Assignment #4

Gustavo Estrela de Matos
CSCE 433: Formal Languages and Automata

March 24, 2016

Question 1.

Question 2.

Question 3.

(a) $G = (V, \Sigma, R, S)$ where:

- $V = \{S\}$
- $\Sigma = \{a, b\}$

Note that $\epsilon \in L_1$, therefore our first rule is $S \to \epsilon$. To build other strings we have to add 1's into the end and twice this number of 0's at the beggining of the string, then the second rule is $S \to 00S1$.

- $R = \{S \to \epsilon,\ S \to 00S1\}$

(b) $G = (V, \Sigma, R, S)$ where:

- $V = \{S\}$
- $\Sigma = \{a, b\}$

Note that $\epsilon \in L_1$, therefore our first rule is $S \to \epsilon$. To build other strings of this language we have to guarantee that $m \geq n$ and keep m and n both odd or even. Which means that:
1 $S \to aSb$; whenever adding an a you have to add a new b to keep $m \geq n$ and $m - n$ even.
2 $S \to Sbb$; whenever adding a b we need to add two of them to keep $m - n$ even.
Therefore the rules for this language are:

- $R = \{S \to \epsilon,\ S \to Sbb,\ S \to aSb\}$

(c) $G = (V, \Sigma, R, S)$ where:

- $V = \{S, S_1, R_1, T_1, S_2, R_2, T_2\}$
- $\Sigma = \{a, b, c, d\}$

Note that the equation $m + n = p + q$ means that for every a or b added we need to add a new c or d. To do that we are going to build the string from the outside to inside adding for every a or b a new d or a new c. You should also note that it's hard have the pair of rules "for every a add a new c" and "for every b add a new d" because they intersect each other; to deal with this we are going to create rules for two different situations: $m \geq q$ and $m \leq q$.

In the case where $m \geq q$ we are looking for the language $L_1 = \{a^m b^n c^p d^q \mid m + n = p + q, m \geq q\}$ generate the strings with the following rules:

1 $S_1 \to aS_1d|R_1$

2 $R_1 \to aR_1c|T_1$

3 $T_1 \to bT_1c|\epsilon$

We do not need a rule that adds a new d when adding a b because we assumed that $m \geq q$ therefore every d of a string would already have been added by rule 1.

In the case where $m \leq q$ we are looking for the language $L_2 = \{a^m b^n c^p d^q \mid m + n = p + q, m \leq q\}$ generate the strings with the following rules:

4 $S_2 \to aS_2d|R_2$

5 $R_2 \to bR_2d|T_2$

6 $T_2 \to bT_2c|\epsilon$

Simmilarly we do not need a rule that adds a new c when adding a a because we assumed that $m \leq q$ therefore every a of a string would already have been added by rule 4. Now the last rule we need is the one that makes the union of L_1 and L_2:

7 $S \to S_1|S_2$

- $R = \{S \to S_1|S_2,\ S_1 \to aS_1d|R_1,\ R_1 \to aR_1c|T_1,\ T_1 \to bT_1c|\epsilon,\ S_2 \to aS_2d|R_2,\ R_2 \to bR_2d|T_2,\ T_2 \to bT_2c|\epsilon\}$

Question 4.

$G = (V, \Sigma, R, S)$ where:

- $V = \{S, R\}$
- $\Sigma = \{a, b, *, (,), +, \varepsilon\}$; to avoid ambiguity we are going to use the symbol ε to represent the empty string in the regular expressions we want to build.

To create all the regular expressions we are going to use two different variables: R and S. The variable S represents any non-empty regular expression while R can be either S or the empty string ϵ.

1 $S \to aR$
2 $S \to bR$
3 $S \to \varepsilon R$
4 $S \to S^*R$
5 $S \to (S)R$
6 $S \to S + S$
7 $R \to S$
8 $R \to \epsilon$

- $R = \{S \to aR, S \to bR, S \to \varepsilon R, S \to S^*R, S \to (S)R, S \to S + S, R \to S, R \to \epsilon\}$