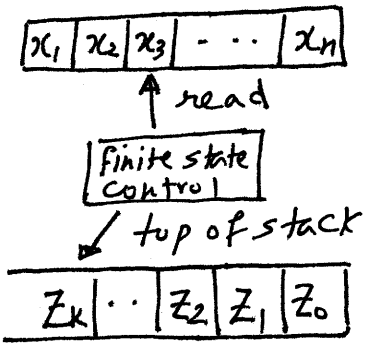
Pushdown Automata (p.d.a.)

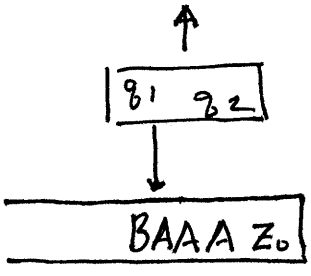
What type of "machine" can recognize a c.f.l. such as $L = \{ wcw^R : w \in \{a,b\}^+ \}$

Note the following c.f.g.: $S \to aSa \mid bSb \mid c$

Informal:



Recognizing wcw^R; example string a a a b b c b b a a a



This seems o.k. change state when "c" is encountered

How about recognizing WW^R a a a b b b b a a a
Guess the middle! Let automata make a non-deterministic move!

How to accept: (1) empty stack (2) final state
Why won't we recognize a a b b b ?
If "wrong" move made, either input remains or stack symbols remain or not in final state

A p.d.a. M is a 7-tuple $\langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$

where:

1. Q is a finite set of states
2. Σ is a finite set of inputs
3. Γ is a finite set of stack symbols
4. $q_0 \in Q$ is the initial state
5. $Z_0 \in \Gamma$ is the initial stack symbol
6. $F \subseteq Q$ is the set of final states
7. δ is the transition function

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \to 2^{Q \times \Gamma^*} \quad (\text{finite subset})$$

Example $\delta(q_1, a, A) = \{(q_1, AA), (q_2, \lambda)\}$

Example: p.d.a. accepting WW^R by empty stack.

$$M = \langle \{q_1, q_2\}, \{a, b\}, \{A, B, Z\}, \delta, q_1, Z, \Phi \rangle$$

$\delta(q_1, a, Z) = \{(q_1, AZ)\}$ "stack A" (or push A)

$\delta(q_1, b, Z) = \{(q_1, BZ)\}$ "stack B"

$\delta(q_1, a, A) = \{(q_1, AA), (q_2, \lambda)\}$ "stack A" or "match & pop A"

$\delta(q_1, b, B) = \{(q_1, BB), (q_2, \lambda)\}$ "stack B" or "match & pop B"

$\delta(q_1, b, A) = \{(q_1, BA)\}$ "stack B"

$\delta(q_1, a, B) = \{(q_1, AB)\}$ "stack A"

$\delta(q_2, a, A) = \{(q_2, \lambda)\}$ "match & pop A"

$\delta(q_2, b, B) = \{(q_2, \lambda)\}$ "match & pop B"

$\delta(q_2, \lambda, Z) = \{(q_2, \lambda)\}$ "pop initial stack symbol"

$\delta(q_1, \lambda, Z) = \{(q_1, \lambda)\}$ "pop Z to recognize λ"

Instantaneous Descriptions (IDs)

A triple (q, w, γ) (state, remaining input, stack)

$(q, aw, Z\alpha) \vdash (q', w, \beta\alpha)$
if $(q', \beta) \in \delta(q, a, Z)$

$\vdash$ is called "turnstile" and is the "derives" notion for pda.
$\vdash^*$ derives in 0 or more steps
$\vdash^i$ derives in i steps

Given a p.d.a $\langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$ we consider two notions of accepting strings:

1. Accepting by final state: $L(M)$, language accepted by final state:
$\{w \mid (q_0, w, Z_0) \vdash^* (q_f, \lambda, \gamma)$ for $q_f \in F$ and $\gamma \in \Gamma^*\}$

2. Accepting by null or empty stack: $N(M)$, language accepted by null stack:
$\{w \mid (q_0, w, Z_0) \vdash^* (q, \lambda, \lambda)$ for $q \in Q\}$

Note that p.d.a. means a nondeterministic pda or npda. We can also define a dpda or deterministic p.d.a. by insisting on only one move (at most) from any I.D.
Thus a dpda is a pda such that:

(a) for $q \in Q, Z \in \Gamma$, if $\delta(q, \lambda, Z)$ is non-empty (ie. λ moves are possible) then $\delta(q, a, Z)$ is empty (no move using up an input is possible).

(b) $\delta(q, a, Z)$ contains at most <u>one</u> element, for $q \in Q, Z \in \Gamma, a \in \Sigma \cup \{\lambda\}$

① Theorem: acceptance by null stack is equivalent to acceptance by final state.

② Theorem: a language L is a c.f.l. $\Leftrightarrow L = N(M)$ for some pda M

Theorem ① ($\Leftarrow$) Suppose $L = L(M_2)$ for some M_2 accepting by final state. Then $L = N(M_1)$ for some M_1 accepting by null stack.

Given $M_2 = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$,

define $M_1 = \langle Q \cup \{q_e, q_0'\}, \Sigma, \Gamma \cup \{X_0\}, \delta', q_0', X_0, \Phi \rangle$

with:

$\delta'(q_0', \lambda, X_0) = \{(q_0, Z_0X_0)\}$

$\delta'(q, a, Z)$ contains $\delta(q, a, Z)$ for all $q \in Q$, $Z \in \Gamma$, $a \in \Sigma \cup \{\lambda\}$

$\delta'(q, \lambda, Z)$ contains (q_e, λ) for $q \in F$, $Z \in \Gamma \cup \{X_0\}$

$\delta'(q_e, \lambda, Z)$ contains (q_e, λ) for $Z \in \Gamma \cup \{X_0\}$

Note that:

1. Note that M_1 simulates M_2 except that it has the option of "emptying the stack" whenever M_2 would have entered a final state.

2. M_1 has its own initial stack symbol to avoid accepting if M_2 would have emptied the stack without going to a final state.

Proof.

Let $x \in L(M_2)$. $\therefore (q_0, x, Z_0) \overset{*}{\vdash}_{M_2} (q_f, \lambda, \gamma)$

(note that stack never becomes empty except possibly at last move. Why?)

Now, $(q_0', x, X_0) \vdash_{M_1} (q_0, x, Z_0 X_0) \overset{*}{\vdash}_{M_1} (q_f, \lambda, \gamma X_0)$

$\overset{*}{\vdash}_{M_1} (q_e, \lambda, \lambda)$

$\therefore x \in N(M_1)$.

Conversely, suppose $x \in N(M_1)$. $\therefore$ the moves of M_1 must be:

$(q_0', x, X_0) \vdash_{M_1} (q_0, x, Z_0 X_0) \overset{*}{\vdash}_{M_1} (q_e, \lambda, \lambda)$.

Now, the only way that M_1 can get to q_e is that some intermediate ID, K, must be $(q, \lambda, \gamma X_0)$ for $q \in F$. Also, the input must be λ, since there is no other way to use up the input. Thus, $x \in L(M_2)$ modifying 2nd ID to K^{th} ID.

Theorem ① ($\Rightarrow$). Suppose $L = N(M_1)$ for some M_1 accepted by null stack. Then $L = L(M_2)$ for some M_2 accepting by final state.

Proof

The idea is to have M_2 simulate M_1 and detect when M_1 empties the stack and then have M_2 go to a final state.

Let $M_1 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \phi)$ accepting by null stack.

Define $M_2 = (Q \cup \{q_0', q_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta', q_0', X_0, \{q_f\})$

$$\delta'(q_0', \lambda, X_0) = \{(q_0, Z_0 X_0)\}$$
$$\delta'(q, a, Z) = \delta(q, a, Z) \text{ for } q \in Q, a \in \Sigma \cup \{\lambda\}, Z \in \Gamma.$$
$$\delta'(q, \lambda, X_0) = (q_f, \lambda).$$

It is straightforward as previously to show that

$x \in N(M_1) \iff x \in L(M_2)$.

Theorem 2. ($\Rightarrow$). If L is a context-free language (cfl), then there exists a pda M s.t. $L = N(M)$.

Idea If $\lambda \notin L$, start with Greibach Normal Form, and simulate a left-most derivation by putting the non-terminals in the production on a stack.

Example:

$S \to a A_1 A_2 A_3$

$A_1 \to b A_1 A_3$

$A_1 \to a A_2$

$A_2 \to b$

$A_3 \to a$

Consider derivation:

$S \Rightarrow a A_1 A_2 A_3 \Rightarrow a b A_1 A_3 A_2 A_3 \Rightarrow ab a A_2 A_3 A_2 A_3$

$\Rightarrow abab A_3 A_2 A_3 \Rightarrow ababa A_2 A_3 \Rightarrow ababab A_3$

$\Rightarrow abababa$

Input tape	abababa	S	(stack initialization)
read	a	$A_1 A_2 A_3$	(replace S by $A_1 A_2 A_3$)
read	b	$A_1 A_3 A_2 A_3$	(replace A_1 by $A_1 A_3$)
read	a	$A_2 A_3 A_2 A_3$	(replace A_1 by A_2)
read	b	$A_3 A_2 A_3$	(pop A_2)
read	a	$A_2 A_3$	(pop A_3)
read	b	A_3	(pop A_2)
read	a	ϕ	(pop A_3)

Stack is empty, string is accepted. We can in general define a pda given a grammar in G.N.F.

Proof

Let $G = (V, T, P, S)$ be a c.f.g. in Greibach normal form.

Consider

$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle = \langle \{q\}, T, V, \delta, q, S, \phi \rangle$

where $\delta(q, a, A)$ contains (q, γ) whenever $A \to a\gamma$ is in P.

Claim: $S \overset{*}{\Rightarrow} x\alpha$ by leftmost derivations, $x \in T^*$, $\alpha \in V^*$

$\iff$ $(q, x, S) \overset{*}{\vdash} (q, \lambda, \alpha)$

$\Leftarrow$ Suppose $(q, x, S) \overset{i}{\vdash} (q, \lambda, \alpha)$ then $S \overset{i}{\Rightarrow} x\alpha$

$i = 0$, $x = \lambda$, $S = \alpha$ but $S \overset{0}{\Rightarrow} S$ ✓.

$(q, x, S) \overset{i-1}{\vdash} (q, a, \beta) \vdash (q, \lambda, \alpha)$ $\qquad x = ya$

$\therefore (q, y, S) \overset{i-1}{\vdash} (q, \lambda, \beta)$

by induction $S \overset{*}{\Rightarrow} y\beta$

also $(q, a, \beta) \vdash (q, \lambda, \alpha)$ is a move, ie.,

$\beta = A\beta'$ and $A \to a\gamma$ in P with $\gamma\beta' = \alpha$.

$\therefore S \overset{*}{\Rightarrow} y\beta = yA\beta' \Rightarrow ya\gamma\beta' = x\alpha$ ✓

$\Rightarrow$ Suppose $S \overset{i}{\Rightarrow} x\alpha$ by a leftmost derivation.

$i = 0$, $S \overset{0}{\Rightarrow} S$ $\therefore (q, \lambda, S) \overset{0}{\vdash} (q, \lambda, S)$

$S \overset{i}{\Rightarrow} ya\alpha$ $\qquad (x = ya)$

$S \overset{i-1}{\Rightarrow} yA\alpha' \Rightarrow ya\gamma\alpha' = x\alpha$ {note: $ya = x$; $\gamma\alpha' = \alpha$ and $A \to a\gamma$ is a production.

By induction

$(q, y, S) \overset{*}{\vdash} (q, \lambda, A\alpha')$

also $(q, a, A) \vdash (q, \lambda, \gamma)$

$\therefore (q, x, S) = (q, ya, S) \overset{*}{\vdash} (q, a, A\alpha') \vdash (q, \lambda, \gamma\alpha') = (q, \lambda, \alpha)$

Done!

Note handling $S \to \lambda$ is easy as an optional initial move.

Example (to be used to explain Thm 2, part 2).

Let $L = \{a^n b^m a^n \mid n, m \geq 1\} \cup \{a^n b^m c^m \mid n, m \geq 1\}$

Find a p.d.a that accepts L.

Idea of p.d.a. Stack a's & b's. If next terminal is an a, then pop b' and match a's. If next is a c, then match b's & pop c's.

$M = (\{q_0, q_1, q_2\}, \{a, b, c\}, \{Z_0, A, B\}, \delta, q_0, Z_0, \phi)$

1. $\delta(q_0, a, Z_0) = \{(q_0, AZ_0)\}$
2. $\delta(q_0, a, A) = \{(q_0, AA)\}$
3. $\delta(q_0, b, A) = \{(q_0, B)\}$
4. $\delta(q_0, b, B) = \{(q_0, BB)\}$] used up 1 a.
5. $\delta(q_0, a, B) = \{(q_1, \lambda)\}$
6. $\delta(q_1, \lambda, B) = \{(q_1, \lambda)\}$
7. $\delta(q_1, a, A) = \{(q_1, \lambda)\}$
8. $\delta(q_1, \lambda, Z_0) = \{(q_2, \lambda)\}$
9. $\delta(q_0, c, B) = \{(q_2, \lambda)\}$
10. $\delta(q_2, c, B) = \{(q_2, \lambda)\}$
11. $\delta(q_2, \lambda, A) = \{(q_2, \lambda)\}$
12. $\delta(q_2, \lambda, Z_0) = \{(q_2, \lambda)\}$

Try examples aabbbaa

aabbbcc

Theorem 2 (cont)

If $L = N(M)$ for some p.d.a M then L is a c.f.l

idea: During any stage of the computation of the p.d.a, the stack will vary in the symbols on the stack. Let the variables of the grammar encode these stack variables.

Proof:

Let M be a p.d.a $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, \Phi)$

Let G be the grammar (V, Σ, P, S) where the set V is variables

$$[q, A, p] \quad q, p \in Q, A \in \Gamma$$

Also, S is a new nonterminal.

Note: the "variable" $[q, A, p]$ is such that

$$[q, A, p] \overset{*}{\Rightarrow} w \iff (q, w, A) \overset{*}{\vdash_M} (p, \lambda, \lambda).$$

Productions are:

(1) $S \to [q_0, Z_0, q]$ for all $q \in Q$.

(2) $[q_i, A, q_j] \to a$ if $\delta(q_i, a, A)$ contains (q_j, λ)

(3) $[q, A, q_{m+1}] \to a\,[q_1, B_1, q_2]\,[q_2, B_2, q_3] \cdots [q_m, B_m, q_{m+1}]$
for each $(q, q_1, \ldots, q_{m+1}) \in Q$'s such that
$\delta(q, a, A)$ contains $(q_1, B_1 \cdots B_m)$.

Productions related to previous example.

① $[q_0, Z_0, q_2] \rightarrow a[q_0, A, q_1][q_1, Z_0, q_2]$
$\rightarrow a[q_0, A, q_0][q_0, Z_0, q_2]$
$\rightarrow a[q_0, A, q_2][q_2, Z_0, q_2]$

$[q_0, Z_0, q_1] \rightarrow$
⋮

$[q_0, Z_0, q_0] \rightarrow$
⋮

(Note must are useless variables.)

④ $[q_0, B, q_0] \rightarrow b[q_0, B, q_0][q_0, B, q_0]$
⋮

$[q_0, B, q_1] \rightarrow b[q_0, B, q_1][q_1, B, q_1]$
⋮

$[q_0, B, q_2] \rightarrow b[q_0, B, q_2][q_2, B, q_2]$

⑤ $[q_0, B, q_1] \rightarrow a$

⑥ $[q_1, B, q_1] \rightarrow \lambda$

⑦ $[q_1, A, q_1] \rightarrow a$

⑧ $[q_1, Z_0, q_2] \rightarrow \lambda$

⑨ $[q_0, B, q_2] \rightarrow c$

⑩ $[q_2, B, q_2] \rightarrow c$

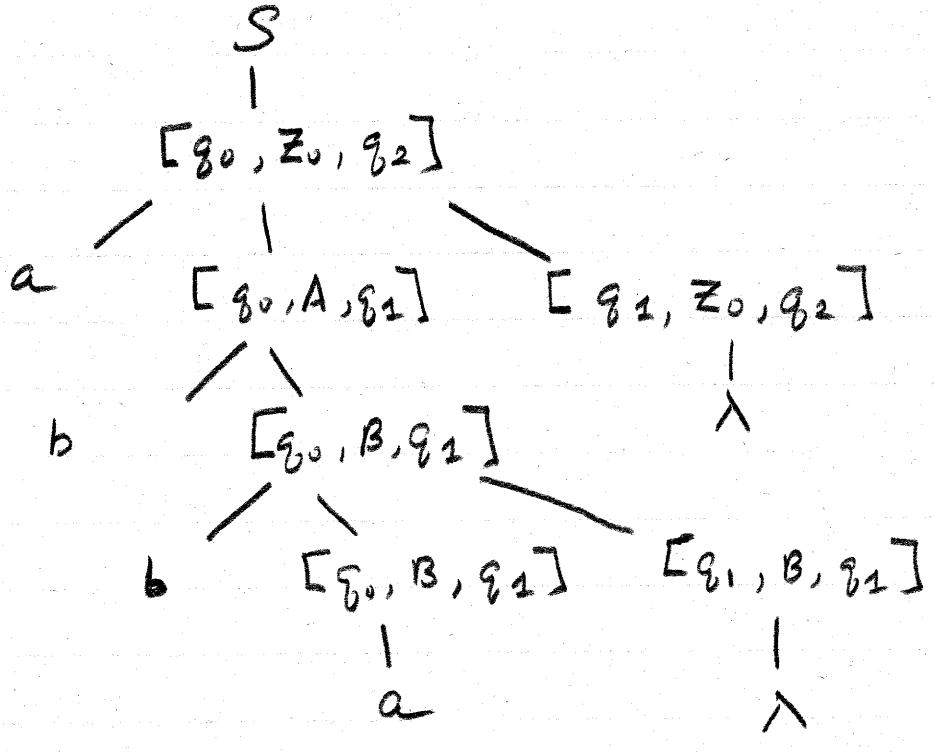
⑪ $[q_2, A, q_2] \rightarrow \lambda$

⑫ $[q_2, Z_0, q_2] \rightarrow \lambda$

Consider:

$$\begin{aligned}(q_0, abba, Z_0) &\vdash (q_0, bba, AZ_0)\\ &\vdash (q_0, ba, BZ_0)\\ &\vdash (q_0, a, BBZ_0)\\ &\vdash (q_1, \lambda, BZ_0)\\ &\vdash (q_1, \lambda, Z_0)\\ &\vdash (q_2, \lambda, \lambda)\end{aligned}$$

Follow left most derivation of associated grammar.



It can be proven using the definition that

$$[q, A, p] \overset{*}{\underset{G}{\Rightarrow}} w \quad \text{iff} \quad (q, w, A) \overset{*}{\underset{M}{\vdash}} (p, \lambda, \lambda).$$

$$\therefore \quad S \overset{*}{\underset{G}{\Rightarrow}} w \quad \text{iff} \quad (q_0, w, Z_0) \overset{*}{\underset{M}{\vdash}} (p, \lambda, \lambda)$$