Autómatas y Lenguajes Formales 2017-1
Facultad de Ciencias UNAM
Nota de Clase 6

Favio E. Miranda Perea A. Liliana Reyes Cabello Lourdes González Huesca

25 de septiembre de 2016

1. Lenguajes regulares o ¿no?

Los lenguajes que hemos tratado han sido caracterizados por ser sencillos, es decir son conjuntos de cadenas cuyas propiedades son fáciles de abstraer mediante lenguajes básicos y operaciones como la concatenación, unión y cerradura de Kleene. A partir de ellos, las propiedades de cerradura nos permiten construir nuevos lenguajes regulares: Si L, M son lenguajes regulares entonces:

$L \cup M$ es regular	LM es regular
$L^\star$ es regular	L^+ es regular
$\overline{L}$ es regular	$L \cap M$ es regular
$L - M$ es regular	

Así mismo, hemos considerado la equivalencia entre lenguajes regulares y los autómatas finitos a través del teorema de Kleene. Pero ¿cómo es posible decidir si un lenguaje dado, sin una representación de expresión regular o autómata que lo reconoce, es regular?

Un caso conocido de un lenguaje "sencillo" no regular es

$$L = \{a^i b^i \mid i \in \mathbb{N}\}$$

Veamos dos ejemplos de lenguajes no regulares al tratar de dar una expresión que los genere:

Ejemplo: Considere el lenguaje $L = \{a^i b^j \mid i \neq j,\ i, j \in \mathbb{N}\}$. Decida si es regular.
Para la demostración, supongamos primero que L es regular. Ahora procedemos a construir una expresión que lo represente: partimos del lenguaje $a^\star b^\star$ que claramente es regular.
Por propiedades de cerradura, el lenguaje $a^\star b^\star - L$ también debe ser regular.
¡Pero $a^\star b^\star - L = \{a^i b^i \mid i \in \mathbb{N}\}$ no es regular!
Por lo tanto L no puede ser regular.

Ejemplo: Ahora considere el lenguaje $L = \{wb^n \mid |w| = n, n \geq 1\}$. Decida si es regular.
Nuevamente, supongamos que L es regular. Y partimos otra vez de $a^\star b^\star$ que claramente es regular. Usando las propiedades de cerradura $L \cap a^\star b^\star$ también debe ser regular. Pero $L \cap a^\star b^\star = \{a^i b^i \mid \in \mathbb{N}\}$ no es regular.
Por lo tanto L no puede ser regular.

1.1. ¿Cuántos lenguajes regulares hay?

Los lenguajes son variados y requerimos de una forma eficiente para decidir si un lenguaje es regular y entonces atrevernos a proporcionar una máquina que lo reconozca.

Consideremos el conjunto de todos los lenguajes regulares, dado un alfabeto:

$$REG = \{L \subseteq \Sigma^\star \mid L \text{ es regular } \}$$

¿Cuál es la cardinalidad de REG? Analicemos el conjunto:

- Dado que cualquier lenguaje L es un subconjunto de $\Sigma^\star$, existen tantos lenguajes como elementos en $\mathcal{P}(\Sigma^\star)$.
- Puesto que $\Sigma^\star$ es infinito numerable, es decir es del tamaño del conjunto $\mathbb{N}$ de los números naturales, entonces $\mathcal{P}(\Sigma^\star)$ es del tamaño del conjunto de los números reales $\mathbb{R}$.
- Existen sólo tantos lenguajes regulares como números naturales, $|REG| = |\mathbb{N}|$:
  - La idea de la prueba es enumerar lexicográficamente *todos* los AFD posibles con alfabeto de entrada Σ, es decir, primero los autómatas con un sólo estado, luego los de dos estados, etc.
  - Esto implica que el número de lenguajes regulares es a lo más numerable.
  - Además, claramente es numerable pues hay una infinidad numerable de lenguajes regulares, por ejemplo
    $$\{a\}, \{aa\}, \{aaa\}, \ldots$$
- De manera que el conjunto $\mathcal{P}(\Sigma^\star) - REG$ no puede ser numerable, pues la unión de numerables sigue siendo numerable.
- Es decir, hay tantos lenguajes **no** regulares como números reales.

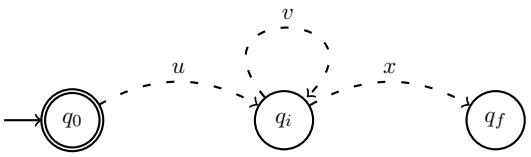
En esta nota nos dedicaremos a discutir dos métodos para probar que un lenguaje **no** es regular.

2. Lema de Bombeo para lenguajes regulares

Lema 1 (Lema del Bombeo) *Si L es un lenguaje regular infinito entonces existe un número $n \in \mathbb{N}$, llamado constante de bombeo para L, tal que para cualquier cadena w de L con $|w| \geq n$ existen cadenas u, v, x tales que:*

1. $w = uvx$
2. $v \neq \varepsilon$
3. $|uv| \leq n$
4. $\forall i \geq 0,\ uv^i x \in L$.

Demostración. Informalmente, la idea del lema se basa en que la característica de un lenguaje regular consiste en *repetir* o *ciclar* una subcadena y de ahí que exista un ciclo en un autómata que reconoce a L. Recordemos que un autómata es una máquina con una memoria corta, es decir sólo recuerda lo que está abstraido en un estado y no tiene memoria auxiliar. El siguiente diagrama resume el lema:



Para probar que un lenguaje L **no** es regular se procede por *contradicción* usando del Lema del Bombeo como sigue:

1. Si L fuera regular entonces existiría una constante de bombeo n.
2. Y cualquier palabra $w \in L$ con longitud mayor o igual a n se descompone como $w = uvx$ donde $v \neq \boldsymbol{\varepsilon},\ |uv| \leq n$.
3. Se llega a una contradicción como sigue:

   por el lema del bombeo la cadena uv^ix debe pertenecer a L, para toda $i \geq 0$. Pero por la definición particular de L, se puede mostrar alguna i tal que $uv^ix \notin L$.

Debemos observar que encontrar la i adecuada depende del problema particular y no hay un método general, pero usualmente basta con valores pequeños de i.

Ejemplo: Veamos que $L = \{a^ib^i \mid i \in \mathbb{N}\}$ **no** es regular usando el Lema del bombeo.
Supóngase que L es regular y sea n una constante de bombeo. Ahora considere una palabra cualquiera del lenguaje, es decir tiene la forma $w = a^nb^n$. Una descomposición de w en uvx es la siguiente: u, v contienen sólo a's, digamos

$$u = a^k,\ v = a^\ell,\ k \geq 0, \ell \geq 1$$

Con esto aseguramos que se cumple que $v \neq \varepsilon$, $|uv| \leq n$ y además $x = a^{n-k-\ell}b^n$.
Si tomamos $i = 2$, por el lema del bombeo la cadena uv^2x debe pertenecer a L. Pero, realmente sucede que

$$uv^2x = a^ka^\ell a^\ell a^{n-k-\ell}b^n = a^{n+\ell}b^n \notin L$$

Por lo tanto, L no es regular.

Ejemplo: Demuestre que $L_1 = \{w \in \{a, b\}^\star \mid w = w^R\}$ **no** es regular.
Suponer que L es regular y sea n una constante de bombeo. Ahora considere que la palabra $w = a^nb^na^n$ en L_1 tiene una descomposición en $w = uvx$ con $v \neq \varepsilon,\ |uv| \leq n$ y en donde u, v tienen sólo a's digamos $u = a^k,\ v = a^\ell,\ \ell \geq 1$.

Por tanto $x = a^{n-k-\ell}b^n a^n$. Al hacer $i = 2$ debemos tener a $uv^2x \in L$, por el lema del bombeo. Pero por otra parte se tiene que:

$$uv^2x = a^k a^\ell a^\ell a^{n-k-\ell} b^n a^n = a^{n+\ell} b^n a^n \notin L$$

Y por lo tanto L no es regular.

Así el lema anterior *no* permite demostrar que un lenguaje es regular, generalmente se usa como método de demostración para probar que un lenguaje **no** es regular. Veamos a continuación otro método, el cual permite decidir si un lenguaje es regular.

3. Lema de Myhill-Nerode

La estrecha relación entre lenguajes regulares y autómatas finitos invita a reflexionar de otra forma un método para identificar lenguajes regulares. Una manera de hacer lo anterior fue estudiada a través de la minimización de autómatas finitos al abstraer partes de una máquina que reconcen cierto tipo de subcadenas, las clases de equivalencia de estados.

El método expuesto en esta sección utiliza también relaciones de equivalencias pero sin considerar *a priori* una máquina que acepte cierto lenguaje, sino que se analizarán las cadenas de un lenguaje para decidir si éste es regular o no.

3.1. Relaciones de equivalencia sobre cadenas

Considérense las siguientes relaciones de equivalencia sobre $\Sigma^\star$ relacionadas a un lenguaje dado L y a un autómata finito determinista dado M, sean u, v dos cadenas:

- $u \equiv_L v$ si y sólo si
$$\forall w \in \Sigma^\star (uw \in L \Leftrightarrow vw \in L)$$
Se dice que u, v son cadenas **indistinguibles** para L.

- $u \equiv_M v$ si y sólo si
$$\delta^\star(q_0, u) = \delta^\star(q_0, v)$$
Es decir, u, v son cadenas **indistinguibles** según M.

Ejemplo: Dado el lenguaje $L = \{a^i b^i \mid i \in \mathbb{N}\}$ sobre $\Sigma = \{a, b\}$ se tiene que

$$x \equiv_L y \text{ si y sólo si } \forall z \in \Sigma^\star (xz \in L \Leftrightarrow yz \in L)$$

- $a^4b^3 \equiv_L a^3b^2$ pues
$$\forall z \in \Sigma^\star (a^4b^3z \in L \Leftrightarrow a^3b^2z \in L)$$
En particular con $z = b$, las concatenaciones están en L. En otro caso no lo están pero se cumple la equivalencia.

- $a^2b^2 \not\equiv_L a^3b^2$ pues para $z = \varepsilon$ se tiene
$$a^2b^2z \in L \text{ y } a^3b^2z \notin L$$

- $a^4b \not\equiv_L a^3b^2$ pues para $z = b$, se tiene

$$a^4b^2z \notin L \text{ y } a^3b^2z \in L$$

- $abb \equiv_L baba$ pues ambas cadenas no pertenecen a L.

- La relación $\equiv_L$ tiene una infinidad de clases de equivalencia, por ejemplo:

$$[\varepsilon], [a], [a^2], \ldots, [a^n], \ldots$$

  Todas estas clases son diferentes pues si $i \neq j$ entonces $a^i \not\equiv_L a^j$ ya que si consideramos a $z = b^i$ entonces $a^iz \in L$ pero $a^jz \notin L$.

Observación: Por lo general no hay relación alguna entre un lenguaje L y un autómata M. Más aún, la relación $\equiv_L$ puede definirse para cualquier lenguaje L aún cuando este no sea regular. Sin embargo, en el caso particular en que $L = L(M)$ se cumple que $\equiv_M$ es un *refinamiento* de $\equiv_L$:

Proposición: $\forall x, y \in \Sigma^\star(x \equiv_M y \rightarrow x \equiv_L y)$

Esta proposición nos deja ver la más importante limitación de los autómatas finitos, el hecho de que carecen de memoria más allá de lo que recuerde el estado actual:

si $x \equiv_M y$ entonces $x \equiv_{L(M)} y$, por lo que ninguna cadena w procesada después de x o y permitirá que M determine cuál de x o y se procesó anteriormente.

Preparémonos ahora para enunciar el siguiente método:

Definición 1 *Una relación de equivalencia $\equiv$ sobre $\Sigma^\star$ es invariante por la derecha si y sólo si*

$$\forall x, y, w \in \Sigma^\star(x \equiv y \rightarrow xw \equiv yw).$$

Así la relación $\equiv_L$ es invariante por la derecha.

Lema 2 (Lema de continuación) *Sean $x, y \in \Sigma^\star$. Si $\delta^\star(q_0, x) = \delta^\star(q_0, y)$ entonces para cualquier $z \in \Sigma^\star$, se cumple que*

$$\delta^\star(q_0, xz) = \delta^\star(q_0, yz)$$

Del lema anterior se sigue que la relación $\equiv_M$ es invariante por la derecha.

Propiedades de $\equiv_M$. Recordemos que el índice de una relación de equivalencia $\equiv$ es el número de clases de equivalencia generadas por $\equiv$. Así, dado un AFD $M = \langle Q, \Sigma, q_0, \delta, F\rangle$ se cumple lo siguiente:

- La relación $\equiv_M$ es invariante por la derecha.

- La relación $\equiv_M$ es de índice finito.

- $L(M)$ es la unión de algunas de las clases de equivalencia de la relación $\equiv_M$.

Teorema 1 (Myhill-Nerode) *Sea $L \subseteq \Sigma^\star$. Las siguientes condiciones son equivalentes:*

1. *L es regular.*
2. *Existe una relación de equivalencia $\equiv$ sobre $\Sigma^\star$, invariante por la derecha y de índice finito, tal que L es la unión de algunas de las clases de equivalencia de $\equiv$.*
3. *La relación de equivalencia $\equiv_L$ tiene índice finito.*

El teorema anterior permite demostrar que un lenguaje L **no** es regular al mostrar que L no es de índice finito. Es decir, basta ver que $\equiv_L$ tiene una infinidad de clases de equivalencia. Esto se hace explícito mediante el siguiente lema que es una consecuencia directa del teorema de Myhill-Nerode.

Lema 3 (Lema del índice finito) *Sea $L \subseteq \Sigma^\star$ un lenguaje regular infinito. Cualquier conjunto $S \subseteq \Sigma^\star$ suficientemente grande contiene al menos dos cadenas distintas, $x, y \in S$ tales que $x \equiv_L y$.*

Las pruebas de no regularidad se sirven de la contrapositiva del lema del índice finito. Es decir, hallando un conjunto $S \subseteq \Sigma^\star$ que no cumpla la propiedad del lema, habremos probado que L no es regular. La siguiente definición de conjuntos estafadores[1] servirá para encontrar los conjuntos que no cumplen la propiedad, así para mostrar que un lenguaje L **no** es regular basta construir un conjunto estafador para L.

Definición 2 *Un conjunto infinito $S \subseteq \Sigma^\star$ es un conjunto estafador para L si y sólo si para cualesquiera $x, y \in S$ existe una cadena $z \in \Sigma^\star$ tal que una y sólo una de xz y de yz pertenece a L. Es decir, S es un conjunto estafador para L si y sólo si*

$$\forall x, y \in S\,(x \not\equiv_L y).$$

Ejemplo: Demostrar que $L = \{a^i b^i \mid i \in \mathbb{N}\}$ **no** es regular usando el Teorema de Myhill-Nerode. Como dijimos antes, basta hallar un conjunto estafador para $L = \{a^n b^n \mid n \in \mathbb{N}\}$, es decir un conjunto de cadenas $S \subseteq \{a, b\}^\star$ tal que para las cadenas $x, y \in S$ y $z \in \Sigma^\star$ se tiene que $xz \in L$ y $yz \notin L$. Sea $S = \{a^k \mid k \in \mathbb{N}\}$, veamos que S es un conjunto estafador:

$$\text{si } a^i, a^j \in S \text{ con } i \neq j \text{ entonces claramente } a^i b^i \in L \text{ y } a^i b^j \notin L.$$

Por lo tanto $a^i \not\equiv_L a^j$ con S es un conjunto estafador para L.

Ejemplo: Decidir si el lenguaje $L = \{a^{i^2} \mid i \in \mathbb{N}\}$ es regular usando el Teorema de Myhill-Nerode. Analizando informalmente, se puede ver que este lenguaje requiere memoria más allá de la que ofrece un autómta finito: se necesita contabilizar i^2.
Tratemos de encontrar un conjunto estafador para el lenguaje. Sea S exactamente L, entonces sean $a^{i^2}, a^{j^2} \in S$ con $j > i$.

- Por un lado tenemos que $a^{i^2} a^{2i+1} = a^{i^2+2i+1} = a^{(i+1)^2} \in L$

[1] En inglés *fooling set*.

- Por otra parte, $a^{j^2}a^{2i+1} = a^{j^2+2i+1} \notin L$ puesto que $j^2 < j^2 + 2i + 1 < j^2 + 2j + 1 = (j+1)^2$.

Por lo tanto $a^{i^2} \not\equiv_L a^{j^2}$ y S es un conjunto estafador para L. Podemos concluir que el lenguaje L no es regular.

Usar el teorema anterior para demostrar que un lenguaje es regular también es posible al proporcionar una relación de equivalencia $\equiv$ invariante por la derecha y de índice finito, es decir, se debe dar $\equiv_L$ tal que es la unión de algunas clases de equivalencia sobre $\Sigma^\star$. Para ello se siguen los siguientes pasos:

1. Proponer unas clases de equivalencia que describan las cadenas en el lenguaje, una de las clases debe ser la clase que contiene a la cadena vacía: $[\boldsymbol{\varepsilon}]$.

2. Mostrar que todas las clases antes propuestas se distinguen y son invariantes por la derecha siguiendo la definición.
   Para mostrar que dos clases se distinguen, se toman dos cadenas u, v una en cada clase y se suponen equivalentes $u \equiv_L v$. Basta encontrar una cadena w tal que $uw \in L$ y $vw \notin L$ para romper la definición y decir que las dos clases se distinguen.

3. Crear un autómata finito usando las clases anteriores de la siguiente forma:

   - Los estados son las clases de equivalencia.
   - El estado inicial es la clase $[\boldsymbol{\varepsilon}]$.
   - Los estados finales son aquellas clases que contienen a las cadenas en el lenguaje.
   - La función de transicón está dada por:

     $$\forall a \in \Sigma, \delta(q_{[w]}, a) = [w']$$

     donde w es el representante de la clase $[w]$ y wa pertenece a la clase $[w']$.

Ejemplo: Decidir si el lenguaje $L = \{w \in \{a,b\}^\star \mid bb \text{ no es subcadena de } w \text{ y } |w| = 2k, k \in \mathbb{N}\}$ es regular.
Después de un análisis informal, podemos pensar que L es regular dado que se deben identificar cadenas de longitud par y deshechar las cadenas que contengan bb.
Veamos cómo usar el Teorema de Myhill-Nerode para obtener un autómata que reconozca este lenguaje.
Se debe mostrar que $\equiv_L$ es la unión de algunas clases de equivalencia sobre $\{a,b\}^\star$. Se proponen las siguientes clases:

- La clase $[\boldsymbol{\varepsilon}]$ representa a las cadenas de longitud par sin la subcadena bb.
- La clase $[ab]$ que representa a las cadenas de longitud par con potencial para tener la subcadena bb.
- La clase $[bb]$ representa a las cadenas que contienen a la subcadena bb.
- La clase $[a]$ engloba a las cadenas de longitud impar sin la subcadena bb.
- La clase $[b]$ contiene a las cadenas de longitud impar con potencial para tener a la subcadena bb.

Estas cuatro clases separan a las cadenas en $\{a, b\}^\star$, de ellas sólo la clase $[\varepsilon]$ es la que contiene las cadenas en L.

Para mostrar que las cuatro clases se distiguen se comparan dos a dos:

- Sean $\varepsilon \in [\varepsilon]$ y $bb \in [bb]$, suponer que $\varepsilon \equiv_L bb$. Tomar $w = \varepsilon$, entonces $\varepsilon\varepsilon = \varepsilon$ está en el lenguaje pero $bb\varepsilon = bb$ no lo está.
- Sean $ab \in [ab]$ y $bb \in [bb]$, suponer que $ab \equiv_L bb$. Tomar $w = \varepsilon$, entonces $ab\varepsilon = ab$ está en el lenguaje pero $bb\varepsilon = bb$ no lo está.
- Sean $ab \in [ab]$ y $b \in [b]$, suponer que $ab \equiv_L b$. Tomar $w = \varepsilon$, entonces $ab\varepsilon = ab$ está en el lenguaje pero $b\varepsilon = b$ no lo está.
- Sean $a \in [a]$ y $b \in [b]$, suponer que $a \equiv_L b$. Sea $w = b$ entonces ab está en el lenguaje pero bb no está.
- Sean $\varepsilon \in [\varepsilon]$ y $a \in [a]$, suponer que $\varepsilon \equiv_L a$. Tomar $w = b$ entonces $\varepsilon b = b$ no está en L y ab sí está en L.
- Sean $b \in [b]$ y $bb \in [bb]$ y suponer que $b \equiv_L bb$. Sea $w = a$ entonces ba que pertenece a L y bba que no pertenece a L.

Ahora crearemos el autómata a través de la tabla de transiciones δ, abajo está su representación en diagrama:

$wa \in w'$	$wb \in w''$		δ	a	b
$\varepsilon a \in [a]$	$\varepsilon b \in [b]$	$inicial,\ final$	$q_{[\varepsilon]}$	$q_{[a]}$	$q_{[b]}$
$aa \in [\varepsilon]$	$ab \in [ab]$		$q_{[a]}$	$q_{[\varepsilon]}$	$q_{[ab]}$
$aba \in [a]$	$abb \in [bb]$	$final$	$q_{[ab]}$	$q_{[a]}$	$q_{[bb]}$
$ba \in [\varepsilon]$	$bb \in [bb]$		$q_{[b]}$	$q_{[\varepsilon]}$	$q_{[bb]}$
$bba \in [bb]$	$bbb \in [bb]$		$q_{[bb]}$	$q_{[bb]}$	$q_{[bb]}$