Enumeración de resultados: Autómatas con anotaciones

Clase 26

IIC 2223

Prof. Cristian Riveros

Extracción de información (recordatorio)

```
18:30 ERROR 06
19:10 OK 00
20:00 ERROR 19
```

"Obtener todas las horas HH:MM"

R := (\d\d:\d\d)

¿cómo podemos **automatizar** esta tarea de extraer datos?

18:30		ERROR		06↵	19:10		OK		00↵	20:00		ERROR		19

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40

Extracción de información

La tarea de automatizar la extracción de datos
desde documentos no estructurados o semi-estructurados.

Representación de intervalos: spans (recordatorio)

Definición

- Para un documento $d = a_0 a_1 \ldots a_{n-1}$ se define un **span** s de d como:

$$s \;=\; [i, j\rangle$$

  tal que $0 \leq i \leq j \leq n$.

- Si $s = [i, j\rangle$ es un span de d se define:

$$d[s] \;=\; d[[i, j\rangle] \;=\; a_i a_{i+1} \ldots a_{j-1}$$

  como el **contenido del span** s en d. Si $i = j$, entonces $d[[i, i\rangle] = \epsilon$.

Ejemplo

```
                   s1                            s2
18:30 ERROR 06↲19:10 OK 00↲20:00 ERROR 19
```

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40

$s_1 = [15, 20\rangle \quad w[s_1] = 19{:}10 \qquad s_2 = [32, 32\rangle \quad w[s_2] = \epsilon$

Mappings como outputs de regex (recordatorio)

Definiciones

- Un **mapping** de R sobre d es una función $\mu : \text{Var}(R) \to \text{Spans}(d)$.
- Se define el mapping $\mu = \perp$ como el **mapping vacío** donde $\text{dom}(\perp) = \emptyset$.
- Para una variable $\mathbf{x}$ y span s se define el **mapping de una variable**:

$$\mu = [\mathbf{x} \mapsto s] \qquad \text{tal que} \quad \text{dom}(\mu) = \{\mathbf{x}\} \quad \text{y} \quad \mu(x) = s$$

- Para $k \in \mathbb{N}$ se define el **mapping** $\mu + k$ tal que $\text{dom}(\mu + k) = \text{dom}(\mu)$ y:

$$\text{si} \quad \mu(\mathbf{x}) = [i, j\rangle \quad \text{entonces} \quad [\mu + k](\mathbf{x}) = [i + k, j + k\rangle.$$

- Para mappings μ_1, μ_2 con $\text{dom}(\mu_1) \cap \text{dom}(\mu_2) = \emptyset$ se define **la únion**:

$$\mu = \mu_1 \cup \mu_2 \qquad \text{tal que} \quad \mu(\mathbf{x}) = \begin{cases} \mu_1(\mathbf{x}) & \text{si } \mathbf{x} \in \text{dom}(\mu_1) \\ \mu_2(\mathbf{x}) & \text{si } \mathbf{x} \in \text{dom}(\mu_2) \end{cases}$$

Cada regex define un conjunto de mappings (recordatorio)

Semántica regex (completa)

Para una regex válida R cualquiera,
se define la función $\llbracket R \rrbracket$ **inductivamente** sobre documentos $d \in \Sigma^*$:

1. $\llbracket a \rrbracket(d) = \{\bot\}$ si $d = a$, y $\varnothing$ en otro caso
2. $\llbracket \epsilon \rrbracket(d) = \{\bot\}$ si $d = \epsilon$, y $\varnothing$ en otro caso
3. $\llbracket \mathbf{x}\{R_1\} \rrbracket(d) = \left\{ \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in \llbracket R_1 \rrbracket(d) \text{ y } s = [0, |d|\rangle \right\}$
4. $\llbracket R_1 \cdot R_2 \rrbracket(d) = \left\{ \mu_1 \cup (\mu_2 + |d_1|) \;\middle|\; \begin{array}{l} \text{existe } d_1, d_2 \text{ tal que } d = d_1 \cdot d_2, \\ \mu_1 \in \llbracket R_1 \rrbracket(d_1) \text{ y } \mu_2 \in \llbracket R_2 \rrbracket(d_2) \end{array} \right\}$
5. $\llbracket R_1 + R_2 \rrbracket(d) = \llbracket R_1 \rrbracket(d) \cup \llbracket R_2 \rrbracket(d)$
6. $\llbracket R_1^* \rrbracket(d) = \bigcup_{k=0}^{\infty} \llbracket (R_1)^k \rrbracket(d)$

Autómata con variables (recordatorio)

Definición

Un vset automata (VA) es una tupla:

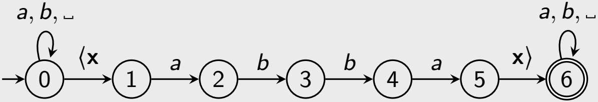
$$\mathcal{A} \quad = \quad (Q, \Sigma, \mathcal{X}, \Delta, I, F)$$

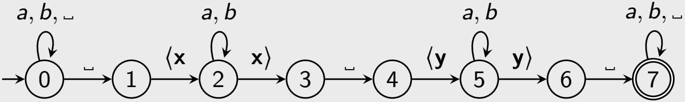
- Q es un conjunto finito de estados.
- Σ es el alfabeto de input.
- $\mathcal{X}$ es un conjunto finito de variables.
- $\Delta \subseteq Q \times \big(\Sigma \cup \{\epsilon\} \cup \{\langle \mathbf{x}, \mathbf{x}\rangle \mid \mathbf{x} \in \mathcal{X}\}\big) \times Q$ es la relación de transición.
- $I \subseteq Q$ es un conjunto de estados iniciales.
- $F \subseteq Q$ es el conjunto de estados finales (o aceptación).

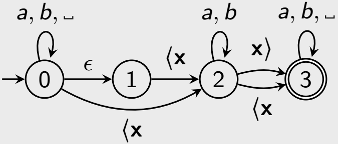
'$\langle \mathbf{x}$' simboliza **abrir** y '$\mathbf{x}\rangle$' simboliza **cerrar** la variable **x**

Autómata con variables (recordatorio)

Ejemplos de vset autómata

1. 

2. 

3. 

Ejecución de un vset autómata (recordatorio)

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un VA y $d = a_0 \dots a_{n-1} \in \Sigma^*$ un documento.

Definiciones

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

$$\rho : (p_0, i_0) \xrightarrow{o_1} (p_1, i_1) \xrightarrow{o_2} \dots \xrightarrow{o_m} (p_m, i_m)$$

tal que cumple todas las siguientes condiciones:

- $o_k \in \Sigma \cup \{\epsilon\} \cup \{\langle \mathbf{x}, \mathbf{x} \rangle \mid \mathbf{x} \in \mathcal{X}\}$ con $k \leq m$
- (p_0, i_0) es una configuración inicial
- para todo $k < m$, $(p_k, o_{k+1}, p_{k+1}) \in \Delta$
- para todo $k \leq m$, si $o_k \in \Sigma$, entonces $o_k = a_{i_{k-1}}$ y $i_k = i_{k-1} + 1$
- para todo $k \leq m$, si $o_k \in \{\epsilon\} \cup \{\langle \mathbf{x}, \mathbf{x} \rangle \mid \mathbf{x} \in \mathcal{X}\}$, entonces $i_k = i_{k-1}$.

Una ejecución ρ es de **aceptación** si (p_m, i_m) es de aceptación.

Ejecución válida y el mapping que define (recordatorio)

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un VA y $d = a_0 \ldots a_{n-1} \in \Sigma^*$ un documento.

Definiciones

Para una ejecución ρ de $\mathcal{A}$ sobre d:

$$\rho : (p_0, i_0) \xrightarrow{o_1} (p_1, i_1) \xrightarrow{o_2} \ldots \xrightarrow{o_m} (p_m, i_m)$$

decimos que ρ es **válida** si, y solo si, para todo $\mathbf{x} \in \mathcal{X}$:

- existe un único $k_1 \leq m$ tal que $o_{k_1} = \langle \mathbf{x}$
- existe un único $k_2 \leq m$ tal que $o_{k_2} = \mathbf{x} \rangle$ y
- $k_1 < k_2$.

Si ρ es **válido** se define el **mapping de ρ** $\text{map}(\rho) : \mathcal{X} \to \text{Spans}(d)$ tal que:

$$[\text{map}(\rho)](\mathbf{x}) = [i_{k_1}, i_{k_2} \rangle$$

para todo $\mathbf{x} \in \mathcal{X}$ y $k_1, k_2 \leq m$ con $o_{k_1} = \langle \mathbf{x}$ y $o_{k_2} = \mathbf{x} \rangle$.

Función de extracción de un vset autómata (recordatorio)

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un vset autómata.

Definición

Se define la función $[\![\mathcal{A}]\!]$ tal que para todo documento $d \in \Sigma^*$:

$$[\![\mathcal{A}]\!](d) \;=\; \left\{ \text{map}(\rho) \;\middle|\; \begin{array}{l} \rho \text{ es una ejecución} \\ \textbf{válida} \text{ y de } \textbf{aceptación} \text{ de } \mathcal{A} \text{ sobre } d \end{array} \right\}$$

VA nos entrega otra forma de extraer información de un documento.

Vset automata funcionales

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un vset autómata.

$$[\![\mathcal{A}]\!](d) \;=\; \left\{ \text{map}(\rho) \;\middle|\; \begin{array}{l} \rho \text{ es una ejecución} \\ \textbf{válida} \text{ y de } \textbf{aceptación} \text{ de } \mathcal{A} \text{ sobre } d \end{array} \right\}$$

Definición

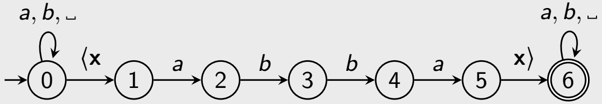
Decimos que un vset autómata $\mathcal{A}$ es **funcional** si, y solo si, para todo documento d y para toda ejecución ρ de $\mathcal{A}$ sobre d:

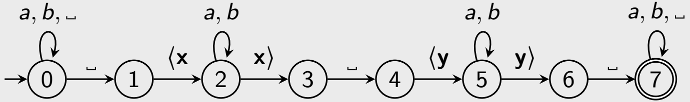
si ρ es de **aceptación**, entonces ρ es **válida**.

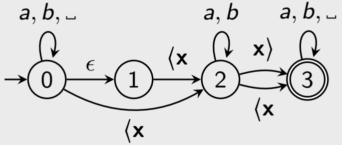
Para funcional solo necesitamos verificar que la ejecución es de aceptación.

Vset automata funcionales

¿cuáles VA son funcionales?

1. 

2. 

3. 

Desde regex a vset autómata

Teorema

Para toda regex R válida, existe un vset autómata **funcional** $\mathcal{A}_R$ de **tamaño lineal** en $|R|$ tal que para todo documento d:

$$[\![R]\!](d) = [\![\mathcal{A}_R]\!](d)$$

Demostración: ejercicio

¿és verdad la otra dirección?

¿cómo evaluamos una regex?

`PROBLEMA:`	Evaluación de regex
`INPUT:`	una regex R y
	un documento d
`OUTPUT:`	Enumerar todos los mappings en $[\![R]\!](d)$

1. Transformamos R a un vset autómata $\mathcal{A}_R$.
2. Enumeramos los resultados $[\![\mathcal{A}_R]\!](d)$.

¿cómo computamos todos los mappings en $[\![\mathcal{A}_R]\!](d)$?

…y ¿cómo los encontramos si son demasiados?

Outline

Representación de mappings

Autómata con anotaciones

Desde Vset a AnnA

Determinismo

Outline

Índice invertido y secuenciación de un mapping

Sea $d = a_0 \dots a_{n-1}$, un conj. de variables $\mathcal{X}$ y un mapping $\mu : \mathcal{X} \to \mathrm{Spans}(d)$.

Definiciones

1. Se define el **conjunto de marcas de $\mathcal{X}$** como:

$$\mathrm{Markers}(\mathcal{X}) \; = \; \{ \langle \mathbf{x} \mid \mathbf{x} \in \mathcal{X} \} \cup \{ \mathbf{x}\rangle \mid \mathbf{x} \in \mathcal{X} \}$$

2. Se define el **mapping inverso** de μ como $\mu^{\mathrm{inv}} : [0, n] \to 2^{\mathrm{Markers}(\mathcal{X})}$:

$$\mu^{\mathrm{inv}}(i) \; = \; \{ \langle \mathbf{x} \mid \exists j.\ \mu(\mathbf{x}) = [i, j\rangle \in \mathcal{X} \} \cup \{ \mathbf{x}\rangle \mid \exists j.\ \mu(\mathbf{x}) = [j, i\rangle \in \mathcal{X} \}$$

Ejemplo

```
                   x     z      y
18:30 ERROR 06↵19:10 OK 00↵20:00 ERROR 19
```

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40

$\mu(\mathbf{x}) = [15, 20\rangle$ $\qquad \mu^{\mathrm{inv}}(15) = \{ \langle \mathbf{x}, \langle \mathbf{z} \}$ $\qquad \mu^{\mathrm{inv}}(20) = \{ \mathbf{x}\rangle \}$

$\mu(\mathbf{y}) = [24, 26\rangle$ $\qquad \mu^{\mathrm{inv}}(24) = \{ \langle \mathbf{y} \}$ $\qquad \mu^{\mathrm{inv}}(26) = \{ \mathbf{y}\rangle, \mathbf{z}\rangle \}$

$\mu(\mathbf{z}) = [15, 26\rangle$ $\qquad \forall i \notin \{15, 20, 24, 26\}$ $\qquad \mu^{\mathrm{inv}}(i) = \emptyset$

Índice invertido y secuenciación de un mapping

Sea $d = a_0 \dots a_{n-1}$, un conj. de variables $\mathcal{X}$ y un mapping $\mu : \mathcal{X} \to \text{Spans}(d)$.

Definiciones

1. Se define el **conjunto de marcas de $\mathcal{X}$** como:

$$\text{Markers}(\mathcal{X}) \;=\; \big\{ \langle \mathbf{x} \mid \mathbf{x} \in \mathcal{X} \big\} \cup \big\{ \mathbf{x}\rangle \mid \mathbf{x} \in \mathcal{X} \big\}$$

2. Se define el **mapping inverso** de μ como $\mu^{\text{inv}} : [0, n] \to 2^{\text{Markers}(\mathcal{X})}$:

$$\mu^{\text{inv}}(i) \;=\; \big\{ \langle \mathbf{x} \mid \exists j.\ \mu(\mathbf{x}) = [i, j\rangle \in \mathcal{X} \big\} \cup \big\{ \mathbf{x}\rangle \mid \exists j.\ \mu(\mathbf{x}) = [j, i\rangle \in \mathcal{X} \big\}$$

3. Se define la **secuenciación** de μ como $\text{seq}(\mu) = \text{seq}_0(\mu) \cdot \ldots \cdot \text{seq}_n(\mu)$:

$$\text{seq}_i(\mu) = \begin{cases} (i, \mu^{\text{inv}}(i)) & \mu^{\text{inv}}(i) \neq \varnothing \\ \epsilon & \mu^{\text{inv}}(i) = \varnothing \end{cases}$$

Índice invertido y secuenciación de un mapping

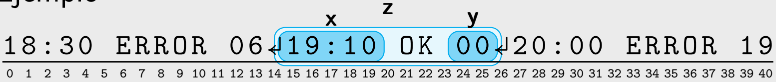
Sea $d = a_0 \dots a_{n-1}$, un conj. de variables $\mathcal{X}$ y un mapping $\mu : \mathcal{X} \to \text{Spans}(d)$.

Definiciones

3. Se define la **secuenciación** de μ como $\text{seq}(\mu) = \text{seq}_0(\mu) \cdot \ldots \cdot \text{seq}_n(\mu)$:

$$\text{seq}_i(\mu) = \begin{cases} (i, \mu^{\text{inv}}(i)) & \mu^{\text{inv}}(i) \neq \varnothing \\ \epsilon & \mu^{\text{inv}}(i) = \varnothing \end{cases}$$

Ejemplo



$$\mu(\mathbf{x}) = [15, 20\rangle \qquad \mu^{\text{inv}}(15) = \{\langle \mathbf{x}, \langle \mathbf{z}\} \qquad \mu^{\text{inv}}(20) = \{\mathbf{x}\rangle\}$$

$$\mu(\mathbf{y}) = [24, 26\rangle \qquad \mu^{\text{inv}}(24) = \{\langle \mathbf{y}\} \qquad \mu^{\text{inv}}(26) = \{\mathbf{y}\rangle, \mathbf{z}\rangle\}$$

$$\mu(\mathbf{z}) = [15, 26\rangle \qquad \forall i \notin \{15, 20, 24, 26\} \qquad \mu^{\text{inv}}(i) = \varnothing$$

$$\text{seq}(\mu) = (15, \{\langle \mathbf{x}, \langle \mathbf{z}\})\,(20, \{\mathbf{x}\rangle\})\,(24, \{\langle \mathbf{y}\})\,(26, \{\mathbf{y}\rangle, \mathbf{z}\rangle\})$$

Índice invertido y secuenciación de un mapping

Sea $d = a_0 \ldots a_{n-1}$, un conj. de variables $\mathcal{X}$ y un mapping $\mu : \mathcal{X} \to \mathsf{Spans}(d)$.

Definiciones

3. Se define la **secuenciación** de μ como $\mathsf{seq}(\mu) = \mathsf{seq}_0(\mu) \cdot \ldots \cdot \mathsf{seq}_n(\mu)$:

$$\mathsf{seq}_i(\mu) = \begin{cases} (i, \mu^{\mathsf{inv}}(i)) & \mu^{\mathsf{inv}}(i) \neq \varnothing \\ \epsilon & \mu^{\mathsf{inv}}(i) = \varnothing \end{cases}$$

$\mathsf{seq}(\mu)$ es una **representación equivalente** de un mapping μ

... y nos será más conveniente para
nuestros algoritmos de enumeración de resultados.

Outline

Autómata con anotaciones (AnnA)

Definición

Un autómata con anotaciones (AnnA) es una tupla:

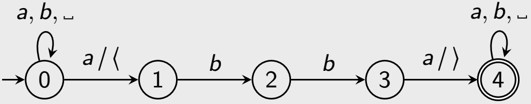
$$\mathcal{N} \quad = \quad (Q, \Sigma, \Lambda, \Delta, I, F)$$

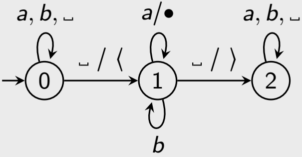
- Q es un conjunto finito de estados.
- Σ es el alfabeto de input.
- Λ es un conjunto finito de etiquetas (Labels).
- $\Delta \subseteq Q \times \left(\Sigma \cup \Sigma \times \mathcal{N}\right) \times Q$ es la relación de transición.
- $I \subseteq Q$ es un conjunto de estados iniciales.
- $F \subseteq Q$ es el conjunto de estados finales (o aceptación).

Las transiciones (p, a, ℓ, q) simbolizan que
al leer la letra a, está letra **será anotada** con ℓ

Autómata con anotaciones (AnnA)

Ejemplos de vset autómata

1. 

2. 

Ejecución de un AnnA

Sea un AnnA $\mathcal{N} = (Q, \Sigma, \Lambda, \Delta, I, F)$ y un documento $d = a_0 \dots a_{n-1} \in \Sigma^*$.

Definiciones

Una **ejecución** ρ de $\mathcal{A}$ sobre d es una secuencia $\rho : p_0 \overset{t_0}{\to} p_1 \overset{t_1}{\to} \dots \overset{t_{n-1}}{\to} p_n$ tal que cumple todas las siguientes condiciones:

- $p_0 \in I$
- para todo $i < n$, t_i es de la forma $t_i = a_i$ o $t_i = (a_i, \ell)$ para algún $\ell \in \Lambda$
- para todo $i < n$, $(p_i, t_i, p_{i+1}) \in \Delta$.

Se define la **anotación de ρ** como $\text{ann}(\rho) = \text{ann}_0(t_0) \cdot \dots \cdot \text{ann}_{n-1}(t_{n-1})$:

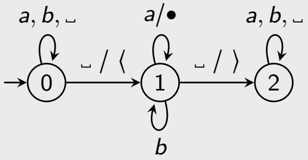
$$\text{ann}_i(t) = \begin{cases} (i, \ell) & t = (a, \ell) \\ \epsilon & t = a \end{cases}$$

Decimos que ρ es de **aceptación** ssi $q_n \in F$.

Ejecución de un AnnA

Ejemplos de ejecuciones

$$d = \underset{0\ \ 1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10}{\underline{\mathtt{a}\ \ \mathtt{abba}\ \ \mathtt{ba}\ \ \mathtt{b}}}$$



Algunas ejecuciones sobre d:

$$\rho_1 :\ p_0 \xrightarrow{a} p_0 \xrightarrow{\sqcup/\langle} p_1 \xrightarrow{a/\bullet} p_1 \xrightarrow{b} p_1 \xrightarrow{b} p_1 \xrightarrow{a/\bullet} p_1 \xrightarrow{\sqcup/\rangle} p_2 \xrightarrow{b} p_2 \xrightarrow{a} p_2 \xrightarrow{\sqcup} p_2 \xrightarrow{b} p_2$$

$$\rho_2 :\ p_0 \xrightarrow{a} p_0 \xrightarrow{\sqcup} p_0 \xrightarrow{a} p_0 \xrightarrow{b} p_0 \xrightarrow{b} p_0 \xrightarrow{a} p_0 \xrightarrow{\sqcup/\langle} p_1 \xrightarrow{b} p_1 \xrightarrow{a/\bullet} p_1 \xrightarrow{\sqcup/\rangle} p_2 \xrightarrow{b} p_2$$

$$\mathrm{ann}(\rho_1) \ = \ (1, \langle)\,(2, \bullet)\,(5, \bullet)\,(6, \rangle) \qquad \mathrm{ann}(\rho_2) \ = \ (6, \langle)\,(8, \bullet)\,(9, \rangle)$$

$$\mathtt{a}\overset{\langle}{\sqcup}\overset{\bullet}{\mathtt{a}}\mathtt{b}\mathtt{b}\overset{\bullet}{\mathtt{a}}\overset{\rangle}{\sqcup}\mathtt{b}\mathtt{a}\sqcup\mathtt{b} \qquad\qquad \mathtt{a}\sqcup\mathtt{a}\mathtt{b}\mathtt{b}\mathtt{a}\overset{\langle}{\sqcup}\mathtt{b}\overset{\bullet}{\mathtt{a}}\overset{\rangle}{\sqcup}\mathtt{b}$$

Output de un AnnA

Sea $\mathcal{N} = (Q, \Sigma, \Lambda, \Delta, I, F)$ un autómata con anotaciones (AnnA).

Definición

Se define la función $[\![\mathcal{N}]\!]$ tal que para todo documento $d \in \Sigma^*$:

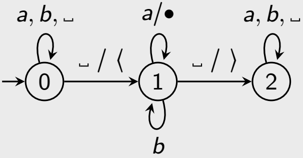
$$[\![\mathcal{N}]\!](d) \;\; = \;\; \left\{ \text{ann}(\rho) \;\middle|\; \rho \text{ es una ejecución } \textbf{aceptación} \text{ de } \mathcal{N} \text{ sobre } d \right\}$$

Output de un AnnA

Ejemplos de output de un AnnA

$$d = \underline{\texttt{a␣abba␣ba␣b}}$$

0 1 2 3 4 5 6 7 8 9 10



Para el documento d se tiene que:

$$[\![\mathcal{N}]\!](d) \;=\; \{\; (1,\langle)\,(2,\bullet)\,(5,\bullet)\,(6,\rangle)\;,\;(6,\langle)\,(8,\bullet)\,(9,\rangle)\;\}$$

¿cuál es la diferencia entre AnnA, vset automata y un **transductor**?

Outline

Desde variables a anotaciones

Sea Σ un alfabeto finito.

Teorema

Para todo vset autómata funcional $\mathcal{A}$ sobre Σ,
existe un AnnA $\mathcal{N}$ sobre $\Sigma \cup \{\#\}$ tal que para todo documento d sobre Σ:

$$[\![\mathcal{N}]\!](d \cdot \#) \;=\; \{\, \mathsf{seq}(\mu) \mid \mu \in [\![\mathcal{A}]\!](d) \,\}$$

$\mathcal{N}$ entrega la **secuenciación** de los mappings en $[\![\mathcal{A}]\!](d)$

Algunas propiedades de vset autómata funcionales

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un vset autómata funcional y $p, q \in Q$.

Suposición

Sin perdida de generalidad, desde ahora supondremos que todos los estados de un vset autómata funcional $\mathcal{A}$ son **útiles**.

En otras palabras, para todo estado $p \in Q$ de $\mathcal{A}$:

- Existe una ejecución (camino de transiciones) desde I a p
- Existe una ejecución (camino de transiciones) desde p a F.

¿por qué podemos hacer esta suposición?

Algunas propiedades de vset autómata funcionales

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un vset autómata funcional y $p, q \in Q$.

Definición

Una **ejecución sin lectura** (ejec-SL) de p a q en $\mathcal{A}$ es una secuencia:

$$\pi : \; p_0 \stackrel{s_0}{\to} p_1 \stackrel{s_1}{\to} \dots \stackrel{s_{k-1}}{\to} p_k$$

- $p_0 = p$ y $p_k = q$
- para todo $i < k$, $(p_i, s_i, p_{i+1}) \in \Delta$ y $s_i \in \text{Markers}(\mathcal{X}) \cup \{\epsilon\}$.

Un ejecución sin lectura es
un **camino de transiciones** de p a q tal que $s_i \notin \Sigma$.

Algunas propiedades de vset autómata funcionales

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un vset autómata funcional y $p, q \in Q$.

Definición

Una **ejecución sin lectura** (ejec-SL) de p a q en $\mathcal{A}$ es una secuencia:

$$\pi: \; p_0 \xrightarrow{s_0} p_1 \xrightarrow{s_1} \ldots \xrightarrow{s_{k-1}} p_k$$

- $p_0 = p$ y $p_k = q$
- para todo $i < k$, $(p_i, s_i, p_{i+1}) \in \Delta$ y $s_i \in \text{Markers}(\mathcal{X}) \cup \{\epsilon\}$.

Propiedades de ejecuciones sin lectura

1. Para todo $i \neq j$, si $s_i = s_j$, entonces $s_i = s_j = \epsilon$.

Demostración: ejercicio

Algunas propiedades de vset autómata funcionales

Sea $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$ un vset autómata funcional y $p, q \in Q$.

Definición

Una **ejecución sin lectura** (ejec-SL) de p a q en $\mathcal{A}$ es una secuencia:

$$\pi : \; p_0 \stackrel{s_0}{\to} p_1 \stackrel{s_1}{\to} \dots \stackrel{s_{k-1}}{\to} p_k$$

- $p_0 = p$ y $p_k = q$
- para todo $i < k$, $(p_i, s_i, p_{i+1}) \in \Delta$ y $s_i \in \text{Markers}(\mathcal{X}) \cup \{\epsilon\}$.

Defina el **conjunto de π** como $\text{set}(\pi) = \{s_i \mid s_i \in \text{Markers}(\mathcal{X})\}$.

Propiedades de ejecuciones sin lectura

1. Para todo $i \neq j$, si $s_i = s_j$, entonces $s_i = s_j = \epsilon$.
2. Para todo par de ejec-SL distintas π_1, π_2 de p a q en $\mathcal{A}$, se cumple que $\text{set}(\pi_1) = \text{set}(\pi_2)$.

Demostración: ejercicio

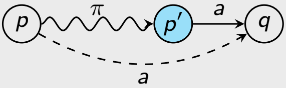
Desde variables a anotaciones

Demostración Teorema

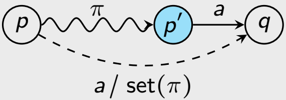
Dado un vset autómata funcional $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$, construimos:

$$\mathcal{N} \;=\; (Q, \Sigma \cup \{\#\}, 2^{\mathrm{Markers}(\mathcal{X})}, \Delta', I, F)$$

$(p, a, q) \in \Delta'$ ssi existe $p' \in Q$ y una ejec-SL π de p a p' tal que $(p', a, q) \in \Delta$ y $\mathrm{set}(\pi) = \varnothing$.



$(p, a, S, q) \in \Delta'$ ssi existe $p' \in Q$ y ejec-SL π de p a p' tal que $(p', a, q) \in \Delta$ y $\mathrm{set}(\pi) = S \neq \varnothing$.



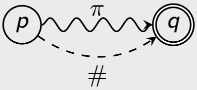
Desde variables a anotaciones

Demostración Teorema

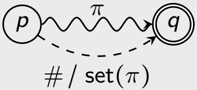
Dado un vset autómata funcional $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$, construimos:

$$\mathcal{N} \;=\; (Q, \Sigma \cup \{\#\}, 2^{\mathsf{Markers}(\mathcal{X})}, \Delta', I, F)$$

$(p, \#, q) \in \Delta'$ ssi existe una ejec-SL π de p a q tal que $\mathsf{set}(\pi) = \varnothing$ y $q \in F$.



$(p, \#, S, q) \in \Delta'$ ssi existe una ejec-SL π de p a q tal que $\mathsf{set}(\pi) = S \neq \varnothing$ y $q \in F$.



Desde variables a anotaciones

Demostración Teorema

Dado un vset autómata funcional $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$, construimos:

$$\mathcal{N} \;=\; (Q, \Sigma \cup \{\#\}, 2^{\text{Markers}(\mathcal{X})}, \Delta', I, F)$$

$(p, a, q) \in \Delta'$ ssi existe $p' \in Q$ y una ejec-SL π de p a p' tal que $(p', a, q) \in \Delta$ y $\text{set}(\pi) = \varnothing$.

$(p, a, S, q) \in \Delta'$ ssi existe $p' \in Q$ y ejec-SL π de p a p' tal que $(p', a, q) \in \Delta$ y $\text{set}(\pi) = S \neq \varnothing$.

$(p, \#, q) \in \Delta'$ ssi existe una ejec-SL π de p a q tal que $\text{set}(\pi) = \varnothing$ y $q \in F$.

$(p, \#, S, q) \in \Delta'$ ssi existe una ejec-SL π de p a q tal que $\text{set}(\pi) = S \neq \varnothing$ y $q \in F$.

Por las propiedades 1. y 2. la construcción es **correcta**

Desde variables a anotaciones

Demostración Teorema

Dado un vset autómata funcional $\mathcal{A} = (Q, \Sigma, \mathcal{X}, \Delta, I, F)$, construimos:

$$\mathcal{N} \;=\; (Q, \Sigma \cup \{\#\}, 2^{\text{Markers}(\mathcal{X})}, \Delta', I, F)$$

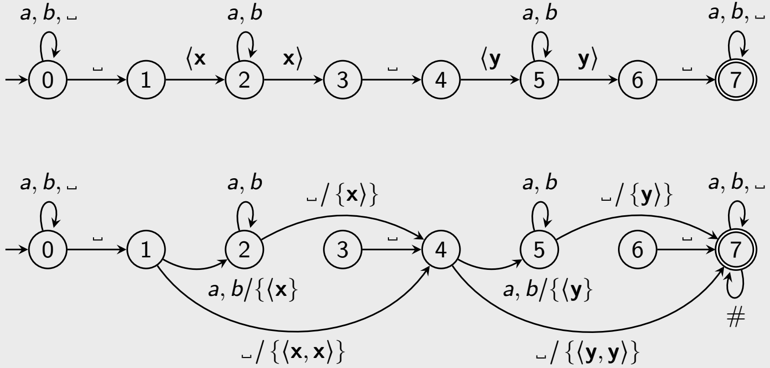
$(p, a, q) \in \Delta'$	ssi	existe $p' \in Q$ y una ejec-SL π de p a p' tal que $(p', a, q) \in \Delta$ y $\text{set}(\pi) = \varnothing$.
$(p, a, S, q) \in \Delta'$	ssi	existe $p' \in Q$ y ejec-SL π de p a p' tal que $(p', a, q) \in \Delta$ y $\text{set}(\pi) = S \neq \varnothing$.
$(p, \#, q) \in \Delta'$	ssi	existe una ejec-SL π de p a q tal que $\text{set}(\pi) = \varnothing$ y $q \in F$
$(p, \#, S, q) \in \Delta'$	ssi	existe una ejec-SL π de p a q tal que $\text{set}(\pi) = S \neq \varnothing$ y $q \in F$

Por último, podemos ver que: $[\![\mathcal{N}]\!](d \cdot \#) \;=\; \{\, \text{seq}(\mu) \mid \mu \in [\![\mathcal{A}]\!](d) \,\}$. ∎

Termine la demostración (ejercicio)

Desde variables a anotaciones

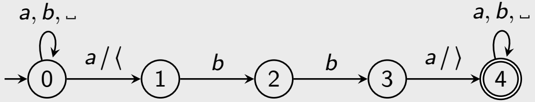
Ejemplo de la construcción

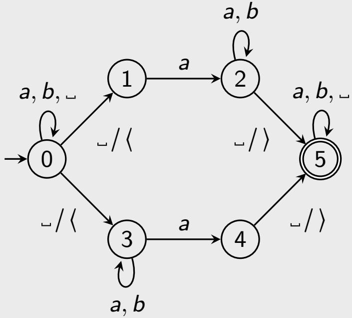


Outline

Determinismo para autómata con anotaciones

¿cuál de los dos AnnA diría que son deterministas?

1. 

2. 

Determinismo para autómata con anotaciones

Sea $\mathcal{N} = (Q, \Sigma, \Lambda, \Delta, I, F)$ un autómata con anotaciones.

Definición

Decimos que $\mathcal{N}$ es **Input-Output determinista** (I/O-determinista) ssi $|I| = 1$ y para todo $(p, t_1, q_1), (p, t_2, q_2) \in \Delta$, si $t_1 = t_2$, entonces $q_1 = q_2$.

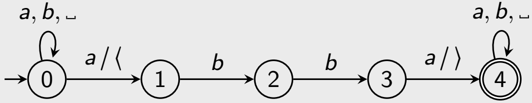
$\mathcal{N}$ funciona de manera determinista al recibir
el documento y una **anotación simultáneamente**

¿qué ventaja tiene un autómata I/O-determinista?

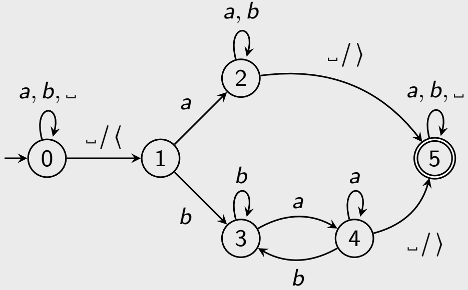
Determinismo para autómata con anotaciones

Ejemplos

1.



2'.



Todo AnnA se puede I/O-determinizar

Teorema

Para todo AnnA $\mathcal{N} = (Q, \Sigma, \Lambda, \Delta, I, F)$,
existe un AnnA I/O-determinista $\mathcal{N}^{\text{det}}$ tal que $[\![\mathcal{N}]\!] = [\![\mathcal{N}^{\text{det}}]\!]$.

Demostración

Considere la determinización de $\mathcal{N}$ como:

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

- $2^Q = \{S \mid S \subseteq Q\}$ es el conjunto potencia de Q.
- $q_0^{\text{det}} = I$.
- $\Delta^{\text{det}} : 2^Q \times (\Sigma \cup \Sigma \times \Lambda) \to 2^Q$ tal que para todo $t \in \Sigma \cup (\Sigma \times \Lambda)$:

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

- $F^{\text{det}} = \{S \in 2^Q \mid S \cap F \neq \varnothing\}$. ∎