PDA and CFG conversions

CSCI 3130 Formal Languages and Automata Theory

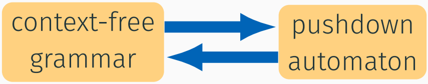
Siu On CHAN

Fall 2022

Chinese University of Hong Kong

CFGs and PDAs

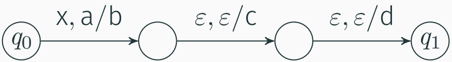
L has a context-free grammar if and only if it is accepted by some pushdown automaton.



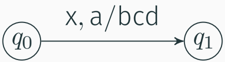
Will first convert CFG to PDA

Convention

A sequence of transitions like



will be abbreviated as



replace a by bcd on stack

Converting a CFG to a PDA

Idea: Use PDA to simulate derivations

$A \to 0A1$
$A \to B$
$B \to \#$

Example:
$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 00B11 \Rightarrow 00\#11$

Rules:

1. Push start symbol A onto stack
2. Rewrite top variable on stack based on production (reversed)

PDA control		stack	input
push start variable	$\varepsilon, \varepsilon/\$A$	$\$A$	00#11
replace by production in reverse	$\varepsilon, A/1A0$	$\$1A0$	00#11

Converting a CFG to a PDA

Idea: Use PDA to simulate derivations

$A \to 0A1$
$A \to B$
$B \to \#$

Example:
$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 00B11 \Rightarrow 00\#11$

Rules:

1. Push start symbol A onto stack
2. Rewrite top variable on stack based on production (reversed)
3. Pop top terminal if it matches input

PDA control		stack	input
push start variable	$\varepsilon, \varepsilon / \$A$	$\$A$	00#11
replace by production in reverse	$\varepsilon, A / 1A0$	$\$1A0$	00#11
pop terminal and match	$0, 0 / \varepsilon$	$\$1A$	0#11
replace by production in reverse	$\varepsilon, A / 1A0$	$\$11A0$	0#11
	$\vdots$		

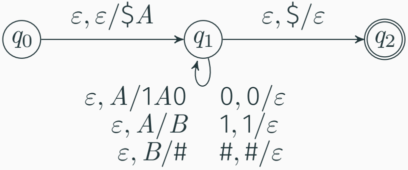
Converting CFG → PDA

CFG

$A \to 0A1$

$A \to B$

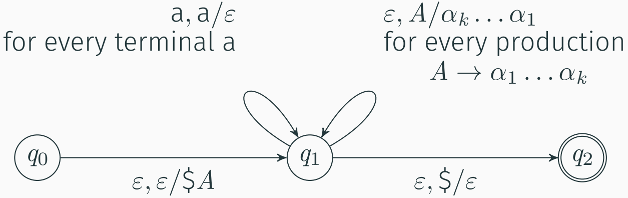
$B \to \#$



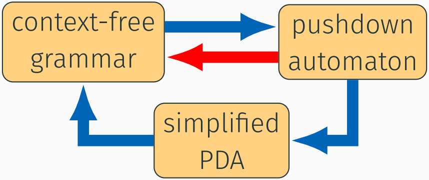
input	00#11	00#11	00#11	00#11	00#11
stack	$\$A$	$\$1A0$	$\$1A$	$\$11A0$	$\$11A$
input	00#11	00#11	00#11	00#11	00#11
stack	$\$11B$	$\$11\#$	$\$11$	$\$1$	$\$$

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 00B11 \Rightarrow 00\#11$$

General CFG → PDA conversion



From PDAs to CFGs

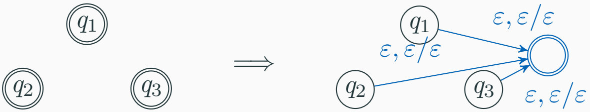


Simplified pushdown automaton:

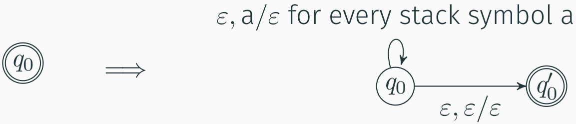
- Has a single accepting state
- Empties its stack before accepting
- Each transition is either a push, or a pop, but not both

Simplifying the PDA

Single accepting state

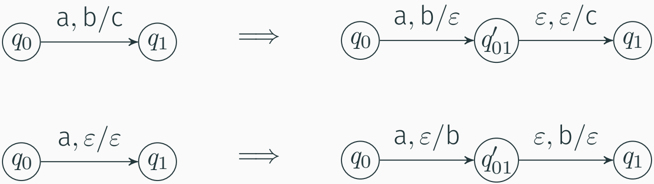


Empties its stack before accepting



Simplifying the PDA

Each transition either pushes or pops, but not both



Simplified PDA to CFG

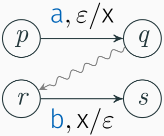
For every pair (q, r) of states in PDA, introduce variable A_{qr} in CFG

Intention:

A_{qr} generates all strings that allow the PDA to go from q to r

(with empty stack both at q and at r)

Simplified PDA to CFG

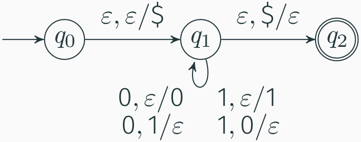
PDA	CFG
	$A_{qq} \to \varepsilon$
	$A_{pr} \to A_{pq}A_{qr}$
	$A_{ps} \to \mathtt{a} A_{qr} \mathtt{b}$ a = ε or b = ε allowed

Notation: 

means p can reach q through a path

Start variable: A_{pq} (initial state p, accepting state q)

Example: Simplified PDA to CFG

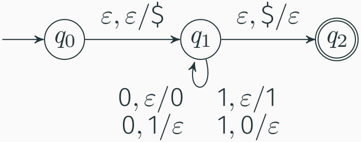


productions:

variables:

start variable:

Example: Simplified PDA to CFG



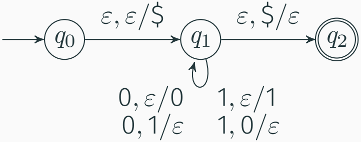
productions:

$$A_{00} \to \varepsilon$$
$$A_{11} \to \varepsilon$$
$$A_{22} \to \varepsilon$$
$$A_{02} \to A_{01} A_{12}$$
$$A_{01} \to A_{01} A_{11}$$
$$A_{12} \to A_{11} A_{12}$$
$$A_{11} \to A_{11} A_{11}$$
$$A_{11} \to 0 A_{11} 1$$
$$A_{11} \to 1 A_{11} 0$$
$$A_{02} \to A_{11}$$

variables: $A_{00}, A_{11}, A_{22}, A_{01}, A_{02}, A_{12}$

start variable: A_{02}

Example: Simplified PDA to CFG



variables: $A_{00}, A_{11}, A_{22}, A_{01}, A_{02}, A_{12}$

start variable: A_{02}

productions:

$$A_{00} \to \varepsilon$$
$$A_{11} \to \varepsilon$$
$$A_{22} \to \varepsilon$$
$$A_{02} \to A_{01} A_{12}$$
$$A_{01} \to A_{01} A_{11}$$
$$A_{12} \to A_{11} A_{12}$$
$$A_{11} \to A_{11} A_{11}$$
$$A_{11} \to 0 A_{11} 1$$
$$A_{11} \to 1 A_{11} 0$$
$$A_{02} \to A_{11}$$