Capítulo 1

Teoría de lenguajes.

En esta sección vamos a introducir los elementos básicos de la teoría de lenguajes formales que utilizaremos en este trabajo.

Consideremos un conjunto no vacío Σ que llamaremos el alfabeto y Σ^k el conjunto de sucesiones finitas de elementos $a_1 \dots a_k$ con $a_i \in \Sigma$. Los elementos de Σ se llaman letras y los elementos de Σ^k serán palabras de longitud k sobre Σ. La palabra vacía que corresponde a Σ^0 la denotaremos por λ.

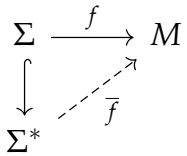
Si w es una palabra sobre el alfabeto Σ luego una subpalabra u de w es una palabra $u \in \Sigma^*$ tal que $w = vuz$ para algunas $v, z \in \Sigma^*$. Si $w = vu$ entonces v es un prefijo de w y u es un subfijo de w.

Definición 1.0.1. El monoide libre sobre un alfabeto Σ es el siguiente conjunto

$$\Sigma^* = \bigcup_{k=0}^{\infty} \Sigma^k$$

con la operación $\cdot$ que es la concatenación de palabras. Es decir dadas $w_1 \in \Sigma^k, w_2 \in \Sigma^l$ luego $w_1 \cdot w_2 \in \Sigma^{k+l} \subset \Sigma^*$. El elemento neutro es la palabra vacía que corresponde a la copia de Σ^0 que es la única palabra sin letras.

Observación 1.0.2. El monoide es libre con la siguiente propiedad: si tenemos una función del alfabeto $f : \Sigma \to M$ donde M es algún monoide entonces existe un único morfismo de monoides $\overline{f} : \Sigma^* \to M$ que hace conmutar al siguiente diagrama.



Definición 1.0.3. Un lenguaje sobre un alfabeto Σ es un subconjunto de Σ^*.

1.1. Gramáticas.

Vamos a considerar lenguajes definidos a partir de lo que se conoce como una gramática. Esto es esencialmente un conjunto de reglas que al irse aplicando nos permiten generar todas las palabras del lenguaje.

Definición 1.1.1. Una gramática es una tupla $\mathcal{G} = (V, \Sigma, P, S)$ donde:

- V es un conjunto finito de *variables*;
- $S \in V$ es el *símbolo inicial*;
- Σ es un conjunto finito de *símbolos terminales* que lo tomamos disjunto de V;
- $P \subseteq (V \cup \Sigma)^* V (V \cup \Sigma)^* \times (V \cup \Sigma)^*$ es un conjunto finito de *producciones*.

Dada una gramática $\mathcal{G} = (V, \Sigma, P, S)$ a cualquiera de sus producciones $(\gamma, \nu) \in P$, la vamos a denotar $\gamma \to \nu$.

A partir de una gramática $\mathcal{G}$ podemos definirnos una relación sobre las cadenas $(\Sigma \cup V)^*$. Dados $x, y \in (\Sigma \cup V)^*$ diremos que x *deriva* en y si existen $u, v, w, z \in (\Sigma \cup V)^*$ tales que $x = uwv$ y tenemos una producción $w \to z \in P$ de manera que $y = uzv$. La notación que usaremos es $x \Rightarrow_{\mathcal{G}} y$. Consideremos la clausura transitiva y reflexiva de esta relación que denotaremos por $\overset{*}{\Rightarrow}_{\mathcal{G}}$.

Definición 1.1.2. El lenguaje generado por la gramática van a ser las palabras en Σ^* que se pueden derivar del símbolo inicial S. Formalmente esto es

$$L(\mathcal{G}) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow}_{\mathcal{G}} w\}.$$

Ejemplo 1.1.3. Consideremos la siguiente gramática $\mathcal{G} = (V, \Sigma, P, S)$ donde $V = \{S, A\}, \Sigma = \{a, b\}$ y tenemos las siguientes producciones,

$$\begin{aligned} S &\to Ab \\ A &\to aA \\ A &\to \lambda \end{aligned}$$

Veamos como podemos derivar la palabra a^2b usando las producciones de esta gramática. Esto es que $S \overset{*}{\Rightarrow}_{\mathcal{G}} a^2b$. Tomamos la siguiente sucesión:

$$S \to Ab \to aAb \to aaAb \to aab$$

y nos queda tal como queríamos ver.

Más aún probemos que $L(\mathcal{G}) = \{a^k b : k \geq 0\}$.

Si $w \in L(\mathcal{G})$ entonces $S \overset{*}{\Rightarrow}_{\mathcal{G}} w$ por definición. La única producción que la gramática tiene donde S está a la izquierda es $S \to Ab$. De esta manera cualquier palabra $w \in L(\mathcal{G})$ va a tener una b como postfijo de la palabra. La variable A vemos que solo puede derivar en $a^k A$ o a^k para $k \geq 0$. Esto se debe a que podemos aplicar la producción $A \to aA$ tantas veces como querramos por lo tanto $A \overset{*}{\Rightarrow}_{\mathcal{G}} a^k$ para cualquier $k \geq 0$. Juntando con lo anterior vemos que $S \overset{*}{\Rightarrow}_{\mathcal{G}} a^k b$ así que terminamos de ver que la gramática genera al lenguaje $\{a^k b : k \geq 0\}$ tal como queríamos ver.

Es posible clasificar los lenguajes a partir de las características de las gramáticas que los generan.

1.2. Lenguajes regulares.

Definición 1.2.1. Decimos que una gramática $\mathcal{G} = (V, \Sigma, P, S)$ es regular si las producciones son del estilo

1. $A \to \lambda$
2. $A \to a$
3. $A \to aB$

donde $A, B \in V$, $a \in \Sigma$ y λ es la palabra vacía. Si $L = L(\mathcal{G})$ para alguna gramática regular $\mathcal{G}$ entonces diremos que L es un lenguaje regular.

En particular la gramática del ejemplo 1.1.3 es regular. De esta manera $L = \{a^k b : k \geq 0\}$ resulta ser un lenguaje regular.

Definición 1.2.2. Un autómata finito no determinístico es una tupla $\mathcal{M} = (Q, \{q_0\}, \Sigma, \delta, F)$ donde:

- Q es el conjunto de los *estados*.
- $q_0 \in Q$ es el *estado inicial*.
- Σ es el *alfabeto*.
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ es la *función de transición*.
- $F \subseteq Q$ es un subconjunto de estados que llamaremos *finales*.

A los automátas finitos los interpretamos como grafos con algunas reglas para movernos sobre ellos. Para empezar Q representan los vértices de nuestro grafo.

Ejemplo 1.2.3. Construyamos un automáta $\mathcal{M}$ tal que acepte al lenguaje $L = \{a^k b \ : \ k \geq 0\}$.

Como vimos en este ejemplo el lenguaje $L = \{a^k b \ : \ k \geq 0\}$ es aceptado por un automáta no determinístico finito y por lo que sabíamos del ejemplo 1.1.3 es un lenguaje regular. Más aún vale que los lenguajes aceptados por automátas finitos no determinísticos son justamente los regulares.

Teorema 1.2.4. *Un lenguaje L es regular sii es aceptado por un autómata finito no determinístico.*

Demostración. Demostración estándar. Ver [**?**]. □

1.3. Lenguajes independientes de contexto.

Definición 1.3.1. Decimos que una gramática $\mathcal{G} = (V, \Sigma, P, S)$ es independiente de contexto si las producciones tienen la siguiente forma:

$$A \to w$$

donde $A \in V, w \in (\Sigma \cup V)^*$. Si $L = L(\mathcal{G})$ para alguna gramática independiente de contexto $\mathcal{G}$ entonces diremos que L es un lenguaje independiente de contexto.

Ejemplo 1.3.2. Sea el alfabeto $\Sigma = \{a, b\}$. Si $w = a_1 \dots a_k$ es una palabra sobre Σ^* entonces podemos considerar a w^r que es la palabra inversa dada por leerla de derecha a izquierda y tiene la siguiente pinta $w^r = a_k \dots a_1$. Consideremos sobre Σ el lenguaje $L = \{w \in \Sigma^* \mid w = w^r\}$ tal que este es el lenguaje de los palíndromos. Construyamos una gramática independiente de contexto para este lenguaje. Sea w una palabra en L luego sabemos que si su longitud es mayor que uno entonces debe ser que $w = aua$ o $w = bub$ para cierta palabra $u \in L$ dado que u necesariamente tiene que ser un palíndromo porque w lo es. Esto sucede para todas las palabras del lenguaje exceptuando las palabras de longitud uno que son justamente las letras del alfabeto. De esta manera podemos considerar la siguiente gramática $\mathcal{G} = (\{S\}, \Sigma, P, S)$ donde las producciones P están dadas por :

$$\begin{aligned} S &\to \epsilon \\ S &\to a \\ S &\to b \\ S &\to aSa \\ S &\to bSb. \end{aligned}$$

Para ver que esta gramática genera al lenguaje L notemos que si tomamos una palabra en el lenguaje $w \in L$ luego si no es una letra al ser palíndromo sobre el alfabeto Σ necesariamente debe comenzar con a o con b y de esta manera tenemos que la primer derivación es $S \to aSa$ o $S \to bSb$. Dado que la subpalabra $w = aua$ que se obtiene de w sin considerar la primera y última letra es un palíndromo (e incluso podría ser la palabra vacía) luego podemos repetir este proceso para llegar a $S \overset{*}{\Rightarrow}_{\mathcal{G}} w$ después de finitos pasos. Por otro lado toda palabra generada por esta gramática es un palíndromo porque todas las reglas son tales que agregan una letra al principio y la misma al final o simplemente agregan letras que también son palíndromos.

Toda gramática independiente de contexto la podemos tomar para que sea de una forma en particular.

Definición 1.3.3. Una gramática $\mathcal{G} = (V, \Sigma, P, S)$ independiente de contexto está en su forma normal de Chomsky si las producciones son de este tipo:

1. $A \to BC$ donde $A \in V$ y $B, C \in V \setminus \{S\}$.
2. $A \to a$ donde $A \in V, a \in \Sigma$.
3. $S \to \lambda$

Ejemplo 1.3.4.

Proposición 1.3.5. *Para toda gramática $\mathcal{G}$ independiente de contexto puede tomar otra $\mathcal{G}'$ tal que esté en forma normal de Chomsky y generen el mismo lenguaje. Esto es que $L(\mathcal{G}) = L(\mathcal{G}')$,*

Demostración. Demostración estándar. Ver [**?**]. □

1.3.1. Autómatas de pila.

Así como las gramáticas nos permiten generar un lenguaje tenemos las máquinas que nos permiten aceptar un lenguaje. En nuestro caso en particular vamos a usar autómatas de pila no determinísticos para aceptar los lenguajes independientes de contexto.

Definición 1.3.6. Un autómata de pila finito no determinístico es una tupla $\mathcal{M} = (Q, \Sigma, Z, \delta, q_0, F, Z_0)$ donde:

- Q es un conjunto finito de *estados*;
- Σ es un conjunto finito que denotaremos el *alfabeto del lenguaje*;
- Γ es un conjunto finito que denotaremos el *alfabeto de la pila*;

- δ es la *función de transición* donde $\delta : Q \times \Sigma \times \Gamma \to \mathcal{P}(Q \times \Gamma^*)$;
- $F \subseteq Q$ es el conjunto de *estados finales*;
- $q_0 \in Q$ es el *estado inicial*;
- $Z_0 \in Z$ es el *símbolo inicial* en la pila.

Funcionamiento del automáta. El autómata de pila finito funciona de la siguiente manera. Dada una palabra $w \in \Sigma$ queremos saber si es aceptada por el autómata de pila o no. Para eso vamos a ir leyendo esta palabra de izquierda a derecha. Al comenzar a leer esta palabra estamos en el estado q_0 que distinguimos como el estado inicial de nuestro autómata. Nos fijamos en la función de transición que es una función parcial cuánto nos da evaluada $\delta(\lambda, q_0, \gamma)$ para algún γ prefijo de w y donde estamos mirando a $\lambda \in Z^*$ el elemento neutro de este monoide. Esto se corresponde a la idea de que al comenzar nuestra pila está vacía. En tal caso nuestra función de transición nos da un resultado que es un par (z, q) donde $z \in Z^*$ es lo que nos va a quedar en la pila y q es el nuevo estado al cual nos movimos. Notemos que nuestra función de transición no tiene porqué tener un (z, q) tal que podamos movernos o podría ser bien que tenga más de uno. En este caso diremos que nuestro automáta es no determinístico dado que en algunos casos existe más de una opción.

En general estamos en la siguiente situación en el proceso de aceptar la palabra w. Tenemos algún $z \in Z$ en el tope de la pila que al ser una pila lo leeremos de derecha a izquierda, estamos en algún estado $q \in Q$ y nos quedará una subpalabra γ de w para leer. Una configuración de nuestro autómata entonces es una manera de describir en que situación de aceptar o no aceptar una palabra y la denotamos (z, q, γ).

Cuando hayamos visto toda la palabra o no tengamos manera de movernos de estado nos fijamos si el estado p en el que estamos es final, es decir si $p \in F$. En tal caso la palabra w es aceptada por el automáta. Formalmente estaremos en alguna configuración zq para $z \in Z^*$ y $q \in Q$ y no nos queda nada de la palabra w porque ya la consumimos toda.

Observación 1.3.7. Así como definimos el automáta de pila no determinístico para que al tener la pila vacía se detenga podríamos haberlo hecho de una manera distinta por ejemplo dejando que la pila sea vacía y así poder transicionar de una configuración a otra. Esta manera es equivalente porque... Más en adelante nos va a resultar conceptualmente más útil para el caso que estemos viendo el problema de la palabra de un grupo mientras que para las demostraciones de esta sección usaremos esta otra definición.

Descripción instántanea del autómata. Veamos ahora como describir formalmente el funcionamiento de un automáta a partir de lo que estamos haciendo en cierto instante. Consideremos que estamos en el instante que nuestra subpalabra que nos queda por leer es w, estamos en un estado q y en nuestra pila tenemos la palabra γ, entonces vamos a representar al instante por medio de esta tupla (q, w, γ). Si ahora tenemos la posibilidad de movernos a otro estado p tal que $(p, w', \alpha\beta) \in \delta(q, aw', x\beta)$ donde $aw' = w$ con a alguna letra posiblemente vacía y similarmente $x\beta = \gamma$ con $x \in \Gamma^*$ una letra de Γ posiblemente vacía. Este movimiento lo denotamos como $(q, aw', x\beta) \vdash (p, w', \alpha\beta)$. Podemos considerar la clausura transitiva de esta relación sobre los triples $Z \times Q \times \Sigma^*$ que denotaremos $\vdash^*$.

El lenguaje aceptado por un autómata de pila. Notemos que en particular el automáta de pila nos da un lenguaje que está formado por las palabras w en el alfabeto de la entrada del automáta Σ que son aceptadas. En general diremos que un automáta acepta un lenguaje L si su lenguaje aceptado es exactamente L. Este lenguaje aceptado por el automáta $\mathcal{M}$ en algunas casos para hacer énfasis en el automáta lo denotaremos $L(\mathcal{M})$ y siguiendo la notación formal anterior lo podemos describir de la siguiente manera,

$$L(\mathcal{M}) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \epsilon, \gamma),\ \ q \in F,\ \gamma \in \Gamma^*\}$$

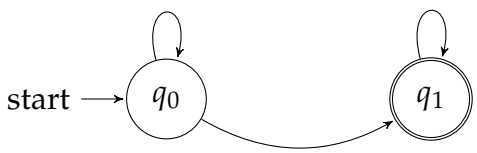
donde (q_0, w, Z_0) es el instante inicial en el cual tenemos la palabra w en el estado inicial q_0 con el símbolo Z_0 de la pila. Al finalizar deberíamos estar en un estado q final y lo que nos queda en la pila γ es totalmente irrelevante en este caso.

Observación 1.3.8. Al ser un automáta de pila no determinístico existen posiblemente más de una manera de consumir alguna palabra en el automáta. Es así que por la definición que dimos la palabra es aceptada por el automáta si al menos alguna de estas derivaciones la lleva a ser aceptada. No importa si existen derivaciones que no lo hagan mientras una sí lo haga.

Ejemplo 1.3.9. Sea nuestro alfabeto $\Sigma = \{a, b\}$ y L el lenguaje de los palíndromos sobre este alfabeto. Consideremos el siguiente autómata de pila

$$M = (Q, \{q_0\}, \{a, b\}, \{a, b, \$\}, \$, \{q_1\})$$

donde nuestra pila tiene el mismo alfabeto que el de entrada con un símbolo extra que es \$ que va a ser nuestro símbolo inicial de la pila.



El automáta tiene dos estados, el inicial y el final. La idea es que en el primer estado vamos apilando la palabra y en la segunda vamos desapilando la palabra anteriormente apilada. Nuestra función de transición va a ser la siguiente,

- $\delta(q_0, a, Z) = (q_0, aZ)$
- $\delta(q_0, b, Z) = (q_0, ba)$
- $\delta(q_1, a, a) = (q_0, \lambda)$.
- $\delta(q_1, b, b) = (q_0, \lambda)$.
- $\delta(q_0, \lambda, Z) = (q_1, Z)$
- $\delta(q_0, X, Z) = (q_1, Z)$

donde $Z = a, b, \$$ es decir cualquier cosa del alfabeto de la pila y $X = a, b$ cualquier elemento de nuestro alfabeto de entrada. El automáta en el primer estado apila lo que sea que estemos leyendo sin importar lo que esté en el tope de pila. En el segundo estado desapila cada vez que lo que estemos leyendo coincida con el tope de pila. Finalmente para ir del estado inicial al final tenemos en cuenta dos casos. Nos podemos mover por λ es decir sin consumir ninguna letra de la palabra de la entrada o leyendo alguna de las letras de la palabra. Estos casos se corresponden a que el palíndromo tenga longitud par o tenga longitud impar.

Este ejemplo nos da un indicio que los lenguajes aceptados por automátas de pila también podrían ser independiente de contexto y esto efectivamente es cierto.

Teorema 1.3.10. *Un lenguaje L es independiente de contexto sii es aceptado por un automáta de pila no determinístico.*

Demostración. Ver **[?]**. □

Hasta ahora definimos los automátas de pila no determinísticos que aceptan por estado final. Otra definición posible de lenguaje aceptado podría ser que acepten por pila vacía. Es decir que una vez que consumimos la palabra w de entrada llegamos a una configuración (q, λ, λ) donde λ es la palabra vacía de ambos alfabetos respectivamente. Formalmente notaremos al lenguaje aceptado por pila vacía por un automáta $\mathcal{M}$ de la siguiente manera,

$$L(\mathcal{M}) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \lambda, \lambda),\ q \in Q, u \in \Sigma^*\}$$

donde arrancamos en el estado inicial y llegamos a algún estado q cualesquiera y nuestra pila está vacía así como lo que nos queda por leer de la palabra. En este

caso obtenemos que nuestro lenguaje es aceptado por un automáta de pila no determinístico por pila vacía.

El siguiente resultado nos dice que en el caso que nuestro automáta sea no determinístico es equivalente usar una u otra manera de definir a nuestro lenguaje.

Teorema 1.3.11. *Un lenguaje L es aceptado por un automáta de pila no determinístico por estado final sii es aceptado por un automáta de pila no determinístico por pila vacía.*

Demostración. Ver [**?**]. □

1.3.2. Propiedades de los lenguajes independientes de contexto.

En esta sección vamos a considerar a los lenguajes independiente de contexto como una clase. Probemos algunas propiedades que cumplen respecto a las operaciones de conjuntos más usuales.

Proposición 1.3.12. *Los lenguajes independiente de contexto son cerrados por uniones.*

Demostración. Ver [**?**]. □

Proposición 1.3.13. *Los lenguajes independientes de contexto no son cerrados por intersecciones.*

Demostración. Ver [**?**]. □

Lo que si sucede es que son cerrados con respecto a intersecciones con lenguajes regulares.

Proposición 1.3.14. *Los lenguajes independiente de contexto son cerrados por intersecciones con lenguajes regulares.*

Demostración. Ver [**?**]. □

Otras propiedades interesantes tienen que ver con su relación con morfismos de monoides.

Proposición 1.3.15. *Los lenguajes independiente de contexto son cerrados por:*

1. *Imágenes de morfismos de monoides.*
2. *Preimágenes de morfismos de monoides.*

Demostración. Ver [**?**]. □

Finalmente la herramienta principal que tenemos para ver que cierto lenguaje L no es independiente de contexto es usar el siguiente lema.

Lema 1.3.16 (Pumping). *Sea L un lenguaje independiente de contexto entonces existe una constante $n \geq 0$ tal que para todas las palabras $w \in L$ de longitud al menos n existe una factorización $w = uvxwy$ con $|vwx| \leq n$ y $|vx| > 0$ tal que para todo $i \in \mathbb{N}$ vale que $uv^i wx^i y \in L$.*

Demostración. Resultado estándar. Ver [**?**]. □

1.4. Automátas de pila determinísticos.

Si en la definición anterior del automáta pedimos que la función de transición de una configuración dada tenga a lo sumo un valor entonces nuestro automáta lo vamos a llamar determinístico. Formalmente esto es que

$$|\delta(q, a, z)| \leq 1 \quad \forall z \in Z,\ p \in Q,\ a \in \Sigma \cup \{\lambda\}.$$

En cierta manera estamos diciendo que de un instante dado solo tenemos a lo sumo una única posibilidad de movernos a otro estado. A los lenguajes aceptados por un automáta de pila determinístico los pensamos que son aceptados por un estado final.

Por lo tanto para cada palabra tenemos un único camino en el automáta para saber si es aceptada o no a diferencia de un automáta de pila no determinístico que podría tener varios caminos posibles para cada palabra.

Observación 1.4.1. Todo automáta de pila determinístico en particular es no determinístico y por lo tanto la clase de lenguajes aceptados por los primeros están contenidos en la clase de los segundos.

Veamos que esta contención es estricta. Para eso volvamos a considerar el ejemplo del lenguaje de los palíndromos.

Ejemplo 1.4.2. El lenguaje $L = \{w \in \{a, b\}^* \ : \ w = w^r\}$ no es aceptado por un automáta de pila determinístico pero sí por uno no determinístico. Supongamos que M es un automáta de pila determinístico que lo acepta. Notemos que para cualquier palabra $w \in \{a, b\}^*$ debe ser que al consumirla la pila no puede quedar vacía dado que $ww^r \in L$ y en tal caso no aceptaría a esta palabra. Esto es que la pila nunca está vacía sea cual sea la configuración que lleguemos. Para cada palabra arbitraria w existe otra x_w tal que al procesar wx_w lo que nos queda en la pila es de tamaño mínimo con respecto a todas las palabras wx. Sea entonces lo que tiene en la pila la palabra α_w que sabemos es de longitud mínima. Si consideramos palabras del estilo $wx_w z$ sabemos que la longitud de lo que quede en la pila no puede disminuir. Ahora consideremos dos palabras del estilo $r = wx_w, s = uy_u$ tales que sus pilas son de longitud mínima al terminar de recorrer las palabras y que resultan tener el mismo tope de pila y terminar en el mismo estado. Podemos asegurar la existencia de estas palabras debido a que tenemos finitos estados y combinaciones de tope de pilas dado

que el automáta de pila es finito pero tenemos infinitas palabras que cumplen esta propiedad. Ahora basta con elegir z de modo que tz sea palíndromo pero que sz no lo sea.

Veamos que podemos elegir a z para que una de las concatenaciones tz, sz sea palíndromo y la otra no. Partamos en distintos casos. Si $|t| = |s|$ basta con tomar $z = s^r$. Si $|t| \neq |s|$ y supongamos que s tiene longitud menor y no es prefijo de t entonces de nuevo podemos tomar el palíndromo ss^r tal que ts^r no es un palíndromo. Finalmente queda el caso que una es un prefijo de la otra, supongamos $t = su$. Si elegimos $x = a, b$ tal que ux no sea un palíndromo luego la palabra ss^r es un palíndromo pero $suxs^r$ no lo es.

Esto muestra que si bien $sz \notin L$ y $tz \in L$ el automáta de pila determinístico no va a poder diferenciarlas por lo tanto no es posible que este lenguaje sea aceptado por un automáta de pila determinístico tal como queríamos ver.

1.4.1. Automátas de pila determinísticos especiales.

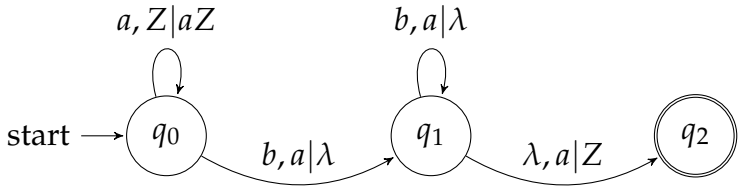
Consideremos ahora un automáta de pila determinístico tal que acepta tanto por estado final como por pila vacía. A estos los llamaremos *automátas de pila determinístico especiales*. Estos automátas son los que nos surgen de la construcción del automáta del problema de la palabra para grupos virtualmente libres.

Ejemplo 1.4.3. Sea el lenguaje $L = \{a^m b^n : m \geq n \geq 1\}$ este no es un lenguaje independiente de contexto determinístico especial pero sí es determinístico.

Construyamos un automáta de pila determinístico que acepte a L. Sea

$$M = (\{q_0, q_1, q_2\}, \{q_0\}, \{a, b\}, \{a, b, Z_0\}, Z_0, q_2)$$

el siguiente automáta de pila determinístico que representamos así:



donde Z es cualquier letra del alfabeto de la pila. El automáta en el estado inicial q_0 apila a todas las a y cambia al estado q_1 cuando lee por primera vez una b y en ese caso desapila la a que está en el tope de la pila. En el estado q_1 sigue desapilando cada vez que ve una b. Finalmente va al estado q_2 cuando en la pila sigue quedando a y ya leímos toda la palabra y en tal caso la acepta porque significa que vimos como máximo tantas b como a y este es el lenguaje que buscábamos generar.

El lenguaje no es aceptado por un automáta de pila determinístico por pila vacía dado que tiene la propiedad de los prefijos. Es decir que existen palabras que están

en el lenguaje tales que alguno de sus prefijos también están. Por ejemplo consideremos $a^m b^i$ y $a^m b^j$ para $m \geq 2$ e $i < j \leq m$. Esto es porque si $M = (Q, \Sigma, Z, Z_0, \delta, q_0, F)$ fuera un automáta de pila determinístico que acepta por pila vacía a este lenguaje tendríamos que $(q_0, a^m b^i, Z_0) \vdash^* (q, \lambda, \lambda)$ con q un estado final pero como tiene la pila vacía no podemos continuar aceptando a la palabra $a^m b^j$ ya que por la definición que empleamos el autómata necesita leer algún elemento de la pila. De esta manera vemos que este lenguaje no puede ser aceptado por pila vacía y estado final por un automáta de pila determinístico concluyendo que los lenguajes determinísticos especiales forman un subfamilia propia de los independientes de contexto determinísticos.

1.5. Lenguajes poly independientes de contexto.

En esta sección vamos a introducir otra familia de lenguajes que generalizan levemente a los lenguajes independientes de contexto. La bibliografía fundamental es [**?**].

1.6. Conos de lenguajes y el problema de la palabra.

En esta sección consideraremos un grupo G finitamente generado por algún alfabeto Σ finito donde lo vemos como un conjunto de generadores como monoide dado que en particular todos los grupos son monoides. De esta manera tenemos un epimorfismo de monoides $\pi : \Sigma^* \twoheadrightarrow G$.

El problema de la palabra es uno de los problemas de teoría de grupos más centrales al área. Explícitamente el problema consiste en dada una palabra $\omega \in \Sigma^*$ en los generadores del grupo encontrar un algoritmo para decidir si esta palabra es la identidad del grupo o no. Notemos que para poder pensar este problema estamos fijando de antemano algún conjunto de generadores posible del grupo.

Dado un grupo finitamente presentado consideramos el siguiente lenguaje

$$\mathrm{WP}(G, \Sigma) = \{\omega \in \Sigma^* \mid \omega \underset{G}{=} 1\}$$

que llamaremos el problema de la palabra de G para los generadores Σ.

Queremos ver sobre qué clases de lenguajes el problema de la palabra queda bien definido y no depende de los generadores elegidos.

Así como definimos los lenguajes regulares vamos a preocuparnos por otros tipos siempre y cuando cumplan las siguientes condiciones.

Definición 1.6.1. Una clase de lenguajes $\mathbb{C}$ es un cono si para todo $L \in \mathbb{C}$ resulta que:

- Es cerrado por imágenes de morfismos de monoides. Sea $L \subset \Sigma^*$ luego si existe $\phi : \Sigma^* \to \Delta^*$ morfismo de monoides debe ser que $\phi(L) \in \mathbb{C}$.
- Es cerrado por preimagenes de morfismos de monoides. Sea $L \subset \Sigma^*$ luego si existe $\phi : \Sigma^* \to \Delta^*$ morfismo de monoides debe ser que $\phi^{-1}(L) \in \mathbb{C}$.
- Es cerrado por intersecciones con lenguajes regulares. Si R es un lenguaje regular sobre Σ^* entonces $L \cap R \in \mathbb{C}$ también resulta serlo.

Ejemplo 1.6.2. Los lenguajes independientes de contexto forman un cono. Esto se puede ver a partir de las proposiciones 1.3.14 y 1.3.15.

Los conos de lenguajes cumplen la siguiente propiedad de gran importancia para el estudio del problema de la palabra.

Proposición 1.6.3. *Sea $WP(G, \Sigma)$ el lenguaje del problema de la palabra de cierto grupo G para algunos generadores Σ y $\mathbb{C}$ cono de lenguajes. Si $WP(G, \Sigma) \in \mathbb{C}$ luego valen las siguientes afirmaciones:*

- *$WP(G, \Delta) \in \mathbb{C}$ para cualquier conjunto de generadores Δ.*
- *$WP(H) \in \mathbb{C}$ para todo subgrupo H grupo finitamente generado de G.*

Demostración.

- Formamos el siguiente diagrama conmutativo,

$$\begin{array}{ccc} \Delta^* & \xrightarrow{\delta} & G \\ {\scriptstyle f}\downarrow & \nearrow_{\pi} & \\ \Sigma^* & & \end{array}$$

  donde f es algún morfismo de monoides y donde usamos la propiedad universal de los monoides libres. Notemos que $WP(G, \Delta) = \delta^{-1}(1)$ y como el diagrama conmuta tenemos que

$$f^{-1}(\delta^{-1}(1)) = f^{-1}(WP(G, \Sigma)) = WP(G, \Delta).$$

  Dado que esto es un cono obtenemos lo que queríamos ver puesto que es cerrado por preimágenes de morfismos de monoides.

- Sea Σ' conjunto de generadores de H. Siempre podemos extenderlo a Σ tal que Σ genere a G. De esta manera

$$WP(H, \Sigma') = WP(H, \Sigma) \cap \Sigma'^*$$

  y de vuelta como es un cono la intersección con lenguajes regulares nos da un lenguaje en el cono.

□

Esto nos dice que es interesante estudiar el problema de la palabra justamente sobre conos.

1.7. Sistemas de reescritura.

Un *sistema de reescritura* podemos pensarlo como un grafo en el sentido de Serre. En este caso los vértices son los *objetos* mientras que las aristas las llamamos *movimientos*. Si nuestro sistema de reescritura Γ tiene un movimiento de un objeto a en otro objeto b diremos que a puede ser *reescrito* a b y lo denotaremos $a \to_\Gamma b$, omitiendo aclarar que estamos considerando el sistema Γ en todo caso que no sea ambiguo. Un camino en el grafo lo llamaremos una *derivación*.

Nosotros queremos que estos sistemas de reescritura sean tales que si tomamos una sucesión de movimientos en algún momento se estabilice y en tal caso llamaremos *terminante.* A su vez los objetos tales que no puedan ser modificados por ningún movimiento llamaremos objetos *terminales.*

Dado un sistema de reescritura Γ por medio de la clausura transitiva que denotaremos $\overset{*}{\to}_\Gamma$ obtenemos una relación transitiva sobre los objetos. Nuestro interés justamente va a estar en estudiar las relaciones de equivalencia que surgen de sistemas de reescritura particulares.

Ejemplo 1.7.1. Considerar como objetos los enteros y como movimiento dividir al número por dos si es posible. Este sistema de reescritura tiene como objetos terminales los números impares.

Otra característica que le vamos a pedir a los sistema de reescritura es que sean *confluentes*. Informalmente esto es que a partir de un objeto, si usamos dos derivaciones distintas entonces eventualmente estas derivaciones se encuentran en algún objeto independientemente de qué derivaciones tomemos. Esto es que dadas derivaciones $a \overset{*}{\to} b, a \overset{*}{\to} c$ existe un objeto d tal que $b \overset{*}{\to} d$ y $c \overset{*}{\to} d$.

En particular los sistemas que nos van a interesar en este trabajo son sistemas tales que son confluentes y terminantes que son conocidos como *Church-Roser*.

1.7.1. Sistemas de reescritura de cadenas.

En nuestro caso en particular vamos a trabajar con sistemas de reescritura donde los objetos son palabras sobre algún alfabeto Σ y los movimientos son reglas del estilo $w \to v$ que interpretamos de la siguiente manera. Si tenemos alguna palabra que tenga como subpalabra a w podemos modificarla por v. Esto lo representamos por el triple $(p, w \to v, q)$ donde p, q son las subpalabras que vienen antes de w tales que pueden ser vacías.

Al sistema de reescritura Γ con alfabeto Σ y las reglas $\mathcal{R}$ lo vamos a denotar por $\mathrm{sr}\langle\Sigma, \mathcal{R}\rangle$. Estos sistemas también son llamados *sistemas de Thue*.

1.8. Sistemas de reescritura.

$$S \to R, \;\; R = \{r_1, \ldots, r_n\}$$