# Teorema Chomsky-Schützenberger

Autómatas y lenguajes formales

17 de septiembre de 2021

## Lenguaje de *Dyck*

Este tema en el plan del curso iba antes, pero lo salté, aquí lo recupero pues es necesario para lo que sigue.

Podemos definir un lenguaje para los paréntesis bien balanceados de distinto tipo[1], que llamaremos $D_n$ de la siguiente forma:

$$S \to \underset{1}{[}S\underset{1}{]}|\underset{2}{[}S\underset{2}{]}|...|\underset{n}{[}S\underset{n}{]}|SS|\epsilon,$$

con esas producciones se pueden generar $n$ pares de paréntesis bien anidados.

Un concepto importante en matemáticas y en las ciencias de la computación es la conversión de un problema a otro ya conocido[2], así si en el problema nuevo no encontramos solución, si lo podemos convertir a un viejo conocido con solución, pues ya solo convirtiendo algunas cosas tenemos la solución del nuevo. Esta es la base del teorema de Chomsky-Schützberger.

## Teorema de Chomsky-Schützberger

**Teorema 1** *Todo lenguaje independiente de contexto es una imagen homomorfa de la intersección de un lenguaje de paréntesis y un conjunto regular. En otras palabras, para todo lenguaje independiente de contexto* $\mathbf{A}$*, hay una* $n \geq 0$*, un conjunto regular* $R$ *y un homomorfismo* $h$ *tales que:*

$$\mathbf{A} = h(D_n \cap R)$$

Diciéndolo en otras palabras, todo lenguaje independiente de contexto es una ampliación a un lenguaje regular, ya que es equivalente a la unión de un lenguaje regular con un lenguaje de paréntesis de *Dyck* (que es un ejemplo muy sencillo de un lenguaje independiente de contexto).

El homomorfismo es un mapeo que va de un dominio a otro, en este caso $h : \Gamma^* \to \Lambda^*$, por ejemplo la función $h(x, y) = h(x)h(y)$, donde $x, y \in \Gamma^*$ y $h(x)h(y) \in \Lambda^*$. Como el nombre lo indica, un mapeo pasa punto por punto a

[1] Como ya su experiencia en la matemáticas les habrá hecho ver, cuando usan más de un paréntesis, en ocasiones es útil usar paréntesis diferenciados, por ejemplo si en una misma expresión requieren tres tipos de paréntesis pueden usar [{()}]. Aunque en programación se suele usar sólo un tipo y la programadora/programador o el editor está al pendiente de donde abre y cierra cada tipo.

[2] Incluso hay un chiste sobre cuántos matemáticos se necesitan para cambiar un foco, no me lo sé y para no arruinarlo no lo cuento.

algo muy parecido, como aquellos mapamundis que usaban en la escuela, que pasan el globo terráqueo a una superficie plana, sin dejar fuera ningún país. Una imagen para recordar un poco como es eso se muestra en la figura1.

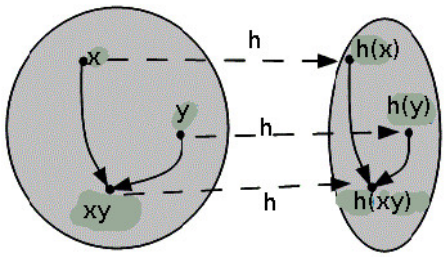


Figura 1: Ejemplo de un homomorfismo, piense que el conjunto de la izquierda es $\Gamma^*$ y el de la derecha $\Sigma^*$. Imagen adaptada de una tomada de la página: https://mathstrek.blog/2012/09/28/casual-introduction-to-group-theory-6/

¿Cómo se hace este homomorfismo de la intersección de un lenguaje de *Dyck* y un lenguaje regular a un lenguaje independiente de contexto? Si recuerdan un lenguaje independiente de contexto se define por un cuarteto del tipo $G = (\Sigma, \Gamma, S, \rightarrow)$, con $\Sigma$ el alfabeto de entrada, $\Gamma$ el alfabeto de la memoria, $S$ la variable inicial y $\rightarrow$ el conjunto de producciones.

Vamos a desglosar un poco este último conjunto, las producciones son varias, en este caso las llamaremos a partir de letras griegas minúsculas. Sólo para identificarlas con las funciones en el nuevo dominio.

$$\{\pi, \rho, \sigma, ...\} \in \rightarrow,$$

esta elección de llamar al conjunto de producciones $\rightarrow$ al momento puede ser medio confuso, pero espero conforme avancemos no lo sea tanto. Otra manera de verlo es como si cada producción tuviera un nombre:

$$\{\underset{\pi}{\rightarrow}, \underset{\rho}{\rightarrow}, \underset{\sigma}{\rightarrow}, ...\} \in \rightarrow,$$

pero sólo las identificaremos por las letras que las designan.

Lo que sigue es transformar las producciones al lenguaje de paréntesis, por ejemplo si se tiene el lenguaje independiente de contexto:

$$A \rightarrow BC$$
$$A \rightarrow a,$$

Las podemos juntar en una sola producción:

$$A \rightarrow BC|a$$

como pueden ver ya en forma normal de Chomsky, a esta le llamamos nuestra producción $\pi$. En el lenguaje de *Dyck* se transforma a:

$$A \to \overset{1}{\underset{\pi}{[}} B \overset{12}{\underset{\pi\pi}{]}} \overset{2}{\underset{\pi}{[}} C \overset{}{\underset{}{]}}$$
$$A \to \overset{1122}{\underset{\pi\pi\pi\pi}{[\,][\,]}},$$

respectivamente. Los números de arriba indican el tipo de paréntesis (un paréntesis del tipo 1 no cierra con uno del tipo 2) y la letra griega minúscula inferior hace referencia a la producción, en este caso $\pi$.

Para hacerlo más claro completemos el lenguaje:

$$A \to BC$$
$$A \to a$$
$$B \to b$$
$$C \to c,$$

aprovechando que ya está en forma normal de Chomsky. Cada regla de producción que involucre a la misma variable del lado izquierdo se marca con el mismo símbolo, así la producción $\pi$ designa a las primeras dos líneas (con la variable $A$), la producción de $B$ la llamaremos $\rho$, y la de $C$ será $\sigma$. Por suerte estas últimas dos solo tienen una regla.

Ponerlo en forma normal de Chomsky lo reduce a que tengamos sólo dos tipos distintos de paréntesis por cada producción, sólo habrá dos tipos para $\pi$, dos tipos para *rho* y dos para $\sigma$, pero un paréntesis de $\pi$ no puede ser cerrado con uno de $\sigma$ o $\rho$, son reglas de producción distintas. El lenguaje se transforma en:

$$A \to \overset{1}{\underset{\pi}{[}} B \overset{12}{\underset{\pi\pi}{]}} \overset{2}{\underset{\pi}{[}} C \overset{}{\underset{}{]}}$$
$$A \to \overset{1122}{\underset{\pi\pi\pi\pi}{[\,][\,]}}$$
$$B \to \overset{1122}{\underset{\rho\rho\rho\rho}{[\,][\,]}}$$
$$C \to \overset{1122}{\underset{\sigma\sigma\sigma\sigma}{[\,][\,]}}.$$

De aquí se pueden ver unas reglas para el acomodo de los paréntesis y de que forma pueden ser identificados unos con variables y otros con terminales, eso ya se los dejo un poco a su análisis, pero éstas son las reglas. Por ejemplo las cadenas en ese lenguaje independiente de contexto son $a$ y $bc$, en el nuevo lenguaje se transforman en $\overset{1122}{\underset{\pi\pi\pi\pi}{[\,][\,]}}$ y $\overset{11122112}{\underset{\pi\rho\rho\rho\rho\pi\pi\sigma\sigma\sigma\sigma\pi}{[\,[\,][\,]]\,[\,[\,][\,]]}}$ respectivamente.

Este lenguaje de puros paréntesis es equivalente al anterior, aunque no lo parezca, esos monstruos de paréntesis ya son un lenguaje independiente de contexto, todos los símbolos son el alfabeto del lenguaje de *Dyck*, pero las reglas para que hagan sentido es la intersección con un lenguaje regular (esta construcción podría representarse por un autómata finito).

## Referencias

[1] Kozen, Dexter C. “Automata and Computability” Springer (1997)