CS 321, Assignment 6

Cody Malick
malickc@oregonstate.edu

November 14, 2016

1

a

Given some machine $M = \{Q, \Sigma, \delta, s, F\}$ and a CFG $G = \{A_p \rightarrow cA_q | \delta(p, c)\} \cup \{A_p \rightarrow \epsilon | p \in F\}$

$$\delta^*(s, w) = q \iff A_s \overset{*}{\iff} wA_q$$

Base Case:
$w = \epsilon, q = q$
$A_s \overset{*}{\iff} A_s = \epsilon A_s$

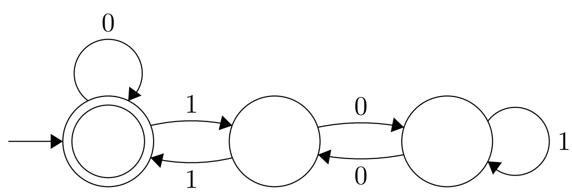
Inductive Step:
$w = xb$, assuming the inductive hypothesis is true for x.
$\delta^*(s, xb) = \delta(\delta^*(s, x), b)$
$p = \delta^*(s, x)$
$q = \delta(p, b) \iff A_p \Rightarrow bA_q$
$\delta^*(s, x) = p \iff A_s \Rightarrow xA_p = A_s \Rightarrow x(bA_q) = A_s \Rightarrow wA_q$

Because $p = \delta^*(s, x)$ then we can show that reading an additional character is no different than reading a character in the original string.

Therefore, the inductive hypothesis holds.

b

Original:



CFG:
$S \rightarrow 0S|1T|\epsilon$
$T \rightarrow 1S|0U$
$U \rightarrow 0T|1U$

2

Starting state:
$S \to aSddd \mid T$
$T \to bTdd \mid R$
$R \to cR \mid \epsilon$

Step 1, add new start symbol S^*, add rule $S^* \to S$
$S^* \to S$
$S \to aSddd \mid T$
$T \to bTdd \mid R$
$R \to cR \mid \epsilon$

Step 2, Shorten RHS rules for each rule $A \to \alpha_1, \alpha_2, ..., \alpha_k$ where $k \geq 3$
$S^* \to S$
$S \to MN \mid T$
$M \to aS$
$N \to dO$
$O \to dd$
$T \to PO \mid R$
$P \to bT$
$R \to cR \mid \epsilon$

Step 3, Clean up mixed RHS
$S^* \to S$
$S \to MN \mid T$
$M \to AS$
$N \to DO$
$O \to DD$
$T \to PO \mid R$
$P \to BT$
$R \to CR \mid \epsilon$
$A \to a$
$B \to b$
$C \to c$

Step 4, determine which nonterminal are "nullable" ($A \Rightarrow^* \epsilon$)
The only rule that goes to ϵ is R, so:
$S^* \to S$
$S \to MN \mid T$
$M \to AS$
$N \to DO$
$O \to DD$
$T \to PO \mid C$
$P \to BT$
$A \to a$
$B \to b$
$C \to c$

Step 5, for each rule $A \to B$, copy all $B \to \alpha$ rules to $A \to \alpha$. Repeat until no more changes, then delete $A \to B$
$S^* \to MN \mid T$
$M \to AS$
$N \to DO$
$O \to DD$
$T \to PO \mid c$
$P \to BT$
$A \to a$

$B \to b$
$C \to c$

3

a

$L = \{a^k b^m c^n \mid k, m, n not all equal\}$
We can solve this problem by building a CFG to show that it's context free:
$S \to\ T \mid U \mid V \mid W$
$T \to\ aTbC \mid aT \mid a$
$U \to\ aUBc \mid aU \mid a$
$V \to\ aVbC \mid Vb \mid b$
$W \to\ AbWc \mid Wb \mid b$
$A \to\ aA \mid a \mid \epsilon$
$B \to\ Bb \mid b \mid \epsilon$
$C \to\ Cc \mid c \mid \epsilon$
In the above rules, T covers the $a > b$ case, U covers the $a > b$ case, V covers the $b > a$ case, and W covers the $b > c$ case. We don't need to cover more cases than this, because with the above rules, one letter will always not be equal to the others.

b

We can show that this language is context free by building a pushdown automaton . The automaton below reads in a set of characters, and for every character read in before the 'c' character, we push to the stack. These are the characters in x. After the 'c' char, we read the characters in y. To do this, we read an arbitrary number of characters, then we read the $rev(x)$ substring and pop the stack until it's empty. Once it's empty, we read an arbitrary number of characters again. By using the stack, we can reverse the substring, giving us $rev(x)$.