
Mohamed G. Gouda — CS 337
Fall 2008 — Midterm 2

Open Book — 75 minutes

---

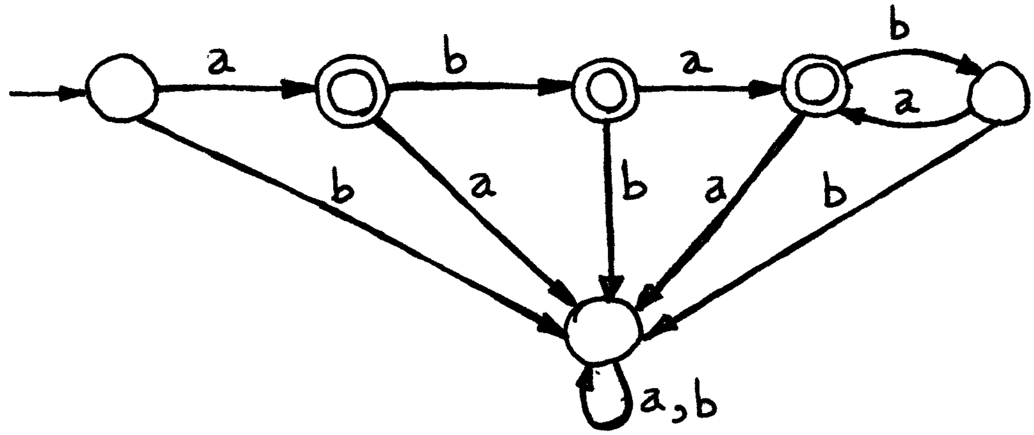
1. Consider the following three regular expression equations:

```
        x = a(y|b)
        y = emptystring | z
        z = bay
```

where x, y, and z are variables whose values are regular expressions over the alphabet {a, b}. Compute the regular expression RE.x of variable x. Then design a Finite State Machine that accepts every string specified by the regular expression RE.x.

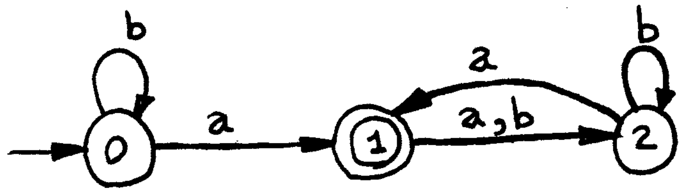
Solution:

```
x = a (y | b)              (1)
y = emptystring | z        (2)
z = bay                    (3)

From (2) and (3),
     y = emptystring | bay
     y = (ba)*                      (4)

From (1) and (4),
     x = a((ba)* | b)
       = a(ba)* | ab

RE.x = a(ba)* | ab
```



---

2. Consider the following Finite State Machine, that has three states, over the alphabet {a, b}.



For each state i in this machine, define a regular expression RE.i specifying every string that starts at the initial state of the machine, state 0, and ends at state i.

Solution:

```
RE.0 = b*
RE.1 = b*a((a|b)b*a)*
RE.2 = b*a((a|b)b*a)*(a|b) b*
```

---

3. A list of integers xs is said to be smaller than a list of integers ys if and only if the length of xs is at most the length of ys and the value of each integer in xs is at most the value of the corresponding integer in ys. Define a Haskell function minl that takes two lists of integers xs and ys and returns one of the following lists:

```
        xs  if xs is smaller than ys
        ys  if ys is smaller than xs
        [ ] otherwise
```

Solution:

```
smaller x:xs [] = False

smaller [] ys = True

smaller x:xs y:ys
        | x <= y = smaller xs ys
        | x > y = False

minl xs ys
     | smaller xs ys = xs
     | smaller ys xs = ys
     | otherwise = []
```

---

4. It is required to design a code that consists of eight codewords such that it is possible to correct one bit inversion in each transmitted codeword. What are the three methods that we discussed in class to design this code? Use each one of these methods to compute one codeword in this code. Which of the three methods is most efficient in this case? And which of them is least efficient in this case?

Solution:

The three methods to compute this code are

i. The trivial method
ii. Hamming code
iii. Reed-Muller code

- One codeword in the code designed by i is 0000000
- One codeword in the code designed by ii is 00C40C2C1

  C1 = C2 = C4 = 0

- One codeword in the code designed by iii is 11111111

ii is most efficient in this case
iii is least efficient in this case
