# Autómatas apiladores

Clase 25

IIC2223 / IIC2224

Prof. Cristian Riveros

# Outline

Autómatas apiladores

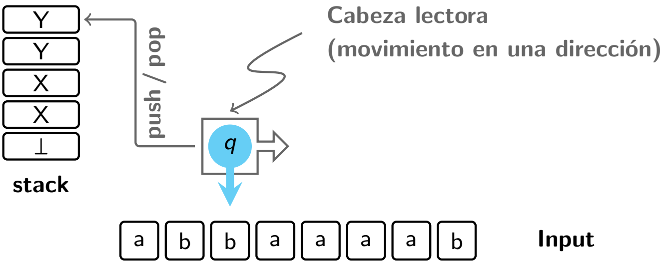
Versión alternativa

# Outline

# Autómatas para lenguajes libres de contexto

¿qué le falta a un autómata para tener el poder de una gramática?

# Autómatas apiladores



# Autómatas apiladores

## Definición

Un **autómata apilador** (PushDown Automata, PDA) es una estructura:

$$\mathcal{P} \quad = \quad (Q, \Sigma, \Gamma, \Delta, q_0, \perp, F)$$

- $Q$ es un conjunto finito de **estados**.
- $\Sigma$ es el alfabeto de **input**.
- $q_0 \in Q$ es el estado **inicial**.
- $F$ es el conjunto de estados **finales**.

$$+$$

- $\Gamma$ es el alfabeto de **stack**.
- $\perp \in \Gamma$ es el símbolo **inicial de stack**.
- $\Delta \subseteq \big(Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma\big) \times \big(Q \times \Gamma^*\big)$ es una relación finita de transición.

# Autómatas apiladores

## Definición

Un **autómata apilador** (PushDown Automata, PDA) es una estructura:

$$\mathcal{P} \quad = \quad (Q, \Sigma, \Gamma, \Delta, q_0, \perp, F)$$

Intuitivamente, la transición:

$$\big(\,(p, a, A),\ (q, B_1 B_2 \cdots B_k)\,\big) \in \Delta$$

si el autómata apilador está:

- en el estado **p**,
- leyendo **a**, y
- en el tope del stack hay una **A**

entonces:

- cambia al estado **q**, y
- modifico el tope **A** por $\mathbf{B_1 B_2 \cdots B_k}$.

# Ejemplo de autómata apilador

## Ejemplo

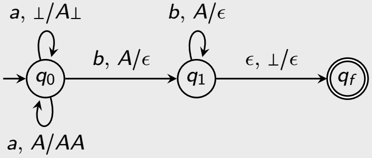
$$\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \perp, \{q_f\})$$

- $Q = \{q_0, q_1, q_f\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{A, \perp\}$
- $\Delta$ :

$$\begin{array}{ll}
(q_0, a, \perp, q_0, A\perp) & q_0\perp \overset{a}{\rightarrow} q_0A\perp \\
(q_0, a, A, q_0, AA) & q_0A \overset{a}{\rightarrow} q_0AA \\
(q_0, b, A, q_1, \epsilon) & q_0A \overset{b}{\rightarrow} q_1 \\
(q_1, b, A, q_1, \epsilon) & q_1A \overset{b}{\rightarrow} q_1 \\
(q_1, \epsilon, \perp, q_f, \epsilon) & q_1\perp \overset{\epsilon}{\rightarrow} q_f
\end{array}$$

# Ejemplo de autómata apilador

## Ejemplo




$$
\begin{array}{ll}
(q_0, a, \bot, q_0, A\bot) & \qquad q_0\bot \overset{a}{\to} q_0A\bot \\
(q_0, a, A, q_0, AA) & \qquad q_0A \overset{a}{\to} q_0AA \\
(q_0, b, A, q_1, \epsilon) & \qquad q_0A \overset{b}{\to} q_1 \\
(q_1, b, A, q_1, \epsilon) & \qquad q_1A \overset{b}{\to} q_1 \\
(q_1, \epsilon, \bot, q_f, \bot) & \qquad q_1\bot \overset{\epsilon}{\to} q_f
\end{array}
$$

# Configuración de un autómata apilador

Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \perp, F)$ un autómata apilador.

## Notación

Dado una palabra $A_1 A_2 \dots A_k \in \Gamma^+$ decimos que:

- $A_1 A_2 \dots A_k$ es un stack (contenido),
- $A_1$ es el **tope** del stack y
- $A_2 \dots A_k$ es la **cola** del stack.

## Definición

Una **configuración** de $\mathcal{P}$ es una tupla $(q \cdot \gamma, w) \in (Q \cdot \Gamma^*, \Sigma^*)$ tal que:

- $q$ es el estado actual.
- $\gamma$ es el contenido del stack.
- $w$ es el contenido del input.

# Configuración de un autómata apilador

Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \perp, F)$ un autómata apilador.

## Definición

Decimos que una configuración:

$$(q \cdot \gamma, w) \;\in\; (Q \cdot \Gamma^*, \Sigma^*)$$

- es **inicial** si $q \cdot \gamma \;=\; q_0 \cdot \perp$.
- es **final** si $q \cdot \gamma \;=\; q_f \cdot \epsilon$ con $q_f \in F$ y $w = \epsilon$.

# Ejecución de un autómata apilador

Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \bot, F)$ un autómata apilador.

## Definición

Se define la relación $\vdash_{\mathcal{P}}$ de **siguiente-paso** entre configuraciones de $\mathcal{P}$:

$$(q_1 \cdot \gamma_1, w_1) \;\vdash_{\mathcal{P}}\; (q_2 \cdot \gamma_2, w_2)$$

si, y solo si, existe una transición $(q_1, a, A, q_2, \alpha) \in \Delta$ y $\gamma \in \Gamma^*$ tal que:

- $w_1 = a \cdot w_2$
- $\gamma_1 = A \cdot \gamma$
- $\gamma_2 = \alpha \cdot \gamma$.

Se define $\vdash^*_{\mathcal{P}}$ como la clausura **refleja** y **transitiva** de $\vdash_{\mathcal{P}}$.

$(q_1\gamma_1, w_1) \vdash^*_{\mathcal{P}} (q_2\gamma_2, w_2)$ si uno puede ir de $(q_1\gamma_1, w_1)$ a $(q_2\gamma_2, w_2)$ en **0 o más pasos**.

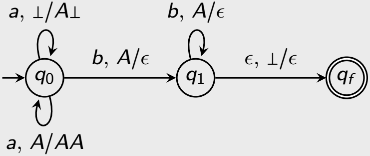
# Ejecución de un autómata apilador

## Ejemplo

Para la palabra:

$$a\,a\,a\,b\,b\,b$$

Tenemos la ejecución:




| | |
|---|---|
| $q_0 \perp$ | |
| $q_0\, A \perp$ | a |
| $q_0\, A\, A \perp$ | a |
| $q_0\, A\, A\, A \perp$ | a |
| $q_1\, A\, A \perp$ | b |
| $q_1\, A \perp$ | b |
| $q_1 \perp$ | b |
| $q_f$ | $\epsilon$ |

# Lenguajes de un autómata apilador

Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \bot, F)$ un autómata apilador y $w \in \Sigma^*$.

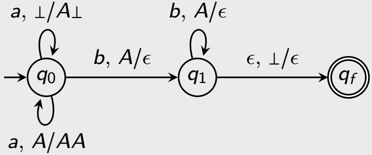
## Definiciones

- $\mathcal{P}$ **acepta** $w$ si, y solo si, $(q_0 \bot, w) \vdash^*_{\mathcal{P}} (q_f, \epsilon)$ para algún $q_f \in F$.
- El **lenguaje** aceptado por $\mathcal{P}$ se define como:

$$\mathcal{L}(\mathcal{P}) \;=\; \{w \in \Sigma^* \mid \mathcal{P} \text{ acepta } w\}$$

# Lenguajes de un autómata apilador

¿cuál es el lenguaje aceptado por $\mathcal{P}$?




$$\mathcal{L}(\mathcal{P}) \quad = \quad \{a^n b^n \mid n \geq 0\}$$

# Ejemplo de autómatas apiladores

¿cuál es un autómata apilador para cada lenguaje?

1. Todas las palabras $w \in \{\,[\,,\,]\,\}$
   que tienen los paréntesis bien balanceados.

2. Todas las palabras $w \in \{a, b\}^*$ que son palindromes.

# Outline

Autómatas apiladores

Versión alternativa

# Autómatas apiladores alternativos

Veremos otra definición **alternativa** y
**poco común** de autómatas apiladores.

¿por qué?

1. Este modelo nos ayudara a entender mejor los algoritmos de evaluación para gramáticas.
2. Modelo menos estándar pero mucho más sencillo.
3. El profesor le gusta y lo encuentra interesante :).

# Autómatas apiladores alternativos

## Definición

Un **PDA alternativo** es una estructura:

$$\mathcal{D} \quad = \quad (Q, \Sigma, \Delta, q_0, F)$$

- $Q$ es un conjunto finito de **estados**.
- $\Sigma$ es el alfabeto de **input**.
- $q_0 \in Q$ es el estado **inicial**.
- $F$ es el conjunto de estados **finales**.

$$+$$

- $\Delta \subseteq Q^+ \times (\Sigma \cup \{\epsilon\}) \times Q^*$ es una **relación finita de transición**.

# Autómatas apiladores alternativos

## Definición

Un **PDA alternativo** es una estructura:

$$\mathcal{D} \quad = \quad (Q, \Sigma, \Delta, q_0, F)$$

Intuitivamente, la transición:

$$\big(\, A_1 \cdots A_i,\ a\,, B_1 \cdots B_j \,\big) \ \in\ \Delta$$

si el autómata apilador tiene:

- $A_1 \ldots A_i$ en el tope del stack y
- leyendo **a**

entonces:

- cambia el tope $A_1 \ldots A_i$ por $B_1 \ldots B_j$.

**No hay diferencia** entre estados y alfabeto del stack.

# Configuración de un autómata apilador alternativo

Sea $\mathcal{D} = (Q, \Sigma, \Delta, q_0, F)$ un PDA alternativo.

## Definición

Una **configuración** de $\mathcal{D}$ es una tupla $(q_1 \dots q_k, w) \in (Q^+, \Sigma^*)$ tal que:

- $q_1 \dots q_k$ es el contenido del stack con $q_1$ el tope del stack.
- $w$ es el contenido del input.

Decimos que una configuración:

- $(q_0, w)$ es **inicial**.
- $(q_f, \epsilon)$ es **final** si $q_f \in F$.

# Ejecución de un autómata apilador alternativo

Sea $\mathcal{D} = (Q, \Sigma, \Delta, q_0, F)$ un PDA alternativo.

## Definición

Se define la relación $\vdash_{\mathcal{D}}$ de **siguiente-paso** entre configuraciones de $\mathcal{D}$:

$$(\gamma_1, w_1) \ \vdash_{\mathcal{D}} \ (\gamma_2, w_2)$$

si, y solo si, existe una transición $(\alpha, a, \beta) \in \Delta$ y $\gamma \in Q^*$ tal que:

- $w_1 = a \cdot w_2$
- $\gamma_1 = \alpha \cdot \gamma$
- $\gamma_2 = \beta \cdot \gamma$.

Se define $\vdash_{\mathcal{D}}^*$ como la clausura **refleja** y **transitiva** de $\vdash_{\mathcal{D}}$.

# Lenguajes de un autómata apilador alternativo

Sea $\mathcal{D} = (Q, \Sigma, \Delta, q_0, F)$ un PDA alternativo y $w \in \Sigma^*$.

## Definiciones

- $\mathcal{D}$ **acepta** $w$ si, y solo si, $(q_0, w) \vdash^*_{\mathcal{D}} (q_f, \epsilon)$ para algún $q_f \in F$.
- El **lenguaje** aceptado por $\mathcal{D}$ se define como:

$$\mathcal{L}(\mathcal{D}) = \{w \in \Sigma^* \mid \mathcal{D} \text{ acepta } w\}$$

# Lenguajes de un autómata apilador alternativo

## Ejemplo

$$\mathcal{D} \quad = \quad (Q, \{a, b\}, \Delta, q_0, F)$$

- $Q = \{q_0, q_1, q_2, q_f\}$
- $\Delta$ :

$$\begin{array}{lrcl}
(q_0, a, q_1 q_0) & q_0 & \xrightarrow{a} & q_1 q_0 \\
(q_1, a, q_1 q_1) & q_1 & \xrightarrow{a} & q_1 q_1 \\
(q_1, b, q_2) & q_1 & \xrightarrow{a} & q_2 \\
(q_2 q_1, b, q_2) & q_2 q_1 & \xrightarrow{b} & q_2 \\
(q_2 q_0, \epsilon, q_f) & q_2 q_0 & \xrightarrow{\epsilon} & q_f
\end{array}$$

$$\begin{array}{r|l}
q_0 & \\
q_1\ q_0 & \text{a} \\
q_1\ q_1\ q_0 & \text{a} \\
q_1\ q_1\ q_1\ q_0 & \text{a} \\
q_2\ q_1\ q_1\ q_0 & \text{b} \\
q_2\ q_1\ q_0 & \text{b} \\
q_2\ q_0 & \text{b} \\
q_f & \epsilon
\end{array}$$

$$\mathcal{L}(\mathcal{D}) \ = \ \{a^n b^n \mid n \geq 1\}$$

# Equivalencia entre modelos

## Teorema

Para todo autómata apilador $\mathcal{P}$
existe un autómata apilador alternativo $\mathcal{D}$, y viceversa, tal que:

$$\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{D})$$

Desde ahora, usaremos ambos modelos de manera **equivalente**.

# Equivalencia entre modelos

## Demostración: de $\mathcal{P}$ a $\mathcal{D}$

Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \bot, F)$ un PDA.

Construimos un **PDA alternativo** $\mathcal{D}_{\mathcal{P}} = (Q', \Sigma, \Delta', q_0', F')$ tal que:

- $Q' \;=\; Q \cup \Gamma \cup \{q_0'\}$
- $F' \;=\; F$
- $\Delta' \;=\; \{\, (q_0', \epsilon, q_0 \bot) \,\} \;\cup$

  $\{\, (pA, a, q\gamma) \mid (p, a, A, q, \gamma) \in \Delta \,\}$

Ejercicio: demuestre que $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{D}_{\mathcal{P}})$.

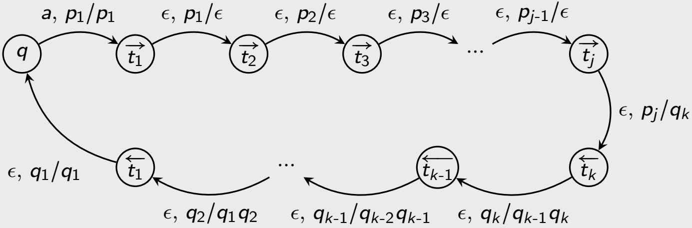
# Equivalencia entre modelos

## Demostración: de $\mathcal{D}$ a $\mathcal{P}$

Sea $\mathcal{D} = (Q, \Sigma, \Delta, q_0, F)$ un PDA alternativo.

Construimos un **PDA** $\mathcal{P}_{\mathcal{D}} = (Q', \Sigma, \Gamma', \Delta', q_0', \perp', F')$ tal que ...

... para cada $t = (p_1 \dots p_j, a, q_1 \dots q_k) \in \Delta$, haremos lo siguiente:



# Equivalencia entre modelos

## Demostración: de $\mathcal{D}$ a $\mathcal{P}$

Sea $\mathcal{D} = (Q, \Sigma, \Delta, q_0, F)$ un PDA alternativo.

Construimos un **PDA** $\mathcal{P}_{\mathcal{D}} = (Q', \Sigma, \Gamma', \Delta', q_0', \perp', F')$ tal que:

- $Q' = \{q, q_f\} \cup \bigcup_{t:(\alpha,a,\beta)\in\Delta} \{\overrightarrow{t_i} \mid 1 \leq i \leq |\alpha|\} \cup \{\overleftarrow{t_i} \mid 1 \leq i \leq |\beta|\}$
- $\Gamma' = Q$
- $\perp' = q_0$
- $q_0' = q$
- $F = \{q_f\}$

# Equivalencia entre modelos

## Demostración: de $\mathcal{D}$ a $\mathcal{P}$

Sea $\mathcal{D} = (Q, \Sigma, \Delta, q_0, F)$ un PDA alternativo.

Construimos un **PDA** $\mathcal{P}_{\mathcal{D}} = (Q', \Sigma, \Gamma', \Delta', q_0', \perp', F')$ tal que:

- $\Delta'$ : para cada $t = (p_1 \ldots p_j, a, q_1 \ldots q_k) \in \Delta$ tenemos:

$$
\begin{array}{rll}
 & (q, a, p_1, \overrightarrow{t_1}, p_1) \in \Delta' & \\
 & (\overrightarrow{t_i}, \epsilon, p_i, \overrightarrow{t_{i+1}}, \epsilon) \in \Delta' & \text{para todo } 1 \leq i < j \\
\text{si } k = 0\text{:} & (\overrightarrow{t_j}, \epsilon, p_j, q, \epsilon) \in \Delta' & \\
\text{si } k > 0\text{:} & (\overrightarrow{t_j}, \epsilon, p_j, \overleftarrow{t_k}, q_k) \in \Delta' & \\
 & (\overleftarrow{t_i}, \epsilon, q_i, \overleftarrow{t_{i-1}}, q_{i-1}q_i) \in \Delta' & \text{para todo } 1 < i \leq k \\
 & (\overleftarrow{t_1}, \epsilon, q_1, q, q_1) \in \Delta' & \\
\text{para todo } p \in F\text{:} & (q, \epsilon, p, q_f, \epsilon) \in \Delta' &
\end{array}
$$

Ejercicio: demuestre que $\mathcal{L}(\mathcal{D}) = \mathcal{L}(\mathcal{P}_{\mathcal{D}})$.

# Cierre de clase

En esta clase vimos:

1. Definición de autómatas apiladores.
2. Definición alternativa de autómatas apiladores.
3. Equivalencia entre ambos modelos.

**Próxima clase:** Conección con gramáticas libres de contexto.