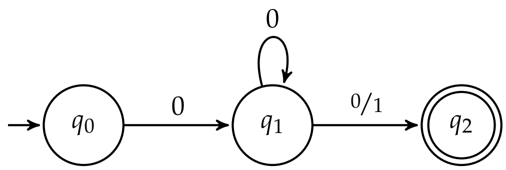
Formal Languages and Computability 3

Ragnar Björn Ingvarsson, rbi3

17. september 2024

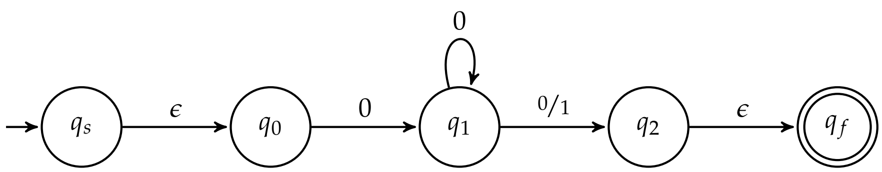
1 Let us consider the language $L(00^*(0 \cup 1))$

a) **Draw the state diagram of an NFA that accepts it.**

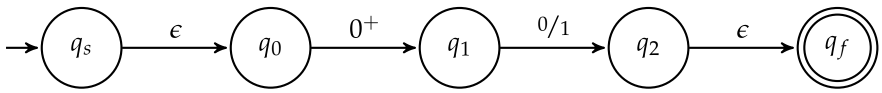


b) **Convert the NFA to a simple RE.**

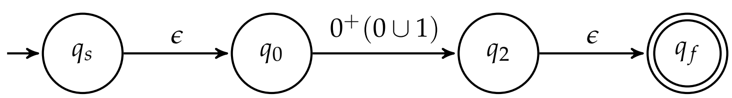
We add states q_s and q_f and add an epsilon transition from q_s to the start state and an epsilon transition from each finish state to q_f, changing each finish state to a normal state and making q_f the only finish state, and get



And we simplify, eliminating firstly the loop on q_1:



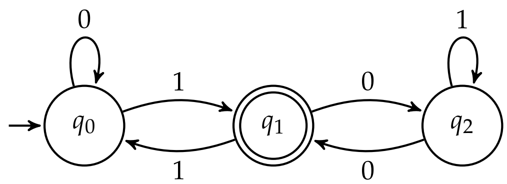
And then eliminating state q_1:



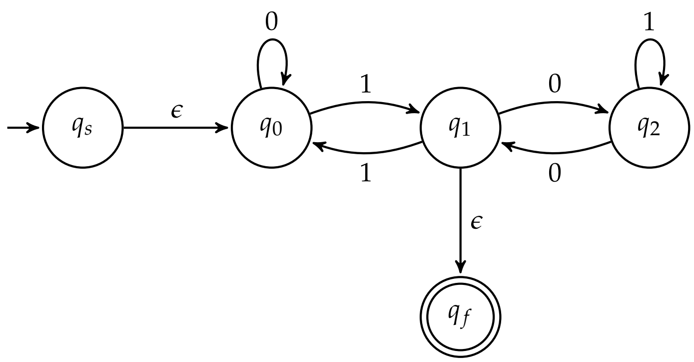
And now we have simplified the NFA and we get the regular expression $0^+(0 \cup 1)$.

2 Suppose the input alphabet is binary numbers, $\Sigma = \{0, 1\}$, and language A consists of binary numbers x such that $x \bmod 3 = 1$. Determine a regular expression for A.

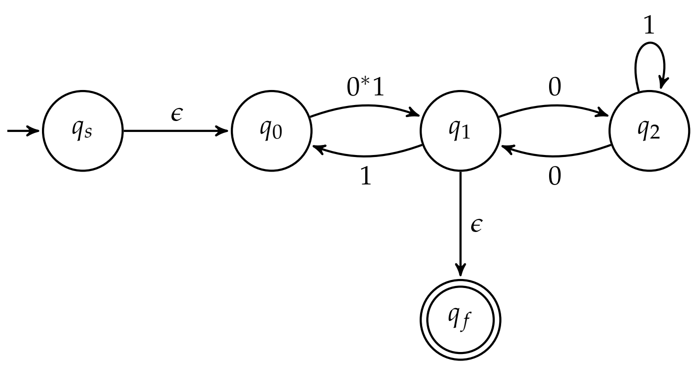
We can start from making a DFA describing the language:



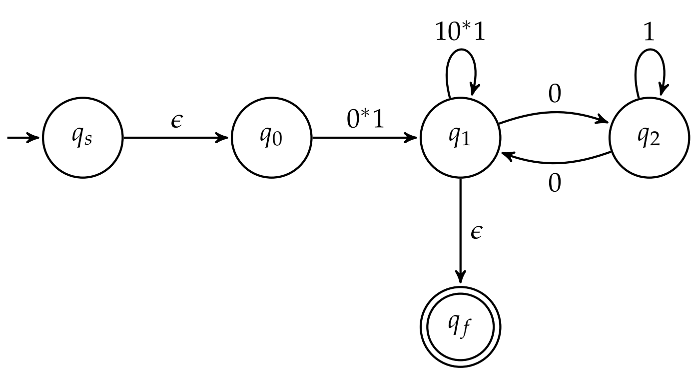
And then start converting to a regular expression:



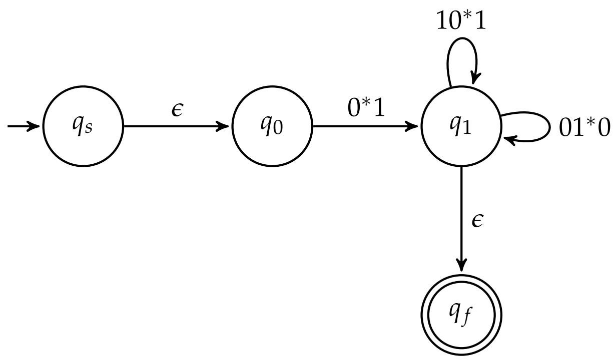
Where from we can simplify the path between q_0 and q_1:



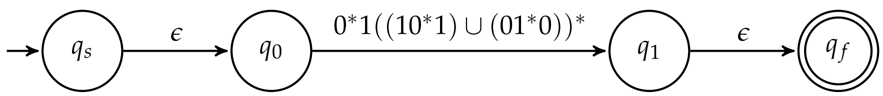
And then we can remove the path from q_1 to q_0:



Finally we do the same for q_2 and get:



Then combine everything into a single path:



So the regular expression is $0^*1((10^*1) \cup (01^*0))^*$.

3 Let $\Sigma = a, ..., t$ denote the numbers $0, ..., 19$ in that order. Let A denote the language that consists of all strings whose numerical value is less than 42. Determine a regular expression for A.

We notice here that the string can start with an undetermined amount of a's (0's), and after that we have three cases:

- Any letter in the alphabet with nothing to follow, $0 - 19$.
- b, followed then by any letter in the alphabet, $20 - 39$.
- c, followed exclusively by either one a or one b, $40 - 41$.

We can then write out the regular expression:

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

Here we have also excluded the possibility of the empty string since it does not have a numerical value.

4 Let us continue with the Mayan numbers. Let language A consist of numbers x such that $x \bmod 3 = 1$.

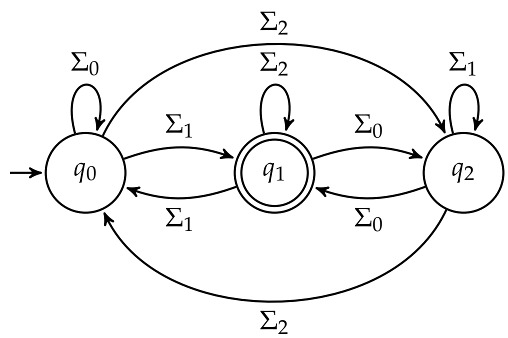
a) **Determine a DFA that recognizes A.**

Let us divide the alphabet Σ into three subsets,

- $\Sigma_0 = \{x \in \Sigma | x \bmod 3 = 0\}$
- $\Sigma_1 = \{x \in \Sigma | x \bmod 3 = 1\}$
- $\Sigma_2 = \{x \in \Sigma | x \bmod 3 = 2\}$

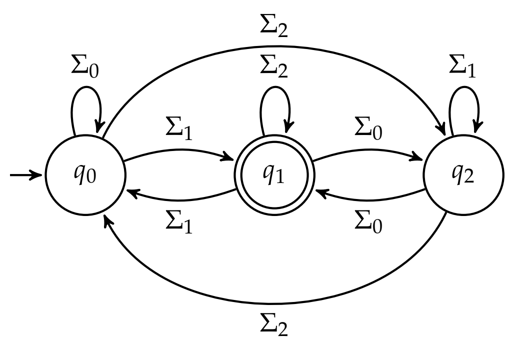
And then notice that since we are multiplying by 20 with each letter added, if we research what happens only if we add a, we get that from a remainder of 0, we get a remainder of $0 * 20 \bmod 3 = 0$, from 1 we get $1 * 20 \bmod 3 = 2$ and from 2 we get $2 * 20 \bmod 3 = 1$.

That leads to this DFA:

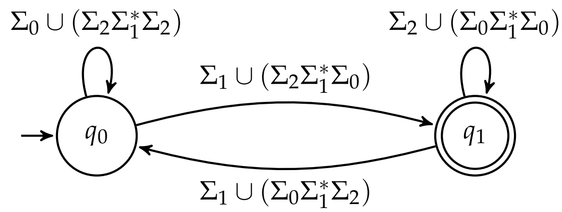


b) **Give a regular expression for A.**

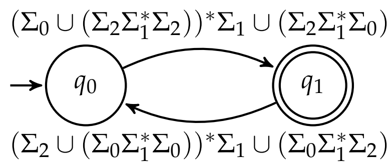
We use the DFA in a) and convert it to a regular expression, and so we begin with this:



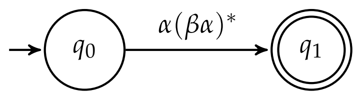
From where we want to eliminate q_2 by using the state elimination method:



Now we remove the loops:



And now, for simplification, we can say that the transition from q_0 to q_1 is α and from q_1 to q_0 is β, and we get



And that is the regular expression, $\alpha(\beta\alpha)^*$, or in its entirety:

$$(\Sigma_0 \cup (\Sigma_2\Sigma_1^*\Sigma_2))^*\Sigma_1 \cup (\Sigma_2\Sigma_1^*\Sigma_0)((\Sigma_2 \cup (\Sigma_0\Sigma_1^*\Sigma_0))^*\Sigma_1 \cup (\Sigma_0\Sigma_1^*\Sigma_2)(\Sigma_0 \cup (\Sigma_2\Sigma_1^*\Sigma_2))^*\Sigma_1 \cup (\Sigma_2\Sigma_1^*\Sigma_0))^*$$