

Estos son algunos análisis de complejidad de algoritmos correspondientes a operaciones básicas en ABB, tomados del libro de Guerequeta y Vallecillo. Interesa que estén resueltos de diversas formas.

### **Procedimiento Inorden**

```

PROCEDURE Inorden(t:arbol);      (* recorrido en inorden de t *)
BEGIN
  IF NOT Esvacio(t) THEN          (* 1
*)
    Inorden(Izq(t));              (* 2 *)
    Opera(Raiz(t));               (* 3 *)
    Inorden(Der(t));              (* 4 *)
  END;                             (* 5 *)
END Inorden;

```

Para calcular el tiempo de ejecución, calcularemos primero el número de operaciones elementales (OE) que se realizan:

- En la línea (1) se ejecutan  $2+c$  OE: la llamada a Esvacio (1 OE), el tiempo de ejecución de este procedimiento ( $c$ ) y una negación.
- En la línea (2) se efectúa la llamada a Izq (1 OE), lo que tarda ésta en ejecutarse ( $c$  OE) más la llamada a Inorden (1 OE) y lo que tarde ésta en ejecutarse, que va a depender del número de elementos del árbol Izq(t).
- En la línea (3) se ejecutan  $2+c+d$  OE: dos llamadas a procedimientos y sus respectivos tiempos de ejecución.
- El número de OE de la línea (4) se calcula de forma análoga a la línea (2):  $2+c$  más lo que tarda Inorden en ejecutarse con el número de elementos que hay en Der(t).

Para estudiar el tiempo de ejecución, vamos a considerar dos casos extremos: que el árbol sea degenerado (es decir, una lista) y que sea equilibrado. Cualquier árbol se encuentra en una situación intermedia a estos dos casos.

- Si  $t$  es degenerado, podemos suponer sin pérdida de generalidad que Esvacio(Izq(t)) y que para todo a subárbol de  $t$  se verifica que Esvacio(Izq(a)). Por tanto, el número de OE que se realizan en la ejecución de Inorden(t) para un árbol  $t$  con  $n$  elementos es:

$$T(n) = (2+c) + (2+c+T(0)) + (2+c+d) + (2+c+T(n-1)) = 8+4c+d+T(0)+T(n-1). \\ T(0) = 2+c.$$

Con esto,  $T(n) = 10 + 5c + d + T(n-1)$ , ecuación en recurrencia no homogénea que podemos resolver desarrollándola telescópicamente:

$$T(n) = 10 + 5c + d + T(n-1) = (10 + 5c + d) + (10 + 5c + d) + T(n-2) = \dots = \\ (10 + 5c + d)n + (2+c) \in \Theta(n)$$

- Si  $t$  es equilibrado sus dos subárboles (izquierdo y derecho) tienen del orden de  $n/2$  elementos y son a su vez equilibrados. Por tanto, el número de OE que se realizan en la ejecución de Inorden(t) para un árbol  $t$  con  $n$  elementos es:

$$T(n) = (2+c) + (2+c+T(n/2)) + (2+c+d) + (2+c+T(n/2)) = 8+4c+d+2T(n/2). \\ T(0) = 2+c.$$

Para resolver esta ecuación en recurrencia se hace el cambio  $t_k = T(2k)$ , con lo que obtenemos  $t_k - 2t_{k-1} = 8 + 4c + d$ , ecuación no homogénea con ecuación característica  $(x-2)(x-1) = 0$ .

Por tanto,

$$T_k = c_1 2^k + c_2$$

y, deshaciendo los cambios,

$$T(n) = c_1 n + c_2.$$

Para calcular las constantes, nos apoyamos en la condición inicial  $T(0)=2+c$ , junto con el valor de  $T(1)$ , que puede ser calculado basándonos en la expresión de la ecuación en recurrencia:  $T(1) = 8 + 4c + d + 2(2 + c)$ , obteniendo

$$T(n) = (10 + 5c + d)n + (2+c) \in \Theta(n).$$

## **Función Altura**

```
PROCEDURE Altura(t:arbol):CARDINAL; (* altura de t *)
BEGIN
  IF Esvacio(t) THEN                                (* 1 *)
    RETURN 0                                         (* 2 *)
  ELSE                                              (* 3 *)
    RETURN 1+Max2(Altura(Izq(t)),Altura(Der(t)))    (* 4 *)
  END                                              (* 5 *)
END Altura;
```

Para determinar el tiempo de ejecución, calcularemos primero el número de operaciones elementales (OE) que se realizan:

– En la línea (1) se ejecutan  $1+c$  OE: la llamada a Esvacio (1 OE) más el tiempo de ejecución de este procedimiento ( $c$  OE).

– En la línea (2) se realiza 1 OE.

– En la línea (4) se efectúan:

a) la llamada a Izq (1 OE), lo que tarda ésta en ejecutarse ( $c$  OE) más la llamada a Altura (1 OE) y lo que tarde ésta en ejecutarse, que va a depender del número de elementos del árbol  $Izq(t)$ ; más

b) la llamada a Der (1 OE), lo que tarda ésta en ejecutarse ( $c$  OE) más la llamada a Altura (1 OE) y lo que tarde ésta en ejecutarse, que va a depender del número de elementos del árbol  $Der(t)$ ; más

c) el cálculo del máximo de ambos números (1 OE), un incremento (1 OE) y el RETURN (1 OE).

Para estudiar el tiempo de ejecución de esta función consideraremos los mismos casos que para la función Inorden: que el árbol sea degenerado (es decir, una lista) o que sea equilibrado.

• Si  $t$  es degenerado, podemos suponer sin pérdida de generalidad que  $Esvacio(Izq(t))$  y que para todo a subárbol de  $t$  se verifica que  $Esvacio(Izq(a))$ . Por tanto, el número de OE que se realizan en la ejecución de  $Altura(t)$  para un árbol  $t$  con  $n$  elementos es:

$$T(n) = (1+c) + (1+c+1+T(0) + 1+c+1+T(n-1)) + 3 = 8 + 3c + T(0) + T(n-1). \\ T(0) = (1+c) + 1 = 2+c.$$

Con esto,  $T(n)=10+4c+T(n-1)$ , ecuación en recurrencia no homogénea que podemos resolver desarrollándola telescópicamente:

$$T(n) = 10 + 4c + T(n-1) = (10 + 4c) + (10 + 4c) + T(n-2) = \dots = \\ (10 + 4c)n + (2 + c) \in \Theta(n)$$

• Si  $t$  es equilibrado sus dos subárboles tienen del orden de  $n/2$  elementos y son también equilibrados. Por tanto, el número de OE que se realizan en la

ejecución de  $\text{Altura}(t)$  para un árbol  $t$  con  $n$  elementos es:

$$T(n) = (1+c) + (1+c+1+T(n/2)) + 1+c+1+T(n/2) + 3 = 8 + 3c + 2T(n/2).$$

$$T(0) = 2+c.$$

Para resolver esta ecuación en recurrencia se hace el cambio  $t_k = T(2^k)$ , con lo que obtenemos

$$t_k - 2t_{k-1} = 8 + 3c, \text{ ecuación no homogénea de ecuación característica } (x-2)(x-1) = 0.$$

$$\text{Por tanto, } t_k = c_1 2^k + c_2.$$

Deshaciendo los cambios,

$$T(n) = c_1 n + c_2.$$

Para calcular las constantes, nos apoyamos en la condición inicial  $T(0) = 2+c$ , junto con el valor de  $T(1)$ , que puede ser calculado basándonos en la expresión de la ecuación en recurrencia:  $T(1) = 8 + 3c + 2(2 + c)$ . Finalmente obtenemos

$$T(n) = (10 + 4c)n + (2 + c) \in \Theta(n).$$

**Función Mezcla** : en este problema interesa el tratamiento de los tamaños de cada árbol involucrado

```
PROCEDURE Mezcla(t1,t2:arbol):arbol;
(* devuelve un arbol binario de busqueda con los elementos de
   los dos arboles binarios de busqueda t1 y t2. La funcion Ins
   inserta un elemento en un arbol binario de busqueda *)
BEGIN
  IF Esvacio(t1) THEN                                (* 1 *)
    RETURN t2                                         (* 2 *)
  ELSIF Esvacio(t2) THEN                              (* 3 *)
    RETURN t1                                         (* 4 *)
  ELSE                                                (* 5 *)
    RETURN Mezcla(Mezcla(Ins(t1,Raiz(t2)),Izq(t2)),
                  Der(t2))                            (* 6 *)
  END                                                (* 7 *)
END Mezcla;
```

Supondremos que las operaciones básicas del tipo abstracto de datos *arbol* (*Raiz*, *Izq*, *Der*, *Esvacio*) son  $O(1)$ , así como las operaciones *Opera* (que no es relevante lo que hace) y *Max2* (que calcula el máximo de dos números). Por otro lado, supondremos que la complejidad de la función *Ins* es  $O(\log n)$ .

Para resolver este problema vamos a suponer que el tiempo de ejecución del procedimiento *Ins*, que inserta un elemento en un árbol binario de búsqueda, es  $A \log n + B$ , siendo  $A$  y  $B$  dos constantes. Supongamos también que  $n$  y  $m$  son el número de elementos de  $t_1$  y  $t_2$  respectivamente.

Para estudiar el tiempo de ejecución  $T(n,m)$  consideraremos, al igual que hicimos para la función anterior, dos casos extremos: que el árbol  $t_2$  sea degenerado (es decir, una lista) o que sea equilibrado.

• Si  $t_2$  es degenerado, podemos suponer sin pérdida de generalidad que  $\text{Esvacio}(\text{Izq}(t_2))$  y que para todo  $a$  subárbol de  $t_2$  se verifica que  $\text{Esvacio}(\text{Izq}(a))$ . Por tanto, vamos a ver el número de OE que se realizan en cada línea de la función en este caso:

- En la línea (1) se invoca a  $\text{Esvacio}(t_1)$ , lo que supone  $1+c$  OE.
- En la línea (2) se efectúa 1 OE.
- Análogamente, las líneas (3) y (4) realizan  $(1+c)$  y 1 respectivamente.

– Para estudiar el número de OE que realiza la línea (6), vamos a dividirla en cuatro partes:

a)  $a1 := \text{Ins}(t1, \text{Raiz}(t2))$ , siendo  $a1$  una variable auxiliar para efectuar los cálculos. Se efectúan  $2+c+A\log n+B$  operaciones elementales: la llamada a  $\text{Raiz}$  (1), el tiempo que ésta tarda ( $c$ ), la llamada a  $\text{Ins}$  (1 OE), y su tiempo de ejecución ( $A\log n+B$ ).

b)  $a2 := \text{Mezcla}(a1, \text{Izq}(t2))$ , siendo  $a2$  una variable auxiliar para efectuar los cálculos. Se efectúan aquí  $2+c+T(n+1,0)$  operaciones elementales: llamada a  $\text{Izq}$  (1), el tiempo que ésta tarda ( $c$ ), la llamada a  $\text{Mezcla}$  (1 OE), y su tiempo de ejecución, que será  $T(n+1,0)$ , pues estamos suponiendo que  $\text{Esvacio}(\text{Izq}(a))$  para todo a subárbol de  $t2$ .

c)  $a3 := \text{Mezcla}(a2, \text{Der}(t2))$ , siendo  $a3$  una variable auxiliar para efectuar los cálculos. Se efectúan  $2+c+T(n+1,m-1)$  operaciones elementales: la llamada a  $\text{Der}$  (1), el tiempo que ésta tarda ( $c$ ), la llamada a  $\text{Mezcla}$  (1 OE), y su tiempo de ejecución, que será  $T(n+1,m-1)$ , pues estamos suponiendo que  $\text{Esvacio}(\text{Izq}(a))$  para todo a subárbol de  $t2$  o, lo que es igual, que el número de elementos de  $\text{Der}(t)$  es  $m-1$ .

d) RETURN  $a3$ , que realiza 1 OE.

Por tanto, la ejecución de  $\text{Mezcla}(t1, t2)$  en este caso es :

$$T(n,m) = 9 + 5c + B + A\log n + T(n+1,0) + T(n+1,m-1)$$

con las condiciones iniciales  $T(0,m) = 2 + c$  y  $T(n,0) = 3 + 2c$ .

Para resolver la ecuación en recurrencia podemos expresarla como:

$T(n,m) = 12 + 7c + B + A\log n + T(n+1,m-1)$  haciendo uso de la segunda condición inicial. Desarrollando telescópicamente la ecuación:

$$\begin{aligned} T(n,m) &= 12 + 7c + B + A\log n + T(n+1,m-1) = \\ &= (12+7c+B+A\log n) + (12+7c+B+A\log(n+1)) + T(n+2,m-2) = \\ &\dots\dots\dots \\ &= m(12 + 7c + B) + \left( \sum_{i=0}^{m-1} A\log(n+i) \right) + T(n+m,0) = \\ &= m(12 + 7c + B) + 2c + 3 + A \left( \sum_{i=0}^{m-1} \log(n+i) \right). \end{aligned}$$

Pero como  $\log(n+i) \leq \log(n+m)$  para todo  $0 \leq i \leq m$ ,

$$T(n,m) \leq m(12 + 7c + B) + 2c + 3 + Am\log(n+m) \in O(m\log(n+m))$$

- El segundo caso es que  $t2$  sea equilibrado, para el que se demuestra de forma análoga que

$$T(n,m) \in O(m\log(n+m)).$$