Problem Set 1

Spring 10

Due: Tuesday Feb 9 in class before the lecture.

Please follow the homework format guidelines posted on the class web page:

`http://www.cs.uiuc.edu/class/sp10/cs373/`

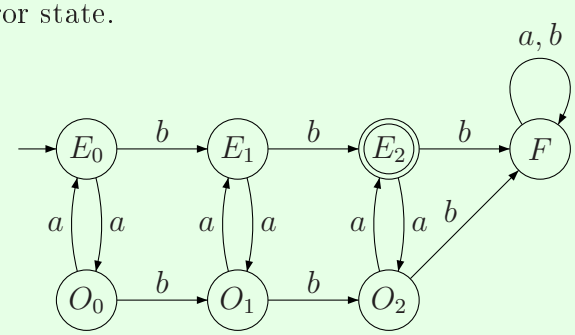
1. DFA building [**Category**: Construction, **Points**: 20]

   Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. Let L be the set of strings s in Σ^* such that s has an even number of a's and exactly two b's. Construct a deterministic finite automaton for L that has at most 7 states. Make sure that your DFA is complete.

   If you find this hard, you can also give a DFA with more states that accepts L for partial credit.

Solution:

Consider the number of a's and b's of the current read prefix of s. We denote even a's by E, odd a's by O. Similarly, the number of b's occured is denoted by 0, 1 and 2, respectively. Combining the two states for a's and the three states for b's yields six states. Moreover, F stands for the error state.



2. [**Category**: Construction, **Points**: 20]

   Your goal is to design an automatic lights control system. The system has a sensor that detects motion in the room and sends the data to controller every 2 seconds; controller turns the lights on when somebody enters the room, and turns off when nobody is in the room. However, due to some uncertainty in the sensor, it may not detect the motion sometimes. That is why we do not want the lights to turn off immediatly after the sensor tells that there's nobody in the room. We want our controller to wait for additional 2 seconds (till the next signal from the sensor), and only then turn the lights off, if the sensor still thinks that the room is empty. We want to ensure that the system works as desired. Your task is to build an automaton to verify the behaviour of the system. Assume that the system starts with the light beign off. Your automaton is given a sequence of events and actions in the alphabet $\Sigma = \{on, off, yes, no\}$, where every odd event is either *no* when the sensor thinks that the room is empty, or *yes* otherwise. Every even event is the response from controller: *on*, if it decides to turn the

lights on, and off otherwise. The given sequence is of even length. Your automaton should decide whether or not the behaviour is legitimate.

Examples of good sequences:

(a) no, off, yes, on

(b) $yes, on, yes, on, no, on, yes, on$

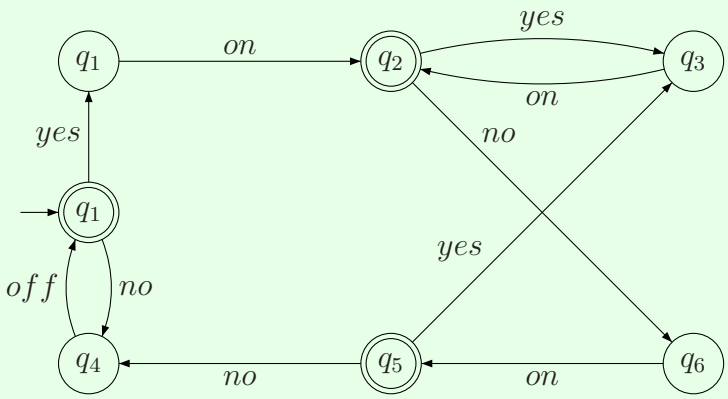
(c) $no, off, yes, on, yes, on, no, on, no, off, yes, on$

Examples of bad sequences:

(a) yes, on, no, off

(b) $no, off, yes, on, yes, off$

Solution:

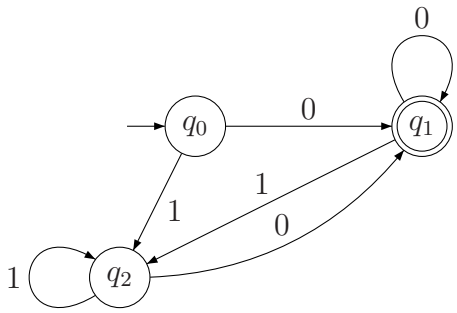


In addition to these states, automaton has a "trap" state q_7, where automata goes if it finds an enexpected event in the sequence. The full δ function is:

δ	on	off	yes	no
q_0	q_7	q_7	q_1	q_4
q_1	q_2	q_7	q_7	q_7
q_2	q_7	q_7	q_3	q_6
q_3	q_2	q_7	q_7	q_7
q_4	q_7	q_0	q_7	q_7
q_5	q_7	q_7	q_3	q_4
q_6	q_5	q_7	q_7	q_7
q_7	q_7	q_7	q_7	q_7

3. [**Category**: Analysis, **Points**: 20]

What is the language of the following DFA. Try to describe its language clearly and as simple as you can (for example in one short sentence).



Solution:

It accepts the set of binary strings that either start with a 0 and have an even number of transitions (from 0 to 1 or from 1 to 0) or start with a 1 and have an odd number of transitions. Note how state q_1 keeps track of a run starting with a 0 while q_2 keeps track of a run starting with a 1.

4. [**Category**: Notation, **Points**: 20]

   Write down the DFA in the previous problem using formal notation (make sure to clearly describe all five important pieces of a DFA).

Solution:

It is the 5-tupe $(Q, \Sigma, \delta, q_0, F)$ where:

$$\Sigma = \{0, 1\}$$
$$Q = \{q_0, q_1, q_2\}$$
$$F = \{q_1\}$$

δ	0	1
q_0	q_1	q_2
q_1	q_1	q_2
q_2	q_1	q_2

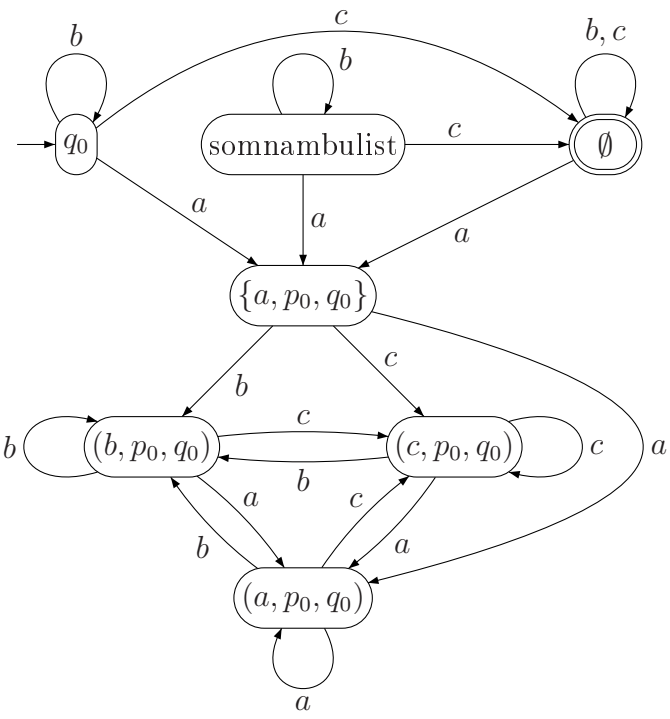
5. [**Category**: Notation, **Points**: 10]

   Draw the state-diagram of the DFA described below.

   DFA $A = (Q, \Sigma, \delta, q_0, F)$ where

   - $\Sigma = \{a, b, c\}$
   - $Q = \{q_0, (a, p_0, q_0), (b, p_0, q_0), (c, p_0, q_0), somnambulist, \{a, p_0, q_0\}, \emptyset\}$
   - $F = \{\emptyset\}$
   - δ is defined as follows:
     - For every $d \in \Sigma$, $\delta(\{a, p_0, q_0\}, d) = (d, p_0, q_0)$.
     - For every $e, d \in \Sigma$, $\delta((e, p_0, q_0), d) = (d, p_0, q_0)$.
     - For every $q \in \{q_0, somnambulist, \emptyset\}$, $\delta(q, c) = \{\}$.

– For every $q \in \{q_0, somnambulist, \emptyset\}$, $\delta(q, a) = \{a, p_0, q_0\}$.
– For every $q \in \{q_0, somnambulist, \emptyset\}$, $\delta(q, b) = q$.



6. [**Category**: Construction, **Points**: 20]

Present a DFA over $\Sigma = \{0, 1\}$ that accepts the set of all strings that, when interpreted in reverse as a binary number, is divisible by 3. Examples: 011 (110 = 6), 0, 0011 (1100 = 12), 01001 (10010 = 18), etc. Prove, that your DFA accepts exactly this language.

Solution:

First Solution: $A = (\Sigma, Q, \delta, q_s, F), Q = \{q_s, q_0^0, q_0^1, q_1^0, q_1^1, q_2^0, q_2^1, \}, F = \{q_0^0, q_0^1\}$

$\delta(q_s, b) = q_b^1$
$\delta(q_r^t, b) = q_k^m$, where $k = (r + b * 2^t)\ mod\ 3$ and $m = (t + 1)\ mod\ 2$

Lemma 1: $\delta^*(q_s, w) = q_k^m$ where $m = |w|\ mod\ 2$
Proof by induction on the length of w.

(a) Base case ($|w| = 1$): $\delta^*(q_s, b) = \delta(q_s, b) = q_b^1$ for $b \in \Sigma$, we have that $m = 1 = 1\ mod\ 2 = |w|\ mod\ 2$

(b) Inductive hypothesis: the statement holds for $|w| = n - 1$

(c) Inductive step: $w = w'b, |w'| = n - 1$. $\delta^*(q_s, w) = \delta(\delta^*(q_s, w'), b) = \delta(q_r^t, b)$ where $t = |w'|\ mod\ 2 = (n - 1)\ mod\ 2$. By definition of δ: $\delta(q_r^t, b) = q_k^m$ where $m = (t+1)\ mod\ 2 = ((n-1)\ mod\ 2+1)\ mod\ 2 = (n-1+1)\ mod\ 2 = n\ mod\ 2 = |w|\ mod\ 2$

Lemma 2: $\delta^*(q_s, w) = q_k^m$ where $k = w^R \ mod \ 3$
Proof by induction on the length of w.

(a) Base case ($|w| = 1, w = b$): $\delta^*(q_s, b) = \delta(q_s, b) = q_b^1$ where $b = b \ mod \ 3 = b^R \ mod \ 3 = w^R \ mod \ 3 = k$

(b) Inductive hypothesis: the statement holds for $|w| = n - 1$

(c) Inductive step ($w = w'b, |w'| = n - 1$):

i. $w = w'0$. Let $\delta^*(q_s, w') = q_r^t$ and $\delta^*(q_s, w') = \delta(q_r^t, 0) = q_k^m$.
By definition of δ: $k = (r \ + \ 0 * 2^t) \ mod \ 3 = r \ mod \ 3 = w'^R \ mod \ 3$. Since $w^R = (w'0)^R = 0w'^R$, and w^R and w'^R are the same in binary $k = w'^R \ mod \ 3 = w^R \ mod \ 3$.

ii. $w = w'1$. Let $\delta^*(w', q_s) = q_r^t$ and $\delta^*(w, q_s) = \delta(q_r^t, 1) = q_k^m$.
By definition of δ: $k = (r \ + \ 1 * 2^t) \ mod \ 3 = ((r \ mod \ 3) + (2^t \ mod \ 3)) \ mod \ 3$, which is equal to $(|w'^R| \ mod \ 3) + (2^t \ mod \ 3)) \ mod \ 3$, by inductive hypothesis.
Since $w^R = 1w'^R$ is equal to $2^{n-1} + w'^R$ in binary, $w^R \ mod \ 3 = (2^{n-1} + w'^R) \ mod \ 3 = ((w'^R \ mod \ 3) + (2^{n-1} \ mod \ 3)) \ mod \ 3$.
We have two cases: n is even and n is odd. When n is even, $2^{n-1} \ mod \ 3 = 2$ and, by Lemma1, $t = |w'| \ mod \ 2 = (n - 1) \ mod \ 2 = 1$, and hence $(2^t \ mod \ 3)$ is also equal to 2. When n is odd, $2^{n-1} \ mod \ 3 = 1$ and, by Lemma1, $t = |w'| \ mod \ 2 = (n - 1) \ mod \ 2 = 0$, and hence $(2^t \ mod \ 3)$ is also equal to 1.
$k = (|w'^R| \ mod \ 3) + (2^t \ mod \ 3)) \ mod \ 3 = ((w'^R \ mod \ 3) + (2^{n-1} \ mod \ 3)) \ mod \ 3 = w^R \ mod \ 3$

By Lemma2 and definition of DFA ($F = \{q_0^0, q_0^1\}$), A accepts w iff $w^R \ mod \ 3 = 0$, that is w^R is divisible by 3.

Note: this is not the smallest DFA that accepts the desired language.

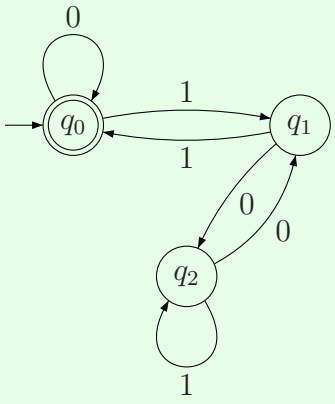
Second Solution:

Assume a number t is given in binary (and not in reverse binary). Let t's representation in binary be $b_n b_{n-1} \dots b_0$, i.e. $n = \Sigma_{i=0}^n b_i 2^i$.

Let $x\%3$ represent the least number k, with $0 \leq k < 3$ with the property that there is a number $n \in \mathbb{N}$ such that $3n + k = x$. (In other words, k is the remainder when x is divided by 3.)

Now assume that we have read a binary representation $b_n \dots b_j$, and the number that corresponds to this is r. Then, when we read the next bit b_{j-1}, the number represented by $b_n \dots b_{j-1}$ is clearly $r' = 2r + b_{j-1}$. Assume that we have calculated already $k = r\%3$. Then clearly $r'\%3 = (2r + b_{j-1})\%3 = ((2r\%3) + b_{j-1})\%3 = (2 * (r\%3) + b_{j-1})\%3 = (2k + 1)\%3$. Hence an automaton can compute the remainder as it reads the word: when reading a binary representation of a number t, say $b_n b_{n-1} \dots b_0$, it can keep track of the number represented by the prefix read so far mod 3, by updating the remainder using the above rule. The above

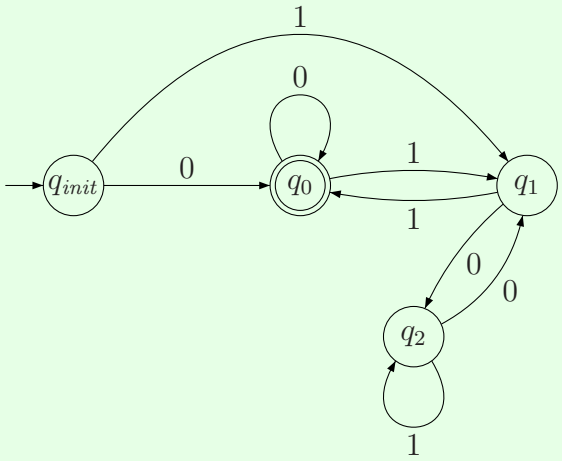
rule says that if the current remainder is k ($k = 0/1/2$) and the automaton reads i ($i = 0/1$), then the new value of the remainder is $(2k + i)\%3$. We hence get the following automaton:



Now, we want to reverse this automaton to accept the reversal of the binary strings. So let us reverse all the edges. Usually, doing this for a DFA gives an NFA, but we are lucky as the reversal of the above gives a DFA. In fact, it's the same DFA!

Now, we need to handle the border cases. We need to reject ϵ, as it does not correspond to any number. So we add a new initial state that is not final, such that on reading any letter, it goes to the state q_0 goes to on reading the same letter.

Hence we get the required DFA:



Aside: Given that the automaton constructed above (without handling ϵ) is its own reversal in fact proves the following fact: any number written in binary is divisible by three iff the number corresponding to the reversal of the binary string is also divisible by three!