

11-711: Algorithms for NLP

Recitation #3

Greg Hanneman

October 3, 2007

Simulation of CFGs with PDAs

Given the following CFG G in Greibach Normal Form

$$\begin{aligned} S &\rightarrow aB \mid aSB \mid aBS \mid aSBS \\ B &\rightarrow b \end{aligned}$$

construct a PDA M that accepts by empty stack such that $L(G) = L(M)$. Then show step by step the computation of the machine and the corresponding derivation in the grammar for the input string $abaabbb$.

Solution

We saw in lecture a proof that, for a CFG G in Greibach Normal Form, there exists a PDA M such that $L(G) = L(M)$. In general, the construction says that, given $G = (V, T, P, S)$, we create $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ where

- $Q = \{q\}$
- $\Sigma = T$
- $\Gamma = V \cup T$
- $q_0 = q$
- $z_0 = S$
- $F = \emptyset$
- $\delta(q, \epsilon, A) = \{(q, \beta)\}$ for each $A \rightarrow \beta$, $A \in V$
- $\delta(q, a, a) = \{(q, \epsilon)\} \forall a \in T$

The definitions of the δ function show the two conceptual operations that this machine will perform. First, it can “apply” a rule of the form $A \rightarrow \beta$ by replacing the left-hand side of the rule (A) on the top of its stack with the right-hand side of the rule (β). Second, it can “read” the next letter a from the input string when an a is on top of the stack.

Following the construction given above, the elements of M for the grammar given is this problem are:

- $Q = \{q\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{S, B, a, b\}$
- $q_0 = q$
- $z_0 = S$
- $F = \emptyset$
- $\delta(q, \epsilon, S) = \{(q, aB), (q, aSB), (q, aBS), (q, aSBS)\}$
- $\delta(q, \epsilon, B) = \{(q, b)\}$
- $\delta(q, a, a) = \{(q, \epsilon)\}$
- $\delta(q, b, b) = \{(q, \epsilon)\}$

The initial configuration of this one-state PDA for an input word w is (q, w, S) . It accepts w when it has read the entire string and has an empty stack — that is, when the configuration is (q, ϵ, ϵ) .

We now simulate the computation and derivation of the string $aabaabbb$. Remember that PDAs are non-deterministic machines, which means that we can “guess” the correct transition or rule to apply at each step. In the table below, we will show sentential forms in G along with the corresponding ID in M as M simulates the derivation in G . Based on the way we constructed M , the machine simulates a leftmost derivation of the string.

Grammar	PDA	δ Applied
S	$(q, aabaabbb, S)$	
aSB	$(q, aabaabbb, aSB)$	$\delta(q, \epsilon, S) = (q, aSB)$
	$(q, abaabbb, SB)$	$\delta(q, a, a) = (q, \epsilon)$
$aaBSB$	$(q, abaabbb, aBSB)$	$\delta(q, \epsilon, S) = (q, aBS)$
	$(q, baabbb, BSB)$	$\delta(q, a, a) = (q, \epsilon)$
$aabSB$	$(q, baabbb, bSB)$	$\delta(q, \epsilon, B) = (q, b)$
	$(q, aabbb, SB)$	$\delta(q, b, b) = (q, \epsilon)$
$aabaSBB$	$(q, aabbb, aSBB)$	$\delta(q, \epsilon, S) = (q, aSB)$
	$(q, abbb, SBB)$	$\delta(q, a, a) = (q, \epsilon)$
$aabaaBBB$	$(q, abbb, aBBB)$	$\delta(q, \epsilon, S) = (q, aB)$
	(q, bbb, BBB)	$\delta(q, a, a) = (q, \epsilon)$
$aabaabBB$	(q, bbb, bBB)	$\delta(q, \epsilon, B) = (q, b)$
	(q, bb, BB)	$\delta(q, b, b) = (q, \epsilon)$
$aabaabbB$	(q, bb, bB)	$\delta(q, \epsilon, B) = (q, b)$
	(q, b, B)	$\delta(q, b, b) = (q, \epsilon)$
$aabaabbb$	(q, b, b)	$\delta(q, \epsilon, B) = (q, b)$
	(q, ϵ, ϵ)	$\delta(q, b, b) = (q, \epsilon)$

The PDA has an empty stack after consuming the entire input word, so we have shown that M accepts $aabaabbb$.