Algoritmos y Estructuras de Datos - UNLP

Resumen Propiedades Matemáticas para Tiempo de Ejecución

Maira Diaz
maira.diaz.73@gmail.com *maira1001001*

Junio, 2013

Porpiedades de Logaritmos

Definición

$$\log_a(x) = y \Leftrightarrow a^y = x,$$

si se cumple $x > 0$, $a > 0$, y $a \neq 1$

Propiedades Triviales

- $\log_a(1) = 0 \Leftrightarrow a^0 = 1$
- $\log_a(a) = 1 \Leftrightarrow a^1 = a$

Propiedades menos triviales, menos

- $\log_a(b.c) = \log_a(b) + \log_a(c)$
- $\log_a(\frac{b}{c}) = \log_a(b) - \log_a(c)$
- $\log_a(b^n) = \log_a(\underbrace{b \ldots b}_{n\ veces}) = \underbrace{\log_a(b) + \ldots + \log_a(b)}_{n\ terminos} = n.\log_a(b)$
- $\log_a(\sqrt[n]{b}) = \log_a(b^{\frac{1}{n}}) = \frac{1}{n}.\log_a(b)$
- Cambio de base: $\log_a(x) = \frac{\log_b(x)}{\log_b(a)}$
- $a^{\log_a(b)} = b$

De código Java a expresión matemática

Tenemos dos algoritmos distintos, pero con el mismo tiempo de ejecución. Vean que el orden de ambos algoritmos es logarítmico.
La cantidad de veces que se ejecutan ambos algoritmos depende de "N"

Primer código

```
int c = N;
while(c > 1){
  algo_de_O(1);
  c = c / 2;
}
```

Por ejemplo, si $N = 32$, entonces entramos al while cuando $c = 32$,$c = 16$, $c = 8$, $c = 4$, $c = 2$ y cuando $c = 1$ ya no entra al while. Entonces, nos preguntamos, ¿cuantas veces se ejecuta el *while* teniendo en cuenta a N y el cálculo $c/2$? Se ejecuta 5 veces
Ahora debemos **relacionar N con la cantidad de veces que se ejecuta el algoritmo**. En este caso relacionamos $N = 32$ con 5, y esto nos lleva a pensar que $32 = 2^5 \Rightarrow \log_2(32) = 5$.

Se puede observar que todos los valores que recibamos de N, con $32 \leq N < 64$, el "while" se va a ejecutar 5 veces. Si N ahora esta $64 \leq N < 128$ el "while" se ejecuta unas 6 veces.
Observen que existen intervalos en donde el "while" se ejecuta la misma cantidad de veces. Esos intervalos son:

N se encuentra entre	cantidad de veces que se ejecuta
$[2^0, 2^1)$	el "while" se ejecuta 0 vec
$[2^1, 2^2)$	el "while" se ejecuta 1 vec
$[2^2, 2^3)$	el "while" se ejecuta 2 vec
$[2^3, 2^4)$	el "while" se ejecuta 3 vec
$[2^4, 2^5)$	el "while" se ejecuta 4 vec
$[2^5, 2^6)$	el "while" se ejecuta 5 vec
$[2^6, 2^7)$	el "while" se ejecuta 6 vec

Podemos concluir que 'N" se ejecuta la misma cantidad de veces para un determinado intervalo, y que dicho intervalo varía en potencias de 2 para el código arriba mencionado.

Segundo código

```
int c = 2;
while(c < N){
  algo_de_O(1);
  c = c * 2;
}
```

Por ejemplo, si $N = 47$, entonces entramos al while cuando $c = 2$,$c = 4$, $c = 8$, $c = 16$, $c = 32$ y cuando $c = 64$ ya no entra al while. Entonces, nos preguntamos, ¿cuantas veces se ejecuta el *while* teniendo en cuenta a N y el cálculo $c * 2$? Se ejecuta 5 veces
Ahora debemos **relacionar N con la cantidad de veces que se ejecuta el algoritmo**. En este caso relacionamos $N = 45$ con 5, y esto nos lleva a pensar que

$\log_2(45) \approx 5$ (tomamos la parte entera del $\log_2(45)$).

Porpiedades de Exponenciales

Definición

Potencia de exponente N

$$a^n = \underbrace{aa\ldots a}_{n\ veces}, \quad n \in N \quad a \in R^+$$

Potencia de exponente nulo

$$a^0 = 1, \quad a \neq 0$$

Potencia de exponente entero negativo

$$a^{-n} = \frac{1}{a^n}, \quad n \in N, \quad a \neq 0$$

Potencia de exponente fraccionario

$$a^{\frac{m}{n}} = \sqrt[n]{a^m}, \quad m \in Z, \quad n \in N$$

Propiedades de Potenciación

- $a^{n+m} = a^n.a^m$
- $a^{n-m} = a^n : a^m = \frac{a^n}{a^m}$
- $(a^n)^m = a^{n.m} = a^{m.n} = (a^m)^n$
- $(a.b)^n = a^n.b^n$
- $(\frac{a}{b})^n = \frac{a^n}{b^n}$
- $(\frac{a}{b})^{-n} = (\frac{b}{a})^n$

Propiedades de la Radicación

- $\sqrt[n]{a^m} = a^{\frac{m}{n}}$
- $\sqrt[n]{a^m} = \sqrt[\frac{n}{p}]{a^{\frac{m}{p}}}$
- $\sqrt[n]{a}.\sqrt[n]{b} = \sqrt[n]{a.b}$
- $\frac{\sqrt[n]{a}}{\sqrt[n]{b}} = \sqrt[n]{\frac{a}{b}}$
- $\sqrt[n]{\sqrt[m]{a}} = \sqrt[n.m]{a}$