Problem Set 4

Spring 10

Due: Thursday Feb 25 in class before the lecture.

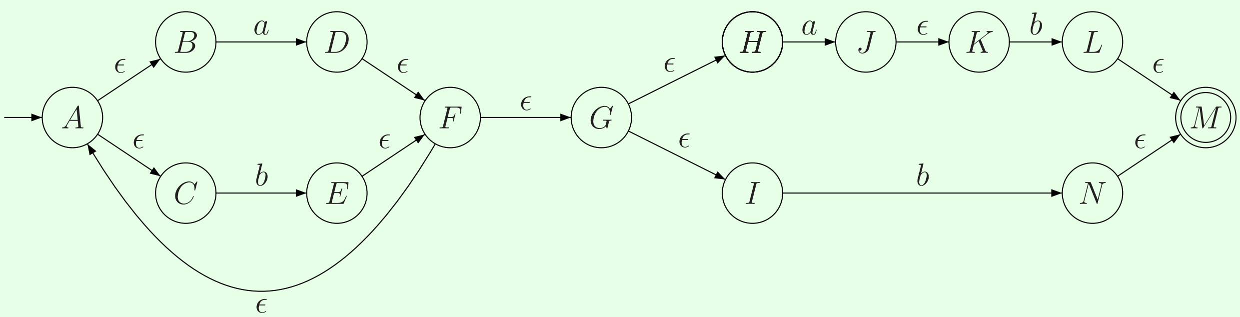
Please <u>follow</u> the homework format guidelines posted on the class web page:

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

1. Regular expressions [**Category**: Comprehension, **Points**: 20]

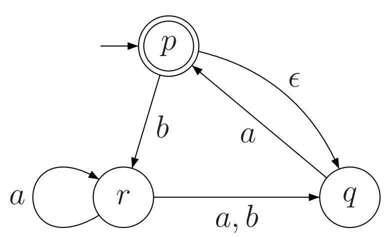
   Construct an NFA with the same language as the regular expression $(a+b)^*(ab+b)$.

Solution:

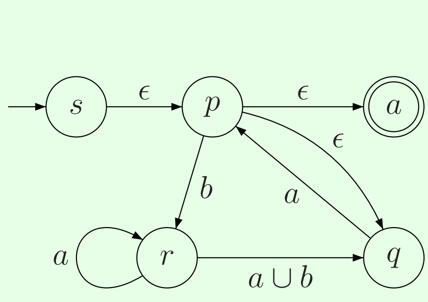


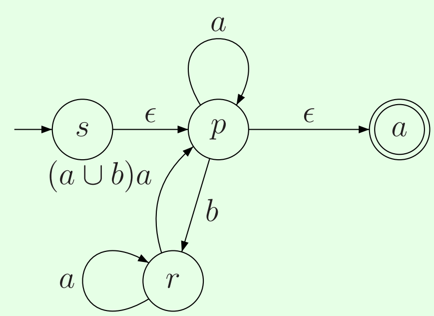
2. Regular expressions [**Category**: Comprehension, **Points**: 20]

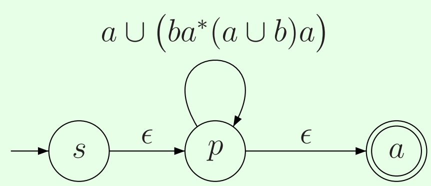
   Use the algorithm learned in class to obtain a regular expression with the same language as the NFA below.

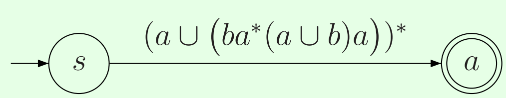


Solution:









3. NFA and non-determinism [**Category**: Construction, **Points**: 20]

   Given $k > 0$, construct an NFA with at most $7k^2$ states for the following language over alphabet $\{0, 1, 2\}$

$$L_k = \{ww' : |w| = |w'| = k, w \neq w'\}.$$

   (Hint: you need to exploit non-determinism to obtain such a small NFA).

Solution:

The idea is to remember the i-th character of string w (non-deterministically) and compare it with i-th character of string w'. NFA accepts if these two are not the same (note that this would give the NFA enough evidence to conclude $w \neq w'$ and also that if $w \neq w'$, then at least one such evidence exists).

We construct the NFA $N = (Q, \Sigma, \delta, (1), F)$ as follows:

$\Sigma = \{1, 2, 3\}$

$Q = \{(i) : 1 \leq i \leq k\} \cup \{(i, a, j) : a \in \Sigma, 1 \leq i \leq j \leq k\} \cup \{(i, a, k, j) : a \in \Sigma, 1 \leq i, j \leq k\}$

$F = \{(i, a, k, k) : 1 \leq i \leq k, a \in \Sigma\}$

transition below: non-deterministically either just counting one symbol of w, or counting and remembering it

$\forall a \in \Sigma, i < k, \quad \delta((i), a) = \{(i, a, i), (i + 1)\}$

$\forall a \in \Sigma, \quad \delta((k), a) = \{(k, a, k)\}$

$\forall a, b \in \Sigma, j < k, \quad \delta((i, a, j), b) = \{(i, a, j + 1)\}$

$\forall a, b \in \Sigma, \quad \delta((i, a, k), b) = \{(i, a, k, 1)\}$ passes the boundary of w and w'

$\forall a, b \in \Sigma, j < k, j \neq i - 1 \quad \delta((i, a, k, j), b) = \{(i, a, k, j + 1)\}$ counting w', looking for i-th character

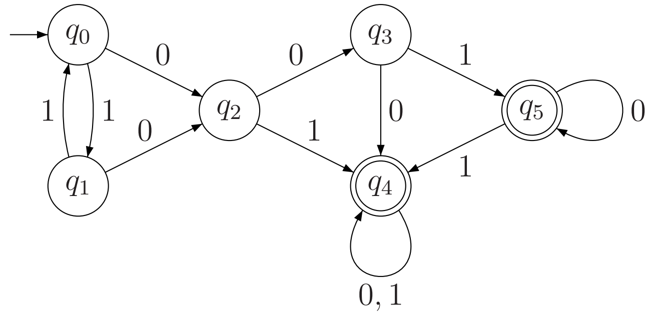
$\forall a \in \Sigma, b \in \Sigma - \{a\}, \ \delta((i, a, k, i - 1), b) = \{(i, a, k, i)\}$ checks for non-equality of i-th character of w and w'

4. Suffix languages [**Category**: Comprehension, **Points**: 20]

(a) Give the suffix languages for the following DFA.

(b) Merge states with the same suffix language to get a smaller DFA. Draw the resulting DFA.

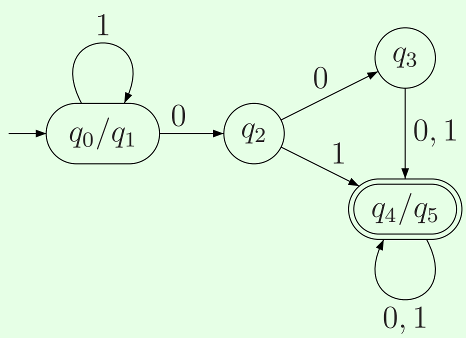
(c) Prove that all your suffix languages are different: for each pair of suffix languages give a sample string that belong to one languages and does not belong to the other one.



Solution:

(a) Give the suffix languages for the following DFA.
$[\![L/\epsilon]\!] = L_0 = L_1 = 1^*0(00 + 01 + 1)(0 + 1)^*$
$[\![L/0]\!] = L_2 = (00 + 01 + 1)(0 + 1)^*$
$[\![L/00]\!] = L_3 = (0 + 1)(0 + 1)^*$
$[\![L/01]\!] = L_4 = L_5 = (0 + 1)^*$

(b) Merge states with the same suffix language to get a smaller DFA. Draw the resulting DFA.



(c) Prove that all of your suffix languages are indeed different: for each pair of suffix languages give a sample string that belong to one languages and does not belong to the other one.
$1 \notin [\![L/\epsilon]\!]$ but $1 \in [\![L/0]\!]$
$1 \notin [\![L/\epsilon]\!]$ but $1 \in [\![L/00]\!]$
$\epsilon \notin [\![L/\epsilon]\!]$ but $\epsilon \in [\![L/01]\!]$
$0 \notin [\![L/0]\!]$ but $0 \in [\![L/00]\!]$
$\epsilon \notin [\![L/0]\!]$ but $\epsilon \in [\![L/01]\!]$
$\epsilon \notin [\![L/00]\!]$ but $\epsilon \in [\![L/01]\!]$

5. Non-regularity [**Category**: Proof., **Points**: 20]

Prove that the following language is non-regular.
$L = \{a^{4n}b^{3n} | n >= 0\}, \Sigma = \{a, b\}$

Solution:

We will use MNT to prove this claim.
Let $S = \{a^{4n} | n >= 0\}$, obviously S is an infinite set. Let $x, y \in S$ and $x \neq y$, then $x = a^{4i}$ and $y = a^{4j}$, $i \neq j$. We next choose our witness z to be $z = b^{3i}$, then $xz = a^{4i}b^{3i} \in L$ and $yz = a^{4j}b^{3i} \notin L$, since $i \neq j$. Hence, by MNT, L is not regular.

Alternative solution using pumping lemma.
Assume p to be the pumping length for L, we pick w to be $w = a^{4p}b^{3p}$. Consider all possible partitioning of w into x, y and z, such that $w = xyz$. From the fact that $|xy| <= p$ we can conclude that xy consists solely of as. Let $k = 0$, then $xy^kz = xz$. Since $|y| >= 1$, $xz = a^tb^{3p} \notin L$, where $t < 4p$. $xy \notin L$ for all p and all possible partitionings, hence, by PL, L is not regular.

6. Concatenation [**Category**: EXTRA CREDIT, **Points**: 20]

In this question, we want to show that the smallest DFA accepting the concatenation of two languages L_1 and L_2 may be exponentially larger than the sizes of the DFAs for L_1 and L_2. Recall that when doing closure under concatenation, the construction in class was done for NFAs; we can do the same construction on DFAs, but the resulting automaton will be an NFA, and converting it to a DFA will give exponentially many states. We want to show that this exponential blow-up is unavoidable.

To this end, let us fix $\Sigma = \{a, b\}$. Show that for any $k \in \mathbb{N}$, there exist two DFAs $A_1 = (Q_1, \Sigma, \delta_1, q_1^s, F_1)$ and $A_2 = (Q_2, \Sigma, \delta_2, q_2^s, F_2)$ with $O(k)$ states each, such that any DFA accepting $L(A_1)L(A_2)$ has at least $O(2^k)$ states. That is, any DFA accepting the concatenation of the languages of A_1 and A_2 will require states exponential in k.