INF2080
Oblig 1

Rune Hovde, runehovd

February 4, 2018

Problem 1: Regular languages

Let A and B be regular languages defined by DFAs $\mathcal{A}$ and $\mathcal{B}$. Let $n_{\mathcal{A}}$ and $n_{\mathcal{B}}$ be the number of states in $\mathcal{A}$ and $\mathcal{B}$, respectively.

Problem 1a

What are the worst-case (highest) number of states in **DFAs** for the languages $A \cap B$ and A^*?
$A \cap B : n_A * n_B$
$A^* : n_A$

Problem 1b

What are the worst-case (higehst) number of states in **NFAs** for the languages $A \cap B$, AB and A^*?
$A \cap B : n_A * n_B$
$AB : n_A + n_B$
$A^* : n_A + 1$

Problem 1c

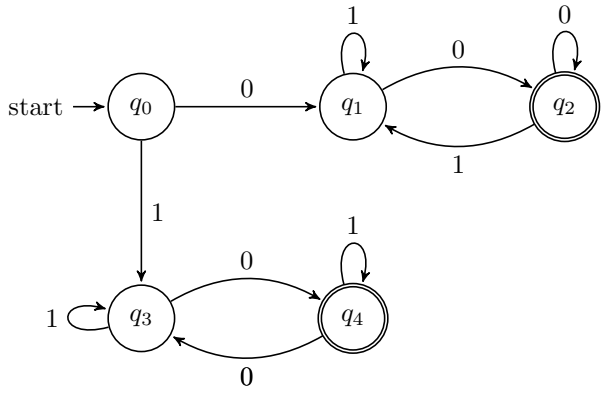
Create a regular expression defining the same language as the NFA

(aa+)*b(cc+)*

Problem 1d

Create a DFA for the language

$\{w \mid w$ contains equally many occurences of the substrings 01 and 10$\}$.



Problem 2: all-NFAs

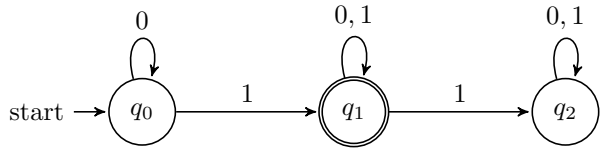
An all-NFA is defined in Sipser, problem 1.43 as a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ that accepts $x \in \Sigma^*$ if *every* possible state that M could reach after reading input x is in F (as opposed to *at least one*).

If any brach in an all-NFA computation reaches an inplicit or explicit sink state, the input is not accepted.

Show how an all-NFA can be converted to an equivalent DFA.

Hint: Adjust the conversion from NFA to DFA shown in the lectures.

For this problem I will convert two NFA's, one all-NFA and one normal NFA. They both have the same states and transitions, but when converting to a DFA, the accepted states are different.
Here is the NFA:



The NFA:

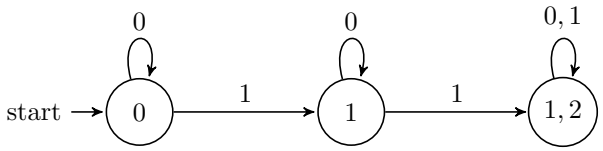
$L = \{w \mid w$ contains at least one 1 and an arbitrary number of 0s$\}$.

The all-NFA:

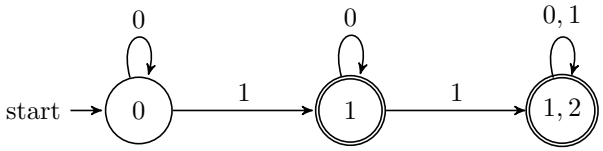
$$A = \{w \mid w \text{ contains exactly one 1 and an arbitrary number of 0s}\}.$$

The states in the DFA is the set of states you can go to from the NFA-state(s).
The final states of a DFA converted from the NFA is the states that contain at least one final state from the NFA.
The final states of a DFA converted from the all-NFA is the states that contain only final-states from the all-NFA.

When converting the state diagram over to a DFA you get the following diagram:



The difference of converting from an all-NFA to a DFA and an NFA to a DFA is when you are choosing the accepting states. As stated above, when converting from an NFA, the DFA-state only has to have one of the accepting states in the state-set, whereas for it to be an all-NFA all of the states in the state-set in the DFA has to be an accepted state.
The DFA from the NFA will look like this:



Whereas the all-NFA conversion will look like this: