

Autómata característico

Clase 26

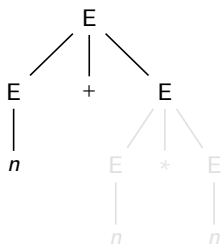
IIC 2223

Prof. Cristian Riveros

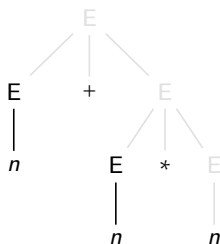
Bottom-up parsing (recordatorio)

$$E \rightarrow E + E \mid E * E \mid n$$

Top-down parsing



Bottom-up parsing



Cambio en la notación de stack (recordatorio)

Notación (desde ahora y hasta el término del curso)

Para un stack $q_0 \dots q_{n-1} q_n$, usaremos a q_n como el **tope de stack** y $q_0 \dots q_{n-1}$ como la cola de stack.

Para un PDA \mathcal{P} y una transición $(\overbrace{p_0 \dots p_i}^{\alpha}, a, \overbrace{q_0 \dots q_j}^{\beta})$ de \mathcal{P} , p_i es el símbolo en el tope del stack y q_j será el símbolo del tope del stack resultante.

La relación $\vdash_{\mathcal{P}}$ de **siguiente-paso** quedará como:

$$(\gamma \cdot \alpha, a \cdot u) \vdash_{\mathcal{P}} (\gamma \cdot \beta, u)$$

Como **recordatorio**, (algunas veces) marcaremos el tope del stack:

$$q_0 \dots q_{n-1} q_n^{\downarrow}$$

Gramática aumentada (recordatorio)

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto reducida.

Definición

Se define la **gramática aumentada** de \mathcal{G} como:

$$\mathcal{G}' = (V \cup \{S'\}, \Sigma, P \cup \{S' \rightarrow S\}, S')$$

tal que S' es una variable nueva con $S' \notin V$.

“Usaremos S' para saber cuando hemos llegado al final de una derivación.”

Desde ahora, trabajaremos siempre con una gramática aumentada.

Apilador bottom-up (recordatorio)

Sea $\mathcal{G}' = (V, \Sigma, P, S')$ una gramática libre de contexto aumentada.

Definición

El **apilador bottom-up** de \mathcal{G} (bottom-up-PDA) es un PDA alternativo:

$$\mathcal{P}_{\uparrow} = (\underbrace{V \cup \Sigma \cup \{\$\}}_Q, \Sigma, \Delta, \underbrace{\$}_{q_0}, \underbrace{\{S'\}}_F)$$

Tres tipos de transiciones en $\Delta \subseteq Q^+ \times (\Sigma \cup \{\epsilon\}) \times Q^*$:

Shift: $q \xrightarrow{a} qa$ para $q \in V \cup \Sigma \cup \{\$\}$ y $a \in \Sigma$

Reduce: $\alpha \xrightarrow{\epsilon} X$ si $X \rightarrow \alpha \in P$

Termino: $\$S' \xrightarrow{\epsilon} S'$

Correctitud de apilador bottom-up (recordatorio)

Teorema

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto. Entonces:

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

Corolario

Si $(\$, xy) \vdash_{\mathcal{P}_{\uparrow}}^* (\$ \alpha, y) \vdash_{\mathcal{P}_{\uparrow}}^* (\$ S', \epsilon)$, entonces $S' \xRightarrow[\text{rm}]{*} \alpha y \xRightarrow[\text{rm}]{*} xy$.

Las **configuraciones** del apilador bottom-up
son **derivaciones** por la derecha de \mathcal{G} .

Prefijos viables, reducibles y handles (recordatorio)

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática y \mathcal{G}' su gramática aumentada.

Definiciones

- $\alpha \in (V \cup \Sigma)^*$ es un **prefijo viable** de \mathcal{G} ssi
existe una derivación $S' \xRightarrow[\text{rm}]{*} \alpha\beta w$ tal que $\beta \in (V \cup \Sigma)^*$ y $w \in \Sigma^*$.
- $\alpha \cdot \beta \in (V \cup \Sigma)^*$ es **reducible** a $\alpha \cdot X$ ssi
existe una derivación $S' \xRightarrow[\text{rm}]{*} \alpha X w \xRightarrow[\text{rm}]{*} \alpha\beta w$ con $w \in \Sigma^*$.
En cuyo caso, decimos que $X \rightarrow \beta$ es un **handle** de $\alpha\beta$.
- $\alpha \cdot \beta \in (V \cup \Sigma)^*$ es un **prefijo reducible** ssi
 $\alpha \cdot \beta$ es un prefijo viable y existe X tal que $\alpha \cdot \beta$ es reducible a $\alpha \cdot X$.

Prefijos viables y prefijos reducibles
representan **configuraciones válidas** del apilador bottom-up.

¿podemos usar el bottom-up PDA para hacer parsing?

Problemas

1. Conflicto **Shift-Reduce**.
2. Conflicto **Reduce-Reduce**.
3. Configuraciones **no-viables**.

Ejemplo: ¿es esta configuración viable?

$$S \rightarrow ab \mid B$$
$$B \rightarrow b$$

Stack	Input	Operaciones
\$	ab	
\$a	b	shift
\$ab	.	shift
\$aB	.	reduce $b \xrightarrow{\epsilon} B$
×	×	×

Outline

Autómata característico

Determinización

LR-Parser

¿cómo usar LR-Parser?

Outline

Autómata característico

Determinización

LR-Parser

¿cómo usar LR-Parser?

Items de una gramática

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto.

Definición

Un **item** de \mathcal{G} es un triple:

$$(X, \alpha, \beta) \in V \times (V \cup \Sigma)^* \times (V \cup \Sigma)^*$$

tal que $X \rightarrow \alpha\beta$ es una regla en P .

Por conveniencia, usaremos un item como: $[X \rightarrow \alpha.\beta]$.

Ejemplos de items

$$E \rightarrow E + E \mid E * E \mid n$$

- $[E \rightarrow E. + E]$
- $[E \rightarrow .n]$
- $[E \rightarrow E * E.]$

Items de una gramática

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto.

Definición

Un **item** de \mathcal{G} es un triple:

$$(X, \alpha, \beta) \in V \times (V \cup \Sigma)^* \times (V \cup \Sigma)^*$$

tal que $X \rightarrow \alpha\beta$ es una regla en P .

Por conveniencia, usaremos un item como: $[X \rightarrow \alpha.\beta]$.

1. Un item $[X \rightarrow \alpha.]$ decimos que esta **completado**.
2. Se define $\text{Items}_{\mathcal{G}}$ como el conjunto de todos los items de \mathcal{G} .

¿cuál es el tamaño de $|\text{Items}_{\mathcal{G}}|$?

Autómata característico

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto cualquiera.

Definición

El **autómata característico** de \mathcal{G} es un ϵ -NFA:

$$\text{char}[\mathcal{G}] = (Q_0, \Sigma', \Delta_0, I_0, F_0)$$

- $Q_0 = \text{Items}_{\mathcal{G}} \cup \{[S' \rightarrow .S], [S' \rightarrow S.]\}$
- $\Sigma' = V \cup \Sigma$
- $I_0 = \{[S' \rightarrow .S]\}$
- $F_0 = \{[X \rightarrow \alpha.] \mid [X \rightarrow \alpha.] \in Q_0\}$

Autómata característico

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto cualquiera.

Definición

El **autómata característico** de \mathcal{G} es un ϵ -NFA:

$$\text{char}[\mathcal{G}] = (\underbrace{\text{Items}_{\mathcal{G}} \cup \{[S' \rightarrow .S], [S' \rightarrow S.]\}}_{Q_0}, V \cup \Sigma, \Delta_0, \underbrace{[S' \rightarrow .S]}_{I_0}, \underbrace{\{[X \rightarrow \alpha.]\}}_{F_0})$$

Dos tipos de transiciones en Δ_0 :

$$\begin{array}{ll} \text{Bajar:} & [X \rightarrow \alpha.Y\beta] \xrightarrow{\epsilon} [Y \rightarrow .\gamma] \quad \text{para cada } X \rightarrow \alpha Y \beta \in Q_0 \\ & \text{y } Y \rightarrow \gamma \in P \end{array}$$

$$\text{Avanzar:} \quad [X \rightarrow \alpha.Y\beta] \xrightarrow{Y} [X \rightarrow \alpha Y.\beta] \quad Y \in V \cup \Sigma$$

$\text{char}[\mathcal{G}]$ “navega” por un árbol de derivación.

Autómata característico

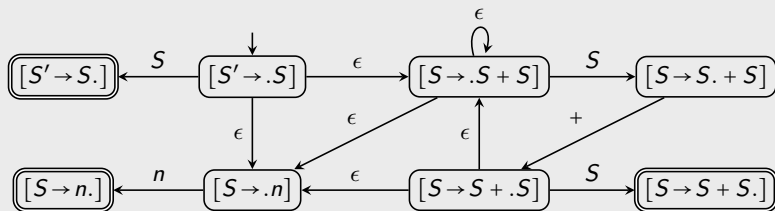
Definición

$$\text{char}[\mathcal{G}] = (\text{Items}_{\mathcal{G}} \cup \{[S' \rightarrow .S], [S' \rightarrow S.]\}, V \cup \Sigma, \Delta_0, [S' \rightarrow .S], \{[X \rightarrow \alpha.]\})$$

Bajar: $[X \rightarrow \alpha.Y\beta] \xrightarrow{\epsilon} [Y \rightarrow .\gamma]$ para cada $X \rightarrow \alpha Y \beta \in Q_0$
y $Y \rightarrow \gamma \in P$

Avanzar: $[X \rightarrow \alpha.Y\beta] \xrightarrow{Y} [X \rightarrow \alpha Y.\beta]$ $Y \in V \cup \Sigma$

Ejemplo: $S \rightarrow S + S \mid n$



Autómata característico y prefijos viables

Teorema

1. $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ si, y solo si, γ es un **prefijo viable** de \mathcal{G} .

Demostración (\Rightarrow)

PD: Si $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$, entonces $S' \xRightarrow[\text{rm}]{*} \gamma\beta w$ para algún $w \in \Sigma^*$.

Caso inductivo: Como $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$, entonces para $\gamma = \gamma'\alpha$:

$$[S' \rightarrow .S] \xrightarrow{\gamma'} [Y \rightarrow \alpha'.X\beta'] \xrightarrow{\epsilon} [X \rightarrow .\alpha\beta] \xrightarrow{\alpha} [X \rightarrow \alpha.\beta]$$

Por **Hipótesis de Inducción:** $S' \xRightarrow[\text{rm}]{*} \gamma'X\beta'v$ para algún $v \in \Sigma'$.

Como \mathcal{G} es reducida, entonces $\beta' \xRightarrow[\text{rm}]{*} u$ para algún $u \in \Sigma^*$ y tenemos:

$$S' \xRightarrow[\text{rm}]{*} \gamma'X\beta'v \xRightarrow[\text{rm}]{*} \gamma'X\overbrace{uv}^w \xRightarrow[\text{rm}]{*} \overbrace{\gamma'\alpha}^{\gamma}\beta w$$



Autómata característico y prefijos viables

Teorema

1. $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ si, y solo si, γ es un **prefijo viable** de \mathcal{G} .

Demostración (\Leftarrow)

PD: Si $S' \xRightarrow[\text{rm}]{*} \gamma' X w \Rightarrow_{\text{rm}} \gamma' \alpha \beta w$, entonces $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ con $\gamma = \gamma' \alpha$.

¿por qué esto es suficiente para demostrar (\Leftarrow)?

Autómata característico y prefijos viables

Teorema

1. $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ si, y solo si, γ es un **prefijo viable** de \mathcal{G} .

Demostración (\Leftarrow)

PD: Si $S' \xRightarrow[\text{rm}]{*} \gamma' X w \Rightarrow_{\text{rm}} \gamma' \alpha \beta w$, entonces $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ con $\gamma = \gamma' \alpha$.

Demostración (\Leftarrow) usando **PD**

Sea γ un prefijo viable de \mathcal{G} y sea $S' \xRightarrow[\text{rm}]{*} \gamma \beta w$ la derivación de **menor largo** donde aparece γ . Entonces:

$$S' \xRightarrow[\text{rm}]{*} \gamma' X w \Rightarrow_{\text{rm}} \overbrace{\gamma' \alpha}^{\gamma} \beta w$$

y por **PD** tenemos que $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$.

(¿por qué $X \rightarrow \alpha\beta$?)

Autómata característico y prefijos viables

Teorema

1. $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ si, y solo si, γ es un **prefijo viable** de \mathcal{G} .

Demostración (\Leftarrow)


PD: Si $S' \xRightarrow[\text{rm}]{*} \gamma' X w \Rightarrow_{\text{rm}} \gamma' \alpha \beta w$, entonces $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ con $\gamma = \gamma' \alpha$.

Caso inductivo: Suponga que:

$$S' \xRightarrow[\text{rm}]{*} \gamma_1 Y v \Rightarrow_{\text{rm}} \overbrace{\gamma_1 \gamma_2}^{\gamma'} X \overbrace{uv}^w \Rightarrow_{\text{rm}} \gamma' \alpha \beta w$$

Por HI, tenemos que $[S' \rightarrow .S] \xrightarrow{\gamma'} [Y \rightarrow \gamma_2.Xu]$. Por definición de $\text{char}[\mathcal{G}]$:

$$[Y \rightarrow \gamma_2.Xu] \xrightarrow{\epsilon} [X \rightarrow .\alpha\beta] \xrightarrow{\alpha} [X \rightarrow \alpha.\beta]$$

Por lo tanto, $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ con $\gamma = \gamma' \alpha$. 

Autómata característico y prefijos reducibles

Teorema

1. $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.\beta]$ si, y solo si, γ es un **prefijo viable** de \mathcal{G} .
2. $\mathcal{L}(\text{char}[\mathcal{G}]) = \{\gamma \mid \gamma \text{ es un prefijo reducible de } \mathcal{G}\}$

Demostración

Sea $\gamma \in \mathcal{L}(\text{char}[\mathcal{G}])$ tal que $[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.]$.

Por 1., γ es un **prefijo viable**.

Por demostración (\Rightarrow) sabemos que $\gamma = \gamma'\alpha$ para algún γ' .

Por lo tanto, $X \rightarrow \alpha$ es un **handle** para γ y γ es un **prefijo reducible**.

$\text{char}[\mathcal{G}]$ sirve para **identificar** las configuraciones viables del bottom-up PDA

Outline

Autómata característico

Determinización

LR-Parser

¿cómo usar LR-Parser?

Determinización de autómata característico

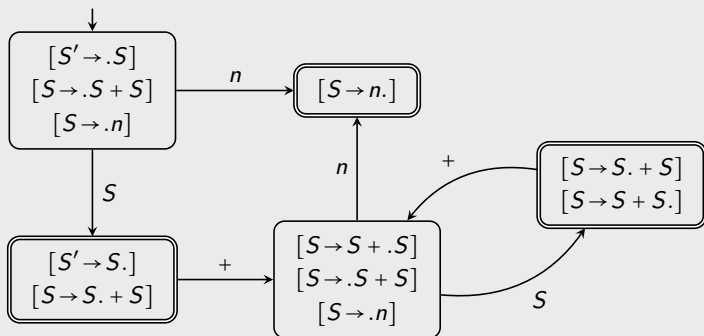
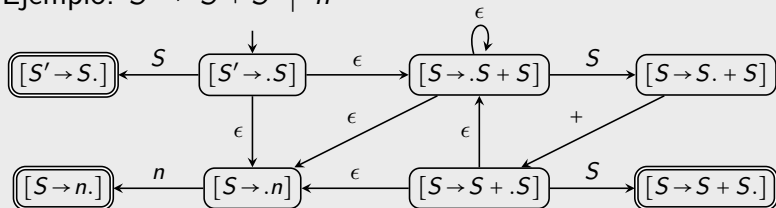
Sea $\text{det}[\mathcal{G}] = (Q_0^{\text{det}}, V \cup \Sigma, \Delta_0^{\text{det}}, I_0^{\text{det}}, F_0^{\text{det}})$ la determinización de $\text{char}[\mathcal{G}]$:

- $Q_0^{\text{det}} = \{ C \mid C \subseteq \text{Items}_{\mathcal{G}} \cup \{ [S' \rightarrow .S], [S' \rightarrow S.] \} \}$
- $I_0^{\text{det}} = \{ [X \rightarrow .\alpha] \mid ([S' \rightarrow .S], \epsilon) \vdash_{\text{char}[\mathcal{G}]}^* ([X \rightarrow .\alpha], \epsilon) \}$
- $\Delta_0^{\text{det}} = \{ (C_1, Y, C_2) \mid \text{para todo } g_2 \in C_2, \text{ existe } g_1 \in C_1 \text{ y } g \in Q_0, \\ (g_1, Y) \vdash_{\text{char}[\mathcal{G}]} (g, \epsilon) \vdash_{\text{char}[\mathcal{G}]}^* (g_2, \epsilon) \}$
- $F_0^{\text{det}} = \{ C \mid \text{existe } [X \rightarrow \alpha.] \in C \}$

Hacemos la determinización usual, pero eliminando ϵ -transiciones.

Determinización de autómata característico

Ejemplo: $S \rightarrow S + S \mid n$



Determinización de autómatas característicos

Propiedades

1. $\mathcal{L}(\text{det}[\mathcal{G}]) = \{\gamma \mid \gamma \text{ es un prefijo reducible de } \mathcal{G}\}$
2. Si $I_0^{\text{det}} \xrightarrow{\gamma} C$ y $[X \rightarrow \alpha.] \in C$, entonces $X \rightarrow \alpha$ es un **handle** para γ .

Demostración 2.

Si $I_0^{\text{det}} \xrightarrow{\gamma} C$ y $[X \rightarrow \alpha.] \in C$, entonces $\gamma \in \mathcal{L}(\text{det}[\mathcal{G}])$ y γ es un prefijo reducible de \mathcal{G} (por definición de $\text{det}[\mathcal{G}]$). Más aún,

$$[S' \rightarrow .S] \xrightarrow{\gamma} [X \rightarrow \alpha.]$$

Por Demostración (\Rightarrow) sabemos que $\gamma = \gamma' \cdot \alpha$ y $X \rightarrow \alpha$ es un **handle** para γ .

Podemos encontrar **todos los handles** de un prefijo viable mirando el último estado de $\text{det}[\mathcal{G}]$.

Outline

Autómata característico

Determinización

LR-Parser

¿cómo usar LR-Parser?

LR parser

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto cualquiera.

Definición

El **LR-autómata apilador** de \mathcal{G} (LR-PDA) es un PDA alternativo:

$$\text{LR}[\mathcal{G}] = (Q_1, \Sigma, \Delta_1, l_1, F_1)$$

$$\blacksquare Q_1 = Q_0^{\text{det}} \cup \{f\}$$

$$\blacksquare l_1 = l_0^{\text{det}}$$

$$\blacksquare F_1 = \{f\}$$

LR parser

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto cualquiera.

Definición

El **LR-autómata apilador** de \mathcal{G} (LR-PDA) es un PDA alternativo:

$$\text{LR}[\mathcal{G}] = (\underbrace{Q_0^{\text{det}} \cup \{f\}}_{Q_1}, \Sigma, \Delta_1, \underbrace{l_0^{\text{det}}}_{l_1}, \underbrace{\{f\}}_{F_1})$$

Tres tipos de transiciones en $\Delta_1 \subseteq Q_1^+ \times \Sigma \times Q_1^*$ para $C, C_1, \dots, C_n \in Q_0^{\text{det}}$:

Shift: $C \xrightarrow{a} C \cdot \Delta_0^{\text{det}}(C, a)$ si $\Delta_0^{\text{det}}(C, a) \neq \emptyset$

Reduce: $C C_1 \dots C_n \xrightarrow{\epsilon} C \cdot \Delta_0^{\text{det}}(C, X)$ si $[X \rightarrow \alpha.] \in C_n$ y $n = |\alpha|$

Término: $l_0^{\text{det}} C \xrightarrow{\epsilon} f$ si $[S' \rightarrow S.] \in q$

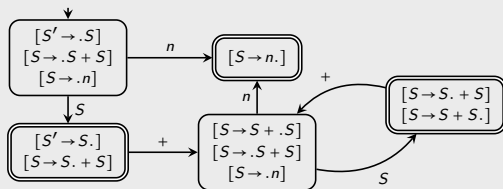
LR parser

Shift: $C \xrightarrow{a} C \cdot \Delta_0^{\text{det}}(C, a)$ si $\Delta_0^{\text{det}}(C, a) \neq \emptyset$

Reduce: $C C_1 \dots C_n \xrightarrow{\epsilon} C \cdot \Delta_0^{\text{det}}(C, X)$ si $[X \rightarrow \alpha.] \in C_n$ y $n = |\alpha|$

Término: $I_0^{\text{det}} C \xrightarrow{\epsilon} f$ si $[S' \rightarrow S.] \in q$

Ejemplo: $S \rightarrow S + S \mid n$



- $\{\dots, [S \rightarrow .n]\} \xrightarrow{n} \{\dots, [S \rightarrow .n]\} \{[S \rightarrow n.]\}$
- $\{\dots, [S \rightarrow .S + S], [S \rightarrow .n]\} \{[S \rightarrow n.]\} \xrightarrow{\epsilon} \{\dots, [S \rightarrow .S + S], [S \rightarrow .n]\} \{\dots, [S \rightarrow S + S.]\}$

¿qué representa el stack del LR-PDA?

Definición LR-PDA

$$\text{LR}[\mathcal{G}] = (\underbrace{Q_0^{\text{det}} \cup \{f\}}_{Q_1}, \Sigma, \Delta_1, \underbrace{l_0^{\text{det}}}_{l_1}, \underbrace{\{f\}}_{F_1})$$

Proposición

Si $(l_0^{\text{det}}, uv) \vdash_{\text{LR}[\mathcal{G}]}^* (l_0^{\text{det}} C_1 \dots C_n, v)$, entonces existe $\gamma_1 \dots \gamma_n \in (V \cup \Sigma)^*$:

$$l_0^{\text{det}} \xrightarrow{\gamma_1} C_1 \xrightarrow{\gamma_2} \dots \xrightarrow{\gamma_n} C_n$$

es una **ejecución** de $\text{char}[\mathcal{G}]^{\text{det}}$ sobre $\gamma_1 \dots \gamma_n$.

El stack $l_0^{\text{det}} C_1 \dots C_n$ de una configuración cualquiera corresponde a una **ejecución** de la determinización $\text{char}[\mathcal{G}]$.

Relación entre LR-PDA y bottom-up PDA

Sea \mathcal{G} una gramática libre de contexto.

Teorema

Sea $\gamma = \gamma_1 \dots \gamma_n \in (V \cup \Sigma)^*$ y

$I_0^{\text{det}} \xrightarrow{\gamma_1} C_1 \xrightarrow{\gamma_2} \dots \xrightarrow{\gamma_n} C_n$ la ejecución de $\text{det}[\mathcal{G}]$ sobre γ . Para todo $u, v \in \Sigma^*$:

- $(\$, uv) \vdash_{\mathcal{P}_{\uparrow}}^* (\$ \gamma, v) \vdash_{\mathcal{P}_{\uparrow}}^* (\$ S', \epsilon)$ si, y solo si,
- $C_n \neq \emptyset$ y $(I_0^{\text{det}}, uv) \vdash_{\text{LR}[\mathcal{G}]}^* (I_0^{\text{det}} C_1 \dots C_n, v) \vdash_{\text{LR}[\mathcal{G}]}^* (f, \epsilon)$.

En particular, $\mathcal{L}(\mathcal{G}) = \mathcal{L}(\text{LR}[\mathcal{G}])$.

En otras palabras, $\text{LR}[\mathcal{G}]$ simula a \mathcal{P}_{\uparrow} llevando solo **configuraciones viables**.

(NO haremos esta demostración en clases)

Outline

Autómata característico

Determinización

LR-Parser

¿cómo usar LR-Parser?

LR parser y su uso

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto cualquiera.

Definición

El **LR-autómata apilador** de \mathcal{G} (LR-PDA) es un PDA alternativo:

$$\text{LR}[\mathcal{G}] = (\underbrace{Q_0^{\text{det}} \cup \{f\}}_{Q_1}, \Sigma, \Delta_1, \underbrace{l_0^{\text{det}}}_{l_1}, \underbrace{\{f\}}_{F_1})$$

Tres tipos de transiciones en $\Delta_1 \subseteq Q_1^+ \times \Sigma \times Q_1^*$ para $C, C_1, \dots, C_n \in Q_0^{\text{det}}$:

Shift: $C \xrightarrow{a} C \cdot \Delta_0^{\text{det}}(C, a)$ si $\Delta_0^{\text{det}}(C, a) \neq \emptyset$

Reduce: $C C_1 \dots C_n \xrightarrow{\epsilon} C \cdot \Delta_0^{\text{det}}(C, X)$ si $[X \rightarrow \alpha.] \in C_n$ y $n = |\alpha|$

Término: $l_0^{\text{det}} C \xrightarrow{\epsilon} f$ si $[S' \rightarrow S.] \in q$

¿dónde está el **no-determinismo** en el LR-PDA?

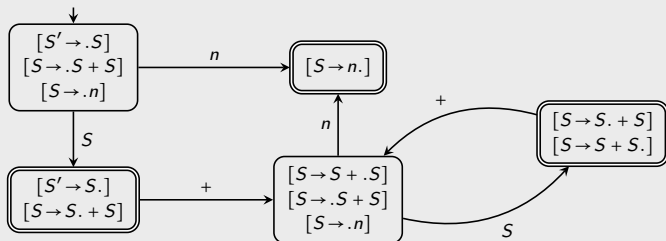
Determinación de conflictos en LR-PDA

Definición

Para un estado $C \in Q_0^{\text{det}}$ decimos que:

1. C tiene un conflicto **Shift-Reduce** si existen $[X \rightarrow \alpha.a\beta]$ y $[Y \rightarrow \gamma.]$ en C , simultáneamente.
2. C tiene un conflicto **Reduce-Reduce** si existen $[Y_1 \rightarrow \gamma_1.]$ y $[Y_2 \rightarrow \gamma_2.]$ distintos en C .

¿qué conflicto tiene nuestra determinización de $\text{char}[\mathcal{G}]$?



Determinación de conflictos en LR-PDA

Definición

Para un estado $C \in Q_0^{\text{det}}$ decimos que:

1. C tiene un conflicto **Shift-Reduce** si existen $[X \rightarrow \alpha.a\beta]$ y $[Y \rightarrow \gamma.]$ en C , simultáneamente.
2. C tiene un conflicto **Reduce-Reduce** si existen $[Y_1 \rightarrow \gamma_1.]$ y $[Y_2 \rightarrow \gamma_2.]$ distintos en C .

Decimos que C es **libre de conflictos** si

NO tiene un conflicto Shift-Reduce o Reduce-Reduce.

Notar que...

Si C es libre de conflictos para todo $C \in Q_0^{\text{det}}$,

entonces $\text{LR}[\mathcal{G}]$ es un autómata apilador **determinista**.

Determinación de conflictos en LR-PDA

Notar que...

Si C es libre de conflictos para todo $C \in Q_0^{\text{det}}$,
entonces $\text{LR}[\mathcal{G}]$ es un autómata apilador **determinista**.

Shift: $C \xrightarrow{a} C \cdot \Delta_0^{\text{det}}(C, a)$ si $\Delta_0^{\text{det}}(C, a) \neq \emptyset$

Reduce: $C C_1 \dots C_n \xrightarrow{\epsilon} C \cdot \Delta_0^{\text{det}}(C, X)$ si $[X \rightarrow \alpha.] \in C_n$ y $n = |\alpha|$

Término: $I_0^{\text{det}} C \xrightarrow{\epsilon} f$ si $[S' \rightarrow S.] \in q$

Por lo tanto, podemos usar $\text{LR}[\mathcal{G}]$ para hacer **parsing** y encontrar una **derivación por la derecha (invertida)** de \mathcal{G} para cada input.