EQUIVALENCE OF PDA

Dr. A. Beulah
AP/CSE

LEARNING OBJECTIVE

- To Design pushdown automata for any CFL (K3)
  - To Understand the equivalence between pushdown automata and CFL

CONVERSION OF CFG TO PDA

Dr. A. Beulah

AP/CSE

LEARNING OBJECTIVE

- To Design pushdown automata for any CFL (K3)
  - To convert a CFG to a PDA

FROM CFG TO PDA

- For any context free language L, there exists a PDA M such that L = L(M)

N

- Let G = (V, T, P, S) be a grammar. Then we can construct PDA M = (Q, Σ, Γ, δ, q_0, z, F) which simulates left most derivations in this grammar.

L

FROM CFG TO PDA

- Convert CFG $G=(V, T, P, S)$

  To PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$,

  where

  – $Q = \{q_0, q_1, q_f\}$ = set of states

  – Σ = terminals of grammar G

  – $\Gamma = V \cup T$ where V is the variables in grammar G

  – $F = \{q_f\}$ = final state

FROM CFG TO PDA

- The transition function will include

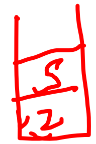
i) Push start Symbol

$Q = \{q_0, q_1, q_2\}$

$\Sigma = \{T$

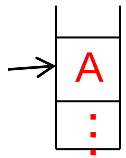
$\Gamma = T \cup N \cup \{z\}$

$q_0 \rightarrow \quad z \rightarrow \quad \delta$



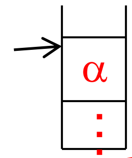
$$\delta(q_0, \varepsilon, z) = \{(q_1, Sz)\}$$

Before: ii) $A \rightarrow \alpha$ NT → Grammar



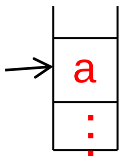
$$\delta(q_1, \varepsilon, A) = \{(q_1, \alpha)\} \text{ for each } A \rightarrow \alpha \text{ in } P$$

After:



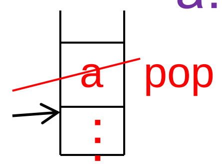
$F = \{q_2\}$

Before: iii) ∀ terminals do the pop



$$\delta(q_1, a, a) = \{(q_1, \varepsilon)\} \text{ for each } a \in T$$

After: a…



iv) Accepting reaching the FS

$$\delta(q_1, \varepsilon, z) = \{(q_2, z)\} \text{ reach final state}$$

$q \in F$

EXAMPLE

Transition diagram

- $S \rightarrow aSA \mid a$
- $A \rightarrow bB$
- $B \rightarrow b$

CFG → PDA

i) push Start symbol to the stack 'S'

$\delta(q_0, \varepsilon, Z) = \{(q_1, SZ)\}$

(ii) ∀ N ∋ $A \rightarrow \alpha$

S, A, B

$\delta(q_1, \varepsilon, S) = \{(q_1, aSA), (q_1, a)\}$

$\delta(q_1, \varepsilon, A) = \{(q_1, bB)\}$

$\delta(q_1, \varepsilon, B) = \{(q_1, b)\}$

(iii) ∀ T i/p, top stack pop

a, b

$\delta(q_1, a, a) = \{(q_1, \varepsilon)\}$

$\delta(q_1, b, b) = \{(q_1, \varepsilon)\}$

(iv) Reaching the final state

$\delta(q_1, \varepsilon, Z) = \{(q_2, Z)\}$

$Q = \{q_0, q_1, q_2\}$

$\Sigma = \{a, b\}$

$\Gamma = \{S, A, B, a, b, Z\}$

q_0 δ

Z $F = \{q_2\}$

aabb

$S \overset{lm}{\Rightarrow} aSA \overset{lm}{\Rightarrow} aaA \overset{lm}{\Rightarrow} aabB \overset{lm}{\Rightarrow} aabb$

$(q_0, aabb, Z) \vdash (q_1, aabb, SZ)$
$\vdash (q_1, aabb, aSAZ) \vdash (q_1, abb, SAZ)$
$\vdash (q_1, abb, aAZ) \vdash$
$(q_1, bb, AZ) \vdash (q_1, bb, bBZ)$
$\vdash (q_1, b, BZ) \vdash (q_1, b, bZ)$
$\vdash (q_1, \varepsilon, Z) \vdash (q_2, \varepsilon, Z)$

$q_2 \in F$ ∴ Accepted

EXAMPLE

- $S \rightarrow aSa \mid bSb \mid c$

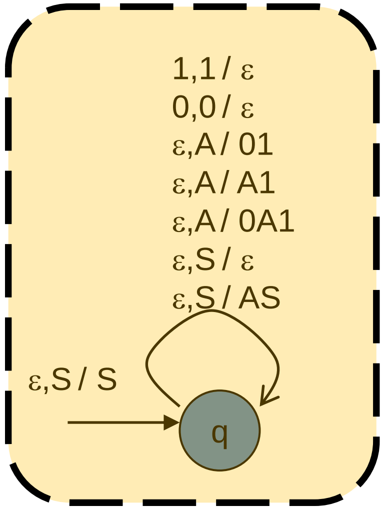
EXAMPLE

- $S \rightarrow 0S1 \mid A$
- $A \rightarrow 1A0 \mid S \mid \varepsilon$

- $A \rightarrow aABC \mid bB \mid a$
- $B \rightarrow b$
- $C \rightarrow c$

EXAMPLE: CFG TO PDA

- $G = (\{S,A\}, \{0,1\}, P, S)$
- P:
  - $S \rightarrow AS \mid \varepsilon$
  - $A \rightarrow 0A1 \mid A1 \mid 01$
- $PDA = (\{q\}, \{0,1\}, \{0,1,A,S\}, \delta, q, S)$
- δ:
  - $\delta(q, \varepsilon, S) = \{ (q, AS), (q, \varepsilon)\}$
  - $\delta(q, \varepsilon, A) = \{ (q,0A1), (q,A1), (q,01) \}$
  - $\delta(q, 0, 0) = \{ (q, \varepsilon) \}$
  - $\delta(q, 1, 1) = \{ (q, \varepsilon) \}$



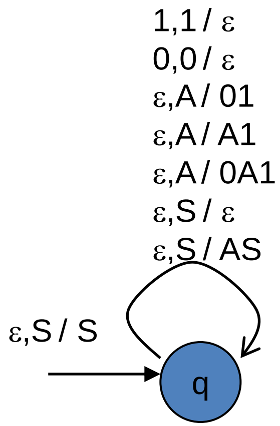
How will this new PDA work?

Lets simulate string <u>0011</u>

SIMULATING STRING 0011 ON THE NEW PDA …

PDA (δ):

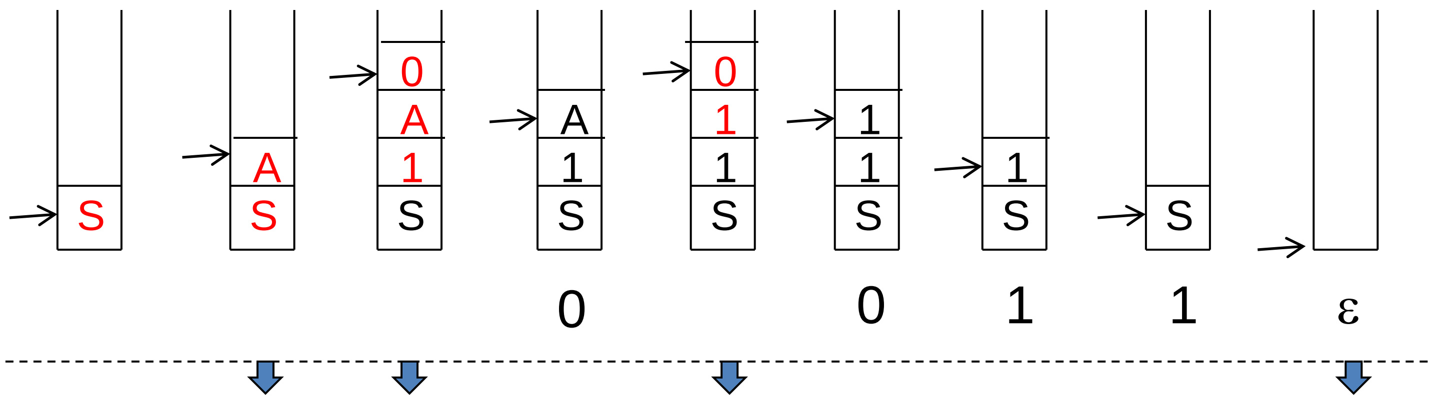
$\delta(q, \varepsilon, S) = \{ (q, AS), (q, \varepsilon) \}$
$\delta(q, \varepsilon, A) = \{ (q,0A1), (q,A1), (q,01) \}$
$\delta(q, 0, 0) = \{ (q, \varepsilon) \}$
$\delta(q, 1, 1) = \{ (q, \varepsilon) \}$



Leftmost deriv.:

$S \Rightarrow AS$
$\Rightarrow 0A1S$
$\Rightarrow 0011S$
$\Rightarrow 0011$

Stack moves (shows only the successful path):



$S \quad \Rightarrow AS \quad \Rightarrow 0A1S \quad \Rightarrow 0011S \quad \Rightarrow 0011$

Accept by empty stack

CONVERSION OF PDA TO CFG

FROM PDA TO CFG

- If L(M) is for some PDA M, then L(M) is CFL. → CFG
- Let
  - It has a single final state q_f iff the stack is empty.
  - All transitions must have the form

    $\delta(q_i, a, A) = (q_j, \lambda)$

    $\delta(q_i, a, A) = (q_j, BA)$
  - That is, each move either increases or decreases the stack content by a single symbol.

FROM PDA TO CFG

– Given M = (Q, Σ, Γ, δ, q_0, z_0, {q_f}) PDA

– Construct G = (V, T, P, S) CFG

– V → elements of the form [q X p], q and p in Q and X in Γ S

- The intuitive meaning of non-terminals [q X p], it represents the language (set of strings) that label paths from q to p that have the net effect of popping X off the stack.

– T = Σ

– S - start symbol

– S→[q_0 z_0 q] for each q in Q.

EXAMPLE

PDA

state	input	stack	new state	stack
q	0	Z	q	XZ
q	0	X	q	XX
q	1	X	r	ϵ
r	1	X	r	ϵ
* r	ϵ	Z	r	ϵ

$\delta(q, 0, Z) = \{(q, XZ)\}$

$\delta(q, 1, X) = \{(r, \epsilon)\}$

$L = \{0^n 1^n / n \geq 1\}$

- $(Q, \Sigma, \Gamma, \delta, q_0, z, F)$ where $Q = \{q, r\}$

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

  $\Gamma = \{Z,X\}$

EXAMPLE

state	input	stack	new state	stack
q	0	Z	q	XZ
q	0	X	q	XX
q	1	X	r	ϵ
r	1	X	r	ϵ
r	ϵ	Z	r	ϵ

N, T, P, S ←

- Possible NTs

$S \to [qZq] \mid [qZr]$

$[qZq] \to 0[qXq][qZq] \mid \cancel{0[qXr][rZq]}$

$[qZr] \to 0[qXq][qZr] \mid 0[qXr][rZr]$

$\cancel{[qZq]}$

$[rZr] \to \varepsilon$

q, r

$\to [qXp]$

q q
q r
r q
r r

2

X, Z

$[qXq] \to 0[qXq][qXq] \mid \cancel{0[qXr][rXq]}$

$[qXr] \to 1 \mid 0[qXq][qXr] \mid 0[qXr][rXr]$

$\cancel{[rXq]}$

$[rXr] \to 1$

$S \to A \mid B$

Simplified grammar.

FROM PDA TO CFG

- The productions P of G has 2 forms:
- First, for the start symbol S we add productions to the "[startState, startStackSymbol, state]" NT for every state in the PDA.
- The language generated by S will correspond to the set of strings labeling paths from S to any other state that have the net effect of emptying the stack (popping off the starting stack symbol).

S→[startState, startStackSymbol, state]

EXAMPLE

state	input	stack	new state	stack
q	0	Z	q	XZ
q	0	X	q	XX
q	1	X	r	ϵ
r	1	X	r	ϵ
r	ϵ	Z	r	ϵ

- Start production

$S \rightarrow [qZq] \mid [qZr]$

FROM PDA TO CFG

- Next, for each transition in the PDA of the form

  $\delta\ (s, a, Y\ ) = \{(t, \varepsilon)\}$

  i.e. state s goes to t while reading symbol a from the input and popping Y off the stack)

  $[q\ X\ p]$

- add the production

  $$[sY\ t] \rightarrow a$$

  to the grammar ("push nothing" transitions)

- This corresponds to the fact that there is a path from s to t labeled by a that has the net effect of popping Y off the stack.

EXAMPLE

state	input	stack	new state	stack
q	0	Z	q	XZ
q	0	X	q	XX
q	1	X	r	ϵ
r	1	X	r	ϵ
r	ϵ	Z	r	ϵ

- "push nothing" transitions

$\delta(q, 1, X) = \{(r, \varepsilon)\}$ $\quad [sYt] \rightarrow a$

$[qXr] \rightarrow 1$

$\delta(r, 1, X) = \{(r, \varepsilon)\}$

$[rXr] \rightarrow 1$

$\delta(r, \varepsilon, Z) = \{(r, \varepsilon)\}$

$[rZr] \rightarrow \varepsilon$

FROM PDA TO CFG

- These "push nothing" transitions are just a special case of the general rule: $\delta(s, a, Y) = \{(t, Y_1 Y_2 Y_3 \ldots Y_k)\}$
- **If there is a transition from s to t that reads a from the input, Y from the stack, and pushes k symbols $Y_1 Y_2 \ldots Y_k$ onto the stack, add all productions of the form**

$$[s\ Y\ s_k] \rightarrow a[t\ Y_1 s_1][s_1 Y_2 s_2] \ldots [s_{k-1} Y_k s_k]$$

$k \rightarrow$ no of states in PDA

to the grammar (for all combinations of states $s_1, s_2, \ldots, s_k$).

EXAMPLE

state	input	stack	new state	stack
q	0	Z	q	XZ
q	0	X	q	XX
q	1	X	r	ϵ
r	1	X	r	ϵ
r	ϵ	Z	r	ϵ

- In this case, we are pushing the string XZ of length 2 on the stack, and need to add all productions of the form:

$$[qZs_2] \rightarrow 0[qXs_1][s_1Zs_2]$$

2 states q, r

$$\underset{}{\delta}(\underset{s}{q}, \underset{a}{0}, \underset{Y}{Z}) = \{(\underset{t}{q}, \underset{Y_1}{X}\underset{Y_2}{Z})\}$$

$$[sYs_k] \rightarrow a[tY_1s_1][s_1Y_2s_2]$$

$$\rightarrow [qZs_2] \rightarrow 0[qXs_1][s_1Zs_2]$$

$s_2 = q$ $\quad s_1 = q$ $\quad s_1 = r$

$$[qZq] \rightarrow 0[qXq][qZq] \,/\, 0[qXr][rZq]$$

$s_2 = r$ $\quad s_1 = q$ $\quad s_1 = r$

$$[qZr] \rightarrow 0[qXq][qZr] \,/\, 0[qXr][rZr]$$

EXAMPLE

state	input	stack	new state	stack
q	0	Z	q	XZ
q	0	X	q	XX
q	1	X	r	ϵ
r	1	X	r	ϵ
r	ϵ	Z	r	ϵ

- In this case, we are pushing the string XX of length 2 on the stack, and need to add all productions of the form:

$$[qXs_2] \rightarrow 0[qXs_1][s_1\cancel{Z}s_2]$$

$$\delta(\underset{s}{q}, \underset{a}{0}, \underset{Y}{X}) = \{(\underset{t}{q}, \underset{Y_1, Y_2}{XX})\}$$

$S_2 = q$ $S_1 = q$ $S_2 = r$

$$[qXq] \rightarrow 0[qXq][qXq] \mid 0[qXr][rXq]$$

$S_2 = r$ $S_1 = q$

$$[qXr] \rightarrow 0[qXq][qXr] \mid 0[qXr][rXr]$$

FROM PDA TO CFG

- To verify that the language generated by the grammar is the same as the language accepted by the PDA (by empty-stack) use two proofs by induction, one to show that every word in the language of the PDA can be generated by the grammar, and another to show that every word that can be generated by the grammar is accepted by the PDA.

SUMMARY

- Equivalence of PDA and CFG

TEST YOUR KNOWLEDGE

- The entity which generate Language is termed as:
  a) Automata
  b) Tokens
  c) Grammar
  d) Data
- The minimum number of productions required to produce a language consisting of palindrome strings over $\sum=\{a,b\}$ is
  a) 3
  b) 7 c) 5 d) 6

REFERENCE

- Hopcroft J.E., Motwani R. and Ullman J.D, “Introduction to Automata Theory, Languages and Computations”, Second Edition, Pearson Education, 2008