

Autómatas y Lenguajes Formales

Tema 8: Gramáticas, jerarquía de Chomsky, ambigüedad

Tema 8: Gramáticas, jerarquía de Chomsky, ambigüedad

Dr. Favio Ezequiel Miranda Perea
favio@ciencias.unam.mx

Facultad de Ciencias UNAM¹

27 de abril de 2019



¹ Con el apoyo del proyecto PAPIME PE102117

Gramáticas

Introducción

- Un mecanismo relevante para generar un lenguaje es mediante el concepto de gramática formal.



Gramáticas

Introducción

- Un mecanismo relevante para generar un lenguaje es mediante el concepto de gramática formal.
- Las gramáticas formales fueron introducidas por Noam Chomsky en 1956.



Gramáticas

Introducción

- Un mecanismo relevante para generar un lenguaje es mediante el concepto de gramática formal.
- Las gramáticas formales fueron introducidas por Noam Chomsky en 1956.
- La intención era tener un modelo para la descripción de lenguajes naturales.



Gramáticas

Introducción

- Un mecanismo relevante para generar un lenguaje es mediante el concepto de gramática formal.
- Las gramáticas formales fueron introducidas por Noam Chomsky en 1956.
- La intención era tener un modelo para la descripción de lenguajes naturales.
- Posteriormente se utilizaron como herramienta para presentar la sintaxis de lenguajes de programación y para el diseño de analizadores léxicos de compiladores



Gramáticas

Definición

Una gramática es una cuaterna $G = \langle V, T, S, P \rangle$ tal que:



Gramáticas

Definición

Una gramática es una cuaterna $G = \langle V, T, S, P \rangle$ tal que:

- V es un alfabeto de **variables** o **símbolos no-terminales**, los cuales se denotan con mayúsculas A, B, C, \dots



Gramáticas

Definición

Una gramática es una cuaterna $G = \langle V, T, S, P \rangle$ tal que:

- V es un alfabeto de **variables** o **símbolos no-terminales**, los cuales se denotan con mayúsculas A, B, C, \dots
- T es un alfabeto de **símbolos terminales**, los cuales se denotan con minúsculas a, b, c, \dots . Además se requiere $T \cap V = \emptyset$.



Gramáticas

Definición

Una gramática es una cuaterna $G = \langle V, T, S, P \rangle$ tal que:

- V es un alfabeto de **variables** o **símbolos no-terminales**, los cuales se denotan con mayúsculas A, B, C, \dots
- T es un alfabeto de **símbolos terminales**, los cuales se denotan con minúsculas a, b, c, \dots . Además se requiere $T \cap V = \emptyset$.
- $S \in V$ es una variable distinguida llamada el **símbolo inicial**.



Gramáticas

Definición

Una gramática es una cuaterna $G = \langle V, T, S, P \rangle$ tal que:

- V es un alfabeto de **variables** o **símbolos no-terminales**, los cuales se denotan con mayúsculas A, B, C, \dots
- T es un alfabeto de **símbolos terminales**, los cuales se denotan con minúsculas a, b, c, \dots . Además se requiere $T \cap V = \emptyset$.
- $S \in V$ es una variable distinguida llamada el **símbolo inicial**.
- P es un conjunto finito de reglas de reescritura, llamadas **reglas de producción** o producciones.



Reglas de Producción

Gramáticas

El conjunto de reglas de producción P es un conjunto finito de pares $\langle \alpha, \beta \rangle$ tales que



Reglas de Producción

Gramáticas

El conjunto de reglas de producción P es un conjunto finito de pares $\langle \alpha, \beta \rangle$ tales que

- $\alpha \in (V \cup T)^* - T^*$. Es decir, α es una cadena de símbolos terminales ó no terminales, con al menos un símbolo no-terminal.



Reglas de Producción

Gramáticas

El conjunto de reglas de producción P es un conjunto finito de pares $\langle \alpha, \beta \rangle$ tales que

- $\alpha \in (V \cup T)^* - T^*$. Es decir, α es una cadena de símbolos terminales ó no terminales, con al menos un símbolo no-terminal.
- $\beta \in (V \cup T)^*$. Es decir, β es una cadena de símbolos de $V \cup T$, los cuales podrían ser todos terminales.



Reglas de Producción

Gramáticas

El conjunto de reglas de producción P es un conjunto finito de pares $\langle \alpha, \beta \rangle$ tales que

- $\alpha \in (V \cup T)^* - T^*$. Es decir, α es una cadena de símbolos terminales ó no terminales, con al menos un símbolo no-terminal.
- $\beta \in (V \cup T)^*$. Es decir, β es una cadena de símbolos de $V \cup T$, los cuales podrían ser todos terminales.
- Usualmente en vez de escribir $\langle \alpha, \beta \rangle \in P$ escribimos

$$\alpha \rightarrow \beta$$



Reglas de Producción

Gramáticas

El conjunto de reglas de producción P es un conjunto finito de pares $\langle \alpha, \beta \rangle$ tales que

- $\alpha \in (V \cup T)^* - T^*$. Es decir, α es una cadena de símbolos terminales ó no terminales, con al menos un símbolo no-terminal.
- $\beta \in (V \cup T)^*$. Es decir, β es una cadena de símbolos de $V \cup T$, los cuales podrían ser todos terminales.
- Usualmente en vez de escribir $\langle \alpha, \beta \rangle \in P$ escribimos

$$\alpha \rightarrow \beta$$

y decimos que α produce a β , o que α se reescribe en β .



Derivaciones

Generación de cadenas

Las reglas de producción sirven para generar cadenas, proceso que se formaliza mediante las derivaciones formales:



Derivaciones

Generación de cadenas

Las reglas de producción sirven para generar cadenas, proceso que se formaliza mediante las derivaciones formales:

Dadas dos palabras $w, v \in (V \cup T)^*$ decimos que v es **derivable** a partir de w en un paso ($w \rightarrow v$) si y sólo si:



Derivaciones

Generación de cadenas

Las reglas de producción sirven para generar cadenas, proceso que se formaliza mediante las derivaciones formales:

Dadas dos palabras $w, v \in (V \cup T)^*$ decimos que v es **derivable** a partir de w en un paso ($w \rightarrow v$) si y sólo si:

- Existe una regla $\alpha \rightarrow \beta$ en P y cadenas $\gamma_1, \gamma_2 \in (V \cup T)^*$ tales que:

$$w = \gamma_1 \alpha \gamma_2 \text{ y } v = \gamma_1 \beta \gamma_2$$



Derivaciones

Generación de cadenas

Las reglas de producción sirven para generar cadenas, proceso que se formaliza mediante las derivaciones formales:

Dadas dos palabras $w, v \in (V \cup T)^*$ decimos que v es **derivable** a partir de w en un paso ($w \rightarrow v$) si y sólo si:

- Existe una regla $\alpha \rightarrow \beta$ en P y cadenas $\gamma_1, \gamma_2 \in (V \cup T)^*$ tales que:

$$w = \gamma_1 \alpha \gamma_2 \text{ y } v = \gamma_1 \beta \gamma_2$$

- Algunos autores utilizan \Rightarrow en vez de \rightarrow para denotar la relación de derivación. Nosotros preferimos sobrecargar el operador \rightarrow



Derivaciones formales

\rightarrow^*

Decimos que una cadena v es **derivable** a partir de w si existen palabras $\gamma_2, \dots, \gamma_n$ tales que

$$w = \gamma_1 \rightarrow \gamma_2 \dots \gamma_{n+1} \rightarrow \gamma_n = v$$



Derivaciones formales

\rightarrow^*

Decimos que una cadena v es **derivable** a partir de w si existen palabras $\gamma_2, \dots, \gamma_n$ tales que

$$w = \gamma_1 \rightarrow \gamma_2 \dots \gamma_{n+1} \rightarrow \gamma_n = v$$

En tal caso escribimos **$w \rightarrow^* v$** .



Lenguaje generado por una gramática

$L(G)$

Dada una gramática $G = \langle V, T, S, P \rangle$ definimos al lenguaje generado por G , denotado $L(G)$, como el conjunto de palabras de símbolos **terminales** derivables a partir del símbolo inicial S . Es decir,



Lenguaje generado por una gramática

$L(G)$

Dada una gramática $G = \langle V, T, S, P \rangle$ definimos al lenguaje generado por G , denotado $L(G)$, como el conjunto de palabras de símbolos **terminales** derivables a partir del símbolo inicial S . Es decir,

$$L(G) = \{w \in T^* \mid S \rightarrow^* w\}$$



$$L = (a + b)^*$$

Ejemplos

- Cualquier cadena de aes y bes debe generarse.



$$L = (a + b)^*$$

Ejemplos

- Cualquier cadena de aes y bes debe generarse.
- La cadena vacía debe generarse:

$$S \rightarrow \varepsilon$$



$$L = (a + b)^*$$

Ejemplos

- Cualquier cadena de aes y bes debe generarse.
- La cadena vacía debe generarse:

$$S \rightarrow \varepsilon$$

- Si $w \in L$ entonces $wa \in L$

$$S \rightarrow Sa$$



$$L = (a + b)^*$$

Ejemplos

- Cualquier cadena de aes y bes debe generarse.
- La cadena vacía debe generarse:

$$S \rightarrow \varepsilon$$

- Si $w \in L$ entonces $wa \in L$

$$S \rightarrow Sa$$

- Si $w \in L$ entonces $wb \in L$

$$S \rightarrow Sb$$



$$L = (a + b)^*$$

Ejemplos

- Cualquier cadena de aes y bes debe generarse.
- La cadena vacía debe generarse:

$$S \rightarrow \varepsilon$$

- Si $w \in L$ entonces $wa \in L$

$$S \rightarrow Sa$$

- Si $w \in L$ entonces $wb \in L$

$$S \rightarrow Sb$$

- Ejemplo de derivación: $w = ababb$

$$S \rightarrow Sb \rightarrow Sbb \rightarrow Sabb \rightarrow Sbabb \rightarrow Sababb \rightarrow ababb$$



$$L = \{a^i b^j \mid i, j \in \mathbb{N}, i \leq j\}$$

Ejemplos

- La cadena vacía debe generarse ($i = j = 0$):

$$S \rightarrow \varepsilon$$



$$L = \{a^i b^j \mid i, j \in \mathbb{N}, i \leq j\}$$

Ejemplos

- La cadena vacía debe generarse ($i = j = 0$):

$$S \rightarrow \varepsilon$$

- Debe haber al menos tantas bes como aes, primero aes y luego bes:

$$S \rightarrow aSb$$



$$L = \{a^i b^j \mid i, j \in \mathbb{N}, i \leq j\}$$

Ejemplos

- La cadena vacía debe generarse ($i = j = 0$):

$$S \rightarrow \varepsilon$$

- Debe haber al menos tantas bes como aes, primero aes y luego bes:

$$S \rightarrow aSb$$

- Puede haber más bes al final:

$$S \rightarrow Sb$$



$$L = \{a^i b^j \mid i, j \in \mathbb{N}, i \leq j\}$$

Ejemplos

- La cadena vacía debe generarse ($i = j = 0$):

$$S \rightarrow \varepsilon$$

- Debe haber al menos tantas *b*s como *a*s, primero *a*s y luego *b*s:

$$S \rightarrow aSb$$

- Puede haber más *b*s al final:

$$S \rightarrow Sb$$

- Ejemplo de derivación: $w = aabbbb$

$$S \rightarrow aSb \rightarrow aaSbb \rightarrow aaSbbb \rightarrow aabbbb$$



$$L = \{a^i b^j a^j b^i \mid i, j \in \mathbb{N}\}$$

Ejemplos

- Primero generamos el centro de la palabra, $b^j a^j$:

$$S \rightarrow B \quad B \rightarrow bBa \quad B \rightarrow \varepsilon$$



$$L = \{a^i b^j a^j b^i \mid i, j \in \mathbb{N}\}$$

Ejemplos

- Primero generamos el centro de la palabra, $b^j a^j$:

$$S \rightarrow B \quad B \rightarrow bBa \quad B \rightarrow \varepsilon$$

- Después los extremos a^i , b^i :

$$S \rightarrow aSb$$



$$L = \{a^i b^j a^j b^i \mid i, j \in \mathbb{N}\}$$

Ejemplos

- Primero generamos el centro de la palabra, $b^j a^j$:

$$S \rightarrow B \quad B \rightarrow bBa \quad B \rightarrow \varepsilon$$

- Después los extremos a^i , b^i :

$$S \rightarrow aSb$$

- Ejemplo de derivación: $w = aababb$

$$S \rightarrow aSb \rightarrow aaSbb \rightarrow aaBbb \rightarrow aabBabb \rightarrow aababb$$



$$L = \{a^i b^j a^j b^i \mid i, j \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$



$$L = \{a^i b^j a^j b^i \mid i, j \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$

- Para generar a L simplemente agregamos:

$$S \rightarrow PP$$



$$L = \{a^i b^j a^i b^j \mid i, j \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^j \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$

- Para generar a L simplemente agregamos:

$$S \rightarrow PP$$

- Ejemplo de derivación: $w = aabbab$

$$S \rightarrow PP \rightarrow aPbP \rightarrow aPbaPb \rightarrow aaPbbaPb \rightarrow aaPbbab \rightarrow aabbab$$



$$L = \{a^i b^i \mid i \in \mathbb{N}\} \cup \{b^i a^i \mid i \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$



$$L = \{a^i b^i \mid i \in \mathbb{N}\} \cup \{b^i a^i \mid i \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$

- El lenguaje $\{b^i a^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$Q \rightarrow \varepsilon \quad Q \rightarrow bQa$$



$$L = \{a^i b^i \mid i \in \mathbb{N}\} \cup \{b^i a^i \mid i \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$

- El lenguaje $\{b^i a^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$Q \rightarrow \varepsilon \quad Q \rightarrow bQa$$

- Para generar a L simplemente agregamos:

$$S \rightarrow P \quad S \rightarrow Q$$



$$L = \{a^i b^i \mid i \in \mathbb{N}\} \cup \{b^i a^i \mid i \in \mathbb{N}\}$$

Ejemplos

- El lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$P \rightarrow \varepsilon \quad P \rightarrow aPb$$

- El lenguaje $\{b^i a^i \mid i \in \mathbb{N}\}$ se genera mediante:

$$Q \rightarrow \varepsilon \quad Q \rightarrow bQa$$

- Para generar a L simplemente agregamos:

$$S \rightarrow P \quad S \rightarrow Q$$

- Ejemplo de derivación: $w = bbbaaa$

$$S \rightarrow P \rightarrow aPb \rightarrow aaPbb \rightarrow aaaPbbb \rightarrow aaabbb$$



Correctud y completud

Diseño de gramáticas

- Si bien muchas veces el diseño de una gramática G para un lenguaje dado L es intuitivamente claro y correcto, esto debe mostrarse formalmente, mostrando que $L = L(G)$. Esto se hace, por supuesto, mostrando lo siguiente:



Correctud y completud

Diseño de gramáticas

- Si bien muchas veces el diseño de una gramática G para un lenguaje dado L es intuitivamente claro y correcto, esto debe mostrarse formalmente, mostrando que $L = L(G)$. Esto se hace, por supuesto, mostrando lo siguiente:
- Correctud: la gramática G genera únicamente cadenas de L , es decir, $L(G) \subseteq L$.



Correctud y completud

Diseño de gramáticas

- Si bien muchas veces el diseño de una gramática G para un lenguaje dado L es intuitivamente claro y correcto, esto debe mostrarse formalmente, mostrando que $L = L(G)$. Esto se hace, por supuesto, mostrando lo siguiente:
- Correctud: la gramática G genera únicamente cadenas de L , es decir, $L(G) \subseteq L$.
- Completud: toda cadena de L es generada por G , es decir, $L \subseteq L(G)$.



Lenguajes recursivamente enumerables o tipo 0

Jerarquía de Chomsky

Son aquellos lenguajes generados por una gramática sin restricciones adicionales.

Tales gramáticas pueden incluir reglas de la forma

$$\alpha \rightarrow \varepsilon$$

De manera que la gramática es capaz de borrar cadenas.



Lenguajes recursivamente enumerables o tipo 0

Jerarquía de Chomsky

Son aquellos lenguajes generados por una gramática sin restricciones adicionales.

Tales gramáticas pueden incluir reglas de la forma

$$\alpha \rightarrow \varepsilon$$

De manera que la gramática es capaz de borrar cadenas. Tales gramáticas se conocen como **contraibles**.

Ejemplo:

$$aS \rightarrow bSb, aSb \rightarrow \varepsilon, SbS \rightarrow bcS$$



Lenguajes recursivamente enumerables o tipo 0

Jerarquía de Chomsky

- La siguiente es una gramática de tipo 0:

$$S \rightarrow AT \quad A \rightarrow 0AO \quad A \rightarrow 1AI \quad OO \rightarrow OO$$

$$O1 \rightarrow 1O \quad I0 \rightarrow 0I \quad I1 \rightarrow 1I \quad OT \rightarrow 0T$$

$$IT \rightarrow 1T \quad A \rightarrow \varepsilon \quad T \rightarrow \varepsilon$$



Lenguajes recursivamente enumerables o tipo 0

Jerarquía de Chomsky

- La siguiente es una gramática de tipo 0:

$$S \rightarrow AT \quad A \rightarrow 0AO \quad A \rightarrow 1AI \quad OO \rightarrow OO$$

$$O1 \rightarrow 1O \quad I0 \rightarrow 0I \quad I1 \rightarrow 1I \quad OT \rightarrow 0T$$

$$IT \rightarrow 1T \quad A \rightarrow \varepsilon \quad T \rightarrow \varepsilon$$

- $L(G) = \{ww \mid w \in \{0,1\}^*\}$



Lenguajes recursivamente enumerables o tipo 0

Jerarquía de Chomsky

- La siguiente es una gramática de tipo 0:

$$S \rightarrow AT \quad A \rightarrow 0AO \quad A \rightarrow 1AI \quad OO \rightarrow OO$$

$$O1 \rightarrow 1O \quad I0 \rightarrow 0I \quad I1 \rightarrow 1I \quad OT \rightarrow 0T$$

$$IT \rightarrow 1T \quad A \rightarrow \varepsilon \quad T \rightarrow \varepsilon$$

- $L(G) = \{ww \mid w \in \{0,1\}^*\}$
- La idea del diseño de esta gramática y la razón del nombre *recursivamente enumerable* se discutirán más adelante.



Lenguajes dependientes del contexto o tipo 1

Jerarquía de Chomsky

Son aquellos generados por gramáticas con todas sus producciones son de la forma

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$$

con $\alpha_1, \alpha_2 \in (V \cup T)^*$, $A \in V$, $\beta \neq \varepsilon$.

Con la posible excepción de la regla $S \rightarrow \varepsilon$, en cuyo caso se prohíbe la presencia de S a la derecha de las producciones.



Lenguajes dependientes del contexto o tipo 1

Jerarquía de Chomsky

Son aquellos generados por gramáticas con todas sus producciones son de la forma

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$$

con $\alpha_1, \alpha_2 \in (V \cup T)^*$, $A \in V$, $\beta \neq \varepsilon$.

Con la posible excepción de la regla $S \rightarrow \varepsilon$, en cuyo caso se prohíbe la presencia de S a la derecha de las producciones.

Por ejemplo la siguiente gramática dependiente del contexto genera al lenguaje $L = \{a^i b^i c^i \mid i \geq 0\}$

$$S \rightarrow A \quad A \rightarrow aABC \mid abC \quad CB \rightarrow BC$$

$$bB \rightarrow bb \quad bC \rightarrow bc \quad cC \rightarrow cc$$



Lenguajes libres del contexto o tipo 2

Jerarquía de Chomsky

- Son aquellos generados por gramáticas con todas sus producciones de la forma

$$A \rightarrow \alpha$$

con $A \in V$, $\alpha \in (V \cup T)^*$.



Lenguajes libres del contexto o tipo 2

Jerarquía de Chomsky

- Son aquellos generados por gramáticas con todas sus producciones de la forma

$$A \rightarrow \alpha$$

con $A \in V$, $\alpha \in (V \cup T)^*$.

- Esta definición incluye a la regla $S \rightarrow \varepsilon$.



Lenguajes libres del contexto o tipo 2

Jerarquía de Chomsky

- Son aquellos generados por gramáticas con todas sus producciones de la forma

$$A \rightarrow \alpha$$

con $A \in V$, $\alpha \in (V \cup T)^*$.

- Esta definición incluye a la regla $S \rightarrow \varepsilon$.
- La mayoría de las gramáticas para lenguajes de programación caen en esta categoría.



Lenguajes regulares o tipo 3

Jerarquía de Chomsky

Son aquellos generados por una gramática de una de las siguientes formas:

- Lineal por la derecha: todas las producciones de la forma

$$A \rightarrow aB \quad A \rightarrow a \quad A \rightarrow \varepsilon$$

con $A, B \in V$, $a \in T$



Lenguajes regulares o tipo 3

Jerarquía de Chomsky

Son aquellos generados por una gramática de una de las siguientes formas:

- Lineal por la derecha: todas las producciones de la forma

$$A \rightarrow aB \quad A \rightarrow a \quad A \rightarrow \varepsilon$$

con $A, B \in V$, $a \in T$

- Lineas por la izquierda: todas las producciones de la forma

$$A \rightarrow Ba \quad A \rightarrow a \quad A \rightarrow \varepsilon$$

con $A, B \in V$, $a \in T$



Lenguajes regulares o tipo 3

Jerarquía de Chomsky

Son aquellos generados por una gramática de una de las siguientes formas:

- Lineal por la derecha: todas las producciones de la forma

$$A \rightarrow aB \quad A \rightarrow a \quad A \rightarrow \varepsilon$$

con $A, B \in V$, $a \in T$

- Lineas por la izquierda: todas las producciones de la forma

$$A \rightarrow Ba \quad A \rightarrow a \quad A \rightarrow \varepsilon$$

con $A, B \in V$, $a \in T$

- No se permite mezclar ambos tipos de producciones.



Jerarquía de Chomsky

Observaciones

- Decimos que un lenguaje es de tipo i si y sólo si i es el índice mas grande tal que existe una gramática de tipo i que genera a L



Jerarquía de Chomsky

Observaciones

- Decimos que un lenguaje es de tipo i si y sólo si i es el índice mas grande tal que existe una gramática de tipo i que genera a L
- La jerarquía de gramáticas genera una jerarquía en los lenguajes generados:

$$\mathcal{L}_3 \subsetneq \mathcal{L}_2 \subsetneq \mathcal{L}_1 \subsetneq \mathcal{L}_0$$



Jerarquía de Chomsky

Observaciones

- Decimos que un lenguaje es de tipo i si y sólo si i es el índice mas grande tal que existe una gramática de tipo i que genera a L
- La jerarquía de gramáticas genera una jerarquía en los lenguajes generados:

$$\mathcal{L}_3 \subsetneq \mathcal{L}_2 \subsetneq \mathcal{L}_1 \subsetneq \mathcal{L}_0$$

- La jerarquía de Chomsky permite refinar la teoría de la computación clasificando lenguajes en función de los recursos computacionales necesarios para reconocerlos.



Gramáticas Regulares

- Una gramática regular es una gramática lineal por la derecha o lineal por la izquierda.



Gramáticas Regulares

- Una gramática regular es una gramática lineal por la derecha o lineal por la izquierda.
- No se permite mezclar ambos tipos de producciones.



Gramáticas Regulares

- Una gramática regular es una gramática lineal por la derecha o lineal por la izquierda.
- No se permite mezclar ambos tipos de producciones.
- Se puede probar que toda gramática lineal por la izquierda es equivalente a una gramática lineal por la derecha.



Gramáticas Regulares

Lenguajes Regulares

- Decimos que un lenguaje L es regular si **existe** una gramática regular G que lo genere, es decir, si $L = L(G)$.



Gramáticas Regulares

Lenguajes Regulares

- Decimos que un lenguaje L es regular si **existe** una gramática regular G que lo genere, es decir, si $L = L(G)$.
- Si L es generado por una gramática de tipo i , no se puede asegurar de inmediato que L sea un lenguaje de tipo i . Debe asegurarse que i es máximo.



$$L = 0^*10^*10^*$$

Ejemplos

L es generado por:

$$\begin{aligned} S &\rightarrow A1A1A \\ A &\rightarrow 0A \mid \varepsilon \end{aligned}$$

esta gramática no es regular,



$$L = 0^*10^*10^*$$

Ejemplos

L es generado por:

$$\begin{aligned} S &\rightarrow A1A1A \\ A &\rightarrow 0A \mid \varepsilon \end{aligned}$$

esta gramática no es regular, pero el lenguaje si lo es al existir una gramática regular equivalente:

$$\begin{aligned} S &\rightarrow 0S \mid 1A \\ A &\rightarrow 0A \mid 1B \\ B &\rightarrow 0B \mid \varepsilon \end{aligned}$$



$$L = (a + b)^*b$$

Ejemplos

L es generado por:

$$\begin{array}{lcl} S & \rightarrow & aS \mid bC \\ C & \rightarrow & bC \mid aS \mid \varepsilon \end{array}$$



Lenguajes y gramáticas regulares

AF \Rightarrow GR

Dado un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe una gramática regular $G = \langle V, T, S, P \rangle$ tal que $L(M) = L(G)$. Es decir, todo lenguaje regular es generado por una gramática regular.



Lenguajes y gramáticas regulares

AF \Rightarrow GR

Dado un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe una gramática regular $G = \langle V, T, S, P \rangle$ tal que $L(M) = L(G)$. Es decir, todo lenguaje regular es generado por una gramática regular.

Definimos a G como sigue:

- Suponemos s.p.g. que no hay ε -transiciones.



Lenguajes y gramáticas regulares

AF \Rightarrow GR

Dado un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe una gramática regular $G = \langle V, T, S, P \rangle$ tal que $L(M) = L(G)$. Es decir, todo lenguaje regular es generado por una gramática regular.

Definimos a G como sigue:

- Suponemos s.p.g. que no hay ε -transiciones.
- $V = Q \quad T = \Sigma \quad S = q_0$



Lenguajes y gramáticas regulares

AF \Rightarrow GR

Dado un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe una gramática regular $G = \langle V, T, S, P \rangle$ tal que $L(M) = L(G)$. Es decir, todo lenguaje regular es generado por una gramática regular.

Definimos a G como sigue:

- Suponemos s.p.g. que no hay ε -transiciones.
- $V = Q \quad T = \Sigma \quad S = q_0$
- P se define como sigue:



Lenguajes y gramáticas regulares

AF \Rightarrow GR

Dado un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe una gramática regular $G = \langle V, T, S, P \rangle$ tal que $L(M) = L(G)$. Es decir, todo lenguaje regular es generado por una gramática regular.

Definimos a G como sigue:

- Suponemos s.p.g. que no hay ε -transiciones.
- $V = Q \quad T = \Sigma \quad S = q_0$
- P se define como sigue:
 - ▶ Si $p \in \delta(q, a)$ entonces agregamos $q \rightarrow ap$ a P .



Lenguajes y gramáticas regulares

AF \Rightarrow GR

Dado un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe una gramática regular $G = \langle V, T, S, P \rangle$ tal que $L(M) = L(G)$. Es decir, todo lenguaje regular es generado por una gramática regular.

Definimos a G como sigue:

- Suponemos s.p.g. que no hay ε -transiciones.
- $V = Q$ $T = \Sigma$ $S = q_0$
- P se define como sigue:
 - ▶ Si $p \in \delta(q, a)$ entonces agregamos $q \rightarrow ap$ a P .
 - ▶ Si $q_f \in \delta(q, a)$ con $q_f \in F$ entonces agregamos $q \rightarrow a$.



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.

Definimos a M como sigue:

- Suponemos s.p.g. que G es lineal derecha.



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.

Definimos a M como sigue:

- Suponemos s.p.g. que G es lineal derecha.
- $Q = V \cup \{q_F\}$ $\Sigma = T$ $q_0 = S$ $F = \{q_F\}$



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.

Definimos a M como sigue:

- Suponemos s.p.g. que G es lineal derecha.
- $Q = V \cup \{q_F\}$ $\Sigma = T$ $q_0 = S$ $F = \{q_F\}$
- δ se define como sigue:



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.

Definimos a M como sigue:

- Suponemos s.p.g. que G es lineal derecha.
- $Q = V \cup \{q_F\}$ $\Sigma = T$ $q_0 = S$ $F = \{q_F\}$
- δ se define como sigue:
 - ▶ Si $A \rightarrow aB \in P$ entonces $B \in \delta(A, a)$.



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.

Definimos a M como sigue:

- Suponemos s.p.g. que G es lineal derecha.
- $Q = V \cup \{q_F\}$ $\Sigma = T$ $q_0 = S$ $F = \{q_F\}$
- δ se define como sigue:
 - ▶ Si $A \rightarrow aB \in P$ entonces $B \in \delta(A, a)$.
 - ▶ Si $A \rightarrow a \in P$ entonces $q_F \in \delta(A, a)$.



Lenguajes y gramáticas regulares

GR \Rightarrow AF

Dada una gramática regular $G = \langle V, T, S, P \rangle$ existe un AF $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $L(M) = L(G)$. Es decir todo lenguaje generado por una gramática regular es un lenguaje regular.

Definimos a M como sigue:

- Suponemos s.p.g. que G es lineal derecha.
- $Q = V \cup \{q_F\}$ $\Sigma = T$ $q_0 = S$ $F = \{q_F\}$
- δ se define como sigue:
 - ▶ Si $A \rightarrow aB \in P$ entonces $B \in \delta(A, a)$.
 - ▶ Si $A \rightarrow a \in P$ entonces $q_F \in \delta(A, a)$.
 - ▶ Si $A \rightarrow \varepsilon \in P$ entonces $q_F \in \delta(A, \varepsilon)$.



Gramáticas libres de contexto

Definición

Una gramática es libre o independiente del contexto si todas sus producciones son de la forma

$$A \rightarrow \alpha$$

con $A \in V$, $\alpha \in (V \cup T)^*$.



Gramáticas libres de contexto

Definición

Una gramática es libre o independiente del contexto si todas sus producciones son de la forma

$$A \rightarrow \alpha$$

con $A \in V$, $\alpha \in (V \cup T)^*$.

Esta definición incluye a la regla $S \rightarrow \varepsilon$.



Gramáticas libres de contexto

Definición

Una gramática es libre o independiente del contexto si todas sus producciones son de la forma

$$A \rightarrow \alpha$$

con $A \in V$, $\alpha \in (V \cup T)^*$.

Esta definición incluye a la regla $S \rightarrow \varepsilon$.

Obsérvese que en particular toda gramática regular es libre de contexto.



Gramáticas libres de contexto

Ejemplos

- $L = a^*$

$$S \rightarrow aS \mid \varepsilon$$



Gramáticas libres de contexto

Ejemplos

- $L = a^*$

$$S \rightarrow aS \mid \varepsilon$$

- $L = a^*b^*$

$$\begin{aligned} S &\rightarrow aS \mid bA \mid \varepsilon \\ A &\rightarrow bA \mid b \mid \varepsilon \end{aligned}$$



Gramáticas libres de contexto

Ejemplos

- $L = a^*$

$$S \rightarrow aS \mid \varepsilon$$

- $L = a^*b^*$

$$\begin{aligned} S &\rightarrow aS \mid bA \mid \varepsilon \\ A &\rightarrow bA \mid b \mid \varepsilon \end{aligned}$$

- $L = 0^+1^+$

$$\begin{aligned} S &\rightarrow CU \\ C &\rightarrow 0C \mid 0 \\ U &\rightarrow 1U \mid 1 \end{aligned}$$



Gramáticas libres de contexto

Ejemplos

- $L = \{a^n b a^m \mid n, m \geq 1\} = a^+ b a^+$

$$S \rightarrow aS \mid aB$$

$$B \rightarrow bC$$

$$C \rightarrow aC \mid a$$



Gramáticas libres de contexto

Ejemplos

- $L = \{a^n b a^m \mid n, m \geq 1\} = a^+ b a^+$

$$S \rightarrow aS \mid aB$$

$$B \rightarrow bC$$

$$C \rightarrow aC \mid a$$

- $L = \{a^n b^n \mid n \in \mathbb{N}\}$ que no es regular

$$S \rightarrow aSb \mid \varepsilon$$



Gramáticas libres de contexto

Ejemplos

- $L = \{a^n b a^m \mid n, m \geq 1\} = a^+ b a^+$

$$S \rightarrow aS \mid aB$$

$$B \rightarrow bC$$

$$C \rightarrow aC \mid a$$

- $L = \{a^n b^n \mid n \in \mathbb{N}\}$ que no es regular

$$S \rightarrow aSb \mid \varepsilon$$

- $L = \{w \in \{a, b\}^* \mid w = w^R\}$ que no es regular

$$S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$$



Derivaciones por la izquierda

GLC

Una derivación $S \rightarrow^* w$ es una derivación por la izquierda si en cada paso se reescribe la variable mas a la izquierda en la palabra.



Derivaciones por la izquierda

GLC

Una derivación $S \rightarrow^* w$ es una derivación por la izquierda si en cada paso se reescribe la variable mas a la izquierda en la palabra.

En la gramática

$$S \rightarrow aAs \mid a \quad A \rightarrow SbA \mid SS \mid ba$$



Derivaciones por la izquierda

GLC

Una derivación $S \rightarrow^* w$ es una derivación por la izquierda si en cada paso se reescribe la variable mas a la izquierda en la palabra.

En la gramática

$$S \rightarrow aAs \mid a \quad A \rightarrow SbA \mid SS \mid ba$$

tenemos la siguiente derivación por la izquierda de $aabbbaa$.

$$S \rightarrow aAS \rightarrow aSbAS \rightarrow aabAS \rightarrow aabbaS \rightarrow aabbbaa$$



Derivaciones por la derecha

GLC

Una derivación $S \rightarrow^* w$ es una derivación por la derecha si en cada paso se reescribe la variable mas a la derecha en la palabra.



Derivaciones por la derecha

GLC

Una derivación $S \rightarrow^* w$ es una derivación por la derecha si en cada paso se reescribe la variable mas a la derecha en la palabra.

En la gramática

$$S \rightarrow aAs \mid a \quad A \rightarrow SbA \mid SS \mid ba$$



Derivaciones por la derecha

GLC

Una derivación $S \rightarrow^* w$ es una derivación por la derecha si en cada paso se reescribe la variable mas a la derecha en la palabra.

En la gramática

$$S \rightarrow aAs \mid a \quad A \rightarrow SbA \mid SS \mid ba$$

tenemos la siguiente derivación por la derecha de $aabbbaa$.

$$S \rightarrow aAS \rightarrow aAa \rightarrow aSbAa \rightarrow aSbbaa \rightarrow aabbbaa$$



Árboles de derivación

GLC

- Los árboles de derivación o árboles sintácticos son un mecanismo para representar las derivaciones de gramáticas libres de contexto.



Árboles de derivación

GLC

- Los árboles de derivación o árboles sintácticos son un mecanismo para representar las derivaciones de gramáticas libres de contexto.
- En compiladores se utilizan para el análisis sintáctico de programas fuente (parsing) y sirven de base para la generación de código.



Árboles de derivación

GLC

- Los árboles de derivación o árboles sintácticos son un mecanismo para representar las derivaciones de gramáticas libres de contexto.
- En compiladores se utilizan para el análisis sintáctico de programas fuente (parsing) y sirven de base para la generación de código.
- Puede ser que dos derivaciones distintas tengan el mismo árbol.



Árboles de derivación

GLC

- Puede haber mas de un árbol de derivación para una cadena.



Árboles de derivación

GLC

- Puede haber mas de un árbol de derivación para una cadena.
- Lo ideal es que cada cadena tenga sólo un árbol asociado, esto implica que el lenguaje no es ambiguo.



Árboles de derivación

GLC

- Puede haber mas de un árbol de derivación para una cadena.
- Lo ideal es que cada cadena tenga sólo un árbol asociado, esto implica que el lenguaje no es ambiguo.
- Desafortunadamente existen lenguajes ambiguos.



Construcción de árboles de derivación

GLC

Dada una gramática libre de contexto $G = \langle V, T, S, P \rangle$, un árbol de derivación en G se construye como sigue:



Construcción de árboles de derivación

GLC

Dada una gramática libre de contexto $G = \langle V, T, S, P \rangle$, un árbol de derivación en G se construye como sigue:

- La raíz contiene al símbolo inicial S .



Construcción de árboles de derivación

GLC

Dada una gramática libre de contexto $G = \langle V, T, S, P \rangle$, un árbol de derivación en G se construye como sigue:

- La raíz contiene al símbolo inicial S .
- Cada nodo interior contiene una variable



Construcción de árboles de derivación

GLC

Dada una gramática libre de contexto $G = \langle V, T, S, P \rangle$, un árbol de derivación en G se construye como sigue:

- La raíz contiene al símbolo inicial S .
- Cada nodo interior contiene una variable
- Cada hoja contiene un símbolo de $V \cup T \cup \{\epsilon\}$.



Construcción de árboles de derivación

GLC

Dada una gramática libre de contexto $G = \langle V, T, S, P \rangle$, un árbol de derivación en G se construye como sigue:

- La raíz contiene al símbolo inicial S .
- Cada nodo interior contiene una variable
- Cada hoja contiene un símbolo de $V \cup T \cup \{\epsilon\}$.
- Si un nodo interior contiene una variable A entonces sus hijos contienen símbolos (de izquierda a derecha) a_1, \dots, a_n si y sólo si $A \rightarrow a_1 a_2 \dots a_n$ está en P .



Construcción de árboles de derivación

GLC

Dada una gramática libre de contexto $G = \langle V, T, S, P \rangle$, un árbol de derivación en G se construye como sigue:

- La raíz contiene al símbolo inicial S .
- Cada nodo interior contiene una variable
- Cada hoja contiene un símbolo de $V \cup T \cup \{\varepsilon\}$.
- Si un nodo interior contiene una variable A entonces sus hijos contienen símbolos (de izquierda a derecha) a_1, \dots, a_n si y sólo si $A \rightarrow a_1 a_2 \dots a_n$ está en P .
- La palabra generada se puede leer al leer las hojas de izquierda a derecha.



Ambigüedad

GLC

- Una gramática se dice **ambigua** si existe una palabra w con dos o más árboles de derivación distintos.



Ambigüedad

GLC

- Una gramática se dice **ambigua** si existe una palabra w con dos o más árboles de derivación distintos.
- En general una palabra puede tener mas de una derivación, pero un sólo árbol, en tal caso no hay ambigüedad.



Ambigüedad

GLC

- Una gramática se dice **ambigua** si existe una palabra w con dos o más árboles de derivación distintos.
- En general una palabra puede tener mas de una derivación, pero un sólo árbol, en tal caso no hay ambigüedad.
- Algunas veces se puede suprimir la ambigüedad directamente.



Ambigüedad

GLC

- Una gramática se dice **ambigua** si existe una palabra w con dos o más árboles de derivación distintos.
- En general una palabra puede tener mas de una derivación, pero un sólo árbol, en tal caso no hay ambigüedad.
- Algunas veces se puede suprimir la ambigüedad directamente.
- Sin embargo no hay un algoritmo para remover ambigüedad.



Ambigüedad

GLC

- Una gramática se dice **ambigua** si existe una palabra w con dos o más árboles de derivación distintos.
- En general una palabra puede tener mas de una derivación, pero un sólo árbol, en tal caso no hay ambigüedad.
- Algunas veces se puede suprimir la ambigüedad directamente.
- Sin embargo no hay un algoritmo para remover ambigüedad.
- Pero aún, hay lenguajes cuya ambigüedad es inevitable.



Ejemplos

Ambigüedad

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

La palabra a^5 tiene las siguientes derivaciones:



Ejemplos

Ambigüedad

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

La palabra a^5 tiene las siguientes derivaciones:

- $S \rightarrow AA \rightarrow aA \rightarrow aaSa \rightarrow aaAAa \rightarrow aaaAa \rightarrow aaaaa$



Ejemplos

Ambigüedad

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

La palabra a^5 tiene las siguientes derivaciones:

- $S \rightarrow AA \rightarrow aA \rightarrow aaSa \rightarrow aaAAa \rightarrow aaaAa \rightarrow aaaaa$
- $S \rightarrow AA \rightarrow aSaA \rightarrow aAAaA \rightarrow aaAaA \rightarrow aaaaA \rightarrow aaaaa$



Ejemplos

Ambigüedad

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

La palabra a^5 tiene las siguientes derivaciones:

- $S \rightarrow AA \rightarrow aA \rightarrow aaSa \rightarrow aaAAa \rightarrow aaaAa \rightarrow aaaaa$
- $S \rightarrow AA \rightarrow aSaA \rightarrow aAAaA \rightarrow aaAaA \rightarrow aaaaA \rightarrow aaaaa$
- Las dos derivaciones son por la izquierda y generan árboles distintos.



Lenguajes Ambiguos

Ambigüedad

- Un lenguaje L es ambiguo si existe una gramática ambigua G que genera a L .



Lenguajes Ambiguos

Ambigüedad

- Un lenguaje L es ambiguo si existe una gramática ambigua G que genera a L .
- $L = \{a^{2+3i} \mid i \geq 0\}$ es ambiguo.



Lenguajes Ambiguos

Ambigüedad

- Un lenguaje L es ambiguo si existe una gramática ambigua G que genera a L .
- $L = \{a^{2+3i} \mid i \geq 0\}$ es ambiguo.
- Un lenguaje es inherentemente ambiguo si todas las gramáticas que lo generan son ambiguas.



Lenguajes Ambiguos

Ambigüedad

- Un lenguaje L es ambiguo si existe una gramática ambigua G que genera a L .
- $L = \{a^{2+3i} \mid i \geq 0\}$ es ambiguo.
- Un lenguaje es inherentemente ambiguo si todas las gramáticas que lo generan son ambiguas.
- $L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$ es inherentemente ambiguo.



$$L = \{a^{2+3i} \mid i \geq 0\}$$

Lenguajes ambiguos

L es ambiguo por se generado por la gramática ambigua

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$



$$L = \{a^{2+3i} \mid i \geq 0\}$$

Lenguajes ambiguos

L es ambiguo por se generado por la gramática ambigua

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

Sin embargo este lenguaje también es generado por una gramática no ambigua:

$$S \rightarrow aa \mid aaU \quad U \rightarrow aaaU \mid aaa$$



$$L = \{a^{2+3i} \mid i \geq 0\}$$

Lenguajes ambiguos

L es ambiguo por se generado por la gramática ambigua

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

Sin embargo este lenguaje también es generado por una gramática no ambigua:

$$S \rightarrow aa \mid aaU \quad U \rightarrow aaaU \mid aaa$$

en este caso la derivación de a^5 es:

$$S \rightarrow aaU \rightarrow aaaaa$$



$$L = \{a^{2+3i} \mid i \geq 0\}$$

Lenguajes ambiguos

L es ambiguo por se generado por la gramática ambigua

$$S \rightarrow AA \quad A \rightarrow aSa \mid a$$

Sin embargo este lenguaje también es generado por una gramática no ambigua:

$$S \rightarrow aa \mid aaU \quad U \rightarrow aaaU \mid aaa$$

en este caso la derivación de a^5 es:

$$S \rightarrow aaU \rightarrow aaaaa$$

por lo tanto L no es un lenguaje inherentemente ambiguo



$$L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

Lenguajes inherentemente ambiguos

L es generado por la gramática:

$$\begin{aligned} S &\rightarrow AB \mid C & A &\rightarrow aAb \mid ab & B &\rightarrow cBd \mid cd \\ C &\rightarrow aCd \mid aDd & D &\rightarrow bDc \mid bc \end{aligned}$$



$$L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

Lenguajes inherentemente ambiguos

L es generado por la gramática:

$$\begin{aligned} S &\rightarrow AB \mid C & A &\rightarrow aAb \mid ab & B &\rightarrow cBd \mid cd \\ C &\rightarrow aCd \mid aDd & D &\rightarrow bDc \mid bc \end{aligned}$$

La cadena $aabbccdd$ tiene dos derivaciones por la izquierda:



$$L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

Lenguajes inherentemente ambiguos

L es generado por la gramática:

$$\begin{aligned} S &\rightarrow AB \mid C & A &\rightarrow aAb \mid ab & B &\rightarrow cBd \mid cd \\ C &\rightarrow aCd \mid aDd & D &\rightarrow bDc \mid bc \end{aligned}$$

La cadena $aabbccdd$ tiene dos derivaciones por la izquierda:

$$S \rightarrow AB \rightarrow aAbB \rightarrow aabbB \rightarrow aabbcBd \rightarrow aabbccdd$$



$$L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

Lenguajes inherentemente ambiguos

L es generado por la gramática:

$$\begin{aligned} S &\rightarrow AB \mid C & A &\rightarrow aAb \mid ab & B &\rightarrow cBd \mid cd \\ C &\rightarrow aCd \mid aDd & D &\rightarrow bDc \mid bc \end{aligned}$$

La cadena $aabbccdd$ tiene dos derivaciones por la izquierda:

$$S \rightarrow AB \rightarrow aAbB \rightarrow aabbB \rightarrow aabbcBd \rightarrow aabbccdd$$

$$S \rightarrow C \rightarrow aCd \rightarrow aaDdd \rightarrow aabDcdd \rightarrow aabbccdd$$



$$L = \{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

Lenguajes inherentemente ambiguos

L es generado por la gramática:

$$\begin{aligned} S &\rightarrow AB \mid C & A &\rightarrow aAb \mid ab & B &\rightarrow cBd \mid cd \\ C &\rightarrow aCd \mid aDd & D &\rightarrow bDc \mid bc \end{aligned}$$

La cadena $aabbccdd$ tiene dos derivaciones por la izquierda:

$$S \rightarrow AB \rightarrow aAbB \rightarrow aabbB \rightarrow aabbcBd \rightarrow aabbccdd$$

$$S \rightarrow C \rightarrow aCd \rightarrow aaDdd \rightarrow aabDcdd \rightarrow aabbccdd$$

Probar la ambigüedad inherente es complicado.

