Tecnológico de Costa Rica

San José

Compiladores e Intérpretes

Apuntes de la clase del día miércoles 03 de mayo del 2017

Apuntador:
Kevin Lobo Chinchilla (2015088135)

Profesor:
Dr. Francisco Torres Rojas

I Semestre 2017

Quiz 08

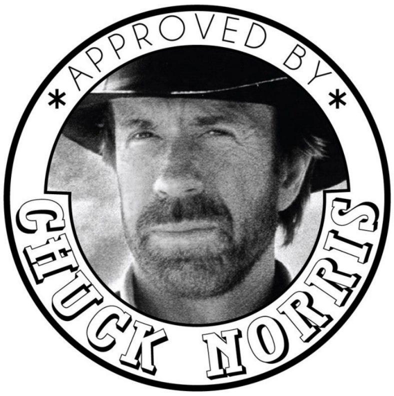
1. Usando el pumping lemma demuestre que el lenguaje de palíndromos sobre {0, 1} no es regular.
2. Presente una CFG que genere palíndromos no vacíos sobre {x, y, z}.

Repaso de la Clase Anterior

Ejemplo 6

Presente una CFG que genere el lenguaje $\{a^n b^m \mid n \geq 0, m \geq 0\}$

$$\mathbf{S} \rightarrow \mathbf{AB}$$
$$\mathbf{A} \rightarrow a\mathbf{B} \mid \varepsilon$$
$$\mathbf{B} \rightarrow b\mathbf{B} \mid \varepsilon$$



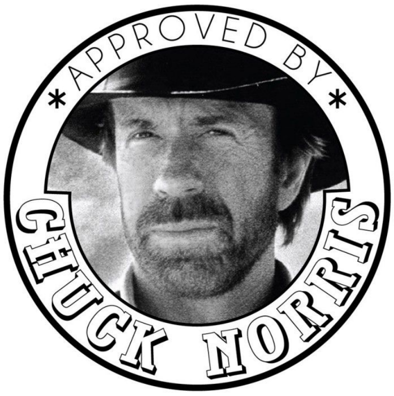
Ejemplo 7

Sea G definida como:

$$\mathbf{S} \rightarrow a\mathbf{S}a$$
$$\mathbf{S} \rightarrow a\mathbf{B}a$$
$$\mathbf{B} \rightarrow b\mathbf{B}$$
$$\mathbf{B} \rightarrow b$$

¿Cuál es el lenguaje de G?

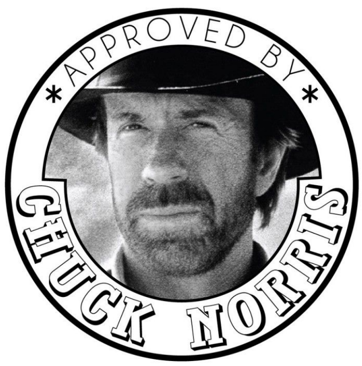
$$\{a^n b^m a^n \mid n > 0, m > 0\}$$



Ejemplo 8

Presente una CFG que genere el lenguaje $\{a^n b^m c^m d^{2n} \mid n \geq 0, m \geq 0\}$

$$\mathbf{S} \rightarrow a\mathbf{S}dd$$
$$\mathbf{S} \rightarrow \mathbf{B}$$
$$\mathbf{B} \rightarrow b\mathbf{B}c$$
$$\mathbf{B} \rightarrow bc$$



500 Ejemplos más de CFG

Ejemplo 9

Presente una CFG que genere el lenguaje de palíndromos sobre Σ = {0, 1}

Forma 1: Separa por casos

S → 0**S**0 | 1**S**1 | **E**

E → 0 | 1 | ε

Forma 2: Más directa

S → 0**S**0 | 1**S**1 | 0 | 1 | ε

Solución de la 2 del Quiz

Presente una CFG que genere palíndromos no vacíos sobre {x, y, z}.

En esta había que tener cuidado, por que si se soluciona como la anterior quitando el ε estaría incorrecta ya que se omiten los casos de hileras de tamaño 2. La forma correcta es:

S → **U** | **D**

U → x | y | z

D → x**A**x | y**A**y | z**A**z

A → **D** | **U** | ε

Ejemplo 10

Presente una CFG que genere el lenguaje de palíndromos sobre Σ = {0, 1} que tengan una longitud impar y un 1 en el centro.

S → 1 | 0**S**0 | 1**S**1

Se logra que la longitud sea impar gracias al 1 en el centro siempre. Para lograr que sea par basta con no colocar nada en el centro, es decir, ε.

Ejemplo 11

Presente una CFG que genere el lenguaje $\{a^n b^m \mid 0 \leq n \leq m \leq 2n\}$.

Es decir, hay más b's que a's, pero a lo sumo la cantidad de b's será el doble que la cantidad de a's.

Forma 1 (Incorrecta): Esta forma genera hileras con la misma cantidad de a's y b's, o con el doble de b's que de a's. Omite el caso en el que hay más b's que a's sin necesidad de que sea el doble.

S → **A** | **B**

A → a**A**b | ε

B → a**B**bb | ε

Forma 2 (Correcta):

S → a**S**b | a**S**bb | ε

Ejemplo 12

Presente una CFG que genere el lenguaje $\{a^n b^n \mid n \geq 0\}$.

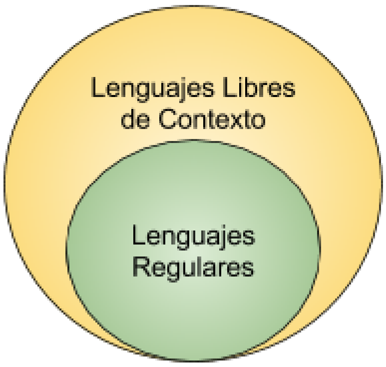
Este fue el primer lenguaje no regular que vimos en clase, al crear una CFG podemos determinar que aunque un lenguaje no sea regular, es libre de contexto.

Le César en medias: $S \rightarrow aSb \mid \varepsilon$

"Hasta sin zapatos se pueden hacer gramáticas"

- Torres

Todo lenguaje regular es libre de contexto, pero no todo lenguaje libre de contexto es regular. Este es un ejemplo de que existen lenguajes libres de contexto y no regulares. Esto gracias a la siguiente sección de la jerarquía de Chomsky:



Todos los lenguajes que hemos visto tienen CFG. Bison utiliza lenguajes libres de contexto.

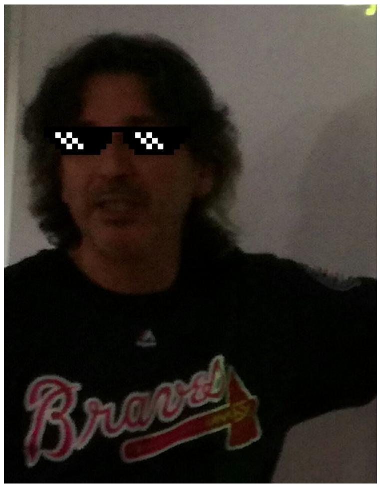
Ejemplo 13

Según Torres es vacilón. Presente una CFG que genere el lenguaje de hileras de paréntesis correctamente anidados.

S → (**S**) | **SS** | ε

Colochos 1 preguntó cómo podría generar (()()()) y nuestro señor Torres hizo lo siguiente:

S => (S) => (SS) => (SSS)



Durante este ejercicio le rogamos al profe más tiempo para la progra, y nos decía que no :'v con el violín más pequeño del mundo. Liza le ofreció galletas y garbanzos y nada… Y nos salió con la orquesta más pequeña del mundo :"v… ***Laurasad****. "Mucho blabla y nada de paréntesis" -Torres*

Ejemplo 14

Presente una CFG que genere expresiones aritméticas con sumas, restas, multiplicaciones y el uso de paréntesis.

Exp → **Exp OP Exp**

Exp → (**Exp**)

Exp → <número>

OP → + | - | *

Definición

Dos CFG *(diferentes)* son equivalentes si generan exactamente el mismo lenguaje.

Ejemplo 15

Presente una CFG que genere el lenguaje a^+b^*

S → **AB**	**S** → a**S** | a**B**
A → a**A** | a	**B** → b**B** | ε
B → b**B** | ε	

Ambas CFG son equivalentes y aceptan ε.

Ejemplo 16

Presente una CFG que genere a*ba*ba* *(sólo contiene 2 b's)*

S → **A**b**A**b**A**	**S** → a**S** | b**A**
A → a**A** | ε	**A** → a**A** | b**C**
C → a**C** | ε	

En la gramática de la izquierda, se crea la CFG de una manera estructural, y S lo que hace es agregar de una vez las 2 b's, luego mediante A se agregan las A que se requieran, y aterrizamos gracias a ε.

En la gramática de la derecha, se hace una manera como por casos, como basándonos en el DFA. Por decirlo de alguna manera, parece un DFA codificado en CFG. Esto se verá más adelante.

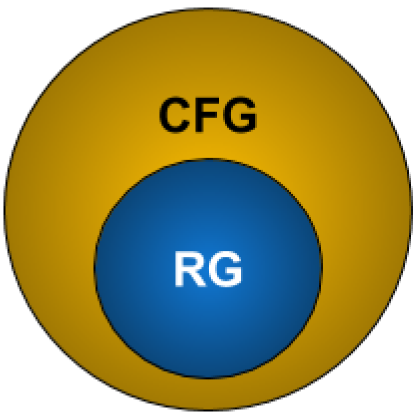
Análisis Sintáctico

Gramáticas Regulares

Definición: Una gramática regular es una CFG G = (V, Σ, P, S), donde toda regla tiene una de las siguientes formas:

1. **A** → a
2. **A** → a**B**
3. **A** → ε

Con **A**, **B** ∈ V y a ∈ Σ.



CFG: Context Free Grammar

RG: Regular Grammar

Toda RG es una CFG, pero no toda CFG es RG.

¿Están los DFA relacionados con las RG?

Somos monstruos peludos en gramáticas y DFAs.

Ejemplo 1

Presente una gramática regular que genere el lenguaje $(ab)^{+}a^{*} \mid \varepsilon$

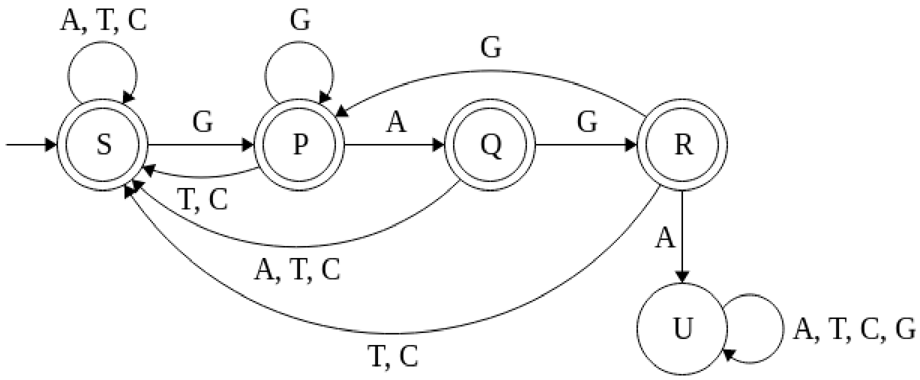
Al ser una gramática regular, se restringe la forma de la CFG, lo que dificulta un poco el ejercicio. En este lenguaje lo que se espera es una hilera que inicie con la subhilera ab una o más veces (ababab...ab) seguida de 0 o más a's; o bien, una hilera vacía.

S → a**B** | ε
B → b**S** | b**A**
A → a**A** | ε

Ejemplo 2

Sea L el lenguaje sobre Σ = {A, T, C, G} de hileras que no contengan la subhilera "GAGA". Presente una gramática regular que genere a $\mathcal{L}$.

Si recordamos bien, este lenguaje lo habíamos utilizado para crear un DFA que lo reconociera, el cuál es el siguiente:



Ahora observen la gramática:

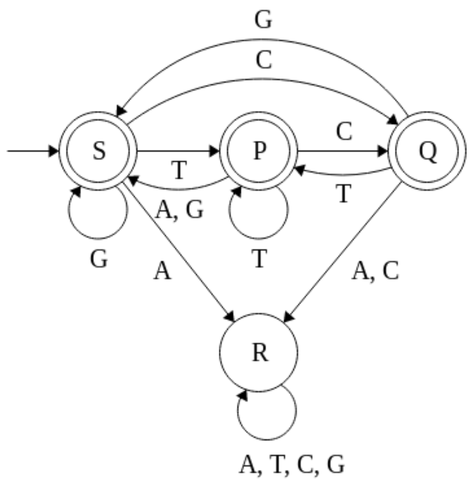
S → A**S** | T**S** | C**S** | G**P** | ε
P → A**Q** | T**S** | C**S** | G**P** | ε
Q → A**S** | T**S** | C**S** | G**R** | ε
R → T**S** | C**S** | G**P** | ε

Se puede notar que tiene relación con el DFA, los no terminales son los estados a los que el terminal que lo acompaña dirige. Los estados de aceptación tienen ε, que es lo mismo a cuando se termina la hilera. Además, los estados sink se eliminan.

Ejemplo 3

Sea ℒ el lenguaje sobre Σ = {A, T, C, G} de hileras que no contengan la subhilera "CC" y donde toda A es precedida de una T. Presente una gramática regular que genere a ℒ.

Veamos el DFA que reconoce a este lenguaje:



Ahora veamos la gramática regular que genera al lenguaje:

S → T**P** | C**Q** | G**S** | ε

P → A**S** | T**P** | C**Q** | G**S** | ε

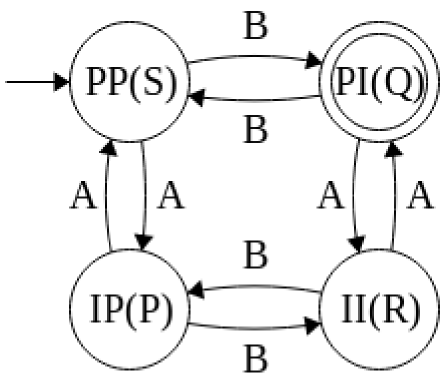
Q → T**P** | G**S** | ε

Nótese que GS es un ciclo. También hay que recordar que una gramática es un ***generador****, mientras que un DFA es un* ***reconocedor****.*

Ejemplo 4

Sea ℒ un lenguaje sobre Σ = {A, B} de hileras que contengan un número par de A's y un número impar de B's. Presente una gramática que genere a ℒ.

DFA:



Gramática:

S → AP | BQ

P → AS | BR

R → AQ | BP

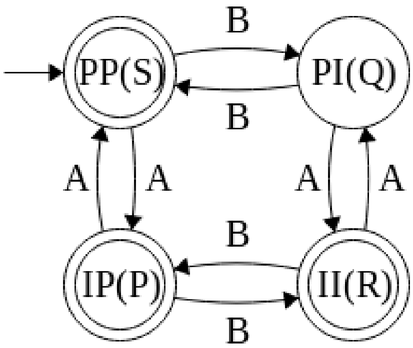
Q → AR | BS | ε

Torres abrió la malparida puerta de vidrio que tanto odiamos para darnos aire, como darle comida a los peces, hasta la mímica hizo, pero luego no quiso repetirla más de una vez :(.

Ejemplo 5

Sea ℒ un lenguaje sobre Σ = {A, B} de hileras que contengan un número impar de A's o un número par de B's. Presente una gramática que genere a ℒ.

DFA:



Gramática:

S → AP | BQ | ε

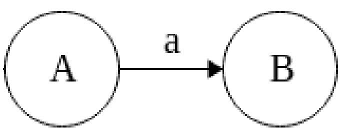
P → AS | BR | ε

R → AQ | BP | ε

Q → AR | BS

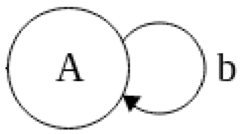
Teorema de Kleene

- A todo NFA/DFA le corresponde una Gramática Regular y viceversa
- De NFA/DFA a Gramática Regular:
    1) A cada estado del NFA/DFA le corresponde un **No terminal**.
    2) Por cada transición de la forma:



En una gramática regular se ve como:

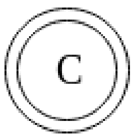
A → aB



En una gramática regular se ve como:

A → bA

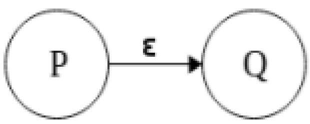
3) Todo estado de aceptación:



En una gramática regular se ve como:

C → ε

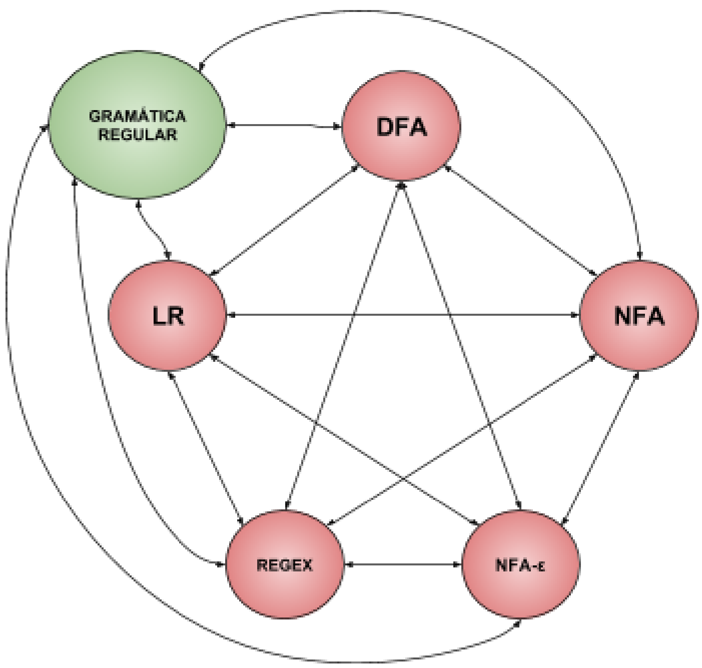
¿Qué pasa con un NFA-ε en las transiciones ε?



➡ P → εQ ➡ P → Q

Supongamos que P → εQ cumple con A → aB

Nota: Las propiedades de cierre de los lenguajes formales las tienen las gramáticas regulares.



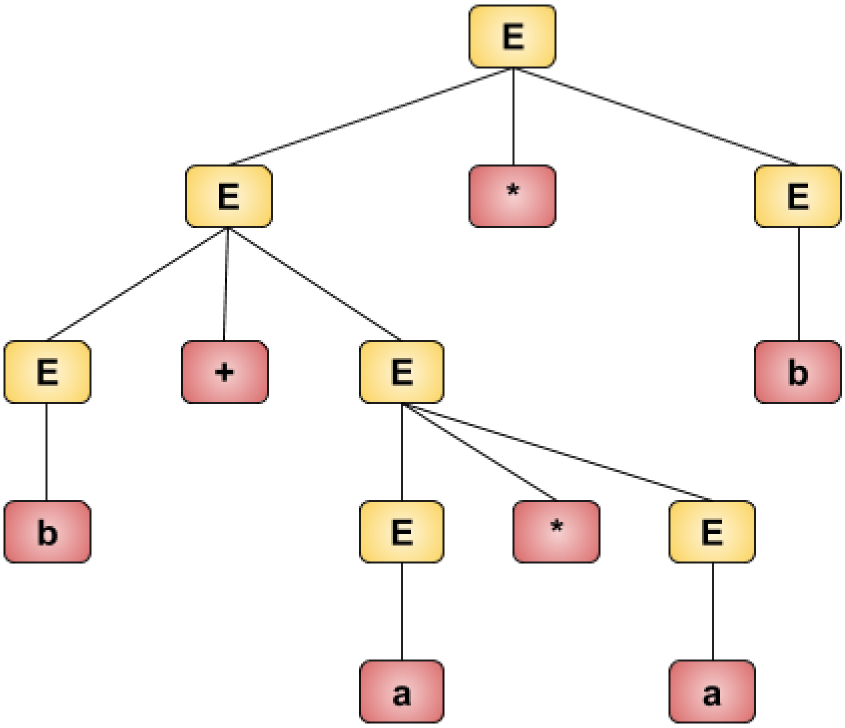
Árboles de Derivación

- Una derivación se puede representar como un **árbol de derivación**.
- Las hojas son terminales, los nodos intermedios son no terminales.
- El símbolo inicial de la gramática es la raíz del árbol.
- Cada vez que se expande un no terminal con una regla, sus componentes pasan a ser los hijos de este nodo.
- Si se leen las hojas de izquierda a derecha, se tiene la hilera derivada.

Ejemplo 1

- Sea G definida como:
  E → **E** + **E**
  E → **E** * **E**
  E → a
  E → b

- **E**
- ⇒ **E** * E
- ⇒ **E** + E * E
- ⇒ b + **E** * E
- ⇒ b + **E** * E * E
- ⇒ b + a * **E** * E
- ⇒ b + a * a * **E**
- ⇒ b + a * a * b



Ejemplo 2

- Considere la CFG:

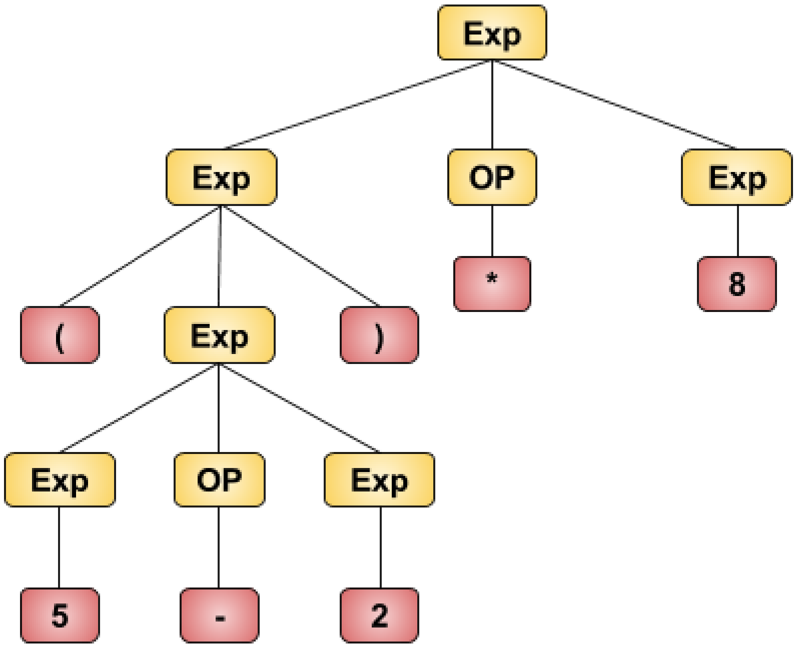
  Exp → **Exp OP Exp**

  Exp → (**Exp**)

  Exp → <número>

  OP → + | - | *

- Hilera: (5-2)*8



El árbol nos brinda una mejor visualización. Si no se puede hacer un árbol de derivación significa que hay un error de sintaxis.

¿Para Qué el Árbol de Derivación?

- Es equivalente a la derivación.
- Su existencia demuestra que la hilera (programa) es correcta sintácticamente.
- ¿Qué pasa si no hay forma de construir un árbol de derivación para una hilera? Significa que hay **error de sintaxis**.
- Es insumo de otras fases del compilador
  - **Análisis Semántico.**
  - **Generación de Código.**
- Un árbol muestra la interpretación que le daremos al código.

Ambigüedad

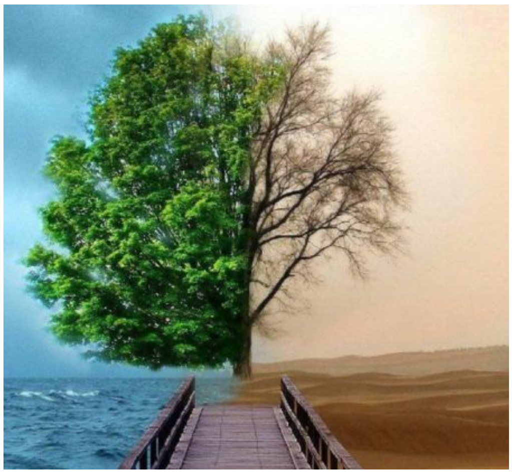
- Una información es ambigua si puede tener más de una interpretación.
- Ocurre el lenguaje natural:
    - “Se murió la perra de María”.
    - “Juan vio a Pedro. Él usaba binoculares”.
- Con información adicional conocida a *priori* o haciendo preguntas adicionales se puede romper la ambigüedad.
- Problema equivalente en Lenguajes Formales y compiladores.

Se murió la perra de María tiene tres interpretaciones según Torres:

- *María es una bitch (o una hp).*
- *Se murió la <u>perra</u> de María.*
- *La perra se llamaba María.*

Si antes de decir la frase recordamos lo que dijo la niña Marlen de que perra es un insulto, se podría evitar esta confusión. En este momento, Torres dijo que perro es un insulto porque son brutos, o sea no tienen inteligencia; y bueno, esto pasó en la clase:

Definición



Una CFG es **ambigua** si existe al menos una hilera para la que haya al menos 2 árboles de derivación más izquierda (o al menos 2 árboles de derivación más derecha) que sean **diferentes**. *(Gramática malparida).*

Ejemplo 1

- Considere la CFG:

  Exp → **Exp OP Exp**

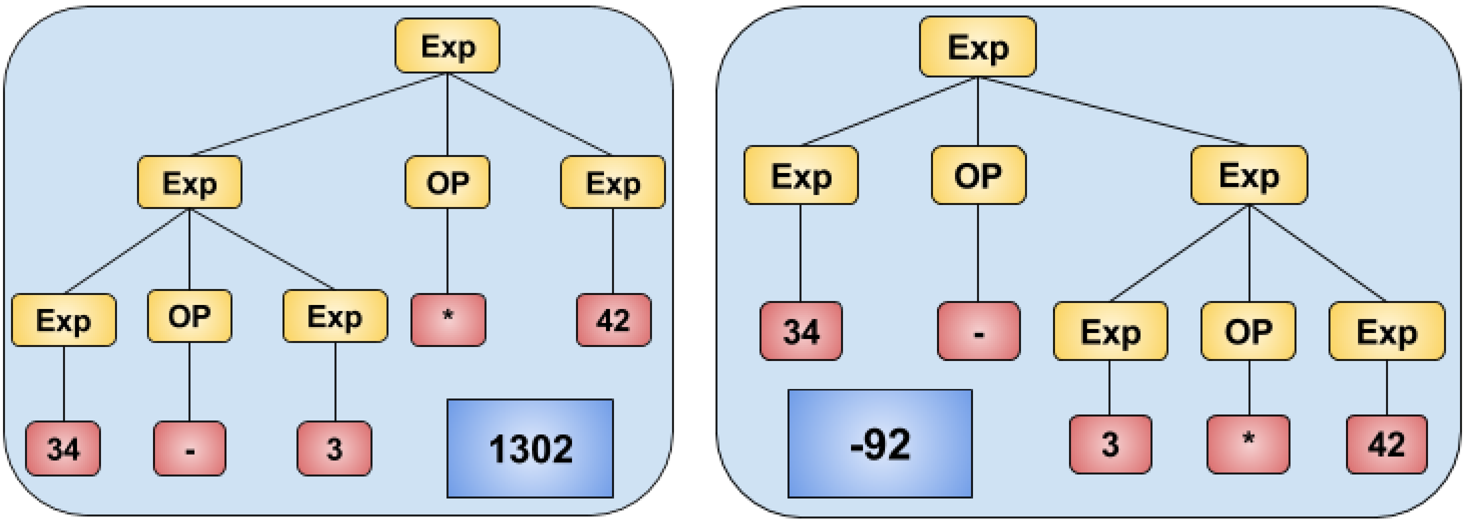
  Exp → (**Exp**)

  Exp → <número>

  OP → + | - | *

- Hilera: 34-3*42

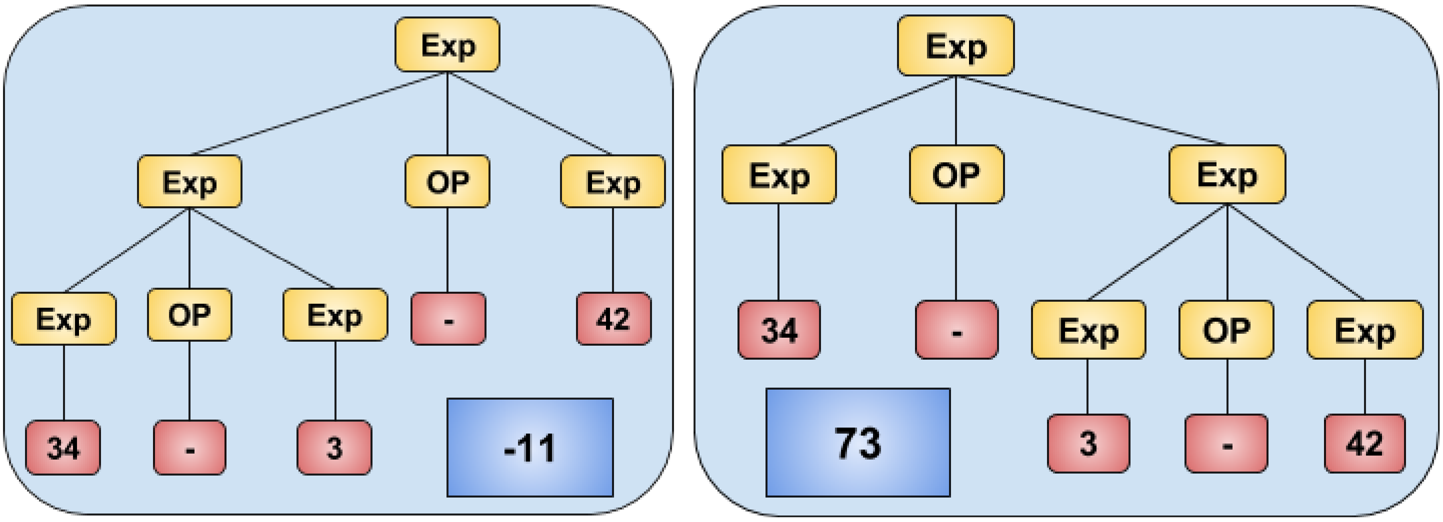
Esta CFG puede generar dos árboles de derivación con diferentes resultados:



*Para calcular el resultado de estos árboles, se lee y calcula desde lo más bajo del árbol hasta lo más alto. Por ejemplo: 34 - 3 = 31, 31 * 42 = 1302.*

Ejemplo 2

Utilizando la CFG del ejemplo anterior y la hilera “34-3-42”. *(Mismo operador malparido).* También se produce ambigüedad *(ambigüedad bitch).*



Aunque dos árboles diferentes generen el mismo resultado (por ejemplo si se usara + en lugar de - en el ejemplo) la gramática sigue siendo ambigua dado que los árboles son diferentes.

Ambigüedad en CFG

- ¿Cómo se demuestra que una CFG es ambigua?
- ¿Cómo se demuestra que una CFG no es ambigua?
- ¿Si se sabe que una CFG es ambigua se puede hacer algo?
  - Regla de desambiguación.
- No hay algoritmos generales conocidos para determinar la ambigüedad en una gramática.
- Los lenguajes de programación típicos incluyen ambigüedades pero estas pueden ser manejadas apropiadamente.

Recordatorios

- Quiz el miércoles 10 de mayo.
- Después del colorido comentario de Colochos 1, cuando perdimos toda esperanza, Torres nos permitió entregar el Proyecto 2 el miércoles 10 de mayo.
- Están por dejarnos una tarea, así que mejor ponerse las pilas para apartar el tiempo.
- ¡Éxitos en lo que queda del semestre!