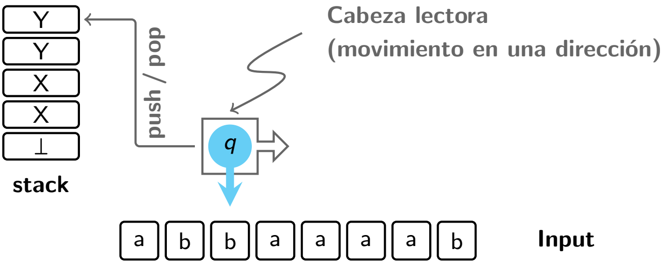
Autómatas apiladores vs gramáticas libres de contexto

Clase 21

IIC 2223

Prof. Cristian Riveros

Recordatorio: Autómatas apiladores



Recordatorio: 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**.
- Σ es el alfabeto de **input**.
- $q_0 \in Q$ es el estado **inicial**.
- F es el conjunto de estados **finales**.

$$+$$

- Γ 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.

Recordatorio: 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**.
- Σ 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**.

¿en qué se parecen CFG a PDA?

Gramáticas libre de contexto

- Terminales (Σ)
- Variables (V)
- Producciones
- $A \rightarrow \gamma$
- $A \rightarrow a$
- Derivaciones
- Árbol de derivación

Autómatas apiladores

- Alfabeto input (Σ)
- Alfabeto stack (Γ)
- Transiciones
- $pA \xrightarrow{c} q\gamma$
- $pA \xrightarrow{c} q$
- Ejecuciones
- Stack

¿en qué se parecen CFG a PDA?

Teorema

Todo **lenguaje libre de contexto** puede ser descrito equivalentemente por:

- Una gramática libre de contexto (**CFG**).
- Un autómata apilador (**PDA**).

Outline

Outline

Desde CFG a PDA

Desde PDA a CFG

Desde CFG a un PDA

Teorema

Para toda gramática libre de contexto $\mathcal{G}$,
existe un **autómata apilador alternativo** $\mathcal{D}$:

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

¿cómo **construimos** $\mathcal{D}$ desde $\mathcal{G}$?

CFG $\rightarrow$ PDA: Construcción

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una CFG.

Construimos un PDA alternativo $\mathcal{D}$ que acepta $\mathcal{L}(\mathcal{G})$:

$$\mathcal{D} \;=\; \big(V \cup \Sigma \cup \{q_0, q_f\}, \Sigma, \Delta, q_0, \{q_f\}\big)$$

La relación de transición Δ se define como:

$$\begin{array}{rcll} \Delta & = & \{\,(q_0, \epsilon, S \cdot q_f)\,\} & \cup \\ & & \{\,(X, \epsilon, \gamma) \mid X \rightarrow \gamma \in P\,\} \quad \cup & (\textbf{Expandir}) \\ & & \{\,(a, a, \epsilon) \mid a \in \Sigma\,\} & (\textbf{Reducir}) \end{array}$$

Ejemplo en clases

$$\mathcal{G}: \quad \mathrm{S} \quad \rightarrow \quad \mathrm{SS} \mid \mathrm{aSb} \mid \epsilon$$

Intuitivamente, ¿qué esta haciendo el autómata apilador $\mathcal{D}$?

CFG → PDA: Demostración

Por demostrar:

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

Dos direcciones:

1. $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$
2. $\mathcal{L}(\mathcal{D}) \subseteq \mathcal{L}(\mathcal{G})$

CFG → PDA: Demostración $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

Para cada $w \in \mathcal{L}(\mathcal{G})$ debemos encontrar
una ejecución de aceptación de $\mathcal{D}$ sobre w.

¿cómo encontramos una ejecución sobre w?

Idea

Para cada árbol de derivacíon $\mathcal{T}$ de $\mathcal{G}$ sobre w, construimos una ejecución de $\mathcal{D}$ sobre w que recorre el árbol $\mathcal{T}$ **en profundidad**.

Inducción sobre la altura del árbol $\mathcal{T}$

CFG → PDA: Demostración $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

Hipótesis de inducción

Para todo árbol de derivacíon $\mathcal{T}$ de $\mathcal{G}$ con **altura** h tal que:

- la raíz de $\mathcal{T}$ es X, y
- $\mathcal{T}$ produce la palabra w

entonces $(X \cdot \gamma, w) \vdash_{\mathcal{D}}^{*} (\gamma, \epsilon)$ para todo $\gamma \in Q^{+}$.

Si demostramos esta hipótesis habremos demostrado que $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

¿por qué?

CFG → PDA: Demostración $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

Caso base: $h = 1$

Si $\mathcal{T}$ tiene altura 1 entonces:

- $\mathcal{T}$ produce la palabra $w = a$ para algún $a \in \Sigma$ y
- $\mathcal{T}$ consiste de un nodo X y un hijo a con $X \rightarrow a$.

Entonces para todo $\gamma \in Q^{+}$:

$$(X \cdot \gamma, a) \vdash_{\mathcal{D}} (a \cdot \gamma, a) \vdash_{\mathcal{D}} (\gamma, \epsilon)$$

es una ejecución de $\mathcal{D}$ sobre a. ✓

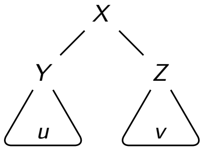
CFG → PDA: Demostración $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

Caso inductivo: $h = n$

Suponemos que el árbol de derivacíon $\mathcal{T}$ de $\mathcal{G}$ tiene **altura** n tal que:

- la raíz de $\mathcal{T}$ es X, y
- $\mathcal{T}$ produce la palabra w.

Sin perdida de generalidad suponga que $\mathcal{T}$ es de la forma:

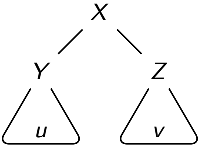


donde $w = u \cdot v$ y $X \to YZ$.

CFG → PDA: Demostración $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

Caso inductivo: $h = n$

Sin perdida de generalidad suponga que $\mathcal{T}$ es de la forma:



donde $w = u \cdot v$ y $X \to YZ$.

Por HI, se tiene que para todo $\gamma_1, \gamma_2 \in Q^+$:

$$(Y \cdot \gamma_1, u) \vdash_{\mathcal{D}}^{*} (\gamma_1, \epsilon)$$

$$(Z \cdot \gamma_2, v) \vdash_{\mathcal{D}}^{*} (\gamma_2, \epsilon)$$

CFG $\rightarrow$ PDA: Demostración $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$

Caso inductivo: $h = n$

Por HI, se tiene que para todo $\gamma_1, \gamma_2 \in Q^+$:

$$(Y \cdot \gamma_1, u) \vdash_{\mathcal{D}}^{*} (\gamma_1, \epsilon)$$

$$(Z \cdot \gamma_2, v) \vdash_{\mathcal{D}}^{*} (\gamma_2, \epsilon)$$

Para $\gamma \in Q^+$ **construimos** la siguiente ejecución de $\mathcal{D}$ sobre $w = uv$:

$$(X \cdot \gamma, uv) \vdash_{\mathcal{D}} (YZ \cdot \gamma, uv) \vdash_{\mathcal{D}}^{*} (Z \cdot \gamma, v) \vdash_{\mathcal{D}}^{*} (\gamma, \epsilon)$$

$\blacksquare$

CFG $\rightarrow$ PDA: Demostración $\mathcal{L}(\mathcal{D}) \subseteq \mathcal{L}(\mathcal{G})$

Para cada $w \in \mathcal{L}(\mathcal{D})$ debemos encontrar
un árbol de derivacíon $\mathcal{G}$ para w.

¿cómo encontramos un árbol de derivacíon para w?

Idea

Si tenemos una ejecución de $\mathcal{D}$ sobre w de la forma:

$$(X \cdot q_f, w) \ \vdash_{\mathcal{D}}^{*} \ (q_f, \epsilon)$$

entonces $X \overset{\star}{\underset{\mathcal{G}}{\Rightarrow}} w$

Inducción en la cantidad de pasos de la ejecución.

CFG $\rightarrow$ PDA: Demostración $\mathcal{L}(\mathcal{D}) \subseteq \mathcal{L}(\mathcal{G})$

Hipótesis de inducción

Para toda ejecución de $\mathcal{D}$ sobre w de largo k de la forma:

$$(X \cdot q_f, w) = (\gamma_0, w_0) \vdash_{\mathcal{D}} (\gamma_1, w_1) \vdash_{\mathcal{D}} \cdots \vdash_{\mathcal{D}} (\gamma_k, w_k) = (q_f, \epsilon)$$

entonces $X \overset{\star}{\underset{\mathcal{G}}{\Rightarrow}} w$.

Ejercicio: terminar la demostración.

Outline

Desde CFG a PDA

Desde PDA a CFG

Desde PDA a un CFG

Teorema

Para todo autómata apilador $\mathcal{P}$, existe una gramática libre de contexto $\mathcal{G}$:

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

Demostración

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

1. Convertir $\mathcal{P}$ a un PDA $\mathcal{P}'$ con **UN solo estado**.
2. Convertir $\mathcal{P}'$ a una gramática libre de contexto $\mathcal{G}$.

¿cómo hacemos cada paso?

PDA $\to$ CFG: Demostración

Paso 2: Covertir $\mathcal{P}'$ a una CFG $\mathcal{G}$

Sea $\mathcal{P}' = \big(\{q\}, \Sigma, \Gamma, \Delta, q, \bot, \{q\}\big)$ con **UN solo estado**.

Construimos la gramática:

$$\mathcal{G} \;=\; (V, \Sigma, P, \bot)$$

- $V = \Gamma$.
- Si $qA \xrightarrow{\epsilon} q\alpha \in \Delta$ entonces: $A \to \alpha \in P$
- Si $qA \xrightarrow{a} q\alpha \in \Delta$ entonces: $A \to a\,\alpha \in P$

Demostración: ejercicio.

PDA $\rightarrow$ CFG: Demostración

Paso 1: Covertir $\mathcal{P}$ a un PDA $\mathcal{P}'$ con UN solo estado

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

- ¿por qué NO necesitamos la información de los estados?
- ¿cómo guardamos la información de los estados en el stack?

Pregunta principal

"Si el PDA esta en el estado p y en el tope del stack hay una A entonces, ¿a cuál estado llegaré al remover A del stack?"

Solución

Podemos **adivinar** (no-determinismo) el estado que vamos a llegar cuando removamos A del stack.

PDA $\rightarrow$ CFG: Demostración

Paso 1: Covertir $\mathcal{P}$ a un PDA $\mathcal{P}'$ con UN solo estado

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

Sin perdida de generalidad podemos asumir que:

1. Todas las transiciones son de la forma:

$$qA \overset{c}{\rightarrow} pB_1B_2 \quad \text{o} \quad qA \overset{c}{\rightarrow} p\epsilon$$

   con $c \in \big(\Sigma \cup \{\epsilon\}\big)$. ¿por qué?

2. Existe $q_f \in Q$ tal que si $w \in \mathcal{L}(\mathcal{P})$ entonces:

$$(q_0\perp, w) \vdash^*_{\mathcal{D}} (q_f, \epsilon)$$

¿por qué?

Siempre llegamos al **mismo estado** q_f.

PDA $\rightarrow$ CFG: Demostración

Paso 1: Covertir $\mathcal{P}$ a un PDA $\mathcal{P}'$ con UN solo estado

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

Construimos el autómata apilador $\mathcal{P}'$ con **un solo estado**:

$$\mathcal{P}' = (\{q\}, \Sigma, \Gamma', \Delta', \{q\}, \perp', \{q\})$$

- $\Gamma' = Q \times \Gamma \times Q$.

  "$(p, A, q) \in \Gamma'$ si desde p leyendo A en el tope de stack llegamos a q al hacer pop de A"

- $\perp' = (q_0, \perp, q_f)$

  "El autómata parte en q_0 y al hacer pop de $\perp$ llegará a q_f"

PDA $\rightarrow$ CFG: Demostración

Paso 1: Covertir $\mathcal{P}$ a un PDA $\mathcal{P}'$ con UN solo estado

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

Construimos el autómata apilador $\mathcal{P}'$ con **un solo estado**:

$$\mathcal{P}' = (\{q\}, \Sigma, \Gamma', \Delta', \{q\}, \perp', \{q\})$$

- Si $pA \overset{c}{\rightarrow} p'B_1B_2 \;\in\; \Delta$ con $c \in \left(\Sigma \cup \{\epsilon\}\right)$, entonces **para todo** $p_1, p_2 \in Q$:

$$q\,(p, A, p_2) \;\overset{c}{\rightarrow}\; q\,(p', B_1, p_1)(p_1, B_2, p_2) \;\;\in\;\; \Delta'$$

- Si $pA \overset{c}{\rightarrow} p' \;\in\; \Delta$ con $c \in \left(\Sigma \cup \{\epsilon\}\right)$, entonces:

$$q\,(p, A, p') \;\overset{c}{\rightarrow}\; q \;\;\in\;\; \Delta'$$

PDA $\rightarrow$ CFG: Demostración $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{P}')$

Hipótesis de inducción (en el número de pasos n)

Para todo $p, p' \in Q$, $A \in \Gamma$, y $w \in \Sigma^*$ se cumple que:

$$(pA, w) \vdash^n_{\mathcal{P}} (p', \epsilon) \quad \text{si, y solo si,} \quad (q(p, A, p'), w) \vdash^n_{\mathcal{P}'} (q, \epsilon)$$

donde $\vdash^n_{\mathcal{P}}$ es la relación de **siguiente-paso** de $\mathcal{P}$ n-veces.

Si demostramos esta hipótesis habremos demostrado que $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{P}')$

¿por qué?

PDA $\rightarrow$ CFG: Demostración $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{P}')$

Caso base: $n = 1$

Para todo $p, p' \in Q$ y $A \in \Gamma$ se cumple que:

$$(pA, c) \vdash_{\mathcal{P}} (p', \epsilon) \quad \text{si, y solo si,} \quad (q(p, A, p'), c) \vdash_{\mathcal{P}'} (q, \epsilon)$$

para todo $c \in \big(\Sigma \cup \{\epsilon\}\big)$.

¿por qué?

PDA $\rightarrow$ CFG: Demostración $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{P}')$

Caso inductivo:

Sin perdida de generalidad,
suponga que $pA \xrightarrow{a} p_1A_1A_2$ y $w = auv$, entonces:

$$(pA, \underbrace{auv}_{w}) \vdash^n_{\mathcal{P}} (p', \epsilon) \quad \text{ssi} \quad (pA, auv) \vdash_{\mathcal{P}} (p_1A_1A_2, uv) \vdash^i_{\mathcal{P}} (p_2A_2, v) \vdash^j_{\mathcal{P}} (p', \epsilon)$$

$$\text{ssi} \quad (p_1A_1, u) \vdash^i_{\mathcal{P}} (p_2, \epsilon) \quad \text{y} \quad (p_2A_2, v) \vdash^j_{\mathcal{P}} (p', \epsilon)$$

$$\text{ssi} \quad (q(p_1, A_1, p_2), u) \vdash^i_{\mathcal{P}'} (q, \epsilon) \quad \text{y} \quad (q(p_2, A_2, p'), v) \vdash^j_{\mathcal{P}'} (q, \epsilon)$$

$$\text{ssi} \quad (q(p, A, p'), auv) \vdash_{\mathcal{P}} (q(p_1, A_1, p_2)(p_2, A_2, q)), uv) \vdash^{i+j}_{\mathcal{P}} (q, \epsilon)$$

$\blacksquare$