Tema 4. Algoritmos recursivos.pdf

raulcb98

Fundamentos de análisis de algoritmos

1º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería
Universidad de Huelva

KEEP CALM AND ESTUDIA UN POQUITO

WUOLAH + #QuédateEnCasa

INGENIERÍA INFORMÁTICA

FUNDAMENTOS DE ANÁLISIS DE ALGORITMOS.
TEMA 4

ALGORITMOS RECURSIVOS

RAÚL CASTILLA BRAVO
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ETSI – HUELVA)
Curso: 1º

ÍNDICE

Tema 4. Algoritmos recursivos

1. CONCEPTO

Un **algoritmo recursivo** es aquel que se llama a sí mismo para realizar su función. Para saber si un algoritmo recursivo está bien planteado debemos tener en cuenta dos aspectos:

-Caso base, debe existir una salida no recursiva.

-Invocar un caso más pequeño en cada repetición.

2. ANÁLISIS DE ALGORTIMOS ITERATIVOS

Análisis por bloques: Un bloque de instrucciones es un algoritmo del que conocemos su complejidad computacional.

Secuencia: Sea $\{I_1,I_2\}$ una secuencia de dos instrucciones con complejidades O(f) y O(g) respectivamente, entonces el orden de secuencia es O(máx(f,g)).

Selección condicional: Se evalúa una condición C. Si es verdadera, se ejecuta el bloque I1 de complejidad O(f) y en caso contrario, I2 de complejidad O(g). La complejidad del algoritmo será:

-En el peor caso, el bloque con mayor complejidad.

-En el mejor caso, el bloque con menor complejidad.

-En el caso promedio, si P es la probabilidad, la complejidad será: O(Pf) + O((1-P)g).

Iteración: la complejidad del bloque dependerá de la cantidad de veces que se repita: O(nf(n)). Si la complejidad depende de un índice, la complejidad de la iteración será:

$$\sum_{i=1}^{n} f(i)$$

3. ANÁLISIS DE ALGORITMOS RECURSIVOS

Para resolver un algoritmo recurrente necesitamos resolver un sistema recurrente con la elección de una operación básica. Se debe cumplir que: el **orden** del número de operaciones que hace el algoritmo, es igual al **orden** del número de veces que se realiza la operación básica.

Como un algoritmo recursivo tiene dos partes: el caso base y la parte recursiva, la función de complejidad tendrá dos ecuaciones. Supongamos el siguiente algoritmo:

```
funcion factorial(n)
    si n=0 entonces
        factorial ← 1
    sino
        factorial ← n*factorial(n-1)
    fin si
fin funcion
```

$$T(n) = \begin{cases} 2 & \text{si } n = 0 \\ 5 + T(n-1) & \text{si } n > 0 \end{cases}$$

Si cogemos como operación básica la multiplicación: 5 + T(n-1) → 1 + T(n-1).

Para resolver este sistema de ecuaciones tenemos varios métodos que veremos a continuación.

Raúl Castilla Bravo

Tema 4. Algoritmos recursivos

3.1 Expansión de recurrencias

El método consiste en ir sustituyendo las llamadas recurrentes por su definición para encontrar una regla general. Para ejemplificarlo vamos a utilizar: T(n) = T(n-1) + 1.

T(n) - T(n - 1) = 1
T(n -1) - T(n -2) = 1
T(n- 2) - T(n -3) = 1
...
T(2) - T(1) = 1
T(1) - T(0) = 1
T(0) = 0

sumamos todas las ecuaciones
$T(n) = \underbrace{1 + 1 + \ldots\ldots\ldots + 1}_{n \text{ veces}}$

T(n) = n

3.2 Método de la ecuación característica

Se utiliza principalmente en recurrencias lineales donde el valor de n de la recurrencia depende de los k anteriores. Pueden darse dos casos:

-Recurrencias homogéneas:

-Recurrencias no homogéneas: $\mathbf{a_0T(n) + a_1T(n-1) + \ldots + a_kT(n-k) = f(n)}$

En ambos casos, el objetivo es obtener una ecuación características cuyas ecuaciones permitan realizar una representación no recurrente de la función de complejidad.

Recurrencias homogéneas

Forma	$\mathbf{a_0T(n) + a_1T(n-1) + \ldots + a_kT(n-k) = 0}$
Cambio $T(n) = x^n$	$\mathbf{a_0}\, x^n + \mathbf{a_1}\, x^{n-1} + \ldots + \mathbf{a_k}\, x^{n-k} = 0.$
Dividimos por x^{n-k} y obtenemos la ecuación característica	$\mathbf{a_0\, x^k + a_1\, x^{k-1} + \ldots + a_k = 0}$

Definimos $r_1,\ldots,r_n$ como las raíces de la ecuación característica. Según su multiplicidad se pueden clasificar en:

a) *Multiplicidad 1*

Se deshace el cambio y r_i^n son las soluciones y la función de complejidad es:

$$T(n) = c_1 r_1^n + c_2 r_2^n + \ldots + c_k r_k^n = \sum_{i=1}^{k} c_i r_i^n$$

b) Multiplicidad distinta de 1

Es una generalización de la ecuación anterior. Supongamos la multiplicidad $m_1,\ldots, m_k$ y que la raíz r_1 tiene multiplicidad $m \geq 1$, la ec. característica y la función de complejidad son:

$(x-r_1)^m (x-r_2)\ldots (x-r_{k-m+1})$ $\qquad$ $T(n) = \sum_{i=1}^{k} \sum_{j=0}^{m_i-1} c_{ij} n^j r_i^n$

Donde los coeficientes C se obtienen de las condiciones iniciales.

Raúl Castilla Bravo

Recurrencias no homogéneas

Forma	$a_0T(n) + a_1T(n-1) + \ldots + a_kT(n-k) = f(n)$
Se conoce solución donde f(n) es un polinomio de grado d de la forma $b^n p(n)$.	$a_0T(n) + a_1T(n-1) + \ldots + a_kT(n-k) = b^n p(n)$
Ecuación característica	$(a_0 x^k + a_1 x^{k-1} + \ldots + a_k)(x - b)^{d+1} = 0$

En el caso de f(n) tenga la forma: $b_1^n p_1(n) + \cdots + b_s^n p_s(n)$, la ecuación característica es:

$$(a_0 x^k + a_1 x^{k-1} + \ldots + a_k)(x - b_1)^{d1+1})(x - b_2)^{d2+1} \ldots (x - b_s)^{ds+1} = 0$$