COMP 335: Assignement #4 Solution

Winter, 2013

Tuesdays & Thursdays 13:15-14:05

Instructor: Professor Grahne

Tutor: F. Boroomand

Solution # 1

Assume that we have a PDA, $L = (Q_1, \Sigma, \Gamma, \delta_1, q_{01}, F_1)$ and a DFA, $M = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$. We can construct the intersection of G_1 and G_2 using product construction as follows:

$$product(L, M) = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

- $Q = Q_1 \times Q_2$
- $\big((p_1, p_2), \gamma'\big) \in \delta\big((p_1, p_2), \sigma, \gamma\big)$ if and only if $(p_1, \gamma') \in \delta_1(q_1, \sigma, \gamma)$ and $p_2 \in \delta_2(q_2, \sigma)$
- $q_0 = (q_{01}, q_{02})$
- $F = F_1 \times F_2$

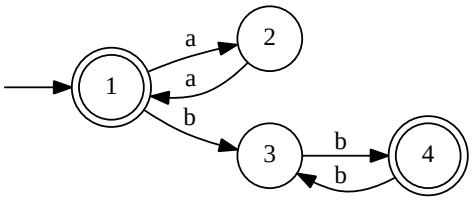


Figure 1: DFA describing Language L

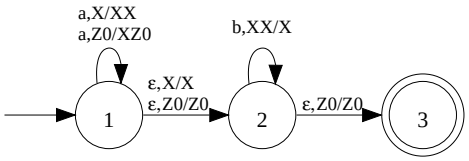


Figure 2: PDA describing language M

Following the product construction procedure we have the following PDA:

$$product(M, L) = (Q, \Sigma, \Gamma, \delta, (1,1), \{(1,3), (4,3)\})$$

state	input	stack	new state	new stack
$(1,1)$	a	Z_0	$(2,1)$	Z_0
$(1,1)$	a	X	$(2,1)$	XX
$(1,1)$	ϵ	Z_0	$(1,2)$	Z_0
$(1,1)$	ϵ	X	$(1,2)$	XX
$(1,2)$	b	XX	$(3,2)$	X
$(1,2)$	ϵ	Z_0	$(1,3)$	Z_0
$(3,2)$	b	XX	$(4,2)$	X
$(4,2)$	b	XX	$(3,2)$	X
$(4,2)$	ϵ	Z_0	$(4,3)$	Z_0

Table 1: δ function

Solution # 2

$$\begin{aligned} S &\rightarrow aA|aBB \\ A &\rightarrow aaA|\epsilon \\ B &\rightarrow bB|bbC \\ C &\rightarrow C|B \end{aligned}$$

For the conversion we have to follow the following steps:

1. Eliminate useless symbols: C and B are nongenerating variables.

$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow aaA|\epsilon \end{aligned}$$

2. Eliminate the ϵ-production

$$\begin{aligned} S &\rightarrow aA|a \\ A &\rightarrow aaA|aa \end{aligned}$$

3. Eliminate all unit rules

$$\begin{aligned} S &\rightarrow aA|a \\ A &\rightarrow aaA|aa \end{aligned}$$

4. Break bodies of length three or more

$$\begin{aligned} S &\rightarrow aA|a \\ A &\rightarrow aA_1|aa \\ A_1 &\rightarrow aA \end{aligned}$$

5. Change variables

$$\begin{aligned} S &\rightarrow V_1A|V_1 \\ A &\rightarrow V_1A_1|V_1V_1 \\ A_1 &\rightarrow V_1A \\ V_1 &\rightarrow a \end{aligned}$$

Solution # 3

a) Consider the string $z = a^n b^n a^n b^n \in L_1$. Let n be the pumping length. We should consider all the following cases and show the contradiction with pumping lemma. We only show the contradiction for the first case here, the rest follows the same procedure.

- Let vwx to be part of first block of a's. $z = \overbrace{a^{n-k}}^{u}\overbrace{a^k}^{vwx}\overbrace{b^n a^n b^n}^{y}$. We see that the conditions for pumping lemma are satisfied ($|vwx| \leq n$, $vx \neq \epsilon$). We see that $z = \overbrace{a^{n-k}}^{u} a^{2k} \overbrace{b^n a^n b^n}^{y} = a^{n+k} b^n a^n b^n \notin L_1$ which contradicts the pumping lemma.
- Let vwx to have some symbols from the first block of a's and first block of b's.
- Let vwx to have some symbols from the first block of b's and second block of a's.
- Let vwx to have some symbols from the second block of a's and second block of b's.
- Let vwx to be part of second block of b's.

b) Consider the string $z = a^{n^2} b^n \in L_2$. Let n be the pumping length. We have to consider the following cases:

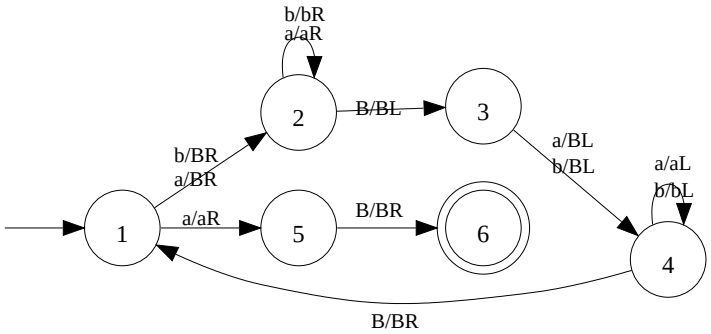
- Let vwx to be all a's. $z = \overbrace{a^{n^2-k}}^{u}\overbrace{a^k}^{vwx}\overbrace{b^n}^{y}$ where $|k| \leq n$. The conditions of pumping lemma are satisfied. Now pumping the strings v and x we see that the resulting string does not belong to L_2. This contradicts the pumping lemma.
- Let vwx to have some a's and some b's.
- Let vwx to be all b's.

c) Consider the string $z = a^n b^{n+1} c^n \in L_3$. Let n be the pumping length. We have to consider the following cases:

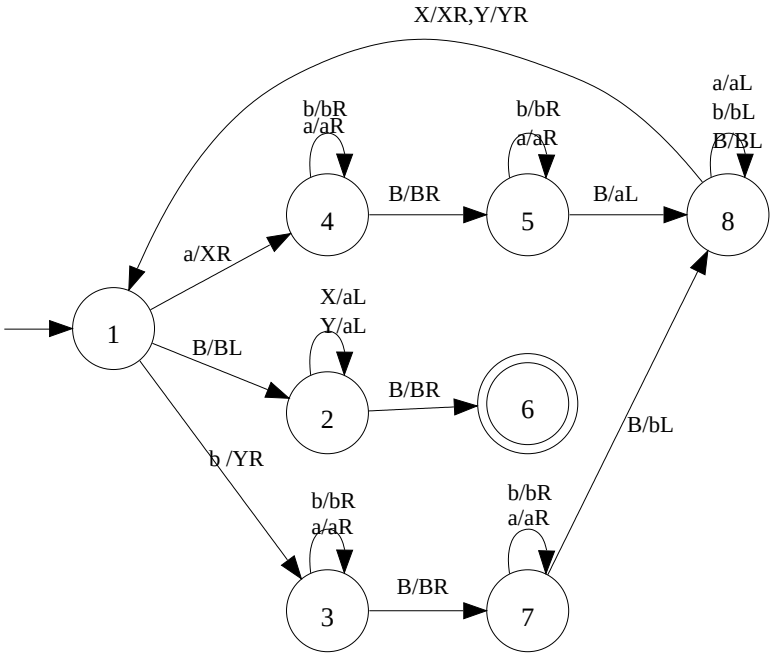
- Let vwx to be all a's. $z = \overbrace{a^{n-k}}^{u}\overbrace{a^k}^{vwx}\overbrace{b^{n+1} c^n}^{y}$ where $|k| \leq n$. The conditions of pumping lemma are satisfied. Now pumping the strings v and x we see that the resulting string does not belong to L_3. This contradicts the pumping lemma.
- Let vwx to have some a's and some b's.
- Let vwx to be all b's.
- Let vwx to have some b's and some c's.
- Let vwx to be all c's.

Solution # 4

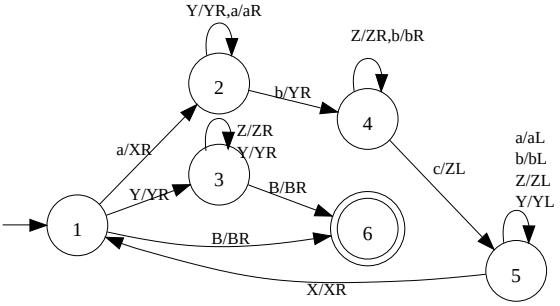
a) The following TM represents the language L_a:



b) The following TM represents the language L_b:



d) The following TM represents the language L_d:



Solution # 5

$$\begin{aligned}
q_1 &\rightarrow 0 \\
q_2 &\rightarrow 00 \\
q_3 &\rightarrow 000 \\
a_1 &\rightarrow 0 \\
a_2 &\rightarrow 00 \\
L &\rightarrow 0 \\
R &\rightarrow 00 \\
\delta(q_1, a_1) &= (q_1, a_1, R) \rightarrow 0101010100 \\
\delta(q_1, a_2) &= (q_3, a_1, L) \rightarrow 010010001010 \\
\delta(q_3, a_1) &= (q_2, a_2, L) \rightarrow 0001010010010
\end{aligned}$$

Thus, the complete encoding of the TM is 0101010100110100100010101100010100100 10.

Solution # 6

- On a slope of -2 in the matrix: Suppose L_i is accepted by some machine. We may assume there is at least one transition defined for this machine, because otherwise we could add in a trivial one (from an extra state that never gets reached, to anywhere). Thus, using the enumeration scheme defined in the textbook we have an index ending in 0 for this machine; i.e., our machine is M_{2j} for some positive integer j. If we now consider the string w_j we find: $w_j \in L_i$ iff M_{2j} accepts w_j iff $w_j \notin L_i$ A, a contradiction. So in fact there can be no machine accepting L_i.
- On a slope of $-1/2$ in the matrix: Let L'_{2i} be the set of all w_{2i} such that w_{2i} is not accepted by M_i. Clearly L_{2i} is not accepted by any Turing machine. On the other hand, if L_{2i} were accepted by some machine M then we could construct a machine M' that accepts L'_{2i} as follows: given a string of odd index, reject; given a string of even index, run M on the string with half that index. This implies there is no TM that accepts L_{2i}.

Solution # 7

- ($\Rightarrow$) L is recursive. This means that L is accepted by a Turing machine M which always halts. We have to show that L and $\overline{L}$ are recursively enumerable. Since every recursive language is recursively enumerable, L is recursively enumerable. To accept $\overline{L}$ we just switch the accepting and non-accepting states of M. This makes $\overline{L}$ recursively enumerable as well.
- ($\Leftarrow$) L and $\overline{L}$ are recursively enumerable. This means that L is accepted by a standard Turing Machine M_1 and that $\overline{L}$ is accepted by another standard Turing machine M_2 . We have to show that L is recursive, so that it is accepted by a standard Turing Machine M which always halts. To get M, we run M_1 and M_2 in parallel, say we have two tapes and we run M_1 on tape 1 and M_2 on tape 2. M accepts if M_1 accepts, and M rejects if M_2 accepts. Note that since either M_1 or M_2 accepts, thus M always halts. Furthermore, $L(M) = L(M_1) = L$. So L is recursive.

Solution # 8

We want to show this by reduction from L_u. Let $< M, w > \in L_u$. Let M' be a machine such that it ignores its input and simulates M on w. If M accepts w then M' will write a 1 on its tape, else it does nothing. We need to make sure that the tape language of M does not contain a 1 that would confuse our solver.

If the language of "writes 1" was decidable, then we could use this machine to decide L_u. We know L_u is undecidable so the "writes 1" language must also be.

Solution # 9

a) There is a solution as follows:

$$\begin{aligned} w &= 11\ \ 100\ \ 111 \\ x &= 111\ \ 001\ \ 11 \end{aligned}$$

b) There is no possible solution. Note that $|w_1| > |x_1|, |w_2| > |x_2|, |w_3| > |x_3|$ So, the list $x_{i1}x_{i2}...x_{im}$ can never catch up with $w_{i1}w_{i2}...w_{im}$. Thus, PCP has no solution.