

Teoría de Lenguajes

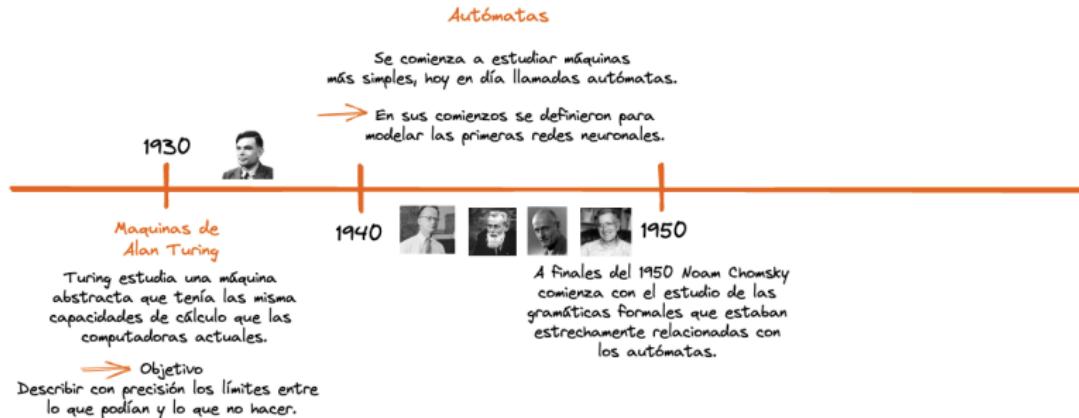
Autómatas de pila

Sabrina Silvero

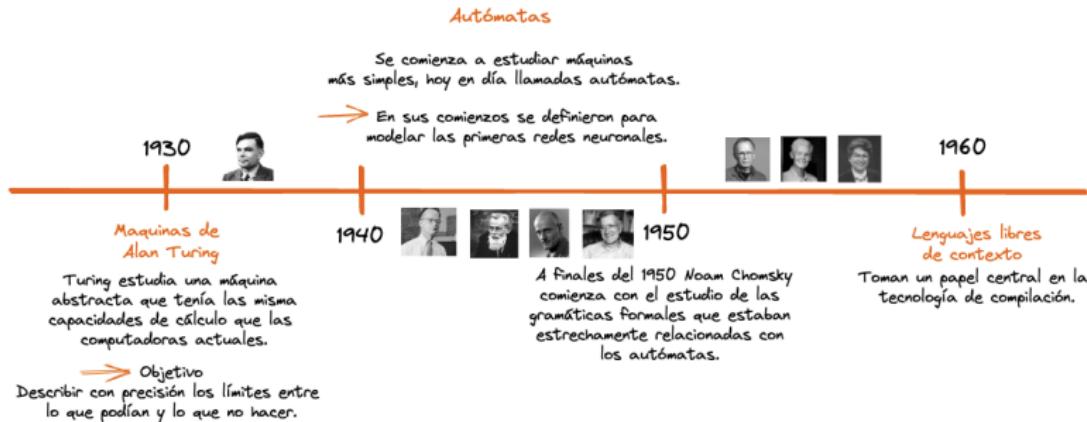
Departamento de Computación,
Facultad de Ciencias Exactas y Naturales,
Universidad de Buenos Aires

Segundo cuatrimestre 2024

¿Por qué estudiamos lo que estudiamos?



¿Por qué estudiamos lo que estudiamos?



¿Por qué estudiamos lo que estudiamos?

Los lenguajes independientes del contexto son reconocidos por los autómatas de pila y gramáticas libres de contexto.

Hoy vamos a trabajar con autómatas de pila, algunas utilidades de estos son:

- Análisis de protocolos de seguridad por Susan Landau
- Investigación en el razonamiento sobre ontologías y el procesamiento del lenguaje natural por Sheila McIlraith
- Análisis de modelos de sistemas de tiempo real por Christel Baier
- Verificación de programas concurrentes y la modelización de sistemas biológicos por Nancy Lynch



Susan Landau



Sheila McIlraith



Christel Baier



Nancy Lynch

Introducción

Lenguajes regulares

- Son los más simples, el tipo 3 de la jerarquía de Chomsky
- Reconocidos por autómatas finitos / expresiones regulares (formalismos equivalentes)
- No tienen memoria (no pueden contar, salvo que se requieran estados finito)
- Lema de pumping para demostrar que un lenguaje es no regular

Introducción

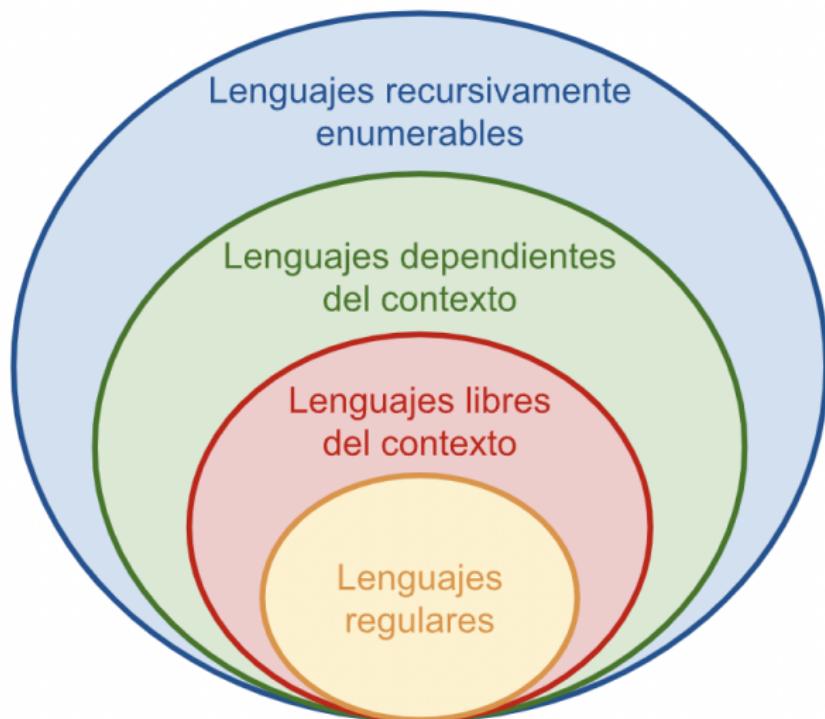
Lenguajes regulares

- Son los más simples, el tipo 3 de la jerarquía de Chomsky
- Reconocidos por autómatas finitos / expresiones regulares (formalismos equivalentes)
- No tienen memoria (no pueden contar, salvo que se requieran estados finito)
- Lema de pumping para demostrar que un lenguaje es no regular

Lenguajes independientes (libres) del contexto

- Tipo 2 en la jerarquía de Chomsky
- Reconocidos por autómatas de pila y gramáticas libres de contexto
- Mayor poder expresivo, la pila permite tener memoria y poder contar (ciertas cosas)

Jerarquía de Noam Chomsky



Autómatas de Pila

¿Podemos construir un autómata finito para reconocer los siguientes lenguajes?

$$L_a = \{s^3i^3\}$$

Autómatas de Pila

¿Podemos construir un autómata finito para reconocer los siguientes lenguajes?

$$L_a = \{s^3i^3\} \quad \checkmark$$

$$L_b = \{s^n i^n \mid 1 \leq n\}$$

Autómatas de Pila

¿Podemos construir un autómata finito para reconocer los siguientes lenguajes?

$$L_a = \{s^3i^3\} \quad \checkmark$$

$$L_b = \{s^n i^n \mid 1 \leq n\} \quad \times$$

$$L_c = \{s^j i^j \mid 1 \leq j \leq k \text{ con } k \text{ fijo } \in \mathbb{N}\}$$

Autómatas de Pila

¿Podemos construir un autómata finito para reconocer los siguientes lenguajes?

$$L_a = \{s^3i^3\} \quad \checkmark$$

$$L_b = \{s^n i^n \mid 1 \leq n\} \quad \times$$

$$L_c = \{s^j i^j \mid 1 \leq j \leq k \text{ con } k \text{ fijo } \in \mathbb{N}\} \quad \checkmark$$

Autómatas de Pila

$$L_b = L_1 = \{s^n i^n \mid 1 \leq n\}$$

En clases anteriores demostramos lenguajes muy similares a este no son regulares. Para poder reconocer lenguajes de este tipo necesitamos un formalismo con mayor poder expresivo: un **autómata de pila**.

Autómatas de Pila

$$L_b = L_1 = \{s^n i^n \mid 1 \leq n\}$$

En clases anteriores demostramos lenguajes muy similares a este no son regulares. Para poder reconocer lenguajes de este tipo necesitamos un formalismo con mayor poder expresivo: un **autómata de pila**.

- La pila es una cadena de símbolos
- Las transiciones pueden depender del símbolo en el tope de la pila, además del primer símbolo de la cadena de entrada
- En cada transición se desapila el símbolo del tope y se puede apilar una cadena

Definición

Un autómata de pila M se define como:

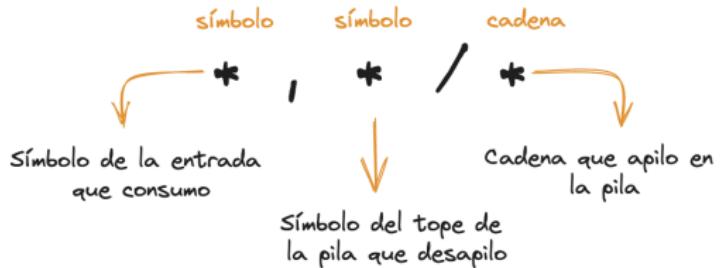
$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$$

Donde:

- Q es un conjunto finito de estados
- Σ es el alfabeto de entrada
- Γ es el alfabeto de la pila
- $q_0 \in Q$ es el estado inicial
- $Z_0 \in \Gamma$ es el símbolo inicial de la pila
- $F \subseteq Q$ es el conjunto de estados finales
- δ es la función de transición: $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow P(Q \times \Gamma^*)$

Formalizando

Las transiciones se definen de la siguiente manera:



Ejercicio 1

Construir un AP que reconozca el siguiente lenguaje,

$$L_1 = \{\omega \# \omega^r \mid w \in (a|b)^*\}$$

Ejercicio 1

Construir un AP que reconozca el siguiente lenguaje,

$$L_1 = \{\omega \# \omega^r \mid w \in (a|b)^*\}$$

Ejemplos de cadenas:

$$\alpha_1 = \# \checkmark \quad \alpha_2 = \lambda \times \quad \alpha_3 = a \# a \checkmark \quad \alpha_4 = ab \# ba \checkmark \quad \alpha_5 = abba \times$$

Ejercicio 1

Construir un AP que reconozca el siguiente lenguaje,

$$L_1 = \{\omega \# \omega^r \mid w \in (a|b)^*\}$$

Ejemplos de cadenas:

$$\alpha_1 = \# \checkmark \quad \alpha_2 = \lambda \times \quad \alpha_3 = a\#a \checkmark \quad \alpha_4 = ab\#ba \checkmark \quad \alpha_5 = abba \times$$

Las pilas funcionan de forma **LIFO** (Last In, First Out)

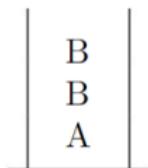


Solución ejercicio 1

Dado que nuestro alfabeto de entrada es $\Sigma = \{a, b\}$ en este caso tomaremos a $\Gamma = \{A, B, Z_0\}$ donde la representación será,



Vamos a aprovechar que la pila es LIFO para leer la reversa de una cadena apilada. Si $\alpha = abb$, entonces en la pila tenemos:



Solución ejercicio 1

Estrategia para construir el autómata de pila:

- Paso 1: Leemos ω y apilamos
- Paso 2: Leemos $\#$ sin tocar la pila
- Paso 3: Desapilamos verificando que coincide con ω^r

Solución ejercicio 1

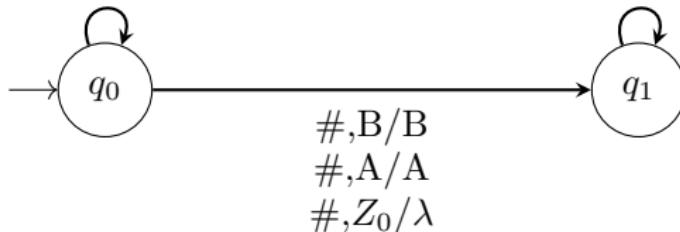
Estrategia para construir el autómata de pila:

- Paso 1: Leemos ω y apilamos
- Paso 2: Leemos $\#$ sin tocar la pila
- Paso 3: Desapilamos verificando que coincide con ω^r

$\delta_1 :$

$$\begin{array}{ll} a, Z_0/A & b, Z_0/B \\ a, A/AA & b, A/BA \\ a, B/AB & b, B/BB \end{array}$$

$$\begin{array}{l} a, A/\lambda \\ b, B/\lambda \end{array}$$

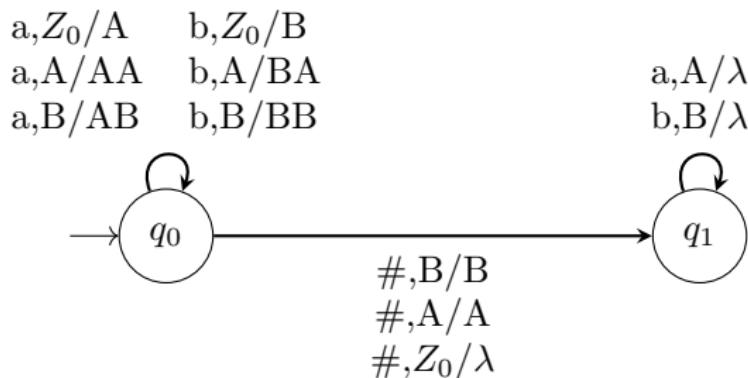


Solución ejercicio 1

Estrategia para construir el autómata de pila:

- Paso 1: Leemos ω y apilamos
- Paso 2: Leemos $\#$ sin tocar la pila
- Paso 3: Desapilamos verificando que coincide con ω^r

$\delta_1 :$



$$M_1 = \langle \{q_0, q_1\}, \{a, b\}, \{A, B, Z_0\}, \delta, q_0, Z_0, \emptyset \rangle$$

Configuración instantánea

En un autómata finito, teníamos como configuración instantánea un elemento de:

$$Q \times \Sigma^*$$

donde q es el estado actual y σ es lo que resta por consumir de la cadena de entrada.

¿Cómo será una configuración instantánea en un autómata de pila?

Configuración instantánea

En un autómata finito, teníamos como configuración instantánea un elemento de:

$$Q \times \Sigma^*$$

donde q es el estado actual y α es lo que resta por consumir de la cadena de entrada.

¿Cómo será una configuración instantánea en un autómata de pila?

$$(q, \alpha, \gamma) \in Q \times \Sigma^* \times \Gamma^*$$

donde:

- q es el estado actual
- α es la cadena de entrada que resta consumir
- γ es el contenido de la pila

Lenguaje aceptado

Relación de transición entre configuraciones:

$\forall q_1, q_2 \in Q, a \in \Sigma, \alpha \in \Sigma^*, b \in \Gamma, \beta \in \Gamma^*:$

- $(q_1, a\alpha, b\gamma) \vdash (q_2, \alpha, \beta\gamma) \iff (q_2, \beta) \in \delta(q_1, a, b)$
- $(q_1, \alpha, b\gamma) \vdash (q_2, \alpha, \beta\gamma) \iff (q_2, \beta) \in \delta(q_1, \lambda, b)$

Notar que *siempre* se saca el tope de la pila. Si en una transición no se quiere modificar la pila hay que volver a apilar el mismo símbolo.

Lenguaje aceptado

Relación de transición entre configuraciones:

$\forall q_1, q_2 \in Q, a \in \Sigma, \alpha \in \Sigma^*, b \in \Gamma, \beta \in \Gamma^*:$

- $(q_1, a\alpha, b\gamma) \vdash (q_2, \alpha, \beta\gamma) \iff (q_2, \beta) \in \delta(q_1, a, b)$
- $(q_1, \alpha, b\gamma) \vdash (q_2, \alpha, \beta\gamma) \iff (q_2, \beta) \in \delta(q_1, \lambda, b)$

Notar que *siempre* se saca el tope de la pila. Si en una transición no se quiere modificar la pila hay que volver a apilar el mismo símbolo.

Lenguaje aceptado por pila vacía:

$$\alpha \in N(M) \iff \exists q_n \in Q \mid (q_0, \alpha, Z_0) \vdash^* (q_n, \lambda, \lambda)$$

Para los LIC aparece el concepto de Lenguaje aceptado por pila vacía: se acepta una cadena si existe una secuencia de transiciones por medio de la cual se consume toda la cadena dejando la pila vacía

Lenguaje aceptado por estado final

Lenguaje aceptado por estado final:

$$\alpha \in L(M) \iff \exists q_f \in F, \gamma \in \Gamma^* \mid (q_0, \alpha, Z_0) \vdash^* (q_f, \lambda, \gamma)$$

Notar qué en este caso nos importa en que estado termina y lo que contiene la pila nos deja de importar.

Estado final vs Pila vacía

Lenguaje aceptado por estado final: Lenguaje aceptado por pila vacía:

$$\alpha \in N(M) \iff \exists q_n \in Q \mid (q_0, \alpha, Z_0) \vdash^* (q_n, \lambda, \textcolor{brown}{\lambda})$$

$$\alpha \in L(M) \iff \exists q_f \in F, \gamma \in \Gamma^* \mid (q_0, \alpha, Z_0) \vdash^* (q_f, \lambda, \textcolor{brown}{\gamma})$$

Solución ejercicio 1

Esta solución es por pila vacía. ¿Si quisiéramos hacerlo por estado final, qué deberíamos hacer?

Solución ejercicio 1

Esta solución es por pila vacía. ¿Si quisiéramos hacerlo por estado final, qué deberíamos hacer?

δ_{1F} :

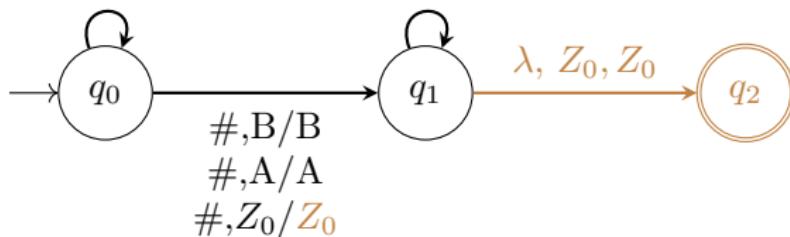
a, Z_0/A $\textcolor{brown}{Z}_0$ b, Z_0/B $\textcolor{brown}{Z}_0$

a,A/AA b,A/BA

a,B/AB b,B/BB

a,A/ λ

b,B/ λ



$$M_{1F} = \langle \{q_0, q_1, q_2\}, \{a, b\}, \{A, B, Z_0\}, \delta, q_0, Z_0, \{q_2\} \rangle$$

Ejercicio 2

Construir un AP que reconozca el siguiente lenguaje,

$$L_2 = \{\omega\omega^r \mid w \in (a|b)^*\}$$

Ejemplos de cadenas:

$$\alpha_1 = \lambda \checkmark \quad \alpha_2 = b \times \quad \alpha_3 = aa \checkmark \quad \alpha_4 = baab \checkmark \quad \alpha_5 = baba \times$$

Solución ejercicio 2

Estrategia para construir el autómata de pila:

- Paso 1: Leemos ω y apilamos
- Paso 2: ¿?
- Paso 3: Desapilamos verificando que coincide con ω^r

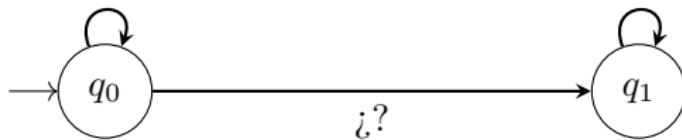
Solución ejercicio 2

Estrategia para construir el autómata de pila:

- Paso 1: Leemos ω y apilamos
- Paso 2: $\lambda?$
- Paso 3: Desapilamos verificando que coincide con ω^r

$\delta_2 :$

$$\begin{array}{lll} a, Z_0/A & b, Z_0/B \\ a, A/AA & b, A/BA & a, A/\lambda \\ a, B/AB & b, B/BB & b, B/\lambda \end{array}$$



Solución ejercicio 2

¿Como sabemos que terminamos de leer ω ?

Solución ejercicio 2

¿Como sabemos que terminamos de leer ω ?

No lo sabemos, hay que predecirlo.

Solución ejercicio 2

¿Como sabemos que terminamos de leer ω ?

No lo sabemos, hay que predecirlo.



No serán 14.000.605 alternativas, pero dependiendo del autómata pueden ser muchas.

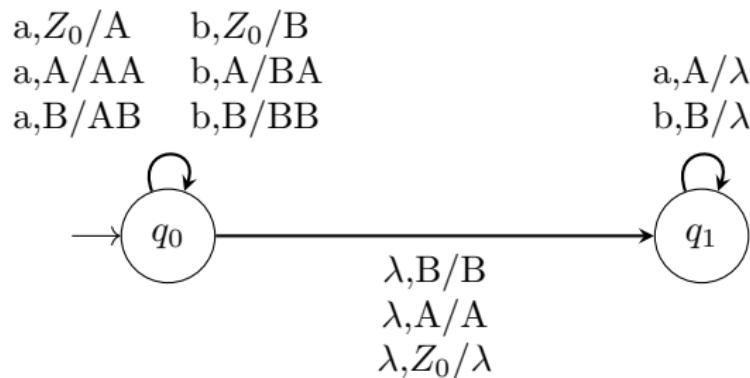
Solución ejercicio 2

- Paso 1: Leemos ω y apilamos
- Paso 2: Decidir no deterministicamente que terminé de consumir ω y comienzo a leer ω^r
- Paso 3: Desapilamos verificando que coincida con ω^r

Solución ejercicio 2

- Paso 1: Leemos ω y apilamos
- Paso 2: Decidir no deterministicamente que terminé de consumir ω y comienzo a leer ω^r
- Paso 3: Desapilamos verificando que coincida con ω^r

$\delta_2 :$



$$M_2 = \langle \{q_0, q_1\}, \{a, b\}, \{A, B, Z_0\}, \delta, q_0, Z_0, \emptyset \rangle$$

Ejercicio 2

Nota: tenemos un AP no determinístico.

¿Cómo vemos eso? Por ejemplo estando en q_0 con B en el tope de la pila y teniendo un B por leer, puedo transicionar a q_0 por b, B/BB o transicionar a q_1 por λ , B/B.

Ejercicio 2

Nota: tenemos un AP no determinístico.

¿Cómo vemos eso? Por ejemplo estando en q_0 con B en el tope de la pila y teniendo un B por leer, puedo transicionar a q_0 por b, B/BB o transicionar a q_1 por λ , B/B.

¿Podemos eliminar el no-determinismo?

Ejercicio 2

Nota: tenemos un AP no determinístico.

¿Cómo vemos eso? Por ejemplo estando en q_0 con B en el tope de la pila y teniendo un B por leer, puedo transicionar a q_0 por b, B/BB o transicionar a q_1 por λ , B/B.

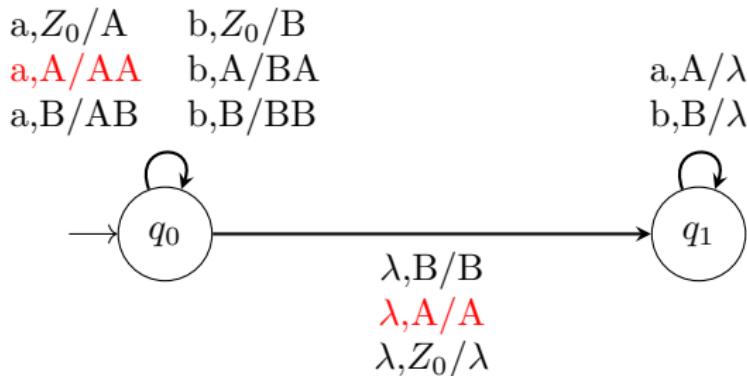
¿Podemos eliminar el no-determinismo? **NO**

Lo necesitamos porque no sabemos cuándo terminamos de leer ω y comenzamos con ω^r (comparar con el ejercicio 1).

Autómata determinístico

Un autómata de pila es determinístico si $\forall q \in Q, z \in \Gamma, a \in \Sigma$ vale:

- $|\delta(q, a, z)| \leq 1$
- $|\delta(q, \lambda, z)| \leq 1$
- $|\delta(q, \lambda, z)| = 1 \implies |\delta(q, a, z)| = 0$



ítem 1 ✓ ítem 2 ✓ ítem 3 ✗

¿Existe equivalencia entre APD y APND?

Mencionamos que no podemos construir un APD para

$$\{\omega\omega^r \mid w \in (a|b)^*\}$$

Es decir, existen lenguajes independientes del contexto que no son generados por ningún APD. Esto implica que (a diferencia de los autómatas finitos) no existe una equivalencia entre APD y APND.

También dijimos que los lenguajes generados por pila vacía con APND son los mismos que los generados por estados finales con APND. ¿Qué pasará si nos restringimos a APD?

¿Nos tomamos un break?



Ejercicio 3

Construir si se puede, un APD por pila vacía que acepte L_3 ,

$$L_3 = \{a^n b^m \mid 1 \leq n \leq m\}$$

Ejercicio 3

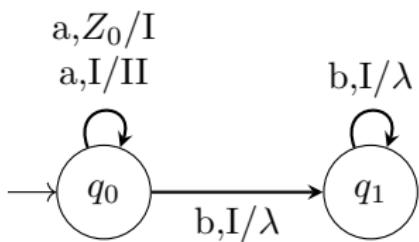
Construir si se puede, un APD por pila vacía que acepte L_3 ,

$$L_3 = \{a^n b^m \mid 1 \leq n \leq m\}$$

Para eso nos podemos basar en L_{T1} (de la teórica) que era "similar",

$$L_{T1} = \{a^n b^n \mid 1 \leq n\}$$

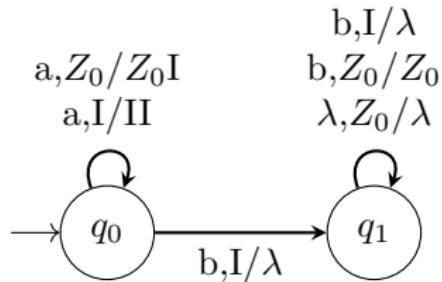
δ_{T1PV} :



Queremos que $|a| \leq |b|$, pero la pila se vacía cuando $|a| = |b|$. ¿Como lo podemos modificar para que cumpla con lo pedido?

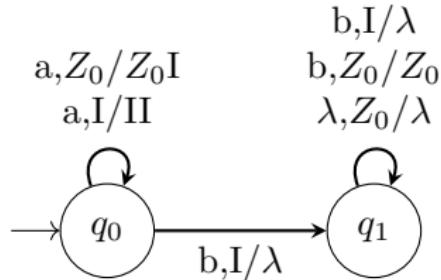
Solución ejercicio 3

Probemos manteniendo a Z_0 en la base de la pila:
 δ_{3PV} :



Solución ejercicio 3

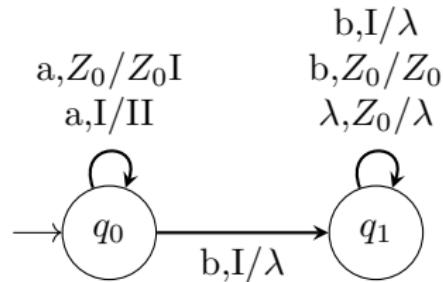
Probemos manteniendo a Z_0 en la base de la pila:
 δ_{3PV} :



Esto funciona pero es no determinístico.
¿Cómo se puede hacer determinístico?

Solución ejercicio 3

Probemos manteniendo a Z_0 en la base de la pila:
 δ_{3PV} :



Esto funciona pero es no determinístico.

¿Cómo se puede hacer determinístico?

No se puede. ☺

¿Por qué? Los lenguajes generados por APD por pila vacía son libres de prefijos. Es decir, si $\alpha \in L$ entonces $\forall \beta \neq \lambda, \alpha\beta \notin L$

Conclusiones

Los lenguajes generados por APD por pila vacía son libres de prefijos.

Es decir, si $\alpha \in L \implies \forall \beta \neq \lambda, \alpha\beta \in L$

Concluimos que los lenguajes generados por APD por pila vacía no son equivalentes a los lenguajes generados por APD por estados finales.

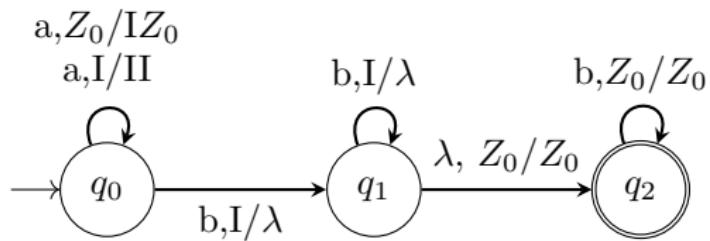
Solución ejercicio 3

Volviendo a nuestro lenguaje, ¿podemos dar un APD por estados finales?

Solución ejercicio 3

Volviendo a nuestro lenguaje, ¿podemos dar un APD por estados finales?

δ_{3EF} :

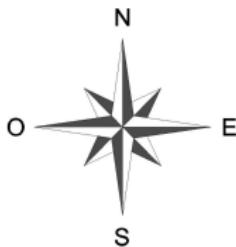


$$M_3 = \langle \{q_0, q_1, q_2\}, \{a, b\}, \{I, Z_0\}, \delta_{3EF}, q_0, Z_0, \{q_2\} \rangle$$

Concluimos que los lenguajes generados por APD por pila vacía **no son equivalentes** a los lenguajes generados por APD por estados finales.

Ejercicio 4 de la práctica

Dado el alfabeto $\{\uparrow, \leftarrow, \downarrow, \rightarrow\}$ podemos interpretar una cadena como una serie de pasos a dar sobre una cuadrícula. Por ejemplo, siguiendo la cadena $\leftarrow\leftarrow\leftarrow\downarrow$, terminamos tres pasos al oeste y un paso al sur del lugar donde comenzamos. Sea L_P el lenguaje de las cadenas que terminan dos pasos al norte del punto inicial (sin importar cuántos pasos al este o al oeste), y en las que un paso al sur nunca es inmediatamente seguido por un paso al este. Por ejemplo, $\downarrow\uparrow\rightarrow\uparrow\rightarrow\uparrow$ es una cadena de L_P , mientras que $\rightarrow\downarrow\rightarrow$ y $\uparrow\downarrow\rightarrow\uparrow\uparrow$ no lo son. Dar un autómata de pila que reconozca L_P . ¿Es un autómata determinístico?



Solución del ejercicio 4 de la práctica

Ejemplos de cadenas:

$$\alpha_1 = \uparrow\uparrow\uparrow\downarrow\rightarrow \checkmark \quad \alpha_2 = \rightarrow\uparrow\uparrow\downarrow \times \quad \alpha_3 = \uparrow\downarrow\leftarrow\uparrow\uparrow \checkmark$$

El alfabeto que vamos a usar es:

cadena		pila
\uparrow	\rightarrow	n
\downarrow	\rightarrow	s
\rightarrow	\rightarrow	e
\leftarrow	\rightarrow	o

Idea : podemos empezar apilando ss entonces cuando lleguemos a Z_0 es porque ya mos hemos movido dos pasos al norte.

Solución del ejercicio 4 de la práctica

- Paso 1: Apilamos ss (ver idea)
- Paso 2: Por cada \uparrow :
 - Si hay una s en el tope de la pila entonces desapilo
 - Si hay una n en el tope de la pila entonces apilo una n
 - Si hay Z_0 en el tope de la pila entonces apilo nZ_0

Por cada \downarrow :

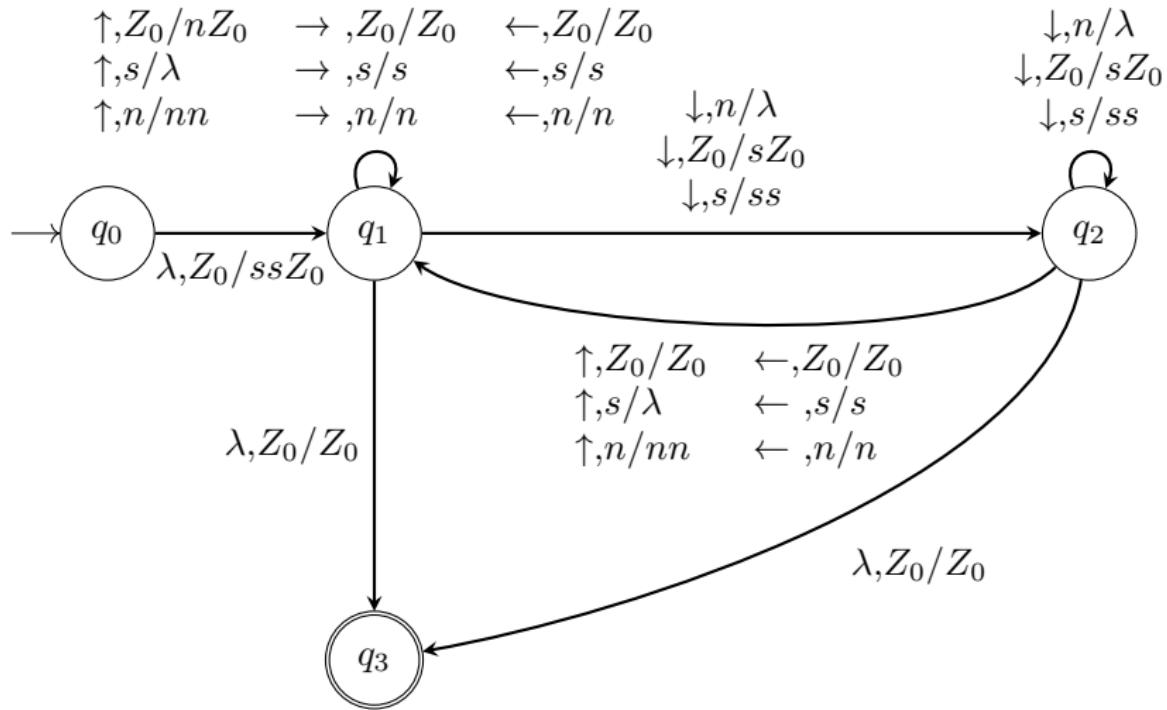
- Si hay una s en el tope de la pila entonces apilo una s
- Si hay una n en el tope de la pila entonces desapilo
- Si hay Z_0 en el tope de la pila entonces apilo sZ_0
- No le puede seguir un \rightarrow

Por cada \leftarrow o \rightarrow :

- Dejo la pila igual
- Paso 3: Acepto si termine de consumir toda la cadena y en la pila hay Z_0

Solución del ejercicio 4 de la práctica

$\delta_P :$



$$M_P = \langle \{q_0, q_1, q_2, q_3\}, \{a, b\}, \{n, sZ_0\}, \delta_P, q_0, Z_0, \{q_3\} \rangle$$

Cosas importantes

- LIC (tipo 2) tienen mayor poder expresivo que los LR, pueden contar ciertas cosas
- Los autómatas de pila por estado final aceptan una cadena si la misma se puede consumir en su totalidad y al finalizar se llega a un estado final, sin importar el estado de la pila
- Los autómatas de pila por pila vacía aceptan una cadena si la misma se puede consumir en su totalidad y si al finalizar se llega la pila esta vacía, sin importar en que estado está

Cosas importantes

- Los lenguajes generados por APND tienen mayor poder expresivo que los APD
- Los lenguajes generados por APND por estado final tienen el mismo poder expresivo que los APND por pila vacía
- Los lenguajes generados por APD por estado final tienen mayor poder expresivo que los APD por pila vacía
- Los lenguajes generados por APD por pila vacía son libres de prefijos

Bibliografía

Hopcroft J.E., Motwani R., Ullman J.D (2007). Introduction to Automata Theory, Languages, and Computation, 3rd edition. Addison Wesley. Capítulo 1 y 6.

Aho A.V. Ullman J.D. (1971). The Theory of Parsing, Translating and Compiling, Volume 1: Parsing. Prentice-Hall. Capítulo 2, sección 5.

¿Preguntas?

