Homework 2

CSC 445-01: Theory of Computation

Matthew Mabrey, Luke Kurlandski

March 8, 2021

1.12

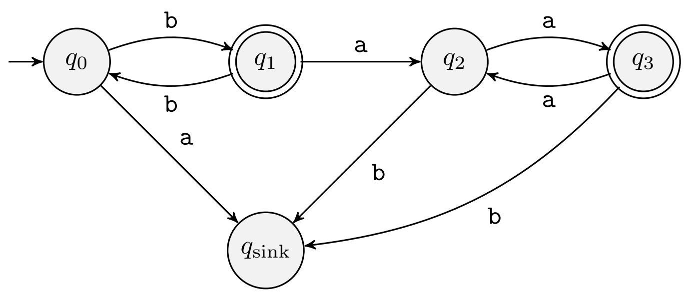
We describe D more simply

$$D = \{w | w \text{ is a word with an odd number of } b\text{'s followed by an even number of } a\text{'s}\}$$

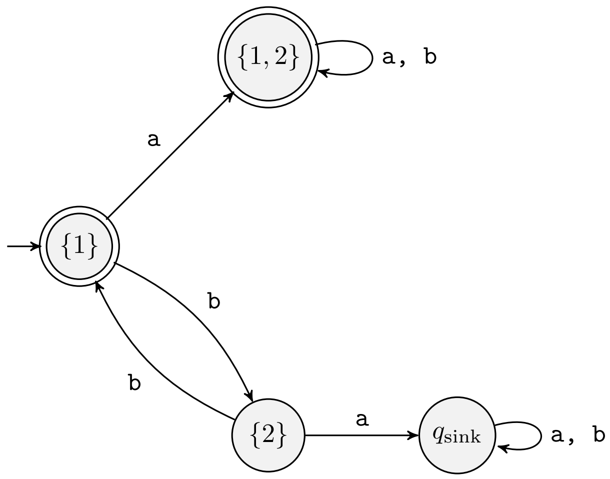
Which lets us realize that the following regular expression describes D

$$R = b(bb)^*(aa)^*$$

And construct a DFA that recognizes D



1.16 a



1.20 g

For the regular expression $R = (\epsilon \cup a)b$

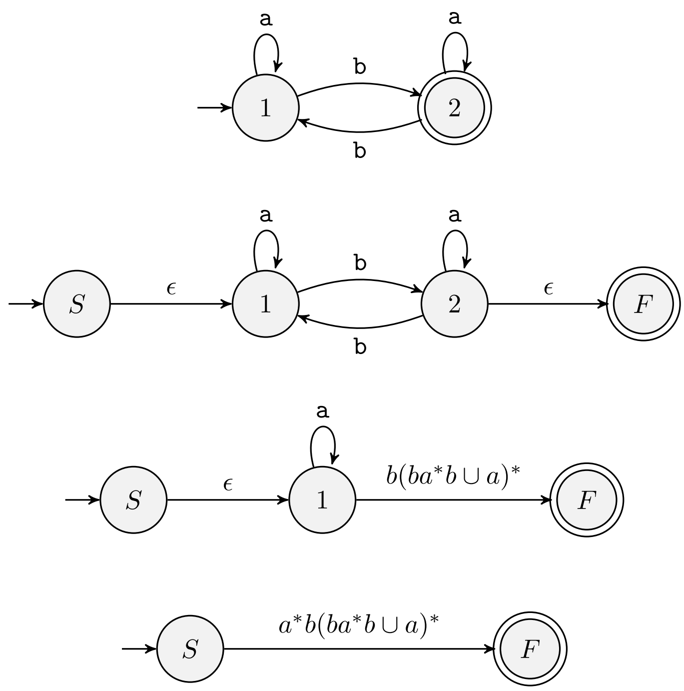
Members include

- b
- ab

Non members include

- a
- aba

1.21 a



1.29 b

Assume that $A = \{www \mid w \in \{a,b\}^*\}$ is regular and let p be the constant for which the pumping lemma holds for A.

Let $w = a^n b^n = a^p b^p$ given the constant p and then let $w' = a^p b^p a^p b^p a^p b^p$ where there are 3 identical copies of the word w in a row.

Then w' is divided into $w' = xyz$ such that $|xy| \leq p$, $|y| \geq 1$, and $(\forall i \geq 0)[xy^i z \in A]$. By these definitions, y can only be composed of the symbol a since the first p symbols of w' is composed of a's and $|xy|$ is at most p symbols long.

Then for any $i \neq 1$, we would have $a^q b^p a^p b^p a^p b^p$ where $q \neq p$ and therefore the word w' cannot be decomposed as three identical substrings www. This is a contradiction and thus the language A is not regular.

1.40 b

Suppose the language A is regular. Then we can construct an NFA N that recognizes A because all regular languages have an equivalence with some nondeterministic finite automata. After that, we can apply a simple modification to N and create a new NFA, N', to have it recognize NoExtend(A). Since NoExtend(A) is recognized by this new NFA, N', it is regular. Therefore, the class of regular languages is closed under the NoExtend() operation.

Given the NFA $N = (Q, \Sigma, \delta, q_0, F)$ that recognizes A, we construct $N' = (Q', \Sigma', \delta', q_0', F')$ that recognizes NoExtend(A) where

- $Q' = Q \cup \{q_f\}$: we add one additional state to Q and make it the one and only final state.
- $\Sigma' = \Sigma$: we keep the same alphabet.

- $\delta' = \delta \cup \delta_{\text{new}}$: we keep all of the old transitions but add some new ones. Out of every state, we add a arrow for every letter in Σ that the state did not already have leaving it. This arrow then leads to q_f. This causes the automata to enter the accept state as soon as it reads a character that confirms the string is not a substring of any word accepted by N.
- $q_0' = q_0$: we keep the same start state
- $F' = \{q_f\}$: we make the set of accept states our new state.