Extracción de información

Clase 25

IIC 2223

Prof. Cristian Riveros

Extracción de información desde documentos

```
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.

Extracción de información desde documentos

```
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?

1. Una tarea como la anterior **se puede programar fácilmente**,
   ¿por qué queremos automatizarla?
2. Las expresiones regulares ya hacen el trabajo anterior,
   ¿por qué queremos **estudiar este problema de nuevo**?

En esta última parte del curso veremos **nuevas técnicas formales y algorítmicas** para entender y resolver este problema.

Outline

Outline

Representación de intervalos: spans

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
```

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

Representación de intervalos: spans

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 $d[[i, i\rangle] = \epsilon$.

- Denotamos el conjunto de **todos los spans** de d como $\mathrm{Spans}(d)$.

Queremos estudiar como **extraer spans desde documentos**.

Outline

Expresiones regulares con variables

Sea Σ un alfabeto finito y $\mathcal{X}$ un conjunto de variables.

Definición (Sintaxis)

R es una expresión regular con variables (o **regex**) sobre Σ si R es igual a:

1. a — para alguna letra $a \in \Sigma$.
2. ϵ
3. $\mathbf{x}\{R_1\}$ — donde R_1 es una regex y $\mathbf{x} \in \mathcal{X}$.
4. $(R_1 + R_2)$ — donde R_1 y R_2 son regex.
5. $(R_1 \cdot R_2)$ — donde R_1 y R_2 son regex.
6. (R_1^*) — donde R_1 es una regex.

$\mathbf{x}\{R_1\}$ representa que el **span capturado** por R_1
lo almacenaremos en la variable $\mathbf{x}$

Expresiones regulares con variables

Ejemplos de regex

Sea $\Sigma = \{a, b, \llcorner\!\lrcorner\}$.

- $\Sigma^* \cdot \mathbf{x}\{abba\} \cdot \Sigma^*$
- $\Sigma^* \cdot \llcorner\!\lrcorner \cdot \mathbf{x}\{ab \cdot (a + b)^*\} \cdot \llcorner\!\lrcorner \cdot \Sigma^*$
- $\Sigma^* \cdot \llcorner\!\lrcorner \cdot \mathbf{x}\{(a + b)^*\} \cdot \llcorner\!\lrcorner \cdot \mathbf{y}\{(a + b)^*\} \cdot \llcorner\!\lrcorner \cdot \Sigma^*$
- $(\Sigma^* \cdot \llcorner\!\lrcorner + \epsilon) \cdot \mathbf{z}\big\{\mathbf{x}\{(a + b)^+\} \cdot \llcorner\!\lrcorner \cdot \mathbf{y}\{(a + b)^+\}\big\} \cdot (\epsilon + \llcorner\!\lrcorner \cdot \Sigma^*)$
- $\Sigma^* \cdot \mathbf{x}\{abba\} \cdot \llcorner\!\lrcorner \cdot \mathbf{x}\{baba\} \cdot \Sigma^*$

Regex válidas

Sea $\text{Var}(R)$ el conjunto de **todas las variables** en $\mathcal{X}$ mencionadas en R.

Definición

R es una regex **válida** ssi todas las siguientes condiciones se cumplen:

1. si $R = R_1 \cdot R_2$, entonces $\text{Var}(R_1) \cap \text{Var}(R_2) = \varnothing$ y R_1, R_2 son válidas
2. si $R = R_1 + R_2$, entonces $\text{Var}(R_1) = \text{Var}(R_2)$ y R_1, R_2 son válidas
3. si $R = R_1^*$, entonces $\text{Var}(R_1) = \varnothing$ y R_1 es válida
4. si $R = \mathbf{x}\{R_1\}$, entonces $\mathbf{x} \notin \text{Var}(R_1)$ y R_1 es válida

Si R es **válida** nos permite asignar las variables de **manera correcta**.

Regex válidas

Definición regex válidas (resumen)

1. si $R = R_1 \cdot R_2$, entonces $\text{Var}(R_1) \cap \text{Var}(R_2) = \varnothing$ y R_1, R_2 son válidas
2. si $R = R_1 + R_2$, entonces $\text{Var}(R_1) = \text{Var}(R_2)$ y R_1, R_2 son válidas
3. si $R = R_1^*$, entonces $\text{Var}(R_1) = \varnothing$ y R_1 es válida
4. si $R = \mathbf{x}\{R_1\}$, entonces $\mathbf{x} \notin \text{Var}(R_1)$ y R_1 es válida

Ejemplos

- $\Sigma^* \cdot \mathbf{x}\{abba\} \cdot \Sigma^*$ ✓
- $\Sigma^* \cdot \mathbf{x}\{abba\} \cdot \text{␣} \cdot \mathbf{y}\{baba\} \cdot \Sigma^*$ ✓
- $\Sigma^* \cdot \text{␣} \cdot \mathbf{x}\{ab \cdot (a + b^*\} \cdot \text{␣} \cdot \Sigma^*$ ✓
- $\Sigma^* \cdot \mathbf{x}\{abba\} \cdot \text{␣} \cdot \mathbf{x}\{baba\} \cdot \Sigma^*$ ✗
- $\Sigma^* \cdot (\mathbf{x}\{abba\} + \mathbf{y}\{baba\}) \cdot \Sigma^*$ ✗
- $\Sigma^* \cdot \text{␣} \cdot (\mathbf{x}\{ab \cdot (a + b)^*\} \cdot \text{␣})^* \cdot \Sigma^*$ ✗

Mappings como outputs de regex

Definiciones

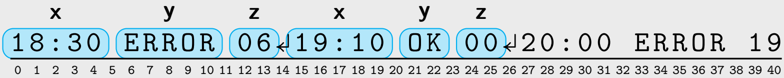
- Un **mapping** de R sobre un documento $d \in \Sigma^*$ es una función:

$$\mu : \ \mathsf{Var}(R) \to \mathsf{Spans}(d)$$

  donde el dominio de μ es $\mathrm{dom}(\mu) = \mathsf{Var}(R)$.

- Se define el mapping $\mu = \perp$ como el **mapping vacío** donde $\mathrm{dom}(\perp) = \varnothing$.

Ejemplo



$$[\mathbf{x} \mapsto [0,5\rangle, \mathbf{y} \mapsto [6,11\rangle, \mathbf{z} \mapsto [12,14\rangle]$$

$$[\mathbf{x} \mapsto [15,20\rangle, \mathbf{y} \mapsto [21,23\rangle, \mathbf{z} \mapsto [24,26\rangle]$$

$$\perp = [\cdot]$$

Mappings como outputs de regex

Definiciones

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

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

- Para $k \in \mathbb{N}$ se define el **mapping $\mu + k$** tal que $\mathrm{dom}(\mu + k) = \mathrm{dom}(\mu)$ 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 $\mathrm{dom}(\mu_1) \cap \mathrm{dom}(\mu_2) = \emptyset$ se define **la únion**:

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

Cada regex define un conjunto de mappings

Semántica regex

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

1. $[\![a]\!](d) = \{\bot\}$ si $d = a$, y $\emptyset$ en otro caso
2. $[\![\epsilon]\!](d) = \{\bot\}$ si $d = \epsilon$, y $\emptyset$ en otro caso

Ejemplos

- $[\![b]\!](a) = \emptyset$
- $[\![b]\!](b) = \{\bot\}$

Cada regex define un conjunto de mappings

Semántica regex

1. $[\![a]\!](d) = \{\bot\}$ si $d = a$, y $\varnothing$ en otro caso
2. $[\![\epsilon]\!](d) = \{\bot\}$ si $d = \epsilon$, y $\varnothing$ en otro caso
3. $[\![\mathbf{x}\{R_1\}]\!](d) = \left\{ \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in [\![R_1]\!](d) \text{ y } s = [0, |d|\rangle \right\}$

Ejemplos

- $[\![b]\!](a) = \varnothing$
- $[\![b]\!](b) = \{\bot\}$
- $[\![\mathbf{x}\{b\}]\!](a) = \varnothing$
- $[\![\mathbf{x}\{b\}]\!](b) = \left\{ [\mathbf{x} \mapsto [0, 1\rangle] \right\}$

Cada regex define un conjunto de mappings

Semántica regex

1. $[\![a]\!](d) = \{\perp\}$ si $d = a$, y $\varnothing$ en otro caso
2. $[\![\epsilon]\!](d) = \{\perp\}$ si $d = \epsilon$, y $\varnothing$ en otro caso
3. $[\![\mathbf{x}\{R_1\}]\!](d) = \left\{ \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in [\![R_1]\!](d) \text{ y } s = [0, |d|\rangle \right\}$

Como $\mathbf{x}\{R_1\}$ es **válida**, entonces sabemos que $\mathbf{x} \notin \mathrm{dom}(\mu)$.

Cada regex define un conjunto de mappings

Semántica regex

3. $[\![\mathbf{x}\{R_1\}]\!](d) = \{ \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in [\![R_1]\!](d) \text{ y } s = [0, |d|\rangle \}$

4. $[\![R_1 \cdot R_2]\!](d) = \left\{ \mu_1 \cup (\mu_2 + |d_1|) \;\middle|\; \begin{array}{l} \text{existe } d_1, d_2 \text{ tal que } d = d_1 \cdot d_2, \\ \mu_1 \in [\![R_1]\!](d_1) \text{ y } \mu_2 \in [\![R_2]\!](d_2) \end{array} \right\}$

Ejemplos

- $[\![b]\!](b) = \{\bot\}$
- $[\![\mathbf{x}\{b\}]\!](b) = \{ [\mathbf{x} \mapsto [0,1\rangle] \}$
- $[\![\mathbf{x}\{b\}b]\!](bb) = \{ [\mathbf{x} \mapsto [0,1\rangle] \}$
- $[\![b\,\mathbf{x}\{b\}]\!](bb) = \{ [\mathbf{x} \mapsto [1,2\rangle] \}$
- $[\![\mathbf{x}\{b\}\,\mathbf{y}\{b\}]\!](bb) = \{ [\mathbf{x} \mapsto [0,1\rangle, \mathbf{y} \mapsto [1,2\rangle] \}$

Cada regex define un conjunto de mappings

Semántica regex

3. $[\![\mathbf{x}\{R_1\}]\!](d) = \{\, \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in [\![R_1]\!](d) \text{ y } s = [0, |d|\rangle \,\}$

4. $$[\![R_1 \cdot R_2]\!](d) = \left\{ \mu_1 \cup (\mu_2 + |d_1|) \;\middle|\; \begin{array}{l} \text{existe } d_1, d_2 \text{ tal que } d = d_1 \cdot d_2, \\ \mu_1 \in [\![R_1]\!](d_1) \text{ y } \mu_2 \in [\![R_2]\!](d_2) \end{array} \right\}$$

Como $R_1 \cdot R_2$ son válidas, entonces sabemos que $\mathrm{dom}(\mu_1) \cap \mathrm{dom}(\mu_2) = \varnothing$

Cada regex define un conjunto de mappings

Semántica regex

3. $[\![\mathbf{x}\{R_1\}]\!](d) = \{\, \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in [\![R_1]\!](d) \text{ y } s = [0, |d|\rangle \,\}$

4. $[\![R_1 \cdot R_2]\!](d) = \left\{ \mu_1 \cup (\mu_2 + |d_1|) \;\middle|\; \begin{array}{l} \text{existe } d_1, d_2 \text{ tal que } d = d_1 \cdot d_2, \\ \mu_1 \in [\![R_1]\!](d_1) \text{ y } \mu_2 \in [\![R_2]\!](d_2) \end{array} \right\}$

5. $[\![R_1 + R_2]\!](d) = [\![R_1]\!](d) \cup [\![R_2]\!](d)$

Ejemplos

- $[\![\mathbf{x}\{b\}b]\!](bb) = \{\, [\mathbf{x} \mapsto [0,1\rangle] \,\}$
- $[\![b\,\mathbf{x}\{b\}]\!](bb) = \{\, [\mathbf{x} \mapsto [1,2\rangle] \,\}$
- $[\![\mathbf{x}\{b\}b + b\,\mathbf{x}\{b\}]\!](bb) = \{\, [\mathbf{x} \mapsto [0,1\rangle],\ [\mathbf{x} \mapsto [1,2\rangle] \,\}$

Cada regex define un conjunto de mappings

Semántica regex

3. $[\![\mathbf{x}\{R_1\}]\!](d) = \left\{ \mu \cup [\mathbf{x} \mapsto s] \mid \mu \in [\![R_1]\!](d) \text{ y } s = [0, |d|\rangle \right\}$

4. $[\![R_1 \cdot R_2]\!](d) = \left\{ \mu_1 \cup (\mu_2 + |d_1|) \;\middle|\; \begin{array}{l} \text{existe } d_1, d_2 \text{ tal que } d = d_1 \cdot d_2, \\ \mu_1 \in [\![R_1]\!](d_1) \text{ y } \mu_2 \in [\![R_2]\!](d_2) \end{array} \right\}$

5. $[\![R_1 + R_2]\!](d) = [\![R_1]\!](d) \cup [\![R_2]\!](d)$

6. $[\![R_1^*]\!](d) = \bigcup_{k=0}^{\infty} [\![(R_1)^k]\!](d)$

Como $\text{Var}(R_1) = \varnothing$, entonces $[\![R_1^*]\!](d) = \{\perp\}$ ssi $d \in \mathcal{L}(R_1^*)$.

Cada regex define un conjunto de mappings

Semántica regex (completa)

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

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

Evaluación de regex

¿cuál es el resultado de las siguientes regex?

$$d = \underset{0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14\ 15\ 16}{\underline{\texttt{a abba ba baba ba}}}$$

- $[\![\Sigma^* \cdot \mathbf{x}\{aba + bab\} \cdot \Sigma^*]\!](d) = \{\, [\mathbf{x} \mapsto [10, 13\rangle],\, [\mathbf{x} \mapsto [11, 14\rangle] \,\}$
- $[\![\Sigma^* \cdot \llcorner\!\lrcorner \cdot \mathbf{x}\{ba \cdot (a+b)^*\} \cdot \llcorner\!\lrcorner \cdot \Sigma^*]\!](d) = \{\, [\mathbf{x} \mapsto [7, 9\rangle],\, [\mathbf{x} \mapsto [10, 14\rangle] \,\}$
- $[\![\Sigma^* \cdot \mathbf{x}\{abba\} \cdot \llcorner\!\lrcorner \cdot \mathbf{y}\{ba\} \cdot \Sigma^*]\!](d) = \{\, [\mathbf{x} \mapsto [2, 6\rangle, \mathbf{y} \mapsto [7, 9\rangle] \,\}$
- $[\![\Sigma^* \cdot \llcorner\!\lrcorner \cdot \mathbf{x}\{(a+b)^*\} \cdot \llcorner\!\lrcorner \cdot \mathbf{y}\{(a+b)^*\} \cdot \llcorner\!\lrcorner \cdot \Sigma^*]\!](d) =$
  $\{\, [\mathbf{x} \mapsto [2, 6\rangle, \mathbf{y} \mapsto [7, 9\rangle],\, [\mathbf{x} \mapsto [7, 9\rangle, \mathbf{y} \mapsto [10, 14\rangle] \}$
- $[\![\Sigma^* \cdot \mathbf{x}\{\Sigma^*\} \cdot \Sigma^*]\!](d) \;=\; \mathrm{Spans}(d)$

Evaluación de regex

¿cuál es el resultado de las siguientes regex?

$$d = \underbrace{\texttt{a abba ba baba ba}}_{0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14\ 15\ 16}$$

- $[\![\Sigma^* \cdot \mathbf{x}\{\Sigma^*\} \cdot \Sigma^*]\!](d) \ = \ \text{Spans}(d)$

- Dado una regex R con una variable, ¿de qué tamaño puede ser $|[\![R]\!](d)|$ con respecto a $|d|$ **en el peor caso**?
- Dado una regex R con k variables, ¿de qué tamaño puede ser $|[\![R]\!](d)|$ con respecto a $|d|$ y k **en el peor caso**?

Entonces, ¿cómo podemos evaluar una regex R **eficientemente**?

Outline

Autómata con variables (vset automata)

¿qué tienen de nuevo?

1. tiene transiciones con **abre** y **cierra** de variable **x**:

$$p \stackrel{\langle \mathbf{x}}{\to} q \quad \text{y} \quad p \stackrel{\mathbf{x}\rangle}{\to} q$$

2. cada ejecución define un **mapping** de las variables a spans.

Vset autómata será nuestro primer módelo para **compilar regex**

Autómata con variables (vset automata)

Definición

Un vset automata (VA) es una tupla:

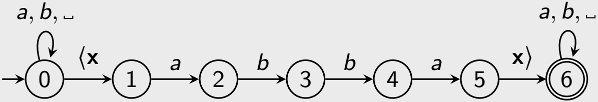
$$\mathcal{A} \quad = \quad (Q, \Sigma, \mathcal{X}, \Delta, I, F)$$

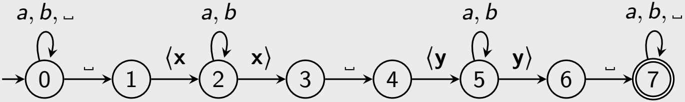
- Q es un conjunto finito de estados.
- Σ es el alfabeto de input.
- $\mathcal{X}$ es un conjunto finito de variables.
- $\Delta \subseteq Q \times \left(\Sigma \cup \{\epsilon\} \cup \{\langle \mathbf{x}, \mathbf{x}\rangle \mid \mathbf{x} \in \mathcal{X}\}\right) \times Q$ es la relación de transición.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

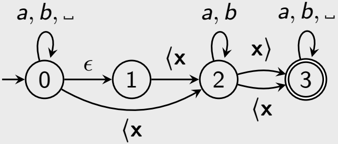
'$\langle \mathbf{x}$' simboliza **abrir** y '$\mathbf{x}\rangle$' simboliza **cerrar** la variable $\mathbf{x}$

Autómata con variables (vset automata)

Ejemplos de vset autómata

1. 

2. 

3. 

Ejecución de un vset autómata (VA)

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

- Un par $(p, i) \in Q \times \{0, \dots, n\}$ es una **configuración** de $\mathcal{A}$ sobre d.
- Una configuración $(p, 0)$ es **inicial** si $q \in I$.
- Una configuración $(p, |d|)$ es **final** si $q \in F$.

"Intuitivamente, una configuración (p, i) representa que $\mathcal{A}$ se encuentra en el estado p antes de leer a_i."

Ejecución de un vset autómata (VA)

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

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

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

tal que cumple todas las siguientes condiciones:

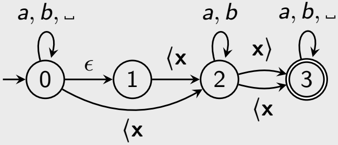
- $o_k \in \Sigma \cup \{\epsilon\} \cup \{\langle \mathbf{x}, \mathbf{x} \rangle \mid \mathbf{x} \in \mathcal{X}\}$ con $k \leq m$
- (p_0, i_0) es una configuración inicial
- para todo $k < m$, $(p_k, o_{k+1}, p_{k+1}) \in \Delta$
- para todo $k \leq m$, si $o_k \in \Sigma$, entonces $o_k = a_{i_{k-1}}$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

Ejemplos de ejecuciones

$$d = \underset{0\ \ 1\ \ 2}{\underline{\texttt{a b a}}}$$



Algunas ejecuciones sobre d:

$$\rho_1 : (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{b} (p_0, 2) \xrightarrow{a} (p_0, 3)$$

$$\rho_2 : (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{\epsilon} (p_1, 1) \xrightarrow{\langle \mathbf{x}} (p_2, 1) \xrightarrow{b} (p_2, 2) \xrightarrow{\mathbf{x}\rangle} (p_3, 2) \xrightarrow{a} (p_3, 3)$$

$$\rho_3 : (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{\langle \mathbf{x}} (p_2, 1) \xrightarrow{b} (p_2, 2) \xrightarrow{\mathbf{x}\rangle} (p_3, 2) \xrightarrow{a} (p_3, 3)$$

$$\rho_4 : (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{\langle \mathbf{x}} (p_2, 1) \xrightarrow{b} (p_2, 2) \xrightarrow{\langle \mathbf{x}} (p_3, 2) \xrightarrow{a} (p_3, 3)$$

¿cuáles ejecuciones son de **aceptación**? ¿cuáles **deberían** dar output?

Ejecución válida de un vset autómata (VA)

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 $\mathbf{x} \in \mathcal{X}$:

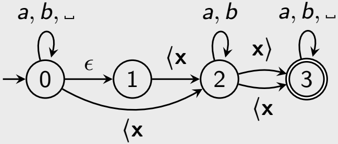
- existe un único $k_1 \leq m$ tal que $o_{k_1} = \langle \mathbf{x}$
- existe un único $k_2 \leq m$ tal que $o_{k_2} = \mathbf{x} \rangle$ y
- $k_1 < k_2$.

"Intuitivamente, ρ es válida si, y solo si, todas las variables se abren y cierran correctamente."

Ejecución válida de un vset autómata (VA)

¿cuáles ejecuciones son válidas?

$$d = \underset{0\;1\;2}{\underline{\texttt{a}\,\texttt{b}\,\texttt{a}}}$$



Algunas ejecuciones sobre d:

$$\rho_1 : \ (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{b} (p_0, 2) \xrightarrow{a} (p_0, 3)$$

$$\rho_2 : \ (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{\epsilon} (p_1, 1) \xrightarrow{\langle \mathbf{x}} (p_2, 1) \xrightarrow{b} (p_2, 2) \xrightarrow{\mathbf{x}\rangle} (p_3, 2) \xrightarrow{a} (p_3, 3)$$

$$\rho_3 : \ (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{\langle \mathbf{x}} (p_2, 1) \xrightarrow{b} (p_2, 2) \xrightarrow{\mathbf{x}\rangle} (p_3, 2) \xrightarrow{a} (p_3, 3)$$

$$\rho_4 : \ (p_0, 0) \xrightarrow{a} (p_0, 1) \xrightarrow{\langle \mathbf{x}} (p_2, 1) \xrightarrow{b} (p_2, 2) \xrightarrow{\langle \mathbf{x}} (p_3, 2) \xrightarrow{a} (p_3, 3)$$

El mapping de una ejecución válida

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 $o_{k_2} = \mathbf{x} \rangle$.

El mapping de una ejecución válida

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 $o_{k_2} = \mathbf{x}\rangle$.

¿cuál es el mapping de las siguientes ejecuciones válidas?

$$\rho_2 : \ (p_0, 0) \overset{a}{\to} (p_0, 1) \overset{\epsilon}{\to} (p_1, 1) \overset{\langle \mathbf{x}}{\to} (p_2, 1) \overset{b}{\to} (p_2, 2) \overset{\mathbf{x}\rangle}{\to} (p_3, 2) \overset{a}{\to} (p_3, 3)$$

$$\rho_3 : \ (p_0, 0) \overset{a}{\to} (p_0, 1) \overset{\langle \mathbf{x}}{\to} (p_2, 1) \overset{b}{\to} (p_2, 2) \overset{\mathbf{x}\rangle}{\to} (p_3, 2) \overset{a}{\to} (p_3, 3)$$

$$\text{map}(\rho_2) = \text{map}(\rho_3) = [\mathbf{x} \mapsto [1, 2\rangle]$$

Función de extracción de un vset autómata

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 $d \in \Sigma^*$:

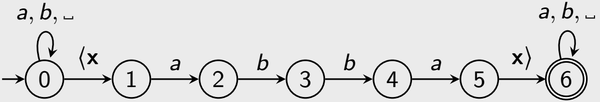
$$[\![\mathcal{A}]\!](d) \;=\; \left\{ \text{map}(\rho) \;\middle|\; \begin{array}{l} \rho \text{ es una ejecución} \\ \textbf{válida} \text{ y de } \textbf{aceptación} \text{ de } \mathcal{A} \text{ sobre } d \end{array} \right\}$$

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

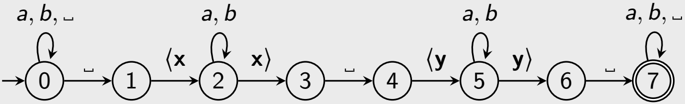
Función de extracción de un vset autómata

¿qué función de extracción define cada VA?

1.



2.



Outline

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:

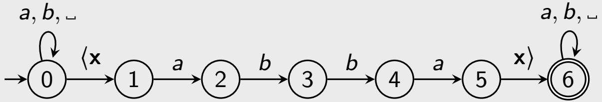
si ρ es de **aceptación**, entonces ρ es **válida**.

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

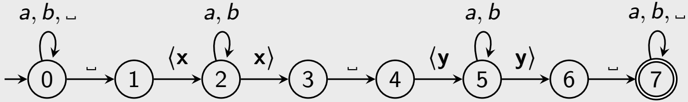
Vset automata funcionales

¿cuáles VA son funcionales?

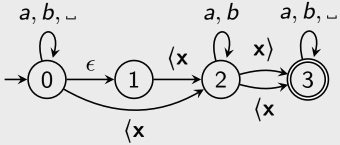
1.



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:

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

Demostración: ejercicio (similar al Teorema de Kleene)

¿és verdad la otra dirección?