Lecture 18

Homework #18: 3.4.1, Misc (see handout – prove by induction on the number of steps in the computation)

From last time: recall that we noted that PDAs accept exactly the CFLs. Gave some intuition for why the stack memory of the PDA is appropriate.

Today: we discuss more formally the equivalence of languages accepted by PDAs and generated by CFGs.

Recall

(1) a derivation is a leftmost derivation iff the nonterminal replaced at each step is the leftmost nonterminal symbol.

(2) For any CFG $G = (V, \Sigma, R, S)$ and any string $w \in \Sigma^*$,
$S \Rightarrow^* w$ iff $S \Rightarrow^{*L} w$.
i.e., any string that can be derived from a grammar has a leftmost derivation.

And now for our theorem

Thm. The class of languages accepted by PDAs is exactly the class of CFLs.

Lemma. Each CFL is accepted by some PDA.

Let $L = L(G)$ for some CFG $G = (V, \Sigma, R, S)$

Then there is a PDA M such that $L = L(M)$, where
$M = (K, \Sigma, \Gamma, \Delta, s, F)$

Intuition: we'll simulate leftmost derivations on the stack.

So $M = (K, \Sigma, \Gamma, \Delta, s, F)$, where

$K = \{p, q\}$

$s = p$

$F = \{q\}$

$\Gamma = V$

Δ contains:

$((p, e, e), (q, S))$	/* starts the derivation */
$((q, e, A), (q, x))$	/* simulates $A \rightarrow x$ */
$((q, a, a), (q, e))$	/* pops terminal symbols $a \in \Sigma$ off stack */

Perhaps best seen in an example.

$E \rightarrow E + E$	E = "expression"
$E \rightarrow E - E$	
$E \rightarrow T$	T = "term"
$T \rightarrow V$	V = "var"
$T \rightarrow C$	C = "const"
$V \rightarrow a$	
$V \rightarrow b$	
$C \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$	

So:
$K = \{p, q\}$
$\Sigma = \{a, b, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -\}$
$\Gamma = \Sigma \cup \{E, T, V, C\}$
$s = p$
$F = \{q\}$

Δ:	$((p, e, e), (q, E))$	1
	$((q, e, E), (q, E+E))$	2
	$((q, e, E), (q, E-E))$	3
	$((q, e, E), (q, T))$	4
	$((q, e, T), (q, V))$	5
	$((q, e, T), (q, C))$	6
	$((q, e, V), (q, a))$	7
	$((q, e, V), (q, b))$	8

((q, e, C), (q, 0)) 9 [also for 1-9 on stack]
((q, +, +), (q, e)) 10
((q, -, -), (q, e)) 11
etc for a, b, 0, ..., 9

State	Unread Input	Stack	Transition
p	a-b+1	e	-
q	a-b+1	E	1
q	a-b+1	E+E	2
q	a-b+1	E-E+E	3
q	a-b+1	T-E+E	4
q	a-b+1	V-E+E	5
q	a-b+1	a-E+E	
q	-b+1	-E+E	
q	b+1	E+E	
q	b+1	T+E	
q	b+1	V+E	
q	b+1	b+E	
q	+1	+E	
q	1	E	
q	1	T	
q	1	C	
q	1	1	
q	e	e	

Basically, it expands out from the start state, i.e., derives something "legal" on the stack and then checks whether the input matches.

Note: parsers work on this principle. (but why don't parsers work exactly this way?)

Now we need to show that the PDA we've constructed actually accomplishes what we want. i.e., that L(M) = L(G).

Claim. $S \Rightarrow^{L*} w\alpha$ iff $(q,w,S) \vdash^* (q,e,\alpha)$,

$\alpha \in (V\text{-}\Sigma)V^* \cup \{e\}$, $w \in \Sigma^*$.

(=>) Given $S \Rightarrow^{L*} w\alpha$, show $(q,w,S) \vdash^* (q,e,\alpha)$.

Proof by induction on the length of the derivation.

Basis. Derivation is of length 0.

> Then $w = e$, $\alpha = S$.
> So $(q, w, S) = (q, e, \alpha) \vdash^* (q,e,\alpha)$.

IH. Assume holds true for derivations of length n or less, $n \geq 0$.

Let $S = u_0 \Rightarrow^L u_1 \Rightarrow^L u_2 \ldots \Rightarrow^L u_n \Rightarrow^L u_{n+1} = w\alpha$.

Let A be the leftmost nonterminal of u_n.

> $u_n = xA\beta$ and
> $u_{n+1} = x\Gamma\beta$

where
> $x \in \Sigma^*$.
> $\Gamma, \beta \in V^*$
> $A \rightarrow \Gamma$.

By the IH,

> $(q,x,S) \vdash^* (q,e,A\beta)$

Because $A \rightarrow \Gamma$ is a rule in R, $((q,e,A),(q,\Gamma))$ is in Δ.

So $(q,x,S) \vdash^* (q,e,A\beta) \vdash (q,e,\Gamma\beta)$.

Now, note that $u_{n+1} = x\Gamma\beta = w\alpha$.

So there exists $y \in \Sigma^*$ such that $w = xy$ and $y\alpha = \Gamma\beta$.

So, we can write $(q, w, S) = (q, xy, S) \vdash^* (q,y,A\beta) \vdash (q,y,\Gamma\beta)$.

But since $y\alpha = \Gamma\beta$, $(q, y, \Gamma\beta) = (q, y, y\alpha) \vdash^* (q, e, \alpha)$.

So, $(q, w, S) \vdash^* (q, e, \alpha)$.

Won't do the other direction in class.