Certamen 1: Introducción al análisis de datos.

Sebastián Miranda Vega

3) Complejidad algoritmos. Suponga que se tiene un algoritmo que se puede descomponer de la siguiente forma:

$$T(n) = c_0 T(n/2) + c_1, \qquad T(1) = t_1$$

En donde c_0, c_1 y t_1 son constantes. Resuelva la ecuación recursiva y obtenga la expresión de $T(n)$ en función de n, c_0, c_1 y t_1. Luego calcule el orden de complejidad del algoritmo para $c_0 = 1$ y $c_0 = 2$.

SOLUCIÓN: Utilizando el método de sustitución vemos que:

i) Para $k = 1$ tenemos:

$$T(n) = c_0 T(n/2) + c_1$$

ii) Para $k = 2$ tenemos:

$$T(n) = c_0 \left[c_0 T(n/4) + c_1\right] + c_1 = c_0^2 \, T\left(\frac{n}{2^2}\right) + c_0 c_1 + c_1$$

iii) Para $k = 3$ tenemos:

$$T(n) = c_0^2 \left[c_0 T(n/8) + c_1\right] + c_0 c_1 + c_1 = c_0^3 \, T\left(\frac{n}{2^3}\right) + c_0^2 c_1 + c_0 c_1 + c_1$$

iv) Para $k = 4$ tenemos:

$$T(n) = c_0^3 \left[c_0 T(n/16) + c_1\right] + c_0^2 c_1 + c_0 c_1 + c_1 = c_0^4 \, T\left(\frac{n}{2^4}\right) + c_0^3 c_1 + c_0^2 c_1 + c_0 c_1 + c_1$$

v) Para $k = 5$ tenemos:

$$\begin{aligned} T(n) &= c_0^4 \left[c_0 T(n/32) + c_1\right] + c_0^3 c_1 + c_0^2 c_1 + c_0 c_1 + c_1 \\ &= c_0^5 \, T\left(\frac{n}{2^5}\right) + c_0^4 c_1 + c_0^3 c_1 + c_0^2 c_1 + c_0 c_1 + c_1 \end{aligned}$$

Si seguimos de forma recursiva, podemos notar que para un cierto valor k tenemos que:

$$T(n) = c_0^k \, T\left(\frac{n}{2^k}\right) + c_0^{k-1} c_1 + c_0^{k-2} c_1 + \cdots + c_0^2 c_1 + c_0 c_1 + c_1$$

$$\Longrightarrow T(n) = c_0^k \, T\left(\frac{n}{2^k}\right) + c_1 \sum_{i=0}^{k-1} c_0^i$$

Utilizando la fórmula para la suma geométrica:

$$\sum_{k=0}^{n} r^k = \frac{1 - r^{n-1}}{1 - r}$$

Entonces:

$$T(n) = c_0^k \, T\left(\frac{n}{2^k}\right) + c_1 \left(\frac{1 - c_0^k}{1 - c_0}\right)$$

Si seguimos utilizando el método, tendremos que hacerlo hasta que lleguemos al valor de k tal que $T(1) = t_1$. En este caso, vemos que:

$$\begin{align} T\left(\frac{n}{2^k} = 1\right) &= t_1 \tag{1} \\ \implies \frac{n}{2^k} &= 1 \tag{2} \\ n &= 2^k \tag{3} \\ \implies \log_2 n &= k \tag{4} \end{align}$$

Por lo tanto:

$$\begin{aligned} T(n) &= c_0^k \, T\left(\frac{n}{2^k}\right) + c_1 \left(\frac{1 - c_0^k}{1 - c_0}\right) \\ &= c_0^{\log_2 n} \, T\left(\frac{n}{2^{\log_2 n}}\right) + c_1 \left(\frac{1 - c_0^{\log_2 n}}{1 - c_0}\right) \\ &= n^{\log_2 c_0} \, T\left(\frac{n}{n^{\log_2 2}}\right) + c_1 \left(\frac{1 - n^{\log_2 c_0}}{1 - c_0}\right) \\ &= n^{\log_2 c_0} \, T\left(\frac{n}{n}\right) + c_1 \left(\frac{1 - n^{\log_2 c_0}}{1 - c_0}\right) \\ &= n^{\log_2 c_0} \, T\left(1\right) + c_1 \left(\frac{1 - n^{\log_2 c_0}}{1 - c_0}\right) \end{aligned}$$

Pero $T(1) = t_1$, entonces la función $T(n)$ en función de los parámetros n, c_0, c_1 y t_1 queda expresada de la siguiente forma:

$$T(n) = t_1 \cdot n^{\log_2 c_0} + c_1 \left(\frac{1 - n^{\log_2 c_0}}{1 - c_0}\right)$$

Por otro lado, para analizar el orden de complejidad del algoritmo, consideremos el teorema maestro, el cual establece que para una función de la forma:

$$T(n) = a \, T\left(\frac{n}{b}\right) + O\left(n^d\right)$$

Entonces, el orden de la función viene dado por:

$$\begin{cases} O\left(n^d\right) & \text{si} \quad d > \log_b a \\ O\left(n^d \log n\right) & \text{si} \quad d = \log_b a \\ O\left(n^{\log_b a}\right) & \text{si} \quad d < \log_b a \end{cases}$$

Luego:

i) Si $c_0 = 1$ entonces $T(n) = T(n/2)+c_1$. Identificamos $a = 1$, $b = 2$, $d = 0$. Entonces $\log_2 1 = 0 = d$. Por lo tanto, el orden de complejidad es: $O(n^0 \log n) = O(\log n)$

ii) Si $c_0 = 2$ entonces $T(n) = 2T(n/2) + c_1$. Identificamos $a = 2$, $b = 2$, $d = 0$. Entonces $\log_2 2 = 1 > 0 = d \Longrightarrow d < 1$. Por lo tanto, el orden de complejidad es: $O(n^{\log_2 2}) = O(n)$