Announcements

Pset 2 out

Midterm 1 returned,
Scores on BB

Pset 2 Q1: $\{ \underbrace{0, 1, \varepsilon, \emptyset}_{\text{"characters"}}, \underbrace{[,]}_{\text{"parentheses"}}, \underbrace{\cup, \circ, *}_{\text{"union" "concat" "star"}} \}$

λ

$w = [^{P}_{\varepsilon}]^{P}$

Strategy:

- Use R from the description
- What can go between individual characters?

$\cup$, or $\circ$

$\underbrace{0***}_{} \underbrace{\circ 1}_{} \cup \varepsilon \cup \emptyset$ ← make a regex R' for this

- What about brackets?

$[\text{---}]*** \cup 0 \circ 1 \text{---}$ ← then use R' to work with brackets

Midterm #1:

$L_1 = \{(011)^{n+1}(010)^{n} : n \geq 0\}$

Claim: L_1 is not reg.

Assume L_1 is reg.

$\Rightarrow \exists$ a p for L_1.

Choose $w = \underline{(011)^p}(010)^{p-1}$

$p+1 \qquad p$

$\cancel{x = (011)^a}$

$\cancel{y = (011)^b} \qquad b \geq 1$

$\cancel{z = (011)^{p-a-b}} (010)^{p-1}$

Consider, for any valid decomposition of w into xyz,

xy^0z. If y contains a 0, then the # of 011 substrings decreases by ≥ 1.

Same conclusion if y contains a 1.

So, in all cases, $xy^0z \notin L_1 \rightarrow L_1$ is not reg.

(b) L_2 = exactly 1 more 011 than 010 substrings

$$L_2 \cap \underbrace{(011)^*(010)^*}_{\text{reg}} = L_1.$$

$\uparrow$

if L_2 were reg, then L_1 must be regular by closure under intersection.

6) $\circ \xrightarrow{a} \circ$

then let $h(a) = h_1 h_2 \cdots h_k$

$$\circ \xrightarrow{h_1} \circ \xrightarrow{h_2} \circ \xrightarrow{h_3} \circ - - - \circ \xrightarrow{h_k} \circ$$

Regrades

Here are steps:

1. Review sample solns (when posted)
2. IF, after step 1, you still want a regrade, give the paper to me by Wednesday next week.
3. The paper will be regraded from top to bottom.
4. Whatever the new grade is, it is final.

CNF $\quad S \to \varepsilon$

$A \to a$

$A \to BC \quad (B, C \neq S)$

after step 3:

$$S_0 \to \varepsilon \mid (S) \mid SS \mid ()$$

$$S \to (S) \mid SS \mid ()$$

Step 4: Ensure the RHS of every rule is either ε (in case of S_0), a single terminal, or only variables.

$$S_0 \to \varepsilon \mid U_(S U_) \mid \underline{SS} \mid U_(U_)$$

$$S \to U_(S U_) \mid \underline{SS} \mid U_(U_)$$

$$U_(\to ($$

$$U_) \to)$$

Step 5: Break up "long" RHSes.

$$S_0 \to \varepsilon \mid \underline{Y_1 U_)} \mid \underline{SS} \mid U_(U_)$$

$$S \to Y_2 U_) \mid \underline{SS} \mid U_(U_)$$

$$U_(\to ($$

$$U_) \to)$$

$$Y_1 \to U_(S$$

$$Y_2 \to U_(S$$

Thm: Any CFG can be converted into an equivalent one in CNF.

For any string w of length $n > 0$, it can be derived by a CFG in CNF in exactly $2n-1$ rule applications.

Proof:

Let $w = w_1 w_2 \dots w_n$

$n-1$

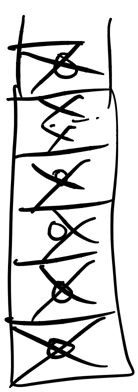
$$S \Rightarrow W_1 W_2 \Rightarrow W_1 W_2 W_3$$
$$\Rightarrow \dots \Rightarrow W_1 W_2 W_3 \cdots W_n$$

n

$$\Rightarrow w_1 W_2 W_3 \cdots W_n$$
$$\Rightarrow w_1 w_2 W_3 \cdots W_n$$
$$\Rightarrow \dots$$
$$\Rightarrow w_1 w_2 w_3 \cdots w_n$$

$\{0^n 1^n : n \geq 0\}$

$0 - - - - 0 1 - - - - 1$



push, pop

want: pop or not
and push or not

want: a "stack" alphabet.

pushdown stores

A pushdown automaton (PDA) is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$

where: Q, Σ, q_0, & F are the same,

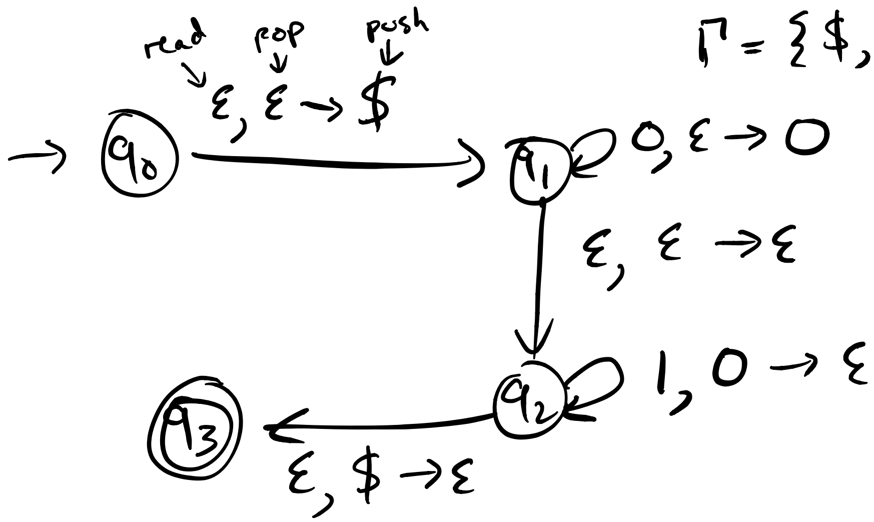
Γ = finite alphabet ("stack alphabet")

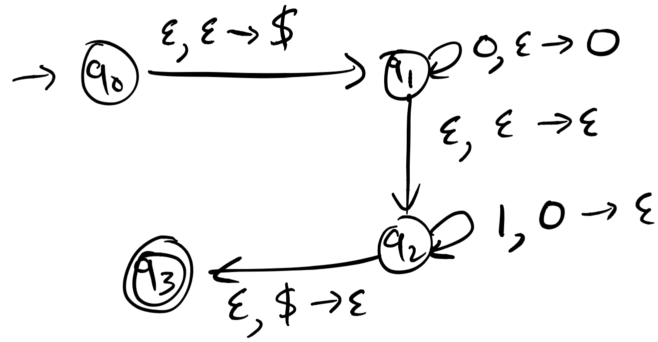
$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \overbrace{(\Gamma \cup \{\varepsilon\})}^{\text{pop}}$$

$$\to \mathcal{P}\Big(Q \times \underbrace{(\Gamma \cup \{\varepsilon\})}_{\text{push}}\Big)$$

Ex: Make a PDA for $\{0^n 1^n : \overset{n>0}{\cancel{n \geq 0}}\}$

$\Gamma = \{\$, 0, 1\}$





	$(0,0)$	$(0,1)$	$(0,\$)$	$(1,0)$	$(1,1)$	$(1,\$)$	$(\varepsilon,0)$	$(\varepsilon,1)$	$(\varepsilon,\$)$
q_0	$\emptyset$								
q_1									
q_2				$\{(q_2,\varepsilon)\}$					
q_3									

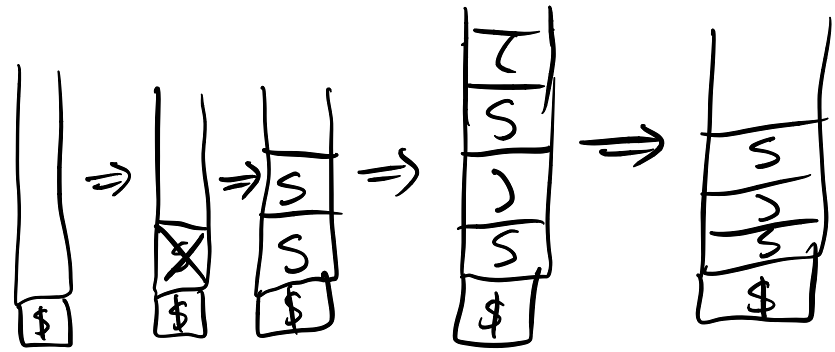
Thm: PDAs and CFGs are equivalent.

CFG $\Rightarrow$ PDA

$$S \to (S) \mid SS \mid \varepsilon$$

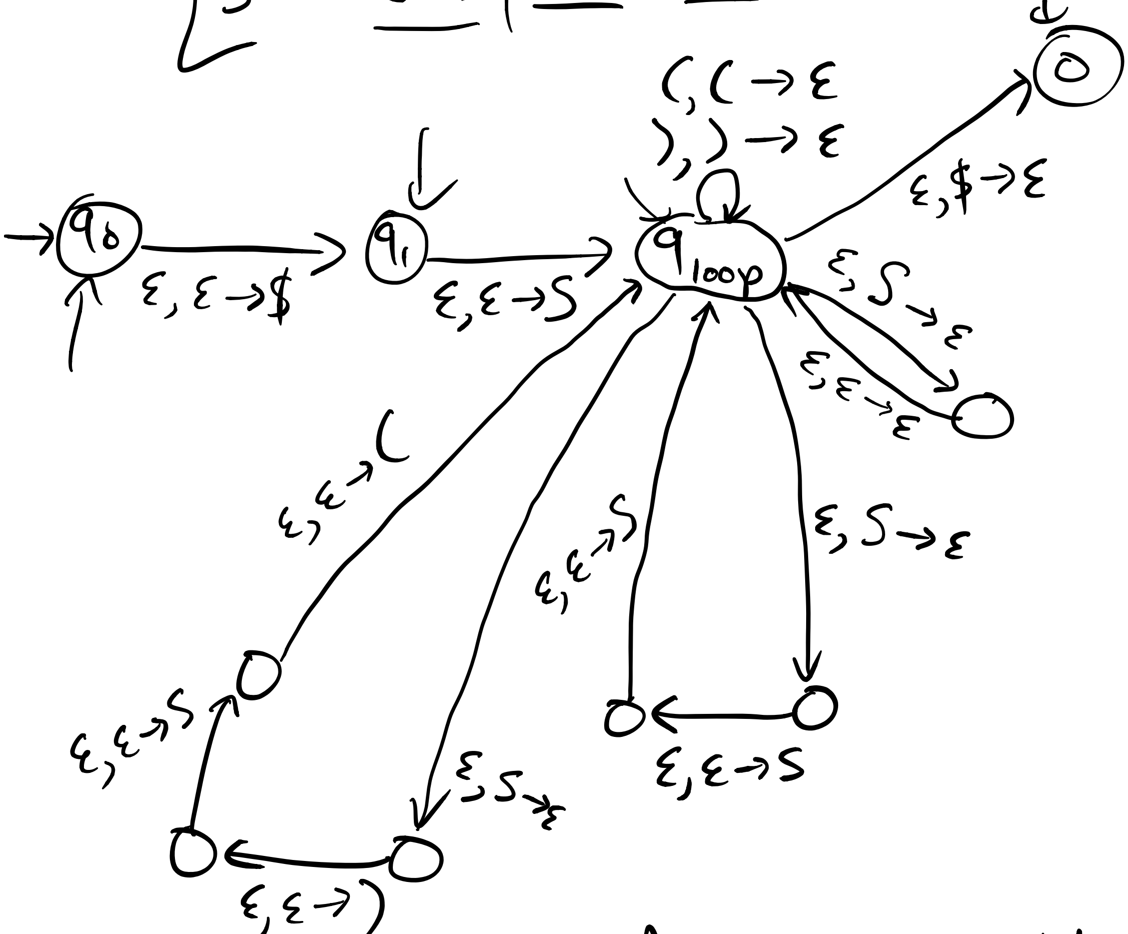
want to derive $(())()$

$$S \Rightarrow \boxed{SS} \Rightarrow (S)S \Rightarrow ((S))S \Rightarrow (())S \Rightarrow (())(S) \Rightarrow \boxed{(())()}$$



Ex:

$$S \to (S) \mid SS \mid \varepsilon$$



QED.

$$A \to w_1 w_2 \cdots w_n$$