Pontificia Universidad Católica de Chile

Departamento de Ciencia de la Computación

Apunte IIC2223

Teoría de Autómatas y Lenguajes Formales

Autor
Cristóbal Rojas

En base a apuntes de
Prof. Cristian Riveros

16 de noviembre de 2022

Índice

1. Algoritmos para lenguajes libres de contexto

1.1. Autómatas apiladores

1.1.1. Versión normal

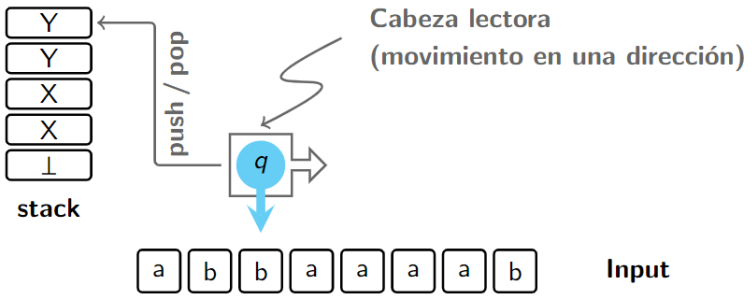


Figura 1: Idea de un autómata apilador

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

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

- Q es un conjunto finito de **estados**.
- Σ es el alfabeto del **input**.
- $q_0 \in Q$ es el estado **inicial**.
- F es el conjunto de estados **finales**.
- Γ es el alfabeto de **stack**.
- $\bot \in \Gamma$ es el símbolo **inicial del stack** (fondo).
- $\Delta \subseteq (Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma) \times (Q \times \Gamma^*)$ es una relación finita de transición.

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 $B_1 B_2 \cdots B_k$.

Intuitivamente, la transición **en vacío**:

$$\Big((p, \epsilon, A), (q, B_1 B_2 \cdots B_k)\Big) \in \Delta$$

si el autómata apilador está:

- en el estado p, *sin lectura de una letra*, y en el tope del stack hay una A,

entonces:

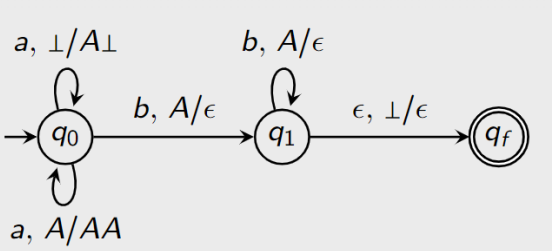
- cambia al estado q, y modifico el tope A por $B_1 B_2 \cdots B_k$.

Ejemplo 1.1

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

- $Q = \{q_0, q_1, q_f\}$, $\Sigma = \{a, b\}$, $\Gamma = \{A, \bot\}$ y Δ:

$$\begin{array}{ll} (q_0, a, \bot, q_0, A\,\bot) & q_0\,\bot \xrightarrow{a} q_0 A\,\bot \\ (q_0, a, A, q_0, AA) & q_0 A \xrightarrow{a} q_0 AA \\ (q_0, b, A, q_1, \epsilon) & q_0 A \xrightarrow{b} q_1 \\ (q_1, b, A, q_1, \epsilon) & q_1 A \xrightarrow{b} q_1 \\ (q_1, \epsilon, \bot, q_f, \epsilon) & q_1\,\bot \xrightarrow{\epsilon} q_f \end{array}$$



Notación. Dada 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.
- γ es el contenido del stack.
- w es el contenido del input.

Decimos que una configuración:

$$\boxed{(q \cdot \gamma, w) \in (Q \cdot \Gamma^*, \Sigma^*)}$$

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

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

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

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

- $w_1 = a \cdot w_2$

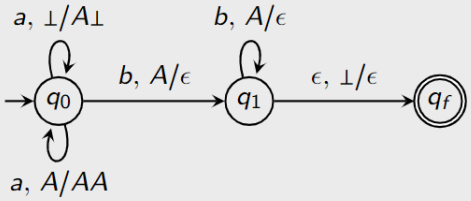
- $\gamma_1 = A \cdot \gamma$
- $\gamma_2 = \alpha \cdot \gamma$

Se define $\vdash_{\mathcal{P}}^*$ como la clausura **refleja** y **transitiva** de $\vdash_{\mathcal{P}}$. En otras palabras:

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

Ejemplo 1.2

Para la palabra $w = aaabbb$, 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	ϵ

Definiciones. $\mathcal{P}$ **acepta** w si, y sólo si, $(q_0 \perp, w) \vdash_{\mathcal{P}}^* (q_f, \epsilon)$ para algún $q_f \in F$.
El **lenguaje aceptado** por $\mathcal{P}$ se define como:

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

Ejemplo 1.3

El lenguaje aceptado por el PDA utilizado en los ejemplos anteriores es $\mathcal{L}(\mathcal{P}) = \{a^n b^n \mid n \geq 0\}$.

1.1.2. Versión alternativa

Esta definición de autómata apilador es poco común pero trae algunas ventajas:

- Es un modelo que ayuda a entender mejor los algoritmos de evaluación para gramáticas.
- Es un modelo menos estándar pero mucho más sencillo.
- Al profe Cristian le gustó y lo encontró interesante.

Definición. Un **PDA alternativo** es una estructura:

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

- Q es un conjunto finito de **estados**.
- Σ es el alfabeto del **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**.

Intuitivamente, la transición:

$$\boxed{\Big((A_1 \dots A_i), a, B_1 \dots B_j\Big) \in \Delta}$$

si el autómata apilador tiene:

- $A_1 \dots A_i$ en el tope del stack y leyendo a,

entonces:

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

En este tipo de autómata apilador, **no hay diferencia** entre estados y alfabeto del stack.

Definición. Una **configuración** de $\mathcal{D}$ es una tupla

$$\boxed{(q_1 \dots q_k, w) \in (Q^+, \Sigma^*)}$$

tal que:

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

Decimos que una configuración:

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

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

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

si, y sólo si, existe una transición $(\alpha, a, \beta) \in \Delta$ y $\gamma \in \Gamma^*$ 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 $\vdash_{\mathcal{D}}$.

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

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

Ejemplo 1.4

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

- $Q = \{\bot, q_0, q_1, q_f\}$ y Δ:

$(\bot, a, q_0\bot)$	$\bot \xrightarrow{a} q_0\bot$	$\bot$	
(q_0, a, q_0q_0)	$q_0 \xrightarrow{a} q_0q_0$	$q_0\ \bot$	a
(q_0, b, q_1)	$q_0 \xrightarrow{a} q_1$	$q_0\ q_0\ \bot$	a
(q_1q_0, b, q_1)	$q_1q_0 \xrightarrow{b} q_1$	$q_0\ q_0\ q_0\ \bot$	a
$(q_1\bot, \epsilon, q_f)$	$q_1\bot \xrightarrow{b} q_f$	$q_1\ q_0\ q_0\ \bot$	b
		$q_1\ q_0\ \bot$	b
		$q_1\ \bot$	b
		q_f	ϵ

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

Teorema 1

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

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

El teorema anterior nos dice que podemos usar ambos modelos de manera **equivalente**.

1.2. Autómatas apiladores vs gramáticas libres de contexto

¿En qué se parecen CFG a PDA?

Gramáticas libre de contexto	Autómatas apiladores
Terminales (Σ)	Alfabeto input (Σ)
Variables (V)	Alfabeto stack (Γ)
Producciones	Transiciones
$A \to \gamma$	$pA \xrightarrow{c} q\gamma$
$A \to a$	$pA \xrightarrow{c} q$
Derivaciones	Ejecuciones
Árbol de derivación	Stack

Figura 2: Gramáticas vs Autómatas apiladores

Teorema 2

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

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

1.2.1. Desde CFG a PDA

Partimos enunciado un teorema:

Teorema 3

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

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

Construcción $\mathcal{D}$ desde $\mathcal{G}$. Sea $\mathcal{G} = (V, \Sigma, P, S)$ una CFG. Construimos un PDA alternativo $\mathcal{D}$ que acepta $\mathcal{L}(\mathcal{G})$:

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

La relación de transición Δ se define como:

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

Demostración $\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{D})$. Debemos demostrar dos direcciones: $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{D})$ y $\mathcal{L}(\mathcal{D}) \subseteq \mathcal{L}(\mathcal{G})$.

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 esta ejecución? La idea es que para cada árbol de derivación $\mathcal{T}$ de $\mathcal{G}$ sobre w, construimos una ejecución de $\mathcal{D}$ sobre w que recorre el árbol $\mathcal{T}$ **en profundidad** (DFS). Por tanto, debemos usar **inducción** sobre la altura del árbol $\mathcal{T}$.

Hipótesis de inducción. Para todo árbol de derivación $\mathcal{T}$ 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 $\gamma \in Q^+$.

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 $X \to a$.

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

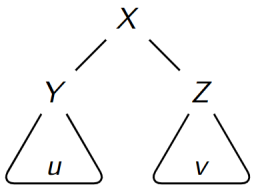
$$(X \cdot \gamma, a) \vdash_{\mathcal{D}} (a \cdot \gamma, a) \vdash_{\mathcal{D}} (\gamma, \epsilon)$$

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

Caso inductivo: $h = n$**.** Suponemos que el árbol de derivación $\mathcal{T}$ de $\mathcal{G}$ tiene **altura** n tal que:

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

Sin pérdida 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 $\gamma_1, \gamma_2 \in Q^+$:

$$(Y \cdot \gamma_1, u) \vdash^*_{\mathcal{D}} (\gamma_1, \epsilon)$$
$$(Z \cdot \gamma_2, v) \vdash^*_{\mathcal{D}} (\gamma_2, \epsilon)$$

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

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

■

La demostración de $\mathcal{L}(\mathcal{D}) \subseteq \mathcal{L}(\mathcal{G})$ se deja como ejercicio propuesto al lector.

1.2.2. Desde PDA a CFG

Partimos enunciando el siguiente teorema:

Teorema 4

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

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

Demostración $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{G})$**.** Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \bot, F)$ un PDA (normal). Los pasos a seguir son:

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

Paso 1. Sea $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, q_0, \bot, F)$ un PDA. Podemos analizar:

- ¿Por qué NO necesitamos la información de los estados?
- ¿Cómo guardamos la información de los estados en el stack?

Esto conlleva a la siguiente pregunta: *Si el PDA está en el estado p y en el tope del stack hay una A, ¿a cuál estado llegaré al remover A del stack?*

La solución a esta pregunta es que podemos **adivinar** (no-determinismo) el estado que vamos a llegar cuando removamos A del stack.

Sin pérdida de generalidad, podemos asumir que

1. Todas las transiciones son de la forma:

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

con $c \in (\Sigma \cup \{e\})$.

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

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

Estos dos puntos nos aseguran que siempre llegamos al **mismo estado** q_f. Luego, construimos el autómata apilador $\mathcal{P}'$ con **un solo estado**:

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

- $\Gamma' = Q \times \gamma \times Q$.
  "$(p, A, q) \in \Gamma'$ si desde p leyendo A en el tope del stack llegamos a q al hacer pop de A".
- $\bot' = (q_0, \bot, q_f)$.
  "El autómata parte en q_0 y al hacer pop de $\bot$ llegará a q_f".
- Si $pA \xrightarrow{c} p'B_1B_2 \in \Delta$ con $c \in (\Sigma \cup \{\epsilon\})$, entonces **para todo** $p_1, p_2 \in Q$:

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

- Si $pA \xrightarrow{c} p' \in \Delta$ con $c \in (\Sigma \cup \{\epsilon\})$, entonces:

$$q\,(p, A, p') \xrightarrow{c} q \in \Delta'$$

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?

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

Caso inductivo. Sin pérdida de generalidad, suponga que $pA \xrightarrow{a} p_1A_1A_2$ y $w = auv$, entonces

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

$$\begin{aligned}
&\text{ssi } (p_1A_1, u) \vdash^i_{\mathcal{P}} (p_2, \epsilon) \text{ y } \quad (p_2A_2, v) \vdash^j_{\mathcal{P}} (p', \epsilon) \\
&\text{ssi } (q\,(p_1, A_1, p_2), u) \vdash^i_{\mathcal{P}'} (q, \epsilon) \text{ y } \quad (q\,(p_2, A_2, p'), v) \vdash^j_{\mathcal{P}'} (q, \epsilon) \\
&\text{ssi } (q\,(p, A, p'), auv) \vdash_{\mathcal{P}} (q\,(p_1, A_1, p_2)\,(p_2, A_2, q)), uv) \vdash^{i+j}_{\mathcal{P}} (q, \epsilon)
\end{aligned}$$

■

Paso 2. Sea $\mathcal{P} = (\{q\}, \Sigma, \Gamma, \Delta, q, \bot, \{q\})$ un PDA con **UN solo estado**. Contruimos 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 $A \to a\alpha \in P$

La demostración de este paso queda como ejercicio propuesto al lector.