

Determinización de autómatas no-deterministas

Clase 04

IIC2223 / IIC2224

Prof. Cristian Riveros

Outline

Recordatorio clase pasada

Determinización de un NFA

Outline

Recordatorio clase pasada

Determinización de un NFA

Autómata finito no-determinista

Definición

Un autómata finito **no-determinista** (NFA) es una estructura:

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

- Q es un conjunto finito de estados.
- Σ es el alfabeto de input.
- $F \subseteq Q$ es el conjunto de estados finales (o aceptación).

+

- $\Delta \subseteq Q \times \Sigma \times Q$ es la **relación de transición**.
- $I \subseteq Q$ es un **conjunto de estados iniciales**.

¿cómo ejecuto un autómata no-determinista?

Sea:

- Un autómata finito no-determinista $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$.
- El input $w = a_1 a_2 \dots a_n \in \Sigma^*$.

Una **ejecución** (o run) ρ de \mathcal{A} sobre w es una secuencia:

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

- $p_0 \in I$
- para todo $i \in \{0, \dots, n-1\}$, $(p_i, a_{i+1}, p_{i+1}) \in \Delta$.

Una ejecución ρ de \mathcal{A} sobre w es de **aceptación** si:

$$p_n \in F.$$

Desde ahora hablaremos de **las ejecuciones** de \mathcal{A} sobre w

Lenguaje aceptado por un autómata no-determinista

Sea un autómata $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ y $w \in \Sigma^*$.

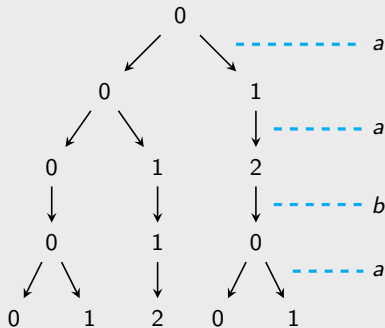
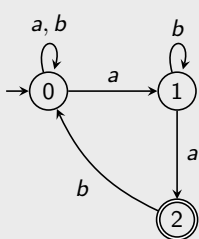
Definiciones

- \mathcal{A} **acepta** w si **existe** una ejecución de \mathcal{A} sobre w que es de aceptación.
- \mathcal{A} **rechaza** w si **todas** las ejec. de \mathcal{A} sobre w **NO** son de aceptación.
- El **lenguaje aceptado** por \mathcal{A} se define como:

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

¿qué tan poderoso es el no-determinismo en autómatas?

Ejemplo



¿puede un autómata determinista almacenar **todas** las ejecuciones?

Outline

Recordatorio clase pasada

Determinización de un NFA

¿qué tan poderoso es el no-determinismo en autómatas?

Teorema

Para todo autómata finito no-determinista \mathcal{A} , existe un autómata determinista \mathcal{A}' tal que:

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

En otras palabras, DFA \equiv NFA.

Ambos modelos computan lo mismo

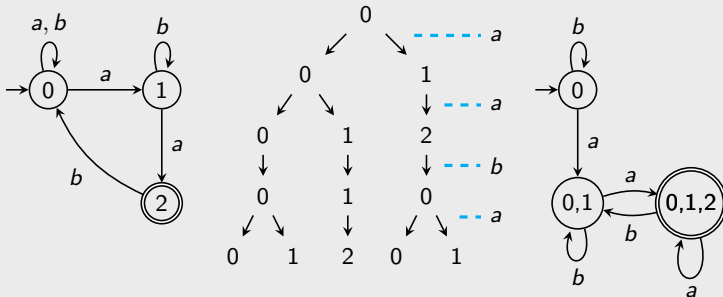
Demostración

Para demostrar este resultado, construiremos la “determinación” del autómata no-determinista \mathcal{A} .

Idea de determinización

“Almacenar en el autómata determinista todos los estados actuales de las ejecuciones en curso (sin repetidos).”

Ejemplo



Determinización

Formalización

Para un autómata no-determinista $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, definimos el autómata determinista (**determinización** de \mathcal{A}):

$$\mathcal{A}^{\text{det}} = (2^Q, \Sigma, \delta^{\text{det}}, q_0^{\text{det}}, F^{\text{det}})$$

■ $2^Q = \{S \mid S \subseteq Q\}$ es el conjunto potencia de Q

■ $q_0^{\text{det}} = I$

■ $\delta^{\text{det}} : 2^Q \times \Sigma \rightarrow 2^Q$ tal que:

$$\delta^{\text{det}}(S, a) = \{q \in Q \mid \exists p \in S. (p, a, q) \in \Delta\}$$

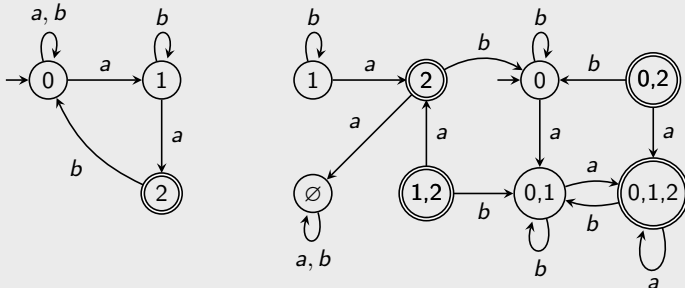
■ $F^{\text{det}} = \{S \in 2^Q \mid S \cap F \neq \emptyset\}$

Determinización

Formalización

- $2^Q = \{S \mid S \subseteq Q\}$ es el conjunto potencia de Q
- $q_0^{\text{det}} = I$
- $\delta^{\text{det}}(S, a) = \{q \in Q \mid \exists p \in S. (p, a, q) \in \Delta\}$
- $F^{\text{det}} = \{S \in 2^Q \mid S \cap F \neq \emptyset\}$

Ejemplo determinización



Determinización: correctitud

Proposición

Dado un autómata no-determinista $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ se tiene que:

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}^{\text{det}})$$

¿cómo demostramos que ambos autómatas definen el mismo lenguaje?

Determinización: correctitud

Demostración: $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}^{\text{det}})$

Sea $w = a_1 a_2 \dots a_n \in \mathcal{L}(\mathcal{A})$.

Existe una ejecución ρ de \mathcal{A} sobre w :

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

- $p_0 \in I$.
- $(p_i, a_{i+1}, p_{i+1}) \in \Delta \quad \forall i \in \{0, \dots, n-1\}$.
- $p_n \in F$.

Como \mathcal{A}^{det} es determinista, entonces existe una ejec. ρ' de \mathcal{A}^{det} sobre w :

$$\rho' : S_0 \xrightarrow{a_1} S_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} S_n$$

- $S_0 = I$.
- $\delta^{\text{det}}(S_i, a_{i+1}) = S_{i+1} \quad \forall i \in \{0, 1, \dots, n-1\}$.

¿qué debemos demostrar?

Determinización: correctitud

Demostración: $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}^{\text{det}})$

PD: $p_i \in S_i$ para todo $i \in \{0, 1, \dots, n-1\}$.

Por **inducción** sobre i .

Caso base: $p_0 \in S_0$ ✓

(¿por qué?)

Inducción: Suponemos que $p_i \in S_i$ y demostramos para $i+1$.

Como sabemos que:

■ $\delta^{\text{det}}(S_i, a_{i+1}) = S_{i+1} = \{q \in Q \mid \exists p \in S_i. (p, a, q) \in \Delta\}$ y

■ $(p_i, a_{i+1}, p_{i+1}) \in \Delta$.

Entonces $p_{i+1} \in S_{i+1}$ (¿por qué?). ✓

Como $p_n \in S_n \xrightarrow{?} S_n \cap F \neq \emptyset \xrightarrow{?} S_n \in F^{\text{det}}$.

Por lo tanto, $w \in \mathcal{L}(\mathcal{A}^{\text{det}})$.

Determinización: correctitud

Demostración: $\mathcal{L}(\mathcal{A}^{\text{det}}) \subseteq \mathcal{L}(\mathcal{A})$

Sea $w = a_1 a_2 \dots a_n \in \mathcal{L}(\mathcal{A}^{\text{det}})$.

Existe una ejecución ρ de \mathcal{A}^{det} sobre w :

$$\rho : S_0 \xrightarrow{a_1} S_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} S_n$$

■ $S_0 = I$.

■ $\delta^{\text{det}}(S_i, a_{i+1}) = S_{i+1} \quad \forall i \in \{0, 1, \dots, n-1\}$.

■ $S_n \in F^{\text{det}}$.

$$(S_n \cap F \neq \emptyset)$$

¿cómo demostramos una **ejecución de aceptación** de \mathcal{A} sobre w ?

Determinización: correctitud


Demostración: $\mathcal{L}(\mathcal{A}^{\text{det}}) \subseteq \mathcal{L}(\mathcal{A})$

PD: Para todo $i \leq n$ y para todo $p \in S_i$, existe una ejecución:

$$\rho : p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_i} p_i = p$$

1. $p_0 \in I$.
2. $(p_j, a_{j+1}, p_{j+1}) \in \Delta \quad \forall j \in \{0, \dots, i-1\}$.

Por **inducción** sobre i .

Caso base: Si $p \in S_0 = I$, entonces la ejec. $\rho : p$ cumple 1. y 2. 

Determinización: correctitud

Demostración: $\mathcal{L}(\mathcal{A}^{\text{det}}) \subseteq \mathcal{L}(\mathcal{A})$

PD: Para todo $i \leq n$ y para todo $p \in S_i$, existe una ejecución:

$$\rho : p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_i} p_i = p$$

1. $p_0 \in I$.
2. $(p_j, a_{j+1}, p_{j+1}) \in \Delta \quad \forall j \in \{0, \dots, i-1\}$.

Inducción: Supongamos que se cumple para todo $p \in S_i$. Sea $q \in S_{i+1}$.

Como $\delta^{\text{det}}(S_i, a_{i+1}) = S_{i+1} = \{q \in Q \mid \exists p \in S_i. (p, a, q) \in \Delta\}$ y $q \in S_{i+1}$
entonces existe $p \in S_i$ tal que $(p, a_{i+1}, q) \in \Delta$.

Por **HI**, existe $\rho : p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_i} p_i = p$ que satisface 1. y 2.

Por lo tanto, $\rho' : p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_i} p_i \xrightarrow{a_{i+1}} q$ también satisface 1. y 2. ✓

Determinización: correctitud

Demostración: $\mathcal{L}(\mathcal{A}^{\text{det}}) \subseteq \mathcal{L}(\mathcal{A})$

Por lo tanto: Para todo $i \leq n$ y para todo $p \in S_i$, existe una ejecución:

$$\rho : p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_i} p_i = p$$

1. $p_0 \in I$.
2. $(p_j, a_{j+1}, p_{j+1}) \in \Delta \quad \forall j \in \{0, \dots, i-1\}$.

Como $S_n \cap F \neq \emptyset$,

(¿por qué?)

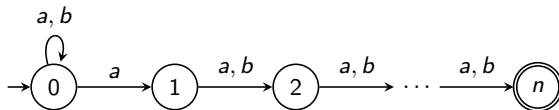
para $p \in S_n \cap F$ existe una ejecución de acept. de \mathcal{A} sobre w .

Por lo tanto, $w \in \mathcal{L}(\mathcal{A})$. 

¿cuál es la ventaja de los autómatas no-deterministas?

Ventajas

1. Su representación es más sencilla para algunos lenguajes.
2. Son **exponencialmente** más compactos.



- La determinización tiene 2^n estados.
- Es posible demostrar que no hay DFA con menos estados.

Cierre de clase

En esta clase vimos:

1. Como convertir un NFA en un DFA.
2. Demostramos la correctitud de la construcción.
3. NFA son exponencialmente más sucintos.

Próxima clase: Expresiones regulares