Sesión 4

Ejemplos y aplicaciones de las expresiones regulares

Igualdad de expresiones regulares

- $(0^*1^*)^* = (0 + 1)^*$
- Lado derecho: $(0 + 1)^*$ = todo
- Lado izquierdo:
  - $(0^*1^*)^* = \underline{0^*1^*}\,\underline{0^*1^*}\,\underline{0^*1^*}\ldots\,\underline{0^*1^*}$
  - En cada segmento tomamos el 0 o el 1

Igualdad de expresiones regulares

- $(0 + 1)^*01(0 + 1)^* + 1^*0^* = (0 + 1)^*$
- Lado derecho: $(0 + 1)^*$ = todo
- Lado izquierdo:
  - Todas las cadenas con 01: $(0 + 1)^*01(0 + 1)^*$
  - Todas las cadenas SIN 01: ?
    - ✓Λ
    - ✓0000…0
    - ✘0000 …01
    - ✓1111 …100000…0
    - ✓1111…1
  - Todas las cadenas SIN 01: 1^*0^*

Cadenas con un número non de 1's

- Enfocandonos en el primer 1:
  - Empezamos con 1: 0^*10^*
  - Subcadenas con un par de 1's y cualquier número de 0's: $(10^*10^*)^*$
  - Su concatenación: $0^*10^*(10^*10^*)^*$
- Enfocandonos en el primer 1, pero también en la segunda subcadena:
  - $0^*1\,(0^*\,10^*10^*)^*\,0^*$

Cadenas con un número non de 1's

- Enfocandonos en el 1 final:
  - $(0^*10^*1)^*\,0^*10^*$
- Enfocandonos en el 1 de en medio:
  - $0^*(10^*10^*)^*\,1(0^*10^*1)^*0^*$
- Pero no:
  - $(10^*10^*)^*\,10^*$
  - Necesitamos permitir los 0's iniciales, entonces: $0^*(10^*10^*)^*\,10^*$

Cadenas de longitud 6 o menos

- De manera concreta:
  - $\Lambda + 0 + 1 + 00 + 01 + 10 + 11 + \ldots + 111110 + 111111$
- Tratando algo mejor:
  - Cadenas de longitud 6 exactamente:
    - $(0 + 1)(0 + 1)(0 + 1)(0 + 1)(0 + 1)(0 + 1)$
  - Notación exponencial:
    - $(0 + 1)^6$
  - Permitiendo cadenas de longitud 6 o menos:
    - $(0 + 1 + \Lambda)^6$

Cadenas que terminan en 1 y sin "00"

- $L = \{x \in \{0, 1\}^* \mid x$ termina en 1 y no tiene 00$\}$
- 0 **NO** puede seguir a 0: el final es 01 o una secuencia de 1's
- Por lo tanto, *x* está formada por bloques de 1's o copias de 01: $\mathbf{(1 + 01)^*}$
  - {1, 01, 11, 101, 011, 0101…}

Cadenas que terminan en 1 y sin "00"

- Pero, esto incluye a Λ, (que no termina en 1 y no tiene 00)
  - Corrigiendo: $\mathbf{(1 + 01)^*1}$
- Pero ahora, 01 no está en el lenguaje:
  - Corrigiendo: $\mathbf{(1 + 01)^*(1 + 01)}$
  - Alternativamente: $\mathbf{(1 + 01)^+}$

El lenguaje de los identificadores de C

- Sean *l* & *d* abreviaturas de letras y dígitos respectivamente:
  - ***l*** abrevia a **a + b + …+ z + A + B + … + Z**
  - ***d*** abrevia a **0 + 1 + 2 + …+ 9**
- Un identificador de C es una cadena de longitud 1 o más que contiene letras, dígitos y el guión bajo ("_"):

  $\mathbf{(\mathit{l} + \_)(\mathit{l} + \mathit{d} + \_)^*}$
- Ejemplos:
  - "cis625", "cis_625", "cis6_2_5", "_625"

Literales reales en Pascal

- Notación:
  - *l* abrevia a **a + b + …+ z + A + B + … + Z**
  - *d* abrevia **0 + 1 + 2 + …+ 9**
  - *s* abrevia a "signo" (**Λ + a + m,** donde a es "+" & m es "–")
  - *p* es "."
  - *E* es un símbolo de Σ
- Literales reales: $sd^+(pd^+ + pd^+Esd^+ + Esd^+)$
- Ejemplos: +6.25, 6.25, -6.25*E*+2, 6.25*E*-2, -2*E*2

Aplicaciones de Expresiones Regulares

- Expresan un patrón que se quiere reconocer
- Pueden compilarse como un autómata determinístico, que puede utilizarse para reconocer patrones en textos
- Dos aplicaciones:
  - Analizadores léxicos
  - Búsqueda de textos en Internet

Notación de *ER* en UNIX

- Σ = El conjunto de ceracteres ASCII
- El comando *grep*:
  - *Global (search for) Regular Expressions and Print*
- Abreviaturas: Clases de caracteres
  - El punto "." representa a cualquier caracter
  - $[a_1a_2\ldots a_k]$ representa la *ER*: $a_1 + a_2 + \ldots + a_k$
    - e.g. Los caracteres para comparación en C: [<>=!]

Notación de *ER* en UNIX

- [*x*-*y*] representa a definiciones por rangos:
  - e.g. [A-Za-z0-9] se interpreta como el conjunto de todos los caracteres y dígitos
  - El signo menos "–" se pone al inicio o al final para evitar la ambigüedad posible:

    [–+.0-9] es el conjunto {–, +, . , 0…9}
- Para caracteres reservados de UNIX se usa el *backslash* \

  [0-9\.] es el conjunto de dígitos y el punto (i.e. no cualquier carácter)

Notación de *ER* en UNIX

- Significado de los operadores de UNIX:
  - | se utiliza para representar a +
  - El operador ? significa 0 o una vez:

    R? = $\Lambda + R$
  - + significa una o más veces: **R+** = RR^*
  - El operador {*n*} significa *n* copias de:

    R{5} = *RRRRR*
  - en UNIX * tiene su signifiado habitual!

Notación de *ER* en UNIX

- La precedencia de operadores es la usual (con ?, + & {*n*} con la misma precedencia de *)
- Las extensiones de UNIX para nombrar y referirse a cadenas previas (permitiendo el reconocimiento de lenguajes no-regulares) no se consideran aquí)

Analizadores léxicos

- Analizador léxico: la parte del compilador que inspecciona el código fuente e identifica *tokens* (i.e. básicos or símbolos atómicos, o entradas en la tabla de símbolos)
  - Keywords
  - Identificadores (constantes, variables, etc.)
- Generador de analizadores léxicos

Analizadores léxicos

- El comando de UNIX *lex* (*flex* en GNU)
  - Acepta una lista de expresiones regulares seguidas de una pieza de código entre brakets, que indica que hay que hacer cuando se identifica una instancia del token descrito por la *ER*
- Ventajas:
  - Descripción de alto nivel de analizadores léxicos
  - Generación automática de código complejo
  - Fácil de crear y de modificar

Ejemplo

- Entrada parcial del comando *lex*:

Else	{return(ELSE)}
[A-Za-z][A-Za-z0-9]*	{codigo para incluir un idenficador en la tabla de símbolos; return(Id); }
>=	{return(GE);}
=	{return(EQ);}

  …enteros, punto flotante, cadenas de caracteres, etc.
- Conversión de *ER* a una autómata finito para el procesamiento de las cadenas correspondientes

Encontrar patrones en textos

- Las *ER* son útiles para describir búsquedas de patrones interesantes
- Descripción de clases de patrones de textos definidas de manera vaga
- Fácil de especificar y modificar

Ejemplo: Detectar direcciones en páginas de web

- Descriptor del tipo de calle (Notación de UNIX):
  - Street|St\.|Ave\.|Road|Rd\.
- Nombre de la calle propiamente:
  - [A-Z][a-z]* (e.g. Island)
- Hay calles cuyo nombre tiene más de una palabra
  - '[A-Z][a-z]*([A-Z][a-z]*)*' (e.g. Road Island Av.)

Ejemplo: Detectar direcciones en páginas de web

- Números de las casas:
  - Cadena de dígitos… seguida probablemente de letras (e.g. "123A Main St.")
  - [0-9]+[A-Z]?
- La expresión compuesta:
  - '[0-9]+[A-Z]? [A-Z][a-z]*([A-Z][a-]*)* (Street|St\.|Ave\.|Road|Rd\.)

Ejemplo: Detectar direcciones en páginas de web

- Pero que tal:
  - Tipos inusuales de calle: "Boulevar", "Place",…
  - Calles con abreviaturas ordinales: 42nd St.
  - Apartados postals o direcciones rurales
  - Calles sin la palabra "Street" al final, como *El Camino Real* en Silicon Valley!
    - *El Camino Real Road*?
    - *2000 El Camino Real*

Ejemplo: Detectar direcciones en páginas de web

- ¡Realmente una tarea de ingeniería del lenguaje!
- El poder de las expresiones regulares:
  - Expresivas
  - Económicas