CPSC 3630
Assignment 3

Cody Barnson
ID: 001172313

Nov 10 2018

1 Prove that for any language L_2, if L_1 is regular, then the quotient L_1/L_2 is also regular by using a construction similar to the proof of Theorem 1.47.

Let L_1 and L_2 be 2 languages over the alphabet Σ. The quotient of L_1 and L_2 is the language $L_1/L_2 = \{x \mid \exists y \in L_2, xy \in L_1\}$. We wish to prove that for any language L_2, if L_1 is regular, then the quotient L_1/L_2 is also regular.

Suppose the L_1 is regular, then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ that recognizes L_1. We construct the DFA $M' = (Q, \Sigma, \delta, q_0, F')$ to recognize the quotient L_1/L_2, where the set of states, Q, is the same as in M; the alphabet Σ is the same; the transition function is the same; the start state q_0 is the start state of M, and the set of accept states is given by $F' = \{q \mid \delta(q, a) \in F\}$, for each $a \in \Sigma$.

Since for any input, say $x \in \Sigma^*$, that results in accept state $q_{accept} \in F'$, we have $xy \in L_1$, since reading x transitions to accept state in F', and the following y transitions to accept state in the original machine's set of accept states of F. Thus L_1/L_2 is regular, whenever L_1 is regular.

2 Use the pumping lemma to show that the following language is not regular.

$$A = \{w \in \{0,1\}^* \mid \text{the length of } w \text{ is a perfect square}\}$$

Suppose language A is regular. Then there exists pumping length n. If we consider the string 0^{n^2}, where $|w| > n$ and $w \in A$ (by definition). Note, we choose the string of 0 symbols for simplicity, but without loss of generality for this proof (since there is no restriction on w other than its length is a perfect square). For any decomposition $w = xyz$, we have $|y| \geq 1$ and $|xy| \leq n$, so $1 \leq |y| \leq n$.

We can pump w to get $w_1 = xy^2z$, with $n^2 \leq |xy^2z| \leq n^2 + n$. We notice that the next perfect square has length $(n+1)^2 = n^2 + 2n + 1$, however since we have the following inequality:

$$\begin{aligned} n^2 + n &< n^2 + 2n + 1 \\ n &< 2n + 1 \end{aligned}$$

And we only added at most n, but need $2n+1$ added, then w_1 must not be a perfect square, and thus $w_1 \notin A$.

Now, since all possible pumping of w must be in language A if A is regular (by definition), and we have just shown that $w_1 \notin A$, we have a contradiction, so A must not be regular.

3 Identify and explain clearly the error in the purported proof below that the language described by the regular expression (r.e.) 0^*1^* is not a regular language.

The proof error is in the following statement: "However s cannot be pumped since the language $\{0^n1^n \mid n \geq 0\}$ is not regular."

Let L be the language in question. The string s, (i.e. $s = 0^p1^p$) has the same number of 0's as 1's, and $0^p1^p \in L$. If 0^p1^p is pumped, it no longer has equal number of 0's and 1's, however the resulting string is still in L because all the 0's come before all 1's. So s can be pumped, which the proof claims otherwise. A correct proof would need to choose string s such that s can not be pumped, in order to show that 0^*1^* is not regular.

4 Let $B_n = \{a^k \mid k \text{is a multiple of } n\}$ where $\Sigma = \{a\}$. Prove that for each $n \geq 1$, the language B_n is regular.

For any $n \geq 1$, B_n is the language consisting of strings a^{nt}, where $t < \mathbb{Z}_{\geq 0}$, and for $t = 0$, we have $\epsilon \in B_n, \forall n \geq 1$. If B_n can be described by some DFA for some fixed $n \geq 1$ then B_n is regular. We will give such DFA, M, that recognizes B_n, as depicted below for fixed n:

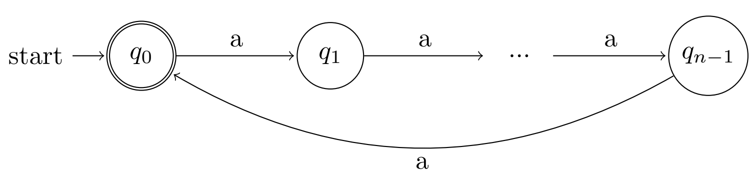

Figure 1: State diagram of DFA M that recognizes B_n for some fixed $n \geq 1$.

Thus, B_n is regular.

5 Give context-free grammars for each of the following languages where $\Sigma = \{0, 1\}$.

Note: for each of the following CFG's, assume the start non-terminal is S.

5.a $\{w \mid w$ contains an odd number of symbols$\}$

For this language, we have CFG:

$$\begin{aligned} S &\longrightarrow S_0 \mid S_1 \\ S_0 &\longrightarrow 0S_1 \mid 1S_1 \\ S_1 &\longrightarrow 00S_1 \mid 01S_1 \mid 10S_1 \mid 11S_1 \mid \epsilon \end{aligned}$$

5.b $\{w \mid w$ contains more 1's than 0's$\}$

For this language, we have CFG:

$$\begin{aligned} S &\longrightarrow TS \mid 1S \mid 1T \\ T &\longrightarrow TT \mid 0T1 \mid 1T0 \mid \epsilon \end{aligned}$$

5.c The empty set.

For this language, we have CFG:

$$S \longrightarrow S$$

Notice that S never resolves, so we get the $\emptyset$.

6 Convert the following CFG where $\Sigma = \{0\}$ and A is the start non-terminal, into an equivalent Chomsky normal form using the procedure given in Theorem 2.9.

We begin with the following CFG:

$$\begin{aligned} A &\longrightarrow BAB \mid B \mid \epsilon \\ B &\longrightarrow 00 \mid \epsilon \end{aligned}$$

Step 1 Create new start variable A_0 and a corresponding new rule.

$$\begin{aligned} A_0 &\longrightarrow A \\ A &\longrightarrow BAB \mid B \mid \epsilon \\ B &\longrightarrow 00 \mid \epsilon \end{aligned}$$

Step 2 Remove the ϵ-rules.

Remove $B \longrightarrow \epsilon$

$$\begin{aligned} A_0 &\longrightarrow A \\ A &\longrightarrow BAB \mid BA \mid AB \mid A \mid B \mid \epsilon \\ B &\longrightarrow 00 \end{aligned}$$

Remove $A \longrightarrow \epsilon$

$$\begin{aligned} A_0 &\longrightarrow A \mid \epsilon \\ A &\longrightarrow BAB \mid BA \mid AB \mid A \mid B \mid BB \\ B &\longrightarrow 00 \end{aligned}$$

Step 3 Remove unit rules.

Remove $A \longrightarrow A$

$$\begin{aligned} A_0 &\longrightarrow A \mid \epsilon \\ A &\longrightarrow BAB \mid BA \mid AB \mid B \mid BB \\ B &\longrightarrow 00 \end{aligned}$$

Remove $A \longrightarrow B$

$$\begin{aligned} A_0 &\longrightarrow A \mid \epsilon \\ A &\longrightarrow BAB \mid BA \mid AB \mid 00 \mid BB \\ B &\longrightarrow 00 \end{aligned}$$

Remove $A_0 \longrightarrow A$

$$\begin{aligned} A_0 &\longrightarrow BAB \mid BA \mid AB \mid 00 \mid BB \mid \epsilon \\ A &\longrightarrow BAB \mid BA \mid AB \mid 00 \mid BB \\ B &\longrightarrow 00 \end{aligned}$$

Step 4 Replace terminals, 00, by new variable, say U and corresponding new rule.

$$\begin{aligned} A_0 &\longrightarrow BAB \mid BA \mid AB \mid UU \mid BB \mid \epsilon \\ A &\longrightarrow BAB \mid BA \mid AB \mid UU \mid BB \\ B &\longrightarrow UU \\ U &\longrightarrow 0 \end{aligned}$$

Step 5 Shorten long rules with RHS length ≥ 3.

$$\begin{aligned} A_0 &\longrightarrow BA_1 \mid BA \mid AB \mid UU \mid BB \mid \epsilon \\ A &\longrightarrow BA_2 \mid BA \mid AB \mid UU \mid BB \\ B &\longrightarrow UU \\ U &\longrightarrow 0 \\ A_1 &\longrightarrow AB \\ A_2 &\longrightarrow AB \end{aligned}$$

The final result from step 5 (above) shows our R of our Chomsky normal form, described by $G = (V, \Sigma, R, A_0)$, where $V = \{A_0, A, B, U, A_1, A_2\}$, $\Sigma = \{0\}$, and start variable A_0.

7 Show that the class of context-free languages is closed under the regular operations: union and star.

Let CFL be the set of all context-free languages.

7.a For all $L_1, L_2 \in CFL$, show $L_1 \cup L_2 \in CFL$

Let S_1, S_2 be the start variable for languages L_1, L_2, respectively. Then we can describe the grammar:

$$S \longrightarrow S_1 \mid S_2$$

Which is the grammar for the union $L_1 \cup L_2$, since it will generate all strings generated by start variables S_1, S_2, or both. Since L_1, L_2 are arbitrary, we have shown that the class of context-free languages is closed under the union regular operation.

7.b For all $L_1 \in CFL$, show $L_1^* \in CFL$

Let S_1 be the start variable for language L_1. Then, the following grammar:

$$S \longrightarrow S_1 S \mid \epsilon$$

Since this grammar will generate zero or more strings from L_1, and this is exactly the definition of the star regular operation, we have shown that the class of context-free languages is closed under the star regular operation.

8 Construct a PDA that recognizes the language $\{ww^R \mid w \in \{a,b\}^*\}$, where w^R is the string w written backwards.

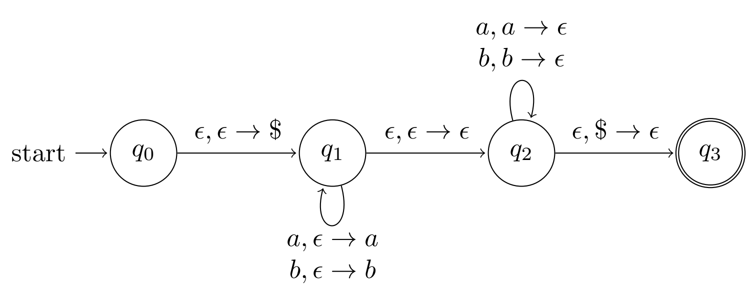

Figure 2: PDA that recognizes the language $\{ww^R \mid w \in \{a,b\}^*\}$, where w^R is the string w written backwards.

The diagram in Figure 2 shows the PDA that recognizes the language in question, since we can stop reading w at any point (and exactly one of these) and begin reading w^R to recognize ww^R, as desired.