Expresiones Regulares

Automatas Finitos y Expresiones Regulares

Fabio Martínez Carrillo

Matemáticas Discretas
Escuela de Ingeniería de Sistemas e Informatica
Universidad Industrial de Santander - UIS

5 de octubre de 2017

Agenda

1. Operadores

2. Automatas Finitos y Expresiones Regulares

Expresiones Regulares

Expresión algebraica que puede describir de forma exacta los lenguajes de los autómatas.

- Forma declarativa de expresar las cadenas que se deben aceptar, e.g., grep: hol*, *.jpeg
- Si E es una expresión regular, entonces $L(E)$

Operadores de las Expresiones Regulares

Union, concatenación, y Clausura (estrella de Kleene)

Union: $\cup$

- Operación natural porque los lenguajes están definidos como conjuntos
- *e.g.:* $\{00, 111, 10\} \cup \{00, 01\}$
  -
- $\{001, 10, 111\} \cup \{\varepsilon, 001\}$
  -

Operadores de las Expresiones Regulares

Union, concatenación, y Clausura (estrella de Kleene)

Union: $\cup$

- Operación natural porque los lenguajes están definidos como conjuntos
- *e.g.:* $\{00, 111, 10\} \cup \{00, 01\}$
  - $\{00, 111, 10\} \cup \{00, 01\} = \{00, 111, 10, 01\}$
- $\{001, 10, 111\} \cup \{\varepsilon, 001\}$
  -

Operadores de las Expresiones Regulares

Union, concatenación, y Clausura (estrella de Kleene)

Union: $\cup$

- Operación natural porque los lenguajes están definidos como conjuntos
- *e.g.:* $\{00, 111, 10\} \cup \{00, 01\}$
    -
- $\{001, 10, 111\} \cup \{\varepsilon, 001\}$
    - $\{001, 10, 111\} \cup \{\varepsilon, 001\} = \{\varepsilon, 10, 001, 111\}$

La concatenación

La concatenación de los lenguajes L y M es LM

- Contiene el conjunto de cadenas de la forma wx tal que $w \in L$ y $x \in M$

Ejemplo

- $\{00, 111, 10\}\{00, 01\}$
  -
- $\{001, 10, 111\}\{\varepsilon, 001\}$
  -

La concatenación

La concatenación de los lenguajes L y M es LM

- Contiene el conjunto de cadenas de la forma wx tal que $w \in L$ y $x \in M$

Ejemplo

- $\{00, 111, 10\}\{00, 01\}$
  - $\{00, 111, 10\}\{00, 01\} = \{0000, 0001, 11100, 11101, 1000, 1001\}$
- $\{001, 10, 111\}\{\varepsilon, 001\}$
  -

La concatenación

La concatenación de los lenguajes L y M es LM

- Contiene el conjunto de cadenas de la forma wx tal que $w \in L$ y $x \in M$

Ejemplo

- $\{00, 111, 10\}\{00, 01\}$
  -
- $\{001, 10, 111\}\{\varepsilon, 001\}$
  - $\{001, 10, 111\}\{\varepsilon, 001\} = \{001, 10, 111, 001001, 10001, 111001\}$

La Claussura (Estrella de Kleene) L^*

Es el conjunto de cadenas formado al concatenar cero o mas estring en L en cualquier orden.

$$L^* = \bigcup_{i \geq 0} L^i = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$$

donde, $L^0 = \{\varepsilon\}$ y $L^1 = L$

Ejemplo

- $\{0, 10\}^*$
- Si $\{0, 11\}^*$ entonces $01011 \in L^*$

La Claussura (Estrella de Kleene) L^*

Es el conjunto de cadenas formado al concatenar cero o mas estring en L en cualquier orden.

$$L^* = \bigcup_{i \geq 0} L^i = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$$

donde, $L^0 = \{\varepsilon\}$ y $L^1 = L$

Ejemplo

- $\{0, 10\}^* = \{\varepsilon, 0, 10, 00, 010, 100, 101, \ldots\}$
- Si $\{0, 11\}^*$ entonces $01011 \in L^*$

La Claussura (Estrella de Kleene) L^*

Es el conjunto de cadenas formado al concatenar cero o mas estring en L en cualquier orden.

$$L^* = \bigcup_{i \geq 0} L^i = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$$

donde, $L^0 = \{\varepsilon\}$ y $L^1 = L$

Ejemplo

- $\{0, 10\}^*$
- Si $\{0, 11\}^*$ entonces $01011 \in L^*$ NO

Definición de expresiones regulares

El álgebra nos permite construir mas expresiones a partir de expresiones elementales y ciertas operaciones.

- Definición de expresiones regulares validas E
- Lenguaje de representación $L(E)$

Caso base 1

Si a es algún simbolo, entonces **a** es una expresión regular y $L(\mathbf{a}) = \{a\}$. En el lenguaje a es una cadena con $|a| = 1$

Caso base 2

Si ε es una expresión regular y $L(\varepsilon) = \{\varepsilon\}$

Caso base 3

Si $\emptyset$ es una expresión regular y $L(\emptyset) = \emptyset$

Paso Inductivo 1

Si E_1 y E_2 son expresiones regulares, entonces $E_1 + E_2$ es una expresión regular.

$$L(E_1 + E_2) = L(E_1) \cup L(E_2)$$

Paso Inductivo 2

Si E_1 y E_2 son expresiones regulares, entonces $(E_1)(E_2)$ es una expresión regular.

$$L(E_1 E_2) = L(E_1)L(E_2)$$

Paso Inductivo

Si E es una expresion regular, entonces E^* es una expresión regular.

$$L(E^*) = (L(E))^*$$

Precedencia de Operadores

- Los parentesis pueden ser utilizados para agrupar operadores: $L((E)) = L(E)$
- Orden de precedencia:
  - * (mas alto)
  - Concatenación:
  - Union: $+$ (mas bajo)

Como se aplica la siguiente expresión: $01^* + 1$

Ejemplo

- $L(\mathbf{01}) = \{01\}$
- $L(\mathbf{01} + \mathbf{0}) = \{01, 0\}$
- $L(\mathbf{0}(\mathbf{1} + \mathbf{0})) = \{01, 00\}$

- Escriba una expresión regular que contenga cadenas con 0's y 1's alternos.

Precedencia de Operadores

- Los parentesis pueden ser utilizados para agrupar operadores: $L((E)) = L(E)$
- Orden de precedencia:
  - $*$ (mas alto)
  - Concatenación:
  - Union: $+$ (mas bajo)

Como se aplica la siguiente expresión: $01^* + 1$

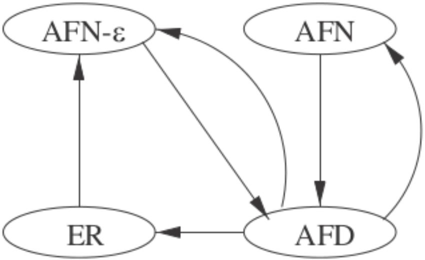
Ejemplo

- $L(\mathbf{01}) = \{01\}$
- $L(\mathbf{01} + \mathbf{0}) = \{01, 0\}$
- $L(\mathbf{0}(\mathbf{1} + \mathbf{0})) = \{01, 00\}$

- Escriba una expresión regular que contenga cadenas con 0's y 1's alternos.
- $L((\varepsilon + \mathbf{1})(\mathbf{01})^*(\varepsilon + \mathbf{0}))$

Automatas Finitos y Expresiones Regulares

- Cada expresión regular es un automata finito que acepta el mismo lenguaje.
  - Lo mas sencillo es demostrarlo con un AFN - ε
- Cada automata finito es una expresión regular del lenguaje.
  - Seleccionamos el tipo mas restrictivo de automatas: AFD



Agenda

1. Operadores
2. Automatas Finitos y Expresiones Regulares

De los AFD a las Expresiones Regulares

Objetivo

Contruir expresiones que describan conjunto de cadenas que etiqueten ciertos caminos de transición de un AFD.

- Expresiones simples (nodos o arcos)
- Contruir inductivamente expresiones que pasen a través de los caminos
- Progresivamente recorrer conjuntos de estados mas grandes
- Finalmente, los caminos podran pasar por cualquier estado

Teorema

Si L es $L(A)$ para algún Automata A, entonces existe una expresión regular R tal que $L = L(R)$

Teorema

Si L es $L(A)$ para algún Automata A, entonces existe una expresión regular R tal que $L = L(R)$

Demostración

- Suponemos que A tiene n estados $\{1, 2, \ldots, n\}$.
- Contruimos una colección de expresiones de forma progresiva para describir conjuntos cada vez mas amplios.
- $R_{ij}^{(k)}$ expresión regular cuyo lenguaje es el conjunto de w
  - w es la etiqueta de un camino desde i hasta j
  - El camino no tiene nodo intermedio cuyo número sea mayor que k

Contrucción de expresiones $R_{ij}^{(k)}$

Definición inductiva desde $k = 0$ hasta $k = n$. En n no hay ninguna restricción de los caminos.

- **Caso Base:** $k = 0$
  - 1) Un arco desde el (estado) nodo i hasta j
  - 2) Un camino de longitud cero, que consta de solo nodo (i)
- Si $i \neq j$ solo se lleva el paso 1)
  - Si no existe el simbolo a, $R_{ij}^{0} = \emptyset$
  - Si existe un solo simbolo a, $R_{ij}^{0} = a$
  - Si existen $a_1, a_2, \ldots, a_k$, entonces $R_{ij}^{0} = a_1 + a_2 + \cdots + ak$

Contrucción de expresiones $R_{ij}^{(k)}$

Definición inductiva desde $k = 0$ hasta $k = n$. En n no hay ninguna restricción de los caminos.

- **Caso Base:** $k = 0$
  - 1) Un arco desde el (estado) nodo i hasta j
  - 2) Un camino de longitud cero, que consta de solo nodo (i)
- Si $i = j$ solo se lleva el paso 1)
  - Los caminos de longitud 0 se representan ε
  - Se adicionan ε a todos los pasos anteriores
    - Si no existe el simbolo a, es ε
    - Si existe un solo simbolo a, $a + \varepsilon$
    - Si existen $a_1, a_2, \ldots, a_k$, entonces $\varepsilon + a_1 + a_2 + \cdots + ak$

Paso Inductivo

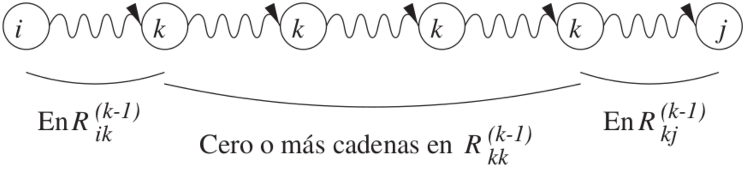
Si existe un camino desde *i* hasta *j* que no pasa por estados mayores a *k*

1. El camino no pasa por *k*, entonces la etiqueta sobre el camino es R_{ij}^{k-1}
2. El camino pasa a través del estado *k* al menos una vez. Entonces:

$$R_{ik}^{(k-1)}(R_{kk}^{(k-1)})^{*}R_{kj}^{(k-1)}$$

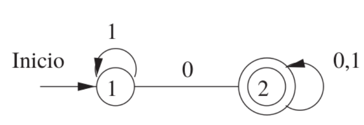
3. Si combinamos las dos expresiones:

$$R_{ij}^{k} = R_{ij}^{k-1} + R_{ik}^{(k-1)}(R_{kk}^{(k-1)})^{*}R_{kj}^{(k-1)}$$

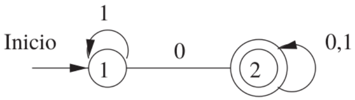


$R_{ij}^{(n)}$ para todo i y j. La expresión regular del lenguaje es la suma (unión) de todas las expresiones $R_{1j}^{(n)}$ tales que el estado j es el estado de de aceptación

Ejemplo: calcular la expresión regular



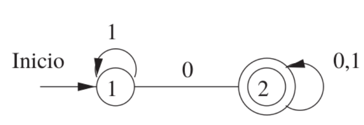
Ejemplo: calcular la expresión regular



1) Expresiones Básicas

$R_{11}^{(0)}$	$\varepsilon + \mathbf{1}$
$R_{12}^{(0)}$	$\mathbf{0}$
$R_{21}^{(0)}$	$\emptyset$
$R_{22}^{(0)}$	$(\varepsilon + \mathbf{0} + \mathbf{1})$

Ejemplo: calcular la expresión regular



Calculamos R_{ij}^{1} y $R_{ij}^{(2)}$

$$R_{ij}^{1} = R_{ij}^{0} + R_{i1}^{(0)}(R_{11}^{(0)})R_{1j}^{(0)}$$

	Por sustitución directa	Simplificada
$R_{11}^{(1)}$	$\varepsilon + \mathbf{1} + (\varepsilon + \mathbf{1})(\varepsilon + \mathbf{1})^*(\varepsilon + \mathbf{1})$	$\mathbf{1}^*$
$R_{12}^{(1)}$	$\mathbf{0} + (\varepsilon + \mathbf{1})(\varepsilon + \mathbf{1})^*\mathbf{0}$	$\mathbf{1}^*\mathbf{0}$
$R_{21}^{(1)}$	$\emptyset + \emptyset(\varepsilon + \mathbf{1})^*(\varepsilon + \mathbf{1})$	$\emptyset$
$R_{22}^{(1)}$	$\varepsilon + \mathbf{0} + \mathbf{1} + \emptyset(\varepsilon + \mathbf{1})^*\mathbf{0}$	$\varepsilon + \mathbf{0} + \mathbf{1}$

- La simplificación se obtiene: $(\varepsilon + R)^* = R^*$
- $\emptyset R = R\emptyset = \emptyset$
- $\emptyset + R = R + \emptyset = R$

	Por sustitución directa	Simplificada
$R_{11}^{(2)}$	$\mathbf{1}^* + \mathbf{1}^*\mathbf{0}(\varepsilon + \mathbf{0} + \mathbf{1})^*\emptyset$	$\mathbf{1}^*$
$R_{12}^{(2)}$	$\mathbf{1}^*\mathbf{0} + \mathbf{1}^*\mathbf{0}(\varepsilon + \mathbf{0} + \mathbf{1})^*(\varepsilon + \mathbf{0} + \mathbf{1})$	$\mathbf{1}^*\mathbf{0}(\mathbf{0} + \mathbf{1})^*$
$R_{21}^{(2)}$	$\emptyset + (\varepsilon + \mathbf{0} + \mathbf{1})(\varepsilon + \mathbf{0} + \mathbf{1})^*\emptyset$	$\emptyset$
$R_{22}^{(2)}$	$\varepsilon + \mathbf{0} + \mathbf{1} + (\varepsilon + \mathbf{0} + \mathbf{1})(\varepsilon + \mathbf{0} + \mathbf{1})^*(\varepsilon + \mathbf{0} + \mathbf{1})$	$(\mathbf{0} + \mathbf{1})^*$

Expresión final

Se obtiene como la unión de todas las expresiones en las que el primer estado sea el estado inicial y el segundo estado sea el estado de aceptación:

$$\mathbf{1}^*\mathbf{0}(\mathbf{0} + \mathbf{1})^*$$

AFD a ER usando eliminación de estados

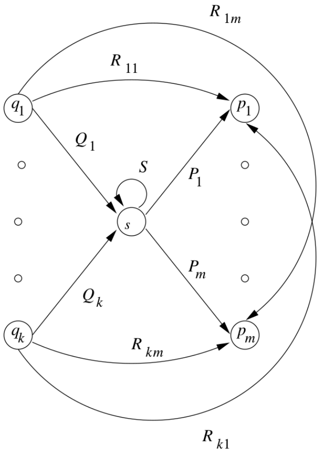
- Conversión típica puede aplicarse a un AFN o un AFN_{ε}
- Es costosa, requiere la construcción de n^3 expresiones
- Las expresiones pueden llegar a ser de 4^n simbolos
- Para todo i, j la expresión $R_{ij}^{(k)}$ la expresión usa $(R_{kk}^{(k-1)})^*$
- Esta expresión se repite n^2 veces

Eliminación de etiquetas

Una solución alternativa

- Elminar un estado s conlleva a eliminar todos los caminos que pasan por s
- Incluimos las etiqutas de s en arco directo de p hasta q
- Debemos considerar expresiones regulares como etiquetas.

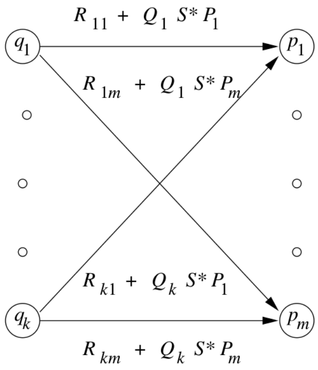
Eliminación de etiquetas



El estado s va a ser eliminado

- Los estados predecesores: $q_1, q_2, \ldots q_k$
- Los estados sucesores: $p_1, p_2, \ldots p_k$

Eliminación de etiquetas

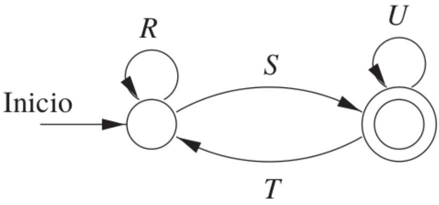


El estado s va a ser eliminado

- Los estados predecesores: $q_1, q_2, \ldots q_k$
- Los estados sucesores: $p_1, p_2, \ldots p_k$

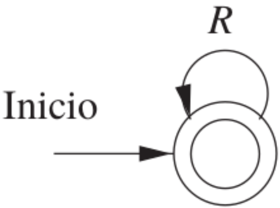
Estrategia por eliminación

1. Para cada estado q aplicamos la reducción con ER sobre los arcos. Se eliminan todos los estados excepto q y el estado inicial q_0.
2. Si $q \neq q_0$ obtenemos la ER: $(R + SU^*T)^*SU^*$

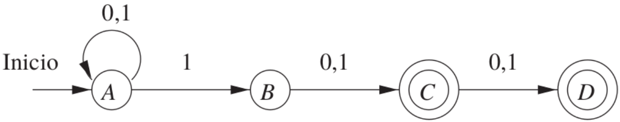


Estrategia por eliminación

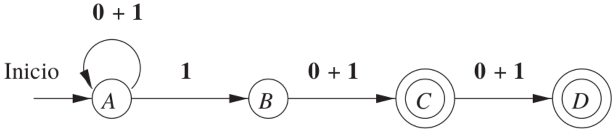
1. Para cada estado final q aplicamos la reducción con ER sobre los arcos. Se eliminan todos los estados excepto q y el estado inicial q_0.
2. Si estado inicial es igual al estado de aceptación
3. La ER es la suma (unión) de todas las expresiones obtenidas para los estados finales.



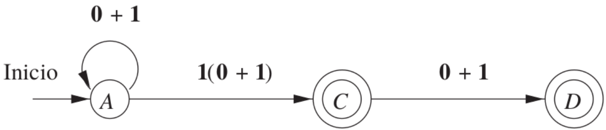
Obtenga la expresión regular por eliminación de estados.



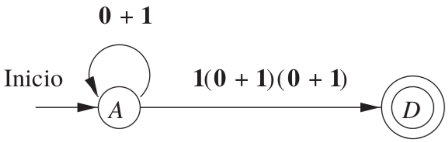
Obtenga la expresión regular por eliminación de estados.

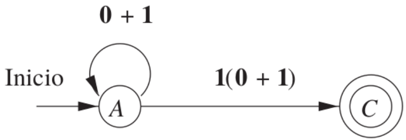


Obtenga la expresión regular por eliminación de estados.



Reducciones Separadas para C y D





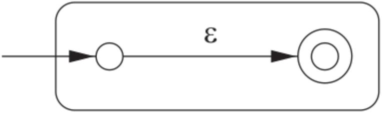
De las Expresiones Regulares a los AFN-ε

Todo lenguaje definido mediante una expresión regular también puede definirse mediante un autómata finito.

ER en AFN$_\varepsilon$

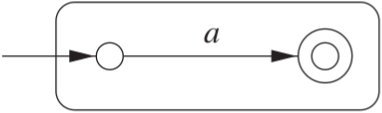
- Existe un estado de aceptación
- No hay arcos que entren al estado inicial
- No hay arcos que salgan del estado de aceptación

Caso Base



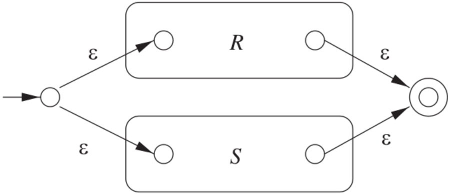
(a)

(b)



(c)

Paso Inductivo

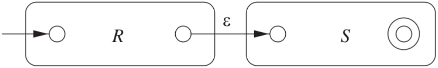


$R + S$

El lenguaje del automata es $L(R) \cup L(S)$.

- Del estado inicial llegamos a los estados iniciales de R o S
- Del estado de aceptación de R o S llegamos a los estados de aceptación del automata

Paso Inductivo

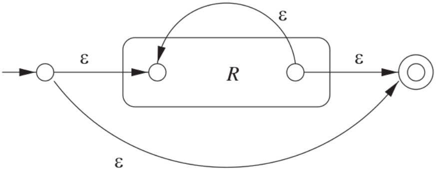


RS

Autómata para la concatenación. El lenguaje de aceptación es $L(R)L(S)$

- Estado inicial del automata R es el estado inicial del conjunto
- Estado final del automata S es el estado final del conjunto

Paso Inductivo



R^*

El lenguaje deaceptación es $L(R^*)$

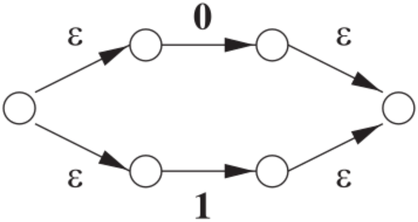
- Permite ir del estado inicial al estado de aceptación
- Desde el estado inicial a R, estando en R uno o más veces y luego al estado de aceptación

Ejemplo

Convertir la expresión: $(0+1)^{*}1(0+1)$

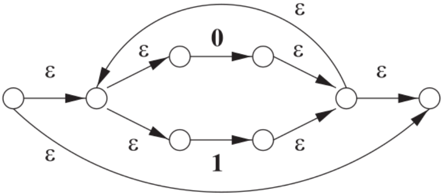
Ejemplo

Convertir la expresión: $(0 + 1)^{*}1(0 + 1)$



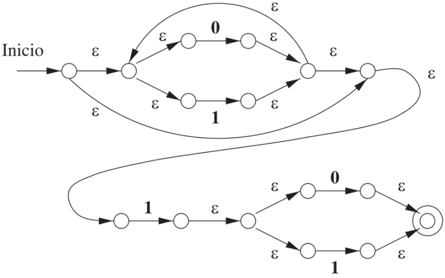
Ejemplo

Convertir la expresión: $(0+1)^*1(0+1)$



Ejemplo

Convertir la expresión: $(0 + 1)^*1(0 + 1)$



Ejercicios propuestos

Contruya el AFN_{ε} para:

- $(0+1)01$
- $00(0+1)^*$

Muchas gracias por su atención