Santiago Bermudez

CSC 28 - Section 05

1. a. The regular expression with the alphabet {c,d} for all strings:

"that begin with c and end with c and there is no more than one d" is:

(2 points)

(i) cdc

(ii) c (c+d)* c

(iii) c^*dc^*

(iv) c^+dc^+

(i) cdc = {cdc}

(ii) $c(c^+d)^*c$ = {cc, ccdc, cccdc, ccccdc, cccccdc, ...}

(iii) c*dc* = {cdc, ccdc, cdcc, ccdcc, cccdc, cdccc, ...}

(iv) c^+dc^+ = {cdc, ccdc, cdcc, ccdcc, cccdc, cdccc, ...}

b. Provide two different possible example strings that the regular expression r = a(a + b)* generates in L(r). In this case, the + symbol has the meaning of Union symbol U. Notice that + is not a superscript.

(2 points)

Example 1: aaa

Example 2: aab

L(r) = {a, a({a}+{b}), a({a}+{b})({a}+{b}), ...}

L(r) = {a, aa, ab, aaa, aab, aba, abb, ...}

c. For the Regular Expression (RE) r = a*(λ + b)*a* Here the + has the meaning of Union symbol U. Notice that + is not a superscript.

Mark True or False next to each string if the string is generated (TRUE) and not generated (FALSE) by the RE r.

(8 points)

(i)	abb
(ii)	bba
(iii)	aa
(iv)	b
(v)	λ
(vi)	ba
(vii)	bbbb
(viii)	a

(i) abb (TRUE)

(ii) bba (TRUE)

(iii) aa (TRUE)

(iv) b (TRUE)

(v) λ (TRUE)

(vi) ba (TRUE)

(vii) bbbb (TRUE)

(viii) a (TRUE)

d. Express in English the meaning of the following Regular Expressions: Here the meaning of a + b is same as a OR b or in other words, a Union b.

(10 points)

(i) a*

(ii) a(bb)*

(iii) (aa)*(bb)*

(iv) (a + b)*

(v) (b)*b

(i) a* = The set of all strings drawn from a.

(ii) a(bb)* = The concatenation of a and the set of all strings drawn from bb.

(iii) (aa)*(bb)* = The concatenation of the set of all strings drawn from aa and the set of all strings drawn from bb.

(iv) (a+b)* = The set of all strings drawn from the union of a and b.

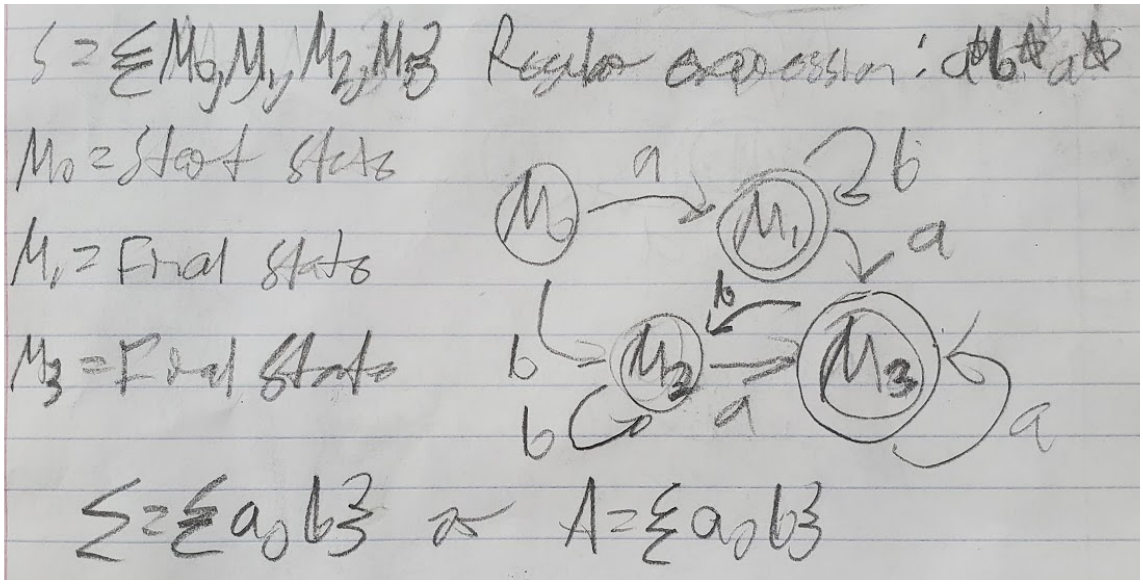
(v) (b)*b = The concatenation of the set of all strings drawn from b and b.

2. (a) Define and draw a finite State Machine (FSA) M1 for the following patterns. Mark your Final States clearly.

all strings that begin OR end with an a.

Note that the alphabet A = {a, b} (in other words contains letters a and b only [Hint: first write the regular expression to make it easy for you].

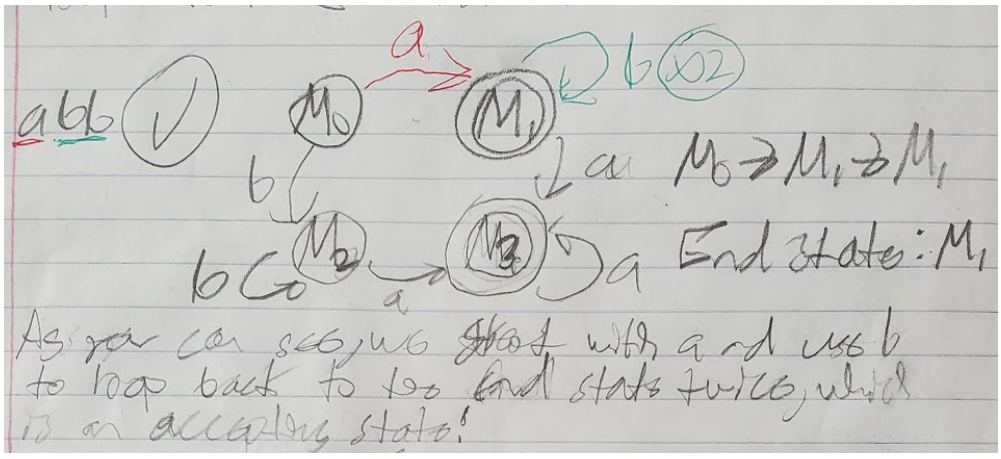
(6 points)



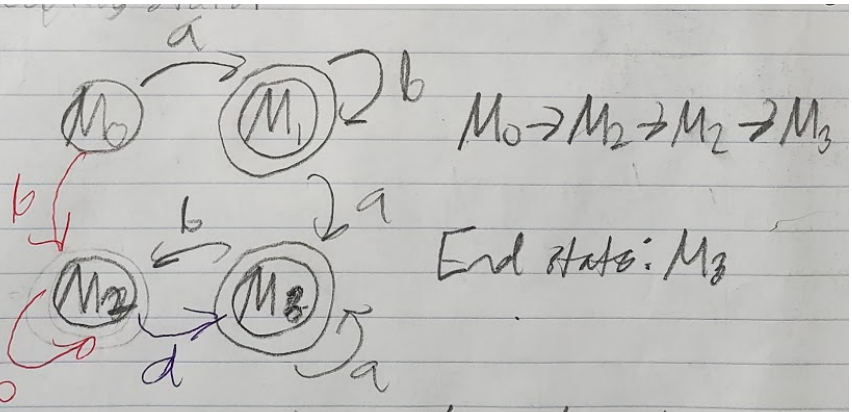
Show that the following strings will be accepted by the FSA M1 (by providing the sequence of states and the type of the end state):

(5 points)

- abb
- bba
- abba
- a
- aa



bba ✓

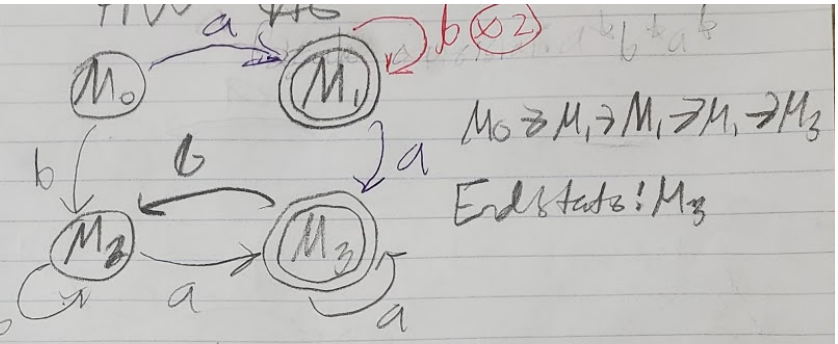


$M_0 \to M_2 \to M_2 \to M_3$

End state: M_3

As you can see, we start with a transition by b, loop back to a state with b again and then we transition with a to to M_3 and state, which is accepting.

abba ✓



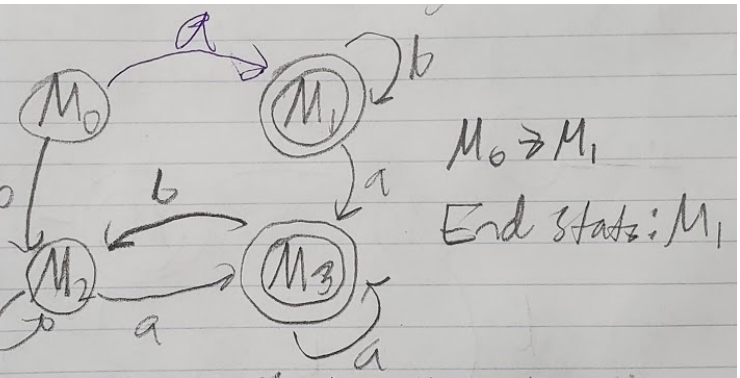
$M_0 \to M_1 \to M_1 \to M_1 \to M_3$

End state: M_3

As you can see, we start with a transition by a, use b to loop back to a state twice, then use a to transition to the end state, which is accepting.

accepting.

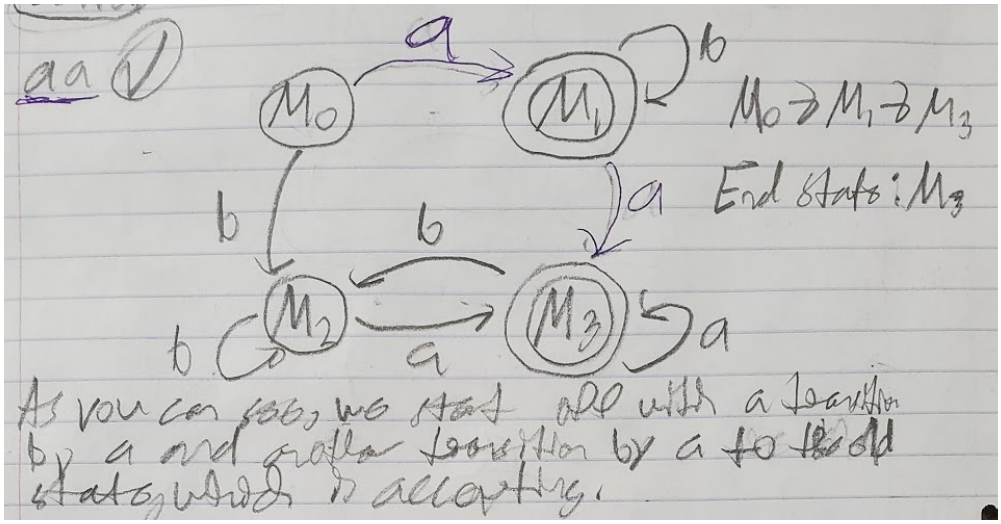
a ✓



$M_0 \to M_1$

End state: M_1

As you can see, we start with a transition by a and get to our end state, which is accepting.

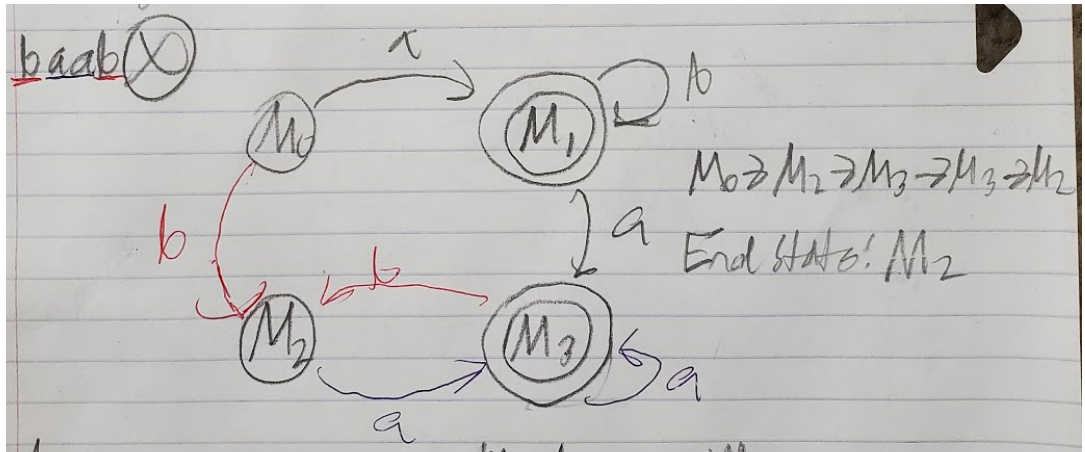


Show that the following strings will not be accepted by the FSA M1 (by providing the sequence of states and the type of the end state):

(5 points)

- baab
- bb
- bbaab
- b
- λ

baab (X)

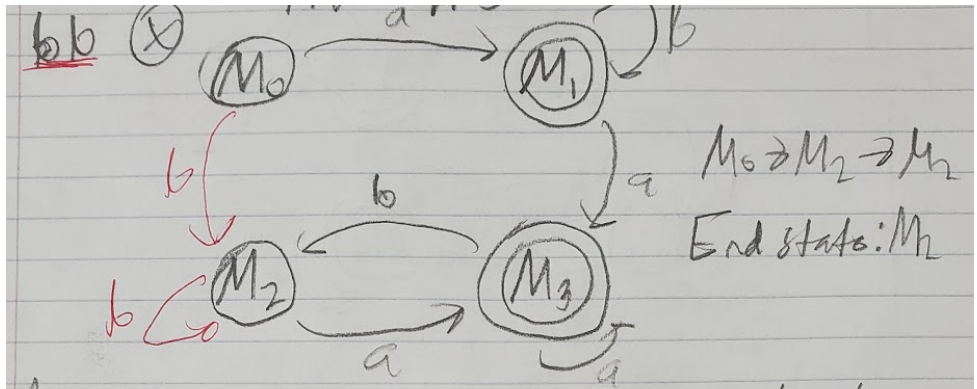


$M_0 \rightarrow M_2 \rightarrow M_3 \rightarrow M_3 \rightarrow M_2$

End state: M_2

As you can see, we start off with a transition by b and another transition by a. We loop back to a state with a and then make a transition with b to the end state which is not accepting.

bb (X)

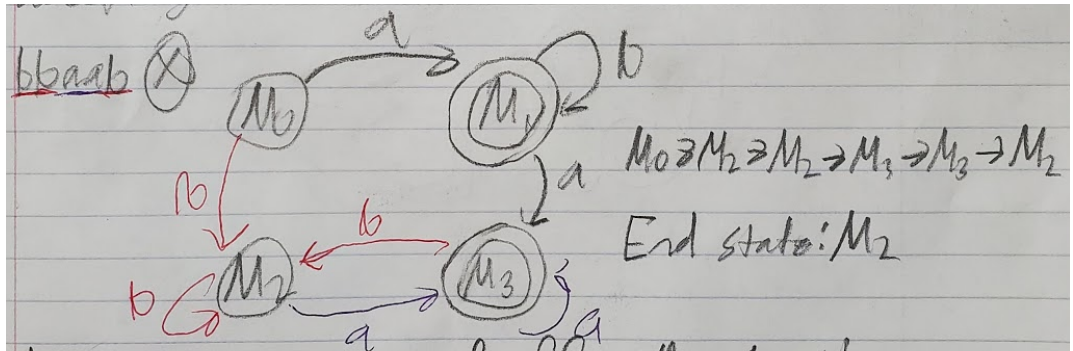


$M_0 \rightarrow M_2 \rightarrow M_2$

End state: M_2

As you can see, we start off with a transition by b, loop back to a state with b and we end up at our end state, which is not accepting.

bbaab ✗

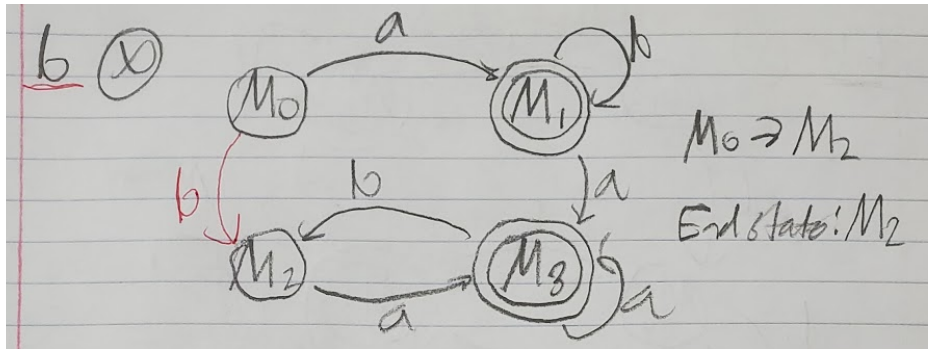


$M_0 \rightarrow M_2 \rightarrow M_2 \rightarrow M_3 \rightarrow M_3 \rightarrow M_2$

End state: M_2

As you can see, we start off with a transition by b, loop back to a state with b and then make another transition with a. From here, we loop back to a state with a and make a final transition to our end state with b, which is not accepting.

b ✗

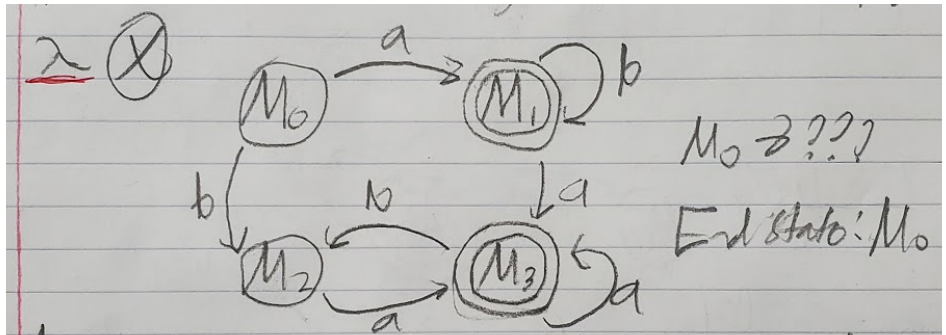


$M_0 \rightarrow M_2$

End state: M_2

As you can see, we start off with a transition by b to our end state, which is not accepting.

λ ✗



$M_0 \rightarrow ???$

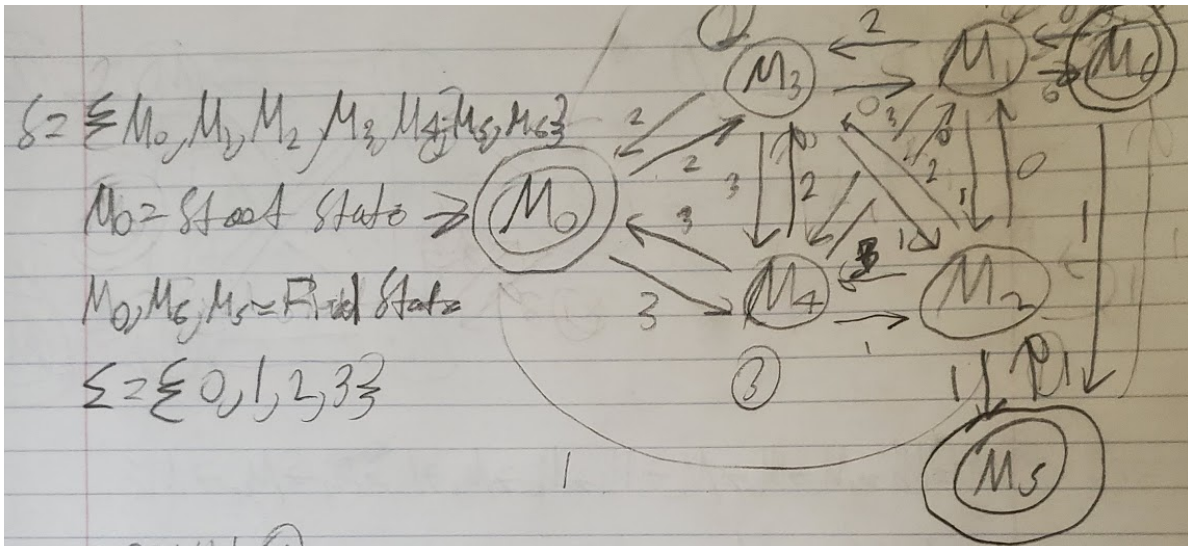
End state: M_0

As you can see here, nothing happens because a lambda means nothing and does nothing. As such, lambda is therefore considered invalid.

(b) Define and draw a finite State Machine (FSA) M2 for the following patterns. Mark your Start State and Final States clearly.

all strings that have even number of 0's followed immediately by even number if 1s. The string may start with anything other than 0 or 1 from the alphabet {0, 1, 2, 3}

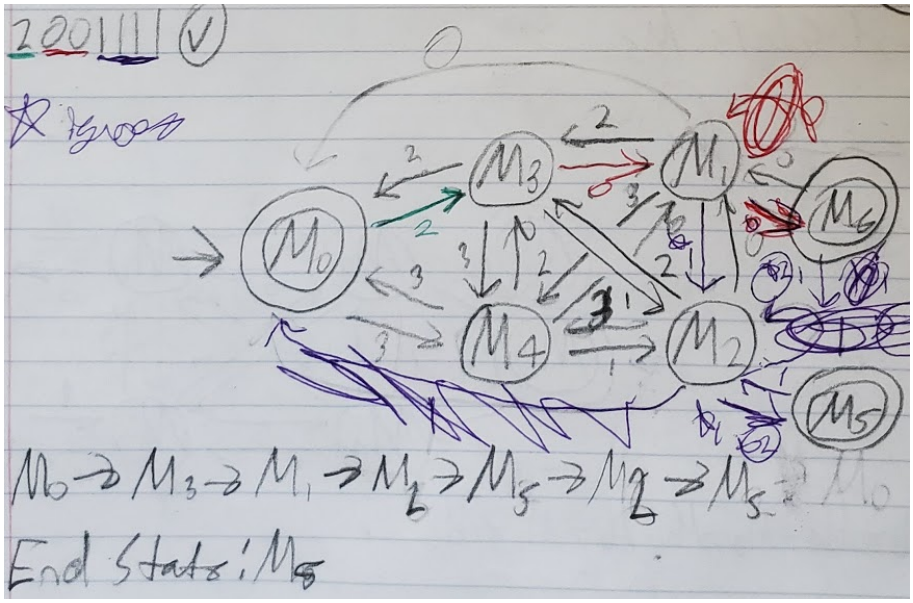
(6 points)



Show that the following strings will be accepted by M2: (by providing the sequence of states and the type of the end state):

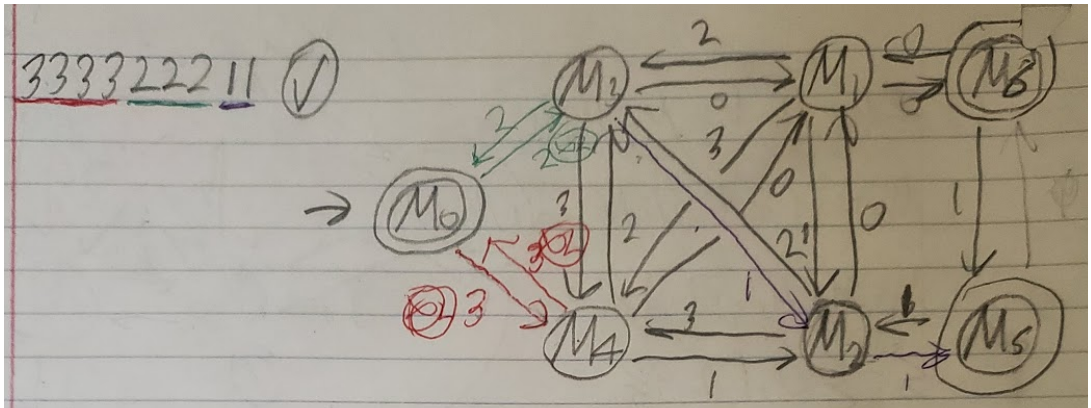
(3 points)

- 2001111
- 333322211
- 323200



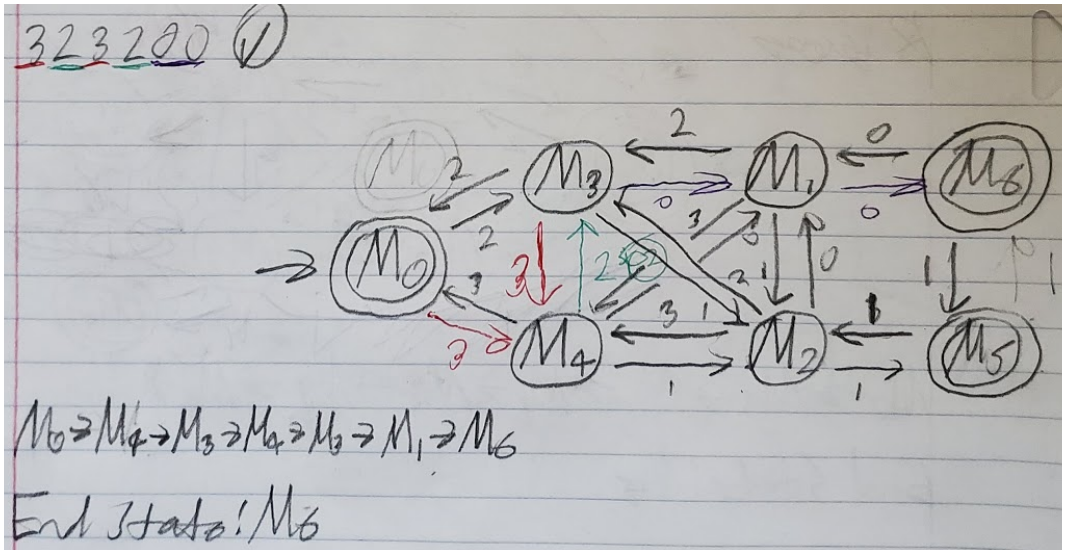
$M_0 \to M_3 \to M_1 \to M_2 \to M_5 \to M_2 \to M_5$

End State: M_5



$M_0 \to M_3 \to M_0 \to M_4 \to M_0 \to M_4 \to M_0 \to M_3 \to M_6 \to M_3 \to M_2 \to M_5$

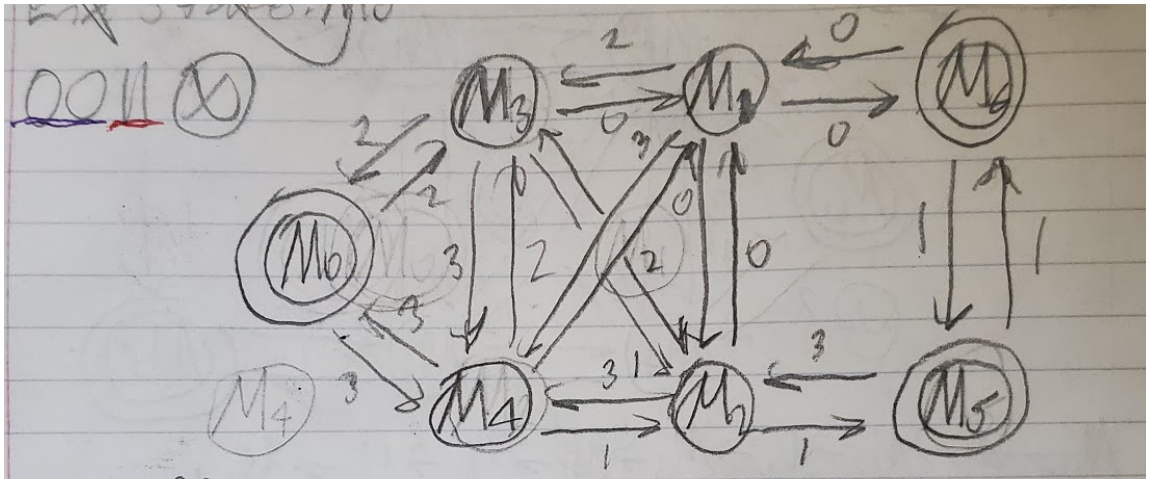
End State: M_5



Show that the following strings will not be accepted by M2: (by providing the sequence of states and the type of the end state):

(3 points)

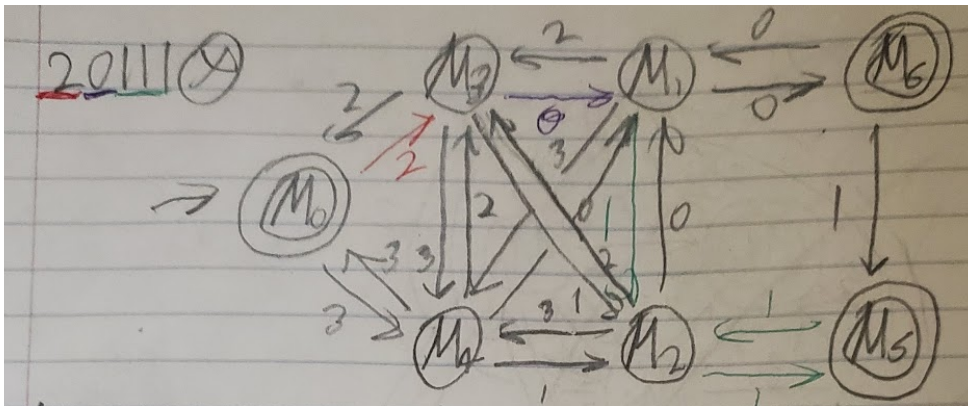
- 0011
- 20111
- 33320101



$M_0 \rightarrow ???$

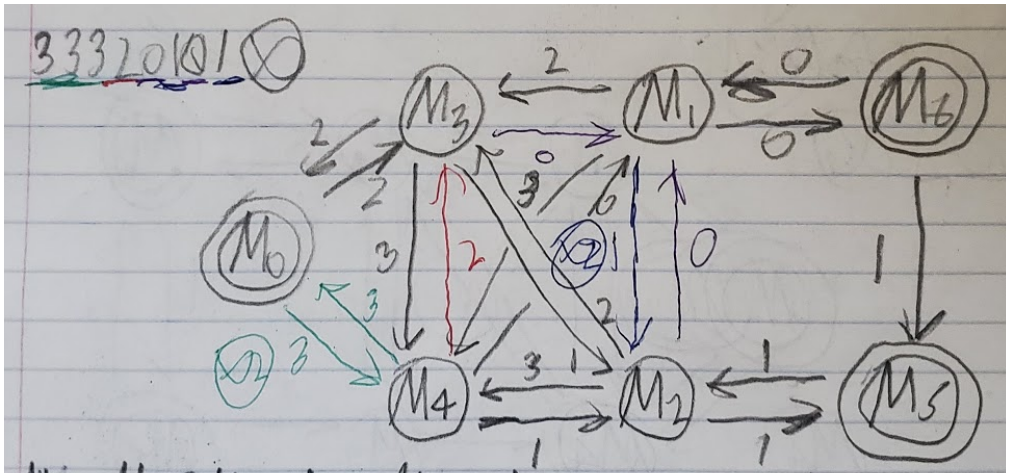
End State: ???

Since we start off with a supposed transition to 0, we go nowhere, so we technically never start, hence can't be accepted.



$M_0 \rightarrow M_3 \rightarrow M_1 \rightarrow M_2 \rightarrow M_5 \rightarrow M_2$

End state: M_2



$M_0 \rightarrow M_4 \rightarrow M_0 \rightarrow M_4 \rightarrow M_3 \rightarrow M_1 \rightarrow M_2 \rightarrow M_1 \rightarrow M_2$

Est stato: M_2