CS 273: Intro to Theory of Computation, Fall 2007
Problem Set 4 Solutions

1. **Interpreting and constructing regular expressions**

   Let $\Sigma = \{a, b\}$. Give regular expressions for the following languages.

   (a) $L_1 = \{w | w \text{ has three } a\text{'s in a row}\}$

   (b) $L_2 = \{\#w_1\#w_2\# \mid w_1 \text{ is any string such that every } a \text{ is followed by a } b \text{ and } w_2 \in \Sigma^*\}$

   (c) $L_3 = \{w | w \text{ ends in } aab \text{ followed by three arbitrary characters from } \Sigma\}$

   Give a brief description of the language represented by the following regular expressions. Assume the alphabet is $\Sigma = \{a, b\}$.

   (d) $(ab^*)^*bab$

   (e) $((a \cup b)(a \cup b)(a \cup b))^*aa$

   (f) $(a^* \cup b^*)^*$

   Solution

   (a) $(a \cup b)^*aaa(a \cup b)^*$

   (b) $\#(b^*(a^*b)^*b^*)^*\#(a \cup b)^*\#$

   (c) $(a \cup b)^*aab(a \cup b)(a \cup b)(a \cup b)$

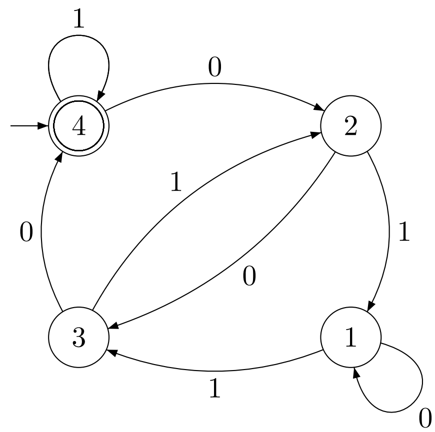
   (d) $L_4 = \{wbab \mid \text{ Every } b \text{ in } w \text{ is preceeded by an } a\}$

   (e) $L_5 = \{waa \mid |w| \text{ is divisible by } 3\}$

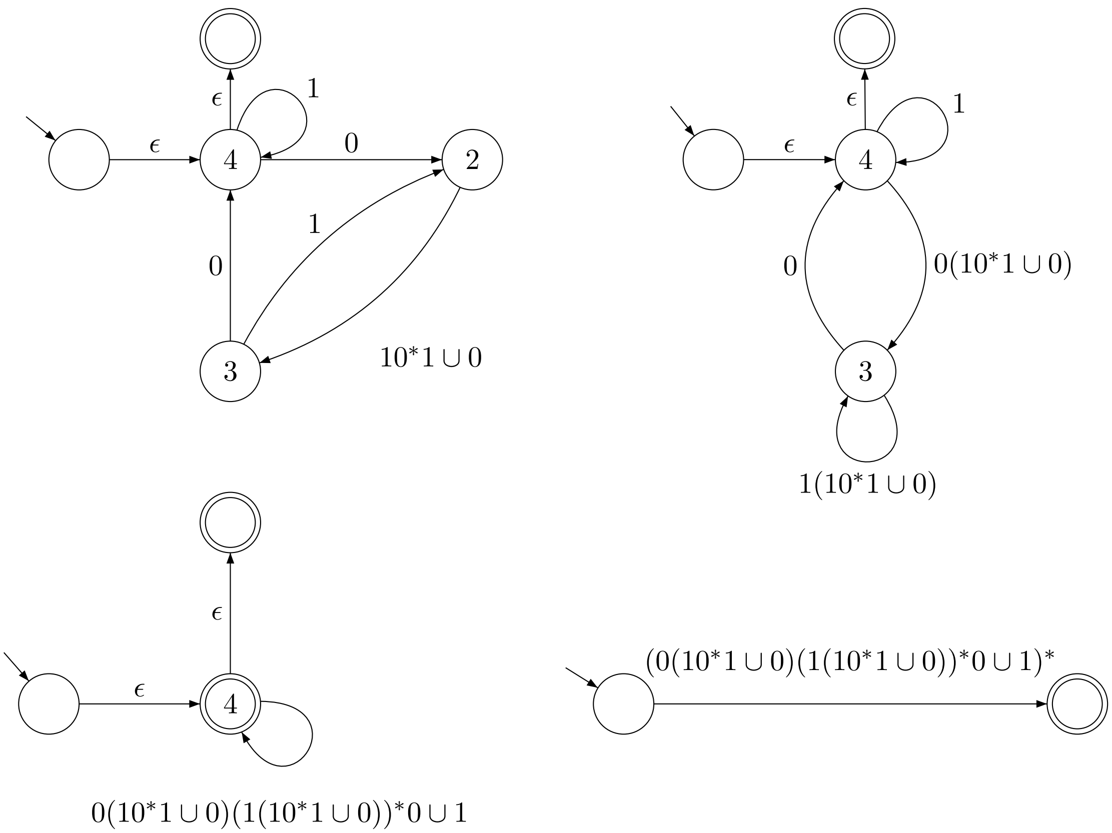
   (f) $L_6 = \{w \mid w \in \Sigma^*\}$

2. **Converting a DFA to a Regex**

Convert the following DFA to a regular expression. You must show all work using the procedure described in Section 1.3 of Sipser. You must follow the following order in eliminating the states: 1, 2, 3, 4.



Solution:



ϵ
1
0
4
2
1
0
3
10*1 ∪ 0
ϵ
1
4
0
0(10*1 ∪ 0)
3
1(10*1 ∪ 0)
ϵ
4
0(10*1 ∪ 0)(1(10*1 ∪ 0))*0 ∪ 1
(0(10*1 ∪ 0)(1(10*1 ∪ 0))*0 ∪ 1)*

3. **More Structural Induction**

Let $\Sigma = \{a, b, c\}$, and recursively define $L \subseteq \Sigma^*$ as the smallest set satisfying the following rules:

Rule 1: $ba \in L$
Rule 2: If $w \in L$, then $baw \in L$.
Rule 3: If $w \in L$, and $w = xaby$ for strings $x, y \in \Sigma^*$, then $xcby \in L$.
Rule 4: If $w \in L$, and $w = xcy$ for strings $x, y \in \Sigma^*$, then $xcbcy \in L$.

Note that x and y do not have to be in L in rules 3 and 4. Let $n_t(w)$ be the number of instances of the character t in w.

(a) Prove that for every $w \in L$, $n_b(w) = n_c(w) + n_a(w)$.

(b) Prove that for every $w \in L$, $n_b(w) > n_c(w)$.

(c) (5 point bonus) Show that L is regular by exhibiting a regular expression for it or an NFA which recognizes it.

In HBS this week, we showed that for every $w \in L$, the last character of w is an a. Feel free to use this fact in solving this problem.

Solution

(a) Base case: $w = ba \in L$. We have that $n_b(w) = 1 = 0 + 1 = n_c(w) + n_a(w)$.

Induction step: Assume that $w \in L$ satisfies $n_b(w) = n_c(w) + n_a(w)$. If $w' = baw$, then $w' \in L$, and $n_b(w') = n_b(w) + 1$. Also, $n_c(w') + n_a(w') = n_c(w) + n_a(w) + 1 = n_b(w) + 1$, as desired.

If $w = xaby$, and $w' = xcby$, then $w' \in L$, and $n_b(w') = n_b(w)$. Also, $n_a(w') + n_c(w') = (n_a(w) - 1) + (n_c(w) + 1) = n_a(w) + n_c(w) = n_b(w) = n_b(w')$, as desired.

If $w = xcy$, and $w' = xcbcy$, then $w' \in L$, and $n_b(w') = n_b(w) + 1$. Also, $n_a(w') + n_c(w') = n_a(w) + n_c(w) + 1 = n_w(b) + 1$, as desired. Thus by induction the identity is satisfied by all strings in L.

(b) By the HBS problem, for each $w \in L$, $0 < n_a(w)$. Subtracting this inequality from the equation $n_b(w) = n_a(w) + n_c(w)$ (and remembering to switch the direction of the inequality since we are subtracting), we get that $n_b(w) > n_a(w) + n_c(w) - n_a(w) = n_c(w)$, as desired.

(c) $b(ab \cup cb)^*a$ is a regular expression equivalent to the above definition.

4. **Pumping Lemma**

Let $L = \{w \in 0^* \mid |w| \neq k^3 \text{ for all } k \in \mathbb{N}\}$. Use the pumping lemma to show that L is not regular.

Solution: First note that L is regular if and only if the complement of L, $\bar{L}$, is regular, so it suffices to prove that $\bar{L}$ is not regular. Note that $\bar{L} = \{w \in 0^* \mid |w| = k^3 \text{ for some } k \in \mathbb{N}\}$.

Assume towards a contradiction that $\bar{L}$ is regular. Then $\bar{L}$ satisfies the pumping lemma. Thus there is a $p \in \mathbb{N}$ such that for any $s \in \bar{L}$ of length at least p, we can divide s into pieces $s = xyz$, where $|xy| \leq p$, $|y| \geq 1$, and for each $i \geq 0$, $xy^iz \in \bar{L}$.

First consider the case $p > 1$. Let $s = 0^{p^3}$, and let $s = xyz$ as above. Then $|xy^0z| = p^3 - |y|$, and $1 \leq |y| \leq p$. The largest perfect cube strictly less than p^3 is $(p-1)^3$. But $(p-1)^3 < p^2(p-1) = p^3 - p \leq |xy^0z|$. Thus $|xy^0z|$ is not a perfect cube, which is a contradiction, since we then have that $xy^0z \notin \bar{L}$.

If $p = 1$, then applying the pumping lemma to the string 0, we see that $\bar{L} = 0^*$, which is clearly false. Thus $\bar{L}$ is not regular, and hence L is not regular.

5. **Modifying automata**

Suppose $\Sigma = \{a, b, c, d\}$. Let $f : \Sigma^* \to \Sigma^*$ which deletes a's from strings. That is, formally,

(a) $f(a) = \epsilon$

(b) $f(t) = t$ for any character $t \neq a$

(c) For any string $w = c_1 c_2 \ldots c_n$, $f(w) = f(c_1)f(c_2)\ldots f(c_n)$

So, for example, $f(badccb) = bdccb$. If L is a language, then $f(L) = \{f(w) \mid w \in L\}$.

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA accepting some language L. Explain how to construct a DFA N recognizing $f(L)$, modeling your construction on the construction used in class/Sipser to convert an NFA to an equivalent DFA.

Solution 1: Create an NFA M' by replacing every a transition in M with an ϵ-transition. That is, $M' = (Q, \{b, c, d\}, \delta', q_0, F)$ The transition function δ' is then defined by

$$\delta'(q, t) = \{\delta(q, t)\} \text{ if } t \neq a$$
$$\delta'(q, \epsilon) = \{\delta(q, a)\}$$

Notice that N' must be an NFA because it uses ϵ-transitions, so the output of δ' must be a **set of states** not a single state.

Now, use the product construction to convert this NFA M' to a DFA N.

Solution 2: Define $A(S)$, the a-closure of a set of states S, to be the set of states reacheable from any state in S using a sequence of zero or more a transitions.

Now define N to be $N = (\mathbb{P}(Q), \{b, c, d\}, \delta', A(\{q_0\}), F')$ The transition function δ' and set of final states F' are then defined by

$$\delta'(q, t) = A(\{\delta(q, t)\})$$

$$F' = \{q \in \mathbb{P}(Q) \mid q \cap F \neq \emptyset\}$$

6. **Warm-up for Minimization (5 point bonus)**

Find a language L and two NFAs M and N recognizing L, such that

- M and N are not the same
- M and N both have 4 states
- Neither M nor N uses any ϵ-transitions.
- L cannot be recognized by an NFA with fewer than 4 states

Explain briefly why you believe that recognizing L requires at least 4 states.

Solution: Here are two NFAs satisfying the conditions, which both recognize the language $L = (a+b)^*aab$. This language requires four states because it must distinguish four cases for how much of the string "aab" it has seen so far. That is, (1) none of this substring, (2) first a, (3) two a's in a row, (4) whole substring aab.

In class, we saw that each language has a unique DFA with the minimum number of states. This example shows that this isn't true for NFAs: a language can have more than one NFA with the same minimum number of states.