# Automata and Grammars

**SS 2018**

**Assignment 12: Solutions to Selected Problems**

**Problem 12.1.** [Turing Machines]

Design a one-tape Turing machine $M_1$ with at most 8 states such that

$$L(M) = \{\, a^n b^n c^n \mid n \geq 0 \,\}.$$

**Solution.** We present a single-tape TM $M = (Q, \{a,b,c\}, \{a,b,c,A,B,C,\Box\}, \Box, \delta, q_0, q_f)$ for the language $\{\, a^n b^n c^n \mid n \geq 0 \,\}$, where $Q = \{q_0, q_1, q_2, q_3, q_4, q_f\}$ and $\delta$ is given by the following table:

| $\delta$ | $q_0$ | $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_f$ |
|---|---|---|---|---|---|---|
| $a$ | $(q_1, A, R)$ | $(q_1, a, R)$ | $-$ | $-$ | $(q_4, a, L)$ | $-$ |
| $b$ | $-$ | $(q_2, B, R)$ | $(q_2, b, R)$ | $-$ | $(q_4, b, L)$ | $-$ |
| $c$ | $-$ | $-$ | $(q_3, C, R)$ | $(q_3, c, R)$ | $(q_4, c, L)$ | $-$ |
| $A$ | $-$ | $-$ | $-$ | $-$ | $(q_0, A, R)$ | $-$ |
| $B$ | $(q_0, B, R)$ | $(q_1, B, R)$ | $-$ | $-$ | $(q_4, B, L)$ | $-$ |
| $C$ | $(q_0, C, R)$ | $-$ | $(q_2, C, R)$ | $-$ | $(q_4, C, L)$ | $-$ |
| $\Box$ | $(q_f, \Box, 0)$ | $-$ | $-$ | $(q_4, \Box, L)$ | $-$ | $-$ |

Here the states are used as follows:
$q_0$: Search to the right for an $a$,
$q_1$: Search to the right for a $b$,
$q_2$: Search to the right for a $c$,
$q_3$: Search to the right for a $\Box$,
$q_4$: Search to the left for an $A$,
$q_f$: Final state.

If one wants the TM $M$ to not halt on any words that do not belong to the language $\{\, a^n b^n c^n \mid n \geq 0 \,\}$, then one just needs to replace every undefined transition for each state $q \in \{q_0, q_1, q_2, q_3, q_4\}$ by an infinite loop.

The computation of $M$ on input $aabbcc$ looks as follows:

$$\begin{array}{llllll}
q_0aabbcc & \vdash_M & Aq_1abbcc & \vdash_M & Aaq_1bbcc & \vdash_M & AaBq_2bcc \\
 & \vdash_M & AaBbq_2cc & \vdash_M & AaBbCq_3c & \vdash_M & AaBbCcq_3\Box \\
 & \vdash_M & AaBbCq_4c & \vdash_M & AaBbq_4Cc & \vdash_M & AaBq_4bCc \\
 & \vdash_M & Aaq_4BbCc & \vdash_M & Aq_4aBbCc & \vdash_M & q_4AaBbCc \\
 & \vdash_M & Aq_0aBbCc & \vdash_M & AAq_1BbCc & \vdash_M & AABq_1bCc \\
 & \vdash_M & AABBq_2Cc & \vdash_M & AABBCq_2c & \vdash_M & AABBCCq_3\Box \\
 & \vdash_M & AABBCq_4C & \vdash_M & AABBq_4CC & \vdash_M & AABq_4BCC \\
 & \vdash_M & AAq_4BBCC & \vdash_M & Aq_4ABBCC & \vdash_M & AAq_0BBCC \\
 & \vdash_M^4 & AABBCCq_0\Box & \vdash_M & AABBCCq_f\Box & &
\end{array}$$

It can now be seen quite easily that $L(M) = \{\, a^n b^n c^n \mid n \geq 0 \,\}$. $\Box$

**Problem 12.2.** [Turing Machines]

Let $L_{\text{copy}} = \{\, wcw \mid w \in \{a,b\}^* \,\}$.

(a) Design a one-tape Turing machine $M_1$ with at most 8 states such that $L(M_1) = L_{\text{copy}}$.

(b) Design a two-tape Turing machine $M_2$ with at most 4 states such that $L(M_2) = L_{\text{copy}}$.

**Solution.** (a) The TM $M_1$ will work as follows. Let $x = ucv$ be given as input, where $u, v \in \{a,b\}^*$. $M_1$ marks the first letter of $u$, stores it in its finite-state control, and moves right until it reaches the first letter to the right of the symbol $c$. It then compares this letter to the stored symbol. If these two symbols coincide, then it also marks the current symbol and returns to the marked symbol in $u$; otherwise it just halts in a non-accepting state. Once $M_1$ has returned to the marked symbol in $u$, it moves one step to the right, marks the new symbol, stores it in its finite-state control, and moves right again to the first unmarked symbol to the right of the symbol $c$. This process is repeated until either a mismatch is found, and then $M_1$ halts without accepting, or until $M_1$ has verified that $u = v$, and then $M_1$ halts and accepts. To realize this behaviour, we define $M_1 = (Q, \{a,b,c\}, \{a,b,c,*,\Box\}, \Box, \delta_1, q_0, q_f)$, where $Q = \{q_0, q_a, q_b, q'_a, q'_b, p, p', q_f\}$ and $\delta_1$ is given by the following table:

| $\delta_1$ | $a$ | $b$ | $c$ | $*$ | $\Box$ | Comments |
|---|---|---|---|---|---|---|
| $q_0$ | $(q_a, *, R)$ | $(q_b, *, R)$ | $(q_f, \Box, R)$ | $-$ | $-$ | Mark and remember a letter |
| $q_a$ | $(q_a, a, R)$ | $(q_a, b, R)$ | $(q'_a, c, R)$ | $-$ | $-$ | Store $a$ and move right |
| $q_b$ | $(q_b, a, R)$ | $(q_b, b, R)$ | $(q'_b, c, R)$ | $-$ | $-$ | Store $b$ and move right |
| $q'_a$ | $(p, *, L)$ | $-$ | $-$ | $(q'_a, *, R)$ | $-$ | Compare to letter in $v$ |
| $q'_b$ | $-$ | $(p, *, L)$ | $-$ | $(q'_b, *, R)$ | $-$ | Compare to letter in $v$ |
| $p$ | $-$ | $-$ | $(p', c, L)$ | $(p, *, L)$ | $-$ | Return left to $c$ |
| $p'$ | $(p', a, L)$ | $(p', b, L)$ | $-$ | $(q_0, \Box, R)$ | $-$ | Return left |
| $q_f$ | $(p, a, 0)$ | $(p, b, 0)$ | $-$ | $(q_f, \Box, R)$ | $-$ | Accept on empty tape |

Given the word $x = abcab$ as input, $M_1$ executes the following computation:

$$\begin{array}{llllll}
q_0abcab & \vdash_{M_1} *q_abcab & \vdash_{M_1} & *bq_acab & \vdash_{M_1} & *bcq'_aab \\
& \vdash_{M_1} *bpc*b & \vdash_{M_1} & *p'bc*b & \vdash_{M_1} & p'*bc*b \\
& \vdash_{M_1} \Box q_0bc*b & \vdash_{M_1} & \Box * q_bc*b & \vdash_{M_1} & \Box * cq'_b * b \\
& \vdash_{M_1} \Box * c * q'_bb & \vdash_{M_1} & \Box * cp** & \vdash_{M_1} & \Box * pc** \\
& \vdash_{M_1} \Box p' * c** & \vdash_{M_1} & \Box\Box q_0c** & \vdash_{M_1} & \Box\Box\Box q_f** \\
& \vdash_{M_1} \Box\Box\Box\Box q_f* & \vdash_{M_1} & \Box\Box\Box\Box\Box q_f\Box, & &
\end{array}$$

that is, $M_1$ accepts the word $abcab$. On the other hand, on input $y = acab$, $M_1$ executes the following computation:

$$\begin{array}{llllll}
q_0acab & \vdash_{M_1} *q_acab & \vdash_{M_1} & *cq'_aab & \vdash_{M_1} & *pc*b \\
& \vdash_{M_1} p'*c*b & \vdash_{M_1} & \Box q_0c*b & \vdash_{M_1} & \Box\Box q_f*b \\
& \vdash_{M_1} \Box\Box\Box q_fb & \vdash_{M_1} & \Box\Box\Box pb, & &
\end{array}$$

that is, $M_1$ does not accept the word $acab$. It can now be seen that $L(M_1) = L_{\text{copy}}$.

(b) The TM $M_2$ will work as follows. Let $ucv$ be given as input, where $u, v \in \{a, b\}^*$. $M_2$ scans the prefix $u$ from left to right, thereby copying it to tape 2. On reaching the symbol $c$, the head on tape 1 pauses on the symbol $c$, while the head on tape 2 is moved back to the first symbol of $u$. Then $M_2$ compares $u$ (by reading from tape 2) to $v$ (from tape 1). If $u = v$, then $M_2$ accepts.
To realize this behavior, we define $M_2 = (\{q_0, q_l, q_r, q_f\}, \{a, b, c\}, \{a, b, c, \Box\}, \Box, \delta_2, q_0, q_f)$, where $\delta_2$ is given by the following table:

| $\delta_2$ | $q_0$ | $q_l$ | $q_r$ | $q_f$ |
|---|---|---|---|---|
| $(a, \Box)$ | $(q_0, \Box, R, a, R)$ | $-$ | $-$ | $-$ |
| $(a, a)$ | $-$ | $-$ | $(q_r, \Box, R, \Box, R)$ | $-$ |
| $(a, b)$ | $-$ | $-$ | $-$ | $-$ |
| $(a, c)$ | $-$ | $-$ | $-$ | $-$ |
| $(b, \Box)$ | $(q_0, \Box, R, b, R)$ | $-$ | $-$ | $-$ |
| $(b, a)$ | $-$ | $-$ | $-$ | $-$ |
| $(b, b)$ | $-$ | $-$ | $(q_r, \Box, R, \Box, R)$ | $-$ |
| $(b, c)$ | $-$ | $-$ | $-$ | $-$ |
| $(c, \Box)$ | $(q_l, c, 0, c, L)$ | $(q_r, \Box, R, \Box, R)$ | $-$ | $-$ |
| $(c, a)$ | $-$ | $(q_l, c, 0, a, L)$ | $-$ | $-$ |
| $(c, b)$ | $-$ | $(q_l, c, 0, b, L)$ | $-$ | $-$ |
| $(c, c)$ | $-$ | $-$ | $-$ | $-$ |
| $(\Box, \Box)$ | $-$ | $-$ | $-$ | $-$ |
| $(\Box, a)$ | $-$ | $-$ | $-$ | $-$ |
| $(\Box, b)$ | $-$ | $-$ | $-$ | $-$ |
| $(\Box, c)$ | $-$ | $-$ | $(q_f, \Box, 0, \Box, 0)$ | $-$ |

Given the word $x = abcab$ as input, $M_2$ executes the following computation:

$$\begin{array}{llll} (q_0abcab, q_0\Box) & \vdash_{M_2} (\Box q_0bcab, aq_0\Box) & \vdash_{M_2} & (\Box\Box q_0cab, abq_0\Box) \\ & \vdash_{M_2} (\Box\Box q_lcab, aq_lbc) & \vdash_{M_2} & (\Box\Box q_lcab, q_labc) \\ & \vdash_{M_2} (\Box\Box q_lcab, q_l\Box abc) & \vdash_{M_2} & (\Box\Box\Box q_rab, \Box q_rabc) \\ & \vdash_{M_2} (\Box^4 q_rb, \Box^2 bc) & \vdash_{M_2} & (\Box^5 q_r\Box, \Box^3 q_rc) \\ & \vdash_{M_2} (\Box^5 q_f\Box, \Box^3 q_f\Box), & & \end{array}$$

that is, $M_2$ accepts on input $abcab$.
On input $y = acab$, $M_2$ executes the following computation:

$$\begin{array}{llll} (q_0acab, q_0\Box) & \vdash_{M_2} (\Box q_0cab, aq_0\Box) & \vdash_{M_2} & (\Box q_lcab, q_lac) \\ & \vdash_{M_2} (\Box q_lcab, q_l\Box ac) & \vdash_{M_2} & (\Box\Box q_rab, \Box q_rac) \\ & \vdash_{M_2} (\Box^3 q_rb, \Box^2 q_rc), & & \end{array}$$

which is non-accepting. It can be shown that $L(M_2) = L$. $\Box$

**Problem 12.3.** [Turing Machines]

Let $f : \mathbb{N} \to \mathbb{N}$ be the function $f(n) = |\mathrm{dya}(n)|_2$, that is, for each non-negative integer $n$, $f(n)$ is the number of occurrences of the digit 2 in the dyadic representation of $n$. Construct a two-tape Turing machine $M$ with at most 8 states that computes the function $f$.

**Hint:** The dyadic representation of a positive integer $n$ is the word $w = a_m a_{m-1} \cdots a_1 a_0 \in \{1,2\}^+$ such that $n = \sum_{i=0}^{m} a_i \cdot 2^i$. The advantage of the dyadic representation over the binary representation is the fact that it establishes a bijection between the set of positive integers and the set of words $\{1,2\}^+$, while the binary representation is not unique if leading zeros are allowed.

**Solution.** Observe that the input $n$ as well as the result $f(n)$ are written on the tape of $M$ in their dyadic representations. Let $M = (Q, \{1,2\}, \{1,2,\Box\}, \Box, \delta, p_0, p_f)$, where $Q = \{p_0, p_1, p_2, p_f, q_0, q_2, q_3\}$ and $\delta$ is given by the following table:

| $\delta$ | $p_0$ | $p_1$ | $q_0$ | $q_2$ | $q_3$ | $p_2$ | $p_f$ |
|---|---|---|---|---|---|---|---|
| $(\Box,\Box)$ | $(p_f,\Box,0,\Box,0)$ | $(p_2,\Box,L,\Box,0)$ | $(q_3,\Box,0,\Box,L)$ | $(p_0,\Box,0,\Box,R)$ | $(p_0,\Box,0,1,0)$ | $(p_f,\Box,R,\Box,0)$ | $-$ |
| $(\Box,1)$ | $(p_1,\Box,0,1,0)$ | $(p_1,1,R,\Box,R)$ | $(q_0,\Box,0,1,R)$ | $(q_2,\Box,0,1,L)$ | $(q_2,\Box,0,2,L)$ | $-$ | $-$ |
| $(\Box,2)$ | $(p_1,\Box,0,2,0)$ | $(p_1,2,R,\Box,R)$ | $(q_0,\Box,0,2,R)$ | $(q_2,\Box,0,2,L)$ | $(q_3,\Box,0,1,L)$ | $-$ | $-$ |
| $(1,\Box)$ | $(p_0,\Box,R,\Box,0)$ | $-$ | $(q_3,1,0,\Box,L)$ | $(p_0,1,0,\Box,R)$ | $(p_0,1,0,1,0)$ | $(p_2,1,L,\Box,0)$ | $-$ |
| $(1,1)$ | $(p_0,\Box,R,1,0)$ | $-$ | $(q_0,1,0,1,R)$ | $(q_2,1,0,1,L)$ | $(q_2,1,0,2,L)$ | $-$ | $-$ |
| $(1,2)$ | $(p_0,\Box,R,2,0)$ | $-$ | $(q_0,1,0,2,R)$ | $(q_2,1,0,2,L)$ | $(q_3,1,0,1,L)$ | $-$ | $-$ |
| $(2,\Box)$ | $(q_0,\Box,R,\Box,0)$ | $-$ | $(q_3,2,0,\Box,L)$ | $(p_0,2,0,\Box,R)$ | $(p_0,2,0,1,0)$ | $(p_2,2,L,\Box,0)$ | $-$ |
| $(2,1)$ | $(q_0,\Box,R,1,0)$ | $-$ | $(q_0,2,0,1,R)$ | $(q_2,2,0,1,L)$ | $(q_2,2,0,2,L)$ | $-$ | $-$ |
| $(2,2)$ | $(q_0,\Box,R,2,0)$ | $-$ | $(q_0,2,0,2,R)$ | $(q_2,2,0,2,L)$ | $(q_3,2,0,1,L)$ | $-$ | $-$ |

Observe that using the states $q_0, q_2, q_3$, $M$ simulates the Turing machine for the dyadic +1-function on its second tape (see the corresponding example in Section 4.1).

Given the number $n = 12$ as input, $M$ executes the following computation. Recall that $\mathrm{dya}(12) = 212$, that is, $f(12) = 2$:

$$\begin{array}{llllll}
(p_0 212, p_0\Box) & \vdash_M (\Box q_0 12, q_0\Box) & \vdash_M & (\Box q_3 12, q_3 \Box\Box) & \vdash_M & (\Box p_0 12, p_0 1) \\
& \vdash_M (\Box\Box p_0 2, p_0 1) & \vdash_M & (\Box^3 q_0 \Box, q_0 1) & \vdash_M & (\Box^3 q_0 \Box, 1 q_0 \Box) \\
& \vdash_M (\Box^3 q_3 \Box, q_3 1) & \vdash_M & (\Box^3 q_2 \Box, q_2 \Box 2) & \vdash_M & (\Box^3 p_0 \Box, p_0 2) \\
& \vdash_M (\Box^3 p_1 \Box, p_1 2) & \vdash_M & (\Box^3 2 p_1 \Box, \Box p_1 \Box) & \vdash_M & (\Box^3 p_2 2, p_2 \Box) \\
& \vdash_M (\Box^2 p_2 \Box 2, p_2 \Box) & \vdash_M & (\Box^3 p_f 2, p_f \Box). & &
\end{array}$$

Thus, first $M$ scans and deletes its input on tape 1 from left to right, simulating the dyadic +1-machine on tape 2 each time it detects a 2 on tape 1. After that it copies the result from tape 2 to tape 1, erasing tape 2 in the process. Finally, the head on tape 1 is moved to the first symbol of the result. Observe that $f(0) = 0$, and that $\mathrm{dya}(0) = \varepsilon$. $\Box$

**Problem 12.4** [Phrase-Structure Grammars]

Determine the languages that are generated by the following general grammars:

$$
\begin{array}{llll}
\text{(a)} & G_1 & = & (\{S,A,B,C,D,E\},\{a,b\},P_1,S), \text{ where } P_1 \text{ is defined as follows:}\\
 & P_1 & = & \{S \to EC, S \to \varepsilon, C \to ACa, C \to BCb, C \to D,\\
 & & & aA \to Aa, bA \to Ab, aB \to Ba, bB \to Bb, EA \to Ea, EB \to Eb,\\
 & & & aD \to Da, bD \to Db, ED \to \varepsilon\},\\
\text{(b)} & G_2 & = & (\{S,A,B\},\{a,b\},P_2,S), \text{ where } P_2 \text{ is defined as follows:}\\
 & P_2 & = & \{S \to ASB, S \to BSA, S \to SS, S \to \varepsilon,\\
 & & & AB \to \varepsilon, BA \to \varepsilon, A \to a, B \to b\}.
\end{array}
$$

**Solution.** (a) We claim that $L(G_1) = L_{\text{copy}} = \{\, ww \mid w \in \{a,b\}^* \,\}$. First we show that

$$\{\, EwCw \mid w \in \{a,b\}^* \,\} \subseteq \hat{L}(G_1) \cap E \cdot (\{C,a,b\})^*.$$

To prove this inclusion, we proceed by induction on $|w|$. If $|w| = 0$, then $w = \varepsilon$, and we see that $S \to_{G_1} EC = EwCw$. If $|w| = 1$, then $w = a$ or $w = b$. In the former case $S \to_{G_1} EC \to_{G_1} EACa \to_{G_1} EaCa$, and the other case is analogous. Now assume that $w = au$. By the induction hypothesis, we have $S \to_{G_1}^* EuCu$. Now we can continue as follows:

$$EuCu \;\to_{G_1}\; EuACau \;\to_{G_1}^*\; EAuCau \;\to_{G_1}\; EauCau = EwCw.$$

As $EwCw \to_{G_1} EwDw \to_{G_1}^* EDww \to_{G_1} ww$, we see that $L_{\text{copy}} \subseteq L(G_1)$.

From we set of productions $P_1$, we see that $\hat{L}(G_1) = \{\varepsilon\} \cup \hat{L}(G_1, EC)$ and that $\hat{L}(G_1, C) = \{\, W^R Cw, W^R Dw \mid W \in \{A,B\}^*, \pi(W) = w\}$, where $\pi(A) = a$ and $\pi(B) = b$. In order to rewrite the nonterminals $A$ and $B$ into the terminals $a$ and $b$, we need the productions containing $E$ on the left-hand side. These show that $EW^R Cw \to_{G_1}^* EwCw$ and $EW^R Dw \to_{G_1}^* EwDw \to_{G_1}^* EDww \to_{G_1} ww$ are essentially the only derivations that rewrite all these nonterminals. Hence, we see that $L(G_1) = L_{\text{copy}}$.

(b) We claim that $L(G_2) = L_{\text{gl}} = \{\, w \in \{a,b\}^* \mid |w|_a = |w|_b \,\}$. From the form of the productions we see that $|\alpha|_A + |\alpha|_a = |\alpha|_B + |\alpha|_b$ for all $\alpha \in \hat{L}(G_2)$. This implies that $L(G_2) \subseteq L_{\text{gl}}$.

To prove the converse inclusion, let $w \in L_{\text{gl}}$. We proceed by induction on $|w|$. If $|w| = 0$, then $w = \varepsilon$, and $S \to_{G_2} \varepsilon = w$. If $|w| = 2$, then $w = ab$ or $w = ba$, and $S \to_{G_2} ASB \to_{G_2}^3 ab$ and $S \to_{G_2} BSA \to_{G_2}^3 ba$. Now let $|w| = 2n+2$. Then $w = aub$ or $w = bua$ for some word $u \in L_{\text{gl}}$ such that $|u| = 2n$, or $w = u_1 a u_2 b$ or $w = u_1 b u_2 a$ for some words $u_1, u_2 \in L_{\text{gl}}$ such that $|u_1| + |u_2| = 2n$. In the former case we have $S \to_{G_2} ASB \to_{G_2}^2 aSb \to_{G_2}^* aub = w$, and analogously for $w = bua$. In the latter case we have $S \to_{G_2} SS \to_{G_2}^* u_1 S \to_{G_2} u_1 ASB \to_{G_2}^* u_1 a u_2 b = w$, and analogously for the other case. Thus, we see that $L(G_2) = L_{\text{gl}}$. $\square$