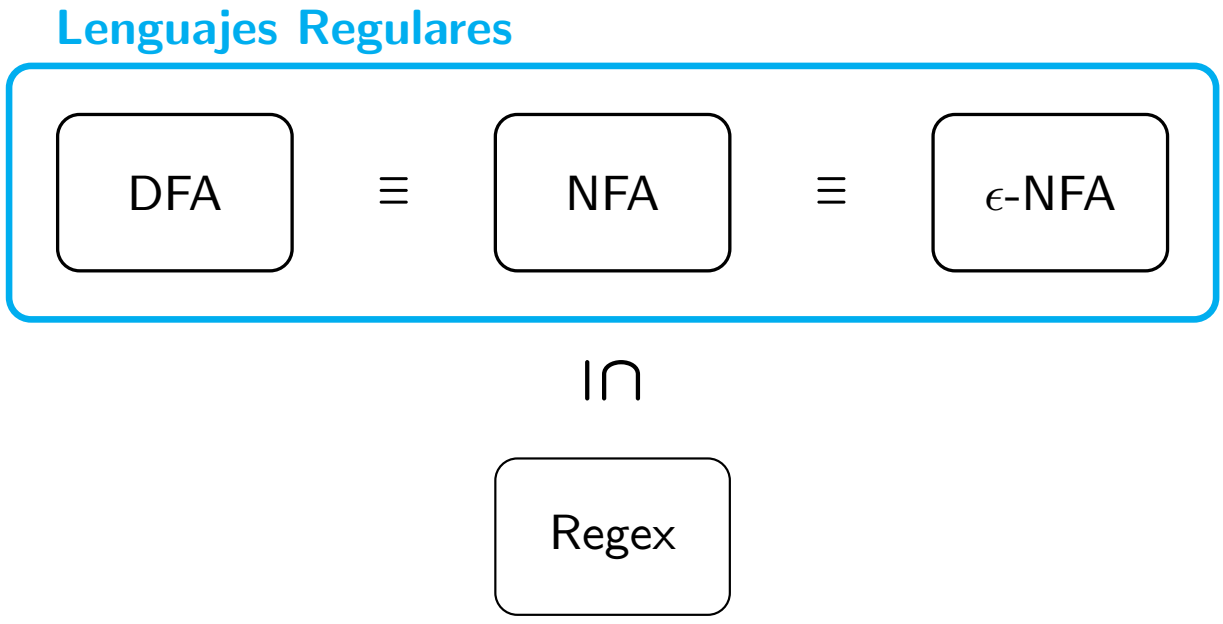
Teorema de Kleene II

Clase 08

IIC2223 / IIC2224

Prof. Cristian Riveros

Todo autómata se puede transformar en un ExpReg



Todo autómata se puede transformar en un ExpReg

Teorema

Para todo NFA $\mathcal{A}$, existe una expresión regular $R_{\mathcal{A}}$ tal que:

$$\mathcal{L}(\mathcal{A}) \;=\; \mathcal{L}(R_{\mathcal{A}})$$

Dos posibles algoritmos para demostrar este resultado:

1. Estrategia "**bottom-up**"

   Algoritmo de McNaughton-Yamada

2. Estrategia "**top-down**"

   Método de eliminación de estados

Todo autómata se puede transformar en un ExpReg

Teorema

Para todo NFA $\mathcal{A}$, existe una expresión regular $R_{\mathcal{A}}$ tal que:

$$\mathcal{L}(\mathcal{A}) \;=\; \mathcal{L}(R_{\mathcal{A}})$$

Dos posibles algoritmos para demostrar este resultado:

1. Estrategia "**bottom-up**"

   Algoritmo de McNaughton-Yamada

   No lo veremos en clases
   (Ver libros o slides al final)

2. Estrategia "**top-down**"

   Método de eliminación de estados

Método de eliminación de estados

Presentación en dos etapas:

1. Autómatas finitos no-deterministas **generalizados** (GNFA).

2. Método de **eliminación de estados** de un GNFA.

Outline

NFAs generalizados

Eliminación de estados

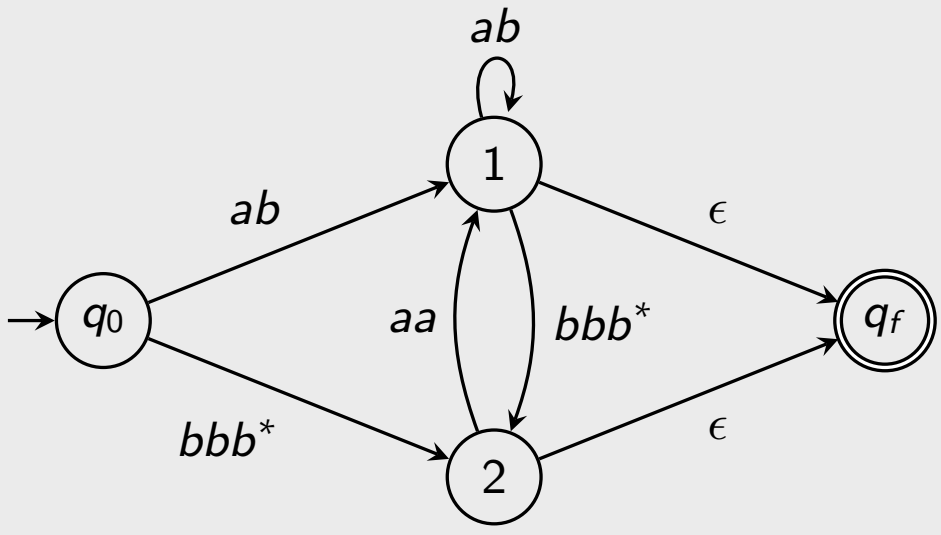
Outline

NFAs generalizados

Eliminación de estados

Autómatas finitos no-deterministas generalizados (GNFA)

Ejemplo de GNFA



GNFAs pueden leer **varias letras** en una sola transición.

Autómatas finitos no-deterministas generalizados (GNFA)

Definición

Un autómata finito no-determinista **generalizado** (GNFA) es:

$$\mathcal{G} \quad = \quad (Q, \Sigma, \delta, q_0, q_f)$$

- Q es un conjunto finito de estados.
- Σ es el alfabeto de input.

$+$

- $q_0 \in Q$ es el único estado inicial.
- $q_f \in Q$ es el único estado final y $q_f \neq q_0$.
- $\delta : (Q - \{q_f\}) \times (Q - \{q_0\}) \rightharpoonup \text{Regex}_\Sigma$ es la función de transición.

¿cómo ejecuto mi GNFA?

Sea $\mathcal{G} = (Q, \Sigma, \delta, q_0, q_f)$ un GNFA.

Definiciones

- Un par $(q, w) \in Q \times \Sigma^*$ es una **configuración** de $\mathcal{G}$.
- Una configuración (q_0, w) es **inicial**.
- Una configuración (q_f, ϵ) es **final**.

¿cómo ejecuto mi GNFA?

Sea $\mathcal{G} = (Q, \Sigma, \delta, q_0, q_f)$ un GNFA.

Definiciones

- Un par $(q, w) \in Q \times \Sigma^*$ es una **configuración** de $\mathcal{G}$.
- Se define la relación $\vdash_{\mathcal{G}}$ de **siguiente-paso** entre configuraciones de $\mathcal{G}$:

$$(p, u) \vdash_{\mathcal{G}} (q, v)$$

  si, y solo si, $\delta(p, q) = R$ y $u = w \cdot v$ con $w \in \mathcal{L}(R)$.

- Se define $\vdash_{\mathcal{G}}^*$ como la clausura **refleja** y **transitiva** de $\vdash_{\mathcal{G}}$:

$$\text{para toda configuración } (p, u): \quad (p, u) \vdash_{\mathcal{G}}^* (p, u)$$

$$\text{si } (p, u) \vdash_{\mathcal{G}} (q, v) \text{ y } (q, v) \vdash_{\mathcal{G}}^* (r, w): \quad (p, u) \vdash_{\mathcal{G}}^* (r, w)$$

$(p, u) \vdash_{\mathcal{G}}^* (q, v)$ si uno puede ir de (p, u) a (q, v) en **0 o más pasos**.

Lenguaje aceptado por un GNFA

Sea $\mathcal{G} = (Q, \Sigma, \delta, q_0, q_f)$ un GNFA y $w \in \Sigma^*$.

Definiciones

- $\mathcal{G}$ **acepta** w si existe una configuración **inicial** (q_0, w) y una configuración **final** (q_f, ϵ) tal que:

$$(q_0, w) \vdash_{\mathcal{G}}^{*} (q_f, \epsilon)$$

- El **lenguaje aceptado** por $\mathcal{G}$ se define como:

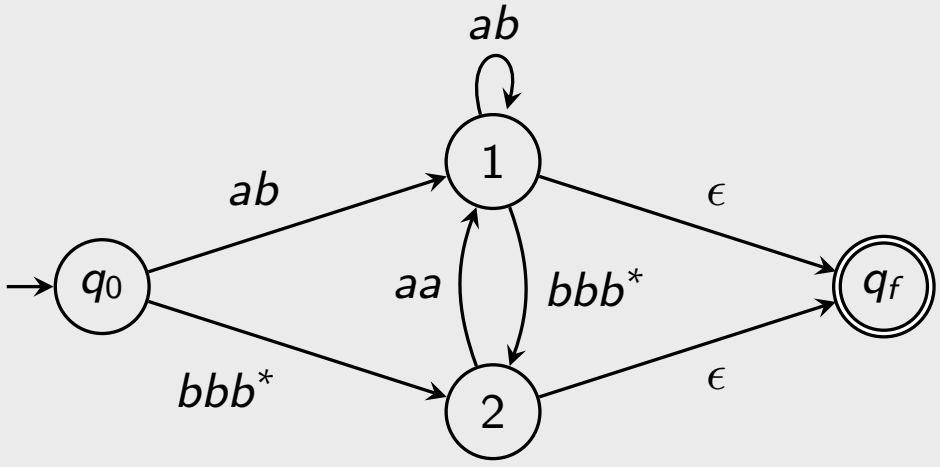
$$\mathcal{L}(\mathcal{G}) = \{w \in \Sigma^* \mid \mathcal{G} \text{ acepta } w\}$$

La misma definición de aceptación y lenguaje de un NFA o ϵ-NFA

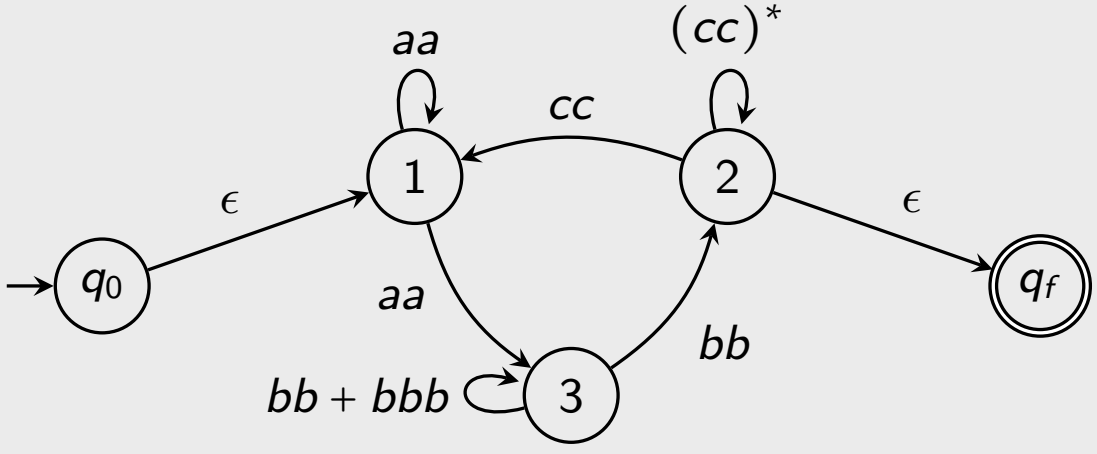
Lenguaje aceptado por un GNFA

¿qué lenguaje acepta el GNFA?

1.



2.



GNFA es otra forma de definir lenguajes regulares

Teorema

Para todo GNFA $\mathcal{G}$, existe un ϵ-NFA $\mathcal{A}$ tal que:

$$\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{A})$$

En otras palabras, GNFA $\equiv$ ϵ-NFA $\equiv$ NFA $\equiv$ DFA.

Demostración: ejercicio.

Outline

NFAs generalizados

Eliminación de estados

Método de eliminación de estados de un GNFA

input : Autómata no-determinista $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ con k estados.
output: Expresión regular equivalente a $\mathcal{A}$.

Convertir $\mathcal{A}$ en un GNFA $\mathcal{G}$ con $k + 2$ estados.

let $i := k + 2$

while $i > 2$ **do**

- Escoger un estado q de $\mathcal{G}$ distinto al inicial o al final.
- Construir el GNFA $\mathcal{G}^{-q}$, eliminando q de $\mathcal{G}$.
- **let** $\mathcal{G} := \mathcal{G}^{-q}$
- **let** $i := i - 1$

return expresión regular entre el estado inicial y final de $\mathcal{G}$

¿cómo convertimos un NFA en un GNFA?

Dado un autómata no-determinista $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, definimos el GNFA:

$$\mathcal{G} = (Q \cup \{q_0, q_f\}, \Sigma, \delta, q_0, q_f)$$

- $q_0 \notin Q$, $q_f \notin Q$ y $q_0 \neq q_f$.
- para todo $q \in I$, $\delta(q_0, q) = \epsilon$.
- para todo $q \in F$, $\delta(q, q_f) = \epsilon$.
- para todo $p, q \in Q$:

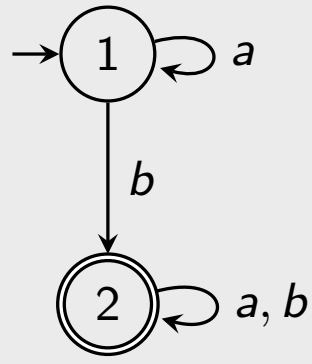
$$\delta(p, q) = \varnothing + a_1 + \ldots + a_n$$

donde $a_1, \ldots, a_n$ son todas las letras en Σ tal que $(p, a_i, q) \in \Delta$.

Ejercicio: demuestre que $\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{A})$

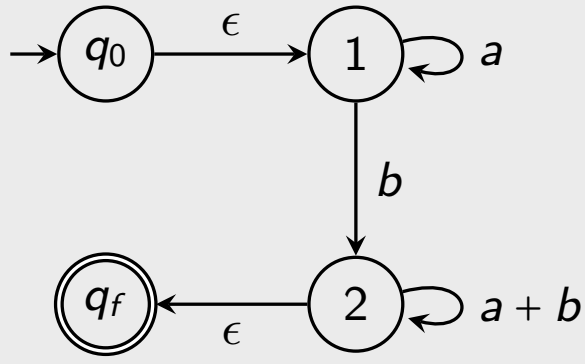
¿cómo convertimos un NFA en un GNFA?

Construya un GNFA a partir del siguiente NFA



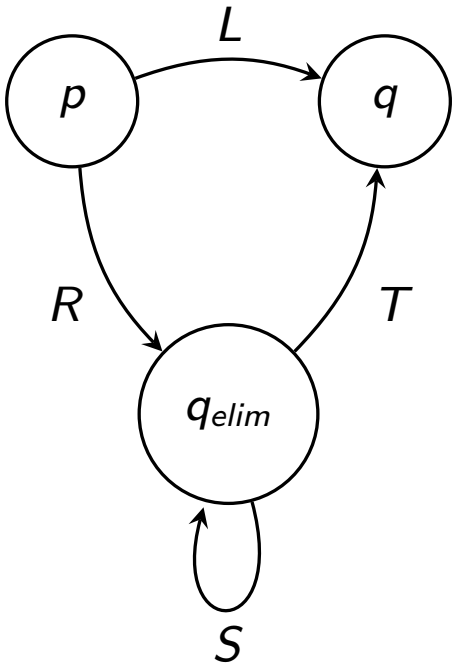
¿cómo convertimos un NFA en un GNFA?

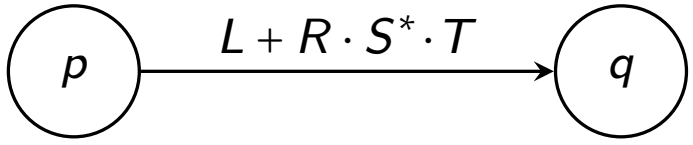
Construya un GNFA a partir del siguiente NFA



¿cómo eliminamos un estado de un GNFA?

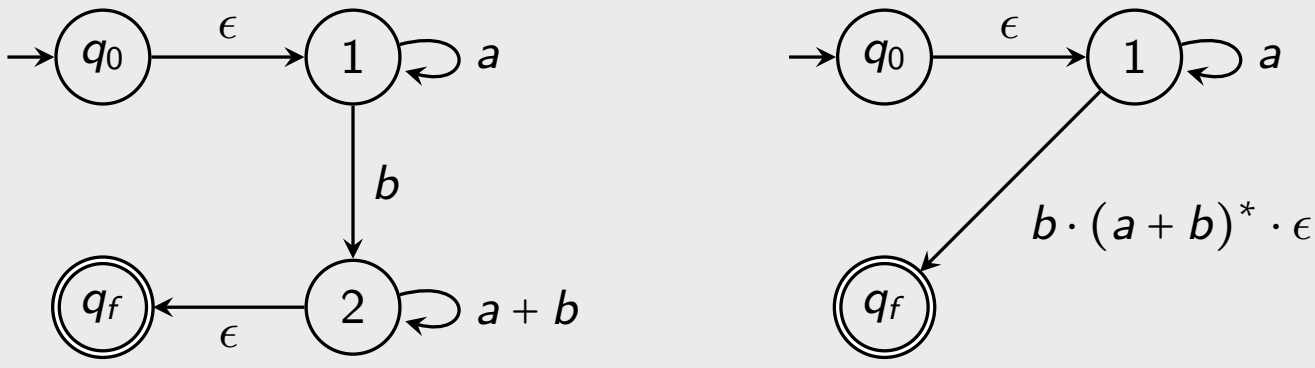
Idea





¿cómo eliminamos un estado de un GNFA?

Ejemplo



¿cómo eliminamos un estado de un GNFA?

Dado:

- un GNFA $\mathcal{G} = (Q, \Sigma, \delta, q_0, q_f)$ con $|Q| > 2$ y
- $q \in Q - \{q_0, q_f\}$

definimos el GNFA:

$$\mathcal{G}^{-q} = (Q - \{q\}, \Sigma, \delta^{-q}, q_0, q_f)$$

tal que para todo $p_1, p_2 \in Q - \{q\}$ se define δ^{-q} como:

$$\delta^{-q}(p_1, p_2) \;=\; \delta(p_1, p_2) \;+\; \delta(p_1, q) \cdot (\delta(q, q))^* \cdot \delta(q, p_2)$$

Demuestre que $\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{G}^{-q})$.

Método de eliminación de estados de un GNFA

input : Autómata no-determinista $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ con k estados.
output: Expresión regular equivalente a $\mathcal{A}$.

Convertir $\mathcal{A}$ en un GNFA $\mathcal{G}$ con $k+2$ estados.

let $i := k + 2$

while $i > 2$ **do**

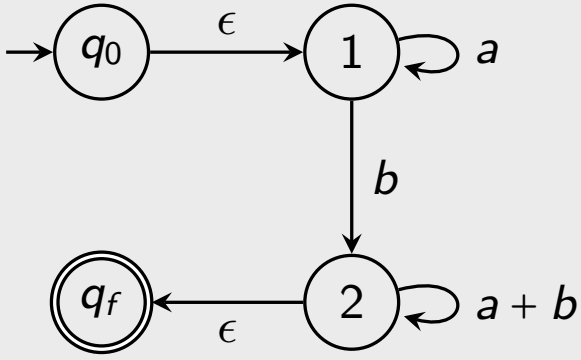
- Escoger un estado q de $\mathcal{G}$ distinto al inicial o al final.
- Construir el GNFA $\mathcal{G}^{-q}$, eliminando q de $\mathcal{G}$.
- **let** $\mathcal{G} := \mathcal{G}^{-q}$
- **let** $i := i - 1$

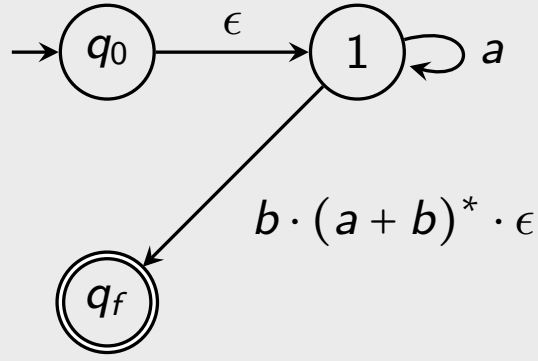
return expresión regular entre el estado inicial y final de $\mathcal{G}$

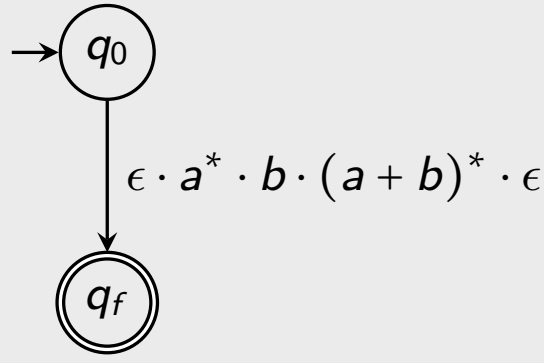
Demuestre que el algoritmo es **correcto**

Método de eliminación de estados de un GNFA

Ejemplo







Cierre de clase

En esta clase vimos:

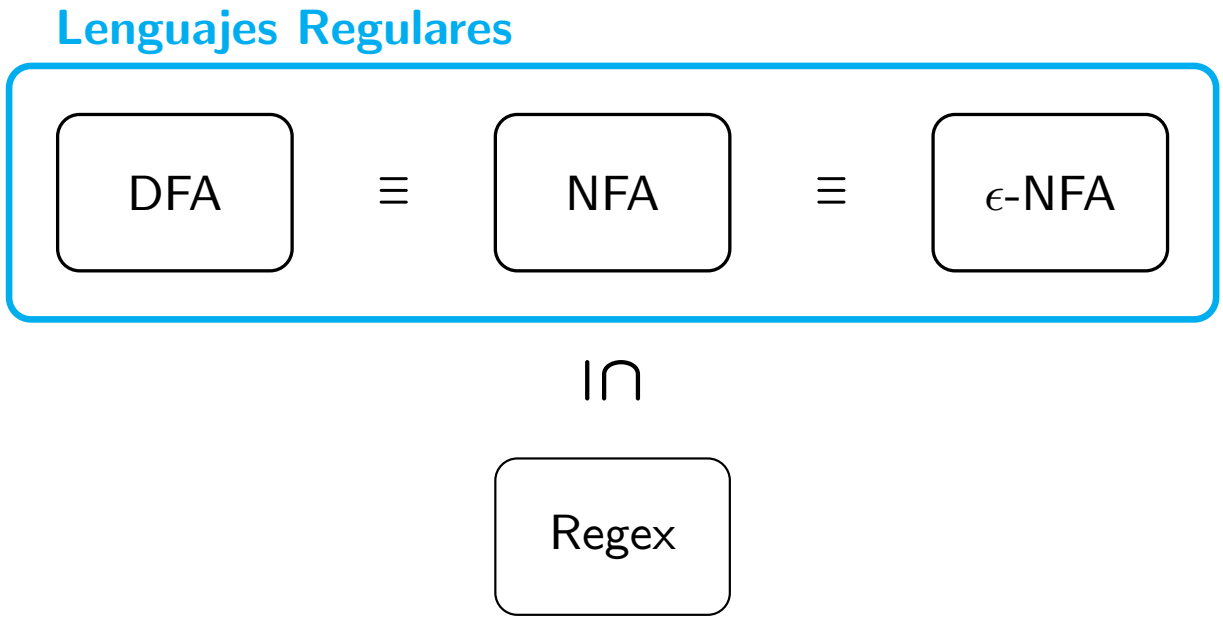
1. Teorema de Kleene: desde autómatas a regex.
2. Autómatas finitos no-deterministas generalizados.
3. Método de eliminación de estados.

Próxima clase: Lema de bombeo.

MATERIAL ADICIONAL:

Algoritmo de McNaughton-Yamada

Todo autómata se puede transformar en un ExpReg



¿cómo obtener una expresión regular desde un autómata?

Dado un autómata finito no-determinista (spdg. con un estado inicial):

$$\mathcal{A} \;=\; (Q, \Sigma, \Delta, q_0, F)$$

Para $X \subseteq Q$ y $p, q \in Q$, considerar el conjunto:

$$\alpha^X_{p,q} \;\subseteq\; \Sigma^*$$

$w = a_1 \ldots a_n \;\in\; \alpha^X_{p,q}$ si, y solo si, existe una **ejecución**:

$$\rho : p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} p_n$$

1. $(p_i, a_{i+1}, p_{i+1}) \in \Delta$ para todo $i \in [0, n-1]$,
2. $p_0 = p$,
3. $p_n = q$, y
4. $p_i \in X$ para todo $i \in [1, n-1]$.

¿cómo obtener una expresión regular desde un autómata?

Dado un autómata finito no-determinista (spdg. con un estado inicial):

$$\mathcal{A} \;=\; (Q, \Sigma, \Delta, q_0, F)$$

Para $X \subseteq Q$ y $p, q \in Q$, considerar el conjunto:

$$\alpha^{X}_{p,q} \;\subseteq\; \Sigma^*$$

"el conjunto de todas las palabras w tal que existe un **camino** *(i.e. ejecución) desde p a q etiquetado por w y* **todos los estados** *en este camino están en X, con la posible* **excepción** *de p y q."*

¿cómo obtener una expresión regular desde un autómata?

Dado un autómata finito no-determinista (spdg. con un estado inicial):

$$\mathcal{A} \;=\; (Q, \Sigma, \Delta, q_0, F)$$

Para $X \subseteq Q$ y $p, q \in Q$, considerar el conjunto:

$$\alpha^X_{p,q} \;\subseteq\; \Sigma^*$$

¿cómo podemos definir $\mathcal{L}(\mathcal{A})$ en términos de $\alpha^X_{p,q}$?

Lema

$$\mathcal{L}(\mathcal{A}) \;=\; \bigcup_{q \in F} \alpha^Q_{q_0,q}$$

¿cómo obtener una expresión regular desde un autómata?

Dado un autómata finito no-determinista (spdg. con un estado inicial):

$$\mathcal{A} \;=\; (Q, \Sigma, \Delta, q_0, F)$$

Estrategia (algoritmo de McNaughton-Yamada)

1. Para cada $\alpha^X_{p,q}$, definir **inductivamente** una expresión regular $R^X_{p,q}$:

$$\mathcal{L}(R^X_{p,q}) \;=\; \alpha^X_{p,q}$$

2. Para $F = \{p_1, \ldots, p_k\}$ definir la **expresión regular**:

$$R_{\mathcal{A}} \;=\; R^Q_{q_0,p_1} + R^Q_{q_0,p_2} + \ldots + R^Q_{q_0,p_k}$$

3. Demostrar que:

$$\mathcal{L}(\mathcal{A}) \;=\; \mathcal{L}(R_{\mathcal{A}})$$

Definición inductiva de $R^X_{p,q}$

Caso base: $X = \varnothing$

Sea $a_1, \ldots, a_k \in \Sigma$ todos las letras tal que:

$$(p, a_i, q) \in \Delta$$

- Si $p \neq q$, entonces:

$$R^{\varnothing}_{p,q} \stackrel{\text{def}}{\equiv} \begin{cases} a_1 + \cdots + a_k & \text{si } k \geq 1 \\ \varnothing & \text{si } k = 0 \end{cases}$$

- Si $p = q$, entonces:

$$R^{\varnothing}_{p,q} \stackrel{\text{def}}{\equiv} \begin{cases} a_1 + \cdots + a_k + \epsilon & \text{si } k \geq 1 \\ \epsilon & \text{si } k = 0 \end{cases}$$

Definición inductiva de $R^X_{p,q}$

Caso general: $X \neq \emptyset$

Por **inducción**, suponemos que para todo $r, s \in Q$ y para todo $Y \subset X$, $R^Y_{r,s}$ es una expresión regular tal que:

$$\mathcal{L}(R^Y_{r,s}) \;=\; \alpha^Y_{r,s}$$

Demostramos la construcción para $R^X_{p,q}$ con $p, q \in Q$. Sea $r \in X$ cualquiera:

$$R^X_{p,q} \;\stackrel{\text{def}}{\equiv}\; R^{X-\{r\}}_{p,q} \;+\; R^{X-\{r\}}_{p,r} \cdot \left(R^{X-\{r\}}_{r,r}\right)^* \cdot R^{X-\{r\}}_{r,q}$$

Proposición

Para todo $X \subseteq Q$ y $p, q \in Q$:

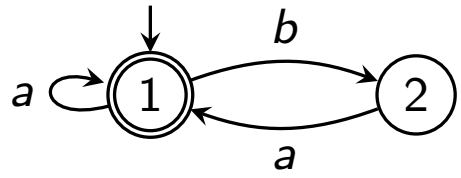
$$\mathcal{L}(R^X_{p,q}) \;=\; \alpha^X_{p,q}$$

Corolario

$$\mathcal{L}(\mathcal{A}) \;=\; \mathcal{L}(R_{\mathcal{A}})$$

Ejemplo de Algoritmo MNY

Considere el autómata:



$R^{\emptyset}$	1	2
1	$a^?$	b
2	a	ϵ

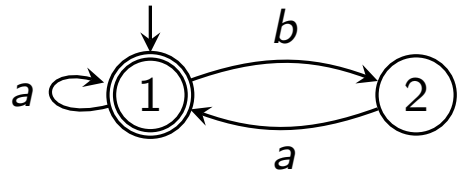
$R^{\{1\}}$	1	2
1	$a^? + a^? \cdot (a^?)^* \cdot a^?$	$b + a^? \cdot (a^?)^* \cdot b$
2	$a + a \cdot (a^?)^* \cdot a^?$	$\epsilon + a \cdot (a^?)^* \cdot b$

$\equiv$

$R^{\{1\}}$	1	2
1	a^*	$a^* b$
2	a^+	$\epsilon + a^+ b$

Ejemplo de Algoritmo MNY

Considere el autómata:



$R^{\{1\}}$	1	2
1	a^*	a^*b
2	a^+	$\epsilon + a^+b$

$R^{\{1,2\}}$	1	2
1	$a^* + a^*b\,(\epsilon + a^+b)^*a^+$	$a^*b + a^*b\,(\epsilon + a^+b)^*(\epsilon + a^+b)$
2	$a^+ + (\epsilon + a^+b)(\epsilon + a^+b)^*a^+$	$(\epsilon + a^+b) + (\epsilon + a^+b)(\epsilon + a^+b)^*(\epsilon + a^+b)$

$\equiv$

$R^{\{1,2\}}$	1	2
1	$a^*(ba^+)^*$	$(a^*b)(a^+b)^*$
2	$(a^+b)^*a^+$	$(a^+b)^*$