Lenguajes formales y autómatas

Expresiones regulares

- Permiten expresar lenguajes regulares
- Se consideran un *metalenguaje*
- Si α es una expresión regular, entonces $\{\alpha\}$ es el conjunto descripto por α . Podemos decir que la expresión regular α denota el lenguaje de la cadena α .
- Para poder describir los lenguajes regulares definimos determinadas operaciones relacionadas a las operaciones de los lenguajes regulares.
- **Dos expresiones regulares son equivalentes si designan el mismo conjunto regular.**

Operaciones.

- Sean α y β dos expresiones regulares
- **Unión o alternativa** $\alpha|\beta$ o $[\alpha\beta]$

Puede aparecer α o β de manera indistinta. $\{\alpha|\beta\} = \{\alpha\} \bigcup \{\beta\}$

- **Concatenación** $\alpha\beta$

Es α seguido de β . $\{\alpha\beta\} = \{\alpha\}\{\beta\}$

- **Cierre** α^*

Viene **cero o más** α . $\alpha^* = \{\alpha\}^*$

- **Cierre positivo** α^+

Viene **una o más** α . $\alpha^+ = \{\alpha\}^+$

Precedencia de las operaciones

1. Paréntesis
2. Cierres
3. Concatenación
4. Alternativa

Indicar el lenguaje que denotan y algunas cadenas de dicho lenguaje para las siguientes ER:

1. aa^*bb^*
2. $1(01)^*$
3. $(0|1)^+$
4. $((a|b)(a|b))^*$
5. $[a..zA..Z\_](([a..zA..Z0..9\_])^*$

Autómatas

- Un **autómata reconocedor de un lenguaje** funciona de tal forma que cuando recibe en su entrada una cadena de símbolos indica si la misma pertenece o no al lenguaje.
- El **estado de un autómata** es toda la información necesaria en un momento dado para poder deducir, dado un símbolo de entrada en el momento actual, cual será el símbolo de salida.
- Un autómata tiene un determinado número de estados y se encontrará en uno u otro dependiendo de la historia de símbolos que han llegado.
- Al arribar un símbolo de entrada se produce una **transición** de un estado a otro.

Definición formal de autómata finito

Una máquina de estados finitos *M* es un quíntuplo (K,Σ,δ,s,F) donde:

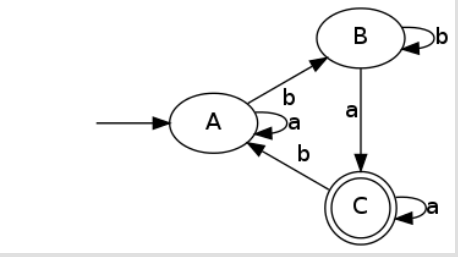
- *K* es un conjunto de identificadores de estados
- Σ es el alfabeto de entrada
- $s \in K$ es el estado inicial
- $F \subset K$ es un conjunto de estados finales
- $\delta : K \times \Sigma \to K$ es la función de transición, donde a partir de un estado y un símbolo indica el nuevo estado

Representación de autómatas

Tabla de transiciones

	a	b
A	A	B
B	C	B
C^*	C	A

Diagrama de Moore



Determinísticos vs No determinísticos

- Un autómata es determinístico cuando para cada par (estado, símbolo) hay solo **un** estado relacionado. δ es una función
- Un autómata no determinístico permite que por cada par (estado, símbolo) haya más de un estado relacionado
- Los autómatas finitos deterministas son más veloces como reconocedores (la velocidad es proporcional al tamaño de la entrada), pero la tabla que los representa es más grande.

Vamos a ver como pasar de ER => AFN => AFD => AFD de estados mínimos

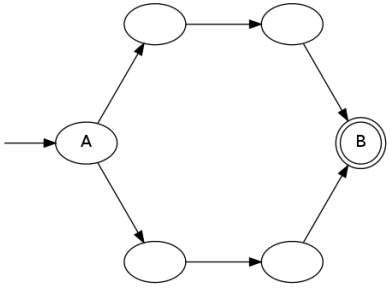
ER => AFN. Construcción de Thompson

- Se dan autómatas básicos para algunas ERs y luego se combinan para formar ERs más complejas
- Todos los AFN tienen un único estado inicial y un único estado final.

1. λ

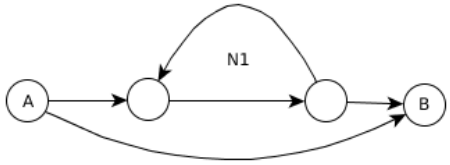
2. a

3. $N_1 | N_2$



4. $N_1 \ N_2$

5. N_1^*



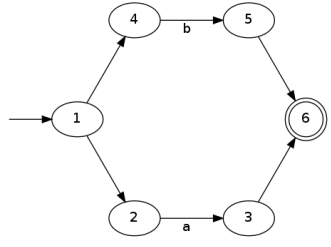
Ejercicio.

- Genere un AFN para la ER $(<letra>|\_)[<letra><digito>\_]^*$

AFN => AFD.

- $Clausura_{\lambda}(s)$: es el conjunto de todos los estados alcanzables desde **s** por medio de λ

Ejemplo



$Clausura_{\lambda}(1) = \{1,2,4\}$
$Clausura_{\lambda}(2) = \{2\}$
$Clausura_{\lambda}(3) = \{3,6\}$

- **Estado propio**: vamos a denominar así a los estados que poseen una transición de salida distinta de lambda o son estados de aceptación. En el ejemplo anterior 2 y 4 serían estados propios.

Algoritmo

```
a_procesar = [Estado(propios(clausura(s0)))]
procesados = []

while a_procesar:
    estado_actual = a_procesar.pop()
    procesados.append(estado_actual)

    for simbolo in estado_actual.simbolos_salida():
        proximo_estado = Estado(propios(clausura(ir(estado_actual, simbolo))))
        if proximo_estado not in (procesados + a_procesar):
            a_procesar.append(proximo_estado)
        hacer_transicion(estado_actual, simbolo, proximo_estado)
```

AFd => AFDEM.

- Dos estados son **equivalentes** si al unirse en un solo estado pueden reconocer el mismo lenguaje regular que si estuvieran separados.
- Dos estados **no equivalentes** son **estados distinguibles.**
- Un estado es **alcanzable** si es posible llegar a él desde el estado inicial.
- Un AFD está minimizado si todos sus estados son **distinguibles** y **alcanzables**

Algoritmo

1. Crear una partición en 2 grupos: estados no finales y estados finales
2. Particionar cada grupo de la partición de manera tal que todas las transiciones de cada uno de los símbolos apunten a estados en la misma partición.
3. Repetir 2 hasta que la partición no varíe.

4. Elegir 1 estado de cada grupo y eliminar los restantes, reemplazando en la tabla aquellos que son eliminados por el elegido del grupo.

Ejemplo:

AFD

	a	b
A	B	C
B	B	D
C	B	C
D	B	E
E*	B	C

AFDEM

	a	b
A	B	A
B	B	D
D	B	E
E*	B	A

1. {A, B, C, D} {E}

2.1. {A, B, C} {D} {E} Separamos D porque con **a** apunta a {A, B, C, D} y con **b** a {E}

2.2. {A, C} {B} {D} {E} Separamos B porque con **a** apunta a {A, B, C} y con **b** a {D}

3. {A, C} no se necesita separar ya que con **a** apuntan a {B} y con **b** apuntan a {A, C}

4. Elegimos A como representante de {A, C} y renombramos todas las C

- **Otro método, aka la escalerita**

B				
C				
D				
E*				
	A	B	C	D

Bibliografía y enlaces útiles.

- Alfonseca Cubero y otros - Teoría de autómatas y lenguajes formales - McGRAW-HILL
- Aho Alfred y Ullman Jeffrey - Compiladores, principios, técnicas y herramientas - PEARSON EDUCACION
- Juan Manuel Cueva Lovelle, Lenguajes, Gramáticas y Autómatas, 2001, http://di002.edv.uniovi.es/~cueva/publicaciones/AUTOMATA.pdf
- http://es.wikipedia.org/wiki/Aut%C3%B3mata_finito