Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC1253 - Matemáticas Discretas

Ayudantía 10 - Algoritmos y notación asintótica

Héctor Núñez, Paula Grune, Manuel Irarrázaval

Resumen

Correctitud de un Algoritmo

Un algoritmo es correcto si:

- Termina en un número finito de pasos para cualquier entrada válida.
- Entrega una salida correcta, es decir, cumple con la especificación del problema.

Muchas veces, para demostrar la correctitud se utiliza:

- Invariante de ciclo (para bucles)
- Inducción matemática

Análisis Temporal

Para modelar la complejidad temporal de un algoritmo, podemos definir una función:

$$T(n) = \text{numero de operaciones que ejecuta el algoritmo en función del tamaño de entrada } n$$

Luego podemos clasificar estas funciones según notación asintótica, para así obtener la complejidad del algoritmo. Podemos realizar un análisis similar con estos pasos para obtener la complejidad en memoria de un algoritmo.

Notación Asintótica

$$g \in \mathcal{O}(f) \iff (\exists c > 0)(\exists n_0)(\forall n > n_0)\ g(n) < c \cdot f(n)$$
$$g \in \Omega(f) \iff (\exists c > 0)(\exists n_0)(\forall n > n_0)\ g(n) \geq c \cdot f(n)$$
$$g \in \Theta(f) \iff g \in \mathcal{O}(f) \wedge g \in \Omega(f)$$

Propiedades

$$f(n) = a_k n^k + \cdots + a_0,\ a_k > 0 \quad \Rightarrow \quad f(n) \in \Theta(n^k)$$
$$f(n) = \log_a(n),\ a > 1 \quad \Rightarrow \quad f(n) \in \Theta(\log_b(n)),\ b > 1$$

Crecimientos comúnes

$\Theta(1)$	Constante
$\Theta(\log n)$	Logarítmico
$\Theta(n)$	Lineal
$\Theta(n \log n)$	$n \log n$
$\Theta(n^2)$	Cuadrático
$\Theta(n^3)$	Cúbico
$\Theta(n^k)$	Polinomial
$\Theta(m^n)$	Exponencial
$\Theta(n!)$	Factorial

Ejercicios

Preguta 1

Demuestre que $\log_2(n!) \in \Theta(n \cdot \log_2(n))$ usando la definición de notación Θ (no puede usar límites). Para esta demostración, usted puede asumir la "fórmula de Stirling":

$$n! \in \Theta\left(\sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n\right)$$

donde $\pi = 3{,}14\ldots$ y $e = 2{,}71\ldots$ son constantes.

Pregunta 2

Una fórmula proposicional φ está en k-DNF si está en DNF y cada conjunción tiene exactamente k literales. Por ejemplo, la fórmula $(p \wedge \neg q \wedge \neg q) \vee (\neg p \wedge q \wedge p)$ está en 3-DNF. Denotamos por C_i a la i-ésima cláusula de φ y por ℓ_{ij} al j-ésimo literal de C_i.

Considere el siguiente algoritmo para determinar si una fórmula en k-DNF es satisfactible.

Algorithm 1 DNF-SAT($C_1 \vee C_2 \vee \cdots \vee C_m$, k)

1: **for** $i \in \{1, \ldots, m\}$ **do**
2:     $sat \leftarrow$ TRUE
3:     **for** $j \in \{1, \ldots, k-1\}$ **do**
4:         **for** $t \in \{j+1, \ldots, k\}$ **do**
5:             **if** ℓ_{mj} y ℓ_{mt} son complementarios **then**
6:                 $sat \leftarrow$ FALSE
7:             **end if**
8:         **end for**
9:     **end for**
10:     **if** sat **then**
11:         **return** TRUE
12:     **end if**
13: **end for**
14: **return** FALSE

Definimos la función $T(n)$ como el número de ejecuciones de la línea 5 de DNF-SAT cuando se llama para φ con n literales en total (contando repetidos). Determine justificadamente una expresión en notación $\mathcal{O}$ para $T(n)$.

Pregunta 3

Sean $f : \mathbb{N} \to \mathbb{R}^+$y $g : \mathbb{N} \to \mathbb{R}^+$dos funciones cualesquiera. Demuestre o entregue un contraejemplo para las siguientes afirmaciones:

1. Si $f(n) \in \Theta(g(n))$ entonces $\min\{f(n), g(n)\} \in \Theta(\max\{f(n), g(n)\})$.
2. Si $f(n) \in O(g(n))$ entonces $f(n)^{g(n)} \in O\left(g(n)^{f(n)}\right)$.