IIC2283 — Diseño y Análisis de Algoritmos
Ayudantía 1

Problema 1

Se define una palabra de paréntesis balanceados, como una secuencia de símbolos (y), que estén correctamente anidados. Por ejemplo, la palabra

$$(()(()))(())()$$

es correcta, mientras que la siguiente no lo es.

$$(()((())))(())()$$

Encuentre una ecuación de recurrencia tal que C_{2n} represente la cantidad de palabras de paréntesis balanceados formadas por $2n$ símbolos.

Idea de solución Toda palabra de paréntesis balanceados w, de largo $2n+2$, se puede expresar como:

$$w = (w_1)w_2$$

donde w_1 y w_2 son palabras de paréntesis balanceados, cuyos largos suman $2n$. Luego, si consideramos una palabra fija w^*, de largo $2m \leq 2n$, entonces la cantidad de palabras de largo $2n+2$ de la forma

$$(w^*)w_2$$

es igual a $C_{2(n-m)}$. Por otra parte, hay C_{2m} palabras de largo $2m$, por lo tanto existen $C_{2m} \cdot C_{2(n-m)}$ palabras $(w_1)w_2$, donde w_1 tiene largo $2m$. Al sumar por todos los largos posibles, se obtiene que:

$$C_{2n+2} = \sum_{i=0}^{n} C_{2i} \cdot C_{2(n-i)}$$

Finalmente, para que la recurrencia este bien definida, es necesario definir casos base. En este ejemplo, basta definir $C_0 = 1$.

Problema 2

Determine cuales de las siguientes propiedades son verdaderas. Como notación, se define la función $\text{máx}[f,g]$ como $\text{máx}[f,g](n) = \text{máx}\{f(n), g(n)\}$ para todo n, y similarmente, $\text{mín}[f,g]$ como $\text{mín}[f,g](n) = \text{mín}\{f(n), g(n)\}$.

1. Si $\text{máx}(f,g) \in O(h)$, entonces $f \in O(h)$

   Verdadero: Para todo n se cumple que $f(n) \leq \text{máx}[f,g](n)$. Por otra parte, como $\text{máx}[f,g] \in O(h)$, entonces por definición existen constantes c, n_0, tales que para todo $n \geq n_0$ se cumple que $\text{máx}[f,g](n) \leq c \cdot h(n)$, y por lo tanto $f(n) \leq \text{máx}[f,g](n) \leq c \cdot h(n)$. Luego, usando las mismas constantes, se concluye que $f \in O(h)$.

2. Si $\text{mín}[f,g] \in \Theta(h)$, entonces $f \in \Theta(h)$

Falso: Un contraejemplo es $f(n) = 2^n$, $g(n) = 1$, $h(n) = 1$. En este caso, mín$[f, g] \in \Theta(h)$, ya que para todo n, mín$[f, g](n) = 1 = h(n)$. Por otra parte, para toda constante c, existe n tal que $2^n > c$, y por lo tanto $f \notin O(1)$.

3. Si $f \in O(h)$ y $g \in \Omega(h)$, entonces máx$[0, f - g] \in \Theta(h)$

   Falso: Un contraejemplo es $f(n) = n^2$, $g(n) = n^2$, $h(n) = n^2$. En este caso, para todo n, máx$[0, f - g](n) = 0$, pero para cualquier constante $c > 0$ y entero $n > 0$, $0 < cn^2$, por lo tanto máx$[0, f-g] \notin \Omega(h)$.

4. Si $f \in O(h)$ y $g \in \Omega(h)$, entonces máx$[0, f - g] \in O(h)$

   Verdadero: Como $g(n) \geq 0$, entonces para todo n máx$[0, f-g](n) \leq f(n)$. Luego, se aplica el mismo argumento de la pregunta 1.

5. Si $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 5$, entonces $f \in \Theta(g)$

   Verdadero: Por la definición de límite, se tiene que para todo $\varepsilon > 0$, existe n_0 tal que para todo $n \geq n_0$:

   $$\left| \frac{f(n)}{g(n)} - 5 \right| < \varepsilon$$

   De esta desigualdad, se obtiene:

   $$\frac{f(n)}{g(n)} - 5 < \varepsilon$$
   $$5 - \frac{f(n)}{g(n)} < \varepsilon$$

   Luego, como $g(n) > 0$, se tiene que:

   $$f(n) < (5 + \varepsilon)g(n)$$
   $$f(n) > (5 - \varepsilon)g(n)$$

   Finalmente, si elegimos $\varepsilon = 1$, lo anterior implica que existe n_0 tal que para todo $n \geq n_0$,

   $$f(n) \leq 6g(n)$$
   $$f(n) \geq 4g(n)$$

   y por lo tanto que $f \in \Theta(g)$.

6. Si $f, g \in O(h)$, entonces máx$[f, g] \in O(h)$

   Verdadero: Por definición existe c_1, c_2, n_1, n_2 tales que para todo $n \geq n_1$ $f(n) \leq c_1 h(n)$, y para todo $n \geq n_2$, $g(n) \leq c_2 h(n)$. Luego, si definimos $c =$ máx$\{c_1, c_2\}$, y $n_0 =$ máx$\{n_1, n_2\}$, y consideramos $n \geq n_0$, entonces máx$[f, g](n)$ será igual a $f(n)$ o a $g(n)$. Además, por construcción $f(n) \leq c \cdot h(n)$ y $g(n) \leq c \cdot h(n)$, por lo que máx$[f, g](n) \leq c \cdot h(n)$, y entonces máx$[f, g] \in O(h)$.

Problema 3

Encuentre funciones f y g tales que $f \in \Theta(g)$, pero $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ no existe.

Solución

$$f(n) = \begin{cases} 1 & n \text{ es par} \\ 2 & n \text{ es impar} \end{cases}$$

$$g(n) = \begin{cases} 1 & n \text{ es impar} \\ 2 & n \text{ es par} \end{cases}$$

Para todo n, $f(n) \leq 2 \cdot g(n)$, y $f(n) \geq \frac{1}{2} g(n)$, luego $f \in \Theta(n)$. Pero por otra parte, la sucesión $f(n)/g(n)$ es:

$$\frac{f(n)}{g(n)} = \begin{cases} 2 & n \text{ es impar} \\ 1/2 & n \text{ es paro} \end{cases}$$

Problema 4

Dado un conjunto de números $Q = \{q_1, \ldots, q_m\}$, donde m es potencia de 2, se busca encontrar un polinomio p tal que $p(q) = 0$, para todo $q \in Q$.

1. Describa el polinomio buscado como producto de polinomios de grado 1

   Solución: $(x - q_1) \cdot (x - q_2) \cdots (x - q_m)$

2. Describa y analice un algoritmo simple para obtener el polinomio buscado en forma extendida. El análisis debe considerar el número de multiplicaciones.

   Solución: Se considera el siguiente algoritmo (escrito recursivamente):

   function EncontrarPolinomio($q_1, \ldots, q_m$)
   if m=1 **then**
   return $(x - q_1)$
   end if
   p←EncontrarPolinomio($q_2, \ldots, q_m$)
   return Mult1($(x - q_1), p$)
   end function

   Ahí, la función Mult1 multiplica un polinomio de grado 1 por otro de grado n, utilizando $2(n + 1)$ multiplicaciones.

   Luego, se obtiene la ecuación de recurrencia:

$$T(m) = \begin{cases} 0 & m = 1 \\ T(m-1) + 2m & m > 1 \end{cases}$$

   Al desarrollarla, se obtiene que

$$\begin{aligned} T(m) =& T(m-1) + 2m \\ =& T(m-2) + 2(m-1) + 2m \\ =& T(m-3) + 2(m-2) + 2(m-1) + 2m \\ & \vdots \\ =& \sum_{i=1}^{m} 2i \end{aligned}$$

De la expresión anterior, se concluye que el algoritmo funcionaría en tiempo $O(n^2)$. De todas formas, para obtener las constantes asociadas, y demostrar formalmente el resultado se debería utilizar inducción constructiva.

3. Suponga que $M(\cdot,\cdot)$ es un algoritmo que permite multiplicar dos polinomios de grado n en tiempo $O(n \log n)$. Analice el tiempo de ejecución del siguiente algoritmo, donde Q_i se define como $(x - q_i)$.

```
function EncontrarPolinomio(Q_1, ..., Q_m)
    if n=1 then
        return Q_1
    end if
    l←EncontrarPolinomio(Q_1, ..., Q_{m/2})
    r←EncontrarPolinomio(Q_{m/2+1}, ..., Q_m)
    return M(l,r)
end function
```

Idea de solución En este caso, se plantea la recurrencia:

$$T(m) \leq \begin{cases} 1 & m = 1 \\ k \cdot \frac{m}{2} \log \frac{m}{2} + 2T\left(\frac{m}{2}\right) & m > 1 \end{cases}$$

Luego, se aplica una sustitución de variables, aprovechando que m es potencia de 2:

$$T(2^\ell) \leq \begin{cases} 1 & \ell = 0 \\ k \cdot 2^{\ell-1} \log 2^{\ell-1} + 2T(2^{\ell-1}) & \ell > 1 \end{cases}$$

Desarrollándola:

$$\begin{aligned} T(2^{\ell+1}) &\leq k2^\ell(\ell) + 2T(2^\ell) \\ &\leq k2^\ell(\ell) + 2(k2^{\ell-1}(\ell-1) + 2T(2^{\ell-1})) \\ &= k2^\ell(\ell) + k2^\ell(\ell-1) + 4T(2^{\ell-1})) \\ &\leq k2^\ell(\ell) + k2^\ell(\ell-1) + 4(2^{\ell-2}(\ell-2) + 2T(2^{\ell-2})) \\ &= k2^\ell(\ell) + k2^\ell(\ell-1) + k2^\ell(\ell-2) + 8T(2^{\ell-3}) \\ &\vdots \\ &\leq k2^\ell \sum_{i=1}^{\ell} i \end{aligned}$$

Luego, considerando que $m = 2^\ell$, lo anterior sería $O(m \log^2(m))$. Al igual que en el caso anterior, para demostrar este resultado y obtener las constantes asociadas, se puede utilizar inducción constructiva.