Una Introducción a la Teoría de Autómatas sobre Arboles

IIC3800

Arboles etiquetados

Σ: Alfabeto (conjunto finito de símbolos)

Definición (Arbol binario)

Un árbol T sobre Σ es una tupla (D, λ):

- *D es un subconjunto finito de $\{0,1\}^*$ cerrado bajo prefijo*
- $\lambda : D \to \Sigma$

Ejemplo

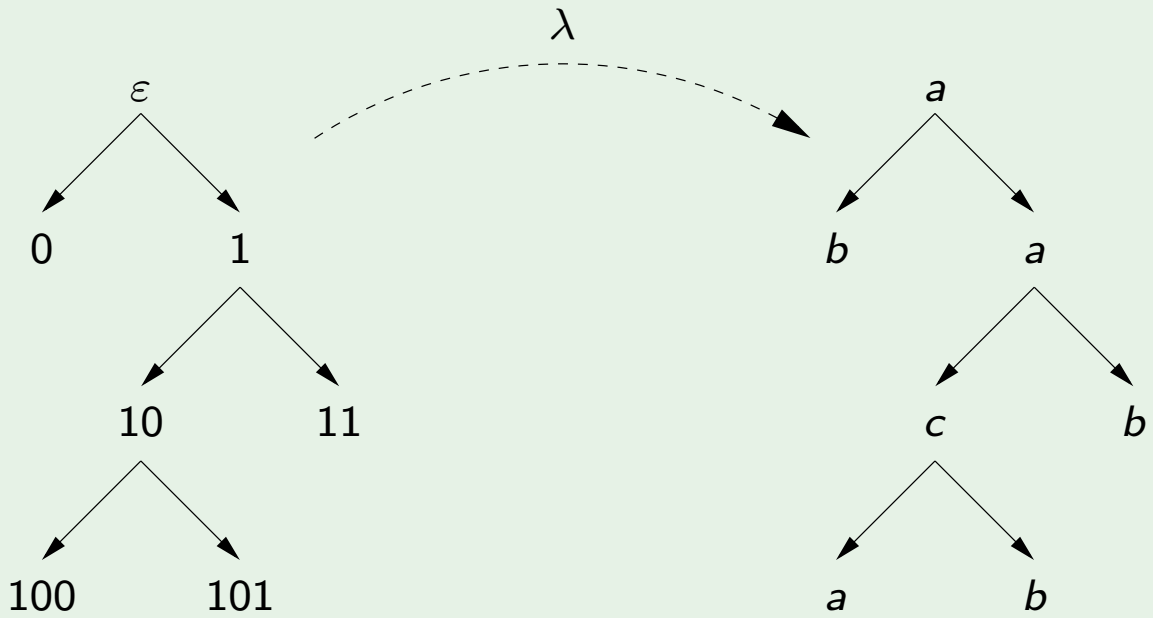
$\Sigma = \{a, b, c\}$ y $T = (D, \lambda)$:

$$\begin{aligned} D &= \{\varepsilon,\ 0,\ 1,\ 10,\ 11,\ 100,\ 101\} \\ \lambda &= \{\varepsilon \mapsto a,\ 0 \mapsto b,\ 1 \mapsto a, \\ &\qquad 10 \mapsto c,\ 11 \mapsto b,\ 100 \mapsto a,\ 101 \mapsto b\} \end{aligned}$$

Arboles etiquetados

Example

Gráficamente:



Autómatas sobre árboles

Supuesto: Cada nodo tiene dos hijos o ninguno.

Definición (Top-down & determinista)

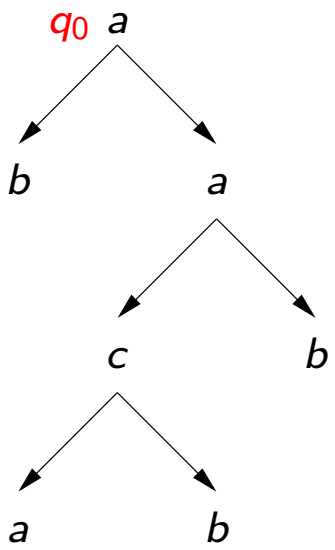
$\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$*:*

- Q *es un conjunto finito de estados*
- q_0 *es el estado inicial*
- F *es un conjunto de estados finales*
- δ *es una función parcial:*

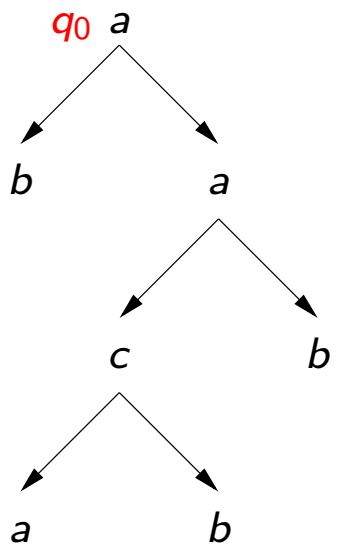
$$\delta \quad : \quad Q \times \Sigma \rightarrow Q \times Q$$

Notación: TD-DTA

Autómatas sobre árboles: Funcionamiento

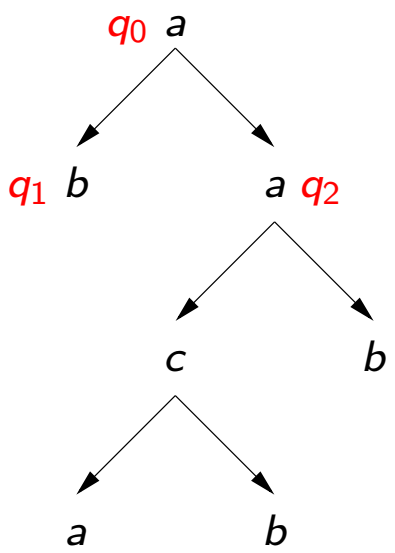


Autómatas sobre árboles: Funcionamiento



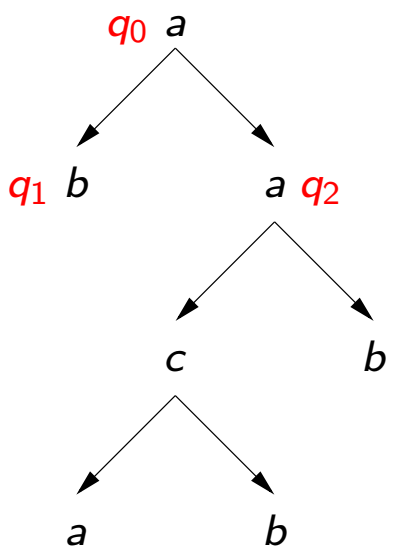
$\delta(q_0, a) = (q_1, q_2)$

Autómatas sobre árboles: Funcionamiento



$\delta(q_0, a) = (q_1, q_2)$

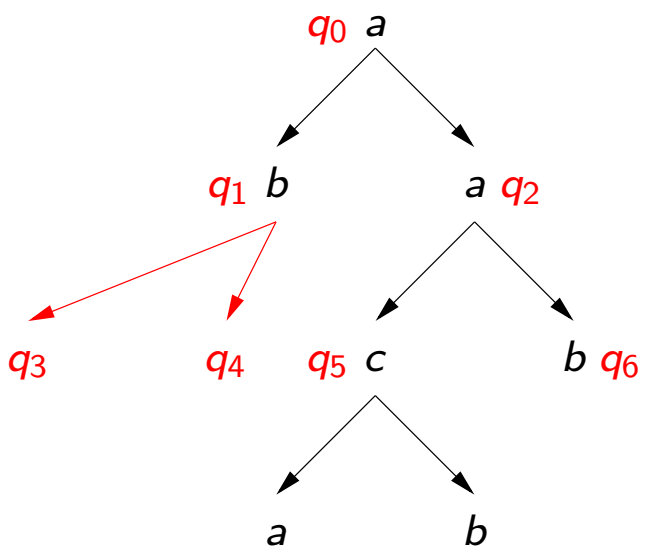
Autómatas sobre árboles: Funcionamiento



$\delta(q_0, a) = (q_1, q_2)$

$\delta(q_1, b) = (q_3, q_4)$
$\delta(q_2, a) = (q_5, q_6)$

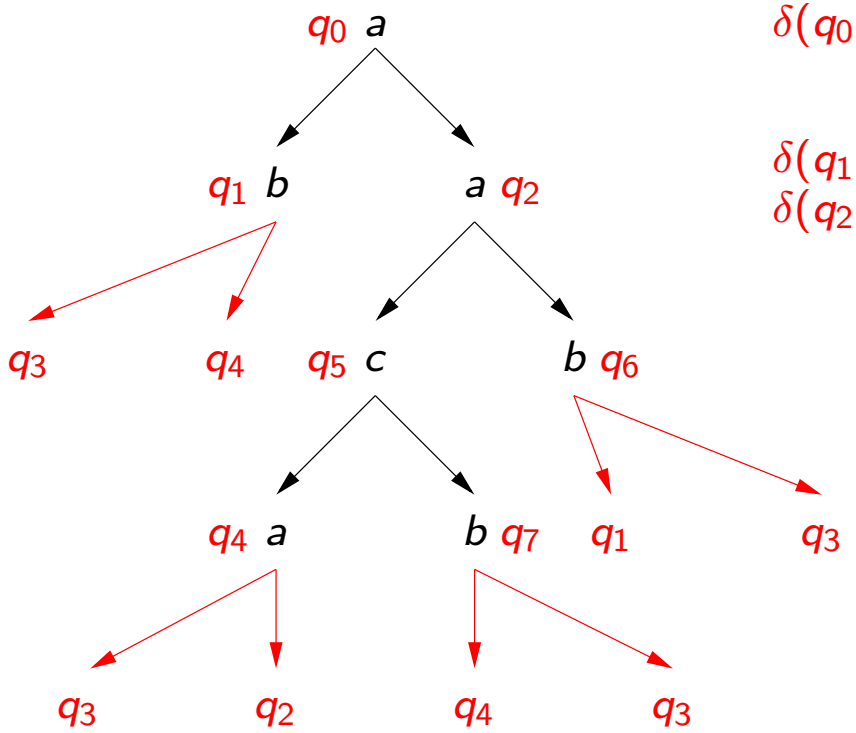
Autómatas sobre árboles: Funcionamiento



$\delta(q_0, a) = (q_1, q_2)$

$\delta(q_1, b) = (q_3, q_4)$
$\delta(q_2, a) = (q_5, q_6)$

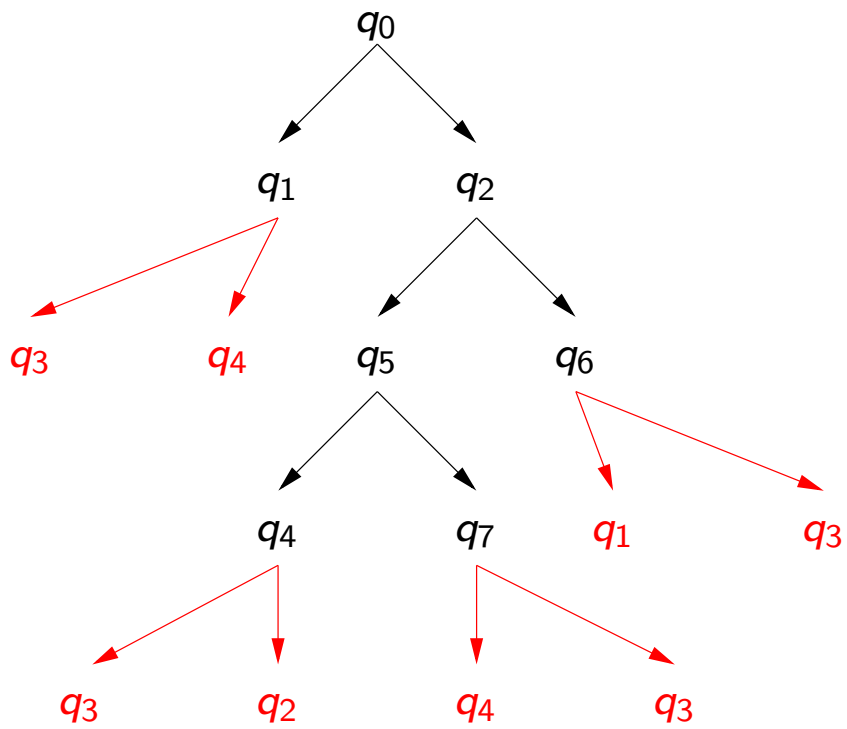
Autómatas sobre árboles: Funcionamiento



$\delta(q_0, a) = (q_1, q_2)$

$\delta(q_1, b) = (q_3, q_4)$
$\delta(q_2, a) = (q_5, q_6)$

Autómatas sobre árboles: Funcionamiento



Condición de aceptación: Cada hoja tiene un estado en F

Funcionamiento: Noción de ejecución

Dado: TD-DTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ y $T = (D, \lambda)$ (con $D \neq \emptyset$).

Definición (Ejecución de un TD-DTA)

$\rho : D \to Q$ *es la ejecución de* $\mathcal{A}$ *sobre* T *si:*

- $\rho(\varepsilon) = q_0$
- *para cada* $w, w0, w1 \in D$: $\delta(\rho(w), \lambda(w)) = (\rho(w0), \rho(w1))$

Condición de aceptación: $\delta(\rho(w), \lambda(w)) \in (F \times F)$ para cada $w \in D$ que es maximal en D bajo la relación de ser prefijo ($w \in \max_{\text{prefijo}}(D)$).

- El árbol vacío es aceptado si $q_0 \in F$

Autómatas sobre árboles: No determinismo

Definición (Top-down & no determinista)

$\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$:

- Q *es un conjunto finito de estados*
- q_0 *es el estado inicial*
- F *es un conjunto de estados finales*
- δ *es una función total:*

$$\delta \quad : \quad Q \times \Sigma \to 2^{Q \times Q}$$

Notación: TD-NTA

Ejecución de un TD-NTA

Dado: TD-NTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ y $T = (D, \lambda)$ (con $D \neq \emptyset$).

Definición (Ejecución de una TD-NTA)

$\rho : D \to Q$ es una ejecución de $\mathcal{A}$ sobre T si:

- $\rho(\varepsilon) = q_0$
- *para cada* $w, w0, w1 \in D$: $(\rho(w0), \rho(w1)) \in \delta(\rho(w), \lambda(w))$

Condición de aceptación: Existe una ejecución ρ de $\mathcal{A}$ sobre T tal que $\delta(\rho(w), \lambda(w)) \cap (F \times F) \neq \emptyset$ para cada $w \in \max_{\text{prefijo}}(D)$.

TD-DTA versus TD-NTA

Proposición

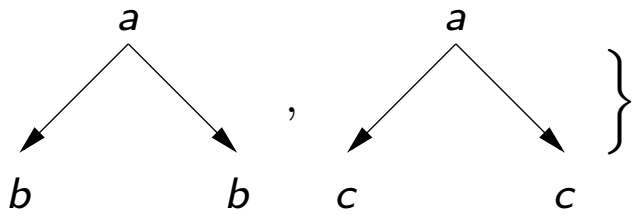
TD-DTA $\subsetneq$ TD-NTA

TD-DTA versus TD-NTA

Proposición

TD-DTA $\subsetneq$ TD-NTA

Demostración: Considere el lenguaje

$L = \left\{ \right.$  $\left. \right\}$

Demuestre que L es aceptado por un TD-NTA y no es aceptado por un TD-DTA. □

Ejercicios

1. Construya un TD-NTA que acepte el lenguaje de los árboles sobre el alfabeto $\{a, b\}$ que tienen un número par de símbolos a.

2. Demuestre que no existe un TD-DTA que acepte el lenguaje del ejercicio anterior.

Lenguajes regulares de árboles

Notación: $L(\mathcal{A})$ es el lenguaje de árboles aceptado por un TD-NTA (TD-DTA) $\mathcal{A}$.

Definición (Lenguaje regular de árboles)

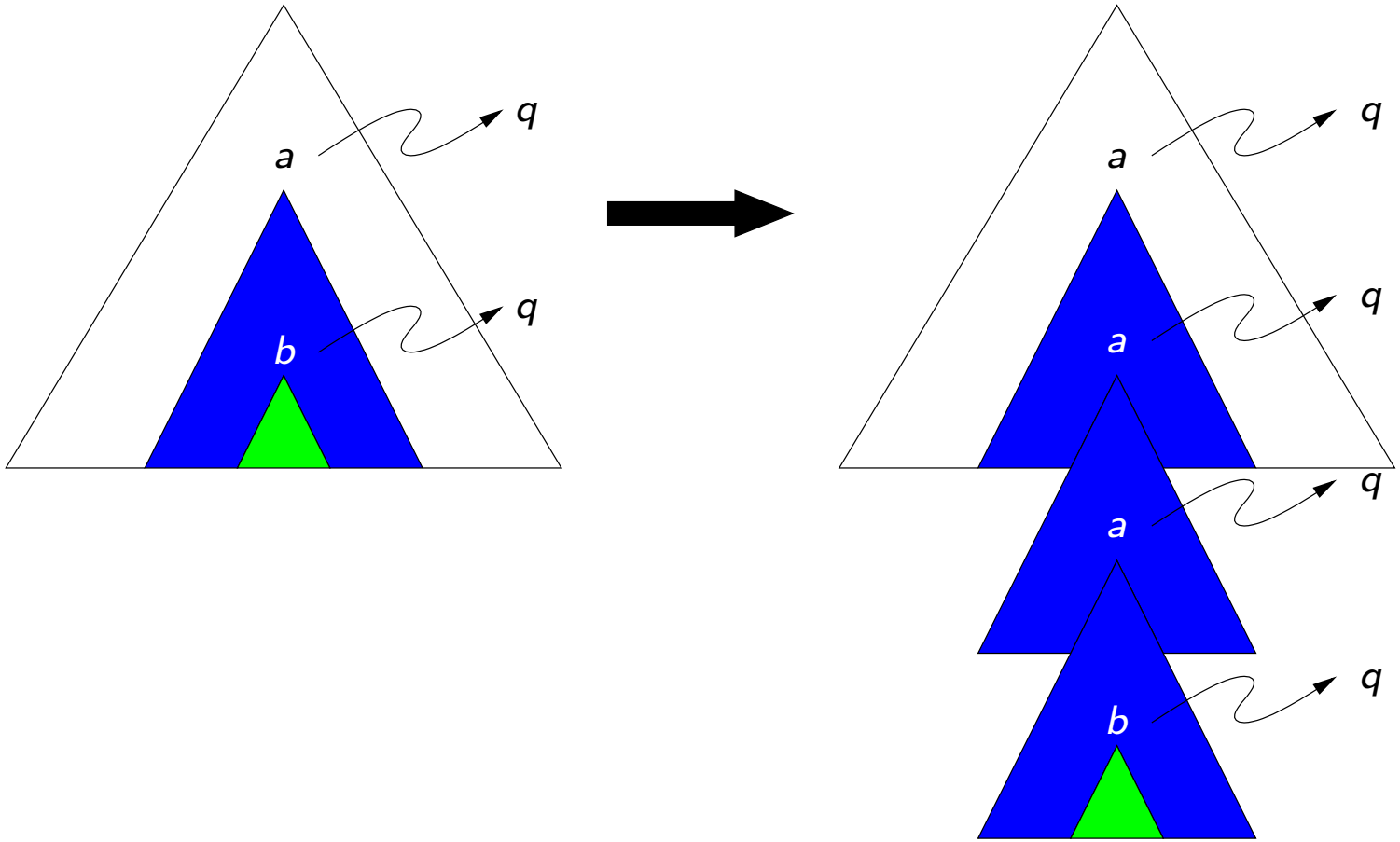
Un lenguaje L de árboles es regular si existe TD-NTA $\mathcal{A}$ *tal que* $L = L(\mathcal{A})$.

¿Qué lenguajes de árboles no son regulares?

- Una herramienta útil: Lema de bombeo

Lema de bombeo para lenguajes regulares de árboles

El lema en una figura:



Ejercicio

Demuestre que el lenguaje de todos los árboles balanceados sobre el alfabeto $\{a, b\}$ no es un lenguaje regular de árboles.

Autómatas sobre árboles: El enfoque bottom-up

Definición (Bottom-up & determinista)

$\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$:

- Q *es un conjunto finito de estados*
- q_0 *es el estado inicial*
- F *es un conjunto de estados finales*
- δ *es una función total:*

$$\delta \quad : \quad Q \times Q \times \Sigma \to Q$$

Notación: BU-DTA

Ejecución de un Bu-DTA

Dado: Bu-DTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ y $T = (D, \lambda)$ (con $D \neq \emptyset$).

Definición (Ejecución de un Bu-DTA)

$\rho : D \to Q$ *es la ejecución de* $\mathcal{A}$ *sobre* T *si:*

- *para cada* $w \in max_{prefijo}(D)$: $\rho(w) = \delta(q_0, q_0, \lambda(w))$
- *para cada* $w, w0, w1 \in D$: $\rho(w) = \delta(\rho(w0), \rho(w1), \lambda(w))$

Condición de aceptación: $\rho(\varepsilon) \in F$

Ejercicios

1. Construya un $\textsc{Bu-DTA}$ que acepte el siguiente lenguaje:

$$L = \left\{ \begin{array}{ccc} a & & a \\ \swarrow\ \searrow & , & \swarrow\ \searrow \\ b \quad b & & c \quad c \end{array} \right\}$$

2. Construya un $\textsc{Bu-DTA}$ que acepte el lenguaje de los árboles sobre el alfabeto $\{a, b\}$ que tienen un número par de símbolos a.

Autómatas sobre árboles bottom-up: No determinismo

Definición (Bottom-up & no determinista)

$\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$:

- Q *es un conjunto finito de estados*
- q_0 *es el estado inicial*
- F *es un conjunto de estados finales*
- δ *es una función total:*

$$\delta \quad : \quad Q \times Q \times \Sigma \to 2^Q$$

Notación: BU-NTA

Ejecución de un Bu-NTA

Dado: Bu-NTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ y $T = (D, \lambda)$ (con $D \neq \emptyset$)

Definición (Ejecución de un Bu-NTA)

$\rho : D \to Q$ es la ejecución de $\mathcal{A}$ sobre T si:

- *para cada $w \in max_{prefijo}(D)$:* $\rho(w) \in \delta(q_0, q_0, \lambda(w))$
- *para cada $w, w0, w1 \in D$:* $\rho(w) \in \delta(\rho(w0), \rho(w1), \lambda(w))$

Condición de aceptación: $\rho(\varepsilon) \in F$

Bu-NTA son determinizables

Proposición

Bu-NTA = Bu-DTA

Bu-NTA son determinizables

Proposición

Bu-NTA = Bu-DTA

Demostración: Dado Bu-NTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$, defina Bu-DTA $\mathcal{B} = (2^Q, \Sigma, \{q_0\}, \delta', F')$:

$$\begin{aligned} \delta'(X_1, X_2, a) &= \left(\bigcup_{q_1 \in X_1} \bigcup_{q_2 \in X_2} \delta(q_1, q_2, a) \right) \\ F' &= \{X \subseteq Q \mid X \cap F \neq \emptyset\} \end{aligned}$$

Se tiene que $L(\mathcal{A}) = L(\mathcal{B})$. □

La relación entre los autómatas estudiados

Proposición

$\textsc{Td}\text{-DTA} \subsetneq \textsc{Td}\text{-NTA} = \textsc{Bu}\text{-NTA} = \textsc{Bu}\text{-DTA}$

La relación entre los autómatas estudiados

Proposición

$\textsc{Td-DTA} \subsetneq \textsc{Td-NTA} = \textsc{Bu-NTA} = \textsc{Bu-DTA}$

$\textsc{Td-NTA} \subseteq \textsc{Bu-NTA}$: Dado $\textsc{Td-NTA}$ $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$, defina $\textsc{Bu-NTA}$ $\mathcal{B} = (Q \cup \{q_0'\}, \Sigma, q_0', \delta', F')$:

$$\begin{array}{rcll} \delta'(q_1, q_2, a) & = & \{q \in Q \mid (q_1, q_2) \in \delta(q, a)\} & q_1, q_2 \in Q \\ \delta'(q_0', q_0', a) & = & \{q \in Q \mid \delta(q, a) \cap (F \times F) \neq \emptyset\} & \\ F' & = & \{q_0\} & \end{array}$$

Se tiene que $L(\mathcal{A}) = L(\mathcal{B})$.

La relación entre los autómatas estudiados

BU-NTA $\subseteq$ TD-NTA: Dado BU-NTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$, defina TD-NTA $\mathcal{B} = (Q \cup \{q_0'\}, \Sigma, q_0', \delta', F')$:

$$\begin{array}{rcll} \delta'(q, a) & = & \{(q_1, q_2) \in Q \times Q \mid q \in \delta(q_1, q_2, a)\} & q \in Q \\ \delta'(q_0', a) & = & \{(q_1, q_2) \in Q \times Q \mid \delta(q_1, q_2, a) \cap F \neq \emptyset\} & \\ F' & = & \{q_0\} & \end{array}$$

Se tiene que $L(\mathcal{A}) = L(\mathcal{B})$. □

Propiedades de clausura

Proposición

La clase de los lenguajes regulares de árboles es cerrada bajo unión, intersección y complemento.

Propiedades de clausura

Proposición

La clase de los lenguajes regulares de árboles es cerrada bajo unión, intersección y complemento.

Demostración: Dados $\mathcal{A}_1 = (Q_1, \Sigma, q_0^1, \delta_1, F_1)$ y $\mathcal{A}_2 = (Q_2, \Sigma, q_0^2, \delta_2, F_2)$, defina $\mathcal{A}_1 \times \mathcal{A}_2 = (Q, \Sigma, q_0, \delta, F)$:

$$\begin{aligned} Q &= Q_1 \times Q_2 \\ q_0 &= (q_0^1, q_0^2) \\ \delta((q_1, q_2), a) &= \bigcup_{(q_3,q_4) \in \delta_1(q_1,a)} \bigcup_{(q_5,q_6) \in \delta_2(q_2,a)} \{((q_3, q_5), (q_4, q_6))\} \\ F &= F_1 \times F_2 \end{aligned}$$

Se tiene que $L(\mathcal{A}_1 \times \mathcal{A}_2) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$. □

Arboles versus palabras

¿Son los autómatas para árboles más expresivos que los autómatas para palabras?

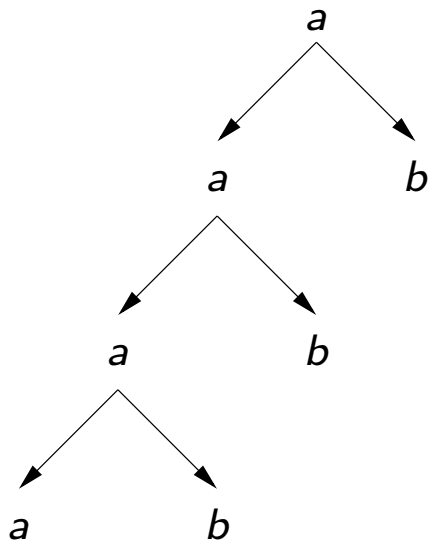
- Si representamos árboles como palabras, ¿qué tipo de lenguajes obtenemos?

Arboles versus palabras

¿Son los autómatas para árboles más expresivos que los autómatas para palabras?

- Si representamos árboles como palabras, ¿qué tipo de lenguajes obtenemos?

¿Cómo se puede representar el siguiente árbol T como una palabra?



Arboles = XML

```
<a>
    <a>
        <a>
            <a>
            </a>
            <b>
            </b>
        </a>
        <b>
        </b>
    </a>
    <b>
    </b>
</a>
```

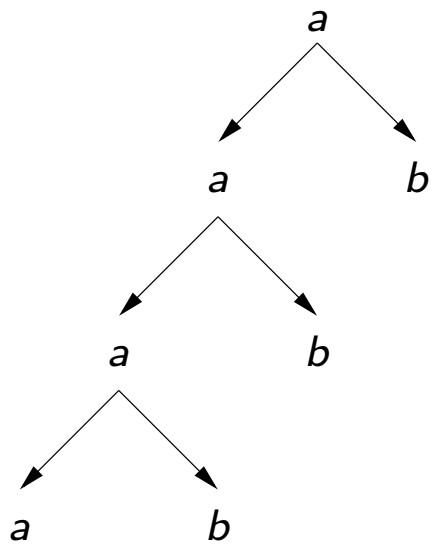
Arboles = XML

```
<a>
    <a>
        <a>
            <a>
            </a>
            <b>
            </b>
        </a>
        <b>
        </b>
    </a>
    <b>
    </b>
</a>
```

Representación: $\text{rep}(T) = aaaa\bar{a}b\bar{b}\bar{a}b\bar{b}\bar{a}b\bar{b}\bar{a}$

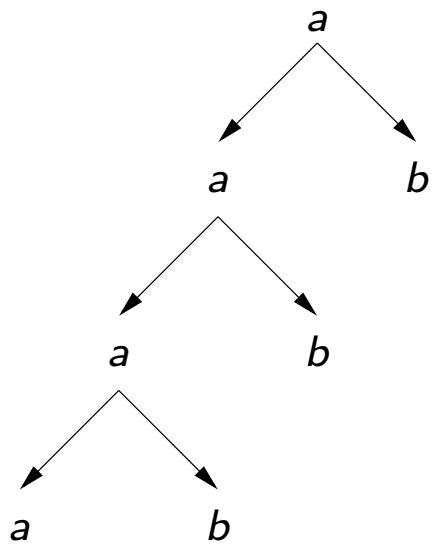
Arboles versus palabras (continuación)

Sea L_0 el lenguaje de los árboles de la forma:



Arboles versus palabras (continuación)

Sea L_0 el lenguaje de los árboles de la forma:

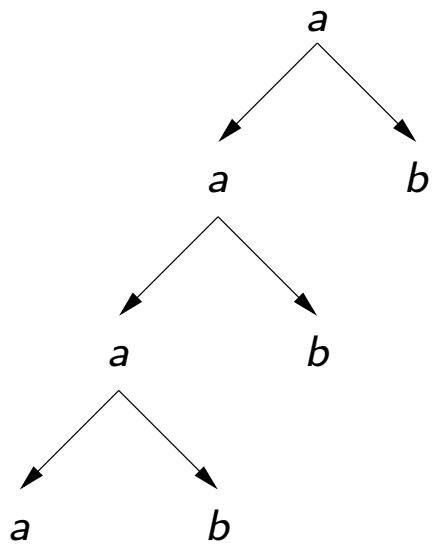


Proposición

L_0 es un lenguaje regular de árboles, pero
$rep(L_0) = \{rep(T) \mid T \in L_0\}$ no es un lenguaje regular.

Arboles versus palabras (continuación)

Sea L_0 el lenguaje de los árboles de la forma:



Proposición

L_0 es un lenguaje regular de árboles, pero
$rep(L_0) = \{rep(T) \mid T \in L_0\}$ no es un lenguaje regular.

Demostración: $\text{rep}(L_0) = \{aa^n(\bar{a}b\bar{b})^n\bar{a} \mid n \geq 0\}$. □

Autómatas sobre árboles versus gramáticas libre de contexto

Proposición

Si L es un lenguaje regular de árboles, entonces $rep(L)$ es un lenguaje libre de contexto.

Autómatas sobre árboles versus gramáticas libre de contexto

Proposición

Si L es un lenguaje regular de árboles, entonces rep(L) es un lenguaje libre de contexto.

Demostración: Dado un TD-NTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$, considere la siguiente gramática libre de contexto:

$$
\begin{array}{rcll}
S & \rightarrow & (q_0, a) & a \in \Sigma \\
(q, a) & \rightarrow & a(q_1, b_1)(q_2, b_2)\bar{a} & (q_1, q_2) \in \delta(q, a) \text{ y } b_1, b_2 \in \Sigma \\
(q, a) & \rightarrow & a\bar{a} & \delta(q, a) \cap (F \times F) \neq \emptyset
\end{array}
$$

Autómatas sobre árboles versus gramáticas libre de contexto

¿Podemos entonces estudiar los lenguajes regulares de árboles usando herramientas de los lenguajes libres de contexto?

Autómatas sobre árboles versus gramáticas libre de contexto

¿Podemos entonces estudiar los lenguajes regulares de árboles usando herramientas de los lenguajes libres de contexto?

Este enfoque no funciona bien.

- Un lenguaje regular de árboles corresponde a los árboles de derivación de una gramática libre de contexto.
  - No hay una correspondencia directa con el lenguaje de las hojas

Autómatas sobre árboles versus gramáticas libre de contexto

¿Podemos entonces estudiar los lenguajes regulares de árboles usando herramientas de los lenguajes libres de contexto?

Este enfoque no funciona bien.

- Un lenguaje regular de árboles corresponde a los árboles de derivación de una gramática libre de contexto.
  - No hay una correspondencia directa con el lenguaje de las hojas

Vamos a ver que esta es una diferencia fundamental.

Autómatas sobre árboles: Problemas de decisión

Problemas a resolver

Dados autómatas $\mathcal{A}$ y $\mathcal{B}$ sobre árboles:

Non-emptiness	:	¿Es $L(\mathcal{A}) \neq \emptyset$?
Containment	:	¿Es $L(\mathcal{A}) \subseteq L(\mathcal{B})$?
Equivalence	:	¿Es $L(\mathcal{A}) = L(\mathcal{B})$?

Para recordar . . .

¿Cuál es la complejidad de los problemas anteriores para lenguajes regulares y libres de contexto sobre palabras?

	NFA	CFG
Non-emptiness		
Containment		
Equivalence		

Para recordar . . .

¿Cuál es la complejidad de los problemas anteriores para lenguajes regulares y libres de contexto sobre palabras?

	NFA	CFG
Non-emptiness	PTIME	
Containment	PSPACE-complete	
Equivalence	PSPACE-complete	

Para recordar . . .

¿Cuál es la complejidad de los problemas anteriores para lenguajes regulares y libres de contexto sobre palabras?

	NFA	CFG
Non-emptiness	PTIME	PTIME
Containment	PSPACE-complete	indecidible
Equivalence	PSPACE-complete	indecidible

Non-emptiness para TD-NTA

Algoritmo para non-emptiness

Input : TD-NTA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$
Output : ¿es $L(\mathcal{A}) \neq \emptyset$?

$A := F$
$O := \emptyset$
while $O \neq A$ **do**
    $O := A$
    for each $q \in Q$ **do**
        if existe $a \in \Sigma$ tal que $\delta(q, a) \cap (O \times O) \neq \emptyset$
            then $A := A \cup \{q\}$
if $q_0 \in A$ **then return**(*true*)
    else return(*false*)

Non-emptiness para TD-NTA

El algoritmo funciona en tiempo polinomial. Se obtiene que:

Proposición

Non-emptiness para TD-NTA *está en PTIME. Containment y equivalence para* TD-NTA *están en EXPTIME.*

Non-emptiness para TD-NTA

El algoritmo funciona en tiempo polinomial. Se obtiene que:

Proposición

Non-emptiness para TD-NTA *está en PTIME. Containment y equivalence para* TD-NTA *están en EXPTIME.*

Demostración: Para verificar si $L(\mathcal{A}) \subseteq L(\mathcal{B})$ haga lo siguiente:

- Construya TD-NTA $\overline{\mathcal{B}}$
  - $L(\overline{\mathcal{B}}) = \Sigma^* \setminus L(\mathcal{B})$
- Construya TD-NTA $\mathcal{A} \times \overline{\mathcal{B}}$
  - $L(\mathcal{A} \times \overline{B}) = L(\mathcal{A}) \cap L(\overline{\mathcal{B}})$
- Verifique si $L(\mathcal{A} \times \overline{\mathcal{B}}) = \emptyset$ □

Complejidad exacta de los problemas de decisión

	NFA	CFG	TD-NTA
Non-emptiness	PTIME	PTIME	
Containment	PSPACE-complete	indecidible	
Equivalence	PSPACE-complete	indecidible	

Complejidad exacta de los problemas de decisión

	NFA	CFG	TD-NTA
Non-emptiness	PTIME	PTIME	PTIME
Containment	PSPACE-complete	indecidible	EXPTIME-complete
Equivalence	PSPACE-complete	indecidible	EXPTIME-complete

XML: Arboles con un número arbitrario de hijos

Definición (Arbol sin rango)

Un árbol sin rango T sobre Σ es una tupla (D, λ):

- *D es un subconjunto finito de $\mathbb{N}^*$ cerrado bajo prefijo y tal que:*

  para cada $i \in \mathbb{N}$ y $w \in \mathbb{N}^*$, si $w \cdot i \in D$, entonces para cada $j \in \{0, \ldots, i-1\}$: $w \cdot j \in D$

- $\lambda : D \to \Sigma$

Ejemplo

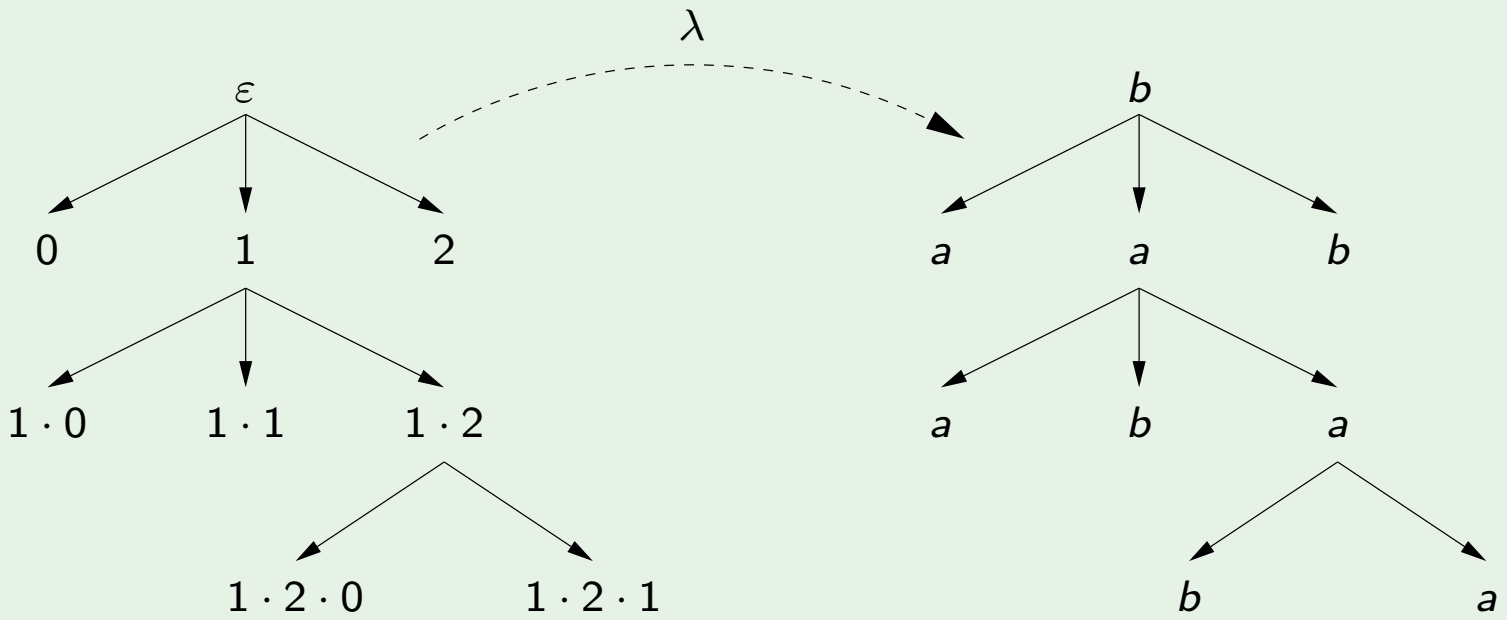
$\Sigma = \{a, b\}$ y $T = (D, \lambda)$:

$$
\begin{aligned}
D &= \{\varepsilon, 0, 1, 2, 1\cdot 0, 1\cdot 1, 1\cdot 2, 1\cdot 2\cdot 0, 1\cdot 2\cdot 1\} \\
\lambda &= \{\varepsilon \mapsto b, 0 \mapsto a, 1 \mapsto a, 2 \mapsto b, 1\cdot 0 \mapsto a, \\
&\qquad 1\cdot 1 \mapsto b, 1\cdot 2 \mapsto a, 1\cdot 2\cdot 0 \mapsto b, 1\cdot 2\cdot 1 \mapsto a\}
\end{aligned}
$$

Arboles sin rango

Example

Gráficamente:



Autómatas para árboles sin rango

$\mathcal{ER}(\Gamma)$: Expresiones regulares sobre Γ

Definición (Bottom-up, no determinista & sin rango)

$\mathcal{A} = (Q, \Sigma, \delta, F)$:

- Q *es un conjunto finito de estados*
- F *es un conjunto de estados finales*
- δ *es una función parcial*

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

tal que para todo $a \in \Sigma$ *y* $q \in Q$*, existe a lo más un* $r \in \mathcal{ER}(Q)$ *tal que* $\delta(r, a) = q$.

Notación: BU-UNTA

Ejecución de un Bu-UNTA

Dado: Bu-UNTA $\mathcal{A} = (Q, \Sigma, \delta, F)$ y $T = (D, \lambda)$ (con $D \neq \emptyset$).

- Para $w_1, w_2 \in D$: w_2 es un hijo de w_1 en D si $w_2 = w_1 \cdot i$ con $i \in \mathbb{N}$

Definición (Ejecución de un Bu-UNTA)

$\rho : D \to Q$ es una ejecución de $\mathcal{A}$ sobre T si para cada $w \in D$ con hijos $w \cdot 0$, $w \cdot 1$, ..., $w \cdot k$ en D, existe $r \in \mathcal{ER}(Q)$:

- $\rho(w \cdot 0)\rho(w \cdot 1) \cdots \rho(w \cdot k) \in L(r)$
- $\delta(r, \lambda(w))$ *está definido y* $\rho(w) = \delta(r, \lambda(w))$

Condición de aceptación: $\rho(\varepsilon) \in F$

Un ejemplo

Construya un $\textsc{Bu}$-$\textsc{UNTA}$ que acepte el lenguaje de los árboles sin rango sobre $\{a, b\}$ que tienen un número par de símbolos a.

Un ejemplo

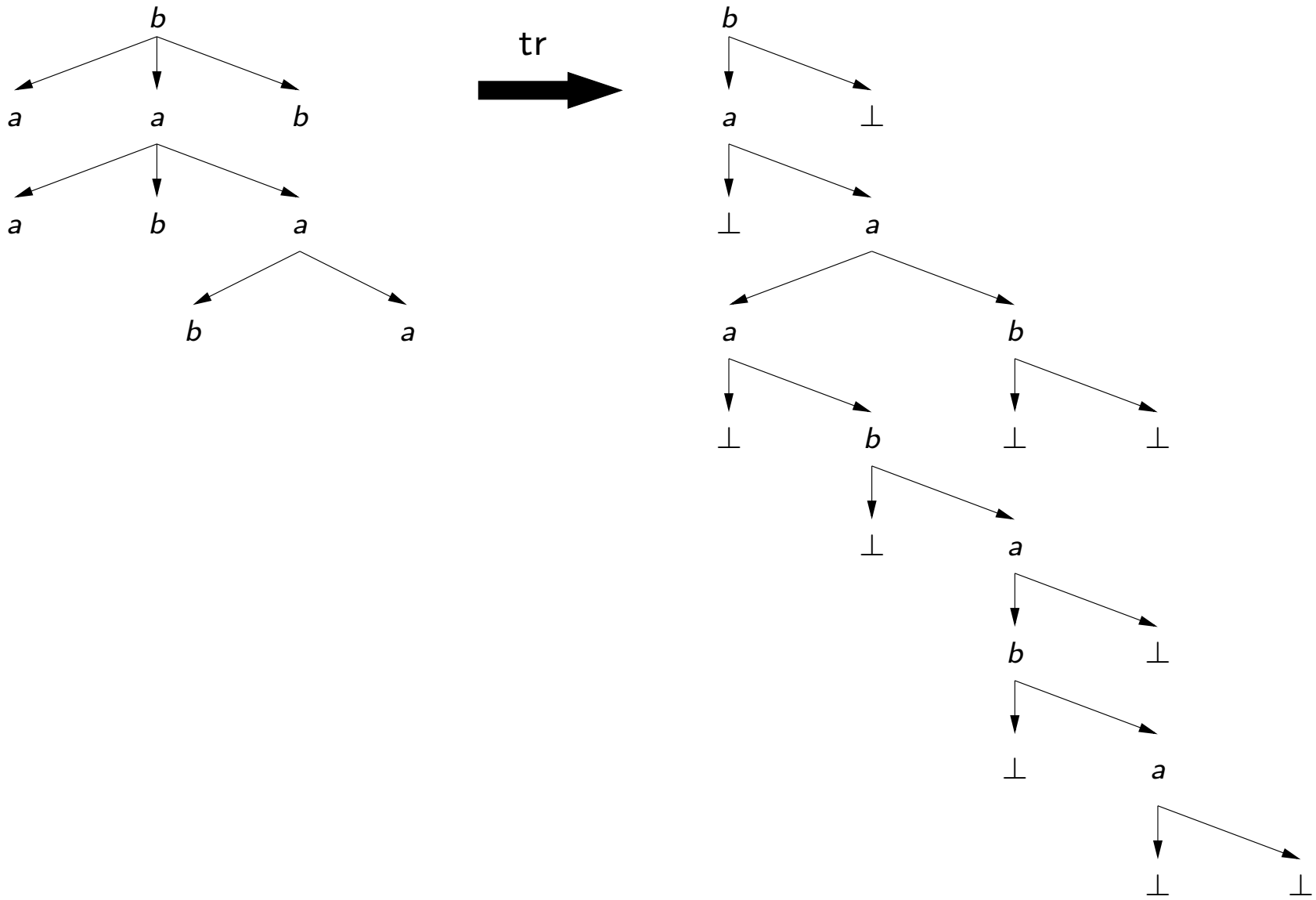
Construya un BU-UNTA que acepte el lenguaje de los árboles sin rango sobre $\{a, b\}$ que tienen un número par de símbolos a.

Solución

Sea $\mathcal{A} = (Q = \{q_0, q_1\}, \Sigma = \{a, b\}, \delta, F = \{q_0\})$ con:

$$\begin{aligned}
\delta((q_0^* q_1 q_0^* q_1 q_0^*)^*, a) &= q_1 \\
\delta((q_0^* q_1 q_0^* q_1 q_0^*)^*, b) &= q_0 \\
\delta(q_0^* q_1 q_0^* (q_0^* q_1 q_0^* q_1 q_0^*)^*, a) &= q_0 \\
\delta(q_0^* q_1 q_0^* (q_0^* q_1 q_0^* q_1 q_0^*)^*, b) &= q_1
\end{aligned}$$

Arboles sin rango versus árboles binarios



Arboles sin rango versus árboles binarios

Función tr puede ser calculada en tiempo polinomial.

Proposición

Existe un algoritmo polinomial para el siguiente problema: Dado un Bu-UNTA $\mathcal{A}$ *sobre un alfabeto* Σ*, construya un* Td-NTA $\mathcal{B}$ *sobre el alfabeto* $\Sigma \cup \{\perp\}$ *tal que para todo árbol* T *sobre* Σ*:*

$$T \in L(\mathcal{A}) \quad \text{si y sólo si} \quad tr(T) \in L(\mathcal{B})$$

Ejercicio

Demuestre la proposición.

Arboles sin rango versus árboles binarios

Corolario

Non-emptiness para Bu-UNTA *está en PTIME. Containment y equivalence para* Bu-UNTA *son EXPTIME-complete.*