Theory of Computation

MIEIC, 2nd Year

João M. P. Cardoso
Email: jmpc@acm.org

Outline

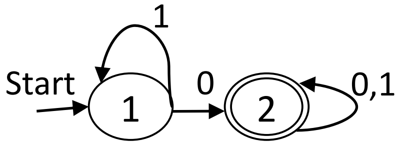
- Push Down Automata (PDAs)
- From CFGs to PDAs
- From PDAs to CFGs

Automata with Stack

- The automata with stack are to the CFLs as the DFA, NFA, and ε-NFA automata are to RLs

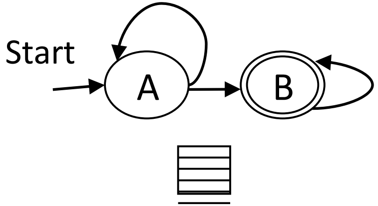
Regular Languages (RLs)

1*0(1+0)*



Context-Free Languages (CFLs)

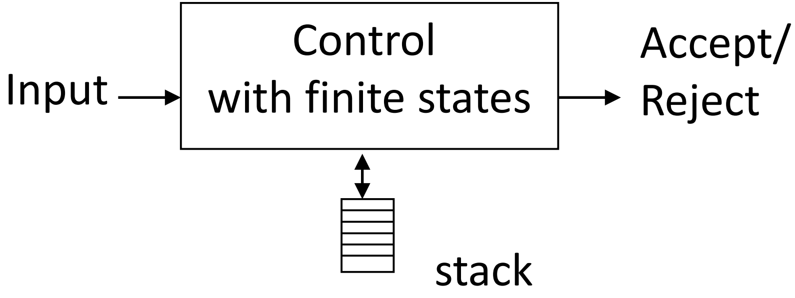
E → I | E+E | E×E | (E)
I → a | b | Ia | Ib | I0 | I1



Idea

- The pushdown automata (PDA) is a ε-NFA with a stack of symbols
  - Add the possibility to memorize an infinite quantity of information
  - The PDA only has access to the top of the stack (LIFO) [no random access memory]

- How it works:
  - The control unit reads and consumes the input symbols
  - Transition to a new state based on the current state, input symbol, and symbol in the top of the stack
  - Spontaneous transitions with ε
  - Top of the stack substituted by symbols



Example of the Palindromes

- $L_{wwr} = \{ww^R \mid w \in (0+1)^*\}$ palindromes of even length
- CFG of the Language:
  - $P \rightarrow \varepsilon \mid 0P0 \mid 1P1$
- Build a PDA
  - Initial state q_0 means that the PDA didn't reach the middle of ww^R; store in stack the symbols of w
  - At every moment we assume that we may have reached the middle (end of w) and we follow a transition ε to q_1; the stack contains w, beginning in the bottom and finishing in the top; the non-determinism is simulated by the maintenance of the two states
  - In q_1 the PDA compares the input symbol with the top of the stack; if there is not a match, the prediction was wrong and this Computing branch dies; other branch might succeed
  - If the stack gets empty (and the input is finished) the PDA discovered w and w^R

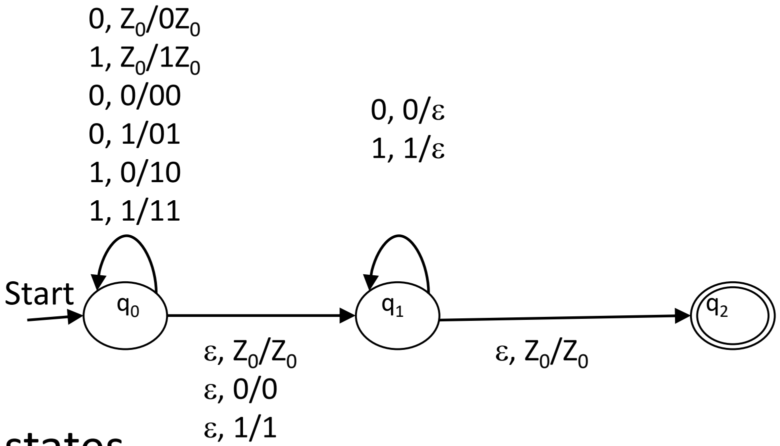
Formal Definition of PDA

- Pushdown Automaton (PDA) $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
  - Q: finite set of states
  - Σ: finite set of input symbols
  - Γ: finite alphabet of the Stack
  - δ: transition function $\delta(q, a, X) = \{(p_1,\gamma_1), \ldots\}$ finite
    - q is a state, a is an input symbol or ε, X is a stack symbol
    - p_1 is a new state, γ_1 is the sequence of symbols that substituted X in the top of the stack
      - $\gamma_1 = \varepsilon$ pop of the top of the stack
      - $\gamma_1 = X$ stack is not changed
      - $\gamma_1 = YZ$ X substituted by Z followed by a push of Y
  - q_0: initial state
  - Z_0: initial symbol in the stack (initial content of the stack)
  - F: set of accept/final states

Back to the Palindrome Example

- PDA of L_{wwr} $P = (\{q_0,q_1,q_2\}, \{0,1\}, \{0,1,Z_0\}, \delta, q_0, Z_0, \{q_2\})$
  - Z_0 is used to mark the bottom of the stack and allows in the end of the Reading of ww^R to move the PDA to the accept state q_2
  - $\delta(q_0,0,Z_0)= \{(q_0,0Z_0)\}$ e $\delta(q_0,1,Z_0)= \{(q_0,1Z_0)\}$ top of the stack on the right
  - $\delta(q_0,0,0)= \{(q_0,00)\}$, $\delta(q_0,0,1)= \{(q_0,01)\}$, $\delta(q_0,1,0)= \{(q_0,10)\}$, $\delta(q_0,1,1)= \{(q_0,11)\}$
  - $\delta(q_0,\varepsilon,Z_0)= \{(q_1,Z_0)\}$, $\delta(q_0, \varepsilon,0)= \{(q_1,0)\}$, $\delta(q_0, \varepsilon,1)= \{(q_1,1)\}$
  - $\delta(q_1,0,0)= \{(q_1, \varepsilon)\}$ and $\delta(q_1,1,1)= \{(q_1, \varepsilon)\}$
  - $\delta(q_1,\varepsilon,Z_0)= \{(q_2,Z_0)\}$

Transition Diagram



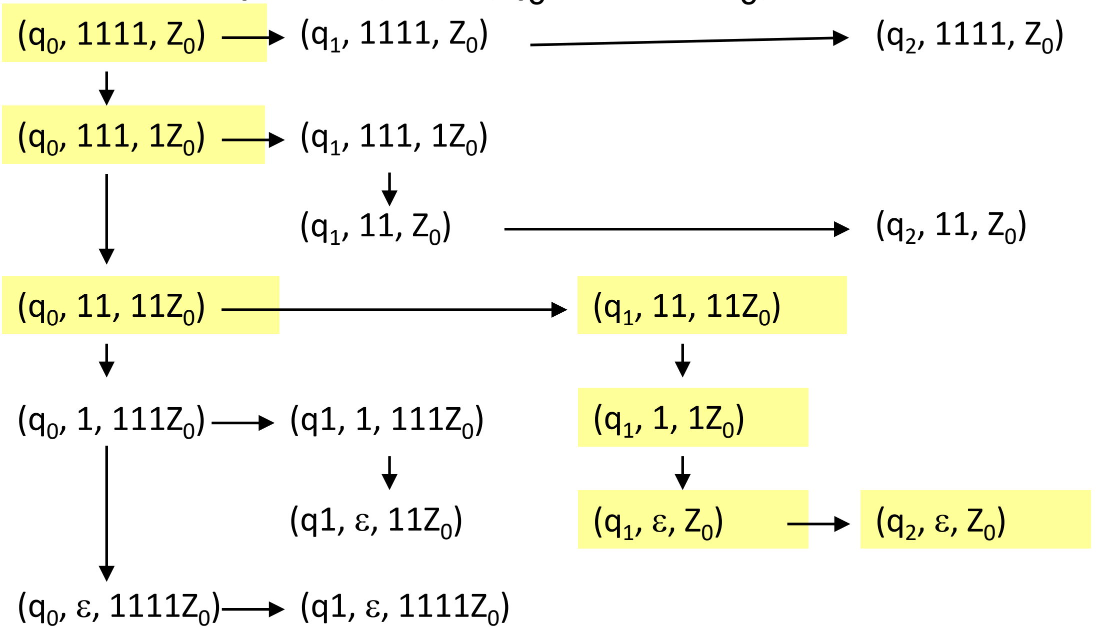
- Nodes are states
- Arrow Start indicates the initial state
- Edges correspond to transitions
  - Label $a,X/\alpha$ from q to p means that $\delta(q,a,X)$ contains (p,α)
  - The edge indicate the input and the top of the stack before and after the transition

Instantaneous Description (ID)

- Computation of a PDA
  - Evolves from configuration to configuration, in response to input symbols (or ε) and modifying the stack
  - In a DFA: all the Information in the state;
  - In a PDA: state + stack
- Instantaneous description (q,w,γ)
  - q: state
  - w: input reminiscent
  - γ: stack content (top at the left)
- Step of a PDA $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
  - If $\delta(q,a,X)$ contains (p,α), for all strings w in Σ^* and β in Γ^*
    - $(q, aw, X\beta) \vdash_P (p,w,\alpha\beta)$
  - We use $\vdash^*$ for zero or more steps (**computation**)

The Palindrome Example

- Input w=1111
- Initial Instantaneous Description (ID): $(q_0, 1111, Z_0)$



Exercise 1

- Given the PDA $P = (\{q,p\}, \{0,1\}, \{Z_0,X\}, \delta, q, Z_0, \{p\})$ with
  - $\delta(q,0,Z_0) = \{(q,XZ_0)\}$
  - $\delta(q,0,X) = \{(q,XX)\}$
  - $\delta(q,1,X) = \{(q,X)\}$
  - $\delta(q,\varepsilon,X) = \{(p, \varepsilon)\}$
  - $\delta(p, \varepsilon,X) = \{(p, \varepsilon)\}$
  - $\delta(p,1,X) = \{(p,XX)\}$
  - $\delta(p,1,Z_0) = \{(p, \varepsilon)\}$
- Starting with the initial instantaneous description (ID), (q,w,Z_0), show all the IDs reachable when the input is:
  - a) 01 b) 0011 c) 010

Principles Related to IDs

- If a sequence IDs (computations) is legal for a PDA P then the computations that result of adding any string w to the input in each ID is also legal
- If a computation is legal for a PDA P then the computations that result of adding any set of symbols below the bottom of the stack in each ID is also legal
  - Theorem 1: If $(q,x,\alpha) \vdash^* (p,y,\beta)$ then $(q,xw,\alpha\gamma) \vdash^* (p,yw,\beta\gamma)$
- If a computation is legal for a PDA P and a tail of the input is not consumed, then the computation that results of removing that tail from the input in each ID is also legal
  - Theorem 2: If $(q,xw,\alpha) \vdash^* (p,yw,\beta)$ then $(q,x,\alpha) \vdash^* (p,y,\beta)$

Comments

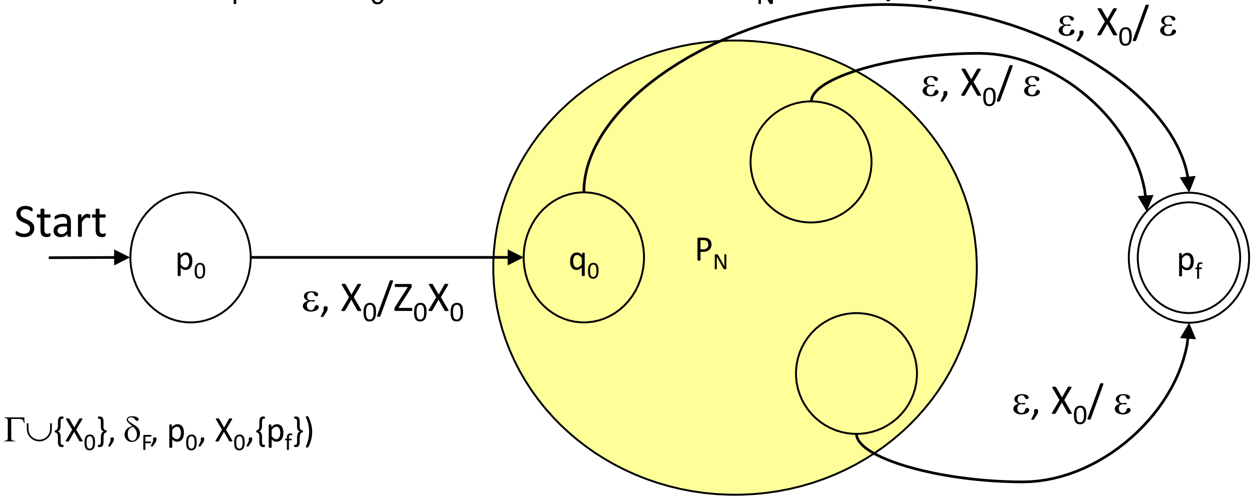
- Symbols for which P never looks cannot affect its computations
- Similar concept to the notion of context-free language:
  - What is in the side does not affect the computation
- Theorem 2 is not the inverse of 1 because what is in the stack can influence the computation even if it is not discarded
  - For example, it can remove from the stack one symbol in each step and in the last step to add everything that was removed

Language of a PDA

- Accepting by final state
  - Given the PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
  - Language of P accepted by final state
  - $L(P) = \{w \mid (q_0,w,Z_0) \vdash^* (q,\varepsilon,\alpha)\}$ and $q \in F$
  - Final content of the stack is irrelevant
- Example:
  - $(q_0,ww^R,Z_0) \vdash^* (q_0,w^R,w^RZ_0) \vdash (q_1,w^R,w^RZ_0) \vdash^* (q_1,\varepsilon,Z_0) \vdash (q_2,\varepsilon,Z_0)$
- Accepting by empty stack
  - $N(P) = \{w \mid (q_0,w,Z_0) \vdash^* (q,\varepsilon,\varepsilon)\}$
  - Language accepted by empty stack, set of input w consumed by P emptying at the same time the stack (N(P) = stack null)
- Same example: modifications to empty the stack and to obtain $N(P_N)=L(P)$
  - $\delta(q_1,\varepsilon,Z_0)= \{(q_2,Z_0)\}$ becomes $\delta(q_1,\varepsilon,Z_0)= \{(q_2,\varepsilon)\}$
  - $(q_0,ww^R,Z_0) \vdash^* (q_0,w^R,w^RZ_0) \vdash (q_1,w^R,w^RZ_0) \vdash^* (q_1,\varepsilon,Z_0) \vdash (q_2,\varepsilon,\varepsilon)$

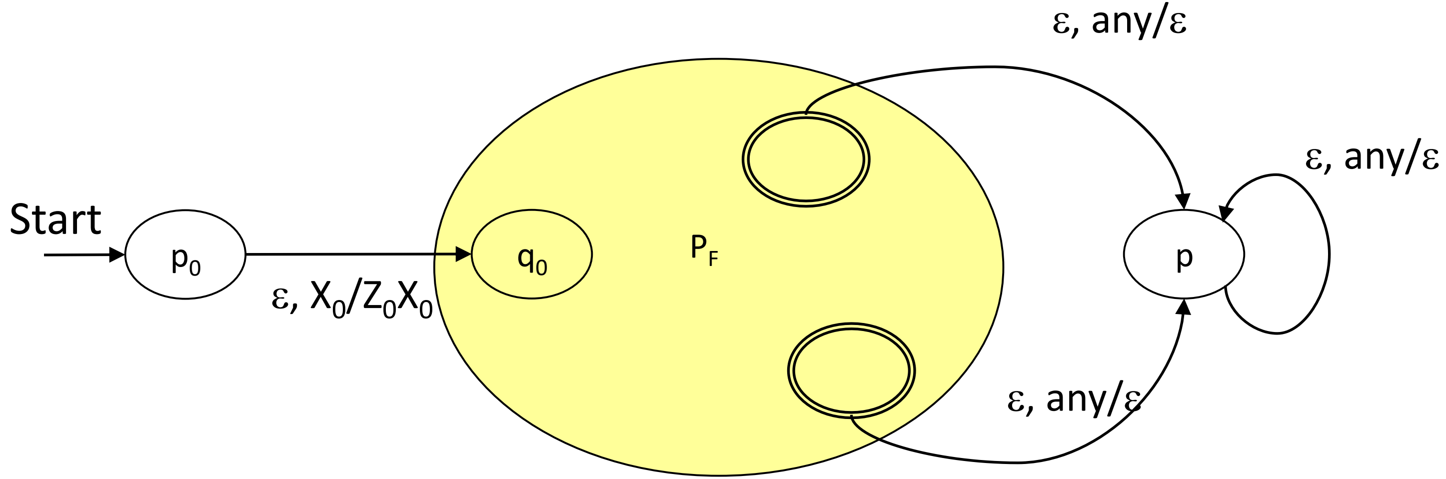
From the Empty Stack to the Final State

- Theorem: If $L = N(P_N)$ for a PDA $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$ then there exists a PDA P_F such that $L = L(P_F)$
  - Two equivalent methods for accepting an input
  - While for a PDA P we can have $L(P) \neq N(P)$
  - Starting from P_N, use a new $X_0 \notin \Gamma$ as initial symbol of P_F and as a mark of the bottom of the stack: P_F sees X_0 when the stack of P_N is empty



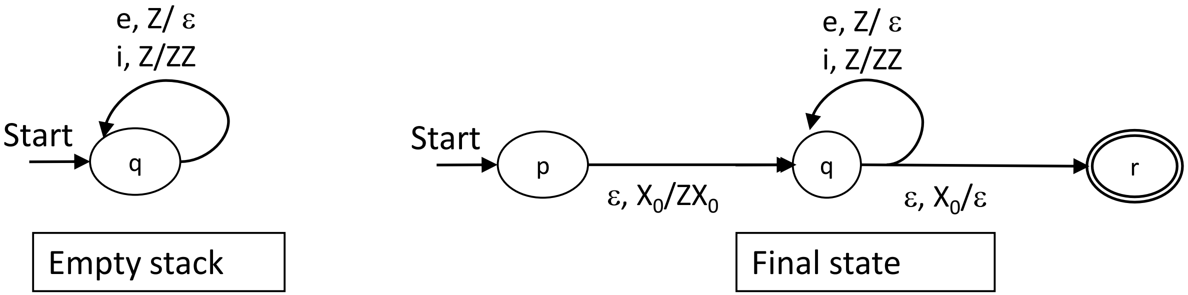
$P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$

From the Final State to the Empty Stack



Example of Conversion

- Define a PDA which processes sequences formed with "i" and "e", meaning the *if* and *else*, constructions present in many programming languages, detecting invalid sequences (i.e., sequences with more "e's" than "i's" in a prefix)
  - Initial symbol: Z; stack with Z^n means that no. of i's – no. e's = n-1
  - Accept by empty stack
  - Conversion to final state



Equivalence between PDAs and CFGs

- It is proved that the CFLs defined by a CFG are the languages accepted by a PDA by empty stack and thus also accepted by a PDA by final state
- Idea: given a CFG G build a PDA that simulates the leftmost derivations of G
  - Any left syntax form non-terminal can be written as $xA\alpha$,
    - Where A if the leftmost variable,
    - x are the terminals in the left of A,
    - and α is the sequence of terminals and variables in the right of A.
    - $A\alpha$ is named tail
  - CFG G = (V,T,Q,S)
  - PDA that accepts L(G) by empty stack: $P = (\{q\}, T, V \cup T, \delta, q, S)$
  - For each variable A:
    - $\delta(q,\varepsilon,A)=\{(q,\beta) \mid A \rightarrow \beta$ is a production in $G\}$
  - For each terminal a:
    - $\delta(q,a,a)=\{(q,\varepsilon)\}$

From CFGs to PDAs

- Given the CFG

$$E \rightarrow I \mid E+E \mid E\times E \mid (E)$$
$$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$$

- Obtain a PDA for accepting the same language by empty stack
  - $P_N = (\{q\}, \{a,b,0,1,(,),+,\times\}, \{a,b,0,1,(,),+,\times,E,I\}, \delta, q, E)$
  - $\delta(q,\varepsilon,I) = \{(q,a), (q,b), (q,Ia), (q,Ib), (q,I0), (q,I1)\}$
  - $\delta(q,\varepsilon,E) = \{(q,I), (q,E+E), (q,E\times E), (q,(E))\}$
  - $\delta(q,a,a) = \{(q,\varepsilon)\}$; $\delta(q,b,b)=\{(q,\varepsilon)\}$, $\delta(q,0,0) = \{(q,\varepsilon)\}$; $\delta(q,1,1) = \{(q,\varepsilon)\}$; $\delta(q,(,() = \{(q,\varepsilon)\}$; $\delta(q,),)) = \{(q,\varepsilon)\}$; $\delta(q,+,+) = \{(q,\varepsilon)\}$; $\delta(q,\times,\times) = \{(q,\varepsilon)\}$
- Only one state
- Processing of variables is spontaneous
- Only the terminals consume inputs

Exercise 2

- Using a CFG and the PDA for the language of expressions
  - a) Obtain a leftmost derivation for $a\times(a+b00)$
  - b) Obtain the computing trace for the PDA, i.e., the sequence of the instantaneous descriptions

$$E \rightarrow I \mid E+E \mid E\times E \mid (E)$$
$$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$$

- a)
  - $E \Rightarrow E\times E \Rightarrow I\times E \Rightarrow a\times E \Rightarrow a\times(E) \Rightarrow a\times(E+E) \Rightarrow a\times(I+E) \Rightarrow$
  - $a\times(a+E) \Rightarrow a\times(a+I) \Rightarrow a\times(a+I0) \Rightarrow a\times(a+I00) \Rightarrow a\times(a+b00)$
- b)
  - $(q, a\times(a+b00), E) \vdash (q, a\times(a+b00), E\times E) \vdash (q, a\times(a+b00), I\times E) \vdash$
  - $(q, a\times(a+b00), a\times E) \vdash (q, \times(a+b00), \times E) \vdash (q, (a+b00), E) \vdash$
  - $(q, (a+b00), (E)) \vdash (q, a+b00), E)) \vdash (q, a+b00), E+E)) \vdash$
  - $(q, a+b00), I+E)) \vdash (q, a+b00), a+E)) \vdash (q, +b00), +E)) \vdash$
  - $(q, b00), E)) \vdash (q, b00), I)) \vdash (q, b00), I0)) \vdash (q, b00), I00)) \vdash$
  - $(q, b00), b00)) \vdash (q, 00), 00)) \vdash (q, 0), 0)) \vdash (q,),)) \vdash (q, \varepsilon, \varepsilon)$

Exercise 3

- Convert to PDA, the CFG with the following productions:

1. $A \rightarrow aAA$
2. $A \rightarrow aS \mid bS \mid a$
3. $S \rightarrow SS \mid (S) \mid \varepsilon$
4. $S \rightarrow aAS \mid bAB \mid aB$
5. $A \rightarrow bBB \mid aS \mid a$
6. $B \rightarrow bA \mid a$

Exercise 4

- Convert the CFG below to a PDA:

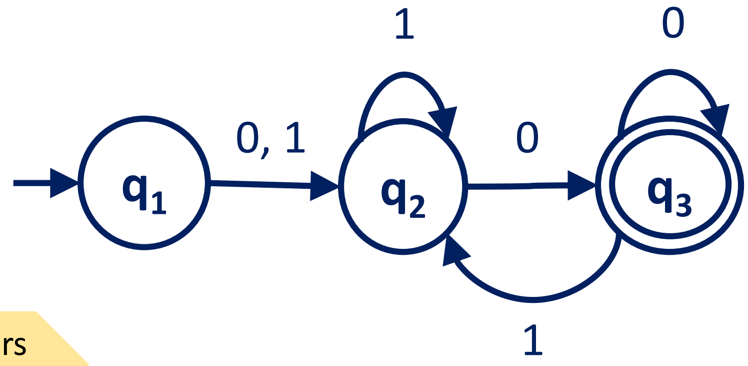
1. $S \rightarrow aAA$
2. $A \rightarrow aS|bS|a$

From PDAs to CFGs

Let’s Start by Converting FAs to CFGs

- One CFG variable for each FA state (e.g., q_i represented by L_i)
- One CFG rule for each transition
- For each accept state q_j we include $L_j \to \varepsilon$
- Example:

$$L_1 \to 0L_2 \mid 1L_2$$
$$L_2 \to 1L_2 \mid 0L_3$$
$$L_3 \to 1L_2 \mid 0L_3 \mid \varepsilon$$



The CFG obtained is usually identified as right-linear of right regular grammars (https://en.wikipedia.org/wiki/Linear_gramar)

From PDAs to CFGs

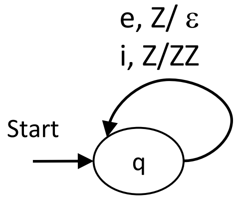
- Idea:
  - Recognizing that the main event of PDA processing is the pop of a symbol while consuming the input
- Add variables to the grammar for
  - Each elimination of a stack symbol X
  - Each state transition from p to q eliminating X, represented by a compound symbol [pXq]
- From PDA $P = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$ build CFG $G = (V, \Sigma, R, S)$
  - Variables V: contain S and the compound symbols [pXq]

From PDAs to CFGs (cont.)

- Productions R:
  - For all states p, G constains S → $[q_0Z_0p]$ (q_0 is the start state of the PDA)
    - Symbol $[q_0Z_0p]$ generates all the strings w that pop Z_0 from the stack while going from state q_0 to state p, $(q_0,w,Z_0) \vdash^* (p,\varepsilon,\varepsilon)$
    - Hence S generates all the strings w that empty the stack
  - If $\delta(q,a,X)$ contains $(r, Y_1Y_2\ldots Y_k)$, $k\geq 0$, $a\in\Sigma$ or $a=\varepsilon$, then for all the lists of states $r_1,r_2,\ldots,r_k$, G contains (when k=0, the pair is (r, ε))

    $[qXr_k] \rightarrow a[rY_1r_1][r_1Y_2r_2]\ldots[r_{k-1}Y_kr_k]$
    - A way to pop X and to go from q to r_k is to read a (it can be ε) and use some input to pop Y_1 while going from r to r_1, etc.

PDA to CFG Example 1



- Convert the PDA $P_N=(\{q\},\{i,e\},\{Z\},\delta_N,q,Z)$ to a CFG
  - Accept strings that for the first time don't follow that each "e" needs to correspond to a previous "i"
- Solution:
  - Only a state q and a stack symbol Z
  - Two variables: S, start symbol; [qZq], unique symbol from the states and symbols of P_N
  - Production:
    - S → [qZq] (if there were more states p and r we would have S→[qZp] and S→[qZr])
    - From $\delta_N(q,i,Z)=\{(q,ZZ)\}$ obtain [qZq]→i[qZq] [qZq] (if there were more states p and r we would have [qZp]→i[qZr] [rZp])
    - From $\delta_N(q,e,Z)=\{(q,\varepsilon)\}$ obtain [qZq]→e (Z is substituted by nothing)
    - Naming A to [qZq] we obtain S→A e A → iAA | e

PDA to CFG Example 2

- $P_N = (\{q,r\},\{0,1\},\{X,Z\},\delta_N,q,Z)$

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

1. $\delta(q, 1, Z) = \{(q, XZ)\}$
2. $\delta(q, 1, X) = \{(q, XX)\}$
3. $\delta(q, 0, X) = \{(p, X)\}$
4. $\delta(q, \varepsilon, X) = \{(q, \varepsilon)\}$
5. $\delta(p, 1, X) = \{(p, \varepsilon)\}$

PDA to CFG Example 2

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

- P_N=({q,r},{0,1},{X,Z},δ_N,q,Z)

- Possible variables V = {S, [qZq], [qZr], [qXq], [qXr], [rZq], [rZr], [rXq], [rXr]}.
- Variables V = {S, [qZq], [qXq], [qXr], [rXr], [rZr]}

- S →[qZq] | [qZr]
- [qZq] → 0[qXq][qZq] | 0 [qXr][rZq]
- [qXq] → 0[qXq][qXq] | 0[qXr][rXq]
- [qXr] → 0[qXq][qXr] | 0[qXr][rXr]
- [rZq] →
- [qZr] →0[qXr][rZr] | 0[qXq][qZr]

- [rXq] →
- [rZr] → ε
- [rXr] → 1
- [qXr] → 1

PDA to CFG Example 2

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

- V = {S, [qZq], [qZr], [qXq], [qXr], [rZq], [rZr], [rXq], [rXr]}.
- [qZq], [qXq], [qXr], [rXr], [rZr]

- S →[qZq] | [qZr]
- [qZq] → 0[qXq][qZq]
- [qXq] → 0[qXq][qXq]
- [qXr] → 0[qXq][qXr] | 0[qXr][rXr] | 1
- [qZr] →0[qXr][rZr] | 0[qXq][qZr]

- [rZr] → ε
- [rXr] → 1

PDA to CFG Example 2

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

- V = {S, [qZq], [qZr], [qXq], [qXr], [rZq], [rZr], [rXq], [rXr]}.
- [qZq], [qXq], [qXr], [rXr], [rZr]

- S →[qZq] | [qZr]
- [qZq] → 0[qXq][qZq]
- [qXq] → 0[qXq][qXq]
- [qXr] → 0[qXq][qXr] | 0[qXr][rXr] | 1
- [qZr] →0[qXr][rZr] | 0[qXq][qZr]

- [rZr] → ε
- [rXr] → 1

PDA to CFG Example 2

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

- V = {S, [qZq], [qZr], [qXq], [qXr], [rZq], [rZr], [rXq], [rXr]}.
- [qZq], [qXq], [qXr], [rXr], [rZr]

- [rZr] → ε
- [rXr] → 1

- S →[qZq] | [qZr]
- [qXr] → 0[qXq][qXr] | 0[qXr][rXr] | 1
- [qZr] →0[qXr][rZr] | 0[qXq][qZr]

PDA to CFG Example 2

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

- V = {S, [qZq], [qZr], [qXq], [qXr], [rZq], [rZr], [rXq], [rXr]}.
- [qZq], [qXq], [qXr], [rXr], [rZr]

- S →[qZr]
- [qXr] → 0[qXr][rXr] | 1
- [qZr] →0[qXr][rZr]

- [rZr] → ε
- [rXr] → 1

PDA to CFG Example 2

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

- V = {S, <u>[qZq]</u>, [qZr], <u>[qXq]</u>, <u>[qXr]</u>, [rZq], <u>[rZr]</u>, [rXq], <u>[rXr]</u>}.
- [qZq], [qXq], [qXr], [rXr], [rZr]

- S → [qZr]
- [qXr] → 0[qXr][rXr] | 1
- [qZr] → 0[qXr][rZr]

- [rZr] → ε
- [rXr] → 1

PDA to CFG Example 2

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

CFG:

- S →[qZr]
- [qXr] → 0[qXr]1 | 1
- [qZr] → 0[qXr]

Substituting [qZr] by A and [qXr] by B:

- S →A
- B → 0B1 | 1
- A → 0B

Exercise 5

- PDA $P = (\{p,q\}, \{0,1\}, \{X,Z\}, \delta, q, Z))$, with the following transition function:
  1. $\delta(q, 1, Z) = \{(q, XZ)\}$
  2. $\delta(q, 1, X) = \{(q, XX)\}$
  3. $\delta(q, 0, X) = \{(p, X)\}$
  4. $\delta(q, \varepsilon, X) = \{(q, \varepsilon)\}$
  5. $\delta(p, 1, X) = \{(p, \varepsilon)\}$
  6. $\delta(p, 0, Z) = \{(q, Z)\}$
- Convert it to a CFG

Exercise 5

- PDA P = ({p,q}, {0,1}, {X,Z}, δ, q, Z)), with the following transition function:
  1. $\delta(q, 1, Z) = \{(q, XZ)\}$
  2. $\delta(q, 1, X) = \{(q, XX)\}$
  3. $\delta(q, 0, X) = \{(p, X)\}$
  4. $\delta(q, \varepsilon, X) = \{(q, \varepsilon)\}$
  5. $\delta(p, 1, X) = \{(p, \varepsilon)\}$
  6. $\delta(p, 0, Z) = \{(q, Z)\}$
- Convert it to a CFG

S is the start symbol:

S -> [qZq] | [qZp]

From rule (1):

[qZq] -> 1[qXq][qZq]

[qZq] -> 1[qXp][pZq]

[qZp] -> 1[qXq][qZp]

[qZp] -> 1[qXp][pZp]

From rule (2):

[qXq] -> 1[qXq][qXq]

[qXq] -> 1[qXp][pXq]

[qXp] -> 1[qXq][qXp]

[qXp] -> 1[qXp][pXp]

From rule (3):

[qXq] -> 0[pXq]

[qXp] -> 0[pXp]

From rule (4):

[qXq] -> ε

From rule (5):

[pXp] -> 1

From rule (6):

[pZq] -> 0[qZq]

[pZp] -> 0[qZp]

Exercise 6

- PDA $P = (\{q,p\}, \{0,1\}, \{Z_0,X\}, \delta, q, Z_0, \{p\})$ with transition function
  - $\delta(q,0,Z_0) = \{(q,XZ_0)\}$
  - $\delta(q,0,X) = \{(q,XX)\}$
  - $\delta(q,1,X) = \{(q,X)\}$
  - $\delta(q,\varepsilon,X) = \{(p, \varepsilon)\}$
  - $\delta(p, \varepsilon,X) = \{(p, \varepsilon)\}$
  - $\delta(p,1,X) = \{(p,XX)\}$
  - $\delta(p,1,Z_0) = \{(p, \varepsilon)\}$
- Obtain a CFG