Lenguajes Formales, Autómatas y Computabilidad

BAS

Resumen de la materia Lenguajes Formales, Autómatas y Computabilidad dictada en el segundo cuatrimestre de 2024, pensado principalmente para uso personal y orientado a practicar para el final

Fecha de compilación: 29 / 01 / 2025.

Este resumen esta basado en la bibliografía y clases de la materia dictadas en el segundo cuatrimestre de 2024, así como también clases de cuatrimestres pasados de las materias (**F**) de Teoría de Lenguajes y Lógica y Computabilidad

El template de este resumen es esencialmente el mismo usado por @valn y como soy un sinvergüenza lo pedí prestado sin avisar. Si quieren usar un resumen como guía de estudio, recomiendo que usen el suyo antes que el mío.

Link al resumen original:
(https://gitlab.com/valn/uba/-/tree/main/Lenguajes%20Formales%2C%20Aut%C3%B3matas%20y%20Computabilidad/Final?ref_type=heads°)

Índice

1. Lenguajes y Gramáticas

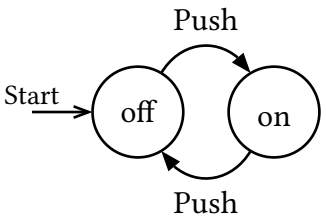
1.1. Por qué nos interesa la teoría de autómatas?

Los autómatas finitos son un modelo útil para varios tipos de software y hardware. Por ejemplo:

- Software para la verificación del comportamiento de circuitos digitales, así como de sistemas con una cantidad finita de estados (protocolos de comunicación, entre otros)
- Los analizadores léxicos de los compiladores
- Software para escanear grandes cuerpos de texto

1.2. Pero entonces, qué son autómatas? (informalmente)

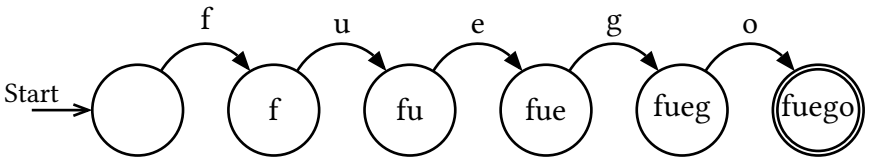
Antes de discutir la definición formal de estos modelos, consideremos primero como un autómata finito se ve y qué es lo que hace.



Acá tenemos un autómata que simula un interruptor de encendido / apagado. el dispositivo recuerda si está en estado «prendido» o «apagado», y permite presionar un botón cuyo efecto va a ser diferente dependiendo del estado en el que se encuentre en ese momento (representado por los arcos saliendo de los estados). Notar la etiqueta «Start», la cual indica el estado inicial del autómata

Hay muchos sistemas que pueden ser vistos de una manera similar, es decir, que cumplen que en cualquier momento determinado se encuentran en uno de un número finito de estados. El propósito de estos estados es recordar la porción relevante de la historia del sistema, para poder actuar de acuerdo a ésta.

Otro ejemplo de un autómata puede ser un analizador léxico. Por ejemplo, un autómata que sólo reconozca la palabra «fuego» podría estar dado por:



Como queremos que sea capaz de reconocer la palabra «fuego», el autómata precisa de 5 estados, cada uno representando una posición diferente de la palabra que ya fue alcanzada. Y los arcos representan un input de una letra Notar ahora el uso de un estado con doble círculos, el mismo denota un estado final, un estado especial que determina que el autómata aceptó el input dado

1.3. Los Conceptos Centrales de la Teoría de Autómatas

Ahora vamos a introducir definiciones esenciales para el estudio de autómatas: el **alfabeto**, las **cadenas**, y el **lenguaje**

1.3.1. Alfabetos

Definición 1.3.1.1: Un *alfabeto* es conjunto finito no vacío de símbolos, que por convención denotamos Σ.

Ejemplo: $\Sigma = \{0, 1\}$. El alfabeto binario

Ejemplo: $\Sigma = \{a, b, ..., z\}$. El alfabeto de todas las letras minúsculas

1.3.2. Cadenas

Definición 1.3.2.1: Una *cadena* (también conocida como palabra) es una secuencia finita de símbolos pertenecientes a un mismo alfabeto

Ejemplo: La cadena 0110 formada por símbolos pertenecientes al alfabeto binario

Definición 1.3.2.2: Nos referimos con *longitud* de una cadena a la cantidad de símbolos en la misma, y denotamos la longitud de la cadena w usando |w|

1.3.2.1. Potencias de un alfabeto

Definición 1.3.2.1.1: Si Σ es un alfabeto, podemos denotar al conjunto de todas las cadenas pertenecientes al alfabeto de cierto longitud usando notación exponencial. Definimos Σ^k como el conjunto de cadenas de longitud k, con todos sus símbolos pertenecientes a Σ

Ejemplo: Para el alfabeto binario, $\Sigma^1 = \{0, 1\}, \Sigma^2 = \{00, 01, 10, 11\}$

Ejemplo: Notar que $\Sigma^0 = \{\lambda\}$, sin importar el alfabeto. Es la única cadena de longitud 0

Definición 1.3.2.1.2: Usamos la notación Σ^* para referirnos al conjunto de todas las cadenas de un alfabeto, y lo denominamos **clausura de Kleene**, y usamos Σ^+ para referirnos a su clausura positiva (es decir, que no incluya la palabra vacía), se definen como:

$$\Sigma^* = \bigcup_{i \geq 0} \Sigma^i \tag{1}$$

$$\Sigma^+ = \bigcup_{i \geq 1} \Sigma^i \tag{2}$$

1.3.3. Hay tantas palabras como números naturales

Teorema 1.3.3.1: $|\Sigma^*|$ es igual a la cardinalidad de $\mathbb{N}$

Definición 1.3.3.1: Definimos el orden lexicográfico $\prec \subset \Sigma^* \times \Sigma^*$. Asumimos el orden lexicográfico entre los elementos de un mimso alfabeto, y luego lo extendemos a un orden lexicográfico entre palabras de una misma longitud. De esta manera, las palabras de menor longitud son menores en este sentido que las de mayor longitud

Demostración: Definimos una biyección $f : \mathbb{N} \to \Sigma^*$ tal que $f(i) =$ la i-ésima palabra en el orden $\prec$ sobre Σ^*. Luego, como tenemos una biyección entre los conjuntos Σ^* y $\mathbb{N}$, tenemos que necesariamente deben tener la misma cardinalidad □

1.4. Lenguajes

Definición 1.4.1: Un lenguaje L sobre un alfabeto Σ es un conjunto de palabras pertenecientes a Σ. Es decir, $L \subseteq \Sigma^*$

Ejemplo:
- $\emptyset$, el lenguaje vacío, es un lenguaje para cualquier alfabeto
- $\{\lambda\}$, el lenguaje que consiste sólo de la cadena vacía, que también es un lenguaje sobre cualquier alfabeto. (Notar que $\emptyset \neq \{\lambda\}$)
- El lenguaje de todas las cadenas formadas por n 1s seguidos por n 0s sobre $\sigma = \{0, 1\}$. Algunas cadenas de este lenguaje son: {λ, 10, 1100, 111000, ...}

La única restricción importante en cuanto qué puede ser un lenguaje, es que el alfabeto del mismo siempre es finito

Definición 1.4.2: Sea A un conjunto, $\mathcal{P}(A)$ es el conjunto de todos los subconjuntos de A, $\mathcal{P}(A) = \{B \subseteq A\}$. Si tenemos que A es un conjunto finito, entonces $\mathcal{P}(A) = 2^{|A|}$

Teorema 1.4.1: $|\Sigma^*| < \mathcal{P}(\Sigma^*)$, es decir, la cantidad de lenguajes sobre un alfabeto Σ no es numerable

Demostración: Supongamos que $\mathcal{P}(\Sigma^*)$ es numerable, entonces tenemos que podemos enlistar los lengajes y para cada lenguaje L_i (es decir cada lenguaje $\subset \mathcal{P}(\Sigma^*)$), podemos ordenar las palabras pertenecientes al mismo según el orden lexicográfico definido anteriormente de la siguiente manera:
L_1: $w_{1,1}, w_{1,2}, w_{1,3}, ...$
L_2: $w_{2,1}, w_{2,2}, w_{2,3}, ...$
...
Ahora consideremos el lenguaje $L = \{u_1, u_2, u_3, ...\}$ tal que, para todo i, se cumpla que $w_{i,i} \prec u_i$. Pero entonces tenemos un lenguaje cuyo iésimo elemento de L siempre es «mayor» que el iésimo elemento de L_i, entonces no puede ser que L pertenezca a nuestra lista de lenguajes. Pero entonces tenemos un lenguaje sobre Σ^* que no estaba incluido en nuestra ennumeración de todos los lenguajes, por lo que tenemos un absurdo. El mismo vino de suponer que $\mathcal{P}(\Sigma^*)$ era numerable.
□

1.5. Gramáticas

Definición 1.5.1: Una gramática es una 4-upla G = $\langle V_N, V_T, P, S\rangle$ donde:
- V_N es un conjunto de símbolos llamados no terminales
- V_T es un conjunto de símbolos terminales
- P es el conjunto de producciones, que es un conjunto finito de

$$\left[(V_N \cup V_T)^* \ V_N \ (V_N \cup V_T)^*\right] \times (V_N \cup V_T)^*$$

  donde las producciones son tuplas (a, b) y usualmente las notamos $a \to b$
- S es el símbolo distinguido o inicial de V_N

Ejemplo: Sea $G_{\text{arithm}} = \langle V_N, V_T, P, S\rangle$ una gramática libre de contexto (esto último lo introducimos más tarde) tal que:
- $V_N = \{S\}$
- $V_T = \{+, *, a, (,)\}$
- Y producciones determinadas por:
  $S \to S + S$,
  $S \to S * S$,
  $S \to a$,
  $S \to (S)$

1.5.1. Lenguaje generado por una gramática

Definición 1.5.1.1: Dada una gramática G = $\langle V_N, V_T, P, S\rangle$, definimos a $\mathcal{L}(G)$ como:

$$\mathcal{L}(G) = \left\{w \ \in V_T^* : S \overset{+}{\underset{G}{\Rightarrow}} w\right\}$$

donde $\overset{+}{\underset{G}{\Rightarrow}}$ es la derivación en uno o más pasos, que se obtiene de la clausura transitiva de $\underset{G}{\Rightarrow}$ (la derivación directa, cuya definición se da en un momento)

1.5.2. Forma sentencial de una gramática

Definición 1.5.2.1: Sea G = $\langle V_N, V_T, P, S\rangle$.
- S es una forma sentencia de G
- Si $\alpha\beta\gamma$ es una forma sentencial de G, y tenemos que $(\beta \to \delta) \in P$, entonces $\alpha\delta\gamma$ es también una forma sentencial de G

Las formas sentenciales pertenecen a $(V_N \cup V_T)^*$

1.5.3. Derivación directa de una gramática

Definición 1.5.3.1: Si $\alpha\beta\gamma \in (V_N \cup V_T)^*$ y $(\beta \to \delta) \in P$, entonces $\alpha\beta\gamma$ deriva directamente en G a $\alpha\delta\gamma$ y lo denotamos:

$$\alpha\beta\gamma \underset{G}{\Rightarrow} \alpha\delta\gamma$$

Observemos que como la derivación directa es una relación sobre $(V_N \cup V_T)$, podemos componerla consigo misma 0 o más veces:

- $\left(\underset{G}{\Rightarrow}\right)^0 = \mathrm{id}_{(V_N \cup V_T)^*}$
- $\left(\underset{G}{\Rightarrow}\right)^+ = \bigcup_{k=1}^{\infty} \left(\underset{G}{\Rightarrow}\right)^k$
- $\left(\underset{G}{\Rightarrow}\right)^* = \left(\underset{G}{\Rightarrow}\right)^+ \cup \mathrm{id}_{(V_N \cup V_T)^*}$

Según como elijamos derivar, podemos considerar:

- Derivación más a la izquierda: $\underset{L}{\Rightarrow}$
- Derivación más a la derecha: $\underset{R}{\Rightarrow}$

Así como también sus derivaciones en uno o más pasos, y sus calusuras transitivas y de Kleene

1.5.4. La Jerarquía de Chomsky

Definición 1.5.4.1: La jerarquía de Chomsky clasifica las gramáticas en 4 tipos en una jerarquía tal que puedan generar lenguajes cada vez más complejos, y donde cada uno de las gramáticas puede generar también los lenguajes de una de complejidad inferior. Las clasificaciones son:

- **Gramáticas de tipo 0 (o sin restricciones)**:
  $\alpha \to \beta$
- **Gramáticas de tipo 1 (o sensibles al contexto)**:

$\alpha \to \beta$, con $|\alpha| \leq |\beta|$

- **Gramáticas de tipo 2 (o libres de contexto)**:

A $\to \gamma$ con A $\in V_N$

- **Gramáticas de tipo 3 (o regulares)**:

$A \to a, A \to aB, A \to \lambda$, con $A, B \in V_N, a \in V_T$

La jerarquía de grmáticas da origen también a la jerarquía de lenguajes:

- **Recursivamente enumerables**
- **Sensitivos al contexto**
- **Libres de contexto**
- **Regulares**

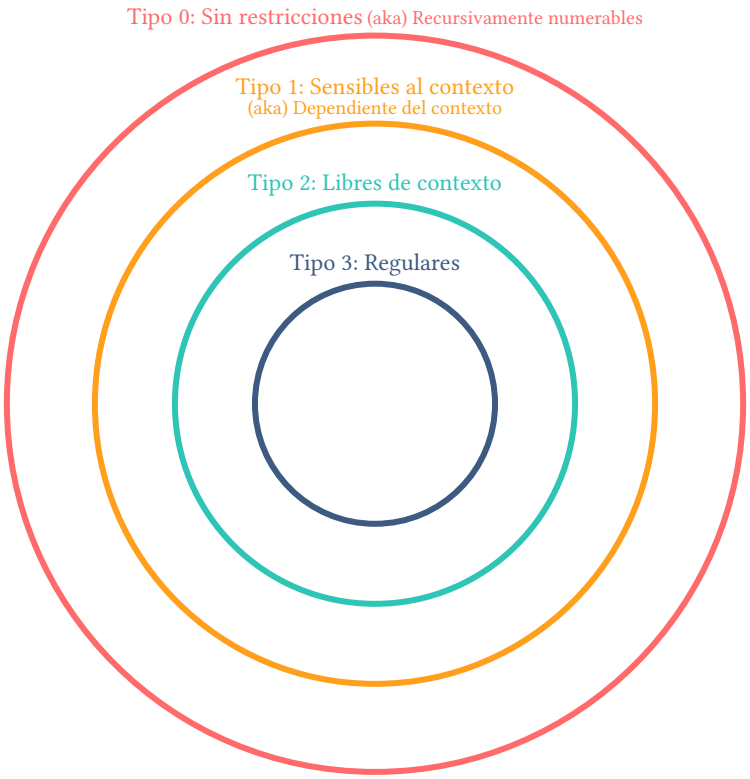


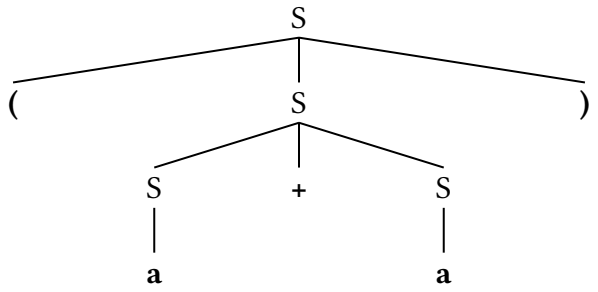
Figura 1: Imagen perteneciente al repo de @Valn (No sé que haces acá todavía a ser honesto).

1.5.5. Árbol de derivación de gramáticas

Definición 1.5.5.1: Un árbol de derivación es una representación gráfica de una derivación (look at me I'm so smart) donde:

- Las etiquetas de las hojas están en $V_T \cup \{\lambda\}$
- Las etiquetas de los nodos internos están en V_N. Las etiquetas de sus símbolos son los símbolos del cuerpo de una producción
- Un nodo tiene etiqueta A y tiene n descendientes etiquetados $X_1, X_2, ..., X_N$ sii hay una derivación que usa una producción $A \to X_1 X_2 ... X_N$

Ejemplo: Considerande la gramática G_{arithm} definida anteriormente, un posible árbol de derivación podría ser el siguiente:

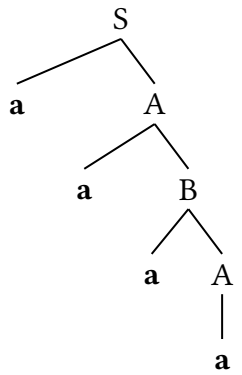


$$S \underset{L}{\Rightarrow} (S) \underset{L}{\Rightarrow} (S+S) \underset{L}{\Rightarrow} (a+S) \underset{L}{\Rightarrow} (a+S) \underset{L}{\Rightarrow} (a+a)$$

1.5.6. Algunos lemas relevantes

Lema 1.5.6.1: Sea $G = \langle V_N, V_T, P, S \rangle$ regular, no recursiva a izquierda. Si $A \overset{i}{\underset{L}{\Rightarrow}} wB$ entonces $i = |w|$

Demostración: Consideremos el árbol de derivación de la cadena w = $a_1 a_2 a_3 ... a_n$. Si entonces cortamos el árbol en una altura h determinada, para $h \geq 1$ se obtiene un subarbol de h hojas, con el único nodo que no es hoja con una etiqueta de V_N. Por la forma que las producciones de una gramática regular tiene, tenemos que cada derivación pone un símbolo no terminal a lo sumo y uno terminal , luego, tenemos que en n derivaciones (altura n del árbol) tenemos n hojas y el no terminal de la derecha □



Un posible árbol de derivación para las producciones:

$$S \to aA \mid \lambda$$
$$A \to a \mid aB$$
$$B \to aA$$

Lema 1.5.6.2: Sea G = $\langle V_N, V_T, P, S\rangle$ libre de contexto, no recursiva a izquierda (es decir, no tiene derivaciones $A \underset{L}{\overset{+}{\Rightarrow}} A\alpha$). Existe una constante c tal que si $A \underset{L}{\overset{i}{\Rightarrow}} \omega B\alpha$ entonces $i \leq c^{|\omega|+2}$

Demostración: Pendiente (soy ***vago***) □

2. Autómatas Finitos Determinísticos, no Determinísticos, y Gramáticas Regulares

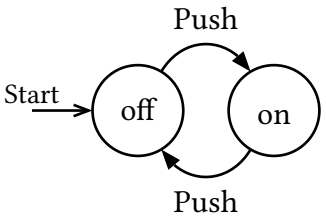
2.1. Autómata Finito Determinístico (AFD)

Definición 2.1.1: Un autómata finito determinístico es una 5-upla $\langle Q, \Sigma, \delta, q_0, F \rangle$ donde:

- Q es un conjunto finito de estados
- Σ es el alfabeto de entrada del autómata
- δ es la función de trancición del autómata que toma como argumentos un estado y un símbolo de input, y devuelve un estado, es decir: $\delta : Q \times \Sigma \to Q$.
- $q_0 \in Q$ es el estado inicial del autómata
- $F \subseteq Q$ es el conjunto de estados finales del autómata

Para determinar si un autómata acepta cierta cadena, se hace uso de su función de transición y se confirma si se termina llegando a un estado $q_f \in F$ final. Es decir, suponiendo que se busca. Como es poco práctico escribir la función δ en su totalidad, se suele hacer uso de diagramas de autómatas (como el usado al principio de este resumen) y se aceptan los arcos del mismo como una definición implícita.

Ejemplo: En nuestro primer autómata, teníamos (notar que «Push» es sólo un símbolo, no una cadena / palabra)



Luego, la 5-upla correspondiente sería $\langle \{\text{off}, \text{on}\}, \{\text{Push}\}, \delta, \text{off}, \emptyset \}$ con δ definido por:
$\delta(\text{off}, \text{Push}) = \text{on}$
$\delta(\text{on}, \text{Push}) = \text{off}$

2.1.1. Función de transición generalizada $\hat{\delta}$

Ahora necesitamos definir precisamente qué ocurre cuando estamos en un estado determinado y recibimos como input una cadena, no un símbolo. Para esto, lo definimos por inducción en el largo de la cadena

Definición 2.1.1.1: Definimos $\hat{\delta}$ como:

- $\hat{\delta}(q, \lambda) = q$
- $\hat{\delta}(q, \omega\alpha) = \delta\Big(\hat{\delta}(q, \omega), \alpha\Big)$ con $\omega \in \Sigma^* y \alpha \in \Sigma$

Cabe recalcar que $\hat{\delta}(q, \alpha) = \delta\Big(\hat{\delta}(q, \lambda), \alpha\Big) = \delta(q, \alpha)$ y que muchas veces vamos a hacer uso de la misma notación para ambas funciones

2.1.2. El Lenguaje de un AFD

Definición 2.1.2.1: Ahora podemos definir el lenguaje aceptado por un AFD como:

$$\mathcal{L}(M) = \left\{\omega \in \Sigma^* : \hat{\delta}(q_0, \omega) \in F\right\}$$

Es decir, el lenguaje de un autómata M es el conjunto de cadenas que mediante $\hat{\delta}$ llegan desde el estado inicial q_0 a un estado final. Llamamos **lenguajes regulares** a aquellos aceptados por un AFD

2.2. Autómatas Finitos no Determinísticos (AFND)

Definición 2.2.1: Un AFND es una 5-upla $\langle Q, \Sigma, \delta, q_0, F\rangle$ donde:

- Q es un conjunto finito de estados
- Σ es el alfabeto de entrada
- δ es la función de trancición, que toma un estado en Q y un símbolo en Σ, pero que ahora devuelve un subconjunto de Q. Es decir:

$$\delta : Q \times \Sigma \to \mathcal{P}(Q)$$

- $q_0 \in Q$ es el estado inicial
- $F \subseteq Q$ es el conjunto de estados finales

2.2.1. Función de trancisión generalizada $\hat{\delta}$

Definición 2.2.1.1: Definimos $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$:

- $\hat{\delta}(q, \lambda) = \{q\}$
- $\hat{\delta}(q, x\alpha) = \left\{p \in Q : \exists r \in \hat{\delta}(q, x) \;\; y \;\; p \in \delta(r, a)\right\}$
- Una definición alternativa, siendo $w = xa$, con $w, x \in \Sigma^* y \;\; a \in \Sigma$ y suponiendo que $\hat{\delta}(q, x) = \{p_1, ..., p_k\}$. Sea

  $\bigcup_{i=1}^{k} \delta(p_i, a) = \{r_1, ..., r_k\}$

  Entonces tenemos $\hat{\delta}(q, w) = \{r_1, ..., r_k\}$

Notar que:

$$\begin{aligned}\hat{\delta}(q, \lambda\alpha) &= \left\{p \in Q : \exists r \in \hat{\delta}(q, \lambda) \;\; y \;\; p \in \delta(r, a)\right\} \\ &= \{p \in Q : \exists r \in \{q\} \;\; y \;\; p \in \delta(r, a)\} \\ &= \{p \in Q : p \in \delta(q, a)\} \\ &= \delta(q, a)\end{aligned}$$

2.2.2. Lenguaje aceptado por un AFND

Definición 2.2.2.1: El lenguaje aceptado por un AFND M, $\mathcal{L}(M)$, es el conjunto de cadenas aceptadas por M y está definido como:

$$\mathcal{L}(M) = \left\{x \in \Sigma^* : \hat{\delta}(q_0, x) \cap F \neq \emptyset\right\}$$

2.2.3. Función de transición de conjuntos de estados

Definición 2.2.3.1: Podemos extender la función de transición aún más, haciendo que mapee conjuntos de estados y cadenas en conjuntos de estados. Sea entonces $\delta : \mathcal{P}(Q) \times \Sigma^* \to \mathcal{P}(Q)$ dada por:

$$\delta(P, x) = \bigcup_{q \in P} \hat{\delta}(q, x)$$

2.3. Equivalencia Entre AFND y AFD

Teorema 2.3.1: Dado un AFND $N = \langle Q_N, \Sigma, \delta_N, q_0, F_N \rangle$, existe un AFD $D = \langle Q_D, \Sigma, \delta_D, \{q_0\}, F_D \rangle$ tal que $\mathcal{L}(N) = \mathcal{L}(D)$

Demostración: La demostración comienza mediante una construcción llamada *construcción de subconjunto*, llamada así porque construye un autómata a partir de subconjuntos del conjunto de estados de otro (es decir, subconjuntos de $\mathcal{P}(Q)$).
Dado el autómata N AFND, construimos al autómata D de la siguiente manera:

- Q_D es el conjunto de subconjuntos de Q_N (es decir, $\mathcal{P}(Q_N)$). Notar que si Q_N tenía n estados, Q_D va a tener 2^n estados (ouch)
- F_D es el conjunto de subconjuntos S de Q_N tal que $S \cap F_N \neq \emptyset$
- Para cada S $\subseteq Q_N$ y símbolo $\alpha \in \Sigma$:

$$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$$

Ahora, probamos primero por inducción sobre $|\omega|$ que vale:

$$\hat{\delta}_D(\{q_0\}, \omega) = \hat{\delta}_N(q_0, \omega)$$

Notar que ambas $\hat{\delta}$ devuelven un conjunto de estados de Q_N, pero para nuestro AFD D el resultado es interpretado como un estado, mientras que en el AFND N se trata de un subconjunto de estados de Q_N.

- Caso Base: $|\omega|$ = 0 (O sea, $\omega = \lambda$)
  Por la definición de ambos $\hat{\delta}$ tenemos que ambos son $\{q_0\}$

- Paso Inductivo: Sea ω delongitud n + 1 y asumiendo que la H.I vale para n. Separamos ω = $x\alpha$, donde α es el último símbolo de ω. Por la H.I, tenemos que $\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x) = \{p_1, ..., p_k\}$. conjuntos de estados $\in Q_N$.

Recordando la definición de AFND teníamos que:

$$\hat{\delta}_N(q_0, \omega) = \bigcup_{i=1}^{k} \delta_N(p_i, \alpha)$$

Por otro lado, al construir el AFD definimos que:

$$\delta_D(\{p_1, p_2, ..., p_k\}, \alpha) = \bigcup_{i=1}^{k} \delta_N(p_i, \alpha)$$

Con esto en mente, podemos resolver:

$$\hat{\delta}_D(\{q_0\}, \omega) = \delta_D\Big(\hat{\delta}_D(\{q_0\}, x), \alpha\Big) \underset{HI}{=} \delta_D(\{p_1, ..., p_k\}, \alpha) = \bigcup_{i=1}^{k} \delta_N(p_i, \alpha) = \hat{\delta}_N(q_0, \omega)$$

Sabiendo ahora que ambas funciones de transición son "equivalentes" y que ambas aceptan una cadena sii el conjunto resultante contiene un estado final, tenemos que $\mathcal{L}(N) = \mathcal{L}(D)$ □

2.4. AFND con transiciones λ $(AFND - \lambda)$

Definición 2.4.1: Un AFND-λ es una 5-upla $\langle Q, \Sigma, \delta, q_0, F\rangle$ donde todos los componentes tienen la misma interpretación que antes con la excepción de que δ ahora tiene su dominio definido como:

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \to \mathcal{P}(Q)$$

Definición 2.4.2: Definimos como **clausura $\boldsymbol{\lambda}$** de un estado q, denotado $Cl_\lambda(q)$, al conjunto de estados alcanzables desde q mediante transiciones λ. Es decir, sea R $\subseteq Q \times Q$ tal que (q, p) $\in R \iff p \in \delta(q, \lambda)$. Definimos $Cl_\lambda : Q \to \mathcal{P}(Q)$ como:

$$Cl_\lambda(q) = \{p : (q, p) \in R^*\}$$

Definición 2.4.3: Podemos también extender la definición para un conjunto de estados P:

$$Cl_\lambda(P) = \bigcup_{q \in P} Cl_\lambda(q)$$

Definición 2.4.4: La función de trancisión δ puede extenderse para acept ar cadenas en Σ, es decir, $\hat{\delta} : Q \times \Sigma^* \to P(Q)$, y la definimos de la siguiente manera:

- $\hat{\delta}(q, \lambda) = Cl_\lambda(q)$.

- $\hat{\delta}(q, x\alpha) = Cl_\lambda\Big(\big\{p : \exists r \in \hat{\delta}(q, x) : p \in \delta(r, \alpha)\big\}\Big)$ con x $\in \Sigma^*$ y $\alpha \in \Sigma$ o, lo que es equivalente
$$\hat{\delta}(q, x\alpha) = Cl_\lambda\Big(\bigcup_{r \in \hat{\delta}(q,x)} \delta(r, \alpha)\Big)$$

Extendiendo la definición a conjunto de estados, tenemos que:

- $\delta(P, \alpha) = \bigcup_{q \in P} \delta(q, \alpha)$
- $\hat{\delta}(P, x) = \bigcup_{q \in P} \hat{\delta}(q, x)$

Lo que nos permite reescribir $\hat{\delta}(q, x\alpha)$ como:
$$\hat{\delta}(q, x\alpha) = Cl_\lambda\Big(\delta\Big(\hat{\delta}(q, x), \alpha\Big)\Big)$$

Definición 2.4.5: Definimos al lenguaje aceptado por un AFND-λ E:
$$\mathcal{L}(E) = \Big\{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\Big\}$$

Teorema 2.4.1: Dado un AFND-λ E = $\langle Q, \Sigma, \delta_\lambda, q_0, F_\lambda\rangle$ puede encontrarse un AFND N = $\langle Q, \Sigma, \delta_N, q_0, F_N\rangle$ que reconoce el mismo lenguaje

Demostración: Tomemos
$$F_N = \begin{cases} F_\lambda \cup \{q_0\} \text{ si } \ Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset \\ F_\lambda \text{ si no} \end{cases}$$

Y tomemos $\hat{\delta}_N(q, \alpha) = \hat{\delta}_\lambda(q, \alpha)$. Para empezar, vamos a probar que $\hat{\delta}_N(q, x) = \hat{\delta}_\lambda(q, x)$ para $|x| \geq 1$. Lo hacemos por inducción:

- Para |x| = 1:
  $x = \alpha$, es decir, un símbolo del alfabeto, por lo que se cumple por definición
- Para |x| >:
  Tomemos $x = \omega\alpha$ entonces:
$$\hat{\delta}_N(q_0, \omega\alpha) = \hat{\delta}_N\Big(\hat{\delta}_N(q_0, \omega), \alpha\Big) \underset{H.I}{=} \hat{\delta}_N\Big(\hat{\delta}_\lambda(q_0, \omega), \alpha\Big)$$

  Por otro lado, tenemos que para $P \subseteq Q$ es cierto que $\hat{\delta}_N(P, \alpha) = \hat{\delta}_\lambda(P, \alpha)$ ya que:

$$\hat{\delta}_N(P,\alpha) = \bigcup_{q\in P} \hat{\delta}_N(q,\alpha) = \bigcup_{q\in P} \hat{\delta}_\lambda(q,\alpha) = \hat{\delta}_\lambda(P,\alpha)$$

Luego, con $P = \hat{\delta}_\lambda(q_0,\omega)$ tenemos que ambas funciones de transición son "equivalentes" pues:

$$\hat{\delta}_N(q_0,\omega) = \hat{\delta}_N\big(\hat{\delta}_\lambda(q_0,\omega),\alpha\big) = \hat{\delta}_\lambda\big(\hat{\delta}_\lambda(q_0,\omega),\alpha\big) = \hat{\delta}_\lambda(q_0,\omega\alpha)$$

Ahora queda ver que $\mathcal{L}(N) = \mathcal{L}(E)$. Para $x = \lambda$ tenemos

- $\lambda \in \mathcal{L}(E) \iff \hat{\delta}_\lambda(q_0,\lambda) \cap F_\lambda \neq \emptyset \underset{\hat{\delta}_\lambda(q_0,\lambda)=Cl_\lambda(q_0)}{\iff} Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset$

$$\implies q_0 \in F_N \iff \lambda \in \mathcal{L}(N)$$

- $\lambda \in \mathcal{L}(N) \iff q_0 \in F_N \implies q_0 \in F_\lambda \vee Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset \iff$

$Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset \vee \lambda \in \mathcal{L}(E) \iff \lambda \in \mathcal{L}(E) \vee \lambda \in \mathcal{L}(E) \iff \lambda \in \mathcal{L}(E)$

Para |x| $\neq \lambda$:

- $x \in \mathcal{L}(E) \underset{\text{def cadena aceptada AFND-}\lambda}{\iff} \hat{\delta}_\lambda(q_0,x) \cap F_\lambda \neq \emptyset \underset{\substack{\text{porque } F_\lambda \subset F_N \\ \hat{\delta}_\lambda(q_0,x)=\hat{\delta}_N(q_0,x)}}{\implies}$

$$\hat{\delta}_N(q_0,x) \cap F_N \neq \emptyset \implies x \in \mathcal{L}(N)$$

Por el otro lado:

- $x \in \mathcal{L}(N) \iff \hat{\delta}_N(q_0,x) \cap F_N \neq \emptyset$

$$\implies \hat{\delta}_\lambda(q_0,x) \cap F_\lambda \neq \emptyset \vee \big(\hat{\delta}_\lambda(q_0,x) \cap \{q_0\} \neq \emptyset \wedge Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset\big)$$

- Del lado izquierdo de la disyunción tenemos:

$$\hat{\delta}_\lambda(q_0,x) \cap F_\lambda \neq \emptyset \implies x \in \mathcal{L}(E)$$

- Del lado derecho tenemos que:
  - $\hat{\delta}_\lambda(q_0,x) \cap \{q_0\} \neq \emptyset \implies$ existe un loop x de q_0 sobre sí mismo
  - $Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset \implies$ existe un camino λ desde q_0 hasta F_λ, por lo que existe un camino x desde q_0 hasta F_λ

- Juntando los dos, tenemos que:

$$\big(\hat{\delta}_\lambda(q_0,x) \cap \{q_0\} \neq \emptyset \wedge Cl_\lambda(q_0) \cap F_\lambda \neq \emptyset\big) \implies x \in \mathcal{L}(E)$$

Finalmente, podemos concluir que:

$$x \in \mathcal{L}(E) \iff x \in \mathcal{L}(N)$$

□

2.5. Gramáticas regulares y AFDs

Recordemos que una gramática G = $\langle V_N, V_T, P, S\rangle$ es regular si todas sus producciones son de la forma $A \to \lambda$, $A \to a$ o $A \to aB$

Teorema 2.5.1: Dada una gramática regular G = $\langle V_N, V_T, P, S\rangle$, existe un AFND $N = \langle Q, \Sigma, \delta, q_0, F\rangle$ tal que $\mathcal{L}(G) = \mathcal{L}(M)$

Demostración: Definamos N de la siguiente manera:

- $Q = V_N \cup \{q_f\}$. A partir de ahora, usamos q_A para referirnos al estado correspondiente al no terminal A
- $\Sigma = V_T$
- $q_0 = q_S$
- $q_B \in \delta(q_A, a) \iff A \to aB \in P$
- $q_f \in \delta(q_A, a) \iff A \to a \in P$
- $q_A \in F \iff A \to \lambda \in P$
- $q_f \in F$

Como paso intermedio, ahora probamos el siguiente lema:

Lema 2.5.2: Para todo $\omega \in V_T^*$, si $A \overset{*}{\to} \omega B$ entonces $q_B \in \hat{\delta}(q_A, \omega)$

Demostración: Por inducción en la longitud de ω

- Caso base $|\omega| = 0$, $(\omega = \lambda)$

Como $A \overset{*}{\to} A$ y $q_a \in \hat{\delta}(q_A, \lambda)$ tenemos por definición de N que $A \overset{*}{\to} A \iff q_a \in \hat{\delta}(q_A, \lambda)$

- Caso $|\omega| = n + 1, n \geq 0$, con, $\omega = x\alpha$:

$$
\begin{aligned}
A \overset{*}{\to} x\alpha B &\iff \exists C \in V_N : A \overset{*}{\to} xC \wedge C \to \alpha B \in P \\
&\underset{HI}{\Longrightarrow} \exists q_C \in Q : q_C \in \hat{\delta}(q_A, x) \wedge q_B \in \delta(q_C, a) \\
&\iff q_B \in \delta\Big(\hat{\delta}(q_A, x), a\Big) \\
&\iff q_B \in \hat{\delta}(q_A, x\alpha)
\end{aligned}
$$

□

Volviendo ahora con el teorema,

$$\omega\alpha \in \mathcal{L}(G) \underset{\substack{\text{def lenguaje generado} \\ \text{por una gramática}}}{\iff} S \overset{*}{\to} \omega\alpha$$

$$\iff \Big(\exists A \in V_N, S \overset{*}{\to} \omega A \wedge A \to \alpha \in P\Big) \vee \Big(\exists B \in V_N, S \overset{*}{\to} \omega\alpha B \wedge B \to \lambda\Big)$$

$$\iff \Big(\exists A \in V_N, S \overset{*}{\to} \omega A \wedge q_f \in \delta(q_A, \alpha)\Big) \vee \Big(\exists B \in V_N, S \overset{*}{\to} \omega\alpha B \wedge q_B \in F\Big)$$

$$\underset{\text{por lema}}{\Longrightarrow} \Big(\exists q_A \in Q : q_A \in \hat{\delta}(q_S, \omega) \wedge q_f \in \delta(q_A, \alpha)\Big) \vee \Big(\exists q_B \in Q : q_B \in \hat{\delta}(q_S, \omega\alpha) \wedge q_B \in F\Big)$$

$$\iff \Big(q_f \in \hat{\delta}(q_S, \omega\alpha)\Big) \vee \Big(\hat{\delta}(q_S, \omega\alpha) \cap F \neq \emptyset\Big)$$

$$\iff \Big(\hat{\delta}(q_S, \omega\alpha) \cap F \neq \emptyset\Big) \vee \Big(\hat{\delta}(q_S, \omega\alpha) \cap F \neq \emptyset\Big)$$

$$\iff \Big(\hat{\delta}(q_S, \omega\alpha) \cap F \neq \emptyset\Big)$$

$$\iff \omega\alpha \in \mathcal{L}(N)$$

Luego, queda demostrado que $\mathcal{L}(G) = \mathcal{L}(N)$

Teorema 2.5.3: Dado un AFD $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ existe una gramática regular $G = \langle V_N, V_T, P, S\rangle$ tal que $\mathcal{L}(M) = \mathcal{L}(G)$

Demostración: Para comenzar definimos una gramática $G = \langle V_N, V_T, P, S\rangle$, donde $V_N = Q$ y llamaremos A_p al no terminal correspondiente a p $\in Q$; $S = A_{q_0}$; $V_T = \Sigma$ y el conjunto P es:

$$A_p \to aA_q \in P \iff \delta(p, a) = q$$
$$A_p \to a \in P \iff \delta(p, a) = q \in F$$
$$S \to \lambda \in P \iff q_0 \in F$$

Ahora vamos a probar el siguiente lema como paso intermedio:

Lema 2.5.4: $\delta(p, \omega) = q \iff A_p \overset{*}{\to} \omega A_q$

Lo probamos por induccion en la longitud de ω:

Para $\omega = \lambda$ tenemos que $\delta(p, \lambda) = p$ y que $A_p \overset{*}{\to} A_p$, por lo que $\delta(p, \lambda) = p \iff A_p \overset{*}{\to} A_p$

Veamos que vale ahora para $\omega = x\alpha$:

$$\begin{aligned}\delta(p, x\alpha) = q &\iff \exists r \in Q, \hat{\delta}(p, x) = r \wedge \delta(r, \alpha) = q \\ &\underset{HI}{\iff} \exists A_r, A_p \overset{*}{\to} xA_r \wedge A_r \to \alpha A_q \in P \\ &\iff A_P \overset{*}{\to} x\alpha A_q\end{aligned}$$

Luego, probamos el lema. Volviendo ahora a la demo del teorema:

$$\begin{aligned}\lambda \in \mathcal{L}(M) &\iff q_0 \in F \\ &\iff (S \to \lambda) \in P \\ &\iff S \overset{*}{\Rightarrow} \lambda \\ &\iff \lambda \in \mathcal{L}(G).\end{aligned}$$

Caso cadena no vacía:

$$\begin{aligned}wa \in \mathcal{L}(M) &\iff \delta(q_0, wa) \in F \\ &\iff \exists p \in Q : \delta(q_0, w) = p \wedge \delta(p, a) \in F \\ &\underset{\text{Lema}}{\iff} \exists A_p \in V_N : A_{q_0} \overset{*}{\Rightarrow} wA_p \wedge \big(A_p \to a\big) \in P \\ &\iff A_{q_0} \overset{*}{\Rightarrow} wa \\ &\iff wa \in \mathcal{L}(G).\end{aligned}$$

Con esto queda demostrado que $\mathcal{L}(M) = \mathcal{L}(G)$.

□

3. Expresiones Regulares

Definición 3.1: Una expresión regular es una cadena de símbolos de un alfabeto que denotan un lenguaje sobre el alfabeto. Las expresiones regulares se construyen a partir de los siguientes elementos:

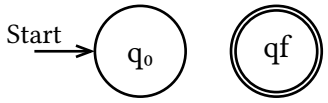
- $\emptyset$ es una expresión regular que representa el lenguaje vacío $\emptyset$
- λ es una expresión regular que representa el lenguaje que contiene sólo la cadena vacía $\{\lambda\}$
- a es una expresión regular que representa el lenguaje que contiene la cadena a para cada $a \in \Sigma$ ($\{a\}$)
- Si r y s son expresiones regulares que denotan los lenguajes R y S, entonces:
  - $(r + s) = (r \mid s)$ es una expresión regular que representa la unión de los lenguajes R y S.
  - (rs) es una expresión regular que representa la concatenación de los lenguajes R y S.
  - (r^*) es una expresión regular que representa la clausura de Kleene del lenguaje representado por r.
  - r^+ representa la clausura positiva del lenguaje representado por r.

Ejemplo: Algunos ejemplos de expresiones regulares son:
- $a + b$ representa el lenguaje que contiene las cadenas a y b
- a^*b representa el lenguaje que contiene las cadenas que comienzan con una cantidad arbitraria de a seguida de una b
- $(a + b)^*$ representa el lenguaje que contiene todas las cadenas que contienen únicamente a y b

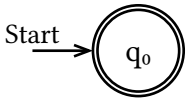
Teorema 3.1: Dada una expresión regular r, existe un AFND-λ con un solo estado final y sin transiciones a partir del mismo $E = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que $\mathcal{L}(r) = \mathcal{L}(E)$

Demostración: Por inducción sobre la estructura de la expresión regular r:

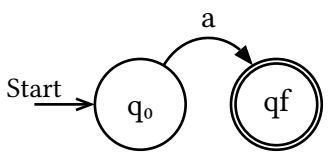
- Caso base: $r = \emptyset$



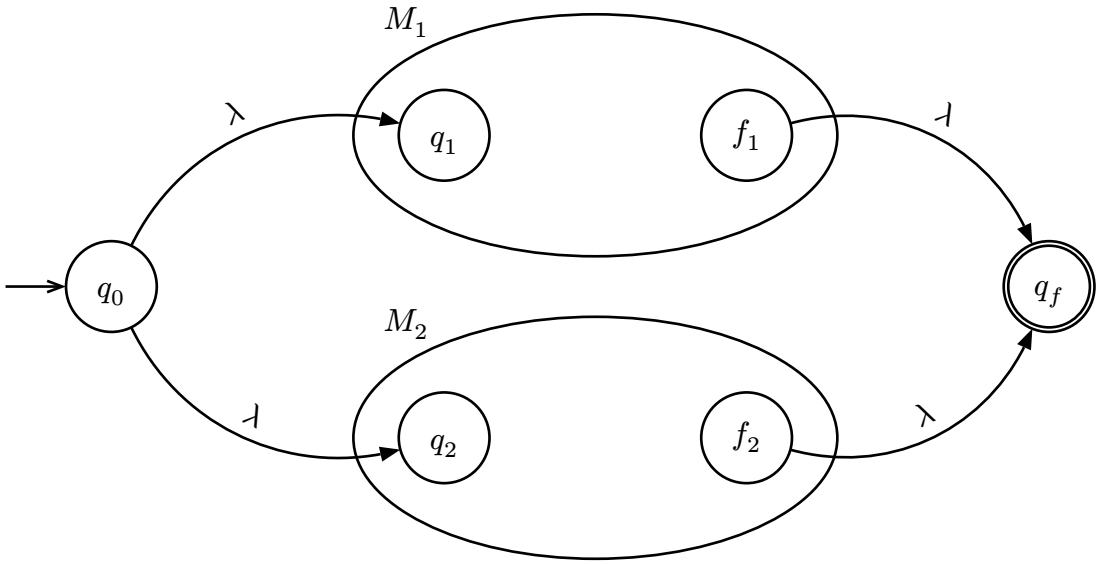
- Caso base: $r = \lambda$



- Caso base: $r = a$



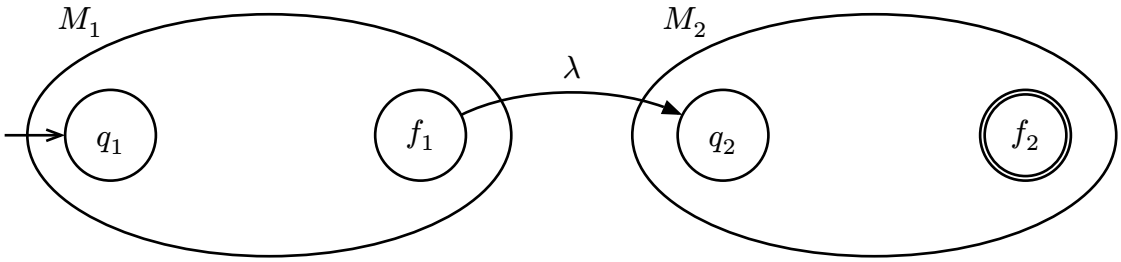
- Caso inductivo: $r = r_1 + r_2$ Por H.I, existen $M_1 = \langle Q_1, \Sigma_1, \delta_1, q_1, \{f_1\}\rangle$ y $M_2 = \langle Q_2, \Sigma_2, \delta_2, q_2, \{f_2\}\rangle$ tales que $\mathcal{L}(M_1) = \mathcal{L}(r_1)$ y $\mathcal{L}(M_2) = \mathcal{L}(r_2)$. Sea $M = \langle Q_1 \cup Q_2 \cup \{q_0, q_f\}, \Sigma_1 \cup \Sigma_2, \delta, q_0, \{q_f\}\rangle$:



- $\delta(q_0, \lambda) = \{q_1, q_2\}$.
- $\delta(q, a) = \delta_1(q, a)$ para $q \in Q_1 - \{f_1\}$ y $a \in \Sigma_1 \cup \{\lambda\}$.
- $\delta(q, a) = \delta_2(q, a)$ para $q \in Q_2 - \{f_2\}$ y $a \in \Sigma_2 \cup \{\lambda\}$.
- $\delta(f_1, \lambda) = \delta(f_2, \lambda) = \{q_f\}$.

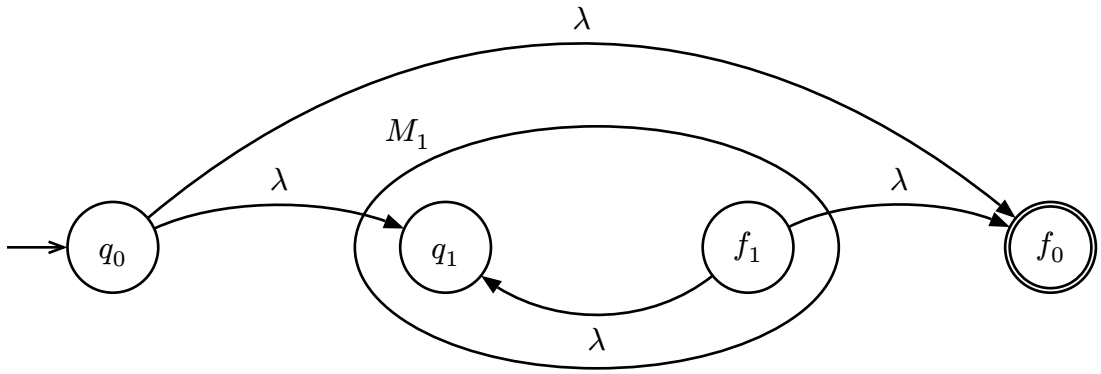
De manera informal, podemos alcanzar, a partir del nuevo estado inicial, cualquiera de los estados iniciales correspondientes a los autómatas para r_1 y r_2 mediante transiciones λ. Luego, podemos seguir las transiciones de los autómatas hasta llegar a sus estados previamente finales correspondientes, y luego tomar una úlima transición λ al nuevo estado final.

- Caso inductivo: $r = r_1 r_2$
  Por HI existen $M_1 = \langle Q_1, \Sigma_1, \delta_1, q_1, \{f_1\}\rangle$ y $M_2 = \langle Q_2, \Sigma_2, \delta_2, q_2, \{f_2\}\rangle$ tales que $\mathcal{L}(M_1) = \mathcal{L}(r_1)$ y $\mathcal{L}(M_2) = \mathcal{L}(r_2)$.
  Entones sea $M = \langle Q_1 \cup Q_2, \Sigma_1 \cup \Sigma_2, \delta, q_1, \{f_2\}\rangle$:



  - $\delta(q, a) = \delta_1(q, a)$ para $q \in Q_1 - \{f_1\}$ y $a \in \Sigma_1 \cup \{\lambda\}$.
  - $\delta(f_1, \lambda) = \{q_2\}$.
  - $\delta(q, a) = \delta_2(q, a)$ para $q \in Q_2 - \{f_2\}$ y $a \in \Sigma_2 \cup \{\lambda\}$.

- Caso inductivo: $r = r_1^*$
  Por HI existe $M_1 = \langle Q_1, \Sigma_1, \delta_1, q_1, \{f_1\}\rangle$ tal que $\mathcal{L}(M_1) = \mathcal{L}(r_1)$.
  Entonces, podemos construir el autómata $M = \langle Q_1 \cup \{q_0, f_0\}, \Sigma_1, \delta, q_0, \{f_0\}\rangle$.



- $\delta(q, a) = \delta_1(q, a)$ para $q \in Q_1 - \{f_1\}$ y $a \in \Sigma_1 \cup \{\lambda\}$.
- $\delta(q_0, \lambda) = \delta(f_1, \lambda) = \{q_1, f_0\}$.

- Caso infuctivo $r = r_1^+$:

  Dado que $r_1^+ = r_1 r_1^*$, queda demostrado por los casos anteriores.

  Con esto, queda demostrado que $\mathcal{L}(r) = \mathcal{L}(M)$ para todo r expresión regular.

□

Teorema 3.2: Dado un AFD $M = \langle Q, \Sigma, \delta, q_0, F\rangle$, existe una expresión regular r tal que $\mathcal{L}(r) = \mathcal{L}(M)$

Demostración: Como tenemos que los estados de un autómata son finitos, podemos renombrar los mismos como $\{q_1, q_2..., q_n\}$con $n = |Q|$. Con esto en cuenta, denotamos con $R_{i,j}^k$ al conjunto de cadenas que llevan al autómata de q_i a q_j sin pasar por ningún estado intermedio con índice mayor que k. Luego, definimos $R_{i,j}^k$ inductivamente como:

- $R_{i,j}^0 = \begin{cases} \{a{:}\delta(q_i,a)=q_j\}\text{: } a \in \Sigma \text{ si } i \neq j \\ \{a{:}\delta(q_i,a)=q_j\} \cup \{\lambda\} \text{ si } i = j \end{cases}$
- $R_{i,j}^k = R_{i,j}^{k-1} \cup R_{i,k}^{k-1}\left(R_{k,k}^{k-1}\right)^* R_{k,j}^{k-1}$

Como paso intermedio, queremos demostrar que para todo $R_{i,j}^k$ existe una e.r $r_{i,j}^k$ tal que $\mathcal{L}\left(r_{i,j}^k\right) = R_{i,j}^k$ Haciendo inducción sobre k:

- Caso base: $k = 0$
  $R_{i,j}^0$ es el conjunto de cadenas de un solo caracter o λ. Por lo que la e.r $r_{i,j}^k$ que lo denota será:
  - $\emptyset$ si no existe ningún a_i que una q_i y q_j y i ≠ j
  - λ si no existe ningún a_i que una q_i y q_j, pero con i = j
  - $a_1 \mid ... \mid a_p$ con a_l simbolos del alfabeto, si delta(q_i, a_l) = q_j y j ≠ i
  - $a_1 \mid ... \mid a_p \mid \lambda$ con a_l simbolos del alfabeto, si delta(q_i, a_l) = q_j y j = i
- Paso inductivo: Por H.I tenemos que:

$$\mathcal{L}\left(r_{i,k}^{k-1}\right) = R_{i,k}^{k-1} \quad \mathcal{L}\left(r_{k,k}^{k-1}\right) = R_{k,k}^{k-1} \quad \mathcal{L}\left(r_{k,j}^{k-1}\right) = R_{k,j}^{k-1} \quad \mathcal{L}\left(r_{i,j}^{k-1}\right) = R_{i,j}^{k-1}$$

Si definimos $r_{i,j}^k = r_{i,k}^{k-1}\left(r_{k,k}^{k-1}\right)^* r_{k,j}^{k-1} \mid r_{i,j}^{k-1}$ tenemos que:

$$\mathcal{L}\left(r_{i,j}^{k}\right) =$$

$$\mathcal{L}\left(r_{i,k}^{k-1}\left(r_{k,k}^{k-1}\right)^{*}r_{k,j}^{k-1} \mid r_{i,j}^{k-1}\right) =$$

$$\mathcal{L}\left(r_{i,k}^{k-1}\left(r_{k,k}^{k-1}\right)^{*}r_{k,j}^{k-1}\right) \cup \mathcal{L}\left(r_{i,j}^{k-1}\right) =$$

$$\mathcal{L}\left(r_{i,k}^{k-1}\right)\mathcal{L}\left(\left(r_{k,k}^{k-1}\right)^{*}\right)\mathcal{L}\left(r_{k,j}^{k-1}\right) \cup \mathcal{L}\left(r_{i,j}^{k-1}\right) =$$

$$\mathcal{L}\left(r_{i,k}^{k-1}\right)\left(\mathcal{L}\left(r_{k,k}^{k-1}\right)^{*}\right)\mathcal{L}\left(r_{k,j}^{k-1}\right) \cup \mathcal{L}\left(r_{i,j}^{k-1}\right) =$$

$$R_{i,k}^{k-1}\left(R_{k,k}^{k-1}\right)^{*}R_{k,j}^{k-1} \cup R_{i,j}^{k-1} =$$

$$R_{i,j}^{k}$$

Con esto en mente, notemos primero que el lenguaje denotado por el autómata M está dado por todas las cadenas que llevan al autómata de q_0 a un estado final. Es decir:

$$\mathcal{L}(M) = \bigcup_{q_j \in F} R_{1,j}^{n}$$

Luego, tenemos que:

$$\mathcal{L}(M) = \bigcup_{q_j \in F} \mathcal{L}\left(R_{1,j}^{n}\right) = \bigcup_{q_j \in F} \mathcal{L}\left(r_{1,j}^{n}\right) = \mathcal{L}\left(r_{1,j1}^{n} \mid \ldots \mid r_{1,jm}^{n}\right)$$

Por lo que concluimos que $\mathcal{L}(M)$ es denotado por la e.r $r_{1,j1}^{n} \mid \ldots \mid r_{1,jm}^{n}$ $\square$

4. Lema de Pumping y propiedades de clausura de los lenguajes regulares

4.1. Configuración instantánea de un AFD

Para comenzar, vamos a introducir una nueva notación que nos facilitará un par de pruebas

Definición 4.1.1: Sea AFD M = $\langle Q, \Sigma, \delta, q_0, F\rangle$. Una configuración instantánea es un par $(q, w) \in Q \times \Sigma *$ donde q es el estado en el que está el autómata y w es la cadena que resta consumir

Definición 4.1.2: Llamamos transición a la siguiente relación sobre $Q \times \Sigma^*$:

$$(q, w) \vdash (p, \beta) \text{ si } (\delta(q, a) = p \text{ y } w = a\beta)$$

De esto tenemos que $(q, \alpha\beta) \overset{*}{\vdash} (p, \beta) \iff \hat{\delta}(q, \alpha) = p$. Es decir, se puede llegar al estado p consumiendo la cadena α

Lema 4.1.1: Sea M = $\langle Q, \Sigma, \delta, q_0, F\rangle$ un AFD. Para todo $q \in Q$ y $\alpha, \beta \in \Sigma^*$ se tiene que:

$$\text{si } (q, \alpha\beta) \overset{*}{\vdash} (q, \beta) \text{ entonces } \forall i \geq 0, (q, \alpha^i\beta) \overset{*}{\vdash} (q, \beta)$$

Demostración: Por inducción sobre i:

- Caso base: $i = 0$
  $(q, \alpha^0\beta) \overset{0}{\vdash} (q, \beta)$
- Paso inductivo:
  Supongamos que vale para i, es decir si $(q, \alpha\beta) \overset{*}{\vdash} (q, \beta)$ entonces $(q, \alpha^i\beta) \overset{*}{\vdash} (q, \beta)$. Veamos que vale para i + 1:
  $(q, \alpha^{i+1}\beta) = (q, \alpha\alpha^i\beta) \underset{\substack{\text{por el antecedente} \\ \text{q asumimos}}}{\Longrightarrow} (q, \alpha\alpha^i\beta) \overset{*}{\vdash} (q, \alpha^i\beta) \underset{HI}{\overset{*}{\vdash}} (q, \beta)$

□

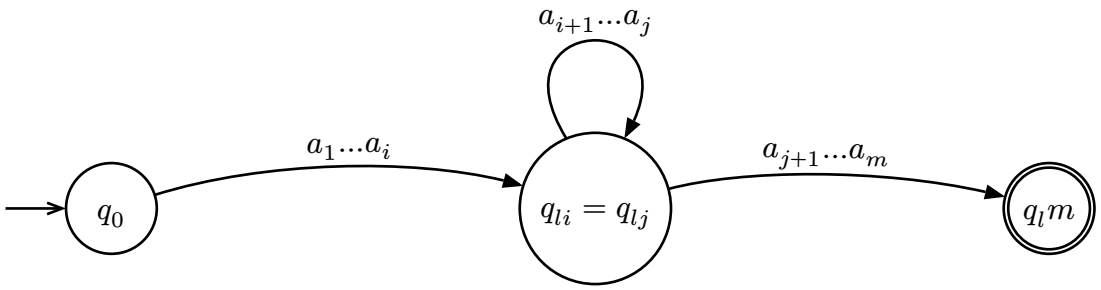
4.2. Lema de Pumping

Teorema 4.2.1: Sea L un lenguaje regular, entonces existe un número n tal que para toda cadena z en L con |z| ≥ n, existen cadenas u, v, w tales que:

- z = uvw,
- $|uv| \leq$ n,
- |v| ≥ 1

  $\forall$i ≥ 0, $uv^iw \in$ L

Demostración: Sea M un AFD tal que $\mathcal{L}(M)$ = L. Sea n su cantidad de estados. Sea z una cadena de longitud $m \geq n$, $z = a_1 ... a_m$ los símbolos que forman la cadena.
Para aceptar z usamos m transiciones, por lo tanto pasamos a través de m + 1 estados. Como $m + 1 > n$, tenemos que necesariamente debemos pasar al menos dos veces por un mismo estado para aceptar la cadena (pigeonhole principle).
Sea entonces $q_{l0} ... q_{lm}$ la sucesión de estados desde $q_0 (q_{l0})$ hasta un estado final (q_{lm})



Existen entonces j y k mínimos tales que $q_{lj} = q_{lk}$ con $0 \leq j < k \leq n$. Esto separa a z en tres subcadenas:

- $u = \begin{cases} a_1 ... a_j \text{ si } j>0 \\ \lambda \text{ si } j=0 \end{cases}$
- $v = a_{j+1} ... a_k$
- $w = \begin{cases} a_{k+1} ... a_m \text{ si } k<m \\ \lambda \text{ si } k=m \end{cases}$

Juntando todo esto, tenemos que:

$$|uv| \leq n$$
$$|v| \geq 0$$

y que:

$$(q_0, uvw) \overset{*}{\vdash} (q_{lj}, vw) \overset{*}{\vdash} (q_{lk}, w) \overset{*}{\vdash} (q_{lm}, \lambda)$$

Pero como $q_{lj} = q_{lk}$, y por el lema probado en la sección anterior, tenemos que:

$\forall i \geq 0 (q_{lj}, v^i w) \overset{*}{\vdash} (q_{lj}, w) = (q_{lk}, w)$ que tenemos alcanza un estado final

Por lo tanto, $uv^i w \in$ L, $\forall i \geq 0$ □

Ejemplo: Sea AFD $M = \langle Q, \Sigma, \delta, q_0, F \rangle$, con $|Q|$ = n. Determinar veracidad y justificar:
1. $\mathcal{L}(M)$ es no vacío si y solo si existe $w \in \Sigma^*$ tal que $\hat{\delta}(q_0, w) \in F y\ |w| < n$

Demostración:

- $\Longleftarrow$) Es trivial ver que el lenguaje no es vacío
- $\Longrightarrow$) Supongamos que el lenguaje no es vacío y regular. Entonces, supongamos existe una cadena $z \in \mathcal{L}(M)$. Hay dos posibilidades, o bien la longitud de la cadena es menor a n, o bien es mayor. En el primer caso, no hace falta demostrar nada mas. En el segundo caso, por el lema de pumping, podemos descomponer la cadena en uvw tal que $|uv| \leq n$ y $|v| \geq 1$ y estar seguros que para todo i se cumple que $uv^i w \in \mathcal{L}(M)$. Luego por el lema de pumping, podemos "pumpear" a v con i = 0, reduciendo el tamaño de la cadena y repitiendo este proceso hasta llegar a una con longitud menor a n.

□

2. $\mathcal{L}(M)$ es infinito si y solo si existe $w \in \Sigma^*$ tal que $\hat{\delta}(q_0, w) \in F$ y $n \leq |w| < 2n$

Demostración:

- $\Longleftarrow$) Suponemos $z \in \mathcal{L}(M)$ y $n \leq |z| < 2n$. Por el lema de Pumping, tenemos $z = uvx$ con $|uv| \leq n$ y $|v| \geq 1$ y $uv^i x \in \mathcal{L}(M)$ para todo i. Luego $\mathcal{L}(M)$ es infinito
- $\Longrightarrow$) Supongamos $\mathcal{L}(M)$ es infinito y regular. Supongamos también que no existe una cadena z en el lenguaje con longitud entre n y 2n - 1. Primero notemos que como el lenguaje es infinito, necesariamente debe haber cadenas de longitud mayor a n (Pues lo único que puede aportar a la "infinitez"(? es que el tamaño de las cadenas pueda ser arbitrariamente grandes)

  Sin pérdida de generalidad, supongamos que |z| = 2n (Notar que si la longitud fuese mayor, Simplemente podríamos bombear hacia repetir el argumento hasta que se cumpla esta condición o se llegue al rango deseado. Notar también que no es posible " saltearse" el rango pues el mismo es de longitud n + 1, y saltearlo implicaría un ciclo que abarca más que la cantidad de estados del autómata)

  Por Lema de Pumping, tenemos que existen u, v, x tales que $z = uvx$, con $|uv\ | \leq n$, $v \geq 1$ y $\forall i uv^i x \in \mathcal{L}(M)$. En particular, $uv^0 x = ux$ está en el lenguaje.

  Como $|uvx| = 2n$ y $1 \leq |v| \leq n$ tenemos que $n \leq |ux| \leq 2n - 1$, contradiciendo nuestra suposición que dicha cadena no existía.

□

4.3. Propiedades de clausura de los lenguajes regulares

4.3.1. Unión

Teorema 4.3.1.1: El conjunto de lenguajes regulares incluidos en Σ^* es cerrado respecto de la unión

Demostración: Sean L_1 y L_2 lenguajes regulares. Sea $M_1 = \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle$ tal que $\mathcal{L}(M_1) = L_1$ y $M_2 = \langle Q_2, \Sigma, \delta_2, q_2, F_2 \rangle$ tal que $\mathcal{L}(M_2) = L_2$ y $Q_1 \cap Q_2 = \emptyset$. Sea $M = \langle Q_1 \cup Q_2 \cup \{q_0\}, \Sigma, \delta, q_0, F_1 \cup F_2 \rangle$ tal que:

- $\forall q_1 \in Q_1, \forall q_2 \in Q_2, \forall a \in \Sigma, \delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$
- $F = \{(f_1, f_2) : f_1 \in F_1 \vee f_2 \in F_2\}$

tal que $\mathcal{L}(M) = L$.

Para probar que $L = L_1 \cup L_2$, basta probar que: $x \in \mathcal{L}(M) \iff x \in \mathcal{L}(M_1) \vee x \in \mathcal{L}(M_2)$.

$$
\begin{aligned}
x \in \mathcal{L}(M) &\iff \delta\Big(\Big(q_{1_0}, q_{2_0}\Big), x\Big) \in F \\
&\iff \Big(\delta_1\Big(q_{1_0}, x\Big), \delta_2\Big(q_{2_0}, x\Big)\Big) \in F \\
&\iff \delta_1\Big(q_{1_0}, x\Big) \in F_1 \vee \delta_2\Big(q_{2_0}, x\Big) \in F_2 \\
&\iff x \in \mathcal{L}(M_1) \vee x \in \mathcal{L}(M_2).
\end{aligned}
$$

□

5. Ejercicios (Después debería pasarlo a otro lado)

1. Demostra que $\hat{\delta}(q, xy) = \hat{\delta}\Big(\hat{\delta}(q, x), y\Big)$ para cualquier estado q y cadenas x e y. **Pista**: hacer inducción sobre |y|

   Demostración: Siguiendo la sugerencia:

   - Caso base: |y| = 0 (y = λ)

   $\hat{\delta}(q, x\lambda) = \hat{\delta}\Big(\hat{\delta}(q, x), \lambda\Big) \underset{\text{def } \hat{\delta}}{=} \hat{\delta}(q, x)$ $\underset{\lambda \text{ es neutro}}{=}$ $\hat{\delta}(q, x\lambda)$

   - Paso inductivo:

   qvq $\hat{\delta}(q, xy) = \hat{\delta}\Big(\hat{\delta}(q, x), y\Big)$. Sea entonces $y = y'\alpha$, tenemos, reescribiendo el lado derecho de la igualdad, $\hat{\delta}\Big(\hat{\delta}(q, x), y'\alpha\Big) \underset{\text{def de } \hat{\delta}}{=} \delta\Big(\hat{\delta}\Big(\hat{\delta}(q, x), y'\Big), \alpha\Big) \underset{\text{H.I}}{=} \delta\Big(\hat{\delta}(q, xy'), \alpha\Big) \underset{\text{def de } \hat{\delta}}{=}$ $\hat{\delta}(q, xy'\alpha) = \hat{\delta}(q, xy)$. □

2. Demostrá que para cualquier estado q, cadena x, y símbolo α, $\hat{\delta}(q, \alpha x) = \hat{\delta}(\delta(q, \alpha), x)$

   Demostración: Por el ejercicio anterior, tenemos que $\hat{\delta}(q, \alpha x) = \hat{\delta}\Big(\hat{\delta}(q, \alpha), x\Big)$. Luego, solo resta probar que $\hat{\delta}(q, \alpha) = \delta(q, \alpha)$ que sale fácil usando la definición □

3. Sea M un AFD y q un estado del mismo, tal que $\delta(q, \alpha) = q$ para cualquier símbolo α. Demostrar por inducción sobre el largo de la cadena de entrada que para toda cadena ω, $\hat{\delta}(q, \omega) = q$

   Demostración: Haciendo inducción sobre el largo de la cadena w:

   - Caso $|w| = 0$ ($w = \lambda$)

     $\hat{\delta}(q, \lambda) \underset{\text{def } \hat{\delta}}{=} \text{q}$

   - Caso $w = xa$

   $\hat{\delta}(q, xa) \underset{\text{def } \hat{\delta}}{=} \delta\Big(\hat{\delta}(q, x), a\Big) \underset{HI}{=} \delta(q, a) \underset{\text{Por enunciado}}{=} q$

   □

4. Sea M un AFD y α un símbolo particular de su alfabeto, tal que para todos los estados q de M tenemos que $\delta(q, \alpha) = q$

   a) Demostrar por inducción sobre n que para cualquier $n \geq 0, \hat{\delta}(q, \alpha^n) = q$

Demostración: Haciendo inducción sobre n:

- Caso $n = 0$

  $\hat{\delta}(q, a^0) = \hat{\delta}(q, \lambda) \underset{(\text{def } \hat{\delta})}{=} \text{q}$
- Caso $n \geq 1$

$\hat{\delta}(q, a^n) \underset{\text{def } \hat{\delta}}{=} \delta\Big(\hat{\delta}(q, a^{n-1}), a\Big) \underset{HI}{=} \delta(q, a) \underset{\text{Por enunciado}}{=} q$

□

b) Demostrar o bien $\{\alpha\}^* \subseteq \mathcal{L}(M)$ o bien $\{\alpha\}^* \cap \mathcal{L}(M) = \emptyset$

Demostración: Primero, una observación. Tenemos que la única manera en la que una cadena a^k para algún k esté en el lenguaje de M es si el estado inicial es final, pues tenemos por enunciado que la transición por a siempre es desde un estado a sí mismo. Con esta observaciones, tratamos primero de probar:

$$\{a\}^* \cap \mathcal{L}(M) \neq \emptyset \iff q_0 \in F$$

Probamos los dos lados:

- $\Longrightarrow$) Que la intersección no sea vacía implica que hay una cadena a^k aceptada por el lenguaje. Supongamos que q_0 no fuese un estado final, por enunciado sabemos que por a no podemos llegar a ningún otro estado, por lo que siempre nos quedamos en q_0 pero, como este no era estado final entonces nuestro autómata en realidad no acepta la cadena a^k. Llegamos a un absurdo, que vino de suponer que q_0 no era estado final
- $\Longleftarrow$) Sabemos por enunciado que tiene una transición por a hacia sí mismo, luego, como $q_0 \in F$, tenemos que $a \in \mathcal{L}(M)$

Con esto en mente, quedaría demostrar por inducción que si acepta a la cadena a, acepta todas las cadenas a^k, y una vez probado esto, como la inclusión de q_0 en F o bien pasa (y entonces todas las cadenas $\{a\}^*$ están en el lenguaje) o bien no está (por lo que ninguna cadena está):

$$q_0 \in F \oplus q_0 \notin F \iff$$
$$\{a\} \subseteq \mathcal{L}(M) \oplus \{a\}^* \cap \mathcal{L}(M) = \emptyset \iff$$
$$\{\alpha\}^* \subseteq \mathcal{L}(M) \oplus \{a\}^* \cap \mathcal{L}(M) = \emptyset$$

(**Considerar terminar esto último, es fácil creo pero bue, lo que sí debería considerr es emprolijar esto porque es mucho mas fácil**)

□

5. Sea M = $\langle Q, \Sigma, \delta, q_0, \{q_f\}\rangle$ un AFD tal que para todos los símbolos $\alpha \in \Sigma$ se cumple que $\delta(q_0, \alpha) = \delta(q_f, \alpha)$

a) Demostrar que para todo $\omega \neq \lambda,\ \hat{\delta}(q_0, \omega) = \hat{\delta}\big(q_f, \omega\big)$

Demostración: Por inducción sobre ω:

- Caso base: $\omega = \alpha$

$\hat{\delta}(q_0, \alpha) = \delta(q_0, \alpha) = \delta\big(q_f, \alpha\big) = \hat{\delta}\big(q_f, \alpha\big)$

- Paso inductivo: $\omega = x\alpha$

$\hat{\delta}(q_0, x\alpha) = \delta\Big(\hat{\delta}(q_0, x), \alpha\Big) \underset{HI}{=} \delta\Big(\hat{\delta}\big(q_f, x\big), \alpha\Big) \underset{\text{def } \hat{\delta}}{=} \hat{\delta}\big(q_f, x\alpha\big)$ □

b) Demostrar que si ω es una cadena no vacía en $\mathcal{L}(M)$, entonces para toda k > 0, ω^k también pertenece a $\mathcal{L}(M)$

Demostración: Por inducción sobre k:

- Caso base: $k = 1$

$\omega^1 = \omega$ y por enunciado $\omega \in \mathcal{L}(M)$

- Paso inductivo: $k = n + 1$

$\omega^{n+1} \in \mathcal{L}(M) \iff \hat{\delta}\big(q_0, w^{n+1}\big) \cap F \neq \emptyset \iff \hat{\delta}\big(q_0, w^{n+1}\big) = q_f \underset{\text{Por 1)}}{\iff} \hat{\delta}\Big(\hat{\delta}\big(q_0, w^n\big), w\Big) =$
$q_f \underset{HI}{\iff} \hat{\delta}\big(q_f, w\big) = q_f$ que sabemos es cierto por a) □

6. Sea N un AFND tal que tenga a lo sumo una opción para cualquier estado y símbolo (es decir, $|\delta(q, \alpha)| = 1$), entonces el AFD D construido tiene la misma cantidad de transiciones y estados más las trancisiones necesarias para ir a un nuevo estado trampa cuando una transición no esté definida para un estado particular
7. Demostrar que para todo AFND-λ E existe un AFD D tal que $\mathcal{L}(E) = \mathcal{L}(D)$ (Usando la demo del libro)

Demostración: Sea E = $\langle Q_\lambda, \Sigma, \delta_\lambda, q_0, F_\lambda \rangle$ un AFND-λ. Definimos
D = $\langle Q_D, \Sigma, \delta_D, q_D, F_D \rangle$ de la siguiente manera:

- $Q_D = \mathcal{P}(Q_\lambda)$. En particular, va a ocurrir que todos los estados accesibles de D son subconjuntos de Q_lambda cerrados por la clausura lambda, es decir sea S el subconjunto, $S = Cl_\lambda(S)$
- $q_D = Cl_\lambda(q_0)$
- $F_D = \{S \subseteq Q_\lambda : S \cap F_\lambda \neq \emptyset\}$
- Para cada $S \subseteq Q_\lambda$ y $\alpha \in \Sigma$, definimos:
  - $\delta_D(S, \alpha) = Cl_\lambda(\{r \in Q_\lambda : \exists p \in S \land r \in \delta_\lambda(p, a)\})$

Ahora probamos que para una cadena $w \in \Sigma^*$, $w \in \mathcal{L}(E) \iff w \in \mathcal{L}(D)$:

- $\Longleftarrow$) Simplemente podemos agregar transiciones λ en todos los estados hacia el estado representando el estado trampa, y convertimos cada una de las transiciones en el equivalente de conjuntos (i.e en vez de $q_i, \{q_i\}$ en las funciones de transición)
- $\Longrightarrow$) Primero demostramos por inducción sobre la longitud de la cadena w que $\hat{\delta}_\lambda(q_0, w) = \hat{\delta}_D(q_D, w)$:
  - |w| = 0:

  $\hat{\delta}_\lambda(q_0, \lambda) = Cl_\lambda(q_0) = q_D = \hat{\delta}_D(q_D, \lambda)$
  - $|w| > 0, w = xa$:

  $\hat{\delta}_\lambda(q_0, xa) = Cl_\lambda\Big(\Big\{r \in Q_\lambda : \exists p \in \hat{\delta}_\lambda(q_0, x) \land r \in \delta_\lambda(p, a)\Big\}\Big) = Cl_\lambda\Big(\Big\{r \in Q_\lambda : \exists p \in \hat{\delta}_D(q_D, x) \land r \in \delta_\lambda(p, a)\Big\}\Big) = \hat{\delta}_D(q_D, xa)$

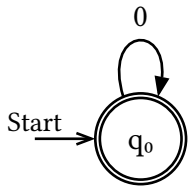
□

8. Indicar V o F y justificar
   - Si $D = \langle Q, \Sigma, \delta, q_0, F \rangle$ es un AFD entonces reconoce al menos |Q| palabras distintas, es decir $\#\mathcal{L}(D) \geq |Q|$

Falso. Como contraejemplo, un AFD que no tiene estados finales

   - Si $N = \langle Q, \Sigma, \delta, q_0, F \rangle$ es un AFND entonces todas las palabras de $\mathcal{L}(N)$ tienen longitud menor o igual a $|Q|^2$

Falso. Como contraejemplo, para $\Sigma = \{0, 1\}, L = \{w \mid 1 \notin w\}$, tenemos el AFND:



9. Cuántos AFD hay con |Q| = 2 y $|\Sigma| = 3$?

La fórmula general para la cantidad de AFDs posibles es $|Q|^{|Q| \times |\Sigma|} \times 2^{|Q|} \times |Q|$, por lo que para $|Q| = 2$ y $|\Sigma| = 3$ tenemos 512 AFDs posibles

10. Qué pasa al invertir los arcos en un AFD?

Noc si a esta pregunta le faltó algo más o quería darnos una pista sobre como arrancar a conseguir un AFD para el inverso de un lenguaje

11. Qué pasa al invertir los estados finales con no finales en un AFND?

A diferencia de con un AFD, invertir los estados no nos provee con un autómata que reconozca el complemento al lenguaje original

12. Demostrar que para cada AFND $N = \langle Q, \Sigma, \delta, q_0, F \rangle$ existe otro AFND-λ $E = \langle Q_\lambda, \Sigma, \delta_\lambda, q_0, F_\lambda \rangle$ tal que $\mathcal{L}(N) = \mathcal{L}(E)$ y F_λ tiene un solo estado final

Demostración: Definimos E de la siguiente manera:

- $Q_\lambda = Q \cup \{q_f\}$, donde q_f es un nuevo estado final
- $F_\lambda = \{q_f\}$
- $\delta_\lambda(q, \alpha) = \delta(q, \alpha)$ para todo $q \in Q, \alpha \in \Sigma$
- $\delta_\lambda(q, \lambda) = \{q_f\}$ para todo $q \in Q$ si $q \in F$ (Esto debería definirlo así, o usar la clausura lambda?)

Queremos demostrar que $\mathcal{L}(N) = \mathcal{L}(E)$, para eso primero notamos que $\hat{\delta}_N(q_0, w) = \hat{\delta}_E(q_{0\lambda}, w)$ para toda cadena $w \in \Sigma^+$. Con esto en mente tenemos que (Separamos en casos):

- $\lambda \in \mathcal{L}(N) \iff \delta(q_0, \lambda) \cap F \neq \emptyset \iff \{q_0\} \cap F \neq \emptyset \implies \delta_\lambda(q_0, \lambda) = \{q_f\} \implies \delta(q_0, \lambda) \cap F_\lambda \neq \emptyset \iff \lambda \in \mathcal{L}(E)$
- $\lambda \in \mathcal{L}(E) \iff \delta(q_0, \lambda) \cap F_\lambda \neq \emptyset \implies \delta(q_0, \lambda) \cap F \neq \emptyset \iff \lambda \in \mathcal{L}(N)$
- $w \in \mathcal{L}(N) \iff \hat{\delta}_N(q_0, w) \cap F \neq \emptyset \implies \delta_\lambda\left(\hat{\delta}_\lambda(q_0, w), \lambda\right) = \{q_f\} \iff \hat{\delta}_\lambda(q_0, w) \cap F_\lambda \neq \emptyset \iff w \in \mathcal{L}(E)$
- $w \in \mathcal{L}(E) \iff \hat{\delta}_\lambda(q_0, w) \cap F_\lambda \neq \emptyset \iff \delta_\lambda\left(\hat{\delta}_\lambda(q_0, w), \lambda\right) = \{q_f\} \implies \hat{\delta}(q_0, w) \cap F \neq \emptyset \iff w \in \mathcal{L}(N)$

□

13. Sea Σ un alfabeto con al menos dos símbolos, y sea a un símbolo de Σ. Sea $N = \langle Q, \Sigma, \delta, q_0, F\rangle$ un AFND, considerar el AFND-λ $E = \langle Q, \Sigma \setminus \{a\}, \delta_\lambda, q_0, F\rangle$ que se obtiene por reemplazar todas las transiciones por el símbolo a por trancisiones λ. Es decir:

    - Para todo $q \in Q, x \in \Sigma : x \neq a, \delta(q, x) = \delta_\lambda(q, x)$
    - Para todo $q \in Q, \delta_\lambda(q, \lambda) = \delta(q, a)$

    Determinar cuál es el lenguaje aceptado por E

14. Es posible acotar superiormente cuántas trancisiones requiere la aceptación de una palabra en un AFND-λ?

Noc (*xd*)

15. Sea $E = \langle Q, \Sigma, \delta, q_0, \{q_f\}\rangle$ un AFND-λ tal queno haya transiciones hacia q_0 ni desde q_f. Describir los lenguajes que se obtienen a partir de las siguientes modificaciones:

    a) Agregar una transición λ desde q_f hacia q_0

    b) Agregar una transición λ desde q_0 hacia cada estado alcanzables desde q_0 (notar que no es sólo aquellos directos)

    c) Agregar una transición λ hacia q_F desde cada estado que tiene un camino hacia q_f

    d) El autómata obtenido haciendo b) y c)

16.