O PDA (pushdown automata) é um ϵ-NFA com uma stack de símbolos

- Adiciona a possibilidade de memorizar uma quantidade infinita de informação.
- O PDA só tem acesso ao topo da stack (LIFO).
- Como funciona?
  - A unidade de controlo lê e consome os símbolos do input.
  - Transição para um novo estado baseado no estado atual, símbolo do input e símbolo no topo da stack.
  - Transições espontâneas com ϵ.
  - Topo da stack substituído por símbolos.

PDA P = $(Q, \sum, \Gamma, \delta, q_0, Z_0, F)$

- Q: conjunto dos estados.
- $\sum$: alfabeto.
- Γ: alfabeto finito da stack.
- δ: função de transição.
- q_0: estado inicial.
- Z_0: símbolo incial da stack.
- F: conjunto dos estados finais (se for um PDA de estado final).

Transições representadas cmo a, X/α

- a: símbolo do input consumido.
- X: topo da stack (que será retirado - pop).
- α: string a colocar na stack (push) - valor mais à esquerda ficará no topo da stack.

Descrição instantânea (q, ω, γ) - (ID):

- q: estado.
- ω: input restante.
- γ: conteúdo da stack.

As CFLs definidas por uma CFG são as linguagens aceites por um PDA por empty stack e também aceites por um PDA por final state.

Conversão de PDAs em CFGs:

- O evento principal do processamento de um PDA é retirar um símbolo da stack enquanto se consome o input.
- Adicionam-se variáveis à gramática para cada:
  - eliminação de um símbolo da stack X.
  - transiçã de p para q eliminando X, representado pelo símbolo composto [pXq]

Determinismo; $|\delta(q, s, t)| + |\delta(q, \epsilon, t)| \leq 1$

- q: estado.
- s: input.
- t: topo da stack.