Práctica 7: Gramáticas libres de contexto

Versión del 18 de marzo de 2024

Ejercicio 1. Para cada uno de los lenguajes del ejercicio 1 de la práctica 6:

a. Dar una gramática libre de contexto que lo genere.

b. Elegir una cadena del lenguaje de longitud mayor o igual a 4, y exhibir una derivación más a la izquierda, una derivación más a la derecha y un árbol de derivación según la gramática dada.

Ejercicio 2. Demostrar que:

a. $\mathcal{L}$ es libre de contexto $\Longrightarrow \mathcal{L}^2$ es libre de contexto.

b. $\mathcal{L}$ es libre de contexto $\Longrightarrow \forall n \in \mathbb{N}$, $\mathcal{L}^n$ es libre de contexto.

c. $\mathcal{L}$ es libre de contexto $\Longrightarrow \mathcal{L}^*$ es libre de contexto.

d. $\mathcal{L}$ es libre de contexto $\Longrightarrow \mathcal{L}^{\mathrm{r}}$ es libre de contexto.

e. $\mathcal{L}_1$ y $\mathcal{L}_2$ son libres de contexto $\Longrightarrow \mathcal{L}_1 \cup \mathcal{L}_2$ es libre de contexto.

Ejercicio 3. Dar una gramática libre de contexto para cada uno de los siguientes lenguajes:

a. Cadenas sobre $\{a, b\}$ cuya longitud es impar y cuyo símbolo central es a.

b. Cadenas sobre $\{a, b\}$ que no son de la forma $\omega\omega$ para ningún $\omega \in \{a, b\}^*$.[1]

c. $\{a^n b^{2m} \mid n \neq m\}$.

d. $\{\omega\#1^n \mid \omega \in \{a, b\}^* \wedge n = (\text{cantidad de apariciones de } ab \text{ en } \omega)\}$.

Ejercicio 4. Dado el alfabeto $\{a, b, 1,$ `[`, `]`, `,`, `:`$\}$, sea $\mathcal{L}$ el lenguaje de las cadenas que poseen las siguientes características:

- consisten en listas de elementos separados por comas y rodeados por corchetes;
- los elementos de las listas pueden ser cadenas no vacías compuestas de los caracteres a y b, en cuyo caso se desea que la cantidad de ambos símbolos sea la misma;
- los elementos de las listas también pueden ser otras listas, es decir, se puede tener listas anidadas;
- al final de cada lista (pero dentro de los corchetes) aparece su cantidad de elementos, expresada en base unaria y precedida por el símbolo `:` (dos puntos);
- una lista puede estar vacía, en cuyo caso se omite el símbolo `:` y se escribe `[]`.

Por ejemplo, la siguiente cadena pertenece a $\mathcal{L}$: `[`$abba$`,[`ab`,`$baba$`:`11`],`ba`,[]:`1111`]`.

a. Dar una gramática independiente del contexto para $\mathcal{L}$.

b. Exhibir un árbol de derivación para la cadena dada como ejemplo. ¿Es único?

[1] *Pista*: Usar el inciso anterior.

Ejercicio 5. Considerar el alfabeto { `(`, `)`, `[`, `]` }. Dar una gramática libre de contexto para cada uno de los siguientes lenguajes:

a. El lenguaje $\mathcal{L}_1$ de las cadenas balanceadas que tienen a lo sumo dos niveles de anidamiento seguidos correspondientes a corchetes.

Por ejemplo, las siguientes cadenas están en $\mathcal{L}_1$:
- `()()()`, `[][][]`, `[[]]`, `((((()))))`, `(()()[])[[]]`.
- `[[([])]]`, `[([[]])]` (ya que los paréntesis reinician el anidamiento).

En cambio, las siguientes cadenas no están en L1:
- `(`, `)[`, `][][`, `[)(]` (ya que no están balanceadas).
- `[[[]]]`, `[[[]()]]`, `[()()[()[]]()]` (por tener más de dos niveles de anidamiento de corchetes seguidos).

b. El lenguaje $\mathcal{L}_2$ de las cadenas que tienen a lo sumo un error de balanceo, ya sea de paréntesis o de corchetes. Contamos un error en el balanceo de un símbolo por cada símbolo de apertura al que no le corresponde un símbolo de cierre, o viceversa.

Por ejemplo, las siguientes cadenas están en $\mathcal{L}_2$: `[()]()`, `[])`, `(()[]()`, `([([)][])`.

En cambio, las siguientes cadenas no están en $\mathcal{L}_2$:
- `(]` (tiene un error de cada tipo).
- `]([[()])` (tiene dos errores de balanceo de corchetes).

c. El lenguaje $\mathcal{L}_3$ de las cadenas balanceadas pero donde cada corchete puede cerrar o bien un corchete o bien una *secuencia completa* de paréntesis abiertos.

Por ejemplo, la siguiente cadena está en $\mathcal{L}_3$:

`(([]][(([](((()](]]`

cierra 1 paréntesis
cierra 5 paréntesis
cierra 2 paréntesis

En cambio, las siguientes cadenas no están en $\mathcal{L}_3$:
- `()(()]` (el último corchete no tiene nada que cerrar).
- `[((]` (el primer corchete queda abierto).
- `([((]]` (el primer paréntesis queda abierto).
- `(((])` (el último paréntesis no tiene nada que cerrar).

Ejercicio 6. Dar una gramática libre de contexto que genere el lenguaje de las fórmulas bien formadas de la lógica de predicados de primer orden, utilizando:

- las variables x, y.
- las constantes c, d.
- los símbolos de predicado p, q, r, s (con cualquier aridad no nula).
- los conectivos lógicos $\neg, \wedge, \vee, \Rightarrow$.
- los cuantificadores $\forall, \exists$.

Por ejemplo: $\forall x\,(\exists y\,(p(x,y))) \Rightarrow \exists x\,((q(x,c) \wedge r(d,x)) \vee \neg s(x))$.

Ejercicio 7.

a. Dar una gramática para expresiones aritméticas sobre identificadores con suma, resta, producto, división y paréntesis. El símbolo del producto se puede omitir.

b. Si es necesario, modificar la gramática dada de manera que no sea ambigua y respete las reglas usuales de asociatividad (todas las operaciones son asociativas a izquierda) y precedencia (la precedencia de la suma y la resta es menor que la del producto y la división).

c. Dar el árbol de derivación de la expresión **id-id id/id*id+id**.

Ejercicio 8. Una lista en el lenguaje Prolog se puede representar como una secuencia de elementos encerrados entre corchetes y separados por comas. Los elementos de la lista pueden ser a su vez listas. De esta manera, los siguientes son ejemplos de listas:

$$A_1 = \texttt{[a]} \qquad A_2 = \texttt{[]}$$
$$A_3 = \texttt{[a,[b,c],d]} \qquad A_4 = \texttt{[[],[a,[a],b,[[]]]]}$$

Llamaremos $\mathcal{L}$ al lenguaje sobre el alfabeto $\Sigma = \{\texttt{[},\texttt{]},\texttt{,},\textbf{id}\}$ formado por las listas recién descriptas. Dar una gramática no ambigua para $\mathcal{L}$.

Ejercicio 9. Consideramos una sintaxis simplificada para expresiones del lenguaje Common Lisp: una expresión puede ser un átomo o una lista. Los átomos pueden ser símbolos, números, o cadenas (terminales **sym**, **num** y **str**, respectivamente). Una lista es una secuencia de expresiones encerradas entre paréntesis: `(` y `)`. Las listas pueden ser vacías. Ejemplos de expresiones válidas pueden ser entonces:

num
`()`
`(` **sym** `()` **num str** `)`
`( (` **num** `)` **sym** `(` **sym num str** `() ) )`

Llamaremos $\mathcal{L}$ al lenguaje sobre el alfabeto $\Sigma = \{\texttt{(},\texttt{)},\textbf{sym},\textbf{str},\textbf{num}\}$ formado por las expresiones recién descriptas. Dar una gramática no ambigua para $\mathcal{L}$.

Ejercicio 10. Una fórmula química es una manera concisa de expresar información sobre los átomos que constituyen un compuesto. Cada elemento es identificado por su símbolo químico y la cantidad de átomos de cada elemento es indicada por un subíndice, si es mayor que uno. Por ejemplo, el metano, una molécula simple compuesta por un átomo de carbono unido a cuatro de hidrógeno, tiene la fórmula química $\mathrm{CH_4}$. Si un ion se repite más de una vez, esto se puede expresar encerrándolo entre paréntesis y agregando un subíndice indicando la cantidad de veces que se repite. Por ejemplo, el sulfato férrico está compuesto por dos átomos de hierro y tres iones sulfato, cada uno de los cuales se compone de un átomo de azufre y cuatro de oxígeno: $\mathrm{Fe_2(SO_4)_3}$. De esta manera, los siguientes son ejemplos de fórmulas químicas:

Oxígeno:	$\mathrm{O_2}$
Agua:	$\mathrm{H_2O}$
Ferrocianuro férrico:	$\mathrm{Fe_2\big(Fe(CN)_3\big)_3}$

Llamaremos $\mathcal{L}$ al lenguaje sobre el alfabeto $\Sigma = \{\texttt{(},\texttt{)},\textbf{elem},\textbf{num}\}$ formado por las fórmulas químicas recién descriptas. Dar una gramática no ambigua para $\mathcal{L}$.

Ejercicio 11. La siguiente gramática representa un fragmento de las expresiones válidas en el lenguaje de programacion C:

$$G = \langle \{E\}, \{\mathbf{id}, \texttt{?}, \texttt{:}, \texttt{+}, \texttt{(}, \texttt{)}\}, P, E \rangle,$$

$$\text{con } P: \quad \begin{array}{lcl} E & \rightarrow & E \texttt{ ? } E \texttt{ : } E \\ E & \rightarrow & E \texttt{ + } E \\ E & \rightarrow & \mathbf{id} \\ E & \rightarrow & \texttt{(} E \texttt{)} \end{array}$$

a. Para las cadenas $\alpha_1 = \mathbf{id} \texttt{ ? } \mathbf{id} \texttt{ : } \mathbf{id} \texttt{ + } \mathbf{id} \texttt{ ? } \mathbf{id} \texttt{ : } \mathbf{id}$ y $\alpha_2 = \mathbf{id} \texttt{ ? } \mathbf{id} \texttt{ ? } \mathbf{id} \texttt{ + } \mathbf{id} \texttt{ : } \mathbf{id} \texttt{ : } \mathbf{id}$, dar todos sus árboles de derivación.

b. Teniendo en cuenta que el operador ternario condicional $\bullet \texttt{ ? } \bullet \texttt{ : } \bullet$ es asociativo a derecha, y tiene menor precedencia que el operador $\texttt{+}$, que es asociativo a izquierda, dar una gramática no ambigua para $\mathcal{L}(G)$. Dar los árboles de derivación resultantes para α_1 y α_2.