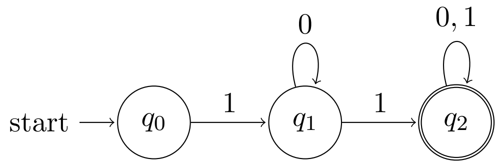
Bram Schuijff
CSCI 387, §S01/02
Homework 2
February 13th, 2025 by 10:30pm

1. (a) This is proven in the book under Example 1.74 starting on page pp. 80.

   (b) This language is regular because there is no reason that k has to be greater than 1. We can demonstrate that L is regular by constructing an NFA:

   (c) Suppose for the sake of contradiction that L is regular. Then, by the pumping lemma, there exists a pumping length p. Let s be the string $1^p0^p1^p$. This is guaranteed to have length at least p and s is a member of L. Therefore, the pumping lemma says that s can be split into 3 substrings $s = xyz$ where for any $i \geqslant 0$ the string $xy^iz \in L$. Then, we have that (1) $|xy| \leqslant p$ and (2) $|y| > 0$. By these conditions, y either has the form 1^j0^k or 1^m. The pumping lemma says that $xy^iz \in S$ for all $i \geqslant 0$. If we let $i = 0$, then s takes the form $1^{p-j}0^{p-k}1^p$ or $1^{p-m}0^p1^p$. In any case, the second substring of 1s now contains more 1s than the first and $s \notin L$, creating a contradiction. Thus, L is not regular.

2. (a) This language is not regular because it is the concatenation of a regular language with an irregular language. Define L' and L'' as follows:

   $$L' = \{0^i \mid i \geqslant 0\}, \qquad L'' = \{1^j2^k \mid j, k \geqslant 0, j = k\}$$

   We see that L'' is irregular because it is almost identical to the language $I = \{0^j1^k \mid j, k \geqslant 0, j = k\}$ which we have demonstrated is irregular in the alphabet $\Sigma = \{0, 1\}$. Adding additional symbols cannot turn an irregular language into a regular one, and therefore L'' is irregular. We can represent L by the following piecewise definition:

   $$L = \begin{cases} \{0^i1^j2^k \mid j, k \geqslant 0, j = k\} & \text{if } i = 0 \\ \{0^i1^j2^k \mid j, k \geqslant 0\} & \text{otherwise} \end{cases}$$

   The first case is just 0 concatenated with L'' which we demonstrated is irregular. Therefore, L is irregular.

   (b) L does not look irregular as far as the pumping lemma is concerned because if we have a pumping length p and i, j, k such that $i + j + k \geqslant p$ and $s = 0^i1^j2^k$. Suppose $i \neq 1$. Then, y may be any substring of s so long as it does not contain $i - 1$ 0s and it consists of all one number. s will still be valid because $i \neq 1$, meaning it does not matter how many 1s or 2s there are. If $i = 1$, then we let $y = 0$ and all powers of a will still result in $s \in L$.

   (c) Quite frankly, I have no idea.

3. (a) We can create the following CFG:

$$\begin{aligned} S &\to A_1A_1A_1 \\ A_1 &\to 1A \mid A1 \\ A &\to BAB \mid \epsilon \\ B &\to 0 \mid 1 \mid \epsilon \end{aligned}$$

(b) We can create the following CFG:

$$S \to 0 \mid 0S0 \mid 1S0 \mid 0S1 \mid 1S1$$

(c) We first construct the CFG for a similar language: $L' = \{0^n1^m \mid n < m\}$. This looks like:

$$\begin{aligned} S' &\to A'1 \\ A' &\to 0A'1 \mid A'1 \mid \epsilon \end{aligned}$$

We can create a similar CFG for the language $L'' = \{0^n1^m \mid n > m\}$:

$$\begin{aligned} S'' &\to 0A'' \\ A'' &\to 0A''1 \mid 0A'' \mid \epsilon \end{aligned}$$

In effect, we find that $L = L' \cup L''$. Then, we can simply do

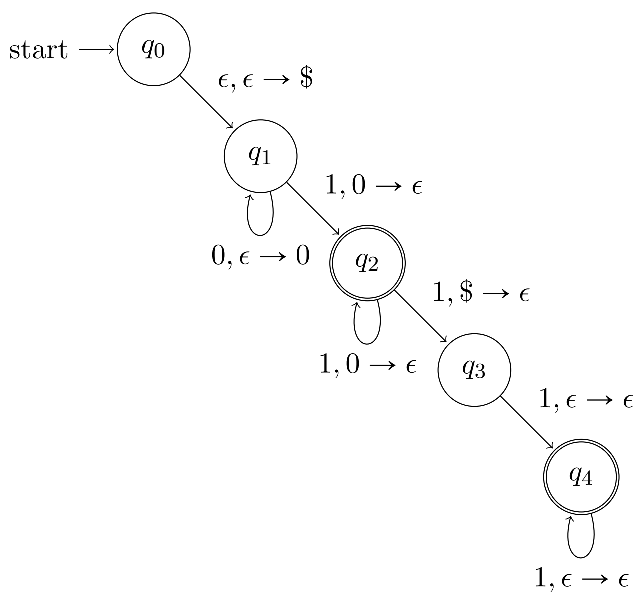
$$S \to S' \mid S''$$

and insert the substitution rules from L' and L''.

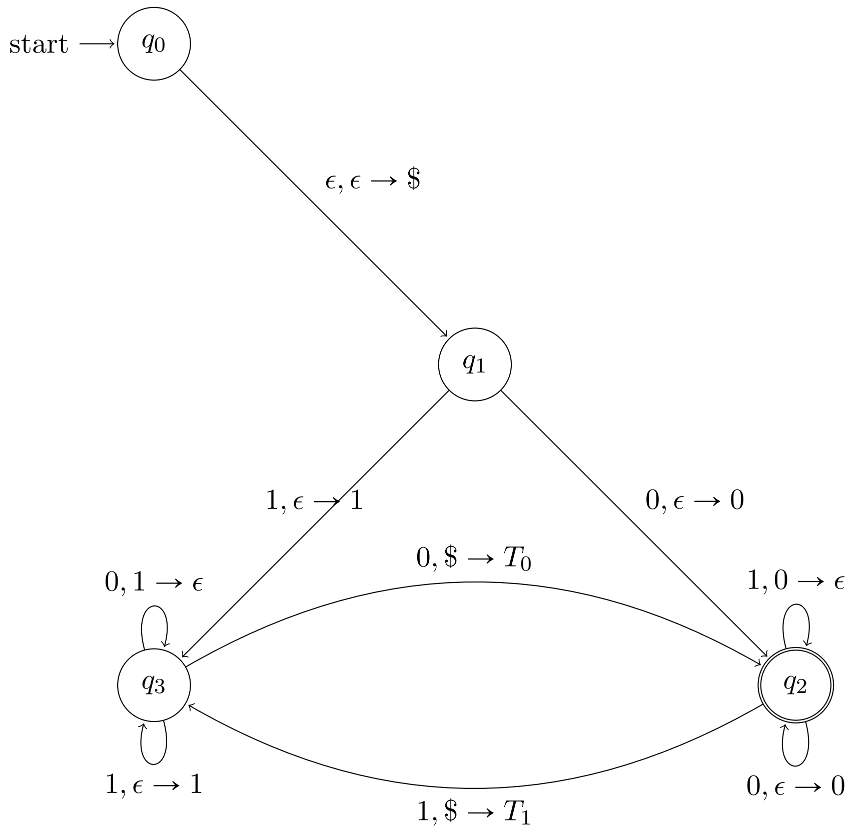
(d) This is similar to concatenating two instances of L'' from (c). Thus, we find that

$$\begin{aligned} S &\to L1R \\ L &\to L1 \mid 0L1 \mid \epsilon \\ R &\to 1R \mid 1R0 \mid \epsilon \end{aligned}$$

4. (a) Using the ideas in Lemma 2.21, we can construct the following PDA:



(b) We can again apply the same principles:



where I use T_i to represent pushing \$ followed by pushing i. This could be achieved by having an intermediary state for each of q_2 and q_3 but it would make the diagram rather cumbersome and so I have used the hopefully-readable shorthand.