Tema 2: Lenguajes y Expresiones Regulares

Autómatas y Lenguajes Formales

Dpto. de Ingeniería de la Información y las Comunicaciones

UNIVERSIDAD DE MURCIA

Sintaxis y semántica de las expresiones regulares

- Las **expresiones regulares** son un **formalismo algebraico** para **describir lenguajes de manera finita**.
- Los lenguajes que pueden describirse mediante expresiones regulares se llaman **lenguajes regulares**

  Ej.- El lenguaje $\{0^i 1 0^j \mid i, j \geq 0\}$ puede describirse de forma abreviada mediante la expresión regular: 0*10*

Sintaxis y semántica de las expresiones regulares

- Las **expresiones regulares** son un **formalismo algebraico** para **describir lenguajes de manera finita**.
- Los lenguajes que pueden describirse mediante expresiones regulares se llaman **lenguajes regulares**

  Ej.- El lenguaje $\{0^i10^j \mid i, j \geq 0\}$ puede describirse de forma abreviada mediante la expresión regular: 0*10*

- **La sintaxis de las expresiones regulares** establece la manera de escribir correctamente una ER
- **La semántica de las expresiones regulares** establece el **significado** o **valor semántico** de una ER R, que es el lenguaje $L(R)$ que describe la expresión regular R.

Sintaxis y semántica de ER (casos básicos)

Sea un alfabeto $V = \{a_1, \ldots, a_k\}$. Se definen las expresiones regulares básicas con alfabeto V y el lenguaje descrito por ellas como:

1. SINTAXIS: la **constante** $\varnothing$ es una ER.
   SEMÁNTICA: el lenguaje descrito es $L(\varnothing) = \varnothing$
2. SINTAXIS: la **constante** λ es una ER.
   SEMÁNTICA: el lenguaje descrito es $L(\lambda) = \{\lambda\}$
3. SINTAXIS: si $a_i \in V$ entonces la **constante** a_i es una ER.
   SEMÁNTICA: el lenguaje descrito es $L(a_i) = \{a_i\}$

Ejemplo

Siendo $V = \{a, b\}$ se tiene que **a** y **b** son ER y describen los lenguajes $L(a) = \{a\}$ y $L(b) = \{b\}$, respectivamente.

Sintaxis y semántica de ER (casos con operadores I)

Los operadores de ER son la **concatenación** ($\circ$), **unión o alternancia** $|$, **clausura** $*$ y **paréntesis de agrupación** ( )

4. SINTAXIS (**regla de concatenación**): si R_1 y R_2 son ER entonces $R_1 \circ R_2 = R_1 R_2$ es una ER.

   SEMÁNTICA: el lenguaje descrito es $L(R_1 \circ R_2) = L(R_1) \circ L(R_2)$

   Ej.- de que **ab** y **b** son ER se deduce que *abb* es una ER y $L(abb) = L(ab \circ b) = L(ab) \circ L(b) = \{ab\} \circ \{b\} = \{abb\} = \{abb\}$

Sintaxis y semántica de ER (casos con operadores I)

Los operadores de ER son la **concatenación** ($\circ$), **unión o alternancia** $|$, **clausura** $*$ y **paréntesis de agrupación** ( )

4. SINTAXIS (**regla de concatenación**): si R_1 y R_2 son ER entonces $R_1 \circ R_2 = R_1 R_2$ es una ER.

   SEMÁNTICA: el lenguaje descrito es $L(R_1 \circ R_2) = L(R_1) \circ L(R_2)$

   Ej.- de que **ab** y **b** son ER se deduce que *abb* es una ER y $L(abb) = L(ab \circ b) = L(ab) \circ L(b) = \{ab\} \circ \{b\} = \{abb\} = \{abb\}$

5. SINTAXIS (**regla de unión**): si R_1 y R_2 son ER entonces $R_1 | R_2$ es una ER.

   SEMÁNTICA: el lenguaje descrito es $L(R_1 | R_2) = L(R_1) \cup L(R_2)$

   Ej.- de que *abb* y *ab* son ER se deduce que $abb|ab$ es una ER y $L(abb|ab) = L(abb) \cup L(ab) = \{abb\} \cup \{ab\} = \{abb, ab\}$

Sintaxis y semántica de ER (casos con operadores II)

6. SINTAXIS (**regla de clausura**): si R es una ER entonces R^* es una ER.
   SEMÁNTICA: el lenguaje descrito es $L(R^*) = (L(R))^*$
   Ej.- de que **a** es una ER se deduce que a^* es una ER y $L(a^*) = (L(a))^* = \{a\}^* = \{a^i \mid i \geq 0\}$.

Sintaxis y semántica de ER (casos con operadores II)

6. SINTAXIS (**regla de clausura**): si R es una ER entonces R^* es una ER.
   SEMÁNTICA: el lenguaje descrito es $L(R^*) = (L(R))^*$
   Ej.- de que **a** es una ER se deduce que a^* es una ER y $L(a^*) = (L(a))^* = \{a\}^* = \{a^i \mid i \geq 0\}$.
7. SINTAXIS (**regla de agrupación**): si R es una ER entonces (R) es una ER
   SEMÁNTICA: el lenguaje descrito es $L((R)) = (L(R))$
   Ej.- de que **aab** es una ER se deduce que (abb) también es una ER. Al ser (abb) y aa ER entonces también lo es $(abb)^* \circ aa = (abb)^*aa$.

Sintaxis y semántica de ER (casos con operadores II)

6. SINTAXIS (**regla de clausura**): si R es una ER entonces R^* es una ER.
   SEMÁNTICA: el lenguaje descrito es $L(R^*) = (L(R))^*$
   Ej.- de que **a** es una ER se deduce que a^* es una ER y $L(a^*) = (L(a))^* = \{a\}^* = \{a^i \mid i \geq 0\}$.
7. SINTAXIS (**regla de agrupación**): si R es una ER entonces (R) es una ER
   SEMÁNTICA: el lenguaje descrito es $L((R)) = (L(R))$
   Ej.- de que **aab** es una ER se deduce que (abb) también es una ER. Al ser (abb) y aa ER entonces también lo es $(abb)^* \circ aa = (abb)^*aa$. Se tiene que $L((abb)^* \circ aa) = L((abb)^*) \circ L(aa) = (L(abb))^* \circ L(aa) = \{abb\}^* \circ \{aa\} = \{(abb)^i aa \mid i \geq 0\}$

Evitando ambigüedad: precedencia de operadores

1. La **precedencia de operadores** de mayor a menor es: $(\,)$ $\quad *$ $\quad \circ$ $\quad |$
   Convenio análogo al de las expresiones aritméticas con paréntesis, potencia, producto y suma.
2. La **regla de asociatividad por la izquierda** establece que $R_1 \circ R_2 \circ R_3$ equivale a la ER con paréntesis $(R_1 \circ R_2) \circ R_3$.
   Y $R_1|R_2|R_3$ equivale a $(R_1|R_2)|R_3$.

Ejemplo de ER con paréntesis y sin paréntesis

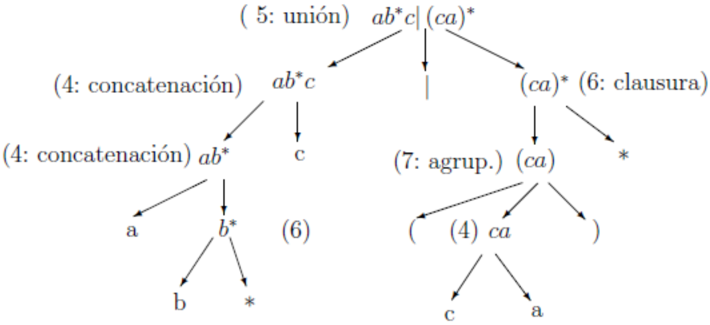
La expresión regular $ab^*c|(ca)^*$ **equivale a** la expresión con paréntesis

$$((a(b^*))c)|(ca)^*$$

Pero $(ca)^*$ **no es equivalente** a ca^*

Análisis de expresiones regulares (sintaxis)

Ej.- La ER $ab^*c|(ca)^*$ es sintacticamente correcta porque puede construirse un árbol de análisis sintáctico siguiendo las reglas de sintaxis de ER:



Concatenando las hojas del árbol de izquierda a derecha se obtiene la ER

Análisis de expresiones regulares (semántica)

Ejemplo

¿Qué cadenas describe la ER $ab^*c|(ca)^*$?

$$L(ab^*c|(ca)^*) = \{a\} \circ \{b\}^* \circ \{c\} \cup (\{c\} \circ \{a\})^*$$

Análisis de expresiones regulares (semántica)

Ejemplo

¿Qué cadenas describe la ER $ab^*c|(ca)^*$?

$$L(ab^*c|(ca)^*) = \{a\} \circ \{b\}^* \circ \{c\} \cup (\{c\} \circ \{a\})^*$$

Resolviendo las operaciones con lenguajes tenemos:

$$L(ab^*c|(ca)^*) = \{ab^ic \vee (ca)^j \mid i, j \geq 0\}$$

Análisis de expresiones regulares (semántica)

Ejemplo

¿Qué cadenas describe la ER $ab^*c|(ca)^*$?

$$L(ab^*c|(ca)^*) = \{a\} \circ \{b\}^* \circ \{c\} \cup (\{c\} \circ \{a\})^*$$

Resolviendo las operaciones con lenguajes tenemos:

$$L(ab^*c|(ca)^*) = \{ab^ic \vee (ca)^j \mid i,j \geq 0\}$$

Emparejamiento entre expresiones regulares y cadenas

Ej.- La expresión regular $ab^*c|c^*$ **casa con** la cadena *ac* o que **tiene coincidencia con** la cadena *ac*.
En inglés se dice que $ab^*c|c^*$ *matches ac*.
También se dice que la cadena *ac* **se ajusta al patrón** $ab^*c|c^*$.
$\rightrightarrows$ Formalmente quiere decir que $ac \in L(ab^*c|c^*)$
La ER $ab^*c|c^*$ también **tiene coincidencia con la cadena vacía**.

Construcción de expresiones regulares

¿Es R correcta para describir cierto lenguaje L_r? Hay que comprobar:

1. **Que R no sea demasiado estricta**. R debe tener coincidencia con todas las cadenas consideradas válidas (pertenecen a L_r).
2. **Que R no sea demasiado general**. R no debe tener coincidencia con cadenas consideradas incorrectas (no pertenecen a L_r).

Construcción de expresiones regulares

¿Es R correcta para describir cierto lenguaje L_r? Hay que comprobar:

1. **Que R no sea demasiado estricta**. R debe tener coincidencia con todas las cadenas consideradas válidas (pertenecen a L_r).
2. **Que R no sea demasiado general**. R no debe tener coincidencia con cadenas consideradas incorrectas (no pertenecen a L_r).

Ejemplo

⇉Obtener ER para describir los números decimales con parte entera y fraccionaria obligatoria.

- ¿Es correcta $(0|1|2|3|4|5|6|7|8|9)^*.(0|1|2|3|4|5|6|7|8|9)^*$?

Construcción de expresiones regulares

¿Es R correcta para describir cierto lenguaje L_r? Hay que comprobar:

1. **Que R no sea demasiado estricta**. R debe tener coincidencia con todas las cadenas consideradas válidas (pertenecen a L_r).
2. **Que R no sea demasiado general**. R no debe tener coincidencia con cadenas consideradas incorrectas (no pertenecen a L_r).

Ejemplo

⇉Obtener ER para describir los números decimales con parte entera y fraccionaria obligatoria.

- ¿Es correcta $(0|1|2|3|4|5|6|7|8|9)^*.(0|1|2|3|4|5|6|7|8|9)^*$? **NO,** el patrón de la ER es **demasiado general** porque tiene coincidencia con cadenas no válidas como "123." o ".7" o incluso con "."
- Una ER correcta sería:

Construcción de expresiones regulares

¿Es R correcta para describir cierto lenguaje L_r? Hay que comprobar:

1. **Que R no sea demasiado estricta**. R debe tener coincidencia con todas las cadenas consideradas válidas (pertenecen a L_r).
2. **Que R no sea demasiado general**. R no debe tener coincidencia con cadenas consideradas incorrectas (no pertenecen a L_r).

Ejemplo

⇉Obtener ER para describir los números decimales con parte entera y fraccionaria obligatoria.

- ¿Es correcta $(0|1|2|3|4|5|6|7|8|9)^*.(0|1|2|3|4|5|6|7|8|9)^*$? **NO,** el patrón de la ER es **demasiado general** porque tiene coincidencia con cadenas no válidas como "123." o ".7" o incluso con "."
- Una ER correcta sería:

$$(0|1|2|3|4|5|6|7|8|9)(0|1|\ldots|9)^*.(0|1|2|3|4|5|6|7|8|9)(0|1|\ldots|9)^*$$

De la sintaxis teórica a sintaxis extendida

Ejemplo

Una ER larga como

$$(0|1|2|3|4|5|6|7|8|9)(0|1|\ldots|9)^{*}.(0|1|2|3|4|5|6|7|8|9)(0|1|\ldots|9)^{*}$$

se puede expresar con **sintaxis extendida** de forma **más breve**:

[0-9]+\.[0-9]+

De la sintaxis teórica a sintaxis extendida

Ejemplo

Una ER larga como

$$(0|1|2|3|4|5|6|7|8|9)(0|1|\ldots|9)^*.(0|1|2|3|4|5|6|7|8|9)(0|1|\ldots|9)^*$$

se puede expresar con **sintaxis extendida** de forma **más breve**:

[0-9]+\.[0-9]+

Ejemplo

La ER $(0|1|2|3|4|5|6|7|8|9)^*(0|2|4|6|8)$ se expresa en sintaxis extendida como: [0-9]*[02468]

⇒Describe las cadenas que representan números naturales pares.
Formalmente describe el **lenguaje regular**:

$$N_{par} = \{xp \mid x \in V_{dig}^* \wedge p \in \{0, 2, 4, 6, 8\}\}$$

Construyendo expresiones regulares por partes

Ejemplo

Queremos obtener una ER para describir al lenguaje B_{alt} que contiene todas las cadenas con alfabeto $\{0, 1\}$ que **tienen los unos y ceros alternados.** Se entiende que $\lambda, 0, 1 \in B_{alt}$.

- Consideramos los **distintos casos de cadenas válidas** (sublenguajes de B_{alt}), **obtenemos una ER para cada caso** y **las unimos** con el operador $|$

Construyendo expresiones regulares por partes

Ejemplo

Queremos obtener una ER para describir al lenguaje B_{alt} que contiene todas las cadenas con alfabeto $\{0, 1\}$ que **tienen los unos y ceros alternados.** Se entiende que $\lambda, 0, 1 \in B_{alt}$.

- Consideramos los **distintos casos de cadenas válidas** (sublenguajes de B_{alt}), **obtenemos una ER para cada caso** y **las unimos** con el operador $|$
  1. Cadenas tipo $(10)^n$ con $n \geq 0 \Rightarrow$ se describen con la ER $\mathbf{(10)}^*$

Construyendo expresiones regulares por partes

Ejemplo

Queremos obtener una ER para describir al lenguaje B_{alt} que contiene todas las cadenas con alfabeto $\{0, 1\}$ que **tienen los unos y ceros alternados.** Se entiende que $\lambda, 0, 1 \in B_{alt}$.

- Consideramos los **distintos casos de cadenas válidas** (sublenguajes de B_{alt}), **obtenemos una ER para cada caso** y **las unimos** con el operador $|$
  1. Cadenas tipo $(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con la ER $(\mathbf{10})^*$
  2. Cadenas del tipo $(01)^n$ con $n \geq 0 \rightrightarrows$ se describen con $(\mathbf{01})^*$

Construyendo expresiones regulares por partes

Ejemplo

Queremos obtener una ER para describir al lenguaje B_{alt} que contiene todas las cadenas con alfabeto $\{0, 1\}$ que **tienen los unos y ceros alternados.** Se entiende que $\lambda, 0, 1 \in B_{alt}$.

- Consideramos los **distintos casos de cadenas válidas** (sublenguajes de B_{alt}), **obtenemos una ER para cada caso** y **las unimos** con el operador $|$
  1. Cadenas tipo $(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con la ER $\mathbf{(10)}^*$
  2. Cadenas del tipo $(01)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{(01)}^*$
  3. Cadenas tipo $0(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{0(10)}^*$

Construyendo expresiones regulares por partes

Ejemplo

Queremos obtener una ER para describir al lenguaje B_{alt} que contiene todas las cadenas con alfabeto $\{0, 1\}$ que **tienen los unos y ceros alternados.** Se entiende que $\lambda, 0, 1 \in B_{alt}$.

- Consideramos los **distintos casos de cadenas válidas** (sublenguajes de B_{alt}), **obtenemos una ER para cada caso** y **las unimos** con el operador $|$
  1. Cadenas tipo $(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con la ER $(\mathbf{10})^*$
  2. Cadenas del tipo $(01)^n$ con $n \geq 0 \rightrightarrows$ se describen con $(\mathbf{01})^*$
  3. Cadenas tipo $0(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{0}(\mathbf{10})^*$
  4. Cadenas del tipo $1(01)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{1}(\mathbf{01})^*$

Construyendo expresiones regulares por partes

Ejemplo

Queremos obtener una ER para describir al lenguaje B_{alt} que contiene todas las cadenas con alfabeto $\{0,1\}$ que **tienen los unos y ceros alternados.** Se entiende que $\lambda, 0, 1 \in B_{alt}$.

- Consideramos los **distintos casos de cadenas válidas** (sublenguajes de B_{alt}), **obtenemos una ER para cada caso** y **las unimos** con el operador $|$
  1. Cadenas tipo $(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con la ER $\mathbf{(10)}^*$
  2. Cadenas del tipo $(01)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{(01)}^*$
  3. Cadenas tipo $0(10)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{0(10)}^*$
  4. Cadenas del tipo $1(01)^n$ con $n \geq 0 \rightrightarrows$ se describen con $\mathbf{1(01)}^*$
- El lenguaje B_{alt} se describe con $(01)^*|(10)^*|0(10)^*|1(01)^*$
  Esta ER **es correcta para describir el lenguaje** B_{alt}

Propiedades de las expresiones regulares (I)

Decimos que dos expresiones regulares R_1 y R_2 son **equivalentes** ($\mathbf{R_1 = R_2}$) si y sólo si $L(R_1) = L(R_2)$

1 : $R_1|(R_2|R_3) = (R_1|R_2)|R_3$ [asociativa–unión]
2 : $R_1|R_2 = R_2|R_1$ [conmutativa–unión]
3 : $R_1 \circ \lambda = \lambda \circ R_1 = R_1$ [identidad]
4 : $R_1 \circ \varnothing = \varnothing \circ R_1 = \varnothing$ [anulación]
5 : $R_1 \circ (R_2 \circ R_3) = (R_1 \circ R_2) \circ R_3$ [asociativa–concatenación]
6 : $R_1 \circ (R_2|R_3) = R_1 \circ R_2|R_1 \circ R_3$ [distributiva derecha]
7 : $(R_2|R_3) \circ R_1 = R_2 \circ R_1|R_3 \circ R_1$ [distributiva izquierda]
8 : $L(R_1) \subseteq L(R_2) \Rightarrow R_1|R_2 = R_2$ [regla de eliminación]

Ej.- Simplificar la expresión regular $(0^*1)^*0|10$.

Propiedades de las expresiones regulares (II)

$$
\begin{aligned}
9 &: \ \lambda^* = \lambda \\
10 &: \ \varnothing^* = \lambda \\
11 &: \ R_1 \circ R_1^* = R_1^* \circ R_1 \\
12 &: \ R_1^* = (R_1^*)^* \\
13 &: \ R_1^* = \lambda | R_1 \circ R_1^* \\
14 &: \ (R_1|R_2)^* = (R_1^* \circ R_2^*)^* \\
15 &: \ (R_1|R_2)^* = (R_1^* \circ R_2)^* \circ R_1^* \\
16 &: \ R_1 \circ (R_2 \circ R_1)^* = (R_1 \circ R_2)^* \circ R_1
\end{aligned}
$$

Ej.- Problema resuelto 1: simplificar la ER $a|a\,(b|aa)\,(b^*aa)^*\,b^*|a\,(aa|b)^*$ de manera que sólo tenga un operador de clausura.

Aplicaciones de las expresiones regulares

- Las expresiones regulares se aplican a distintos niveles en el de desarrollo de software orientado a resolver problemas de **procesamiento de cadenas de patrón regular**: búsqueda, conversión o validación de formato, etc.

Aplicaciones de las expresiones regulares

- Las expresiones regulares se aplican a distintos niveles en el de desarrollo de software orientado a resolver problemas de **procesamiento de cadenas de patrón regular**: búsqueda, conversión o validación de formato, etc.
- Una de las aplicaciones más importantes es en la fase de **análisis léxico** de los compiladores, intérpretes y en general de los traductores de **código fuente** en un lenguaje a **código objeto** en otro lenguaje.
  Un esquema de caja negra del analizador léxico es:

código fuente $\longrightarrow$ $\boxed{\text{ANALIZADOR LÉXICO}}$ $\longrightarrow$ secuencia de tokens

Ejemplo: código fuente visto como secuencia de tokens

```
main ()
{
int varx, varz;

/* Asigna 2 a varz
y esto es un comentario en varias líneas */
varx = 2;
varz = varx;
}
```

Secuencia de tokens:

$\boxed{main}\ \boxed{(}\ \boxed{)}$

$\boxed{\{}$

$\boxed{int}\ \boxed{varx}\ \boxed{,}\ \boxed{varz}\ \boxed{;}$

$\boxed{varx}\ \boxed{=}\ \boxed{2}\ \boxed{;}$

$\boxed{varz}\ \boxed{=}\ \boxed{varx}\ \boxed{;}$

$\boxed{\}}$

Preguntas de evaluación en apuntes

- Contiene problemas de **razonamiento**, **aplicación o cálculo** y **preguntas tipo test**.
- Son una colección de preguntas de examen, aunque no incluye todo tipo de preguntas posibles de examen.

- **Problemas resueltos:** actividad para auto-evaluación.
- **Problemas propuestos:** actividad para resolver en parte en clase (teoría o prácticas).
- **Preguntas tipo test:** actividad para resolver en parte en clase (teoría o prácticas).