q3,0 -> q3,q4
12 q3,1 -> q2
13 q5,0 -> q6
14 q5,1 -> q6
15 q6,0 -> q5
16 q6,1 -> q7
17 q7,0 -> q6
18 q7,1 -> q8
19 q8,0 -> q7
20 q8,1 -> q8

```

DFA/NFA Simulator (copyright)

File

type input here: 1011

Back Forward

(* is before symbol about to be read)

input string: *1011

ACCEPTED

NFA2

State	Transitions	Accepting?
q1	$\lambda \rightarrow \{q2, q5\}$	
q2	$1 \rightarrow \{q3\}$	
q3	$0 \rightarrow \{q3, q4\}, 1 \rightarrow \{q3\}$	
q4		yes
q5	$0 \rightarrow \{q5\}, 1 \rightarrow \{q6\}$	
q6	$0 \rightarrow \{q6\}, 1 \rightarrow \{q7\}$	
q7	$0 \rightarrow \{q7\}, 1 \rightarrow \{q8\}$	
q8	$0 \rightarrow \{q8\}, 1 \rightarrow \{q8\}$	yes

DFA/NFA Simulator (copyright)

File

type input here: 1001

Back Forward

(* is before symbol about to be read)

input string: *1001

REJECTED

NFA2

State	Transitions	Accepting?
q1	$\lambda \rightarrow \{q2, q5\}$	
q2	$1 \rightarrow \{q3\}$	
q3	$0 \rightarrow \{q3, q4\}, 1 \rightarrow \{q3\}$	
q4		yes
q5	$0 \rightarrow \{q5\}, 1 \rightarrow \{q6\}$	
q6	$0 \rightarrow \{q6\}, 1 \rightarrow \{q7\}$	
q7	$0 \rightarrow \{q7\}, 1 \rightarrow \{q8\}$	
q8	$0 \rightarrow \{q8\}, 1 \rightarrow \{q8\}$	yes

- 2 Write a program in Java (using the Pattern class) for matching regular expressions and strings from the language they describe. The program should print the regex, the string and the result of matching them (true/false). Create the regular expressions describing the languages A_1 and A_2 (from Question 1) and for each one show one string that belongs to the language (matches the regex) and one that does not. Include the source code of the program in your report.

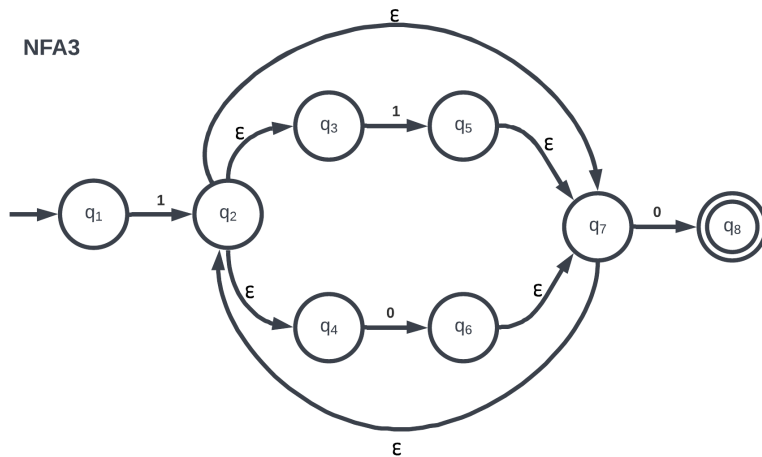
```
homework2]$ /usr/bin/env /usr/lib/jvm/java-11-openjdk/bin/java -cp
./config/Code/User/workspaceStorage/9daf92c87b881617d3daa6194e979cc2/redhat.java
/jdt_ws/cs483_2a10daac/bin homework2
Regex pattern: 1[10]*0
Test string: 1111110
Match: true

Regex pattern: 1[10]*0
Test string: 10000001
Match: false

Regex pattern: [0]*1[0]*1[0]*1[01]*
Test string: 101000010
Match: true

Regex pattern: [0]*1[0]*1[0]*1[01]*
Test string: 1000001
Match: false
```

- 3 Convert the regular expression $1(1 \cup 0)^*0$ to an NFA using the proof of Lemma 1.55. Show the state diagram, implement it for the FASimulator and show its output (copy of the simulator window) for two inputs - one ACCEPTED and one REJECTED. Note that this NFA recognizes the same language as DFA1, NFA1, and DFA3, but it is designed differently.



```

≡ NFA3.nfa ×
homework2 > nfa3 > ≡ NFA3.nfa
1  states: q1,q2,q3,q4,q5,q6,q7,q8
2
3  input alphabet: 0, 1
4
5  start state: q1
6
7  accept states: q8
8
9  delta: q1,1 → q2
10 q2, λ → q3,q4,q7
11 q3,1 → q5
12 q4,0 → q6
13 q5, λ → q7
14 q6, λ → q7
15 q7, λ → q2
16 q7,0 → q8
17
  
```

DFA/NFA Simulator (copyright)

File

type input here: 10110

Back Forward

(* is before symbol about to be read)

input string: *10110

ACCEPTED

NFA3

State	Transitions	Accepting?
q1	1 -> {q2}	
q2	λ -> {q3,q4,q7}	
q3	1 -> {q5}	
q4	0 -> {q6}	
q5	λ -> {q7}	
q6	λ -> {q7}	
q7	0 -> {q8}, λ -> {q2}	
q8		yes

DFA/NFA Simulator (copyright)

File

type input here: 101101

Back Forward

(* is before symbol about to be read)

input string: *101101

REJECTED

NFA3

State	Transitions	Accepting?
q1	1 -> {q2}	
q2	λ -> {q3,q4,q7}	
q3	1 -> {q5}	
q4	0 -> {q6}	
q5	λ -> {q7}	
q6	λ -> {q7}	
q7	0 -> {q8}, λ -> {q2}	
q8		yes