Automata and Formal Languages — Exercise Sheet 7

Exercise 7.1
Let val : $\{0,1\}^* \to \mathbb{N}$ be the function that associates to every word $w \in \{0,1\}^*$ the number val(w) represented by w in the *least significant bit first* encoding.

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

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

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

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

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

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

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

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

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

Sketch an algorithm that takes as input transducers A and B, accepting respectively X_a and X_b for some $a, b \in \mathbb{N}_{>0}$, and that produces a transducer C accepting X_{a+b}.

(e) Let $k \in \mathbb{N}_{>0}$. Using (b) and this, 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:

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

Exercise 7.2
Consider transducers whose transitions are labeled by elements of $(\Sigma \cup \{\varepsilon\}) \times (\Sigma^* \cup \{\varepsilon\})$. Intuitively, each transition reads one or zero letter and writes a word of arbitrary length. Such a transducer can be used to perform operations on strings, e.g. upon reading `"singing in the rain"` it could write `Singing In The Rain`.

Sketch such ε-transducers for the following operations, each of which is informally defined by means of three examples. For each example, when the transducer reads the string on the left, it should write the string on the right. You may assume that the alphabet Σ consists of $\{a, b, \ldots, z, A, B, \ldots, Z\}$, a whitespace symbol, and an end-of-line symbol. Moreover, you may assume that every string ends with an end-of-line symbol and contains no other occurrence of the end-of-line symbol.

(a)

Input	Output
`Hello world`	`HHEELLLLOO WWOORRLLDD`
`Winter is Coming`	`WWIINNTTEERR IISS CCOOMMIINNG`
`happy noises`	`HHAAPPPPYY NNOOIISSEESS`

(b) For this exercise, Σ is extended with $\{,, \cdot\}$.

Input	Output
`Ada Lovelace`	`Lovelace, A.`
`Alan Turing`	`Turing, A.`
`Donald Knuth`	`Knuth, D.`

(c) For this exercise, Σ is extended with $\{0, 1, \ldots, 9, (,), +\}$. We want to transform phone-numbers into a normal form, where they are prefixed with a country code.

Input	Output
`004989273452`	`+49 89 273452`
`(00)4989273452`	`+49 89 273452`
`273452`	`+49 89 273452`
`2 7 3 4 5 2`	`+49 89 273452`
`498949`	`+49 89 498949`
`+49 89 498949`	`+49 89 498949`

Solution 7.1

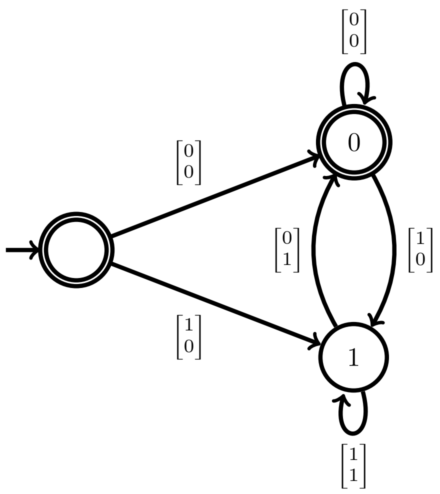
(a) Let $[x_1 x_2 \cdots x_n, y_1 y_2 \cdots y_n] \in (\{0,1\} \times \{0,1\})^n$ where $n \geqslant 2$. Multiplying a binary number by two shifts its bits and adds a zero. For example, the word

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

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

- tests whether the first bit of y is 0,
- tests whether 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.

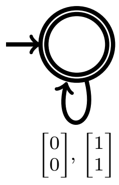
Note that words $[\varepsilon, \varepsilon]$ and $[0, 0]$ both encode the numerical values $[0, 0]$. Therefore, they should also be accepted since $2 \cdot 0 = 0$. We obtain the following transducer:



$\begin{bmatrix}0\\0\end{bmatrix}$
0
$\begin{bmatrix}0\\0\end{bmatrix}$
$\begin{bmatrix}0\\1\end{bmatrix}$
$\begin{bmatrix}1\\0\end{bmatrix}$
$\begin{bmatrix}1\\0\end{bmatrix}$
1
$\begin{bmatrix}1\\1\end{bmatrix}$

★ As seen in class, the initial state can be merged with state 0 as they have the same outgoing transitions.

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



$\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix}$

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

(c) We construct 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 yields a new carry bit r' such that $r' = 1$ if $x + y + r > 1$ and $r' = 0$ otherwise. The folllowing table identifies the new carry bit r' of the tuples that satisfy (**??**):

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

We construct our transducer from the above table:

$$\begin{tikzcd}
{} \arrow[r] & q_0 \arrow[loop above, "{\begin{bmatrix}0\\0\\0\end{bmatrix},\begin{bmatrix}0\\1\\1\end{bmatrix},\begin{bmatrix}1\\0\\1\end{bmatrix}}"] \arrow[d, bend right, "{\begin{bmatrix}1\\1\\0\end{bmatrix}}"'] \\
& q_1 \arrow[u, bend right, "{\begin{bmatrix}0\\0\\1\end{bmatrix}}"'] \arrow[loop below, "{\begin{bmatrix}0\\1\\0\end{bmatrix},\begin{bmatrix}1\\0\\0\end{bmatrix},\begin{bmatrix}1\\1\\1\end{bmatrix}}"]
\end{tikzcd}$$

(d) We construct a transducer C that, intuitively, feeds its input to both A and B, and then feed the respective outputs of A and B to a transducer performing addition. More formally, let $A = (Q_A, \{0,1\}, \delta_A, q_{0A}, F_A)$, $B = (Q_B, \{0,1\}, \delta_B, q_{0B}, F_B)$, and 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

$$\delta_C((p,p',p''),[x,z]) = \{(q,q',q'') : \exists y,y' \in \{0,1\} \text{ s.t. } p \xrightarrow{[x,y]}_A q, p' \xrightarrow{[x,y']}_B q' \text{ and } p'' \xrightarrow{[y,y',z]}_D q''\}.$$

(e) Let $\ell = \lceil \log_2(k) \rceil$. 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 $I = \{0 \leqslant i \leqslant \ell : c_i = 1\}$. Note that $k = \sum_{i \in I} 2^i$. Therefore, we may use transducer A_i from (b) for each $i \in I$, and combine these transducers using (d).

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

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

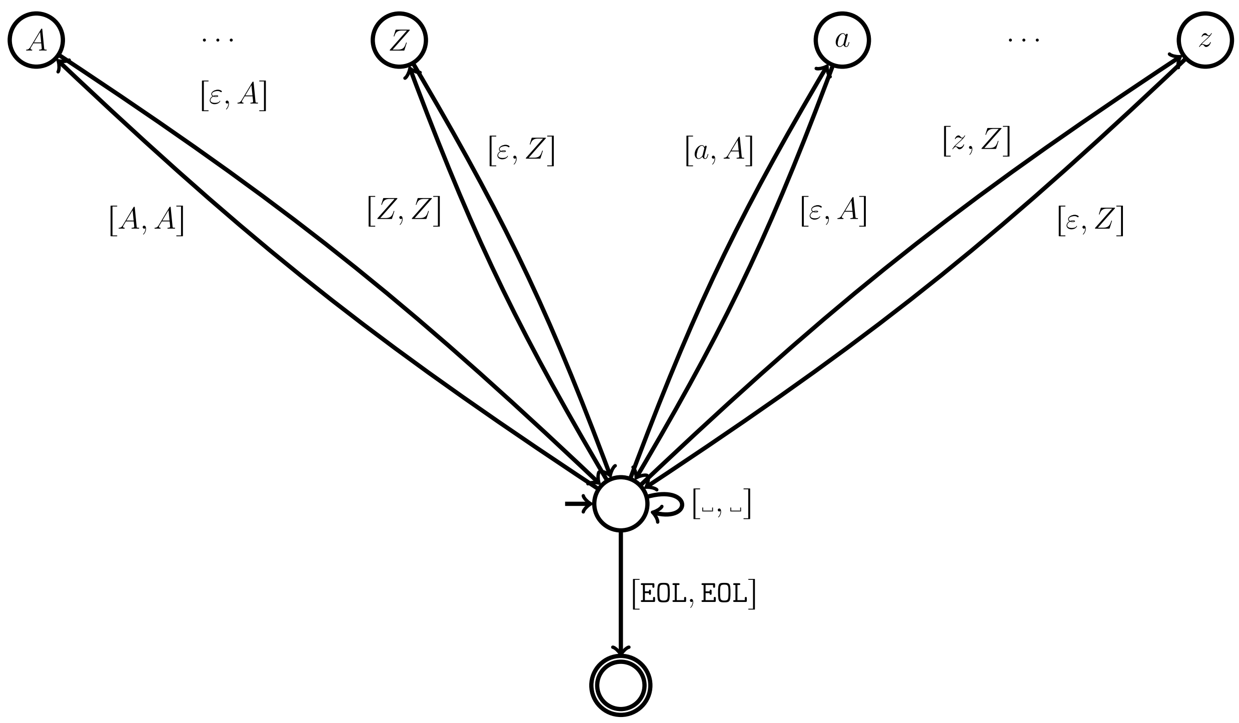
Let $i,j \in \mathbb{N}_{>0}$ be such that $i \neq j$. We claim that $L^{u_i} \neq L^{u_j}$. We have

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

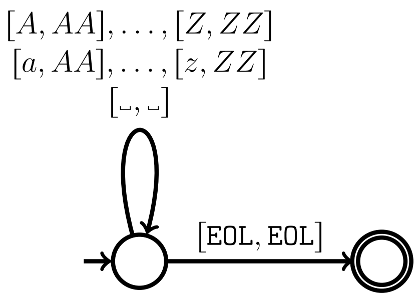
Therefore, $u_i v_i$ encodes $[2^i, 2^{2i}]$, and $u_i v_j$ encodes $[2^j, 2^{i+j}]$. We observe that $u_i v_i$ belongs to the language since $2^{2i} = (2^i)^2$. However, $u_j v_i$ does not belong to the language since $2^{i+j} \neq 2^{2j} = (2^j)^2$. □

Solution 7.2

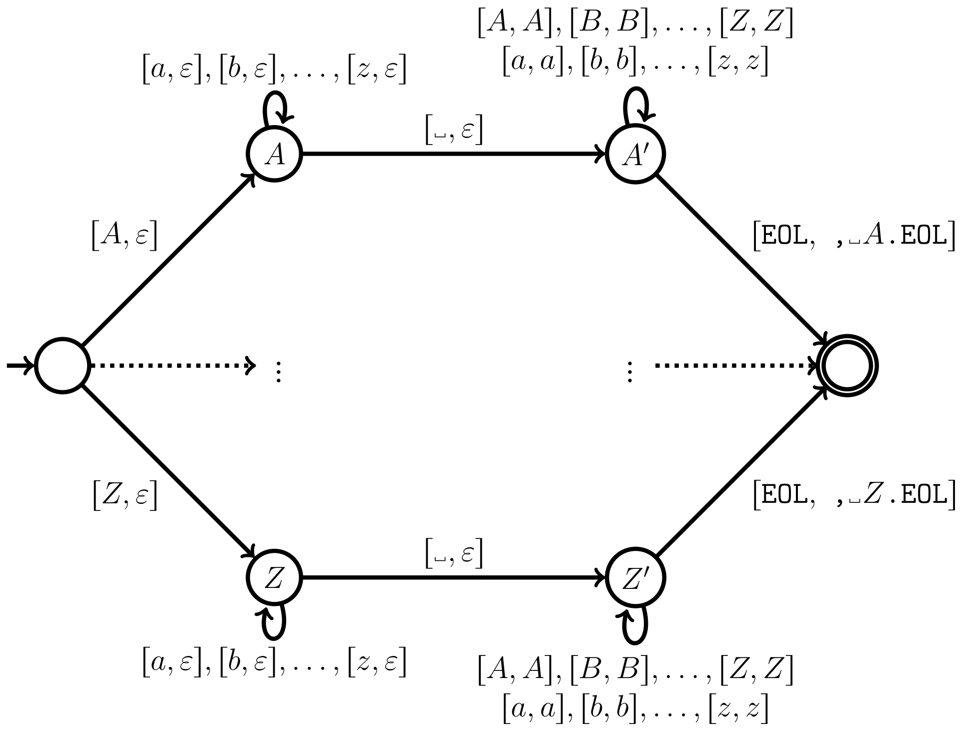
(a)



We can construct a simpler transducer using the fact that we are allowed to write *words*:

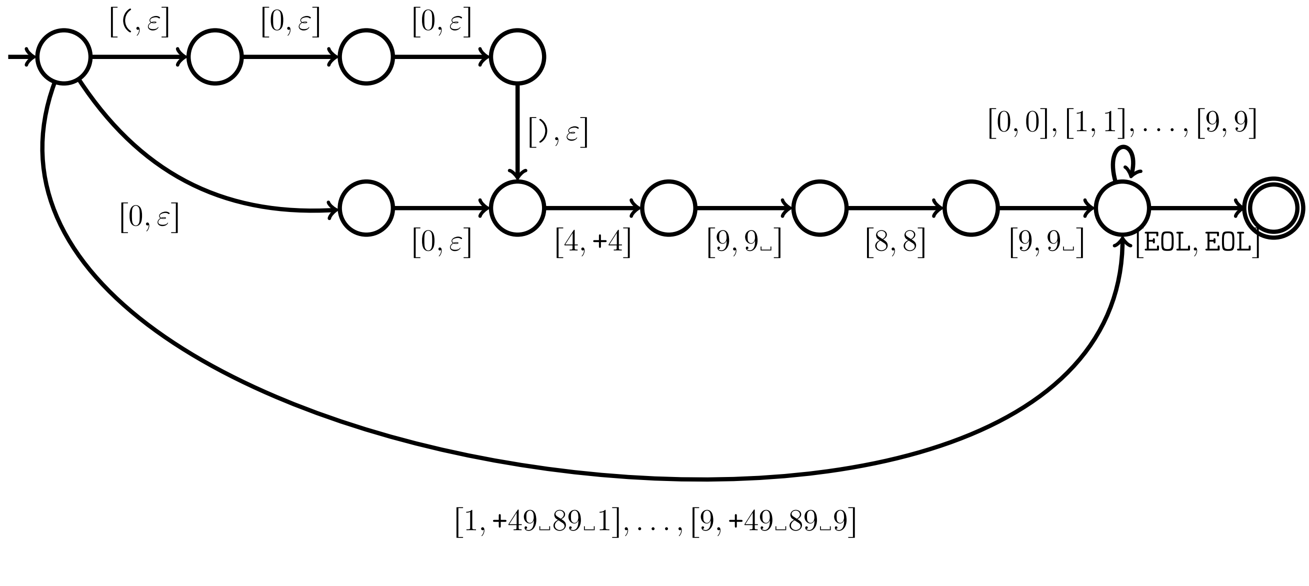

$[A, AA], \ldots, [Z, ZZ]$
$[a, AA], \ldots, [z, ZZ]$
$[␣, ␣]$
$[\texttt{EOL}, \texttt{EOL}]$

(b)



$[A,A],[B,B],\ldots,[Z,Z]$
$[a,a],[b,b],\ldots,[z,z]$
$[a,\varepsilon],[b,\varepsilon],\ldots,[z,\varepsilon]$
$[␣,\varepsilon]$
A
A'
$[A,\varepsilon]$
[EOL, ,␣A.EOL]
$[Z,\varepsilon]$
[EOL, ,␣Z.EOL]
$[␣,\varepsilon]$
Z
Z'
$[a,\varepsilon],[b,\varepsilon],\ldots,[z,\varepsilon]$
$[A,A],[B,B],\ldots,[Z,Z]$
$[a,a],[b,b],\ldots,[z,z]$

(c)



[(, ε]
$[0,\varepsilon]$
$[0,\varepsilon]$
[), ε]
$[0,0],[1,1],\ldots,[9,9]$
$[0,\varepsilon]$
$[0,\varepsilon]$
[4, +4]
[9, 9␣]
[8, 8]
[9, 9␣]
[EOL, EOL]
[1, +49␣89␣1], ..., [9, +49␣89␣9]