Sintaxis y Semántica del Lenguaje

Gramáticas y lenguajes libres de contexto

Gramáticas y lenguajes libres de contexto

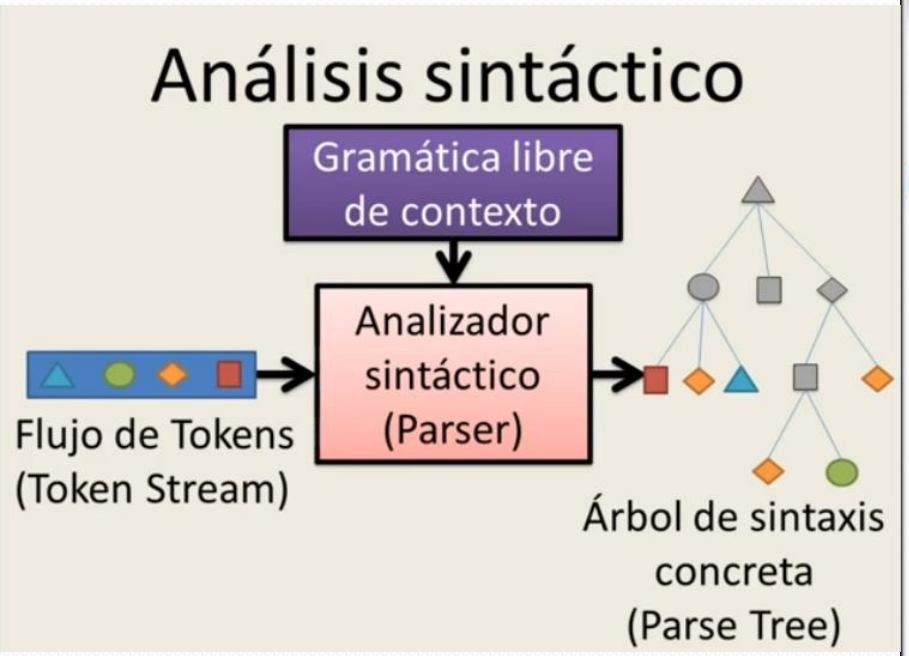
- Los Lenguajes Libres de Contexto (abreviado LLC) forman una clase de lenguajes más amplia que los Lenguajes Regulares, de acuerdo con la Jerarquía de Chomsky.

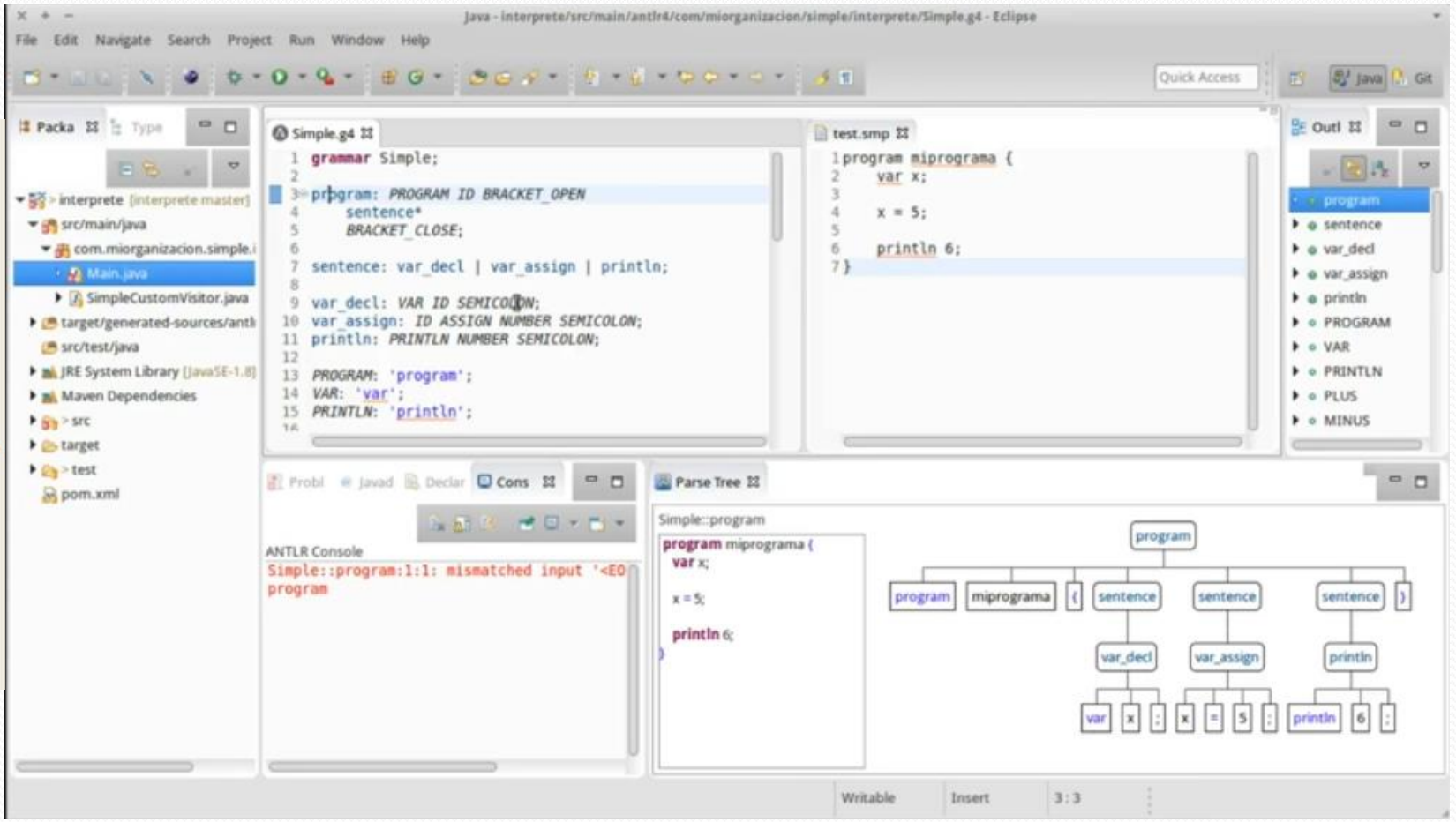
- Estos lenguajes son importantes tanto desde el punto de vista teórico, por relacionar las llamadas Gramáticas Libres de Contexto con los Autómatas de Pila, como desde el punto de vista práctico, ya que casi todos los lenguajes de programación están basados en los LLC.

- En efecto, a partir de los años 70's, con lenguajes como Pascal, se hizo común la práctica de formalizar la sintaxis de los lenguajes de programación usando herramientas basadas en las Gramáticas Libres de Contexto, que representan a los LLC.

- Por otra parte, el análisis automático de los LLC es computacionalmente mucho más eficiente que el de otras clases de lenguajes más generales.

Análisis sintáctico





Reglas

- Una regla es una expresión de la forma $\alpha \rightarrow \beta$, en donde tanto Alpha como Beta son cadenas de símbolos en donde pueden aparecer tanto elementos del alfabeto Sigma (llamados constantes) como unos nuevos símbolos, llamados variables.

- Una gramática es básicamente un conjunto de reglas

Reglas

- Consideremos, por ejemplo, la siguiente gramática para producir un pequeño subconjunto del idioma español:

1. $< frase > \rightarrow < sujeto > < predicado >$
2. $< sujeto > \rightarrow < articulo > < sustantivo >$
3. $< articulo > \rightarrow el \mid la$
4. $< sustantivo > \rightarrow perro \mid luna$
5. $< predicado > \rightarrow < verbo >$
6. $< verbo > \rightarrow brilla \mid corre$

Reglas

- El símbolo “|” separa varias alternativas.

- En esta gramática se supone que las variables son <frase>, <sujeto>, <articulo>, <sustantivo>, <predicado> y <verbo>, mientras que las constantes son el , la, perro y luna. La variable <frase> será considerada símbolo inicial.

Reglas

- La idea central para aplicar una gramática es que se parte de una variable, llamada símbolo inicial, y se aplican repetidamente las reglas gramaticales, hasta que ya no haya variables en la palabra.

- En ese momento se dice que la palabra resultante es generada por la gramática, o en forma equivalente, que la palabra resultante es parte del lenguaje de esa gramática.

Gramáticas

- La Gramática anterior, ¿Genera la frase “el perro corre”?
- ¿Y la frase “la perro brilla”?

Gramáticas y la jerarquía de Chomsky

- Es posible restringir la forma de las reglas gramaticales de manera que se acomoden a patrones predeterminados.

- Por ejemplo, se puede imponer que el lado izquierdo de las reglas sea una variable, en vez de una cadena arbitraria de símbolos. Al restringir las reglas de la gramática se restringen también las palabras que se pueden generar.

Gramáticas y la jerarquía de Chomsky

- N. Chomsky propuso varias formas estándares de reglas que se asocian a varias clases de lenguajes, que ordenó de manera tal que forman una jerarquía, es decir, los lenguajes más primitivos están incluidos en los más complejos.

- Así tenemos las siguientes clases de gramáticas, asociadas a familias de lenguajes

Clases de lenguajes

- Gramáticas regulares, o de tipo 3: las reglas son de la forma $A \rightarrow aB$ o bien $A \rightarrow a$, donde A y B son variables y a es constante.

- Estas gramáticas son capaces de describir los lenguajes regulares.

Clases de lenguajes

- Gramáticas Libres de Contexto (GLC), o de tipo 2: las reglas son de la forma $X \rightarrow \alpha$, donde X es una variable y α es una cadena que puede contener variables y constantes.

- Estas gramáticas producen los lenguajes Libres de Contexto (abreviado "LLC").

Clases de lenguajes

- Gramáticas sensitivas al contexto o de tipo 1: las reglas son de la forma

$$\alpha A \beta \rightarrow \alpha \Gamma \beta$$

donde A es una variable y el resto son cadenas cualesquiera que pueden contener variables y constantes.

Clases de lenguajes

- Gramáticas no restringidas, o de tipo 0, con reglas de la forma $\alpha \rightarrow \beta$, donde α no puede ser vacío, que generan los lenguajes llamados "recursivamente enumerables".

Ejemplo

- Por ejemplo, el lenguaje $\{a^n b^n\}$ –que no es regular tiene la gramática libre de contexto con las siguientes reglas:

$$1. \quad S \rightarrow aSb$$

$$2. \quad S \rightarrow ab$$

Ejemplo

- Aplicar una regla $X \rightarrow \alpha$ de una gramática consiste en reemplazar X por en una palabra. Por ejemplo, la regla especificada debajo se puede aplicar a una palabra aaSbb para obtener la palabra aaaSbbb, en donde es fácil ver que reemplazamos S por aSb.

$$S \rightarrow aSb$$

Paso de Derivación

Al proceso de aplicar una regla se le conoce como "paso de derivación", y se denota usando una flecha gruesa "$\Rightarrow$", como en $aaSbb \Rightarrow aaaSbbb$ (aplicando una regla $S \rightarrow aSb$). Una secuencia de pasos de derivación a partir de una variable especial de la gramática llamada "símbolo inicial" se llama simplemente *derivación*. Por ejemplo, una derivación de la palabra "$aaabbb$" utilizando la gramática de $\{a^n b^n\}$ sería (suponiendo que S es el símbolo inicial):

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaabbb$$

Otro ejemplo

- Como un ejemplo adicional, la gramática con las reglas siguientes permite generar expresiones aritméticas con sumas y multiplicaciones de enteros:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

Otro ejemplo

- Con esta gramática podemos generar, por ejemplo, la expresión 25+3*12 de la manera siguiente:

EXPRESION	JUSTIFICACION
E	Símbolo inicial, inicia derivación
$\Rightarrow E + T$	Aplicación 1a. regla
$\Rightarrow T + T$	2a. regla, sobre la E
$\Rightarrow F + T$	4a. regla, sobre la T izquierda
$\Rightarrow CF + T$	5a. regla, sobre F
$\Rightarrow 2F + T$	7a. regla
$\Rightarrow 2C + T$	6a. regla
$\Rightarrow 25 + T$	7a. regla
$\Rightarrow 25 + T * F$	3a. regla
$\Rightarrow 25 + F * F$	4a. regla
$\Rightarrow 25 + C * F$	6a. regla, sobre la F izquierda
$\Rightarrow 25 + 3 * F$	7a. regla
$\Rightarrow 25 + 3 * CF$	5a. regla
$\Rightarrow 25 + 3 * 1F$	7a. regla
$\Rightarrow 25 + 3 * 1C$	6a. regla
$\Rightarrow 25 + 3 * 12$	7a. regla

Definiciones

Definición.- Una gramática libre de contexto es un cuádruplo (V, Σ, R, S) en donde:

- V es un *alfabeto de variables*, también llamadas *no-terminales*.
- Σ es un *alfabeto de constantes*, también llamadas *terminales*. Suponemos que V y Σ son disjuntos, esto es, $V \cap \Sigma = \emptyset$.
- R, el conjunto de *reglas*, es un subconjunto finito de $V \times (V \cup \Sigma)^*$.
- S, el *símbolo inicial*, es un elemento de V.

Ejemplo.- La gramática de $\{a^n b^n\}$ que presentamos antes se representa formalmente como:

$$(\{S\}, \{a, b\}, \{(S, aSb), (S, ab)\}, S)$$

Usualmente las reglas no se escriben como pares ordenados (X, α), sino como $X \to \alpha$; esto es simplemente cuestión de notación.

Definiciones

Definición.- Una cadena $\alpha \in (V \cup \Sigma)^*$ es *derivable* a partir de una gramática (V, Σ, R, S) si hay al menos una secuencia de pasos de derivación que la produce a partir del símbolo inicial S, esto es:

$$S \Rightarrow \ldots \Rightarrow \alpha$$

Definición.- El lenguaje L(G) generado por una gramática (V, Σ, R, S) es el conjunto de palabras hechas exclusivamente de constantes, que son derivables a partir del símbolo inicial:

$$L = \{w \in \Sigma^* | S \Rightarrow \ldots \Rightarrow w\}$$

Arboles de derivación

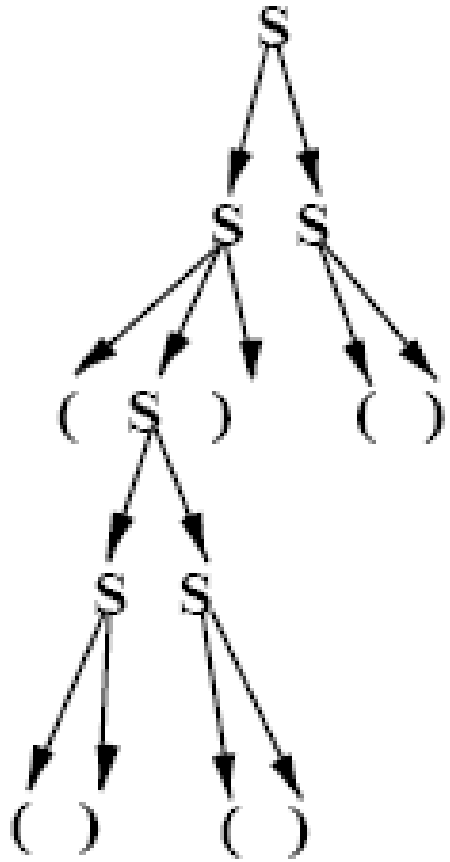
- Las GLC tienen la propiedad de que las derivaciones pueden ser representadas en forma arborescente. Por ejemplo, considérese la gramática siguiente para producir el lenguaje de los paréntesis bien balanceados, que genera palabras como (()), ()(), (()())(), pero no a (() ni)(

1. $S \rightarrow SS$
2. $S \rightarrow (S)$
3. $S \rightarrow ()$

Árboles de Derivación

- Usando esta gramática, la palabra (()())() puede ser derivada de la manera que se ilustra en la figura. En dicha figura se puede apreciar la estructura que se encuentra implícita en la palabra (()())().

- A estas estructuras se les llama árboles de derivación, o también árboles de compilación y son de vital importancia para la teoría de los compiladores de los lenguajes de programación.



Ambigüedad en GLC

- La correspondencia entre los árboles de derivación y sus productos no es necesariamente biunívoca.

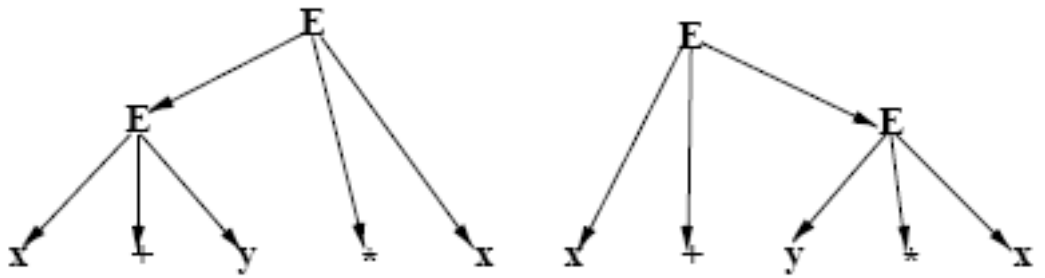
- En efecto, hay GLC en las cuales para ciertas palabras hay más de un árbol de derivación.

- Sea por ejemplo la siguiente GLC, para expresiones aritméticas sobre las variables x e y.

Ejemplo

1. $E \rightarrow E + E$
2. $E \rightarrow E * E$
3. $E \rightarrow x$
4. $E \rightarrow y$

Con esta gramática, para la expresión x + y * x existen los dos árboles de derivación de las figuras



Ejemplo

- En este ejemplo, el hecho de que existan dos árboles de derivación para una misma expresión es indeseable, pues cada árbol indica una manera distinta de estructurar la expresión.

- En efecto, en el árbol de la izquierda, al resultado de la suma (x + y) se multiplica con x, mientras que en el de la derecha sumamos x al resultado de multiplicar x con y; por lo tanto el significado que se asocia a ambas expresiones puede ser distinto.

Ejemplo

- Se dice que una gramática es ambigua ssi alguna palabra del lenguaje que genera tiene más de un árbol de derivación.
- Nótese que la ambigüedad, como la estamos definiendo, es una propiedad de la gramática, no de su lenguaje generado.
- Para un mismo lenguaje puede haber una gramática ambigua y una no ambigua.

Transformación de las GLC y Forma Normal de Chomsky

- En ocasiones es necesario expresar una GLC siguiendo un formato más preciso de las reglas que la simple forma

$$A \rightarrow \alpha$$

- Estos “estándares” reciben el nombre de formas normales.

- FNCH consiste en que las reglas pueden tener dos formas.

- Se utiliza para eliminar la ambigüedad de la gramática.

FNCH

1. $A \rightarrow a,\ a \in \Sigma$

2. $A \rightarrow BC,\ \text{con}\ B, C \in V$

Ejercicio

- S → A | BCa | aDcd | EDF
- A → aAb | c
- B → CD | b | ECd | Ad
- C → Cc | Bb | AaE | ε
- D → aDd | Dd | ε
- E → aaEB | EFG
- F → aFd | d

FNCH

- Eliminar símbolos inútiles: un símbolo es **inútil** cuando el mismo no es generador o bien no es alcanzable.

- Para determinar si un símbolo x es generador usamos el siguiente procedimiento, dada
- G = (V, S, V_o, →):
- Generador (x):
  - x pertenece a S entonces x generador (todos los símbolos terminales de una gramática son generadores, ya que se generan a sí mismos)

  - x no pertenece a S pero entonces existe un α tal que x → α entonces x generador (toda aquella variable que tenga por lo menos una producción que esté conformada únicamente de símbolos generadores).
- Si ninguna de estas condiciones se cumple, entonces x **no es generador**.

FNCH

- $S \rightarrow A \mid BCa \mid aDcd$
- $A \rightarrow aAb \mid c$
- $B \rightarrow CD \mid b \mid Ad$
- $C \rightarrow Cc \mid Bb \mid \varepsilon$
- $D \rightarrow aDd \mid Dd \mid \varepsilon$
- $F \rightarrow aFd \mid d$

FNCH

- Para determinar si un símbolo x es alcanzable usamos el siguiente procedimiento, dada $G = (V, S, V_o, \rightarrow)$:

- Alcanzable (x):
  - $x = V_o$ entonces x **alcanzable**
  - $x \neq V_o$ pero entonces existe un y alcanzable en V tal que $y \rightarrow x$ entonces x **alcanzable**

- En síntesis: Símbolos que no son derivables (directa o indirectamente) desde el símbolo de inicio S

FNCH

- $S \rightarrow A \mid BCa \mid aDcd$
- $A \rightarrow aAb \mid c$
- $B \rightarrow CD \mid b \mid Ad$
- $C \rightarrow Cc \mid Bb \mid \varepsilon$
- $D \rightarrow aDd \mid Dd \mid \varepsilon$

FNCH

- ***Eliminar Producciones ε***
- La eliminación de producciones ε consiste principalmente en eliminar las producciones de la forma $X \rightarrow \varepsilon$.

- Cuando hagamos esto, debemos contemplar la posibilidad de que, en donde antes aparecía la variable X, ésta ahora podrá aparecer o no aparecer.

FNCH

- ¿Qué ocurre si una variable tiene una producción que está compuesta de varios símbolos que pueden ser reemplazables por el string vacío?

- Por ejemplo, digamos que tenemos $Y \rightarrow X_1X_2X_3...X_n$, donde todos los X_i pueden ser el string ε. Entonces nos quedaría algo así: $Y \rightarrow \varepsilon\varepsilon\ \varepsilon...\varepsilon$, que se puede simplificar por $Y \rightarrow \varepsilon$.

- Aquí vemos que Y también puede generar al string vacío.

- Entonces vamos a definir a los símbolos **Anulables. Un símbolo anulable Y** va a ser aquel que tenga entre sus producciones a la producción $Y \rightarrow \varepsilon$, o también aquel que tenga por lo menos una producción en donde todas las variables involucradas son anulables a su vez.

FNCH

- Viendo nuestro ejemplo, podemos identificar rápidamente a dos variables que son símbolos anulables: las variables C y D, ya que ambas tienen la producción de la forma $Y \rightarrow \varepsilon$.

- Son las únicas variables anulables?.

FNCH

- También está B que tiene la producción B → CD, la cual está compuesta enteramente por símbolos anulables.
- En cada producción en la que estaban presentes uno o más símbolos anulables, vemos todas las combinaciones en las que cada una de dichas variables pudiera estar presente o no.
- Por ejemplo, si tenemos la producción Y → abX1, donde X1 es anulable, entonces se generan las producciones Y → abX1 y Y → ab, una en donde X1 está presente y otra en la no.
- Veamos el caso para producciones con dos y tres variables anulables:
- Y → aX1bbX2 => Y → aX1bbX2 | aX1bb | abbX2 | abb
- Y → cX1X2ddX3a => Y → cX1X2ddX3a | cX1X2dda | cX1ddX3a | cX2ddX3a | cX1dda | cX2dda | cddX3a | cdda

FNCH

- S → A | BCa | Ba | Ca | a | aDcd | acd
- A → aAb | c
- B → CD | C | D | b | Ad
- C → Cc | c | Bb | b
- D → aDd | ad | Dd | d

FNCH

- ***Eliminar Producciones Unitarias***
- Las producciones unitarias son aquellas que tienen una única variable en su cuerpo.

- Si tenemos por ejemplo una producción $Y \rightarrow X$, la idea es reemplazar esta producción por todas las producciones de X.

FNCH

- Hay una forma más general de describir este proceso, que involucra la definición de los pares unitarios (Y,X), que se definen así:

- **Base**: El par (Y,Y) es unitario por definición, para toda variable de una gramática dada.

- **Inducción**: Si el par (Y,X) es unitario y la variable X tiene una producción unitaria de la forma $X \rightarrow Z$, entonces (Y,Z) también es un par unitario.

FNCH

- S → aAb | c | BCa | Ba | Ca | a | aDcd | acd
- A → aAb | c
- B → CD | Cc | c | Bb | b | aDd | ad | Dd | d | Ad
- C → Cc | c | Bb | b
- D → aDd | ad | Dd | d

FNCH

- ***Reemplazar Terminales por Variables.***

- El primer paso consiste en reemplazar todos aquellos símbolos terminales que están acompañados en alguna producción por uno o más símbolos.

- A estos terminales los reemplazamos por nuevas variables que tendrán una única producción (la producción de dicho símbolo). En nuestro caso, hay que crear 4 nuevas variables, una para cada símbolo terminal.

FNCH

- $S \rightarrow A_1AB_1 \mid c \mid BCA_1 \mid BA_1 \mid CA_1 \mid a \mid A_1DC_1D_1 \mid A_1C_1D_1$
- $A \rightarrow A_1AB_1 \mid c$
- $B \rightarrow CD \mid CC_1 \mid c \mid BB_1 \mid b \mid A_1DD_1 \mid A_1D_1 \mid DD_1 \mid d \mid AD_1$
- $C \rightarrow CC_1 \mid c \mid BB_1 \mid b$
- $D \rightarrow A_1DD_1 \mid A_1D_1 \mid DD_1 \mid d$
- $A_1 \rightarrow a$
- $B_1 \rightarrow b$
- $C_1 \rightarrow c$
- $D_1 \rightarrow d$

FNCH

- $S \rightarrow A_1Y_1 \mid c \mid BY_2 \mid BA_1 \mid CA_1 \mid a \mid A_1Y_3 \mid A_1Y_4$
- $Y_1 \rightarrow AB_1$
- $Y_2 \rightarrow CA_1$
- $Y_3 \rightarrow DY_4$
- $Y_4 \rightarrow C_1D_1$
- $A \rightarrow A_1Y_1 \mid c$
- $B \rightarrow CD \mid CC_1 \mid c \mid BB_1 \mid b \mid A_1Y_5 \mid A_1D_1 \mid DD_1 \mid d \mid AD_1$
- $Y_5 \rightarrow DD_1$
- $C \rightarrow CC_1 \mid c \mid BB_1 \mid b$
- $D \rightarrow A_1Y_5 \mid A_1D_1 \mid DD_1 \mid d$
- $A_1 \rightarrow a$
- $B_1 \rightarrow b$
- $C_1 \rightarrow c$
- $D_1 \rightarrow d$