Automata and Formal Languages — Homework 6

Due 25.11.2016

Exercise 6.1
Let val : $\{0,1\}^* \to \mathbb{N}$ be such that val(w) is the number represented by w with the "least significant bit first" encoding.

(a) Give a transducer that doubles numbers, i.e. a transducer accepting

$$L_1 = \{[x, y] \in (\{0,1\} \times \{0,1\})^* : \text{val}(y) = 2 \cdot \text{val}(x)\}.$$

(b) Give an algorithm that takes $k \in \mathbb{N}$ as input, and that produces a transducer A_k accepting

$$L_k = \left\{[x, y] \in (\{0,1\} \times \{0,1\})^* : \text{val}(y) = 2^k \cdot \text{val}(x)\right\}.$$

(Hint: use (a) and consider operations seen in class.)

(c) Give a transducer for the addition of two numbers, i.e. a transducer accepting

$$\{[x, y, z] \in (\{0,1\} \times \{0,1\} \times \{0,1\})^* : \text{val}(z) = \text{val}(x) + \text{val}(y)\}.$$

(d) For every $k \in \mathbb{N}_{>0}$, let

$$X_k = \{[x, y] \in (\{0,1\} \times \{0,1\})^* : \text{val}(y) = k \cdot \text{val}(x)\}.$$

Suppose you are given transducers A and B accepting respectively X_a and X_b for some $a, b \in \mathbb{N}_{>0}$. Sketch an algorithm that builds a transducer C accepting X_{a+b}. (Hint: use (b) and (c).)

(e) Let $k \in \mathbb{N}_{>0}$. Using (b) and (d), how can you build a transducer accepting X_k?

(f) Show that the following language has infinitely many residuals, and hence that it is not regular:

$$\left\{[x, y] \in (\{0,1\} \times \{0,1\})^* : \text{val}(y) = \text{val}(x)^2\right\}.$$

Exercise 6.2
Let $\text{LSBF}_k : \{0, 1, \ldots, 2^k - 1\} \to \{0,1\}^k$ be such that $\text{LSBF}_k(n)$ is the "least significant bit first" encoding of size k of n.

(a) For every language $L \subseteq \{0,1\}^*$ of length $k \in \mathbb{N}$, we define $L + 1 = \{\text{LSBF}_k(\text{val}(w) + 1 \bmod 2^k) : w \in L\}$.

Give an algorithm that takes a state q of the master automaton as input, and that computes the state of $L(q) + 1$.

(b) Let $A = (Q, \{0,1\}, \delta, q_0, F)$ be a DFA for some language of length k. Let $X = \{\text{val}(w) : w \in L(A)\}$. In terms of X, describe the set of numbers accepted by $A' = (Q, \{0,1\}, \delta', q_0, F)$ where $\delta'(q, b) = \delta(q, 1 - b)$.

Solution 6.1

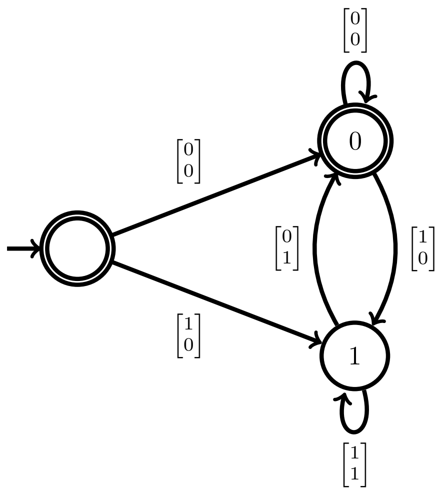
(a) Let $[x_1 x_2 \cdots x_n, y_1 y_2 \cdots y_n] \in (\{0,1\} \times \{0,1\})^n$ where $n > 1$. Multiplying a binary number by two shifts its bits and adds a zero. For example, the word

$$\begin{bmatrix} 10110 \\ 01011 \end{bmatrix}$$

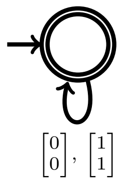
belongs to the language since it encodes $[13, 26]$. Thus, we have $\text{val}(y) = 2 \cdot \text{val}(x)$ if, and only if $y_1 = 0$, $x_n = 0$, and $y_i = x_{i-1}$ for every $1 < i \leq n$. From this observation, we build a transducer that

- makes sure the first bit of y is 0,
- ensures that y is consistent with x by keeping the last bit of x in memory,
- accepts $[x, y]$ if the last bit of x is 0.

Note that $[\varepsilon, \varepsilon]$ and $[0, 0]$ both encode $[0, 0]$. Therefore, they should also be accepted since $2 \cdot 0 = 0$. We obtain the following transducer:



(b) Let A_0 be the following transducer accepting $\{[x, y] \in (\{0,1\} \times \{0,1\})^* : y = x\}$:



Let A_1 be the transducer obtained in (a). For every $k > 1$, we define $A_k = \mathit{Join}(A_{k-1}, A_k)$. A simple inductions show that $L(A_k) = L_k$ for every $k \in \mathbb{N}$.

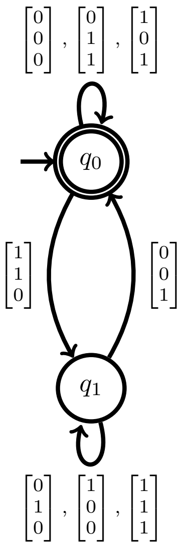
(c) We build a transducer that computes the addition by keeping the current carry bit. Consider some tuple $[x, y, z] \in \{0,1\}^3$ and a carry bit r. Adding x, y and r leads to the bit

$$z = x + y + r \bmod 2. \tag{1}$$

Moreover, it gives a new carry bit r' such that $r' = 1$ if $x + y + r > 1$ and $r' = 0$ otherwise. The folllowing tables identifies the new carry bit r' of the tuples that satisfy (1):

	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\1\end{bmatrix}$	$\begin{bmatrix}0\\1\\0\end{bmatrix}$	$\begin{bmatrix}0\\1\\1\end{bmatrix}$	$\begin{bmatrix}1\\0\\0\end{bmatrix}$	$\begin{bmatrix}1\\0\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\\0\end{bmatrix}$	$\begin{bmatrix}1\\1\\1\end{bmatrix}$
$r = 0$	0	x	x	0	x	0	1	x
$r = 1$	x	0	1	x	1	x	x	1

We deduce our transducer from the above table:



(d) Let $A = (Q_A, \{0,1\}, \delta_A, q_{0A}, F_A)$ and $B = (Q_B, \{0,1\}, \delta_B, q_{0B}, F_B)$. Let $D = (Q_D, \{0,1\}, \delta_D, q_{0D}, F_D)$ be the transducer for addition obtained in (c). We define C as $C = (Q_C, \{0,1\}, \delta_C, q_{0C}, F_C)$ where

- $Q_C = Q_A \times Q_B \times Q_D$,
- $q_{0C} = (q_{0A}, q_{0B}, q_{0D})$,
- $F_C = F_A \times F_B \times F_D$,

and

$$(p, p', p'') \xrightarrow{[a,c]}_C (q, q', q'') \iff \exists b, b' \in \{0,1\} \text{ s.t. } p \xrightarrow{[a,b]}_A q, p' \xrightarrow{[a,b']}_B q' \text{ and } p'' \xrightarrow{[b,b',c]}_D q'' .$$

(e) Let $\ell = \lfloor \log_2(k) \rfloor$. There exist $c_0, c_1, \ldots, c_\ell \in \{0,1\}$ such that $k = c_0 \cdot 2^0 + c_1 \cdot 2^1 + \cdots + c_\ell \cdot 2^\ell$. Let $I = \{0 \leq i \leq \ell : c_i = 1\}$. Note that $k = \sum_{i \in I} 2^i$. Therefore, it suffices to obtain A_i from (b) for each $i \in I$, and to combine them using (d).

(f) For every $n \in \mathbb{N}_{>0}$, let

$$u_n = \begin{bmatrix} 0^n 1 \\ 0^n 0 \end{bmatrix} \text{ and } v_n = \begin{bmatrix} 0^{n-1} 0 \\ 0^{n-1} 1 \end{bmatrix}.$$

Let $i, j \in \mathbb{N}_{>0}$ be such that $i \neq j$. We claim that $L^{u_i} \neq L^{u_j}$. We have

$$u_i v_i = \begin{bmatrix} 0^i 1 0^i \\ 0^{2i} 1 \end{bmatrix} \text{ and } u_j v_i = \begin{bmatrix} 0^j 1 0^i \\ 0^{i+j} 1 \end{bmatrix}.$$

Therefore, $u_i v_i$ encodes $[2^i, 2^{2i}]$, and $u_i v_j$ encodes $[2^j, 2^{i+j}]$. We observe that $u_i v_i$ belongs to the language since $2^{2i} = (2^i)^2$. However, $u_j v_i$ does not belong to the language since $2^{i+j} \neq 2^{2j} = (2^j)^2$. □

Solution 6.2

(a) We have

$$L + 1 = \begin{cases} \emptyset & \text{if } L = \emptyset, \\ \{\varepsilon\} & \text{if } L = \{\varepsilon\}, \\ 1 \cdot L^0 \ \cup \ 0 \cdot (L^1 + 1) & \text{otherwise.} \end{cases}$$

This observation gives rise to an algorithm:

Input: state q of the master automaton.
Output: state q' such that $L(q') = L(q) + 1$.

```
plus-one(q):
    if q ∈ {q_∅, q_ε} then
        return q
    else
        r ← plus-one(q^1)
        return make(r, q^0)
```

(b) A' flips the bits of the numbers accepted by A. Thus, the numbers accepted by A' are $\{2^k - 1 - n : n \in X\}$. In combination with (a), this allows to multiply a set of numbers by -1 in two's complement representation.