UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE CIENCIAS

ALGUNAS APLICACIONES DEL TEOREMA DE MARKOV-POST A TEORÍA DE GRUPOS

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

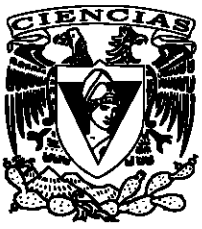
MATEMÁTICO

PRESENTA:

PORFIRIO MAYO MAYO

TUTOR:
MAT. FERNANDO CORNEJO MONTAÑO

COTUTOR:
M. EN C. MARÍA DE LOURDES GUERRERO ZARCO



Ciudad Universitaria, CD. MX., 2018.

Datos del alumno

Apellido paterno	Mayo
Apellido materno	Mayo
Nombre	Porfirio
Universidad	Universidad Nacional Autónoma de México
Facultad	Facultad de Ciencias
Carrera	Matemáticas
Número de cuenta	301040121

Datos del tutor 1

Grado	Mat.
Apellido paterno	Cornejo
Apellido materno	Montaño
Nombre	Fernando

Datos del tutor 2

Grado	M. en C.
Apellido paterno	Guerrero
Apellido materno	Zarco
Nombre(s)	María de Lourdes

Datos del sinodal 1

Grado	Dr.
Apellido paterno	Rincón
Apellido materno	Mejía
Nombre(s)	Hugo Alberto

Datos del sinodal 2

Grado	Dr.
Apellido paterno	Meza
Apellido materno	Alcántara
Nombre	David

Datos del sinodal 3

Grad.	Dr.
Apellido paterno	Alvarado
Apellido materno	García
Nombre	Alejandro

Datos del trabajo escrito

Título	Algunas aplicaciones del Teorema de Markov-Post a Teoría de Grupos
Número de páginas	81
Año	2018

Índice general

Dedicado a
mis padres

Agradecimientos

Quiero empezar agradeciendo a mis padres, por todo el apoyo que me brindaron durante la licenciatura. A mis amigos que me alentaron a terminar este trabajo.

Un agradecimiento singular debo a Fernando, mi tutor y un gran amigo. A mi asesora Lourdes Guerrero, por quien conocí y me adentré en el área de la Lógica Matemática. Agradezco su tiempo y paciencia invertidos en esta tesis. Y a mis sinodales: Dr. Hugo Rincón, Dr. David Meza y Dr. Alejandro Alvarado, cuyas observaciones y correcciones enriquecieron este trabajo.

Introducción

El Problema de la Palabra o Problema de Dehn, es un problema propuesto por el matemático Max Dehn en 1910. Dehn propuso un problema similar al *Entscheidungsproblem* de David Hilbert:

> *¿existe un algoritmo que determine si dos palabras arbitrarias, escritas en términos de sus generadores, son iguales en un grupo finitamente presentado?*

Si bien hay grupos finitamente presentados donde sí existe este algoritmo, en 1947, Andréi Markov y Emil Post, demostraron de manera independiente que existe un semigrupo finitamente presentado donde el problema de la palabra no tiene solución. Más adelante Serguéi P. Novikov (1955), William W. Boone (1957) y John L. Britton (1958), hicieron lo propio para grupos: exhibieron un grupo finitamente presentado donde no existe tal algoritmo.

El presente trabajo aborda el teorema de Markov-Post: mostrar la existencia de un semigrupo libre y finitamente presentado donde no existe algoritmo que determine si dos palabras arbitrarias, expresada en términos de sus generadores, son iguales.

La técnica usada por Markov y Post son parte de la Lógica Matemática y el Álgebra: la máquina de Turing, desarrollada en el Capítulo 2, y los grupos libres, desarrollado en el Capítulo 1. Como consecuencia y aplicación de éste teorema se prueba en el Capítulo 3 el teorema de Novikov-Boone-Britton: la existencia de un grupo finitamente presentado donde el problema de la Palabra no tiene solución.

Capítulo 1

Grupos libres

El objetivo de este capítulo es conocer la estructura algebraica de $W(X)$, el conjunto de las palabras de un conjunto dado X, a partir de $W(X)$ construir un grupo libre L y mostrar la existencia de presentaciones. Las palabras positivas, los semigrupos libres y las presentaciones de grupos son la base algebraica del teorema de Markov-Post. Este capítulo tiene como referencia [11], [7] y [6].

1.1 Semigrupos y monoides

Definición 1.1. Sea X un conjunto. Una *palabra* en X es una sucesión finita $w = x_1 \cdots x_n$ de elementos de X con $n \in \mathbb{N}_0 = \mathbb{N} \cup \{0\}$.

Según el contexto, a las palabras también se les llaman *cadenas* o *cuerdas*.

Al conjunto de todas las palabras sobre X, se le denota por $W(X)$. Este conjunto es no vacío, dado que la *palabra vacía* con cero letras, denotada por 1, es parte del conjunto.

Definición 1.2. La *longitud* $|w|$ de una palabra $w \in W(X)$ es la cantidad de letras o símbolos que la componen.

Ejemplo 1.1. La longitud de la palabra vacía 1 es cero, mientras que la cantidad de letras en la palabra $w = x_1 \cdots x_n$ es n.

Definición 1.3. Dos palabras $x = x_1 \cdots x_n$ y $y = y_1 \cdots y_m$ en $W(X)$, son *iguales* si tienen la misma longitud, es decir $n = m$, y además $x_i = y_i$, para toda $i = 1, \ldots, n$.

Definición 1.4. Sea $W(X)$ el conjunto todas las palabras en X. Se define una operación binaria $\cdot : W(X) \times W(X) \to W(X)$ de la siguiente manera. Para $x, y \in W(X)$ con $x = x_1 \cdots x_n$ y $y = y_1 \cdots y_m$

$$x \cdot y := x_1 \cdots x_n y_1 \cdots y_m$$

Esta operación es llamada la *yuxtaposición* o *concatenación* de x y y, y lo que hace es unir palabras. La concatenación define en $W(X)$ una estructura algebraica.

Definición 1.5. Un conjunto no vacío M y una operación binaria $* : M \times M \to M$ es un *semigrupo* si la operación es asociativa, es decir

$$(x * y) * z = x * (y * z) \quad \text{para todo } x, y, z \in M.$$

Definición 1.6. Un conjunto no vacío M y una operación binaria $* : M \times M \to M$ es un *monoide* si la operación satisface las siguientes condiciones:

1) La operación es *asociativa*, es decir,

$$(x * y) * z = x * (y * z) \quad \text{para todo } x, y, z \in M$$

2) Existe un elemento llamado *neutro* $e \in M$ que satisface

$$e * x = x = x * e \quad \text{para todo } x \in M.$$

La notación usual para un monoide es $(M, *)$, sin embargo, si no hay confusión en la operación, se le denotará simplemente por M. El siguiente resultado es directo de la definición.

Proposición 1.1. *Todo monoide es un semigrupo.*

A menos que se especifique lo contrario, a partir de ahora, el conjunto de los números naturales $\mathbb{N}$ incluye al cero.

Ejemplo 1.2. En el conjunto de los números naturales, $\mathbb{N}$, la suma $+ : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ es asociativa y para cualquier número natural, el 0 es el neutro: $n + 0 = n$. Así $(\mathbb{N}, +)$ es un monoide.

Definición 1.7. Sea $(M, *)$ un monoide con notación multiplicativa. Para cualquier $x \in M$ y $n \in \mathbb{N}$ se define

$$x^n := \begin{cases} e & \text{si} \quad n = 0 \\ x & \text{si} \quad n = 1 \\ x^{n-1} * x & \text{si} \quad n > 2 \end{cases}$$

En cualquier monoide se cumplen las leyes de los exponentes, la demostración es directa de la asociatividad, usando inducción.

Proposición 1.2 (Leyes de los Exponentes)**.** *Sea $(M, *)$ un monoide. Para cualquier $x \in M$ y $m, n \in \mathbb{N}$ se cumple las siguientes propiedades:*

i) $x^n * x^m = x^{n+m}$

ii) $(x^n)^m = x^{nm}$

1.1.1 El monoide W(X)

Teorema 1.3. *Para cualquier conjunto X, el conjunto de las palabras con la concatenación $(W(X), \cdot)$ forma un monoide.*

Demostración. La definición misma del producto en $W(X)$ implica que la palabra vacía 1, con cero letras, es el neutro

$$1 \cdot x_1 \cdots x_n = x_1 \cdots x_n = x_1 \cdots x_n \cdot 1$$

La concatenación de palabras es asociativa. En efecto, si x, y, z son las palabras $x_1 \cdots x_n, y_1 \cdots y_m, z_1 \cdots z_k$ respectivamente; tanto la palabra $(x \cdot y) \cdot z$ y $x \cdot (y \cdot z)$ tienen la misma longitud, $n+m+k$. Para demostrar que son la misma palabra, sólo resta probar que para cualquier índice fijo, tienen los mismos símbolos. Pero esto es directo de la definición de concatenación:

$$\big(x_1 \cdots x_n \cdot y_1 \cdots y_m\big) \cdot z_1 \cdots z_k = x_1 \cdots x_n \cdot \big(y_1 \cdots y_m \cdot z_1 \cdots z_k\big).$$

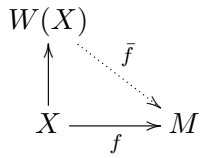
□

Ejemplo 1.3. Si $X = \emptyset$, entonces $W(X) = \{1\}$ es el monoide trivial.

Definición 1.8. Un *morfismo* $f : (M_1, *_1) \to (M_2, *_2)$ de monoides es una función que cumple $f(a *_1 b) = f(a) *_2 f(b)$ para todo $a, b \in M_1$ y $f(1_{M_1}) = 1_{M_2}$. Si además f es biyectiva, f es un *isomorfismo* de monoides.

Ejemplo 1.4. Si $X = \{x\}$, las palabras en $W(X)$ son cadenas finitas de $x's$. Si la palabra de n letras se denota por x^n, para $n \in \mathbb{N}$ existe $w = x^n \in W(X)$ tal que $|w| = n$. Además, es claro que $|x^n \cdot x^m| = |x^{n+m}| = n + m = |x^n| + |x^m|$, y así, la función longitud $|\cdot| : (W(X), \cdot) \to (\mathbb{N}, +)$ es un isomorfismo de monoides.

Proposición 1.4. *Sean X un conjunto, M un monoide y $f : X \to M$ una función arbitraria. Entonces existe un único morfismo de monoides $\bar{f} : W(X) \to M$ tal que $\bar{f}(x) = f(x)$ para todo $x \in X$.*



Demostración. Definimos $\bar{f} : W(X) \to M$ de la siguiente manera.

$$\bar{f}(w) = \begin{cases} 1 & \text{si} \quad w = 1 \\ f(x_1) \cdots f(x_n) & \text{si} \quad w = x_1 \cdots x_n \end{cases}$$

La función $\bar{f}$ es un morfismo de monoides. En efecto, si $w_1 = x_1 \cdots x_n$ y $w_2 = y_1 \cdots y_m$

$$\begin{aligned}\bar{f}(w_1 \cdot w_2) &= \bar{f}(x_1 \cdots x_n \cdot y_1 \cdots y_m) \\ &= f(x_1) \cdots f(x_n) \cdot f(y_1) \cdots f(y_m) \\ &= \bar{f}(w_1) \cdot \bar{f}(w_2).\end{aligned}$$

Para la unicidad note que si $\hat{f} : W(X) \to M$ es otro morfismo con $\hat{f}(x) = f(x)$ para toda $x \in X$. Entonces para $w = x_1 \cdots x_n \in W(X)$ se tiene

$$\begin{aligned}\hat{f}(x_1 \cdots x_n) &= \hat{f}(x_1) \cdot \ldots \cdot \hat{f}(x_n) \\ &= f(x_1) \cdot \ldots \cdot f(x_n) \\ &= \bar{f}(x_1) \cdot \ldots \cdot \bar{f}(x_n) \\ &= \bar{f}(x_1 \cdots x_n)\end{aligned}$$

Así $\hat{f}(w) = \bar{f}(w)$ para toda $w \in W(X)$ y $\bar{f}$ es único. □

Definición 1.9. Sean $u, v \in W(X)$, se dice que u es un *prefijo (sufijo)* de v si existe $\lambda \in W(X)$ tal que $v = u\lambda$ ($v = \lambda u$, respectivamente).

Proposición 1.5. *Sean $\alpha, \beta, \gamma, \delta \in W(X)$ tales que $\alpha\beta = \gamma\delta$.*

i) Si $\alpha = \gamma$ entonces $\beta = \delta$.

ii) Si $|\alpha| = |\gamma|$ entonces $\alpha = \gamma$.

iii) Si $|\alpha| \leq |\gamma|$, entonces α es un prefijo de γ y δ es un sufijo de β.

Demostración. Sean $\alpha = a_1 \cdots a_n, \beta = b_1 \cdots b_m, \gamma = c_1 \cdots c_r$ y $\delta = d_1 \cdots d_s$.

$$a_1 \cdots a_n b_1 \cdots b_m = \alpha\beta = \gamma\delta = c_1 \cdots c_r d_1 \cdots d_s$$

Entonces $n + m = r + s$.

i) Si $\beta \neq \delta$, entonces tienen diferente longitud o bien tienen al menos una letra distinta en su expresión.

En el primer caso, $|\beta| \neq |\delta|$, la hipótesis $\alpha = \gamma$ implica $n = r$; por lo tanto $m + n \neq r + s$ y así $\alpha\beta \neq \gamma\delta$, una contradicción.

Si existe j con $b_j \neq d_j$, entonces tanto β_j y δ_j aparecen en $\alpha\beta$ y $\gamma\delta$ respectivamente. Nuevamente, la hipótesis $|\alpha| = |\gamma|$ implica $n = r$ y por lo tanto las palabras $\alpha\beta$ y $\gamma\delta$ difieren en la letra $m + j$, lo cual es una contradicción.

En ambos casos tenemos una contradicción y por lo tanto $\beta = \delta$.

ii) Supóngase, por contradicción, que $\alpha \neq \gamma$. La hipótesis $|\alpha| = |\gamma|$, implica necesariamente que existe i con $a_i \neq c_i$. Por lo tanto las palabras $\alpha\beta$ y $\gamma\delta$ difieren en el lugar i, una contradicción.

iii) Si $n = |\alpha| < |\gamma| = r$. De la igualdad $\alpha\beta = \gamma\delta$ se tiene que $a_i = c_i$ para toda $1 \leq i \leq n$ y así existen $n+1, \ldots, r$ índices tal que $b_1 = c_{n+1}, \ldots, b_j = c_r$ para algún $1 \leq j \leq r$. Sea $\lambda = b_1 \cdots b_j$, entonces

$$\gamma = c_1 \cdots c_r = a_1 \cdots a_n b_1 \cdots b_j = \alpha\lambda$$

y así α es prefijo de γ. De igual manera δ es sufijo de β.

□

1.2 Grupos y subgrupos

Definición 1.10. Un *grupo* es un conjunto no vacío G junto con una operación binaria $\bullet : G \times G \to G$ que satisface las siguientes condiciones:

(1) La operación es *asociativa*, es decir,

$$(x \bullet y) \bullet z = x \bullet (y \bullet z) \quad \text{para todo } x, y, z \in G.$$

(2) Existe un *elemento neutro* $e \in G$ que cumple

$$x \bullet e = x = e \bullet x \quad \text{para cualquier } x \in G.$$

(3) Para todo $x \in G$ existe $y \in G$ tal que

$$x \bullet y = e = y \bullet x$$

Al elemento y se le llama *inverso* de x, y es denotado por x^{-1}.

Definición 1.11. Si la operación es conmutativa, $x \bullet y = y \bullet x$ para todo $x, y \in G$, el grupo G es *abeliano*.

Al grupo se le denota por $(G, \bullet)$ para enfatizar la importancia de la operación binaria, si no es así, se le denota simplemente como G.

Ejemplo 1.5. El conjunto $\mathbb{Z}$ de los números enteros es un grupo con la operación suma, donde el cero es el neutro aditivo. Este grupo, denotado $(\mathbb{Z}, +)$, es abeliano.

Ejemplo 1.6. Dado X un conjunto no vacío, el conjunto

$$S_X := \{f : X \to X \mid f \text{ es biyectiva}\}$$

y la composición de funciones forman un grupo, donde la función identidad 1_X es el neutro. Este grupo $(S_X, \circ)$, llamado *grupo de permutaciones*, no necesariamente es abeliano, solamente lo es si $|X| \leq 1$.

Los elementos de S_X se llaman *permutaciones*. Si X es un conjunto finito de n elementos, es usual denotarlo por S_n, y es conocido como *grupo simétrico*.

Ejemplo 1.7. El conjunto de matrices invertibles de $n \times n$ con entradas en $\mathbb{R}$ y el producto usual de matrices forman un grupo, el cual se conoce como *grupo lineal general*, denotado por $GL(n, \mathbb{R})$ o $GL_n(\mathbb{R})$. Si $n \geq 2$, este grupo no es abeliano.

Ejemplo 1.8. Los enteros módulo n con la suma de clases residuales forman un grupo, denotado por $(\mathbb{Z}_n, +_n)$. Además, este grupo es abeliano y finito.

Si G es un grupo donde la operación tiene notación multiplicativa, se define $G = \langle x \rangle = \{x^n : n \in \mathbb{Z}\}$ como las potencias de x. Si la operación de G está denotada de manera aditiva, entonces $G = \langle x \rangle = \{nx : n\mathbb{Z}\}$ son los múltiplos de x.

Definición 1.12. Un grupo G es cíclico si existe $x \in G$ tal que $G = \langle x \rangle$.

Los grupos (1.5) y (1.8) son cíclicos, el primero es infinito y el segundo finito.

Por comodidad el producto $x \bullet y$ en el grupo G se denotará como xy.

Definición 1.13. Un subconjunto no vacío H de un grupo G es un *subgrupo* de G, denotado por $H \leq G$, si satisface

(i) $x, y \in H$ entonces $xy \in H$.

(ii) $x \in H$ entonces $x^{-1} \in H$.

Observación 1.1. Las dos condiciones anteriores se reducen a lo siguiente: $H \leq G$ si y sólo si para todo $x, y \in H$ se tiene $xy^{-1} \in H$.

Observación 1.2. Si $H \leq G$ y $x \in H$, necesariamente $x^{-1} \in H$, por lo tanto $e = xx^{-1} \in H$. Así, $H \leq G$ si y sólo si H es un grupo con la operación de G restringida a H.

Ejemplo 1.9. Sea $G = GL(n, \mathbb{R})$ el grupo lineal general. Dado que una matriz invertible tiene determinante distinto de cero, el subconjunto

$$SL(n, \mathbb{R}) := \{A \in GL(n, \mathbb{R}) \mid \det(A) = 1\}$$

es un subgrupo de G. Este grupo es llamado el *grupo lineal especial*.

La intersección de subgrupos es un subgrupo. De lo anterior se define el subgrupo generado por un subconjunto.

Definición 1.14. Sea G un grupo y $S \subseteq G$, el *subgrupo generado por* S como la intersección de todos los subgrupos de G que contienen a S, es decir,

$$\langle S \rangle := \bigcap_{S \subseteq H \leq G} H.$$

Proposición 1.6. *Sea G un grupo y $S \subseteq G$ un subconjunto no vacío. Entonces el subconjunto*

$$L := \{ s_1^{i_1} \cdots s_n^{i_n} \mid s_j \in S; i_j = \pm 1; j = 1, \ldots, n; n \in \mathbb{N}\}$$

es un subgrupo de G. Además, este subgrupo es el menor que contiene a S, es decir, si $H \leq G$ y $S \subseteq H$ entonces $L \subseteq H$. En otras palabras, se tiene

$$L = \langle S \rangle = \bigcap_{S \subseteq H \leq G} H$$

Demostración. Es suficiente mostrar que dados $x, y \in L \Longrightarrow xy^{-1} \in L$. Para mostrar lo anterior, basta observar que dados $x = s_1^{i_1} \cdots s_k^{i_k}$ y $y = \hat{s}_1^{j_1} \cdots \hat{s}_m^{j_m}$ en L, entonces $y^{-1} = \hat{s}_k^{-i_k} \cdots \hat{s}_1^{-i_1}$ con $i_j = \pm 1$, por lo tanto $y^{-1} \in L$, obteniendo lo deseado:

$$xy^{-1} = s_1^{i_1} \cdots s_k^{i_k} \cdot \hat{s}_m^{-i_m} \cdots \hat{s}_1^{-i_1} \in L$$

Para mostrar lo restante, note que

$$\langle S \rangle := \bigcap_{S \subseteq H \leq G} H \subseteq L,$$

por ser L uno de los elementos sobre los cuales se toma la intersección. La otra inclusión se obtiene del hecho de que los elementos de L son productos de elementos de S y $S \subseteq H$, y por lo tanto

$$L \subseteq \bigcap_{S \subseteq H \leq G} H := \langle S \rangle.$$

□

Definición 1.15. Un grupo G es *finitamente generado*, abreviado *f.g.*, si existe un subconjunto finito $S \subseteq G$ tal que $G = \langle S \rangle$.

Ejemplo 1.10 (Grupo Diédrico)**.** Sea $n \geq 3$, el n-ciclo $c = (\, 1\, 2 \cdots n\,) \in S_n$ y σ la siguiente permutación

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & \cdots & i & \cdots & n-1 & n \\ 1 & n & n-1 & n-2 & \cdots & n+2-i & \cdots & 3 & 2 \end{pmatrix}$$

El subgrupo $D_n := \langle c, \sigma \rangle \leq S_n$ se le conoce como *grupo diédrico de orden n*. Más adelante, se presentarán otras formas de visualizar a D_n, entre ellas, como el grupo de simetrías de un polígono de n lados.

Definición 1.16. Un elemento x de un grupo G tiene orden n si $x^n = e$ y n es el mínimo entero positivo que cumple tal propiedad.

Proposición 1.7. *Sea $n \geq 3$ y $G = \langle x, y \rangle$ tal que $x, y \in G$ con $o(x) = n, o(y) = 2$ y $yxy = x^{-1}$, entonces*

$$G = \left\{1, x, x^2, \ldots, x^{n-1}, y, xy, x^2y, x^3y, \ldots, x^{n-1}y\right\}$$

es un grupo con $2n$ elementos.

Demostración. Si del lado derecho hay dos elementos iguales

$$x^k y^j = x^r y^s$$

entonces $x^{k-r} = y^{s-j}$. A partir de la segunda ecuación, $o(y) = 2$, se deduce que $x^{k-r} = y^0 = 1$ ó $x^{k-r} = y^1 = y$. Suponiendo cierto el último caso, $x^{k-r} = y$, las siguientes igualdades también son ciertas

$$\begin{aligned} x^{k-r+1} &= x(x^{k-r}) \\ &= xy \quad \text{hipótesis} \\ &= yx^{-1} \quad \text{dado que } yxy = x^{-1} \Rightarrow y^2xy = yx^{-1} \\ &= (x^{k-r})x^{-1} \quad \text{hipótesis} \\ &= x^{k-r-1} \end{aligned}$$

lo que implica $x = x^{-1}$, es decir $o(x) = 2$, una contradicción pues $n \geq 3$. De esta manera sólo el primer caso es posible: $x^{k-r} = 1$ y $y^{s-j} = 1$, es decir $n \,|\, k - r$ y $2 \,|\, s - j$. Por consiguiente $k \equiv r \pmod{n}, s \equiv j \pmod{2}$ y G contiene al menos los $2n$ elementos de

$$\left\{1, x, x^2, \ldots, x^{n-1}, y, xy, x^2y, x^3y, \ldots, x^{n-1}y\right\}$$

Recíprocamente, si $z \in G = \langle x, y \rangle$, entonces $z = x^{r_1}y^{s_1} \cdots x^{r_k}y^{s_k}$ donde $r_j = 0, \ldots, n-1$ y $s_j = 0, 1$. Dado que $yxy = x^{-1}$

$$z = \begin{cases} x^k y & \text{si } \sum s_j \text{ es impar} \\ x^k & \text{si } \sum s_j \text{ es par} \end{cases}$$

y se tiene $G = \left\{1, x, x^2, \ldots, x^{n-1}, y, xy, x^2y, x^3y, \ldots, x^{n-1}y\right\}$. □

Corolario 1.8. *El grupo diédrico D_n cumple:*

(i) $o(c) = n, o(\sigma) = 2$ *y* $\sigma c \sigma = c^{-1}$.

(ii) $|D_n| = 2n$.

Demostración. Para la primera afirmación observe la descomposición

$$\sigma = (2\ n)(3\ n-1)\cdots(i\ n-i+2)\cdots(k\ n-k+2)$$

donde

$$k = \begin{cases} \frac{n+1}{2} & \text{si } n \text{ es impar} \\ \frac{n}{2} & \text{si } n \text{ es par} \end{cases}$$

y por lo tanto

$$\sigma c \sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & \cdots & n-1 & n \\ n & n-1 & n-2 & n-3 & \cdots & 2 & 1 \end{pmatrix} = c^{-1}$$

La segunda afirmación es directa de la proposición. □

Definición 1.17. Sean $(G, *)$ y $(H, \circ)$ grupos. Una función $f : G \to H$ es un *morfismo de grupos* si para todos $a, b \in G$

$$f(a * b) = f(a) \circ f(b).$$

Si además f es una biyección, se dice que f es un *isomorfismo* y en tal caso se escribe $G \cong H$.

Ejemplo 1.11. Sea G un grupo. La función $\gamma_a : G \to G$ definida por

$$\gamma_a(x) := axa^{-1}$$

es un morfismo, llamado el *morfismo conjugación*.

Definición 1.18. Un subgrupo $K \leq G$ es un *subgrupo normal*, denotado por $K \lhd G$, si $gKg^{-1} = K$ para todo $g \in G$.

Si $K \leq G$ y existe las inclusiones $gKg^{-1} \leq K$ para todo $g \in G$, entonces $K \lhd G$: reemplazando g por g^{-1}, tenemos la inclusión $g^{-1}Kg \leq K$, y esto da la otra inclusión $K \leq gKg^{-1}$.

El núcleo K de un morfismo $f : G \to H$ es un subgrupo normal: si $a \in K$, entonces $f(a) = e_H$; si $g \in G$, entonces

$$f(gag^{-1}) = f(g)f(a)f(g)^{-1} = f(g)f(g)^{-1} = e_H,$$

y así $gag^{-1} \in K$. Así $gKg^{-1} \leq K$ para todo $g \in G$, y por lo tanto $K \lhd G$. Además, todo subgrupo normal K es el núcleo del epimorfismo canónico $\rho : G \to G/K$.

Es claro que la intersección de subgrupos normales es también un subgrupo normal.

Lema 1.9. *La intersección de cualquier familia de subgrupos normales de G es un subgrupo normal de G.*

Definición 1.19. Sea G un grupo y $X \subset G$ un subconjunto. El *subgrupo normal genrado por X* es la intersección de todos los subgrupos normales que contiene a X,

$$\langle X \rangle^G := \bigcap \{N \leq G \mid N \lhd G \text{ y } X \subset N\}$$

A este subgrupo también se le llama la *clausura normal de X*.

Lema 1.10. *Sea G un grupo y $X \subset G$. Si $X = \emptyset$, entonces $\langle X \rangle^G = \{1\}$. Si $X \neq \emptyset$, entonces su clausura normal $\langle X \rangle^G$ es el conjunto de todas las palabras sobre los conjugados de los elementos de X, es decir:*

$$\langle X \rangle^G = \left\{ \lambda_1^{e_1} \cdots \lambda_n^{e_n} \,|\, e_i = \pm 1, \lambda_i = g_i x g_i^{-1} \text{ para alguna } x \in X, n \in \mathbb{N} \right\}$$

Demostración. Si $X = \emptyset$ es claro que su clausura es el subgrupo normal trivial $\{1\}$. Si $X \neq \emptyset$, entonces se demostrará que

$$\langle X \rangle^G = \left\{ \lambda_1^{e_1} \cdots \lambda_n^{e_n} \,|\, e_i = \pm 1, \lambda_i = g_i x g_i^{-1} \text{ para alguna } x \in X, n \in \mathbb{N} \right\}$$

Denotemos por S al cunjunto del lado derecho de la igualdad. Es claro que si $x, y \in S$ el producto $xy \in S$. Además el inverso de x es otro elemento en S y por lo tanto $e = xx^{-1} \in S$, así S es un subgrupo de G.

Si $g_i = e \in G$ y $\lambda_i = x_i$, entonces $g_i x_i g_i^{-1} = x_i$ y $X \subset S$. Además, si $\lambda_i = g_i x_i g_i^{-1}$ y $\lambda_j = g_j x_j g_j^{-1}$ están en S, el conjugado del producto $\lambda_i \lambda_j$ es

$$\begin{aligned} g\lambda_i\lambda_j g^{-1} &= g(g_i x_i g_i^{-1}) e (g_j x_j g_j^{-1}) g \\ &= g g_i x_i g_i^{-1} (g^{-1} g) g_j x_j g_j^{-1} g \\ &= (g g_i) x_i (g g_i)^{-1} (g g_j) x_j (g g_j)^{-1} \in S \end{aligned}$$

Por lo tanto S contiene a todos los conjugados de sus elementos y entonces $S \lhd G$. En conclusión S es un subgrupo normal de G que contiene a X, así

$$\langle X \rangle^G = \bigcap_{S \subseteq H \lhd G} H \subseteq S$$

Por otro lado, si $g_i \in G$ y $x_i \in X$, entonces $g_i x_i g_i^{-1}$ está en $\langle X \rangle^G$ pues éste último es un subgrupo normal de G. Y por definición de grupo se tiene que

$$(g_1 x_1 g_1^{-1})(g_2 x_2 g_2^{-1}) \cdots (g_n x_n g_n^{-1}) \in \langle X \rangle^G$$

□

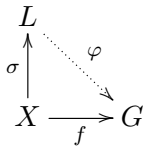
Lema 1.11. *Sea G un grupo y $X \subset G$ un subconjunto no vacío. Entonces la clausura normal de X está dada por*

$$\langle R \rangle^G = \langle\, g x g^{-1} \,|\, g \in G \text{ y } x \in X \,\rangle.$$

Demostración. Sea $N = \langle\, g x g^{-1} \,|\, g \in G \text{ y } x \in X \,\rangle$. Es claro que N es un subgrupo de G y que contiene a X. Además, si $g_1 x g_1^{-1} \in N$, también $g g_1 x g_1^{-1} g^{-1} = (g g_1) x (g g_1)^{-1} \in N$ para todo $g \in G$, por lo tanto N es un subgrupo normal de G. Finalmente, todo subgrupo normal de G que contenga a X, debe contener a N. Por lo tanto $N = \langle X \rangle^G$. □

1.3 Grupos libres

Definición 1.20. (Grupo Libre) Sea L un grupo, X un conjunto no vacío y $\sigma : X \to L$ una función. L es *libre* con base X si para toda función arbitraria $f : X \to G$ con G grupo existe un único morfismo $\varphi : L \to G$ tal que $f = \varphi \circ \sigma$.



Cuando sucede lo anterior, $f = \varphi \circ \sigma$, se dice que el diagrama es *conmutativo*.

Proposición 1.12. *Bajo las condiciones de la definición anterior, la función $\sigma : X \to L$ es inyectiva.*

Demostración. Supóngase que $\sigma(x_1) = \sigma(x_2)$ para $x_1 \neq x_2$. Sea G un grupo con al menos dos elementos distintos g_1 y g_2 y sea $f : X \to G$ tal que $f(x_1) = g_1$ y $f(x_2) = g_2$. Entonces $\varphi(\sigma(x_1)) = \varphi(\sigma(x_2))$, por consiguiente $f(x_1) = f(x_2)$ y así $g_1 = g_2$, lo cual es una contradicción. □

L es libre sobre el subconjunto $\mathrm{Im}(\sigma)$ y la función inclusión $\mathrm{Im}(\sigma) \to L$ toma el lugar de σ. Así, un grupo libre siempre es libre sobre un subconjunto y en este caso la conmutatividad del diagrama asegura que la restricción de φ a X coincide con f, por lo tanto φ es la única extensión de f a L.

1.3.1 Existencia

A continuación se mostrará la existencia de grupos libres, para ello se hará la siguiente construcción.

Dado X un conjunto, considere el conjunto de símbolos

$$X^{\pm} = \left\{x^{+1} : x \in X\right\} \cup \left\{x^{-1} : x \in X\right\}$$

La expresión x^{-1} sólo es un símbolo. También considere $W(X^{\pm})$, el conjunto de todas las palabras de $X^{\pm}$. Una palabra $w \in W(X^{\pm})$ es una sucesión finita de símbolos de $W(X^{\pm})$ de la forma

$$w = x_1^{\varepsilon_1} \cdots x_r^{\varepsilon_r}$$

con $x_i \in X, \varepsilon_i = \pm 1$ y $r \geq 0$.

Definición 1.21. La palabra $w = x_1^{\varepsilon_1} \cdots x_r^{\varepsilon_r}$ es *positiva* si $w = 1$ o bien $\varepsilon_i = +1$ para toda i.

Una palabra es positiva, si es la palabra vacía o si todos sus exponentes son positivos. Se define ahora una subpalabra como una subsucesión de la sucesión original.

Definición 1.22. Una *subpalabra* de la palabra $w = x_1^{\varepsilon_1} \cdots x_r^{\varepsilon_r}$ es la palabra vacía o una palabra de la forma $v = x_j^{\varepsilon_j} \cdots x_k^{\varepsilon_k}$ con $1 \leq j \leq k \leq r$.

Esto es, v es una subpalabra de w si existen palabras (posiblemente vacías) w' y w'' tal que $w = w'vw''$.

Por el Teorema (1.3) se sabe que $(W(X^{\pm}), \cdot)$ es un monoide. Para que $W(X^{\pm})$ sea un grupo con la concatenación se necesita primero definir el inverso de cualquier palabra.

Definición 1.23. Sea $w \in W(X^{\pm})$. Se define la palabra inversa de 1 como palabra 1 y si $w = x_1^{\varepsilon_1} \cdots x_r^{\varepsilon_r}$ su *palabra inversa* es $w^{-1} = x_r^{-\varepsilon_r} \cdots x_1^{-\varepsilon_1} \in W(X^{\pm})$.

En la siguiente proposición sólo se considera $W(X^{\pm})$ como monoide.

Lema 1.13. *En el monoide $W(X^{\pm})$ se cumple $(w^{-1})^{-1} = w$.*

Demostración. Primero observe que tanto w y su palabra inversa w^{-1} tienen la misma longitud por definición. La afirmación es directa de la definición observando que la palabra inversa invierte el orden de las letras de la palabra original. □

Observación 1.3. Si $W(X^{\pm})$ fuera un grupo con la concatenación se cumpliría $w \cdot w^{-1} = 1$ para toda w, en particular si $w = x$, sin embargo, la igualdad $x \cdot x^{-1} = 1$ no se cumple: la longitud de $x \cdot x^{-1}$ es dos, mientras que la longitud de 1 es cero. Para resolver este problema se define una relación de equivalencia sobre las palabras.

1.3.2 Palabras reducidas

Definición 1.24. Una palabra w es *reducida* si $w = 1$ o $w = x_1 \cdots x_n$ con $x_i \in X \cup X^{-1}$ y $x_{i+1} \neq x_i^{-1}$ para toda $1 \leq i < n$.

Las palabras reducidas son aquellas que resultan después de eliminar todos los pares de la forma xx^{-1}.

Definición 1.25. Una *transformación elemental* de una palabra consiste de añadir o eliminar un par de elementos de la forma xx^{-1}.

Definición 1.26. Sean $w_1, w_2 \in W(X)$. Decimos que $w_1 \sim w_2$ si y sólo si w_2 se obtiene a partir de w_1 por una sucesión transformaciones elementales.

Definición 1.27. Sea X un conjunto. Una relación $\sim$ sobre los elementos de X es de *equivalencia* si:

i) Es reflexiva: para todo $x \in X$ se cumple $x \sim x$.

ii) Es simétrica: si $x \sim$ entonces $y \sim x$, para todo $x, y \in X$.

iii) Es transitiva: si $x \sim y$ y $y \sim z$, entonces $x \sim z$, para todo $x, y, z \in X$.

Lema 1.14. *La relación $\sim$ definida en 1.14 es de equivalencia en $W(X)$.*

Demostración. Supongamos que $w_1, w_2, w_3 \in W(X)$. Es claro que $w_1 \sim w_1$, dado que podemos añadir un par xx^{-1} y luego removerla. Si $w_1 \sim w_2$ entonces como las transformaciones elementales son reversibles, podemos revertir los pasos que se llevaron a cabo y así obtener w_1 a partir de w_2, por lo tanto $w_2 \sim w_1$. Finalmente, si suponemos $w_1 \sim w_2$ y $w_2 \sim w_3$, es claro que $w_1 \sim w_3$. Por lo tanto $\sim$ es una relación de equivalencia en $W(X)$. $\square$

Las palabras son una sucesión finita de símbolos, por lo tanto debe existir un número finito de transformaciones elementales entre palabras en la misma clase de equivalencia.

Teorema 1.15. *Cada clase de equivalencia de $\sim$ contiene exactamente una palabra reducida.*

Demostración. Si $X = \emptyset$, entonces existe una única palabra en X, la palabra vacía 1, y es reducida. Si $X \neq \emptyset$, se demostrará primero que existe una palabra reducida equivalente a w. Si w no tiene subpalabra de la forma xx^{-1} donde $x \in X \cup X^{-1}$, entonces w es reducida. En otro caso, borramos el primer par xx^{-1} y obtenemos una nueva palabra w_1, que puede ser vacía, con $|w_1| < |w|$. Y repetimos: si w_1 es reducida, nos detenemos; si existe una subpalabra de la forma xx^{-1}, la borramos y obtenemos una palabra más corta w_2. Como las longitudes van decreciendo, este proceso termina con una palabra reducida equivalente a w.

Para la unicidad. Supongamos que u y v son dos palabras reducidas en la misma clase de equivalencia. Sea $w_1 \cdots w_n$ la sucesión mínima de palabras tal que w_{i+1} es una transformación elemental de w_i, donde w_1 es una transformación de u y v es una transformación elemental de w_n.

Como u es una palabra reducida y $u \neq v$, entonces $|w_1| > |u|$. Análogamente dado que v es una palabra reducida podemos concluir que $|w_n| > |v|$. Entonces la longitud de las palabras unidas u y v primero debe crecer y después en algún momento decrecer. Sea m el valor mínimo tal que $|w_m| > |w_{m-1}|$. Para llegar a w_m debemos añadir algún par aa^{-1} a w_{m-1}. Y después removemos algún bb^{-1} para llegar a w_{m+1}. Tenemos tres opciones de como se relaciona aa^{-1} con bb^{-1}. Y en cada caso se argumentará que $w_1 \cdots w_n$ es la sucesión más corta posible de w_i's.

Supongamos $aa^{-1} = bb^{-1}$, en este caso tenemos $w_{m-1} = w_{m+1}$. Si aa^{-1} y bb^{-1} se traslapan parcialmente, entonces existe una subpalabra de la forma $aa^{-1}a$. Esto es, para obtener w_m debemos reemplazar algún a en w_{m-1} por $aa^{-1}a$ y para obtener w_{m+1} debemos reemplazar el mismo $aa^{-1}a$ con a. Esto es $w_{m-1} = w_{m+1}$. Por último supongamos que aa^{-1} y bb^{-1} son disjuntos. En este caso podemos cortar la palabra entera simplemente removiendo

w_{m-1}, w_m y w_{m+1}. Repitiendo este proceso de cortar palabras, borraremos todas las w_i's y concluimos que $u = v$. □

Sea $[x]$ la clase de equivalencia de w. Nótese que en la clase de equivalencia de w siempre se puede elegir un representante de la clase que sea una palabra reducida.

Podemos ahora probar, respecto a lo anterior, que el conjunto de las clases de equivalencia $\sim$ forman un grupo libre con base X. El neutro es la palabra vacía y la operación es la concatenación.

Teorema 1.16. *Sea X un conjunto no vacío. Existe un grupo L y una función $\sigma : X \to L$ tal que L es libre con base X y $L = \langle Im(\sigma) \rangle$.*

Demostración. Sea $L = W(X)/\sim$ el conjunto de todas las clases de equivalencia

$$L := \{\, [w] \,|\, w \in W(X)\}$$

Afirmación 1.1. L es un grupo.

Sean $v_1, v_2, w_1, w_2 \in W(X)$ tal que $v_1 \sim v_2$ y $w_1 \sim w_2$. Por Teorema (1.15) existen palabras reducidas $v, w \in W(X)$ tales que $v_1 \sim v \sim v_2$ y $w_1 \sim w \sim w_2$, entonces

$$v_1 w_1 \sim vw \sim v_2 w_2$$

y por lo tanto $v_1 w_1 \sim v_2 w_2$.

Se define el producto $[w] \cdot [u] := [wu]$, donde wu es la concatenación de palabras. Por lo anterior se tiene que este producto esta bien definido y $[w] \cdot [1] = [w1] = [w] = [1w] = [1] \cdot [w]$ y $[w] \cdot [w^{-1}] = [ww^{-1}] = [1] = [w^{-1}w] = [w^{-1}] \cdot [w]$. Además, este producto es asociativo dado que para cualesquiera $u, v, w \in W(X)$ se tiene que $(wv)u = w(vu) \in W(X)$ y así

$$([w] \cdot [v]) \cdot [u] = [(wv)u] = [w(vu)] = [w] \cdot ([v] \cdot [u])$$

Por lo tanto $(L, \cdot)$ es un grupo: el neutro es la clase de la palabra vacía $[1]$ y el inverso de $[w]$ es la clase de la palabra inversa $[w^{-1}]$, es decir, $[w]^{-1} = [w^{-1}]$.

Afirmación 1.2. L es libre con base X.

En efecto, sean $\sigma : W(X) \to L$ el epimorfismo canónico de monoides, $\sigma(w) = [w]$, G un grupo y $f : X \to G$ una función arbitraria.

Se define $\bar{f} : W(X) \to G$ como

$$\bar{f}(x_1^{\varepsilon_1} \cdots x_n^{\varepsilon_n}) = g_1^{\varepsilon_1} \cdots g_n^{\varepsilon_n}$$

donde $g_i = f(x_i)$. Sean $v, w \in W(X)$ tal que $w \sim v$, entonces v y w difieren en su expresión escrita por un producto finito de subpalabras xx^{-1} con $x \in X$; para estas expresiones es claro que $\bar{f}(xx^{-1}) = \bar{f}(x)\bar{f}(x)^{-1} = 1_G$ y

$$\bar{f}(v) = \bar{f}(w).$$

Por lo tanto $\bar{f}$ está bien definida.

Si $x = x_1^{\varepsilon_1} \cdots x_n^{\varepsilon_n}$ y $y = y_1^{\eta_1} \cdots x_m^{\eta_m}$ se tiene

$$\begin{aligned}\bar{f}(x \cdot y) &= \bar{f}(x_1^{\varepsilon_1} \cdots x_n^{\varepsilon_n} \cdot y_1^{\eta_1} \cdots x_m^{\eta_m}) \\ &= g_1^{\varepsilon_1} \cdots g_n^{\varepsilon_n} \cdot h_1^{\eta_1} \cdots h_m^{\eta_m} \\ &= \bar{f}(x)\bar{f}(y)\end{aligned}$$

con $h_i = f(y_i)$, por lo que se concluye que $\bar{f}$ es un morfismo de monoides.

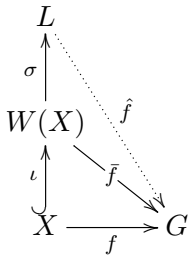
Sea $\hat{f} : L \to G$ dada por $\hat{f}([w]) = \bar{f}(w)$. Por definición de $\bar{f}$ se tiene

$$\begin{aligned}\hat{f}([w] \cdot [v]) &= \hat{f}([wv]) \quad \text{definición de producto en } L \\ &= \bar{f}(wv) \quad \text{definición de } \hat{f} \\ &= \bar{f}(w)\bar{f}(v) \quad \text{porque } \bar{f} \text{ es morfismo} \\ &= \hat{f}([w])\hat{f}([v]) \quad \text{definición de } \hat{f}\end{aligned}$$

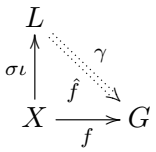
Por lo tanto $\hat{f}$ es morfismo de grupos. Además, por definición de $\sigma, \hat{f}$ y $\bar{f}$, respectivamente, se tienen las igualdades

$$\hat{f}\sigma(x) = \hat{f}([x]) = \bar{f}(x) = f(x)$$

para todo $x \in X$, es decir, el triángulo mayor del siguiente diagrama es conmutativo:



Finalmente, si existe $\gamma : L \to G$ otro morfismo de grupos tal que $\gamma\sigma = f$.



entonces $\gamma\sigma = \hat{f}\sigma$ y tanto γ y $\hat{f}$ coinciden en $\mathrm{Im}(\sigma)$. Pero $L = \langle \mathrm{Im}(\sigma) \rangle$, así tiene que $\gamma = \hat{f}$. □

1.3.3 Algunas propiedades

Por la proposición anterior, todo elemento del grupo libre construido L, puede ser escrito de manera única en la forma $[w]$ con w una palabra

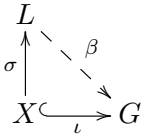
reducida, $w = x_1^{\varepsilon_1} \cdots x_r^{\varepsilon_r}$ donde $\varepsilon_i = \pm 1, r \geq 0$ y no aparecen $x^{\varepsilon} x^{-\varepsilon}$ en la palabra.

Por definición de multiplicación en L se tiene que $[w] = [x_1]^{\varepsilon_1} \cdots [x_r]^{\varepsilon_r}$. Multiplicando términos consecutivos que involucren el mismo término x_i, se deduce que, después de reetiquetar los $x_i's$, el elemento $[w]$ se puede escribir de la forma $[w] = [x_1]^{l_1} \cdots [x_s]^{l_s}$ con $s \geq 0$, l_i es un entero no cero y $x_i \neq x_{i+1}$. Esta es llamada la *forma normal* de w.

La existencia de una forma normal es característica de los grupos libres.

Teorema 1.17 (Forma Normal)**.** *Sea G un grupo y X un subconjunto de G. Suponga que todo elemento $g \in G$ se puede escribir de manera única de la forma $g = x_1^{l_1} \cdots x_s^{l_s}$ donde $x_i \in X, s \geq 0, l_i \neq 0$, entonces G es libre con base X.*

Demostración. Sea L un grupo libre con base X y $\sigma : X \to L$ la inyección asociada. Existe un único morfismo $\beta : L \to G$ tal que $\beta\sigma : X \to G$ es el función inclusión de X en G, es decir, $\beta\sigma = \iota$.



Como $G = \langle X \rangle$, el morfismo β es suprayectivo. Para la inyectividad, note que si $\beta(x) = \beta(y)$ para algunos $x, y \in L$, como $\beta(x), \beta(y) \in G = \langle X \rangle$, existen $x_1, \ldots, x_n, y_1, \ldots, y_m \in X$ y exponentes l_i, k_i tal que

$$x_1^{l_1} \cdots x_n^{l_n} = \beta(x) = \beta(y) = y_1^{k_1} \cdots y_m^{k_m}$$

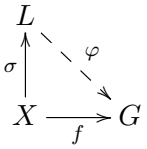
por la unicidad de la forma normal, se tiene que $n = m, l_i = k_i$ y $x_i = y_i$ para toda $1 \leq i \leq n$ y por consiguiente

$$x = \sigma(x_1)^{l_1} \cdots \sigma(x_n)^{l_n} = \sigma(y_1)^{k_1} \cdots \sigma(y_m)^{k_m} = y.$$

Por lo tanto β es inyectivo y β es un isomorfismo de grupos, $G \cong L$, y como L es un grupo libre con base X, G también lo es. □

Corolario 1.18. *Todo grupo es cociente de un grupo libre.*

Demostración. Sean G un grupo y $X := \{ x_g \mid g \in G \}$. La función $f : X \to G$ definida por $f(x_g) := g$ es una biyección. Si L es libre con base X, entonces existe un único morfismo $\varphi : L \to G$ que hace conmutar el siguiente diagrama:



Es decir, $\varphi\sigma = f$ y φ extiende a f. Además φ es suprayectiva. En efecto, sea $g \in G$, como f es suprayectiva existe $x_g \in X$ tal que $\varphi(\sigma(x_g)) = f(x_g) = g$ y así existe $z = \sigma(x_g) \in L$ tal que $\varphi(z) = g$. Por el primer teorema de isomorfismo se obtiene $G \cong L/\text{Nuc}\varphi$. □

Las siguientes propiedades de grupos libres se basan en las propiedades fundamentales de grupos abelianos libres que pueden ser consultadas en [7].

Definición 1.28. Sean G es un grupo y $a, b \in G$, el *conmutador* de a y b es el elemento $[a, b] := aba^{-1}b^{-1} \in G$ y el *subgrupo conmutador*

$$G' := \langle [a, b] | a, b \in G \rangle$$

es el subgrupo de G generado por todos los conmutadores.

Definición 1.29. Sea $x \in G$. Un *conjugado* para x es un elemento de la forma axa^{-1} para algún $a \in G$. Al conjugado de x por a se le denota mediante x^a.

Lema 1.19. *El subgrupo conmutador G' es un subgrupo normal de G. Además, si $H \lhd G$, entonces G/H es abeliano si y sólo si $G' \leq H$. En particular G/G' es abeliano.*

Demostración. Sea $f : G \to G$ un morfismo de grupos. Entonces

$$f([a, b]) = f(aba^{-1}b^{-1}) = f(a)f(b)f(a)^{-1}f(b)^{-1} = [f(a), f(b)] \in G'$$

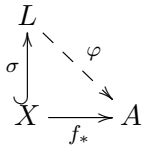
y por lo tanto $f(G') \leq G'$. Además el inverso del conmutador $[x, y]$ es $[y, x]$. Para ver la normalidad de G' recuerde que un subgrupo H es normal si y sólo si contiene todos sus conjugados. Dado que $[x, y] \in G'$ se demostrará que sus conjugados $[x, y]^a \in G'$:

$$[x, y]^a = a(xyx^{-1}y^{-1})a^{-1} = (axa^{-1})(aya^{-1})(ax^{-1}a^{-1})(ay^{-1}a^{-1}) = [x^a, y^a].$$

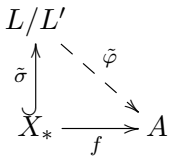
Sea $H \lhd G$. Si G/H es abeliano, entonces $HaHb = HbHa$ para todo $a, b \in G$; esto es $Hab = Hba$, y así $ab(ba)^{-1} = aba^{-1}b^{-1} = [a, b] \in H$ y por lo tanto $G' \leq H$. Inversamente, si $G' \leq H$. Para todo $a, b \in G, ab(ba)^{-1} = [a, b] \in G' \leq H$, y así $Hab = Hba$; esto es, G/H es abeliano. □

Lema 1.20. *Si L es un grupo libre con base X, entonces el cociente L/L' es un grupo abeliano libre con base $X_* = \{xL' : x \in X\}$.*

Demostración. Sea A es un grupo abeliano y sea $f : X_* \to A$ una función arbitraria. Sea $f_* : X \to A$ definida como $f_*(x) := f(xL')$. Como L es libre con base X, existe un único morfismo $\varphi : L \to A$ que extiende a f_*.



Paro todo $a, b \in L$ se tiene $[a, b] \in L'$ y así $L' \leq \text{Nuc}\varphi$. El grupo L/L' es abeliano por el Lema (1.19) y por el Teorema del Factor, existe un morfismo $\tilde{\varphi} : L/L' \to A$ definido por $\tilde{\varphi}(wL') = \varphi(w)$, que extiende a f.



El morfismo $\tilde{\varphi}$ esta bien definido, si $w_1L' = w_2L'$ entonces $w_1w_2^{-1} \in L' \leq \text{Nuc}\varphi$, así $\tilde{\varphi}(w_1) := \varphi(w_1) = \varphi(w_2) := \tilde{\varphi}(w_2)$.

La extensión $\tilde{\varphi}$ es única. Sea $\psi : L/L' \to A$ otro morfismo de grupos abelianos con $\psi(xL') = f(xL')$. Si $\nu : L \to L/L'$ es el epimorfismo natural, entonces $\psi\nu : L \to A$ es un morfismo con $\psi\nu(x) = \psi(xL') = f(xL') = \varphi(x)$ para todo $x \in X$, así $\psi\nu = \varphi = \tilde{\varphi}\nu$. Como ν es epimorfismo, ν es cancelable por la derecha y $\psi = \tilde{\varphi}$. Por lo tanto L/L' es libre con base X_*. □

Los grupos libres con bases de la misma cardinalidad son isomorfos.

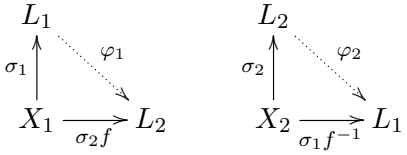
Proposición 1.21. *Sean L_1 y L_2 grupos libres con base X_1 y X_2 respectivamente. Entonces $|X_1| = |X_2|$ si y sólo si $L_1 \cong L_2$.*

Demostración. Si $\varphi : L_1 \to L_2$ es un isomorfismo, entonces $L_1/L_1' \cong L_2/L_2'$.

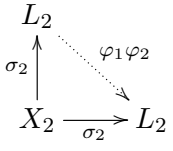
En efecto, por Lema (1.20), el grupo abeliano L_1/L_1' es libre con base $(X_1)_* := \{xL_1' : x \in X_1\}$. Además, como X_1 es base de L_1 se tiene $|(X_1)_*| = |X_1|$, y por lo tanto $|X_1| = |L_1/L_1'|$. De manera análoga se obtiene $|X_2| = |L_2/L_2'|$. Así existen dos grupos abelianos libres, L_1/L_1' y L_2/L_2', con bases equipotentes $|(X_1)_*| = |(X_2)_*|$, luego son isomorfos $L_1/L_1' \cong L_2/L_2'$ (ver Apéndice A). Es decir,

$$|X_1| = |L_1/L_1'| = |L_2/L_2'| = |X_2|.$$

Sea $\sigma_1 : X_1 \to L_1$ y $\sigma_2 : X_2 \to L_2$ las inyecciones dadas y $f : X_1 \to X_2$ una biyección. Existen φ_1 y φ_2 morfismos de grupos y diagramas conmutativos:



es decir $\varphi_1\sigma_1 = \sigma_2 f$ y $\varphi_2\sigma_2 = \sigma_1 f^{-1}$. Así $\varphi_1\varphi_2\sigma_2 = \varphi_1\sigma_1 f^{-1} = \sigma_2 f f^{-1} = \sigma_2$ y el siguiente diagrama



también conmuta. Pero el morfismo identidad 1_{L_2} en L_2 también hace conmutar el diagrama, así $\varphi_1\varphi_2 = 1_{L_2}$ por la unicidad. Con un argumento similar se tiene $\varphi_2\varphi_1 = 1_{L_1}$, así que φ_1 es un isomorfismo y $L_1 \cong L_2$. □

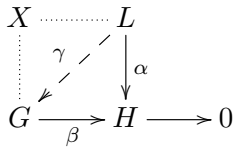
Definición 1.30. Sea L un grupo libre con base X. El *rango* de L es la cardinalidad de su base X.

La proposición anterior afirma que el rango de L no depende de la elección del conjunto X, su *base*. Formalmente se tiene el siguiente resultado.

Corolario 1.22. *Si L es libre con base X, entonces L es generado por X.*

Demostración. Es directo de la construcción del Teorema (1.16) tomando σ como el morfismo inclusión. □

Teorema 1.23 (**Propiedad proyectiva**). *Sea $\beta : G \to H$ un morfismo suprayectivo. Si L es libre y si $\alpha : L \to H$ es un morfismo, entonces existe $\gamma : L \to G$ que hace conmutar el siguiente diagrama, $\beta\gamma = \alpha$:*



Demostración. Sea L un grupo libre con base X. Si $x \in X \subset L$, entonces $\alpha(x) \in H = \text{Im}\beta$, así existe $g_x \in G$ tal que $\beta(g_x) = \alpha(x)$. Por la propiedad universal de grupos libres se puede extender la función $x \mapsto g_x$ a un único morfismo $\gamma : L \to G$, es decir $\gamma(x) = g_x$. Como $\beta\gamma(x) = \beta(g_x) = \alpha(x)$ para toda $x \in X$ y así, X genera a L, y se sigue que $\beta\gamma = \alpha$. □

1.4 Presentaciones de grupos

Todo grupo es cociente de un libre, el siguiente resultado es otra forma de escribir el Corolario (1.18).

Corolario 1.24. *Sean G un grupo, X un conjunto y $f : X \to G$ una función tal que $G = \langle Imf \rangle$. Entonces el morfismo $\bar{f} : L(X) \to G$ es suprayectivo y, en particular, $G \cong L(X)/Nuc\bar{f}$.*

Sea R el conjunto de generadores del subgrupo N del grupo libre L. Como el grupo L está totalmente determinado por X y el subgrupo normal N lo está por el conjunto R, el grupo $G \cong L/N$ puede definirse dando un conjunto cuyos elementos se llamaremos generadores de G y mediante un conjunto R cuyos elementos llamaremos relaciones que definen G.

Definición 1.31. Sean G un grupo, $X \subset G$ un subconjunto y R una familia de palabras en X. G tiene *generador* X y *relaciones* R si $G \cong L/N$, donde L es el grupo libre con base X y N es el subgrupo normal de L generado por R.

El subgrupo N existe, más aún, por el Lema (1.9), $N = \langle R \rangle^G$, es el menor subgrupo normal de G que contiene a R, la clausura normal de R.

Definición 1.32. El par $(X \mid R)$ es una *presentación* del grupo $G \cong L/N$.

Definición 1.33. Un grupo G es *finitamente presentado* (o finitamente relacionado) si tiene una presentación con un número finito de generadores y un conjunto finito de relaciones.

Pueden haber diferentes presentaciones de un mismo grupo. En tal caso, se llamarán *presentaciones isomorfas.* Los siguientes ejemplos son grupos finitamente presentados.

Ejemplo 1.12. El grupo $\mathbb{Z}_2$ tiene por presentación $(x \mid x^2)$.

En efecto, sea $G = \mathbb{Z}_2$, $X = \{x\}$ y $f : X \to \mathbb{Z}_2$ tal que $f(x) = \bar{1}$. Como $X \subset W(X) \subseteq L(X)$, por el ejemplo (1.4), $L(X)$ se puede ver de la forma

$$L(X) = \{ x^n \mid n \in \mathbb{Z} \}$$

Además la función $\hat{f} : L(X) \to n\mathbb{Z}$ dada por $\hat{f}(x) = x^n$, es un isomorfismo de grupos. Sea $\bar{f} : L(X) \to \mathbb{Z}_2$ definida como $\bar{f}(x^n) := \bar{n}$. En particular $\bar{f}$ es suprayectiva y

$$\mathrm{Nuc}\bar{f} = \{x^n |\ n \in 2\mathbb{Z}\} = \{x^{2n} | n \in \mathbb{Z}\} := \langle x^2 \rangle.$$

El primer teorema de isomorfismo afirma que $\mathbb{Z}_2 \cong L(X)/\mathrm{Nuc}\bar{f}$ es precisamente el isomorfismo $G \cong \mathbb{Z}/2\mathbb{Z}$ usual.

Lema 1.25. *Sea D_n es el grupo diédrico. Si $G = \langle x, y \rangle$, con $o(x) = n, o(y) = 2$ y $yxy = x^{-1}$, entonces $G \cong D_n$.*

Demostración. Sea $f : D_n \to G$ dada por $f(c^k\sigma^j) := x^k y^j$. El grupo diédrico tiene $2n$ elementos según el ejemplo (1.10) y la Proposición (1.7) asegura que G tiene $2n$ elementos, por lo tanto f es una biyección.

Sean $\alpha, \beta \in D_n = \langle c, \sigma \rangle$, entonces $\alpha = \prod(c^k\sigma^j)$ y $\beta = \prod(c^r\sigma^s)$ donde $0 \leq k, r < n$ y $j, s = 0, 1$ por la Proposición (1.7). Por lo tanto

$$\begin{aligned} f(\alpha \cdot \beta) &:= f(\prod(c^r\sigma^s) \cdot \prod(c^r\sigma^s)) \\ &= f(\prod(c^k\sigma^j) \cdot (c^r\sigma^s)) \\ &= \prod(x^k y^j) \cdot (x^r y^s) \\ &= \prod(x^k y^j) \cdot \prod(x^r y^s) \\ &= f(\alpha) f(\beta) \end{aligned}$$

Y así f es un morfismo de grupos. □

El grupo diédrico D_n tiene varias presentaciones, la siguiente es una de ellas, como el grupo de simetrías de un polígono de n lados.

Proposición 1.26. *El n-ésimo grupo diédrico D_n tiene como presentación* $(\rho, \sigma \,|\, \rho^n, \sigma^2, \rho\sigma\rho\sigma^{-1})$.

Demostración. Sea $n \geq 2$ y $\alpha = 2\pi/n$. Consideremos los siguientes dos elementos del grupo lineal general $GL_2(\mathbb{R})$:

$$\rho = \begin{pmatrix} \cos\alpha & \operatorname{sen}\alpha \\ -\operatorname{sen}\alpha & \cos\alpha \end{pmatrix} \quad \sigma = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Sea $D = \langle \rho, \sigma \rangle$. Es claro que $\rho^n = \sigma^2 = 1$. Por otro lado, un cálculo elemental muestra que $\rho\sigma = \sigma\rho^{-1}$. Esto implica

$$D = \{\sigma^i \rho^j : \ 0 \leq i < 2, \ 0 \leq j < n\}.$$

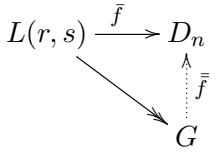
Más aún, calculando explícitamente los $2n$ elementos se puede ver que son todos distintos y así $|D| = 2n$. Por el Lema (1.25) se tiene $D \cong D_n$ y D es el n-ésimo grupo diédrico.

Sea ahora $X = \{r, s\}$ con $r \neq s$ y sea $f : X \to D_n$ tal que $f(r) = \rho$ y $f(s) = \sigma$. Por la propiedad universal de grupos libres, existe $\bar{f} : L(X) \to D_n$. Se busca describir el núcleo de este morfismo.

Se afirma que $\mathrm{Nuc}\bar{f} = \langle\langle r^n, s^2, rsrs^{-1} \rangle\rangle$ es el menor subgrupo normal de $L(r, s)$ que contiene a r^n, s^2 y $rsrs^{-1}$.

Primero, observe que $\bar{f}(r^n) = (\bar{f}(r))^n = \rho^n = 1$. De igual manera se tiene $\bar{f}(s) = \bar{f}(rsrs^{-1}) = 1$, así $\{\rho^n, s^2, rsrs^{-1}\} \subseteq \mathrm{Nuc}\bar{f}$ y es clara la inclusión $\langle\langle r^n, s^2, rsrs^{-1} \rangle\rangle \subset \mathrm{Nuc}\bar{f}$.

Para la otra contención, sea $G = L(r, s)/\langle\langle r^n, s^2, rsrs^{-1} \rangle\rangle$. La propiedad universal del cociente asegura la existencia de un morfismo $\bar{\bar{f}} : G \to D_n$ que hace conmutativo el siguiente diagrama:



Como $\bar{f}$ es suprayectiva, la conmutatividad implica que $\bar{\bar{f}}$ es suprayectiva. Si se muestra que G tiene a lo sumo $2n$ elementos, se concluirá que tiene exactamente $2n$, que $\bar{\bar{f}}$ es un isomorfismo y, en particular que

$$\mathrm{Nuc}\bar{f} = \langle\langle r^n, s^2, rsrs^{-1} \rangle\rangle$$

como se afirmó arriba.

Sean $\overline{r}, \overline{s} \in G$ las clases de r y s respectivamente. Es claro que $\overline{r}^n = \overline{r^n} = \overline{1}$. Además, también se cumple $\overline{s}^2 = \overline{1}$ y $\overline{r} \cdot \overline{s} = \overline{rs} = \overline{sr^{-1}} = \overline{s} \cdot \overline{r^{-1}} = \overline{s} \cdot \overline{r}^{-1}$.

Es evidente que $G = \langle \overline{r}, \overline{s} \rangle$. Por el Lema (1.25), se concluye que $G \cong D_n$ y esto implica $|G| = 2n$, como se buscaba demostrar. Finalmente, se tiene

$$D_n \cong L(r,s)/\langle\langle r^n, s^2, rsrs^{-1} \rangle\rangle.$$

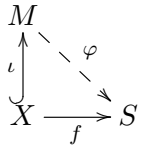
□

Ejemplo 1.13. ($x \mid \emptyset$) es una presentación del grupo libre $\mathbb{Z}$. Esto es, los enteros tienen por presentación un generador y ninguna relación.

1.5 Semigrupos y monoides libres

Algunos resultados presentados en el Capítulo 3 son para los semigrupos libres y en particular para el monoide y semigrupo $W(X)$, las palabras positivas del conjunto X. En esta sección se presentan algunos resultados básicos sobre semigrupos y monoides libres.

Definición 1.34 (Semigrupo libre)**.** Sea M un semigrupo y $X \subseteq M$ un subconjunto. M es un *semigrupo libre* con base X si para todo semigrupo S y toda función $f : X \to S$, existe un único morfismo de semigrupos $\varphi : M \to S$ que extiende a f, es decir, el siguiente diagrama conmuta:



Por la Proposición (1.3) el conjunto $W(X)$ es un monoide (y por lo tanto un semigrupo), pero no necesariamente es un grupo. En la construcción del semigrupo sólo se consideran las palabras positivas, es decir, la palabra vacía 1 o $w = x_1^{\varepsilon_1} \cdots x_n^{\varepsilon_n}$ donde todos los exponentes $\varepsilon_i = +1$.

Definición 1.35. Una *congruencia* en un semigrupo $(M, *)$ es una relación de equivalencia en M tal que $a \equiv c$ y $b \equiv d$ implica $a * b \equiv c * d$.

Definición 1.36 (Semigrupo cociente)**.** Sea $\equiv$ una congruencia en un semigrupo $(M, *)$. El *semigrupo cociente* es el conjunto de todas las clases de equivalencia

$$M/\equiv := \{\, [a] \mid a \in M \,\}$$

con la operación $[a] \cdot [b] := [a * b]$. Este producto esta bien definido porque $\equiv$ es una congruencia en el semigrupo M.

Análogamente al Corolario (1.18) se tiene que todo semigrupo es cociente de un semigrupo libre.

Ejemplo 1.14. Sea $\varphi : (M, *_1) \to (N, *_2)$ un morfismo de semigrupos. Entonces la relación $a \equiv b$ si y sólo si $\varphi(a) = \varphi(b)$ es una congruencia en M. En efecto, es claro que es una relación de equivalencia y si $a \equiv c$ y $b \equiv d$ entonces $\varphi(a) = \varphi(c)$ y $\varphi(b) = \varphi(d)$ y por lo tanto

$$\varphi(a *_1 b) = \varphi(a) *_2 \varphi(b) = \varphi(c) *_2 \varphi(d) = \varphi(c *_1 d),$$

así $a *_1 c \equiv b *_1 d$. Esta congruencia es llamada $Nuc\,\varphi$.

Teorema 1.27 (Primer Teorema de Isomorfismo para Semigrupos)**.** *Sea $\varphi : M \to N$ un morfismo de semigrupos. Entonces $Im(\varphi) = \{\varphi(x) | x \in M\}$ es un subsemigrupo de N y además $M/Nuc(\varphi) \cong Im(\varphi)$.*

Demostración. Dado un morfismo de semigrupos $\varphi : M \to N$, el núcleo $Nuc\,\varphi$ es una congruencia por ejemplo (1.14) y por lo tanto, existe el semigrupo cociente $M/Nuc\,\varphi = \{[a] : a \in M\}$.

Por otro lado el conjunto $Im(\varphi) = \{\varphi(x) | x \in M\} \subset N$ es claro que es un subsemigrupo de N. La función $\bar{\varphi} : M/Nuc\,\varphi \to Im\,\varphi$ definida como $\bar{\varphi}([a]) = \varphi(a)$ esta bien definida porque $Nuc(\varphi)$ es una congruencia. Además es un morfismo porque φ lo es. Finalmente para la inyectividad note que si $\bar{\varphi}[a] = \bar{\varphi}[b]$ entonces que $\varphi(a) = \varphi(b)$, es decir $a \equiv b$ y $\bar{\varphi}$ es un isomorfismo. □

Ejemplo 1.15. Es fácil ver que la intersección de congruencias es otra congruencia de M. Además si $E \subset M \times M$ es un subconjunto, la *congruencia generada* por E es la intersección de todas las congruencias que contienen a E.

Definición 1.37 (Presentación de semigrupos)**.** Sea M un semigrupo libre con base X y $\{w_i = u_i : i \in I\}$ una familia de ecuaciones con $w_i, u_i \in M$. Se define la congruencia $\equiv$ como la congruencia generada por el subconjunto $\{(w_i, u_i) | w_i = u_i : i \in I\} \subset M \times M$. El semigrupo cociente $M/\equiv$ se dice que tiene *presentación*:

$$(\, X \,|\, w_i = u_i \text{ para todo } i \in I \,)$$

Capítulo 2

Computabilidad y la tesis de Church-Turing

En este capítulo se introduce el concepto de máquina de Turing, un modelo matemático simple de una computadora. Independientemente de su simplicidad, la máquina de Turing modela la capacidad de cálculo de una computadora de propósitos generales. Se estudia la máquina de Turing debido tanto a la clase de lenguajes que define (los conjuntos recursivamente enumerables) como por la clase de funciones enteras que calcula (funciones parcialmente recursivas).

La máquina de Turing y el concepto de algoritmo son la otra base del teorema de Markov-Post, con ello se da un criterio para la solución del Problema de la Palabra para semigrupos. La primera parte del capítulo está basado en [4], la tesis de Church-Turing en [10] y [12], y las secciones finales en [7].

2.1 Máquinas de Turing

Se puede imaginar una máquina de Turing como una cinta, subdividida en celdas, la cual se extiende infinitamente en ambas direcciones (izquierda y derecha), y una cabeza que lee y escribe.

En el lenguaje cotidiano, cuando se habla de una máquina de Turing T, realmente se refiere a su *programa*. Para describir tal programa se necesitan los símbolos de programación, que consisten de:

i) Los *símbolos de la cinta* $s_0, s_1, \ldots, s_n$ que serán escritos en la cinta. El símbolo s_0 se interpreta como blanco. Cada celda contiene un sólo símbolo. A la colección de todos los símbolos de la cinta distintos de s_0 se le llama el *alfabeto* Σ de T.

ii) Una lista infinita de *estados internos* $q_0, q_1, \ldots$ Al inicio de cada paso de un cálculo o cómputo, la cabeza lectora estará en uno de esos estados.

En un programa de Turing sólo se usará un número finito de estados, sin embargo se necesita un número no acotado de estados disponibles.

iii) Los *símbolos de acción* usados por el programa, que dirá a la cabeza lectora-escritora qué hará en relación a la celda presente: L significa moverse a la izquierda una celda, R significa moverse a la derecha una celda. Además de Imprimir (escribir o borrar) un símbolo.

2.1.1 La máquina de Turing estándar

Definición 2.1 (Máquina de Turing estándar)**.** Una *máquina de Turing* T es una séptupla

$$T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

donde: Q es el conjunto de *estados internos*, Σ es el *alfabeto de entrada*, Γ es un conjunto finito de símbolos llamado el *alfabeto de la cinta*, δ es la *función de transición*, $\square \in \Gamma$ es un símbolo especial llamado el *símbolo blanco*, $q_0 \in Q$ es el *estado inicial* y $F \subseteq Q$ es el conjunto de *estados finales*.

En la definición de máquina de Turing, se supone que $\Sigma \subseteq \Gamma - \{\square\}$, es decir, el alfabeto de entrada es un subconjunto del alfabeto de la cinta, que no incluye al símbolo blanco. La función de transición δ esta definida como

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$$

El argumento de la función δ es la pareja ordenada (q_i, s_i) formada por el estado actual y el símbolo que lee en ese momento. El resultado es la terna ordenada $\delta(q_i, s_i) = (q_j, s_j, A)$; un nuevo estado, un nuevo símbolo sobre la cinta (puede ser el mismo) y un símbolo de acción, L o R. que indica el movimiento de la cabeza lectora-escritora, hacia la izquierda o derecha. En general, δ es una función parcial, esto es, puede no estar definida para algunos valores.

Se puede imaginar una máquina de Turing como un computadora simple. Tiene una unidad de procesador, la cual tiene memoria finita, su cinta es un almacenamiento secundario de capacidad ilimitada. Las instrucciones que puede llevar a cabo tal computadora son muy limitadas: puede leer un símbolo sobre la cinta y utilizar el resultado para decidir qué hacer en el siguiente paso.

La función de transición δ define cómo actúa esta computadora, y frecuentemente es llamado el *programa* de la máquina.

La máquina de Turing empieza en un estado inicial dado con alguna información sobre la cinta. Entonces sigue una sucesión de pasos controlados por la función de transición δ. Durante este proceso, el contenido de cada celda sobre la cinta puede ser leído y cambiado muchas veces. Finalmente, el proceso completo puede terminar, el cual en una máquina de Turing se logra en un *estado de detención*.

Definición 2.2. Una máquina de Turing se *detiene* si alcanza una configuración para la cual δ, la función de transición, no está definida; esto es posible porque δ es una función parcial.

Se asume que para un estado final la función de transición no está definida, así la máquina de Turing se detendrá cuando entre en un estado final.

Ejemplo 2.1. Considérese la máquina de Turing $T_1 = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ dada por $Q = \{q_0, q_1\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{\square, 0, 1\}$, $F = \{q_1\}$ y la función de transición definida por

$$\begin{aligned}\delta(q_0, 0) &= (q_0, 1, R)\\ \delta(q_0, 1) &= (q_0, 1, R)\\ \delta(q_0, \square) &= (q_1, \square, L)\end{aligned}$$

Si esta máquina de Turing inicia en el estado q_0 con el símbolo 0 bajo su cabeza lectora-escritora, la regla de transición aplicable es $\delta(q_0, 0) = (q_0, 1, R)$. Entonces la cabeza lectora-escritora reemplaza 0 por un 1, y luego se mueve a la derecha sobre la cinta. La máquina permanecerá en el estado q_0. Todo 0 subsecuente será reemplazado por un 1, pero los 1's no cambiarán (o serán reemplazado por el mismo símbolo). Cuando la máquina encuentra el primer símbolo blanco, se moverá una celda a la izquierda y se detendrá en el estado final q_1.

Ejemplo 2.2 (Máquina copiadora)**.** Dada una palabra w se diseñará una máquina de Turing que dé como resultado la palabra ww. Por comodidad sólo se trabaja con palabras de un sólo símbolo, el 1.

El proceso para construir la máquina es intuitivo:

1. Reemplazar todo los 1s por una x.

2. Encontrar la x que está más a la derecha y reemplazarla por un 1.

3. Desplazarse hasta el extremo derecho de la región no vacía y en el primer espacio en blanco crear un 1.

4. Repetir el paso 1 y 2 hasta que no haya más xs.

Sean $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{1, x\}$, $\Gamma = \{\square, 1, x\}$, $F = \{q_3\}$ y la función

de transición dada por

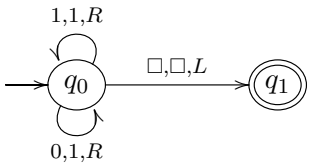
$$\begin{aligned}
\delta(q_0, \square) &= (q_1, \square, L) \\
\delta(q_0, 1) &= (q_0, x, R) \\
\delta(q_1, \square) &= (q_3, \square, R) \\
\delta(q_1, 1) &= (q_1, 1, L) \\
\delta(q_1, x) &= (q_2, 1, R) \\
\delta(q_2, \square) &= (q_1, 1, L) \\
\delta(q_2, 1) &= (q_2, 1, R)
\end{aligned}$$

La máquina $T_2 = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ se conoce como la máquina copiadora. Más adelante se comprobará que, en efecto, copia o duplica palabras de 1s.

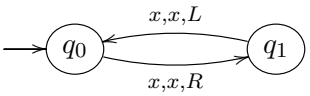
2.1.2 Diagramas de transición

Se usan gráficas de transición para representar máquinas de Turing. Se etiquetan los vértices con los estados y las aristas con los siguientes tres elementos: el símbolo actual de la cinta, el símbolo que lo reemplaza, y la dirección en la cual la cabeza lectora-escritora se mueve.

Ejemplo 2.3. A partir de la función δ se obtiene la siguiente gráfica de transición de la máquina T_1 del ejemplo (2.1):



Ejemplo 2.4 (Máquina con un loop infinito)**.** Observe la máquina de Turing de la siguiente figura, donde x representa cualquier símbolo 0 o 1:



Para obtener la función de transición, supóngase que inicialmente la cinta contiene $01\ldots$, con la cabeza lectora-escritora sobre 0. La máquina entonces lee el símbolo 0, pero no lo cambia. Su siguiente estado es q_1 y la cabeza se mueve a la derecha, así que ahora está sobre el 1. Este símbolo también es leído y se deja sin cambio. La máquina regresa al estado q_0 y la cabeza lectora-escritora se mueve a la izquierda. Ahora se regresa exactamente al estado original, y la sucesión de movimientos inicia otra vez. La función de

transición es entonces:

$$\delta(q_0, 0) = (q_1, 0, R)$$
$$\delta(q_0, 1) = (q_1, 1, R)$$
$$\delta(q_1, 0) = (q_0, 0, L)$$
$$\delta(q_1, 1) = (q_0, 1, L)$$

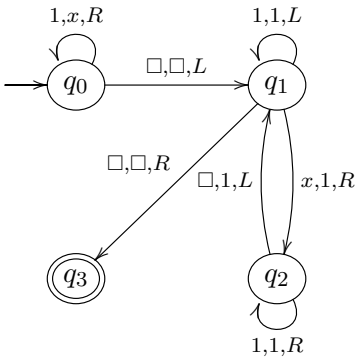
La máquina asociada a esta gráfica de transición es

$$T_3 = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

donde $Q = \{q_0, q_1\}, \Sigma = \{0, 1\}, \Gamma = \{\square, 0, 1\}$ y $F = \{\emptyset\}$.

Para esta máquina es claro que, cualquiera que sea la información inicial sobre su cinta, corre infinitamente, con la cabeza moviéndose a la derecha e izquierda de manera alternada, sin hacer modificación en la cinta. Este es un ejemplo de una máquina de Turing que no se detiene. Análogamente a lo que pasa en Programación, se dice que esta máquina de Turing entra en un *bucle infinito*.

Ejemplo 2.5. Con ayuda de la función de transición δ se obtiene el diagrama de transición de la máquina de Turing copiadora del ejemplo (2.2):



Toda configuración está completamente determinada por el estado actual de la unidad de control, el contenido de la cinta, y la posición de la cabeza lectora-escritora.

Retomando la Definición (1.21). Una palabra positiva es una sucesión finita de símbolos del alfabeto con todos los exponentes positivos o la palabra vacía.

Definición 2.3. Sea $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ una máquina de Turing. Una *descripción instantánea* α es una palabra positiva de la forma $\alpha = \sigma q_i \tau$, donde σ y τ son s-palabras y τ es no vacía, es decir, es una expresión de la forma

$$s_1 s_2 \cdots s_{k-1} q_i s_k s_{k+1} \cdots s_n$$

donde $\Gamma = \{s_1, \ldots, s_n\}$ son los símbolos de la cinta y $\sigma, \tau \in W(\Gamma)$.

Esta elección convenida es tal que la posición de la cabeza lectora está sobre la celda que contiene el símbolo inmediatamente después de q.

La descripción instantánea sólo da una cantidad finita de información hacia el lado izquierdo o derecho de la cabeza lectora. La parte no especificada de la cinta se asume que tiene sólo símbolos blancos; normalmente esos son irrelevantes y no se muestran explícitamente en la descripción instantánea. Si la posición del símbolo blanco es relevante en la discusión, aparecerá en la descripción instantánea. Por ejemplo, la descripción $q\square w$ indica que la cabeza está sobre la celda inmediata a la izquierda del símbolo w y esta celda contiene un blanco.

El movimiento $\delta(q_1, c) = (q_2, e, R)$ se denota

$$abq_1cd \to abeq_2d$$

y es llevado a cabo si el estado interno es q_1, la cinta contiene $abcd$ y la cabeza lectora-escritora está sobre c. El símbolo $\to^*$ significa que hay un número arbitrario de movimientos.

Definición 2.4. Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ una máquina de Turing. Para cualquier descripción instantánea $s_1 \cdots s_{k-1} q_i s_k s_{k+1} \cdots s_n$ con $s_j \in \Gamma$ y $q_i \in Q$. Un *movimiento básico*

$$s_1 \cdots \underline{s_{k-1} q_i s_k} s_{k+1} \cdots s_n \to s_1 \cdots \underline{s_{k-1} b q_j s_{k+1}} \cdots s_n$$

es posible si y sólo si $\delta(q_i, s_k) = (q_j, b, R)$. Un *movimiento básico*

$$s_1 \cdots \underline{s_{k-1} q_i s_k} s_{k+1} \cdots s_n \to s_1 \cdots \underline{q_j s_{k-1} b s_{k+1}} \cdots s_n$$

es posible si y sólo si $\delta(q_i, s_k) = (q_j, b, L)$

Definición 2.5. Una descripción instantánea α es *terminal* si no existe descripción instantánea β tal que $\alpha \to \beta$.

Definición 2.6. Si w es una palabra positiva en el alfabeto Σ de T, entonces T *computa* o *calcula* w si existe una sucesión finita de descripciones instantáneas $\alpha_1 = q_0 w, \alpha_2, \ldots, \alpha_n$ donde $\alpha_i \to \alpha_{i+1}$ es un movimiento básico, para toda $i \leq n-1$, y α_n es terminal.

Ejemplo 2.6. El cómputo de la máquina de Turing T_1 del ejemplo (2.1), sobre la entrada 00 es: $q_000 \to 1q_00 \to 11q_0\square \to 11q_1\square$ o bien

$$q_000 \xrightarrow{*} 11q_1\square.$$

Ejemplo 2.7. La máquina T_2 funciona correctamente. La palabra a duplicar es $w = 11$. Entonces el cómputo realizado es

$$\begin{aligned} q_011 &\to xq_01 \to xxq_0 \to xq_1x \to x1q_2\square \\ &\to xq_111 \to q_1x11 \to 1q_211 \to 11q_21 \\ &\to 111q_2\square \to 11q_111 \to 1q_1111 \\ &\to q_11111 \to q_1\square1111 \to q_31111 \end{aligned}$$

Ejemplo 2.8. El ejemplo (2.4) muestra que la máquina de Turing nunca se detiene, entra en un loop infinito del cual no tiene forma de salir. Se puede representar usando la siguiente notación especial:

$$\alpha_1 q \alpha_2 \xrightarrow{*} \infty$$

lo cual indica que, a partir de una configuración inicial $\alpha_1 q \alpha_2$, la máquina entra en un bucle y nunca se detiene; no tiene descripción instantánea final.

2.1.3 Aceptadores de lenguajes

Las máquinas de Turing pueden ser vistas como reconocedores o aceptadores de lenguaje en el siguiente sentido. Una palabra w es leída sobre la cinta, con símbolos blancos en las otras celdas no usadas. La máquina empieza en un estado inicial q_0 con la cabeza posicionada en el primer símbolo del lado izquierdo de la palabra w. Si después de una sucesión de movimientos básicos, la máquina de Turing entra en un estado final y se detiene, entonces w es considerada como aceptada. Recuerde que una palabra es una sucesión finita de símbolos del alfabeto.

Definición 2.7. Dado Γ un alfabeto. Un *lenguaje* L es un subconjunto de las palabras sobre Γ, es decir, $L \subseteq W(\Gamma)$.

Definición 2.8. Sea $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ una máquina de Turing. El lenguaje aceptado por T es el conjunto

$$L(T) = \{w \in W(\Sigma) : q_0 w \to^* \alpha q_f \beta \text{ para algún } q_f \in F, \alpha, \beta \in W(\Sigma)\}$$

La definición anterior dice lo que sucede cuando $w \in L(T)$. Pero no afirma nada acerca del resultado de cualquier otra entrada. Cuando $w \notin L(T)$, una de dos cosas sucede: la máquina puede detenerse en un estado no final o puede entrar en un bucle infinito y nunca detenerse. Una palabra para la cual T no se detiene, por definición, no está en $L(T)$.

Ejemplo 2.9. El lenguaje aceptado por la máquina de Turing T_1 es

$$L(T_1) = \{w \,|\, w \text{ contiene 0 y 1}\}$$

Ejemplo 2.10. Para $\Sigma = \{0, 1\}$, diseñe una máquina de Turing que reconozca el lenguaje que sólo contenga ceros.

Se inicia en el lado izquierdo de la entrada, la máquina lee cada símbolo y comprueba si es un 0. Si es así, deja el mismo símbolo y sigue moviéndose a la derecha. Si encuentra un símbolo blanco sin encontrarse otro distinto de 0, termina y acepta la palabra. Si el dato de entrada contiene un 1 en alguna parte, la palabra no está en el lenguaje buscado, y se detiene en un estado no final. Para hacer el cómputo, dos estados internos $Q = \{q_0, q_1\}$

y un estado final $F = \{q_1\}$ son suficientes. Como función de transición se puede tomar:

$$\delta(q_0, 0) = (q_0, 0, R)$$
$$\delta(q_0, \square) = (q_1, \square, R)$$

Mientras aparezca 0 bajo la cabeza lectora-escritora, la cabeza se moverá a la derecha. Si en algún momento se lee un 1, la máquina se detendrá en un estado no final q_0, dado que $\delta(q_0, 1)$ no está definida. Observe que la máquina de Turing se detiene en un estado final si empieza en estado q_0 sobre el símbolo blanco. Por lo tanto $T_5 = (\{q_0, q_1\}, \{0, 1\}, \{\square, 0, 1\}, \delta, q_0, \square, \{q_1\})$ es la máquina buscada.

2.1.4 Funciones Turing-calculables

Las máquinas de Turing no sólo son interesantes como aceptadores de lenguajes, sino que proporciona un modelo abstracto de computadora convencional.

El dato de entrada para un cómputo son todos los símbolos no blancos sobre la cinta en un tiempo inicial. Al concluir el cómputo, el dato de salida será lo que esté escrito sobre la cinta. Esto es, se puede ver a la máquina de Turing T, como una implementación de la función f definida por $\hat{w} = f(w)$ si

$$q_0 w \xrightarrow[T]{*} q_f \hat{w}$$

para algún estado final q_f.

Definición 2.9. Dado un conjunto X y $W(X)$ el conjunto de las palabras en X. Una función $f : D \subseteq W(X) \to W(X)$ es *Turing-calculable* o *Turing-computable* si existe una máquina de Turing $T = (Q, \Sigma, \Gamma, \delta_0, \square, F)$ tal que

$$q_0 w \xrightarrow{*} q_f f(w) \quad q_f \in F \quad \text{para toda } w \in D.$$

Ejemplo 2.11. La función $f : \mathbb{N} \to \mathbb{N}$ definida por $f(n) := 2n$, es Turing-calculable. En efecto, por el ejemplo (2.2) existe una máquina de Turing T_2 tal que

$$q_0 w \xrightarrow{*} q_3 ww \quad q_3 \in F \quad \text{para toda } w \in \Sigma$$

donde $1 \in \Sigma$, w se representa con una cadena de 1s según la longitud de ésta y q_3 es el estado final de T_2.

El siguiente ejemplo es tomado de [3].

Proposición 2.1. *La sustracción propia de números naturales* $\ominus : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *definida como*

$$m \ominus n := \begin{cases} m - n & \text{si } m \geq n \\ 0 & \text{si } m < n \end{cases}$$

es una función Turing-calculable.

Demostración. Se probará que existe una máquina de Turing que calcula la diferencia propia de dos números naturales m y n, donde m aparece con m ceros y n con n ceros separados por un 1 sobre la cinta, representados por 0^m10^n.

Sea $T_4 = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ con $Q = \{q_0, \ldots, q_6\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{\square, 0, 1\}$, $F = \{q_6\}$ y δ se construye como sigue.

La máquina empieza con 0^m10^n en su cinta y se define δ de tal manera que se detenga en $0^{m \ominus n}$. T_4 sustituye de manera repetida el 0 del frente por un espacio en blanco, entonces busca hacia la derecha un 1 seguido por un 0 y cambia el 0 por un 1. En seguida T_4 se mueve hacia la izquierda hasta que encuentra un espacio en blanco y entonces repite el ciclo. La repetición termina si:

i) Buscando hacia la derecha un 0, T_4 encuentra un espacio en blanco. Entonces, los n 0's, que están en 0^m10^n han sido cambiados todos a 1s y $n+1$ de los m 0s se han cambiado por $\square$. T_4 sustituye a los $n+1$ 1's por un 0 y después por n $\square$'s dejando $m - n$ 0's en su cinta.

ii) Comenzando el ciclo, T_4 no puede encontrar un 0 que se ha cambiado a un espacio en blanco, debido a que los primeros m 0's ya han sido cambiados. Entonces $n \geq m$, de modo que $m \ominus n = 0$. T_4 sustituye todos los 1's y 0's por $\square$.

La función δ se describe a continuación.

1. $\delta(q_0, 0) = (q_1, \square, R)$

   Comienza el ciclo. Sustituye el 0 del frente por $\square$.

2. $\delta(q_1, 0) = (q_1, 0, R)$

   $\delta(q_1, 1) = (q_2, 1, R)$

   Escruta hacia la derecha, buscando el primer 1.

3. $\delta(q_2, 1) = (q_2, 1, R)$

   $\delta(q_2, 0) = (q_3, 1, L)$

   Escruta hacia la derecha saltándose los 1's hasta que encuentra un 0. Cambia ese 0 por 1.

4. $\delta(q_3, 0) = (q_3, 0, L)$

   $\delta(q_3, 1) = (q_3, 1, L)$

   $\delta(q_3, \square) = (q_0, \square, R)$

   Se mueve hacia la izquierda a un espacio en blanco. Accesa al estado q_0 para repetir el proceso.

5. $\delta(q_2, \square) = (q_4, \square, L)$

   $\delta(q_4, 1) = (q_4, \square, L)$

   $\delta(q_4, 0) = (q_4, 0, L)$

   $\delta(q_4, \square) = (q_6, 0, R)$

   Si en el estado q_2 se encuentra un $\square$ antes que 0, se tiene la situación i) descrita anteriormente. Se accesa al estado q_4 y se mueve hacia la izquierda, cambiando todos los 1s por $\square$s hasta encontrar un $\square$. Este $\square$ se cambia de nuevo a 0, el estado q_6 es accesado y T_4 se detiene.

6. $\delta(q_0, 1) = (q_5, \square, R)$

   $\delta(q_5, 0) = (q_5, \square, R)$

   $\delta(q_5, 1) = (q_5, \square, R)$

   $\delta(q_5, \square) = (q_6, \square, R)$

   Si en el estado q_6 se encuentra un 1 en lugar de un 0, el primer bloque de 0s ha sido agotado, como en la situación ii) anterior. T_4 accesa al estado q_5 para borrar el resto de la cinta, entonces accesa al estado q_6 y se detiene.

Una muestra del cálculo de T_4 sobre la entrada 0010 es:

$$\begin{aligned} q_0 0010 &\to \square q_1 010 \to \square 0 q_1 00 \to \square 01 q_2 0 \\ &\to \square 0 q_3 11 \to \square q_3 011 \to q_3 \square 011 \to \square q_0 011 \\ &\to \square\square q_1 11 \to \square\square 1 q_2 1 \to \square\square 11 q_2 \to \square\square 1 q_4 1 \\ &\to \square\square q_4 1 \to \square q_4 \to \square 0 q_6 \end{aligned}$$

Sobre la entrada 0100, T_4 se comporta de la manera siguiente:

$$\begin{aligned} q_0 0100 &\to \square q_1 100 \to \square 1 q_2 00 \to \square q_3 1100 \\ &\to q_3 \square 110 \to \square q_0 110 \to \square\square q_5 10 \to \square\square\square q_5 0 \\ &\to \square\square\square\square q_5 \to \square\square\square\square\square q_6 \end{aligned}$$

Así, se ha demostrado que la *sustracción propia* de números naturales $m \ominus n$ es una función Turing-calculable. $\square$

Como se ha visto hasta el momento, lo que interesa de cualquier máquina son las *instrucciones* que determina la función δ, es decir, su *programa*. El programa para una máquina de Turing T estará formado por un conjunto finto de instrucciones, llamadas *cuádruplas*, conformada de símbolos de programación. Cualquier cuádrupla Q tendrá la forma:

$$Q = q_i S A q_j$$

donde q_i y q_j son estados internos, S el símbolo de la cinta y A el símbolo de acción. Entonces, la instrucción o cuádrupla $Q = q_iSAq_j$ se lee de izquierda a derecha: *si T está en un estado q_i leyendo en la cinta el símbolo S, entonces lleva a cabo la acción A y pasa a un nuevo estado interno q_j.*

Nota 2.1. De ahora en adelante, cuando se hable de una máquina de Turing, se referirá a su *programa*. Así, una *máquina de Turing* T consistirá de un conjunto *finito de cuádruplas* $\{Q_1, \ldots, Q_n\}$, llamado el *programa de Turing*.

Ejemplo 2.12. El conjunto finito de cuádruplas

$$\{q_0 0 R q_0, q_0 1 R q_0, q_0 s_0 L q_1\}$$

obtenido a partir de la función de transición δ es el programa de la máquina de Turing T_1 del ejemplo (2.1).

Ejemplo 2.13. Para la máquina de Turing T_2, la máquina copiadora del ejemplo (2.2), el programa es el conjunto finito

$$\{q_0 s_0 L q_1, q_0 1 R q_0, q_1 s_0 R q_3, q_1 1 L q_1, q_1 x R q_2, q_2 s_0 L q_1, q_2 1 R q_2\}.$$

Ejemplo 2.14. El programa de Turing de la máquina con loop infinito, del ejemplo (2.4) es el conjunto finito de cuádruplas

$$\{q_0 0 R q_1, q_1 1 L q_0\}.$$

Los movimientos básicos se clasificarán en cinco tipos de acuerdo al movimiento de la máquina. Esta clasificación será de utilidad en el Capítulo 3, por tal razón el símbolo blanco $\square$ es denotado por s_0, la palabra vacía, para estar acorde a la notación de [7] usada en el siguiente capítulo.

Definición 2.10. Sea $T = (Q, \Sigma, \Gamma, \delta, q_0, s_0, F)$ una máquina de Turing. El par (α, β) de descripciones instantáneas es un *movimiento básico* de T, denotado por $\alpha \to \beta$, si existen s-palabras positivas (posiblemente la palabra vacía s_0) σ y $\sigma' \in W(\Gamma)$ tal que una de las siguientes condiciones se cumple:

Mov. 1) $\alpha = \sigma q_i s_j \sigma'$ y $\beta = \sigma q_l s_k \sigma'$ donde $q_i s_j s_k q_l \in T$

Mov. 2) $\alpha = \sigma q_i s_j s_k \sigma$ y $\beta = \sigma s_j q_l s_k \sigma'$ donde $q_i s_j R q_l \in T$

Mov. 3) $\alpha = \sigma q_i s_j$ y $\beta = \sigma s_j q_l s_0$ donde $q_i s_j R q_l \in T$

Mov. 4) $\alpha = \sigma s_k q_i s_j \sigma'$ y $\beta = \sigma q_l s_k s_j \sigma'$ donde $q_i s_j L q_l \in T$

Mov. 5) $\alpha = q_i s_j \sigma'$ y $\beta = q_l s_0 s_j \sigma'$ donde $q_i s_j L q_l \in T$

Si α describe la cinta en un cierto tiempo, en el estado q_i lee el símbolo s_j, entonces β describe la cinta en el siguiente estado q_l de T que lee el símbolo s_k después del movimiento.

La máquina realiza los movimientos 2) y 3) si su cabeza se mueve a la derecha, donde el tipo 3) ocurre cuando la siguiente descripción instantánea la cabeza está sobre el símbolo blanco s_0. De manera similar, los movimientos 4) y 5) se llevan a cabo si la máquina se mueve hacia la izquierda y en el tipo 5) la cabeza lee el símbolo blanco s_0 después del movimiento.

En el primer movimiento la máquina de Turing sólo cambia el símbolo pero la cabeza no se mueve, es decir, hay que mostrar la existencia de otro modelo de máquina de Turing, las que no mueven la cabeza y sólo imprimen un símbolo, y además demostrar que hace todo lo que hace la máquina estándar. Para probar la afirmación, la tesis de Church-Turing y el concepto de máquina simuladora, serán de gran utilidad.

2.2 La tesis de Church-Turing

La máquina de Turing, planteada por Alan M. Turing en 1936, fue propuesta por Church como un modelo matemático para describir procedimientos.

Definición 2.11. Un *algoritmo* es un procedimiento con un número finito de pasos, cada uno de éstos preciso y suficientemente simple, y cuya ejecución (la del procedimiento) siempre termina.

Según el contexto, por algoritmo también se entenderá como un proceso mecánico, efectivo y calculable (ver [5] Pag. 165). Es intuitivo ver que todo proceso llevado a cabo por una máquina de Turing T es mecánico, efectivo y calculable. Es decir, el proceso llevado a cabo por una máquina de Turing que se detiene es algorítmico. La afirmación inversa se conoce como la Tesis de Church-Turing.

La tesis de Church-Turing afirma que todo proceso que intuitivamente se considere como algoritmo puede ser ejecutado por una máquina de Turing. Hasta el momento esta tesis se ha cumplido. La única manera de desacreditarla es encontrando algún procedimiento que no pueda ser ejecutado por una máquina de Turing y que sin embargo pueda ser calculado por alguna máquina con más capacidad que ella.

De la definición de la máquina de Turing quedará suficientemente claro que cualquier cómputo que se pueda describir con una de ellas puede ser llevado mecánicamente. También se puede demostrar que cualquier cómputo que se pueda llevar a cabo en una computadora digital como se les conoce hoy en día, puede ser descrito a través de una máquina de Turing, para ello puede consultar [12].

Hay otras formalizaciones de procedimientos que se puede demostrar que son equivalentes a la máquina de Turing, reforzando la idea de que la

definición que dió Turing a su máquina es suficientemente general como para comprender la idea intuitiva que se tiene por un procedimiento. La Tesis de Church-Turing afirma que cualquier proceso que se reconoce naturalmente como algoritmo puede ser llevado a cabo por una máquina de Turing.

Tesis de Church-Turing. *Toda solución algorítmica para un problema puede ser representado por un programa de instrucciones para alguna máquina de Turing.*

En otras palabras todo procedimiento mecánico, efectivo y calculable puede ser llevado a cabo por una máquina de Turing. Asumiendo cierta la Tesis hay una equivalencia entre el concepto de ***algoritmo*** *y el de* ***Turing-calculable****. De ahora en adelante se asume cierta la tesis.*

2.3 Equivalencia de máquinas de Turing

La definición de máquina de Turing estándar no es la única posible, hay definiciones alternativas que pueden ser realmente útiles.

Por la Tesis de Church-Turing, añadir al modelo estándar de máquina de Turing más almacenamiento, no afecta la capacidad de cómputo de la máquina. Cualquier cálculo que realice, con un nuevo arreglo caerá en la categoría de cómputo mecánico y podrá ser llevado a cabo por un modelo estándar.

Definición 2.12. Dos máquinas de Turing, T y T', son *equivalentes* si aceptan o reconocen el mismo lenguaje $L(T) = L(T')$.

Sean $\mathcal{T}_1$ y $\mathcal{T}_2$ dos clases de máquinas de Turing. Si para toda máquina de Turing T_1 en $\mathcal{T}_1$ existe una máquina de Turing T_2 en $\mathcal{T}_2$ tal que

$$L(T_1) = L(T_2)$$

se dice que la capacidad de cómputo de $\mathcal{T}_2$ es tanto como la de $\mathcal{T}_1$. Si el recíproco también ocurre y para toda máquina T_2 en $\mathcal{T}_2$ existe una máquina T_1 en $\mathcal{T}_1$ tal que $L(T_1) = L(T_2)$, se dice que $\mathcal{T}_1$ y $\mathcal{T}_2$ son equivalentes. Para demostrar la equivalencia de máquinas de Turing se usará la técnica de simulación.

2.3.1 Máquina simuladora

Definición 2.13. Sea T una máquina de Turing. Se dice que otra máquina de Turing T' puede *simular* un cómputo de T si T' puede imitar el cómputo de T de la siguiente manera. Dada $\alpha_0, \alpha_1, \alpha_2, \ldots$ una sucesión de descripciones instantáneas del cómputo de T

$$\alpha_0 \xrightarrow[T]{} \alpha_1 \xrightarrow[T]{} \alpha_2 \xrightarrow[T]{} \cdots \xrightarrow[T]{} \alpha_n \xrightarrow[T]{} \cdots$$

Entonces T' simula este cómputo si realiza

$$\hat{\alpha}_0 \xrightarrow[T']{} \hat{\alpha}_1 \xrightarrow[T']{} \hat{\alpha}_2 \xrightarrow[T']{} \cdots \xrightarrow[T']{} \hat{\alpha}_n \xrightarrow[T']{} \cdots$$

donde $\hat{\alpha}_0, \hat{\alpha}_1, \ldots$ son descripciones instantáneas, de tal manera que cada una de ellas está asociada a una única configuración de T. En otras palabras, si se conoce el cómputo llevado a cabo por T', se puede determinar exactamente qué cómputos pudo haber realizado T, dada la correspondiente configuración inicial.

La simulación de un único movimiento $\alpha_i \to \alpha_{i+1}$ de T puede implicar varios movimientos de T'. La configuración intermedia en

$$\hat{\alpha}_i \xrightarrow[T']{*} \hat{\alpha}_{i+1}$$

puede no corresponder a una configuración de T, pero esto no lo afecta si se dice qué configuraciones de T' son relevantes. Siempre que se pueda determinar a partir de los cómputos en T' lo que T pudo realizar, la simulación es propia. Si T' puede simular todo cómputo de T, se dice que T' *simula* T. Es claro que si T' simula T, T' hace un re-arreglo de las configuraciones de T, así que las máquinas T y T' aceptan el mismo lenguaje y son equivalentes.

Para demostrar la equivalencia de dos clases de máquinas de Turing, se mostrará que para toda máquina de la primera clase, existe una máquina en la segunda clase que la simule.

2.3.2 Máquina con opción de no mover la cabeza

En la definición de máquina de Turing estándar, la cabeza lectora-escritora se mueve a la izquierda o a la derecha. Algunas veces es conveniente proveer una tercera opción, que la cabeza permanezca en el mismo lugar después de reescribir el contenido de la celda. Esto es, se puede definir una máquina de Turing con la opción de no mover la cabeza reemplazando δ en la definición por

$$\hat{\delta} : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$$

donde S significa que no hay ningún movimiento en la cabeza lectora-escritora. Esta opción no extiende la capacidad de cómputo en la máquina de Turing.

Teorema 2.2. *La clase de máquinas de Turing con la opción de no mover la cabeza es equivalente a la clase de máquinas de Turing estándar.*

Demostración. Como una máquina de Turing con opción de no mover la cabeza es claramente una extensión del modelo estándar, es claro que toda máquina de Turing estándar puede ser simulada por la máquina con opción de no mover la cabeza.

Para demostrar el recíproco, sea $T' = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ una máquina de Turing con la opción de no mover la cabeza y sea $T = (\hat{Q}, \Sigma, \Gamma, \hat{\delta}, \hat{q_0}, \square, \hat{F})$ una máquina de Turing estándar. Supóngase que T' es simulada por T. Para cada movimiento de T', la máquina simuladora T hace lo siguiente.

Si el movimiento de T' *no involucra* la opción *Stay* de no mover la cabeza, la máquina simuladora realiza un movimiento, *esencialmente idéntico* al movimiento que simulará. Si S *está involucrado* en el movimiento de T', entonces T *hará dos movimientos*:

i) El primero: reescribe el símbolo y mueve la cabeza a la derecha.

ii) El segundo: mueve la cabeza a la izquierda y deja el contenido de la cinta sin alterar.

La máquina simuladora se construye a partir de T' definiendo $\hat{\delta}$ como sigue. Para cada transición $\delta(q_i, a) = (q_j, b, L \text{ o } R)$ se define

$$\hat{\delta}(\hat{q}_i, a) := (\hat{q}_j, b, L \text{ o } R)$$

Para cada S-transición $\delta(q_i, a) = (q_j, b, S)$ la correspondiente transición $\hat{\delta}$ es

$$\hat{\delta} := \begin{cases} \hat{\delta}(\hat{q}_i, a) = (\hat{q_j}_S, b, R) \\ \hat{\delta}(\hat{q}_{jS}, c) = (\hat{q}_j, c, L) \quad \text{para toda } c \in \Gamma \end{cases}$$

De lo anterior, todo cómputo de T' tiene su correspondiente cómputo en T, así T simula T'. Por lo tanto T y T' son máquinas de Turing equivalentes. □

2.3.3 Máquinas multicintas

Definición 2.14. Una *máquina de Turing multicinta* consiste de un control finito con k cabezas lectoras y k cintas. Cada cinta es infinita en ambos sentidos. La máquina define su movimiento dependiendo del símbolo que está leyendo cada una de sus cabezas, da reglas de sustitución para cada uno de los símbolos y dirección de movimiento para cada una de las cabezas. Inicialmente, la máquina empieza con la entrada en la primera cinta y el resto de las cintas en blanco.

Teorema 2.3. *La clase de máquinas de Turing multicintas es equivalente a la clase de máquinas de Turing estándar.*

Puede consultar la demostración en [12].

Además de las máquinas multicintas, existen otros modelos de máquinas de Turing que son equivalentes a la estándar:

- Máquinas con una cinta extendida sólo en una dirección.
- Máquinas multidimensional, como aquella cuya cinta puede extenderse infinitamente en más de una dirección.

2.4 La máquina universal de Turing

Las máquinas de Turing construidas hasta el momento fueron programadas para realizar un determinado proceso, mientras que la computadora digital es de propósitos generales, se le puede reprogramar para que la misma máquina pueda llevar a cabo cualquier proceso susceptible de ser programado.

Una máquina Turing reprogramable, esto es, que sea capaz de llevar a cabo cualquier proceso que se le indique, se le conoce como la *máquina universal de Turing*. Esta máquina es capaz de seguir cualquier programa para cualquier máquina de Turing que se le dé como dato de entrada.

La máquina universal de Turing consiste de tres cintas:

- En la primera se encuentra el programa o la descripción (la tabla) de la máquina de Turing que debe simular.
- En la segunda cinta se encuentra lo que corresponde a la cinta de la máquina de Turing a simular.
- En la tercera cinta se encuentra el estado en el que la máquina de Turing original se encontraría.

Para poder hacer esta simulación se debe codificar la tabla original, de alguna manera que sea suficientemente general, en este caso, usando notación unaria. Suponga que la máquina de Turing original tiene n estados $\{q_1, q_2, \ldots, q_n\}$, esta lista ordenada de tal manera que q_1 es el estado inicial y q_2 es el único estado final. Se codifica a los estados mediante el índice:

$$q_1 = 1, q_2 = 11, \ldots, q_k = 1^k.$$

Se hace lo mismo con los símbolos del alfabeto, reservando la palabra 1 para el símbolo blanco. En cuanto a la dirección de movimientos de la cabeza, se hace la siguiente codificación:

$$L = 1, R = 11$$

Para denotar a una transición, se da una quinteta, donde se separan a los elementos de la quinteta entre sí con un 0. Por ejemplo, si los símbolos del alfabeto de la máquina original son $\{a, b, c, d, e\}$, la quinteta 11101011011110110 representa la transición $\delta(q_3, \square) = (q_2, c, R)$ en la máquina de Turing original.

Al iniciar la máquina universal su funcionamiento, tiene en la primera cinta la tabla de la máquina original. En la segunda cinta tiene la palabra de entrada, pero codificada de esta manera, y en la tercera cinta tendrá un 1, indicando que se encuentra en el estado inicial de la máquina original.

Cada transición de la máquina original, la máquina universal M_u la consigue haciendo lo siguiente:

i) Localiza en la primera cinta aquella quinteta en la que el primer componente sea igual al que se encuentra en la tercera cinta, y el segundo componente sea igual al que se encuentra en la segunda cinta.

ii) Copia a la tercera cinta el tercer componente de la primera cinta (el estado al que se transfiere).

iii) Sustituye la segunda cinta por lo que dice el cuarto componente.

iv) Interpreta el movimiento de la cabeza que se encuentra en el quinto componente, moviendo acorde la cabeza de la segunda cinta.

Cuando la máquina universal M_u no encuentra una transición (la máquina universal pararía), M_u se traslada a un estado donde verifica si se encuentra o no en estado final (si el contenido de la tercera cinta es 11).

La construcción de la máquina universal de Turing M_u es un proceso sumamente detallado. Sin embargo, no es difícil hacerlo en un lenguaje de programación tradicional.

2.5 Subconjuntos recursivamente enumerables

Toda máquina de Turing determina dos conjuntos, las palabras que están en la cinta antes del cómputo y las palabras escritas después del cómputo. En el primer caso, están los subconjuntos recursivamente enumerables y los subconjuntos recursivos.

Definición 2.15. Sea $S = \{s_1, \ldots, s_m\}$ y $\Omega := W(S^+)$ el conjunto de todas las palabras positivas en S. Si $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ es una máquina de Turing cuyo alfabeto Σ contiene a S, se define el conjunto

$$e(T) := \{w \in \Omega \mid T \text{ calcula o acepta } w\}$$

y se dice que T *enumera* $e(T)$.

Definición 2.16. Un subconjunto $E \subseteq \Omega$ es *recursivamente enumerable* (*r.e.*) si existe alguna máquina de Turing T que enumera E, es decir, si $E = e(T)$.

El conjunto E es recursivamente enumerable si E como lenguaje, es aceptado por alguna máquina de Turing T, es decir, toda $w \in E$ es aceptado por T.

Ejemplo 2.15. El conjunto de los números naturales es un subconjunto recursivamente enumerable. En efecto, existe la máquina de Turing T_2 que reconoce al conjunto de todos los números naturales. Así, existe

$$e(T_2) = \{n \in \mathbb{N} \mid T_2 \text{ acepta } n\}$$

Lema 2.4. *Sea S un conjunto infinito numerable. Entonces su conjunto potencia 2^S es no numerable.*

Demostración. Sea $S = \{s_1, s_2, s_3, \ldots\}$ un conjunto infinito y numerable. Entonces todo elemento $t \in 2^S$ puede ser representado por una sucesión de 0's y 1's, con 1 en posición i si y sólo si s_i está en t. Por ejemplo, el conjunto formado por $\{s_2, s_3, s_6\}$ está representado por $01100100\ldots$. Es claro que, todo elemento de 2^S puede ser representado por una sucesión de ésta manera, y que tal sucesión representa un único elemento de 2^S.

Si 2^S es infinito numerable; entonces sus elementos pueden ser escritos en algún orden, por ejemplo $t_1, t_2, t_3, \ldots$, y por lo tanto sus elementos se pueden poner en una tabla, como la siguiente:

t_1	(1)	0	0	0	0	$\cdots$
t_2	1	(1)	0	0	0	$\cdots$
t_3	1	1	(0)	1	0	$\cdots$
t_4	1	1	0	(0)	1	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

En esta tabla, se toman los elementos que están en la diagonal principal, y se complementa cada entrada, es decir, se reemplaza cada 0 por un 1 y viceversa. En el ejemplo de la tabla, el elemento es $1100\ldots$, así se obtiene $0011\ldots$ como resultado. La nueva sucesión a lo largo de la diagonal representa algún elemento de 2^S, sin pérdida de generalidad t_i, para alguna i. Pero no puede ser t_1 porque difiere de t_1 en s_1 por construcción. Por la misma razón no puede ser t_2, t_3 o cualquier otra entrada de los elementos enumerados. Se llega a esta contradicción al suponer 2^S numerable, lo cual debe ser falso. Por lo tanto 2^S es no numerable. □

Como una consecuencia inmediata de este resultado, se puede demostrar, que existen menos máquinas de Turing que lenguajes, así debe haber un lenguaje que no sea recursivamente enumerable. Pero antes, es necesario demostrar que el conjunto de las máquinas de Turing es infinito numerable.

De la Nota (2.1), el programa o máquina de Turing, consiste de un conjunto finito de cuádruplas de la forma q_iSAq_j donde cada una de las letras están en el conjunto numerable

$$\{R, L, s_0, s_1, s_2, \ldots; q_0, q_1, q_2, \ldots\}$$

Se asigna un número natural a cada una de las letras de la siguiente manera:

$$\begin{array}{ccccccc} R \mapsto 0 & L \mapsto 1 & q_0 \mapsto 2 & q_1 \mapsto 4 & q_2 \mapsto 6 & \cdots \\ & s_0 \mapsto 3 & s_1 \mapsto 5 & s_2 \mapsto 7 & \cdots & \end{array}$$

Definición 2.17. Sea T una máquina de Turing con m cuádruplas

$$\{q_0 S_1 A_1 q_1, \ldots, q_m S_m A_m q_f\}$$

se ordenan y concatenan las cuádruplas para formar la palabra

$$w(T) = q_0 S_1 A_1 q_1 \cdots q_m S_m A_m q_f$$

de longitud $4m$.

Definición 2.18 (Número de Gödel)**.** El número de Gödel asociado a la máquina T es

$$G(T) = \prod_{i=1}^{4m} p_i^{e_i}$$

donde p_i es el i-ésimo primo y e_i es el número natural asignado en a la i-ésima letra en $w(T)$.

Ejemplo 2.16. El programa de Turing de la máquina T_1 tiene 3 cuádruplas, así la nueva palabra concatenada

$$w(T_1) = q_0 0 R q_0 q_0 1 R q_0 q_0 s_0 L q_1$$

es de longitud $4 \cdot 3$ y su correspondiente número de Gödel es

$$\begin{aligned} G(T_1) &= \prod_{i=1}^{12} p_i^{e_i} \\ &= 2^2 3^5 5^0 7^2 11^2 13^7 17^0 19^2 23^2 29^3 31^1 37^4 \\ &\approx 9.7853562 \times 10^{31} \end{aligned}$$

Ejemplo 2.17. El programa de Turing de la máquina T_3 tiene 2 cuádruplas, la palabra concatenada

$$w(T_3) = q_0 0 R q_1 q_1 1 L q_0$$

tiene longitud $4 \cdot 2$. El número de Gödel asociado a ésta máquina es

$$\begin{aligned} G(T_3) &= \prod_{i=1}^{8} p_i^{e_i} \\ &= 2^2 3^5 5^0 7^4 11^4 13^7 17^1 19^2 \\ &\approx 1.315796585 \times 10^{22} \end{aligned}$$

Teorema 2.5. *Existe una cantidad infinita numerable de máquinas de Turing.*

Demostración. Primero se probará que máquinas de Turing distintas tienen números de Gödel distintos.

Afirmación 2.1. $T \neq T'$ si y sólo si $G(T) \neq G(T')$.

En efecto, por nota (2.1), la máquina de Turing consiste del conjunto finito de cuádruplas en su programa. Así, toda máquina T queda determinada por la palabra $w(T)$, que consiste en la concatenación de las cuádruplas del programa de Turing. Por lo tanto, si T y T' máquinas de Turing distintas, entonces las palabras $w(T)$ y $w(T')$ también son distintas y por lo tanto los respectivos números de Gödel, por el teorema Fundamental de la Aritmética, también son distintos.

Por la afirmación anterior, se pueden numerar todas las máquinas de Turing: $T_0, T_1, T_2, \ldots, T_n, \ldots$, donde T precede a T' si $G(T) < G(T')$. En consecuencia, hay tantas máquinas de Turing como números naturales. □

Corolario 2.6. *Para cualquier conjunto no vacío Σ, existe un lenguaje que no es recursivamente enumerable.*

Demostración. Un lenguaje es un subconjunto de $W(\Sigma)$, y todos y cada uno de los subconjuntos es un lenguaje. Entonces el conjunto de todos los lenguajes es exactamente $2^{W(\Sigma)}$. Como $W(\Sigma)$ es infinito, el Lema (2.4) asegura que el conjunto de todos los lenguajes sobre $W(\Sigma)$ es no numerable. Pero el conjunto todas las máquinas de Turing es numerable por el Teorema (2.5), y así el conjunto de todos los lenguajes recursivamente enumerable es numerable. Esto implica necesariamente que existe un lenguaje sobre Σ que no es recursivamente enumerable. □

Esta prueba, aunque corta y simple, no es del todo satisfactoria. No es constructiva y, mientras que asegura la existencia de un lenguaje que no es recursivamente enumerable, no dice quién es.

2.6 Subconjuntos recursivos

Definición 2.19. Sea $S = \{s_1, \ldots, s_m\}$ y $\Omega = W(S^+)$ el conjunto de todas las palabras positivas en S. Un subconjunto $E \subseteq \Omega$ es *recursivo* si ambos, E y su complemento $\Omega - E$ son subconjuntos recursivamente enumerables.

Si E es recursivo, no existe una espera infinita para decidir si una palabra positiva w está o no en E. Si T es una máquina de Turing con $e(T) = E$ y si T' es una máquina de Turing con $e(T') = \Omega - E$, entonces para cada $w \in \Omega$, o bien T calcula w ó T' calcula w. Es decir, se puede decidir en una cantidad finita de tiempo si una palabra dada w está o no en E: sólo poniendo w en

cada una de los dos máquinas T y T' corriendo simultáneamente, y una de las dos se detendrá.

Es claro que todo conjunto recursivo es recursivamente enumerable.

Lema 2.7. *Todo subconjunto recursivo es recursivamente enumerable.*

Asumiendo la Tesis de Church-Turing, si todo lenguaje que puede ser descrito directamente en forma algorítmica puede ser aceptado por una máquina de Turing y así es recursivamente enumerable, la descripción de un lenguaje que no es recursivamente enumerable debe ser indirecta. Sin embargo, es posible. El argumento utilizado es similar al proceso de diagonalización de Cantor.

Teorema 2.8. *Existe un subconjunto recursivamente enumerable de los números naturales $\mathbb{N}$ que no es recursivo.*

Demostración. Por el Teorema (2.5), existe una cantidad infinita numerable de máquinas de Turing, se enumeran de la siguiente manera:

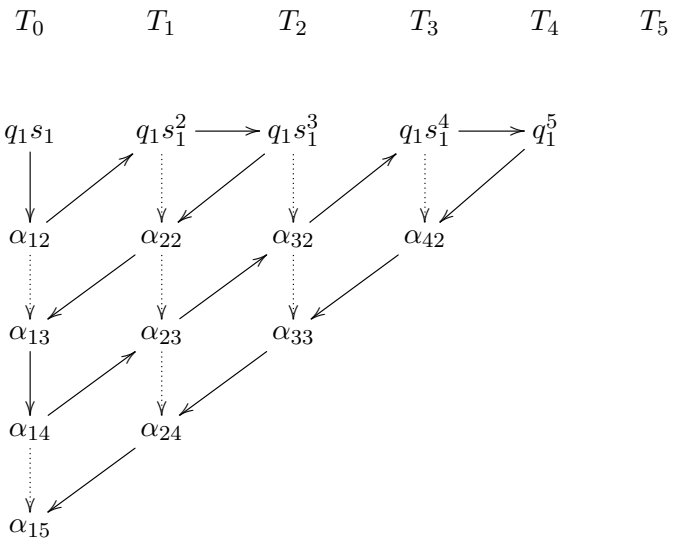
$$T_0, T_1, T_2, \ldots T_n, \ldots,$$

donde T precede a T' si $G(T) < G(T')$. Considere el conjunto

$$E = \{n \in \mathbb{N} : T_n \text{ computa } s_1^{n+1}\},$$

entonces $n \in E$ si y sólo si la n-ésima máquina de Turing T_n computa n, donde n se representa por $s_1^{n+1} := n$.

Se afirma que E es un conjunto recursivamente enumerable. En efecto, observe la siguiente figura:



La segunda columna consiste de una sucesión de movimientos básicos de la segunda máquina de Turing $T_1 = (Q_1, \Sigma, \Gamma, \delta_1, q_{0_1}, \square, F_1)$ que inicia con

$q_{0_1}s_1^2$. Como T_1 computa 1, aquí representado por s_1^2, ésta sucesión es finita y se tiene

$$q_{0_1}s_1^2 \xrightarrow{*} q_{f_1}w_1$$

La n-ésima columna consiste de la sucesión de movimientos básicos de la n-ésima máquina de Turing $T_n = (Q_n, \Sigma, \Gamma, \delta_n, q_{0_n}, \square, F_n)$ que inicia con $q_1 s_1^{n+1}$. Por el mismo argumento T_n computa $n := s_1^{n+1}$ y ésta sucesión es finita

$$q_0^n s_1^{n+1} \xrightarrow{*} q_{f_n}w_n$$

Es claro que existe una enumeración del número natural $n \in E$: siguiendo las flechas en la figura, y tan pronto como se tome n en E se llega a una descripción terminal α_{ni} en la columna n.

Se construirá una máquina de Turing T^* que lleve a cabo esas instrucciones.

Sea $T^* = (Q^*, \Sigma, \Gamma, \delta^*, q_0^*, \square, F^*)$ una máquina de Turing con el mismo alfabeto Σ y Γ que las máquinas T_n para toda $n \in \mathbb{N}$. Además, sea $Q^* = F^* = \{q_0^*\}$.

Entonces, ésta es una máquina con el mismo alfabeto y un único estado inicial y final llamado q_0^*, y lo que hará es simular cualquier máquina T_n, para ello la función de transición es

$$\delta^* : Q^* \times \Gamma \to Q^* \times \Gamma \times \{L, R\} \quad \text{como} \quad \delta^*(q_0^*, s_1^{n+1}) := \delta_n(q_{0_n}, s_1^{n+1})$$

¿Qué hace esta máquina T^*? Dado $n = s_1^{n+1} \in \Sigma$, la máquina definida por la función de transición $\delta^*(q_0^*, s_1^{n+1})$ busca la correspondiente máquina T_n y lleva a cabo $\delta_n(q_{0_n}, s_1^{n+1})$. Como T_n computa n, existe una sucesión finita y por lo tanto una descripción instantánea terminal. Esto es, E es un subconjunto recursivamente enumerable de $\mathbb{N}$.

El argumento para probar que E no es recursivo es una variación del argumento diagonal de Cantor. Es suficiente demostrar que el complemento

$$\overline{E} := \{n \in \mathbb{N} : n \notin E\} = \{n \in \mathbb{N} : T_n \text{ no computa } s_1^{n+1}\}$$

no es un subconjunto recursivamente enumerable de $\mathbb{N}$.

Suponga que $\overline{E}$ es r.e., así existe una máquina de Turing T' que enumera $\overline{E} = e(T')$; como todas las máquinas de Turing han sido listadas, $T' = T_m$ para algún $m \in \mathbb{N}$. Si $m \in \overline{E} = e(T') = e(T_m)$, entonces T_m computa s_1^{m+1}, y así $m \in E$, una contradicción. Finalmente, si $m \notin \overline{E}$, entonces $m \in E$ y así T_m computa s_1^{m+1} por definición de E; por lo tanto $m \in e(T_m) = e(T') = \overline{E}$. Entonces, $\overline{E}$ no es un conjunto recursivamente enumerable y por consiguiente E no es recursivo. □

Del teorema anterior se concluye que hay más conjuntos recursivamente enumerables que recursivos. La familia de los subconjuntos recursivos cumplen propiedades realmente útiles.

Teorema 2.9. *Sea Ω un conjunto de palabras y $E, E_1, E_2 \subseteq \Omega$.*

i) Si E_1 y E_2 son recursivos, entonces $E_1 \cup E_2$ es recursivo.

ii) Si E_1 y E_2 son recursivos, entonces $E_1 \cap E_2$ es recursivo.

iii) Si E es recursivo, entonces E^c también es recursivo.

Es decir, la familia de conjuntos recursivos forman un álgebra de Boole.

Demostración. Sean $E_1, E_2 \subset \Omega$ subconjuntos recursivos, entonces tanto E_1, E_1^c y E_2, E_2^c son recursivamente enumerables, así existen máquinas de Turing T_1, T_1' y T_2, T_2' que calculan y enumeran E_1, E_1^c, E_2 y E_2^c, respectivamente:

$$\begin{aligned} e(T_1) &= E_1 \\ e(T_1') &= \Omega - E_1 \\ e(T_2) &= E_2 \\ e(T_2') &= \Omega - E_2 \end{aligned}$$

i) Para calcular y enumerar $E_1 \cup E_2$ se define $\widehat{T}$ y $\widehat{T}'$ de la siguiente manera:

$$\widehat{T}(w) = \begin{cases} T_1(w) & \text{si } w \in E_1 \\ T_2(w) & \text{si } w \in E_2 \end{cases} \qquad \widehat{T}'(w) = \begin{cases} T_1'(w) & \text{si } w \in \Omega - E_1 \\ T_2'(w) & \text{si } w \in \Omega - E_2 \end{cases}$$

Por lo tanto $e(\widehat{T}) = E_1 \cup E_2$ y $e(\widehat{T}') = \Omega - (E_1 \cup E_2)$ y por consiguiente $E_1 \cup E_2$ es recursivo.

ii) La máquina $\widehat{T}$ calcula y enumera $E_1 \cap E_2$ y la máquina $\widehat{T}'$ calcula y enumera $\Omega - (E_1 \cap E_2)$. Por lo tanto $E_1 \cap E_2$ es recursivo.

iii) Si E es recursivo entonces existen T y T' máquinas de Turing que calculan E y $\Omega - E$ respectivamente:

$$\begin{aligned} e(T) &= E = \Omega - (\Omega - E) \\ e(T') &= \Omega - E \end{aligned}$$

Así $E^c = \Omega - E$ es recursivo.

□

2.7 El problema de la detención

Es un hecho que hay máquinas de Turing que frente a ciertas palabras de entrada nunca van a detenerse (Ver ejemplo 2.4). Sería útil saber, dada una máquina de Turing T arbitraria y una palabra $w \in W(\Sigma)$, si T se detendrá o no cuando procese a w.

Si se tuviera una máquina de Turing $\mathcal{H}$, que predijera el comportamiento de otra máquina de Turing, sería equivalente a decir que todos los lenguajes recursivamente enumerables son recursivos; en lugar de alimentarle las palabras a la T original, se le alimenta a $\mathcal{H}$, junto con la descripción de T; si $\mathcal{H}$ dice que T va a detenerse, se echa a andar a T con la palabra w para ver si la acepta o no. Si $\mathcal{H}$ dice que no va a a detenerse al procesar w, se sabe que T rechaza a w. Dada esta situación, para una máquina de Turing arbitraria y una palabra w se puede decir si T la acepta o no, sin caer en el ciclo infinito. Se demuestra que $\mathcal{H}$ no puede existir (ver [12], Capítulo 10). A este problema se le conoce como el *problema de la detención* o *problema de la parada*.

Teorema 2.10. *El problema de la detención es indecidible, esto es, no existe un algoritmo (una máquina de Turing reconocedora) que pueda decidir si una máquina de Turing arbitraria se detendrá o no frente a una palabra arbitraria.*

Definición 2.20. Se reduce un problema P en uno o varios problemas $P_1, \ldots, P_k$, cuando se describe las soluciones de P en términos de las soluciones a $P_1, \ldots, P_k$.

Lo natural es elegir a $P_1, \ldots, P_k$ que sean más sencillos que P, aunque a veces será necesario reducirlo en un problema más complejo que ya tenga solución.

2.8 El problema de correspondencia de Post

Definición 2.21. Una instancia del *Problema de Correspondencia de Post (PCP)* consiste en dos listas, $A = w_1, \ldots, w_k$ y $B = x_1, \ldots, x_k$, de palabras sobre algún alfabeto Σ. Esta instancia del PCP *tiene una solución* si existe cualquier sucesión de enteros $i_1, i_2, \ldots, i_m$ con $m \geq 1$, tal que

$$w_{i_1}, w_{i_2}, \ldots, w_{i_m} = x_{i_1}, x_{i_2}, \ldots, x_{i_m}.$$

La sucesión $i_1, i_2, \ldots, i_m$ es una solución a esta instancia del PCP.

Ejemplo 2.18. Sea $\Sigma = \{0, 1\}$. Sean A y B las listas de tres palabras cada una, como se define en la tabla (2.1).

En este caso PCP tiene una solución. Sea $m = 4, i_1 = 2, i_2 = 1, i_3 = 1$ e $i_4 = 3$. Entonces

$$w_2 w_1 w_1 w_3 = x_2 x_1 x_1 x_3 = 101111110.$$

Ejemplo 2.19. Sea $\Sigma = \{0, 1\}$. Sean A y B las listas de tres palabras que se presentan en la tabla (2.2)

	Lista A	Lista B
i	w_i	x_i
1	1	111
2	10111	10
3	10	0

Tabla 2.1: Una instancia del PCP

	Lista A	Lista B
i	w_i	x_i
1	10	101
2	011	11
3	101	011

Tabla 2.2: Otra instancia del PCP

Supóngase que esta instancia del PCP tiene una solución $i_1, i_2, \ldots, i_m$. Es claro que $i_1 = 1$, ya que ninguna palabra que comienza con $w_2 = 011$ puede igualar a la palabra que comienza con $x_2 = 11$; ninguna palabra que comienza con $w_3 = 101$ puede igualarse a una cadena que comienza con $x_3 = 011$.

Escribimos la palabra de la lista A encima de la correspondiente palabra B. Hasta aquí tenemos

$$\begin{matrix} 10 \\ 101 \end{matrix}$$

La siguiente selección de A debe comenzar con 1. Por consiguiente $i_2 = 1$ o $i_2 = 3$. Pero $i_2 = 1$ no funcionará, puesto que ninguna palabra que comienza con $w_1 w_1 = 1010$ puede igualar a una palabra que comience con $x_1 x_1 = 101101$. Con $i_2 = 3$ tenemos

$$\begin{matrix} 10101 \\ 101011 \end{matrix}$$

Con la palabra de la lista B de nuevo excede a la cadena de la lista A por el símbolo simple 1, un argumento similar al anterior muestra que $i_3 = i_4 = \cdots = 3$. Por tanto existe solamente una sucesión de alternativas que genera cadenas compatibles, y para esta sucesión, la palabra de B es siempre un carácter más larga. Por consiguiente esta instancia del PCP no tiene solución.

El PCP es un problema indecidible, para ver su demostración puede consultar [3].

Capítulo 3

El teorema de Markov-Post

Novikov, Boone y Britton probaron de manera independiente que existe un grupo finitamente presentado $\mathcal{B}$ para el cual no existe algoritmo que determine si una palabra arbitraria w, expresada en los generadores de $\mathcal{B}$, sucede o no que $w =_{\mathcal{B}} 1$. El teorema de Markov-Post asegura la misma afirmación para semigrupos. El objetivo de este capítulo es presentar este teorema y como consecuencia, demostrar el teorema de Novikov-Boone-Britton, además de probar que el Problema de la Palabra para grupos finitamente presentados no siempre tiene solución. El contenido de este capítulo está basado en [7].

3.1 El Problema de la Palabra

En 1910, el matemático Max Dehn, formuló tres problemas de decisión para grupos finitamente presentados: el problema del isomorfismo, el problema de conjugación y el problema de la palabra, en el cual se pregunta si existe un algoritmo para determinar si dos grupos son isomorfos, dos palabras son conjugadas y si dos palabras son iguales, respectivamente. En este capítulo se aborda el tercer problema, el *problema de la palabra*.

Definición 3.1 (**Problema de la Palabra**)**.** Sea G un grupo con presentación finita. *¿Existe un algoritmo que determine si una palabra arbitraria w, expresada en los generadores de G, es o no la palabra* 1*?*

Definición 3.2. Sea G un grupo finitamente generado con presentación finita

$$G = (x_1, \ldots, x_n \,|\, r_j = 1,\ j \geq 1)$$

donde toda palabra (no necesariamente reducida) $w \in X = \{x_1, \ldots, x_n\}$ determina un elemento de G (llamado wR, L es un grupo libre con base X y R es el subgrupo normal de L generado por $\{r_j, j \geq 1\}$). El *problema de la palabra para G tiene solución* si existe un algoritmo que determine si el

conjunto de preguntas de la forma: ¿$w \in W(X)$, entonces $w = 1$ en G? tiene por respuesta sí o no.

3.1.1 Ejemplos de solución

En algunos grupos el problema de la palabra tiene solución. Se mostrará que existe el algoritmo describiendo tal proceso de manera intuitiva.

Sea G un grupo finitamente generado con presentación finita

$$G = (x_1, \ldots, x_n \,|\, r_j = 1,\ j \geq 1)$$

Dado el conjunto $X = \{x_1, \ldots, x_n\}$ se ordenan todas las palabras en $W(X^{\pm})$ de la siguiente manera:

- Primero la palabra de longitud cero: la palabra vacía 1.
- Después las palabras de longitud uno: $x_1, x_1^{-1}, x_2, x_2^{-1}, \ldots, x_n, x_n^{-1}$.
- Luego las palabras de longitud dos puestas en orden lexicográfico, es decir, como en un diccionario:

$$x_1x_1 < x_1x_1^{-1} < x_1x_2 < \cdots < x_1^{-1}x_1 < x_1^{-1}x_1^{-1} < \cdots < x_n^{-1}x_n^{-1}$$

- Posteriormente las palabras de longitud tres en el mismo orden lexicográfico, y así sucesivamente.

Se usa el siguiente orden en las palabras: $w_0, w_1, w_2, w_3, \ldots$ para definir la lista $\mathcal{L}$, donde la k-ésima pregunta cuestiona si $w_k = 1$ en G.

Teorema 3.1. *El problema de la palabra para el grupo libre*

$$G = (x_1, \ldots, x_n \,|\, \emptyset),$$

finitamente generado y con presentación vacía, tiene solución.

Demostración. El algoritmo es el siguiente:

1. Si $|w_k| = 0$ o $|w_k| = 1$, proceder al paso 3. Si $|w_k| \geq 2$, subrayar el primer par de letras adyacentes, si es de la forma $x_ix_i^{-1}$ o $x_i^{-1}x_i$; si no es así, subrayar las dos últimas letras y proceder al paso 2.
2. Si el par de letras subrayadas es de la forma $x_ix_i^{-1}$ o $x_i^{-1}x_i$, eliminar y proceder al paso 1. En otro caso, continuar con el paso 3.
3. Si la palabra subrayada es la palabra vacía, escribir $w_k = 1$ detenerse; si no es la palabra vacía, escribir $w_k \neq 1$ y terminar el proceso.

□

Corolario 3.2. *El problema de la palabra para grupos finitos tiene solución.*

Demostración. El algoritmo es el mismo. □

3.1.2 La máquina de Turing y el Problema de la Palabra

Por la Tesis de Church-Turing (sección 2.2), el Problema de la Palabra tiene solución en un grupo finitamente presentado si y sólo existe una máquina de Turing que lleve a cabo tal proceso algorítmico, más precisamente, que el conjunto sea recursivo. Por el Teorema (1.3), el conjunto $W(X)$ sólo es un monoide (y por lo tanto semigrupo), por lo cual, es necesario relacionar el problema de la palabra con semigrupos libres, finitamente generados y finitamente presentados.

Sea Γ un semigrupo con generadores $X = \{x_1, \ldots, x_n\}$ y Ω el conjunto de todas las palabras positivas sobre X, entonces el *problema de la palabra para el semigrupo* Γ *tiene solución* si existe un algoritmo que determine, para un par de palabras w y $w' \in \Omega$, si $w = w'$ en Γ. Esta definición informal da una definición precisa para la *no solución* del problema.

Definición 3.3. Sea Γ un semigrupo con generadores $X = \{x_1, \ldots, x_n\}$ y $\Omega = W(X^+)$ el conjunto de todas las palabras positivas en X. El *problema de la palabra para el semigrupo* Γ *no tiene solución* si existe una palabra $w_0 \in \Omega$ tal que el conjunto $\{w \in \Omega : w = w_0 \text{ en } \Gamma\}$ no es recursivo.

Si L es un grupo libre con base $X = \{x_1, \ldots, x_n\}$, se considera el conjunto Ω de todas las palabras (no necesariamente positivas) en X como el conjunto de todas las palabras positivas sobre el alfabeto $\{x_1, x_1^{-1}, \ldots, x_n, x_n^{-1}\}$.

Definición 3.4. Sea G un grupo con presentación $(x_1, \ldots, x_n \,|\, \Delta)$ y Ω el conjunto de todas las palabras sobre $x_1, \ldots, x_n$ (visto como el conjunto de palabras positivas de $\{x_1, x_1^{-1}, \ldots, x_n, x_n^{-1}\}$). El *problema de la palabra para* G *tiene solución* si el conjunto $\{w \in \Omega : w = 1 \text{ en } G\}$ es recursivo.

En otras palabras, en el grupo G tiene solución para el problema de la palabra si existe un algoritmo (máquina de Turing) que responda con un sí o un no la igualdad $w = 1$, para cualquier $w \in \Omega$.

Teorema 3.3. *Sea G un grupo finitamente generado con presentación finita*

$$G = (x_1, \ldots, x_n \,|\, r_j = 1,\ 1 \leq j \leq m)$$

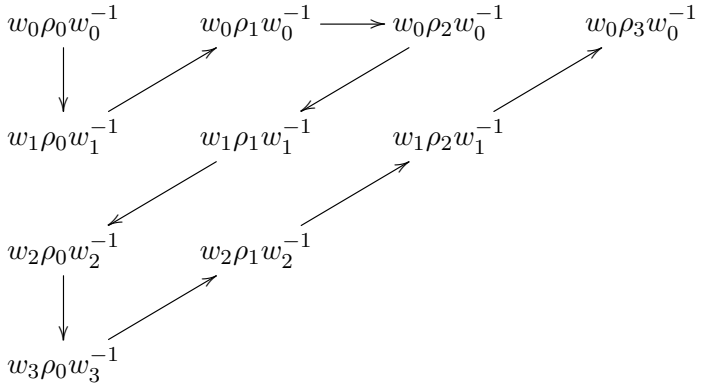
Si Ω es el conjunto de todas las palabras en $\{x_1, \ldots, x_n\}$, entonces el conjunto $E = \{ w \in \Omega : w = 1 \text{ en } G \}$ es recursivamente enumerable.

Demostración. Se enlistan las palabras en Ω como lo hecho previamente al Teorema (3.1): primero la palabra vacía, luego las palabras de longitud uno con el orden $x_1, x_1^{-1}, \ldots, x_n, x_n^{-1}$, después las palabras de longitud 2 en orden lexicográfico, después las palabras de longitud tres en orden lexicográfico y así sucesivamente. A este conjunto lo denotamos por $\{w_0, w_1, w_2, w_3, \ldots\}$.

Análogamente, se enlistan las palabras sobre el conjunto $\{r_1, \ldots, r_m\}$ y denotamos a este conjunto por $\{\rho_0, \rho_1, \rho_2, \rho_3, \ldots\}$.

El grupo $G = L/N$ donde $L = \langle x_1, \ldots, x_n \rangle$ y $N = \langle X \rangle^G$ es la clausura normal de $R = \{r_1, \ldots, r_m\}$. Los elementos de N son de la forma $w\rho w^{-1}$ por el Teorema (1.11).

Como en la prueba del Teorema (2.8), al seguir las flechas en el siguiente diagrama se enumera E.



□

El problema de la palabra para un grupo finitamente presentado G tiene solución si y sólo si E es recursivo, si y sólo si, $E^c = \{w \in \Omega : w \neq 1 \text{ en } G\}$ es recursivamente enumerable, dado que E ya es recursivamente enumerable por el Teorema (3.3).

En lo que sigue, si w y w' son palabras (no necesariamente reducidas) de un alfabeto X, se escribe $w \equiv w'$ si w y w' se escriben de la misma manera.

3.1.3 El Problema de la Palabra para semigrupos

Considere un semigrupo Γ con presentación

$$\Gamma = \{ X \mid \alpha_j = \beta_j, j \in J \}$$

Si w y w' son palabras positivas sobre X, entonces es fácil ver que $w = w'$ en Γ si y sólo si existe una sucesión finita

$$w \equiv w_1 \to w_2 \to \cdots \to w_t \equiv w'.$$

Definición 3.5. Una *operación elemental* $w_i \to w_{i+1}$ ocurre si o $w_i \equiv \sigma\alpha_j\tau$ y $w_{i+1} \equiv \sigma\beta_j\tau$ para alguna j, donde $\sigma, \tau \in W^+(X)$ son palabras positivas o $w_{i+1} \equiv \sigma\beta_j\tau$ y $w_i \equiv \sigma\alpha_j\tau$.

Ejemplo 3.1. Si $\sigma = s_1s_2$ y $\tau = s_4s_5$, entonces $s_0s_1q_1s_3s_4s_5 \to s_0s_1s_3q_2s_4s_5$ es una operación elemental.

Se asociará un semigrupo a una máquina de Turing T con estado de detención q_0. Por conveniencia en la notación, las s-letras y q-letras involucradas en las cuádruplas de T serán $s_0, \ldots, s_M$ y $q_0, \ldots, q_N$. En lo que sigue q y h son nuevas letras.

Definición 3.6. (**Semigrupo asociado a una máquina de Turing**) Sea T una máquina de Turing con estado de detención q_0, el semigrupo $\Gamma(T)$, asociado a T, tiene por presentación:

$$\Gamma(T) = (q, h, s_0, s_1, \ldots, s_M, q_0, q_1, \ldots, q_N \mid R(T))$$

donde el conjunto de las relaciones, es el conjunto finito $R(T)$

$$q_i s_j = q_l s_k \quad \text{si } q_i s_j s_k q_l \in T \tag{3.1}$$
$$q_i s_j s_\beta = s_j q_l s_\beta \quad \text{si } q_i s_j R q_l \in T \tag{3.2}$$
$$q_i s_j h = s_j q_l s_0 h \quad \text{si } q_i s_j R q_l \in T \tag{3.3}$$
$$s_\beta q_i s_j = q_l s_\beta s_j \quad \text{si } q_i s_j L q_l \in T \tag{3.4}$$
$$h q_i s_j = h q_l s_0 s_j \quad \text{si } q_i s_j L q_l \in T \tag{3.5}$$
$$q_0 s_\beta = q_0 \tag{3.6}$$
$$s_\beta q_0 h = q_0 h \tag{3.7}$$
$$h q_0 h = q \tag{3.8}$$

para toda $\beta = 0, 1, \ldots, M$.

Ejemplo 3.2. La máquina de Turing T_1 del ejemplo 2.1 tiene por cuádruplas al conjunto finito

$$\{q_0 0 R q_0, q_0 1 R q_0, q_0 s_0 L q_1\}.$$

Estas cuádruplas determinan las ecuaciones $R(T_1)$:

$$q_0 0 s = 0 q_0 s$$
$$q_0 1 s = 1 q_0 s$$
$$h q_0 \square = h q_1 \square s,$$

donde $s \in \{0, 1\}$. Por lo tanto T_1 tiene asociado el semigrupo finitamente generado y finitamente presentado

$$\Gamma(T_1) = (q, h, \square, 0, 1, q_0, q_1 | R(T_1)).$$

Los primeros cinco tipos de relaciones son sólo las sugeridas por los movimientos básicos de T: sólo imprime un símbolo pero no mueve la cabeza (ecuación 3.1), la nueva letra h hace posible distinguir el movimiento básico 3.2 del movimiento básico 3.3 cuando la cabeza se mueve a la derecha y para distinguir el movimiento básico 3.4 del movimiento 3.5 cuando se mueve a la izquierda.

Definición 3.7. Una palabra es *h-especial* si tiene la forma $h\alpha h$, donde α es una descripción instantánea.

Observación 3.1. Como T tiene estado de detención q_0, cada $h\alpha h$ (con α terminal) tiene la forma $h\sigma q_0 \tau h$, donde σ y τ son s-palabras y τ es no vacía. Por lo tanto, las últimas tres relaciones permiten escribir $h\alpha h = q$ en $\Gamma(T)$ donde α es terminal.

$$\begin{aligned} h\sigma q_0 \tau h &= h\sigma q_0 h \quad \text{por ecuación 3.6} \\ &= h q_0 h \quad \text{por ecuación 3.7} \\ &= q \quad \text{por ecuación 3.8} \end{aligned}$$

Lema 3.4. *Sea T una máquina de Turing con estado de detención q_0 y semigrupo asociado*

$$\Gamma(T) = (\, q, h, s_0, s_1, \ldots, s_M, q_0, q_1, \ldots, q_N \mid R(T)\,)$$

(i) Sean w y w' palabras sobre $\{s_0, s_1, \ldots, s_M, q_0, q_1, \ldots, q_N\}$ con $w \not\equiv q$ y $w' \not\equiv q$. Si $w \to w'$ es una operación elemental, entonces w es h-especial si y sólo si w' es h-especial.

(ii) Si $w = h\alpha h$ es h-especial, $w' \not\equiv q$, y $w \to w'$ es una operación elemental de uno de los primeros cinco tipos, entonces $w' \equiv h\beta h$, donde o $\alpha \to \beta$ o $\beta \to \alpha$ es un movimiento básico.

Demostración. (i) Como $w \not\equiv q$ y $w' \not\equiv q$ se asegura que tanto w y w' no son descripciones instantáneas terminales por la observación 3.1.

Si $w \to w'$ es una operación elemental entonces $w = \sigma\alpha\tau$ y $w' = \sigma\beta\tau$ y $\alpha \to \beta$ es un movimiento básico. Sin pérdida de generalidad se puede suponer $\alpha = s_i q_j s_{i+1}$ y $\beta = s_i s_{i+1} q_k$. Si w es h-especial entonces

$$w = h\sigma s_i q_j s_{i+1} \tau h \to h\sigma s_i s_{i+1} q_k \tau h = w'$$

y w' es h-especial. De manera similar se tiene que si w' es h-especial entonces w también es h-especial.

(ii) Sea $w = h\alpha h$ una palabra h-especial y w' una palabra que no sea descripción instantánea final. Por el inciso anterior, si $w = h\sigma\alpha\tau$ es una palabra h-especial y $w \to w'$ es una operación elemental entonces w' es h-especial y $w' \equiv h\sigma\beta\tau h$, para algunas s-palabras σ y τ.

Una operación elemental en el semigrupo asociado $\Gamma(T)$ corresponde a una substitución usando una ecuación en la relación definida en 3.6; tal relación en $\Gamma(T)$ en uno de los primeros cinco tipos corresponde a una cuádrupla de la máquina de Turing T, y una cuádrupla corresponde a un movimiento básico. Esto es, o $\alpha \to \beta$ o $\beta \to \alpha$.

□

Lema 3.5. *Sea T una máquina de Turing con estado de detención q_0, sea Ω el conjunto de todas las palabras positivas en el alfabeto de T, y sea $E = e(T)$. Si $w \in \Omega$, entonces*

$$w \in E \text{ si y sólo si } hq_i wh = q \text{ en } \Gamma(T)$$

Demostración. Si $w \in \Omega$, entonces existen descripciones instantáneas

$$\alpha_1 = q_1 w \to \alpha_2 \to \cdots \to \alpha_t$$

donde $\alpha_i \to \alpha_{i+1}$ es una operación elemental y α_t involucra el estado final q_0. Aplicando operaciones elementales en $\Gamma(T)$ de los primeros cinco tipos en 3.6 y por el lema anterior 3.4 se observa que

$$hq_1 wh = h\alpha_1 h \to h\alpha_2 h \to \cdots \to h\alpha_t h$$

en $\Gamma(T)$. Usando las últimas tres relaciones y la Observación 3.1 se tiene que $h\alpha_t h = q$ en $\Gamma(T)$.

Ahora veamos la suficiencia. Primero obsérvese que la igualdad en $\Gamma(T)$ es una relación simétrica, mientras que un movimiento básico $\alpha \to \beta$ no implica necesariamente que $\beta \to \alpha$ sea también un movimiento básico, así que la prueba será diferente.

Si $hq_1 wh = q$ en $\Gamma(T)$, entonces existen palabras $w_1, \ldots, w_t$ en el conjunto $\{h, s_0, s_1, \ldots, s_M, q_1, q_1, \ldots, q_N\}$ y operaciones elementales

$$hq_1 wh \equiv w_1 \to w_2 \to \cdots \to w_t \equiv hq_0 h \to q$$

Por el Lema 3.4, cada w_i es h-especial, $w_i = h\alpha_i h$ para alguna descripción instantánea α_i. Por el inciso (ii) del lema anterior se tiene que $\alpha_i \to \alpha_{i+1}$ o $\alpha_{i+1} \to \alpha_i$. Se demostrará por inducción sobre $t \geq 2$, que todas las flechas van hacia la derecha; esto es, para toda $i \leq t-1$, se tiene $\alpha_i \to \alpha_{i+1}$.

Es siempre cierto que $\alpha_{t-1} \to \alpha_t$, para α_t terminal y así $\alpha_{t-1} \leftarrow \alpha_t$ no sucede. En particular, esto prueba que para el caso $t = 2$ es cierto. Supóngase $t > 2$ y que existe alguna flecha hacia la izquierda. Como la última flecha $\alpha_{t-1} \to \alpha_t$ apunta a la derecha, se busca hacia atrás hasta localizar una flecha que apunte hacia la izquierda, por lo tanto existe un índice i con

$$\alpha_{i-1} \leftarrow \alpha_i \to \alpha_{i+1}$$

Pero dado que los movimientos de la máquina de Turing T están determinados por la δ que es una *función* no existe ambigüedad con el próximo movimiento de la máquina, así que $\alpha_{i-1} \equiv \alpha_{i+1}$ y

$$w_{i-1} \equiv h\alpha_{i-1}h \equiv h\alpha_{i+1}h \equiv w_{i+1}$$

Se puede así eliminar w_i y w_{i+1}, de este modo se reduce t, y la prueba se completa usando la hipótesis inductiva.

Finalmente, se tiene $q_1 w \to \alpha_2 \to \cdots \to \alpha_t$ es una sucesión de movimientos básicos con α_t terminal (y α_t involucra el estado de detención q_0); así T computa w y $w \in E$. □

3.2 El teorema de Markov-Post

Teorema 3.6 (**Markov-Post**). *(i) Existe un semigrupo finitamente presentado*

$$\gamma = (q, h, s_0, s_1, \ldots, s_M, q_0, q_1, \ldots, s_N \mid R)$$

donde el problema de la palabra no tiene solución.

(ii) No existe algoritmo que determine, para una palabra h-especial arbitraria $h\alpha h$, si $h\alpha h = q$ en el semigrupo γ.

Demostración. (i) Si T es una máquina de Turing con estado de detención q_0 y alfabeto $A = \{s_0, s_1, \ldots, s_M\}$, sea $\Omega := W^+(A)$ el semigrupo de todas las palabras positivas en A y sea $E = e(T) \subset \Omega$ el conjunto que enumera (computa) T.

Sea $\overline{\Omega} := W^+(A \cup \{q, h, q_0, q_1, \ldots, q_N\})$ el semigrupo de todas las palabras positivas sobre $A \cup \{q, h, q_0, q_1, \ldots, q_N\}$ donde $q_0, q_1, \ldots, q_N$ son las q-letras que aparecen en las cuádruplas de T. Se denota por

$$\overline{E} = \{ \overline{w} \in \overline{\Omega} \mid \overline{w} = q \text{ en } \Gamma(T) \}$$

Sea $\varphi : \Omega \to \overline{\Omega}$ definida por $\varphi(w) := hq_1wh$ y se identifica Ω con su imagen directa $\Omega_1 := \varphi(\Omega) \subset \overline{\Omega}$; el subconjunto $E \subseteq \Omega$ es ahora identificado con

$$E_1 := \varphi(E) = \{ hq_1wh : w \in E \} \subset \varphi(\Omega) := \Omega_1$$

Afirmación 3.1. $E_1 = \overline{E} \cap \Omega_1$.

$\subseteq$) Es claro que $E_1 \subset \Omega_1$. Sea $y = hq_1wh \in E_1$ para alguna $w \in E := e(T)$. Por el Lema 3.5 $w \in E$ si y sólo si $hq_1wh = q \in \Gamma$, por lo tanto $y = hq_1wh \in \overline{E}$.

$\supseteq$) Sea $z \in \overline{E} \cap \Omega_1$, entonces $z \in \Omega_1 = \varphi(\Omega)$ y existe $w \in \Omega$ tal que $z = \varphi(w) = hq_1wh$. Además como $z \in \overline{E}$ se tiene que

$$z := hq_1wh = q \in \Gamma(T).$$

Por el Lema 3.5 $w \in E$ y así $z \in E_1$.

De lo anterior y dado que los subconjuntos recursivos forman un álgebra de Boole (2.9) se tiene la siguiente afirmación:

Afirmación 3.2. E_1 es un subconjunto recursivo de Ω_1 si y sólo si E es un subconjunto recursivo de Ω.

Sea T la máquina de Turing T^* con estado de detención q_0 del Teorema 2.8, así E y por lo tanto E_1 es un conjunto recursivamente enumerable, pero no es recursivo.

Si $\overline{E}$ fuera recursivo, dado que los subconjuntos recursivos forman un álgebra booleana, E_1 sería recursivo y así también E, lo cual es una contradicción. Así, para el semigrupo finitamente presentado $\gamma = \Gamma(T^*)$ el problema de la palabra no tiene solución.

(ii) En un argumento similar al inciso anterior, considere

$$\overline{S} = \{\, h\alpha h \,:\, h\alpha h = q \text{ en } \Gamma(T^*) \text{ y } h\alpha h \text{ palabra especial} \,\}$$

Si $\overline{S}$ fuera un conjunto recursivo de Ω, entonces $\overline{S} \cap \Omega_1$ sería un subconjunto recursivo de Ω_1. Pero $\overline{S} \cap \Omega_1 = E_1$ no es recursivo.

Se concluye que no existe algoritmo que determine si dos palabras h-especiales son iguales en γ, o su equivalente, si sucede $h\alpha h = q \in \gamma(T^*)$. □

Corolario 3.7. *(i) Existe un semigrupo finitamente presentado*

$$\Gamma = (\, q, q_0, \ldots, q_N, s_0, \ldots, s_M \mid F_i q_{i_2} G_i = H_i q_{i_2} K_i, i \in I \,)$$

donde el problema de la palabra no tiene solución, con F_i, G_i, H_i, K_i son s-palabras positivas (posiblemente vacías) y $q_{i_1}, q_{i_2} \in \{q, q_0, \ldots, q_N\}$.

(ii) No existe algoritmo que determine, para q_{i_j} arbitrario y s-palabras positivas X y Y, si $Xq_{i_j}Y = q$ en Γ.

Demostración. (i) Considere el generador h del semigrupo $\gamma = \Gamma(T^*)$ como la última s-letra y re-indique esas s-letras de tal manera que $h = s_M$.

Las relaciones re-escritas en $R(T^*)$ ahora tienen la forma descrita y cumple las condiciones del teorema de Markov-Post (3.6) y la afirmación queda demostrada.

(ii) Sea $\Omega_2 := W^+(q, q_0, \ldots, q_N, s_0, \ldots, s_M)$ el semigrupo de todas las palabras positivas sobre los generadores re-escritos de Γ.

Considere los conjuntos

$$\overline{\Lambda} = \{\, Xq_iY : Xq_{i_j}Y = q \text{ en } \Gamma \,\}$$

donde X y Y son palabras positivas el semigrupo de s-palabras re-escritas en el inciso anterior y

$$\overline{S}_2 = \{\, s_M \alpha s_M \,:\, \alpha \equiv \sigma q_{i_j} \tau \text{ y } s_M \alpha s_M = q \text{ en } \Gamma \,\}$$

con σ y τ son palabras positivas de $s_0, \ldots, s_{M-1}$ (donde h ha sido re-escrita como s_M).

Desde luego $\overline{S}_2$ es el subconjunto $\overline{S}$ del teorema de Markov-Post escrito de otra manera. Ahora $\overline{\Lambda} \cap \Omega_2 = \overline{S}_2$, como $\overline{S}_2$ no es recursivo, por ser un álgebra de Boole se tiene que $\overline{\Lambda}$ no es recursivo. Por lo tanto no existe algoritmo que determine si dos s-palabras positivas son iguales, o de manera equivalente, si tiene la igualdad $Xq_{i_j}Y = q \in \Gamma$.

□

3.3 Aplicaciones del teorema de Markov-Post

3.3.1 El problema de la palabra para grupos

Para la prueba que sigue la igualdad de palabras en un grupo se reducirá a la igualdad de palabras en un semigrupo. Y es importante hacer notar los exponentes, porque mientras que para un grupo una palabra arbitraria tiene sentido, para un semigrupo sólo tienen sentido las palabras positivas, es decir, de exponentes positivos.

Definición 3.8. Sea $X \equiv s_{\beta_1}^{e_1} \cdots s_{\beta_m}^{e_m}$ es una s-palabra (no necesariamente positiva) entonces $X^{\#} \equiv s_{\beta_1}^{-e_1} \cdots s_{\beta_m}^{-e_m}$.

Nota 3.1. De la definición anterior se observa que si X y Y son s-palabras, entonces $(X^{\#})^{\#} \equiv X$ y $(XY)^{\#} \equiv X^{\#}Y^{\#}$.

Por el Corolario 3.7 se sabe que para una máquina de Turing T, existe un semigrupo finitamente generado $\Gamma = \Gamma(T)$ con la presentación finita

$$\Gamma = (q, q_0, \ldots, q_N, s_0, \ldots, s_M \mid F_i q_{i_1} G_i = H_i q_{i_2} K_i, i \in I)$$

donde F_i, G_i, H_i, K_i son s-palabras positivas (posiblemente vacías) y $q_{i_1}, q_{i_2} \in \{q, q_0, \ldots, q_N\}$.

En lo que sigue se asociará un grupo a una máquina de Turing T y se demostrará que existe un grupo finitamente presentado donde el problema de la palabra no tiene solución como lo hace el teorema de Markov-Post si se elige T como la máquina T^* que se escribe en el capítulo 2.

Definición 3.9. (**Grupo asociado a una máquina de Turing**) Sea T máquina de Turing con estado de detención q_0, el grupo asociado a T tiene por presentación

$$\mathcal{B} = \mathcal{B}(T) = (q, q_0, \ldots, q_N, s_0, \ldots, s_M, r_i, i \in I, x, t, k \mid R(T))$$

donde el conjunto de las relaciones $R(T)$ es el siguiente conjunto finito

$$xs_\beta = s_\beta x^2 \tag{3.9}$$
$$r_i s_\beta = s_\beta x r_i x \tag{3.10}$$
$$r_i^{-1} F_i^{\#} q_{i_1} G_i r_i = H_i^{\#} q_{i_2} K_i \tag{3.11}$$
$$t r_i = r_i t \tag{3.12}$$
$$tx = xt \tag{3.13}$$
$$k r_i = r_i k \tag{3.14}$$
$$kx = xk \tag{3.15}$$
$$k(q^{-1}tq) = (q^{-1}tq)k \tag{3.16}$$

para toda $i \in I$ y para toda $\beta = 0, \ldots, M$.

Ejemplo 3.3. La máquina de Turing T_1 del ejemplo 2.1 tiene por cuádruplas al conjunto finito

$$\{q_0 0 R q_0, q_0 1 R q_0, q_0 s_0 L q_1\}.$$

Estas cuádruplas determinan las ecuaciones $R(T_1)$:

$$\begin{gathered} xs_\beta = s_\beta x^2 \\ r_i s_\beta = s_\beta x r_i x \\ r_i F_i^{\#} q_{i_1} G_i r_i = H_i^{\#} q_{i_2} K_i \\ t r_i = r_i t \\ tx = xt \\ k r_i = r_i k \\ kx = xk \\ k(q^{-1}tq) = (q^{-1}tq)k \end{gathered}$$

donde $s_\beta \in \{0, 1\}$, I un conjunto finito e $i \in I$, F_i, G_i, H_i y K_i son s-palabras.

Por lo tanto T_1 tiene asociado el grupo finitamente generado y finitamente presentado

$$\mathcal{B}(T_1) = (\ q, q_0, q_1, \square, 0, 1, r_i, x, t, k \mid R(T_1)\ ).$$

Definición 3.10. Si X y Y son s-palabras, definimos

$$(X q_j Y)^* \equiv X^{\#} q_j Y$$

donde $q_j \in \{q, q_0, \ldots, q_N\}$.

Definición 3.11. Una palabra S es *especial* si $S \equiv X^{\#} q_j Y$, donde X y Y son s-palabras positivas y $q_j \in \{q, q_0, \ldots, q_N\}$

Ejemplo 3.4. Si $X = 101$ y $Y = 010$ en la máquina de Turing T_1 del ejemplo 2.1 entonces la siguiente palabra es especial

$$(101q_0010)^* = (Xq_0Y)^* \equiv X^{\#}q_0Y = 1^{-1}0^{-1}1^{-1}q_0010$$

Nota 3.2. Si S es una palabra especial, entonces $S \equiv X^{\#}q_jY$, donde X y Y son s-palabras positivas, y así $S^* \equiv (X^{\#}q_jY)^* \equiv (X^{\#})^{\#}q_jY \equiv Xq_jY$ es también una palabra positiva; por consiguiente S^* determina un elemento del semigrupo Γ.

Lema 3.8. *Sea L una palabra sobre $\{r, x\} \subset \mathcal{B}$. Existe L' otra palabra en $\{r, x\}$ tal que $Ls_\beta = s_\beta L'$ para cualquier símbolo s_β.*

Demostración. Toda palabra sobre $\{r_i, x\}$ es una sucesión finita de elementos de la forma $r^k x^l$ con $k, l \geq 0$. Sin pérdida de generalidad se puede suponer que $L \equiv r^k x^l$. La demostración se hará por inducción sobre m la longitud de L.

El resultado se cumple para $m = 2$. Si $k = 2$ y $l = 0$, entonces $L \equiv r^2x^0 = r^2$ y así la relación (3.10) indica que

$$r^2 s_\beta = r(rs_\beta) = r(s_\beta xrx) = (rs_\beta)xrx = (s_\beta xrx)xrx = s_\beta(xrx^2rx)$$

y existe $L' = xrx^2rx \in W(\{r, x\})$ tal que $Ls_\beta = r^2 s_\beta = s_\beta L'$. De igual manera, si $L \equiv r^0x^2 = x^2$ la relación (3.9) asegura las igualdades

$$x^2 s_\beta = x(xs_\beta) = x(s_\beta x^2) = (xs_\beta)x^2 = s_\beta(x^2x^2) = s_\beta x^4$$

y la existencia de $L' = x^4$ tal que $Ls_\beta = x^2 s_\beta = s_\beta x^4 = s_\beta L'$. En el último caso para $m = 2$, es decir $L \equiv rx$, se tienen las siguientes igualdades:

$$\begin{aligned} r(xs_\beta) &= r(s_\beta x^2) \quad \text{por la relación (3.9)} \\ &= (rs_\beta)x^2 \\ &= (s_\beta xrx)x^2 \quad \text{por la relación (3.10)} \\ &= s_\beta(xrx^3). \end{aligned}$$

Por lo tanto existe $L' = xrx^3 \in W(\{r, x\})$ de tal manera que

$$Ls_\beta = (rx)s_\beta = s_\beta(xrx^3) = s_\beta L'.$$

Ahora supóngase cierto el resultado para palabras de longitud $m > 2$. Sea $L \equiv L_1x \in W(\{r, x\})$, una palabra de longitud $m + 1$, es decir L_1 tiene longitud m. Entonces

$$\begin{aligned} Ls_\beta &= (L_1x)s_\beta \\ &= L_1(xs_\beta) \\ &= L_1(s_\beta x^2) \quad \text{por la relación (3.9)} \\ &= (L_1s_\beta)x^2 \\ &= (s_\beta L')x^2 \quad \text{hipótesis inductiva} \\ &= s_\beta(L'x^2) \end{aligned}$$

y por consiguiente existe $L'' = L_1x^2 \in W(\{r, x\})$ tal que $Ls_\beta = s_\beta L''$. Finalmente, en el otro caso, si $L \equiv L_1 r$, la relación (3.10) y la hipótesis inductiva nos aseguran la existencia de $L_2 = L'(xrx)$ una palabra sobre $\{r, x\}$ tal que $Ls_\beta = s_\beta L_2$ y la prueba concluye. □

La prueba que sigue es válida para cualquier máquina de Turing T con estado de detención q_0. Utilizamos la notación $\mathcal{B}$ para el grupo $\mathcal{B}(T)$ y Γ para $\Gamma(T)$.

Lema 3.9. *(i) Si V es una s-palabra positiva, entonces*

$$r_iV = VR \text{ en } \mathcal{B} \quad \text{y} \quad r_i^{-1}V = VR' \text{ en } \mathcal{B}$$

donde R y R' son palabras sobre $\{r_i, x\}$ y R es positiva.

(ii) Si U es una s-palabra positiva, entonces

$$U^{\#}r_i = LU^{\#} \text{ en } \mathcal{B} \quad \text{y} \quad U^{\#}r_i^{-1} = L'U^{\#} \text{ en } \mathcal{B}$$

donde L y L' son palabras sobre $\{r_i, x\}$.

Demostración. (i) Se probará que $r_iV = VR$ en $\mathcal{B}$ por inducción sobre la longitud $m \geq 0$, donde $V \equiv s_{\beta_1} \cdots s_{\beta_m}$. El resultado es cierto para $m = 0$, dado que V resulta ser la palabra vacía, la palabra buscada $R = r_i$ es positiva y se cumple

$$r_iV = r_i1 = r_i = 1r_i = VR$$

Supóngase cierto el resultado para palabras de longitud $m > 0$. Sea $V \equiv V's_{\beta_{m+1}}$ una palabra de longitud $m + 1$ con V' una s-palabra de longitud m, por hipótesis inductiva se tiene

$$r_iV \equiv r_i(V's_{\beta_{m+1}}) = (r_iV')s_{\beta_{m+1}} = (V'R_1)s_{\beta_{m+1}} \tag{3.17}$$

donde R_1 es una palabra positiva sobre el conjunto $\{r_i, x\}$. Por el Lema (3.8) existe una palabra $R \in W(\{r_i, x\})$ que cumple la igualdad

$$R_1s_{\beta_{m+1}} = s_{\beta_{m+1}}R.$$

Finalmente, la última igualdad de la ecuación (3.17) se convierte en

$$V'(R_1s_{\beta_{m+1}}) = V'(s_{\beta_{m+1}}R) = (V's_{\beta_{m+1}})R = VR$$

de tal manera que se obtiene $r_iV = VR$, como se buscaba.

(ii) Se probará que $U^{\#}r_i^{-1} = L'U^{\#}$ en $\mathcal{B}$ por inducción sobre la longitud $m \geq 0$, donde $U \equiv s_{\beta_1} \cdots s_{\beta_m}$ y por lo tanto $U^{\#} \equiv s_{\beta_1}^{-1} \cdots s_{\beta_m}^{-1}$. El resultado se cumple para $m = 0$, con $U^{\#}$ la palabra vacía $U = 1 = U^{\#}$ y $L' = r_i^{-1}$ la palabra sobre el conjunto $\{r_i, x\}$ que satisface

$$U^{\#}r_i^{-1} = 1^{-1}r_i^{-1} = 1r_i^{-1} = r_i^{-1}1 = r_i^{-1}1^{-1} = L'U^{\#}$$

Supóngase cierto el resultado para palabras de longitud $m > 0$. Sea $U^{\#} = U_0^{\#}s_{\beta_{m+1}}^{-1}$ una palabra de longitud $m+1$ con $U_0^{\#}$ una s-palabra de longitud m

$$\begin{aligned}
U^{\#}r_i^{-1} &= (U_0^{\#}s_{\beta_{m+1}}^{-1})r_i^{-1} \\
&= U_0^{\#}(s_{\beta_{m+1}}^{-1}r_i^{-1}) = U_0^{\#}(r_i^{-1}x^0 s_{\beta_{m+1}}^{-1}x^0) \\
&= U_0^{\#}(r_i^{-1}s_{\beta_{m+1}}^{-1}) \quad \text{por la relación (3.10)} \\
&= (U_0^{\#}r_i^{-1})s_{\beta_{m+1}}^{-1} \\
&= (L'U_0^{\#})s_{\beta_{m+1}}^{-1} \quad \text{hipótesis inductiva} \\
&= L'(U_0^{\#}s_{\beta_{m+1}}^{-1}) \\
&= L'U^{\#}
\end{aligned}$$

con L' una palabra sobre $\{r_i, x\}$.

□

Boone da un criterio para la existencia del algoritmo en el grupo asociado $\mathcal{B}(T)$ en términos de la existencia del mismo en el semigrupo asociado $\Gamma(T)$. El teorema de Markov-Post se reduce al siguiente lema:

Lema 3.10 (Boone)**.** *Sea T una máquina de Turing con estado de detención q_0 y semigrupo asociado $\Gamma = \Gamma(T)$ (reescrito como en el Corolario 3.7). Si S es una palabra especial, entonces*

$$k(S^{-1}tS) = (S^{-1}tS)k \text{ en } \mathcal{B} = \mathcal{B}(T)$$

si y sólo si $S^* = q$ en $\Gamma(T)$.

Demostración. Si $S \equiv X^{\#}q_jY$ es una palabra especial con $S^* \equiv Xq_jY = q$ en el semigrupo Γ, entonces existe una sucesión finita de operaciones elementales tales que

$$S^* \equiv w_1 \to w_2 \to \cdots w_n \equiv q \text{ en } \Gamma$$

donde, para cada ν, una de las palabras w_ν y $w_{\nu+1}$ tiene la forma $UF_iq_{i_1}G_iV$ con U y V s-palabras, y la otra tiene la forma $UH_iq_{i_2}K_iV$. Por el Lema (3.9), existe una ecuación en $\mathcal{B}$:

$$\begin{aligned} U^{\#}(H_i^{\#} q_{i_2} K_i)V &= U^{\#}(r_i^{-1} F_i^{\#} q_{i_1} G_i r_i)V \quad \text{por la relación 3.11} \\ &= (U^{\#} r_i^{-1}) F_i^{\#} q_{i_1} G_i (r_i V) \quad \text{asociatividad} \\ &= (L' U^{\#}) F_i^{\#} q_{i_1} G_i (V R') \quad \text{por el Lema 3.9} \\ &= L' U^{\#} (F_i^{\#} q_{i_1} G_i) V R' \quad \text{asociatividad} \end{aligned}$$

donde L' y R' son palabras en $\{r_i, x\}$. De manera similar, vemos que existen palabras L'' y R'' sobre $\{r_i, x\}$ tal que

$$\begin{aligned} U^{\#}(F_i^{\#} q_{i_1} G_i)V &= U^{\#}(r_i H_i^{\#} q_{i_2} K_i r_i^{-1})V \quad \text{relación 3.11} \\ &= (U^{\#} r_i) H_i^{\#} q_{i_2} K_i (r_i^{-1} V) \quad \text{asociatividad} \\ &= (L'' U^{\#}) H_i^{\#} q_{i_2} K_i (V R'') \quad \text{por el Lema 3.9} \\ &= L'' U^{\#} (H_i^{\#} q_{i_2} K_i) V R'' \quad \text{asociatividad} \end{aligned}$$

Como $w_\nu = w_{\nu+1}$ en Γ entonces

$$w_\nu^* = (F_i q_{i_1} G_i)^* = (H_i q_{i_2} K_i) = w_{\nu+1}^*$$

en $\mathcal{B}$, de las primeras tres relaciones, se sigue que para cada ν, se tiene

$$w_\nu^* = L_\nu w_{\nu+1}^* R_\nu \text{ en } \mathcal{B}$$

para las palabras L_ν y R_ν sobre algunas r_i y x. Las palabras $L \equiv L_1 \dots L_{n-1}$ y $R \equiv R_{n-1} \dots R_1$ son palabras sobre $\{x, r_i, i \in I\}$, y

$$w_1^* = L_\nu w_n^* R \text{ en } \mathcal{B}.$$

Pero $w_1^* \equiv (S^*)^* \equiv S$ y $w_n^* \equiv q^* \equiv q$, así que

$$S = LqR \text{ en } \mathcal{B}.$$

Como los generadores t y k conmutan con x y todas las r_i, también conmutan con L y R. Entonces, sustituyendo S se obtiene

$$\begin{aligned} kS^{-1}tSk^{-1}S^{-1}t^{-1}S &= k(R^{-1}q^{-1}L^{-1})t(LqR)k^{-1}(R^{-1}q^{-1}L^{-1})t^{-1}(LqR) \\ &= kR^{-1}q^{-1}tqk^{-1}q^{-1}t^{-1}qR \quad \text{conmutatividad de } L, R \text{ con } r, s \\ &= R^{-1}(kq^{-1}tqk^{-1}q^{-1}t^{-1}q)R \quad \text{conmutatividad de } R \text{ con } k \\ &= R^{-1}kq^{-1}tq(q^{-1}tqk)^{-1}R \quad \text{asociatividad} \\ &= R^{-1}1R \quad \text{relación 3.16} \\ &= 1 \end{aligned}$$

□

3.3.2 El teorema de Novikov-Boone-Britton

Teorema 3.11 (**Novikov-Boone-Britton**). *Existe un grupo finitamente presentado $\mathcal{B}$ donde el problema de la palabra que no tiene solución.*

Demostración. Sea T la máquina de Turing T^* del Teorema de Markov-Post (3.6). Si existiera un algoritmo que determine, para una palabra especial arbitraria S, si $kS^{-1}tSk^{-1}S^{-1}t^{-1}S = 1$ en $\mathcal{B}(T^*)$, entonces existe un algoritmo para determinar $S^* = q$ en $\Gamma(T^*)$. Pero el segundo inciso del Corolario 3.7 asegura que no existe tal algoritmo en $\Gamma(T^*)$. □

Corolario 3.12. *Sea T una máquina de Turing con estado de detención q_0 que enumera un subconjunto E de Ω (el conjunto de todas las palabras positivas del alfabeto de T). Si $w \in \Omega$, entonces $w \in E$ si y sólo si $k(h^{-1}q_1wh) = (h^{-1}q_1wh)k$ en $\mathcal{B}(T)$.*

Demostración. Por el Lema 3.5, $w \in E$ si y sólo si $hq_iwh = q$ en $\Gamma(T)$. Pero en $\mathcal{B}(T), (hq_1wh)^* = h^{-1}q_1wh$ (la cual es una palabra especial), y el Lema de Boone muestra que $(h^{-1}q_1wh)^{\#} \equiv hq_1wh = q$ en $\Gamma(T)$ si y sólo si $k(h^{-1}q_1wh) = (h^{-1}q_1wh)k$ en $\mathcal{B}(T)$. □

Con el Teorema de Novikov-Boone-Britton (3.11) se muestra la existencia de un grupo finitamente presentado $\mathcal{B}$ donde el Problema de la Palabra no tiene solución, la construcción de $\mathcal{B}$ es a partir del semigrupo finitamente presentado γ (Teorema de Markov-Post 3.6).

También se puede construir un grupo finitamente presentado $\mathcal{B}$ y mostrar que si en $\mathcal{B}$ el Problema de la Palabra tiene solución, entonces el Problema en γ tendría solución, para su demostración puede consultar el capítulo 12 de [7].

Conclusiones

La principal aplicación del Teorema de Markov-Post es al problema de Dehn o Problema de la palabra. Del capítulo 3 se deduce que este problema tiene solución en los siguientes grupos:

i) Grupos libres con presentación vacía (Teorema 3.1).

ii) Grupos finitos (Corolario 3.2).

Existen grupos y semigrupos finitamente presentados donde tal algoritmo no existe. Sin embargo, la demostración de su existencia es indirecta. El problema se reduce al Problema de la detención (sección 3.1) que es indecidible.

Primero se asocia un semigrupo finitamente presentado Γ a la máquina de Turing T^* del Teorema (2.8), que tiene asociado el conjunto $E = e(T^*) \subset \mathbb{N}$, que es recursivamente enumerable pero no es recursivo. El Teorema de Markov-Post (3.6) demuestra que en este semigrupo no existe algoritmo para determinar si dos palabras son iguales.

Utilizando el trabajo de Markov y Post, se asocia un grupo finitamente presentado $\mathcal{B}$ a la máquina de Turing T^* y se da un criterio para la existencia del algoritmo solución del problema en $\mathcal{B}$ en términos de la existencia del mismo en Γ (Lema 3.10). Finalmente, el Teorema (3.11) muestra el grupo finitamente presentado donde el problema no tiene solución.

Apéndice A

Teoría de Grupos

En este apartado se escriben algunos teoremas importantes que son mencionados en el presente trabajo. La teoría y demostración puede ser consultada en [2], [13] y principalmente en [7].

Grupos y subgrupos

Proposición A.1. *Sea $(G, *)$ un grupo.*

i) El neutro es único: existe un único $e \in G$ tal que para todo $a \in G$

$$e * a = a = a * e.$$

ii) El inverso es único: existe un único $a^{-1} \in G$ tal que para todo $a \in G$ $a * a^{-1} = e = a^{-1} * a$ y además

$$(a^{-1})^{-1} = a.$$

Proposición A.2. *Sea $f : G \to H$ un morfismo de grupos.*

i) $f(e_G) = e_H$, donde e_G y e_H es el neutro de G y H respectivamente.

ii) Si $a \in G$, entonces $f(a^{-1}) = f(a)^{-1}$.

iii) Si $a \in G$ y $n \in \mathbb{Z}$, entonces $f(a^n) = f(a)^n$.

Definición A.1. Sea $S \leq G$ un subgrupo de G y $t \in G$

i) El subconjunto $tS = \{ts : s \in S\}$ es llamado *clase lateral izquierda* de G.

ii) El subconjunto $St = \{st : s \in S\}$ es llamado *clase lateral derecha* de G.

iii) Al elemento t se le llama *representante* de la clase St.

Dadas dos clases laterales, es natural preguntarse cuándo éstas son iguales.

Lema A.3. *Sea G un grupo y $S \leq G$. Entonces*

i) $Sa = Sb$ *si y sólo si* $ab^{-1} \in S$.

ii) $aS = bS$ *si y sólo si* $b^{-1}a \in S$.

Teorema A.4. *Sea $S \leq G$. El conjunto de clases laterales (izquierda o derecha) forman una partición de G:*

i) $Sa \neq \emptyset$ *para cualquier* $a \in G$.

ii) Dadas dos clases Sa y Sb, entonces $Sa = Sb$ ó $Sa \cap Sb = \emptyset$.

iii) G es la unión disjunta de las clases laterales de S, es decir, $G = \biguplus_{a \in G} Sa$.

Proposición A.5. *Sea $S \leq G$, entonces el número de clases laterales derechas de S en G es igual al número de clases laterales izquierdas de S en G.*

El número de clases laterales izquierdas y derechas coinciden, ésto permite definir el índice de S en G.

Definición A.2. Sea $S \leq G$, entonces el *índice* de S en G, denotado por $[G : S]$, es el número de clases laterales derechas de S en G.

Si el grupo G es finito, existe una relación entre el órden del subgrupo, el órden del grupo y el número de clases laterales.

Teorema A.6 (Teorema de Lagrange)**.** *Sea G un grupo finito y $S \leq G$, entonces el órden de S divide al órden de G, es decir, $|S|$ divide a $|G|$ y además $[G : S] = |G|/|S|$.*

Teoremas de Isomorfismos

Definición A.3. Un subgrupo $N \leq G$ es un *subgrupo normal*, denotado por $N \lhd G$, si $gN = Ng$ para todo $g \in G$.

Si un subgrupo N de G es normal, el conjunto de todas las clases laterales puede definirse una operación entre ellas y ser un grupo.

Teorema A.7 (Grupo cociente)**.** *Sea $(G, \cdot)$ un grupo y $N \lhd G$ un subgrupo normal. Entonces el conjunto de las clases laterales $G/N = \{gN : g \in G\}$ forma un grupo con la operación $* : G/N \times G/N \to G/N$ definida como*

$$aN * bN := (a \cdot b)N.$$

Proposición A.8. *Sean G y H grupos, $f : G \to H$ un morfismo. Entonces:*

i) $Nucf = \{x \in G \mid f(x) \in H\}$ *es un subgrupo de* G.

ii) $Imf = \{f(x) \mid x \in G\}$ *es un subgrupo de* H.

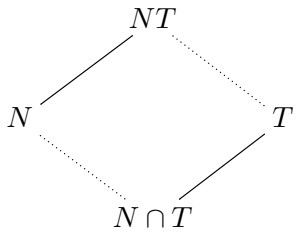
Teorema A.9 (Primer Teorema de Isomorfismo)**.** *Sea $f : G \to H$ un morfismo de grupos con núcleo N. Entonces N es un subgrupo normal de G y $G/N \cong Imf$.*

La unión de subgrupos no necesariamente es un subgrupo, sin embargo, si uno de ellos es normal, la unión sí lo es.

Lema A.10. *Sean S y T subgrupos de G y si uno de ellos es normal, entonces $ST = \langle T \cup S \rangle = TS$, donde $ST = \{st : s \in S, t \in T\}$*

Teorema A.11 (Segundo Teorema de Isomorfismo)**.** *Sea N y T subgrupos de G con N normal. Entonces $N \cap T$ es normal en T y $T/(N \cap T) \cong NT/N$.*

El siguiente diagrama es una mnemotecnia para este teorema:



Teorema A.12 (Tercer Teorema de Isomorfismo)**.** *Sean $H, K \leq G$ subgrupos de G tales que $K \leq H \leq G$ y donde ambos, K y H, son subgrupos normales de G. Entonces H/K es un subgrupo normal de G/K y además*

$$(G/K)/(H/K) \cong G/H.$$

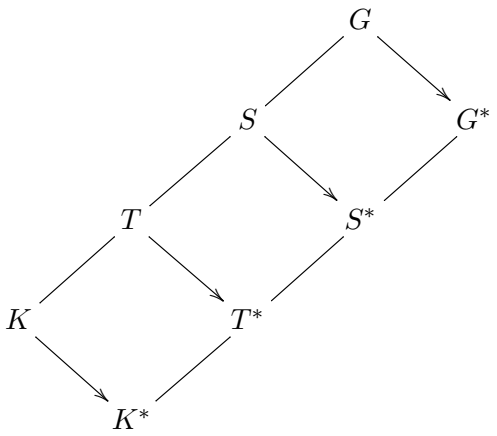
Teorema A.13 (Teorema de la Correspondencia)**.** *Sea $K \lhd G$ y sea $\nu : G \to G/K$ el epimorfismo natural. Entonces $S \mapsto \nu(S) := S/K$ es una biyección entre la familia de de todos los subgrupos S de G que contienen a K y la familia todos los subgrupos de G/K.*

Además, si denotamos S/K por S^*, entonces:

i) $T \leq S$ *si y sólo si* $T^* \leq S^*$ *y* $[S : T] = [S^* : T^*]$.

ii) $T \lhd S$ *si y sólo si* $T^* \lhd S^*$ *y* $S/T \cong S^*/T^*$.

Un diagrama mnemotécnico para este teorema es:



donde $K^* = K/K = \{0\}$ y $G^* = G/K$.

Grupos Cíclicos

Un grupo G es cíclico si existe $x \in G$ tal que $G = \langle x \rangle$. Si G está denotado multiplicativamente, entonces $G = \langle x \rangle = \{x^n : n \in \mathbb{Z}\}$. Si la operación de G está denotada de manera aditiva, entonces $G = \langle x \rangle = \{nx : n\mathbb{Z}\}$.

Teorema A.14. *Toodo grupo cíclico es abeliano.*

Teorema A.15. *Todo subgrupo de un grupo cíclico es cíclico.*

Lo siguiente es una caracterización de los grupos cíclicos finitos.

Teorema A.16. *Un grupo G de orden n es cíclico si y sólo si, para cada divisor d de n, existe a lo más un subgrupo cíclico de G de órden d.*

Recuerde que $(\mathbb{Z}, +)$ y $(\mathbb{Z}_n, +_n)$ son grupos cíclicos, el primero es infinito y el segundo es finito. La siguiente afirmación caracteriza los grupos cíclicos.

Teorema A.17. *Sea $(G, *)$ un grupo cíclico. Entonces:*

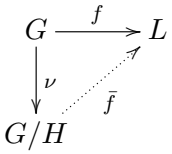
i) Si G es de orden infinito, entonces $(G, *) \cong (\mathbb{Z}, +)$.

ii) Si G es de orden finito n, entonces $(G, *) \cong (\mathbb{Z}_n, +_n)$.

Otros resultados importantes

Teorema A.18 (Teorema del factor)**.** *Sea $f : G \to L$ un morfismo de grupos, $H \lhd G$ tal que $H \subseteq Nuc(f)$ y $\nu : G \to G/H$ el epimorfismo canónico.*

Entonces existe un único morfismo $\bar{f} : G/H \to L$ *que cumple* $\bar{f}\nu = f$*, es decir el siguiente diagrama conmuta:*



Además se satisface:

i) $\bar{f}$ *es un epimorfismo si y sólo si* f *es epimorfismo.*

ii) $\bar{f}$ *es inyectiva si y sólo si* $Nuc(f) = H$*.*

En el grupo de permutaciones se tiene el siguiente resultado importante.

Teorema A.19 (Teorema de Cayley)**.** *Todo grupo* G *puede ser incluído como un subgrupo de* S_G*. En particular, si* $|G| = n$*, entonces* G *se puede incluir como un subgrupo en* S_n*.*

Grupos Abelianos Libres

Lema A.20. *Sea* $\{A_k : k \in K\}$ *una familia de subgrupos de un grupo* G*. Las siguientes afirmaciones son equivalentes:*

i) $G \cong \sum_{k\in K} A_k$*.*

ii) Todo $g \in G$ *tiene una única expresión de la forma*

$$g = \sum_{k\in K} a_k$$

donde $a_k \in A_k$*, los* k *son distintos y* $a_k \neq 0$ *para un conjunto finito de* $k's$*.*

Definición A.4. Un subconjunto finito $X = \{x_1, \ldots, x_n\}$ de elementos distintos de cero de un grupo G es *independiente* si, para todo $m_i \in \mathbb{Z}$, $\sum m_i x_i = 0$ implica $m_i x_i = 0$ para todo i. Un conjunto infinito X de elementos distintos de cero en G es *independiente* si todo subconjunto finito es independiente.

Definición A.5. Un grupo abeliano L es *libre* si es una suma directa de grupos cíclicos infinitos. Más precisamente, si existe un subconjunto $X \subset L$ de elementos de órden infinito, llamado *base* de L, tal que

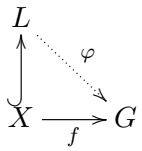
$$L = \sum_{x\in X} \langle x \rangle, \quad \text{es decir} \quad L \cong \sum \mathbb{Z}.$$

Si X es una base de un grupo abeliano L, entonces todo $u \in L$ tiene una única expresión de la forma $u = \sum m_x x$, donde $m_x \in \mathbb{Z}$ y $m_x = 0$ para casi todo $x \in X$; esto es, $m_x \neq 0$ para un número finito de $x's$.

Teorema A.21. *Sea L un grupo abeliano libre con base X, G un grupo abeliano arbitrario y $f : X \to G$ cualquier función. Existe un único morfismo $\varphi : L \to G$ que extiende a f; esto es,*

$$\varphi(x) = f(x) \quad \text{para todo } x \in X.$$

Además, si $u = \sum m_x x \in L$, entonces $\varphi(u) = \sum m_x f(u)$.



Corolario A.22. *Todo grupo abeliano G es un cociente de un grupo abeliano libre.*

Teorema A.23. *Dos grupos abelianos libres $L = \sum_{x \in X} \langle x \rangle$ y $K = \sum_{y \in Y} \langle y \rangle$, con bases X y Y respectivamente, son isomorfos si y sólo si $|X| = |Y|$.*

Definición A.6. El *rango* de un grupo abeliano libre es la cardinalidad de su base.

Bibliografía

[1] Cooper, S. B. *Computability Theory.* Chapman & Hall/CRC, 2004.

[2] Fraleigh, J. B. *A First Course in Abstract Algebra*, 7th ed. Addison Wesley, 2003.

[3] Hopcorft, J. E., and Ullman, J. D. *Introducción a la teoría de Autómatas, Lenguajes y Computación*, primera ed. Edit. Continental, 1993.

[4] Linz, P. *An Introduction to Formal Language and Automata*, 3rd ed. Jones & Bartlett Publishers, 2001.

[5] Mendelson, E. *Introduction to Mathematical Logic*, 3rd ed. Cole Mathematics. The Wadsworth & Brooks, 1987.

[6] Robinson, D. J. S. *A Course in Theory of Groups*, 2nd ed. Graduate Text in Mathematics. Springer, 1995.

[7] Rotman, J. J. *An Introduction to the Theory of Groups*, 4th ed. Graduate Text in Mathematics. Springer, 1995.

[8] Rotman, J. J. *Advanced Modern Algebra*, 2nd ed. Prentice-Hall, 2002.

[9] Sipser, M. *Introduction to the Theory of Computation*, 3rd ed. Cengage Learning, 2013.

[10] Strathern, P. *Turing y la computadora.* Científicos en 90 minutos. Siglo XXI Editores, 2014.

[11] Suárez-Alvarez, M. Grupos libres, Marzo 2007.

[12] Viso, E. *Introducción a la Teoría de la Computación*, primera ed. Las prensas de Ciencias, 2008.

[13] Zaldivar, F. *Introducción a la Teoría de Grupos.* Aportaciones Matemáticas. Sociedad Matemática Mexicana, 2006.