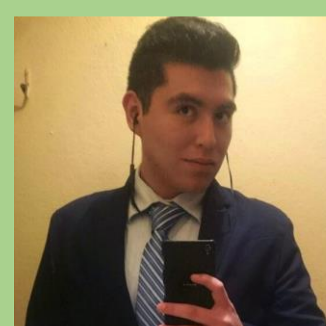
2018

Análisis de algoritmos



Erick Efrain Vargas Romero

Prof. Franco Martínez Edgardo Adrián

Análisis de algoritmos recursivos

3CM2

1. Calcular la cota de complejidad para el algoritmo de la siguiente función recursiva

```
int FuncionRecursiva(int num)
{
            if (num == 0)
                        return 1;        T(0) = 1
            else if (num < 2)
            {
                        resultado=0;
                        for(i=0;i<num*num;i++)
                                    resultado*=num;
            }           return resultado;
            else
T(n-1) + T(n-2) + 1     return FuncionRecursiva( num - 1 )*FuncionRecursiva(num - 2);
}
```

Si consideramos los return y llamadas a función como operaciones básicas obtenemos

$$T(n) = \begin{cases} 1 & n < 2 \\ T(n-1) + T(n-2) + 1 & n \geq 2 \end{cases}$$

Por tanto

$$T(n) = T(n-1) + T(n-2) + 1$$

$$T(n) - T(n-1) - T(n-2) = 1 \quad \textit{Es no homogénea por tanto}$$

$$(x^2 - x^1 - 1)(x - 1) = 0 \quad \textit{Además tenemos 3 raices}$$

$$r_1 = 1, r_2 = \frac{1+\sqrt{5}}{2} \; y \; r_3 = \frac{1-\sqrt{5}}{2} \quad \textit{Todas las raices son diferentes}$$

$$C_1 + \left(\frac{1+\sqrt{5}}{2}\right)^n C_2 + \left(\frac{1-\sqrt{5}}{2}\right)^n C_3 = T(n)$$

Si calculamos C_1, C_2 y C_3 utilizando T(0), T(1) y T(2) obtenemos que

T(0) = C_1=1

T(1) = $C_1 + \left(\frac{1+\sqrt{5}}{2}\right)^1 C_2 + \left(\frac{1-\sqrt{5}}{2}\right)^1 C_3 = 1$

T(2)= $C_1 + \left(\frac{1+\sqrt{5}}{2}\right)^2 C_2 + \left(\frac{1-\sqrt{5}}{2}\right)^2 C_3 = 1$

$$\therefore C_1 = 1, C_2 = \frac{5 - 3\sqrt{5}}{2} \, y \; C_3 = -\frac{8+\sqrt{5}}{2}$$

Finalmente

$$T(n) = 1 + \left(\frac{1+\sqrt{5}}{2}\right)^n \frac{5-3\sqrt{5}}{2} - \left(\frac{1-\sqrt{5}}{2}\right)^n \frac{8+\sqrt{5}}{2}$$

$$\therefore O\left(\frac{1+\sqrt{5}}{2}\right)^n$$

2. Calcular la complejidad de la implementación recursiva del producto

```
int Producto( int a, int b)
{

        if (b==0)
                return 0;                      1
        else
                return a + Producto(a,b-1);          T(n-1) + 1
}
```

Si consideramos los return y llamadas a función como operaciones básicas obtenemos

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

Por tanto

$$T(n) = T(n-1) + 1$$

$$T(n) - T(n-1) = 1 \quad \textit{Es no homogénea por tanto}$$

$$(x^1 + 1)(x - 1) = 0 \quad \textit{Además tenemos 2 raices}$$

$$r_1 = -1, r_2 = 1 \quad \textit{Ambas raices son diferentes}$$

$$T(n) = C_1(-1)^n + C_2(1)^n$$

Utilizando la condición inicial T(0) = 1

$$T(0) = 1 = C_1 + C_2$$

$$T(1) = T(1-1) + 1 => T(1) = 2 \textit{ es decir } T(1) = 2 = -C_1 + C_2$$

Calculando los coeficientes obtenemos que

$$\therefore C_1 = -\frac{1}{2} \; y \; C_2 = \frac{3}{2}$$

Finalmente

$$T(n) = -\frac{(-1)^n}{2} + \frac{2}{3}n$$

$$\therefore O(n)$$

3. Calcular el costo de un recorrido en in-orden de un árbol binario completamente lleno

```
void TraverseInorder (TreeNode root)
{
        if(root!=null)                                  1
        {
                TraverseInorder(root.getLeft());        T(n/2)
                process (root.getValue());
                TraverseInorder(root.getRight());       T(n/2)
        }
}
```

Si consideramos las llamadas a función y la comparación como operaciones básicas obtenemos

$$T(n) = \begin{cases} 1 & n = 0 \\ T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + 1 & n > 0 \end{cases}$$

Debemos aplicar teorema maestro para llegar a la solución correcta

Por tanto, usando caso dos del teorema maestro

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + 1$$

$$Nótese\ que\ a = 1, b = 2\ y\ f(n) = 1\ por\ tanto$$

$$\Theta\left(n^{\log_b a}\right) = \Theta\left(n^{\log_2 1}\right) = \Theta(n^0) = \Theta(1)$$

$$\therefore \Theta(\log(n))$$

4. Calcular la cota de complejidad de un algoritmo de búsqueda ternaria

```
double BusquedaTernaria(double f[] ,int l ,int r ,double absolutePrecision )
{
        if(r-l <=  absolutePrecision)
        {
                return  ( l  +  r ) / 2.0;        1
        }
        else
        {
            int m1  =  ( 2 * l  +  r ) / 3;
                int m2  =  ( l  +  2 * r ) / 3;

            if(f[m1]< f[m2])
            {
            return  BusquedaTernaria( f ,  m1 ,  r ,  absolutePrecision );        T(n/3)
                }
        else
            {
        T(n/3)        return  BusquedaTernaria ( f ,  l ,  m2 , absolutePrecision );
                }
        }
}
```

Si consideramos las llamadas a función y la comparación como operaciones básicas obtenemos

$$T(n) = \begin{cases} 1 & n = 0 \\ T\left(\frac{2n}{3}\right) + 1 & n > 0 \end{cases}$$

Por tanto debemos aplicar teorema maestro para poder obtener la cota del algoritmo de búsqueda ternaria

$$T(n) = T\left(\frac{2}{3}n\right)$$

$$\textit{Nótese que } a = 1, b = \frac{3}{2} \text{ } y \text{ } f(n) = 1$$

$$\textit{Aplicando caso dos del teorema maestro}:$$

$$\Theta\left(n^{\log_{\frac{3}{2}} 1}\right) = \Theta(n^0) = \Theta(1)$$

$$\therefore T(n) = \Theta\left(n^{\log_{\frac{3}{2}} 1} \log n\right) = \Theta(\log n)$$

5. Calcular la cota de complejidad del algoritmo de ordenamiento quick sort

```
QuickSort(lista, inf, sup)
{
    elem_div = lista[sup];
    i = inf - 1;
    j = sup;
    cont = 1;

            if (inf >= sup)          1
                    return;          1

    while (cont)
            {
                    while (lista[++i] < elem_div);
                    while (lista[--j] > elem_div);
                    if (i < j)          n
                    {
                            temp = lista[i];
                            lista[i] = lista[j];
                            lista[j] = temp;
                    }
                    else
                    {
                            cont = 0;
                    }
            }

            temp = lista[i];
            lista[i] = lista[sup];
            lista[sup] = temp;

            QuickSort (lista, inf, i - 1);          T(n/2)
            QuickSort (lista, i + 1, sup);          T(n/2)
}
```

Nuevamente nos apoyaremos del teorema maestro para la resolución de este problema, considerando como operaciones básicas la primer condición que se realiza y los returns

$$(n) = \begin{cases} 2 & inf \geq sup \\ 2T\left(\frac{n}{2}\right) + n & inf < sup \end{cases}$$

$$Por\ teorema\ maestro, en\ su\ caso\ dos$$

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

$$Nótese\ que\ a = 2, b = 2\ y\ f(n) = n\ por\ tanto$$

$$\Theta\left(n^{\log_2 2}\right) = \Theta(n^1) = \Theta(n) \quad lo\ cual\ es\ correcto$$

$$\therefore \Theta\left(n^{\log_2 2}\log(n)\right) = \Theta(nlog(n))$$

6. Resolver las siguientes ecuaciones y dar su orden de complejidad:
   a. $T(n) = 3T(n-1) + 4T(n-2) \Longrightarrow n > 1; T(0) = 0; T(1) = 1$

$$T(n) - 3T(n-1) + 4T(n-2) = 0 \qquad \textit{Tenemos una ecuación homogenea}$$
$$x^2 - 3x - 4 = 0 \ \textit{lo que es igual a} \ (x-4)(x+1) = 0$$
$$\textit{Obtenemos las raices}: r_1 = 4 \ y \ r_2 = -1 \ \textit{las cuales son diferentes}$$
$$\therefore C_1(4)^n + C_2(-1)^n \quad \textit{Calculando los coeficientes} \ C_1 \ y \ C_2$$
$$C_1 = \frac{1}{3} \ y \ C_2 = \frac{1}{3} \quad \textit{si sustituimos}$$
$$T(n) = \frac{1}{3}(4)^n + \frac{1}{3}(-1)^n$$
$$\therefore O\left(\frac{4^n}{3}\right)$$

   b. $T(n) = 3T(n-1) + 4T(n-2) + (n+5)2^n \Longrightarrow n > 1; T(0) = 5; T(1) = 27$

$$T(n) - 3T(n-1) - 4T(n-2) = (n+5)2^n \quad \textit{Tenemos una ecuación no homogenea}$$
$$\textit{Debemos notar que en este caso} \ b = 2 \ y \ d = 1$$
$$\therefore (x^2 - 3x - 4)(x-2)^2 = 0 \qquad \textit{Si factorizamos aún más}$$
$$(x+1)(x-4)(x-2)^2 = 0$$
$$\textit{Obtenemos cuatro raices}: \quad r_1 = -1, r_2 = 4, r_{3,4} = 2$$
$$\therefore T(n) = C_1(-1)^n + C_2(4)^n + C_3 n^2 (2)^n + C_4 n^3 (2)^n \quad \textit{Si calculamos los coeficientes}$$
$$C_4 = \frac{179}{178}, C_3 = -\frac{679}{178}, C_2 = \frac{3348}{445} \ y \ C_1 = -\frac{1123}{445}$$
$$\therefore T(n) = -\frac{1123}{445}(-1)^n + \frac{3348}{445}(4)^n - \frac{679}{178}n^2(2)^n + \frac{179}{178}n^3(2)^n$$
$$O\left(\frac{3348}{445}(4)^n\right)$$

   c. $T(n) - 2T(n-1) = 3^n \Longrightarrow n \geq 2; T(0) = 0; T(1) = 1$

$$\textit{Tenemos una ecuación no homogenea, además notemos que} \ b = 3 \ y \ d = 0$$
$$\therefore (x-2)(x-3) = 0$$
$$\textit{Con esto obtenemos dos raices} \ r_1 = 2 \ y \ r_1 = 3$$
$$\therefore T(n) = C_1 2^n + C_2 3^n \quad \textit{Ahora calculamos los coeficientes} \ C_1 \ y \ C_2$$
$$C_1 = -1 \ y \ C_2 = 1$$
$$\therefore T(n) = -2^n + 3^n$$
$$O(3^n)$$

7. Calcular la cota de complejidad que tendrían los algoritmos con los siguientes modelos recurrentes
   a. $T(n) = 3T\left(\frac{n}{3}\right) + 4T\left(\frac{n}{2}\right) + 2n^2 + n$

$$\textit{Aplicando teorema maestro y dividiendo} \ T(n) \ \textit{a conveniencia}$$
$$T_1(n) = 3T\left(\frac{n}{3}\right) + 2n^2$$
$$\textit{Aplicando el caso} \ 3$$
$$a = b = 3 \ y \ f(n) = 2n^2 \in O(n^2)$$
$$\Omega\left(n^{\log_3 3 + \varepsilon}\right) \quad si \quad \varepsilon = 2 \ entonces \quad \Omega\left(n^{\log_3 5}\right)$$
$$\therefore \theta(n^2)$$

$$\textit{Nuevamente aplicando teorema maestro a la otra parte de la ecuación}$$
$$T_2(n) = 4T\left(\frac{n}{2}\right) + n$$
$$\textit{Aplicamos el caso 1 para teorema maestro}$$
$$a = 4, b = 2\ y\ f(n) = n\ \in O(n)$$
$$O\left(n^{\log_2 4-\varepsilon}\right)\ si\ \varepsilon = 1\ entonces\ O\left(n^{\log_2 3}\right)$$
$$O(n^2)$$

b. $T(n) = T(n-1) + T(n-2) + T\left(\frac{n}{2}\right) \quad si\ n > 1; T(0) = 1; T(1) = 1$

$$\textit{Nuevamente podemos separar } T(n)$$
$$T_1(n) = T(n-1) + T(n-2), despejamos$$
$$T_1(n) - T(n-1) - T(n-2) = 0 \quad \textit{tenemos una ecuación homogenea}$$
$$x^2 - x - 1 = 0 \quad \textit{resolvemos utilizando fórmula general } x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$
$$r_1 = \frac{1+\sqrt{5}}{2}\ y\ r_2 = \frac{1-\sqrt{5}}{2}$$
$$T_1(n) = C_1\left(\frac{1+\sqrt{5}}{2}\right)^n + C_2\left(\frac{1-\sqrt{5}}{2}\right)^n$$
$$\therefore O\left(\frac{\sqrt{5}^n}{5}\right)$$
$$\textit{Resolvemos } T_2(n) \textit{ por teorema maestro}$$
$$T_2(n) = T\left(\frac{n}{2}\right), \textit{podemos notar que al agrupar tenemos que}$$
$$a = 1, b = 2\ y\ f(n) = 0\ \in O(1)$$
$$T_2(n) = \theta\left(n^{\log_2 1}\right) = \theta(1)$$
$$\therefore \theta(\log(n))$$
$$\textit{Concluimos que } O(C^n)$$

c. $T(n) = T\left(\frac{n}{2}\right) + 2T\left(\frac{n}{4}\right) + 2$

$$\textit{Dividamos } T(n) \textit{a conveniencia, calculemos primeramente } T_1(n)$$
$$T_1(n) = T\left(\frac{n}{2}\right) + 2, \textit{por teorema maestro}$$
$$\textit{Nótese que } a = 1, b = 2\ y\ f(n) = 2\ \in O(1)$$
$$\textit{Aplicamos el caso dos de teorema maestro siendo } \varepsilon = 1$$
$$T_1(n) = \theta\left(n^{\log_2 1}\right) = \theta(1)$$
$$\therefore \Theta\left(n^{log_2 1}\log(\mathrm{n})\right) = \log(\mathrm{n})$$
$$\textit{Resolvemos } T_2(n) = 2T\left(\frac{n}{4}\right)$$
$$\textit{Nótese que } a = 2, b = 4\ y\ f(n) = 0 \in O(1)$$
$$\textit{Aplicamos caso uno de teorema maestro siendo } \varepsilon = 1$$
$$T(n) = \Omega(n^{\log_4 3}) = \mathrm{O}(1)$$
$$\therefore\ \Theta(n^{log_4 2})$$
$$\therefore O(\log(n))$$

d. $T(n) = 2T\left(\frac{n}{2}\right) + 4T\left(\frac{n}{4}\right) + 10n^2 + 5n$

$$\textit{Nuevamente podemos separar a conveniencia}$$

$$Resolvamos\ T_1(n) = 2T\left(\frac{n}{2}\right) + 10n^2\ aplicando\ caso\ tres\ de\ teorema\ maestro$$
$$Nótese\ que\ a = 2, b = 2, y\ f(n) = 10n^2$$
$$T(n) = \Omega\left(n^{\log_2 2+1}\right) = \Omega(n^2)$$
$$\therefore \theta(n^2)$$
$$Resolvamos\ T_2(n) = 4T\left(\frac{n}{4}\right) + 5n\ utilizando\ caso\ os\ del\ teorema\ maestro$$
$$T_2(n) = \theta\left(n^{\log_4 4}\right) = \theta(n)\ concluimos\ \theta(n \log n)$$
$$\therefore O(n^2)$$