Lecture 20, 21 COL 352 Feb 25, 27

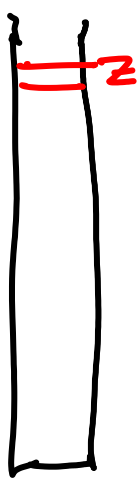
A stack based machine (Push Down Automaton)

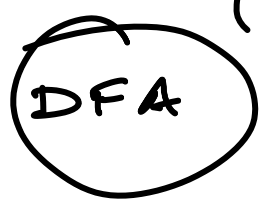
In addition to a finite state transition system, we have an (infinite capacity) stack.

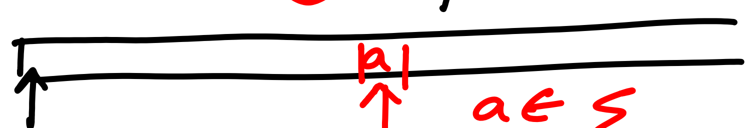
stack alphabet

$$(Q, \Sigma, \delta, \overset{\phi}{F}, q_0, \Gamma, Z_0)$$

bottom stack marker







$a \in \Sigma$

$\delta(q, a, z) = (p, \gamma)$

$z \in \Gamma$

$p, q \in Q$

$\gamma \in \Gamma^*$

ϵ is used for popping stack

Two separate terminating conditions

① The stack is empty when input string is exhausted

② We are in a final state when input is exhausted

$$L = \{ wcw^R \mid w \in (0+1)^* \}$$

$$M = (\{q_1, q_2\}, \{0, 1, c\} \{R, B, G\}, \delta, q_1, R, \phi)$$

1. $\delta(q_1, 0, R) = \{(q_1, BR), \_\}$ $\quad \delta(q_1, 1, R) = (q_1, GR)$
2. $\delta(q_1, 0, B) = (q_1, BB)$ $\quad \delta(q_1, 1, B) = (q_1, GB)$
3. $\delta(q_1, 0, G) = (q_1, BG)$ $\quad \delta(q_1, 1, G) = (q_1, GG)$
4. $\delta(q_2, 0, B) = (q_2, \epsilon)$ $\quad \delta(q_2, 1, G) = (q_2, \epsilon)$
5. $\delta(q_2, \epsilon, R) = (q_2, \epsilon)$

6 $\delta(q_1, c, R) = (q_2, R)$ $\quad \delta(q_1, c, G) = (q_2, G)$

7 $\delta(q_1, c, B) = (q_2, B)$

Instantaneous Description (ID) of a PDA

The complete information about a PDA can be obtained from

1. Current state
(2) the current symbol it is scanning
(3) the stack contents

ID: $(q,\ a\cdot\omega,\ \alpha)$ $\quad q \in Q$, $a \in \Sigma, \omega \in \Sigma^*$, $\alpha \in \Gamma^*$

input string

$I_0 : (q_0, \delta, Z_0)$

$I_0 \vdash I_1 \vdash I_2 \vdash \cdots \quad I_f$

$I_0 \vdash^* I_f$

$I_j \vdash_M I_{j+1}$

$(p, a\omega, A\alpha) \vdash (q, \omega, \beta\alpha)$

$\delta(p, a, A)$ must contain (q, β)

PDAs that accept by empty stack

w is accepted by the PDA iff

$$(q_0, w, Z_0) \vdash^{*} (p, \epsilon, \epsilon) \qquad p \in Q$$

PDAs that accept by final state

$$(q_0, w, Z_0) \vdash^{*} (q_f, \epsilon, \alpha)$$

$q_f \in F$ $\alpha \in \Gamma^*$

Thm Let L be a language accepted by a PDA using empty stack. Then L is also accepted by some PDA that accepts using final state.

and vice versa

Thm Suppose L is a CFL generated by a CFG $G = (V, T, S, P)$. Then we can design a PDA M (accepts using empty stack) s.t.

$$L(M) = L$$

The proof uses Greibach Normal Form

Thm Given a PDA M that accepts a language L. Then we can design a CFG G s.t.

$$L(G) = L$$

We will provide a construction of a PDA M given CFG $G = (S, V, T, P)$ given in Greibach Normal Form

$Q = \{q\}$ only one state

$q_0 = q$

$\Gamma = V \qquad Z_0 = S$ (bottom stack)

$\delta(q, a, A)$ contains (q, α) if $A \to a\alpha \in P$

$\alpha \in V^*$ (including ϵ)

The idea is to simulate a leftmost derivation of the grammar given in GNF

$$S \xrightarrow{*} x\alpha \quad \text{iff} \quad (q, x, S) \vdash^{*} (q, \epsilon, \alpha)$$

where $x \in T^+$ $\alpha \in V^*$

The formal proof will be using induction on $\xrightarrow{i}$

Let us consider a running example

$$L = \{ w = w^R \mid w \in (a+b)^* \}$$, ie. palindromes over $\{a, b\}$

CFG

$$S \to aSS_a \mid bSS_b \mid aS_a \mid bS_b \mid a \mid b$$

$$S_a \to a \qquad S_b \to b \qquad \text{in GNF}$$

(It may be easier to compare with a more intuitive non GNF grammar

$$S \to aSa \mid bSb \mid aa \mid bb \mid a \mid b$$

So the PDA would have transition function as follows

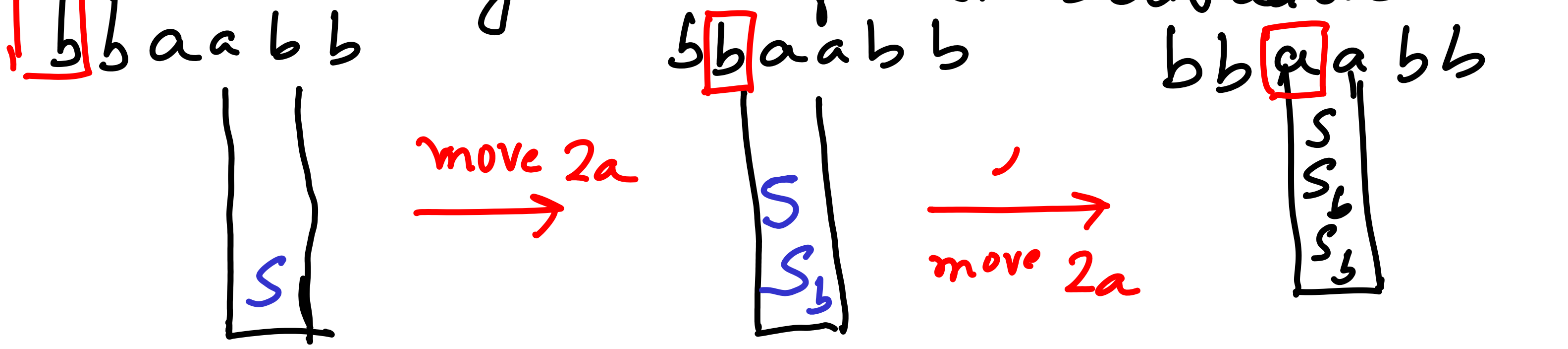
				1a	1b	1c
1	$\delta(q, a, S)$	contains	$\{$	$(q, S S_a)$,	$(q S_a)$,	$(q, \epsilon)\}$
				2a	2b	2c
2	$\delta(q, b, S)$	"	$\{$	(q, S, S_b),	(q, S_b),	$(q, \epsilon)\}$
3	$\delta(q, a, S_a)$	"		(q, ϵ) 3a ← move #		
4	$\delta(q, b, S_b)$	"		(q, ϵ) 4a		

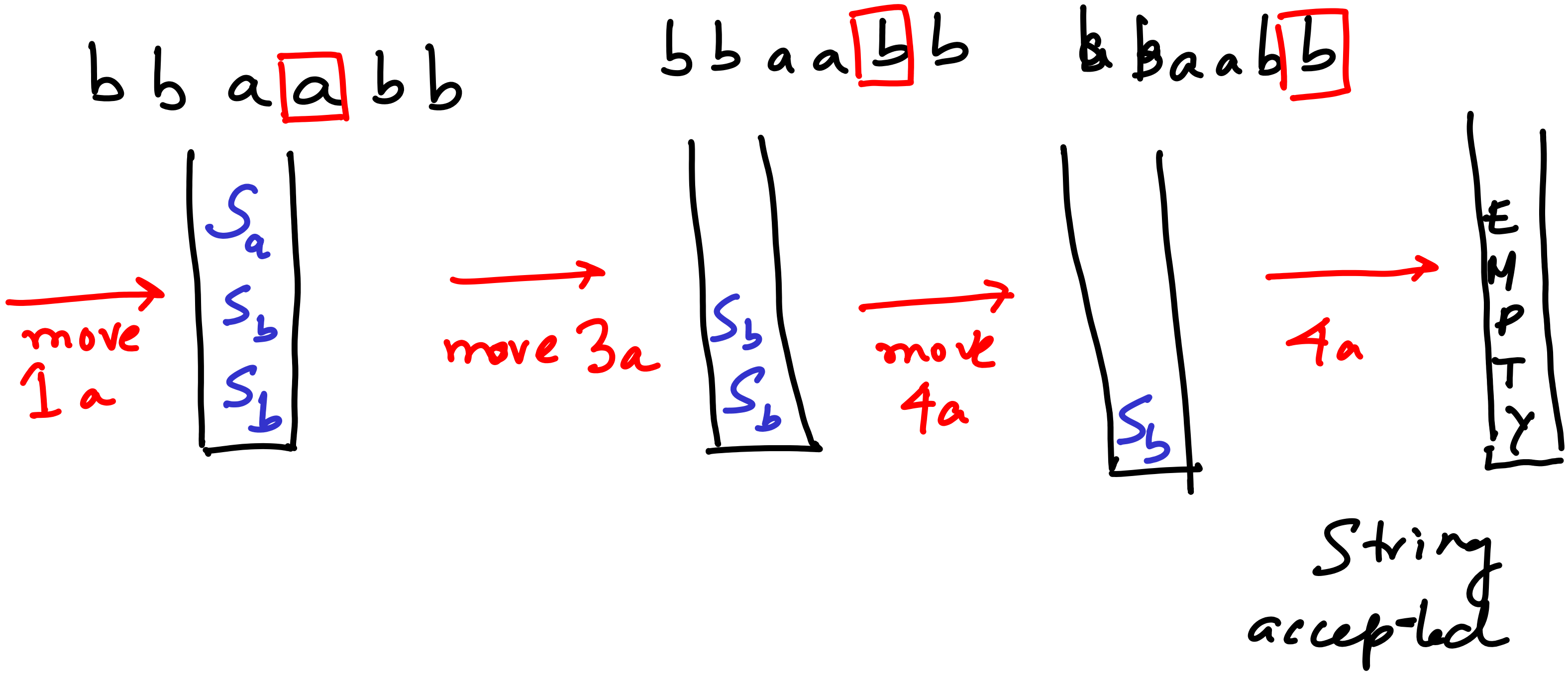
Consider the derivation of bbaabb which is a palindrome

$$S \to b S S_b \to b b S S_b S_b \to b b a S_a S_b S_b$$

$$\to b b a a S_b S_b \to bbaab S_b \to bbaabb$$

Here is how the machine accepts by mimicking the left most derivation





Verify if the string can be accepted using any alternate moves of the machine. The machine crashes (doesn't accept) if there is no well defined next move