Algoritmos y Estructuras de Datos II

Guía 2

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Ejercicios obligatorios de la práctica

Integrante	LU	Correo electrónico
Bruno, Patricio Damián	62/19	pdbruno@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

1. Complejidad

1.1. Funciones polinomiales vs. Funciones exponenciales

(a) Sea $b \in \mathbb{R}$ tal que $b \geq 2$. Demostrar que $n \leq b^n$ para todo $n \in \mathbb{N}$, por inducción en n.

Caso Base $n = 1, b \in \mathbb{R} \;/\; b \geq 2$

$1 \leq b^1 = b \Leftrightarrow 1 \leq 2 \leq b$ al ser b mayor o igual a 2

Hipótesis Inductiva $P(n) \Rightarrow P(n+1)$

$n \leq b^n \Rightarrow n + 1 \leq b^{n+1}$

$n \leq b^n \Leftrightarrow n + 1 \leq b^n + 1$

Ahora quiero ver que $b^n + 1 \leq b^n\, b$

$b^n + 1 \leq b^n\, b \underbrace{\Leftrightarrow}_{b \neq 0} 1 + \frac{1}{b^n} \leq b$

$b \geq 2 \Leftrightarrow b^n \geq 2^n \Leftrightarrow \frac{1}{b^n} \leq \frac{1}{2^n} \Leftrightarrow \boxed{1 + \dfrac{1}{b^n} \leq 1 + \dfrac{1}{2^n}}$ Ahora quiero ver que $1 + \frac{1}{2^n} \leq b$

$n \geq 1 \Leftrightarrow 2^n \geq 2 \Leftrightarrow \frac{1}{2^n} \leq \frac{1}{2} \Leftrightarrow 1 + \frac{1}{2^n} \leq 1 + \frac{1}{2} = \frac{3}{2} \leq 2 \leq b$

Entonces $\boxed{1 + \dfrac{1}{b^n} \leq 1 + \dfrac{1}{2^n} \leq b}$

Así queda demostrado que $n \leq n + 1 \leq b^n + 1 \leq b^{n+1}$

(b) Sea $b \in \mathbb{R}$ tal que $b \geq 2$. Demostrar que $x \leq b^{x+1}$ para todo $x \in \mathbb{R}_{\geq 0}$.

$x \in \mathbb{R}_{\geq 0} \Rightarrow \underbrace{\lfloor x \rfloor \in \mathbb{N}}_{por definición} \wedge \lfloor x \rfloor \geq 0$

$\lfloor x \rfloor \leq x \leq \lfloor x \rfloor + 1$ Ya que la parte entera de x se define como el mayor entero menor o igual a x. Esto implica que $\lfloor x \rfloor + 1$ es necesariamente entero (no negativo) y necesariamente mayor a x (si fuera menor o igual a x, estaríamos contradiciendo la definición de la parte entera)

Sea $n = \lfloor x \rfloor + 1$, por lo demostrado anteriormente sabemos que $n \leq b^n$

Entonces $x \leq \lfloor x \rfloor + 1 \leq b^{\lfloor x \rfloor + 1} \leq b^{x+1}$

(c) Sea $b \in \mathbb{R}, k \in \mathbb{N}$ tal que $b > 1$ y $b^k \geq 2$. Demostrar que $\frac{x}{k} \leq b^{x+k}$ para todo $x \in \mathbb{R}_{\geq 0}$.

Por lo demostrado anteriormente, $\frac{x}{k} \leq b'^{\frac{x}{k}+1}$ para alguna base $b' \geq 2$, ya que $\frac{x}{k} \in \mathbb{R}_{\geq 0}$. Sea esa base b^k.

Entonces vale que $\frac{x}{k} \leq (b^k)^{\frac{x}{k}+1} = b^{x+k}$ para todo $x \in \mathbb{R}_{\geq 0}$

(d) Sea $b \in \mathbb{R}, k \in \mathbb{N}$ tal que $b > 1$ y $b^k \geq 2$. Demostrar que $(\frac{x}{pk})^n \leq b^{n(\frac{x}{p}+k)}$ para todo $x \in \mathbb{R}_{\geq 0}$ y para todo $n, p \in \mathbb{N}$.

(pregunta para el profe que me corrija: no se podría tomar raíz enésima a ambos lado?)

Caso Base $n = 1$

$\frac{x}{pk} \leq b^{\frac{x}{p}+k}$ Sea $y = \frac{x}{x}$

$x \geq 0,\, p \geq 1 \Rightarrow y \geq 0$

$\frac{y}{k} \leq b^{y+k}$ por lo demostrado en el punto c)

Hipótesis Inductiva $P(n) \Rightarrow P(n+1)$

$(\frac{y}{k})^n \leq b^{n(y+k)} \Rightarrow (\frac{y}{k})^{n+1} \leq b^{(n+1)(y+k)}$

$(\frac{y}{k})^n \frac{y}{k} \leq b^{n(y+k)}\, b^{y+k}$ por el Caso Base y por suponer P(n)

En otras palabras C.B $\wedge$ P(n) $\Rightarrow$ P(n+1) (multiplicando sus respectivas desigualdades)

(e) Demostrar que para toda base $b \in \mathbb{R}$ y para todo exponente $p \in \mathbb{N}$ vale que $x^p \in O(b^x)$, vistas como funciones de $x \in \mathbb{R}_{\geq 0}$.

Por lo demostrado en d), y tomando $p = n$ (lo cual es válido porque valía para todo p y n naturales), vale que

$(\frac{x}{pk})^p \leq b^{x+pk} \Leftrightarrow \boxed{x^p \leq b^x (p\,k\,b^k)^p}$

Por otro lado, $x^p \in O(b^x) \Leftrightarrow \exists q, x_0 \in R_{\geq 0} / x \geq x_0 \Rightarrow x^p \leq q\,b^x$

Sea $q = (p\,k\,b^k)^p$, con $k \in \mathbb{N}/b^k \geq 2 \Leftrightarrow k \geq log_b(2)$ entonces q hace que la desigualdad valga para todo $x \geq 0$, en particular vale para $x_0 = 0$

1.2. Cálculo de complejidad mejor y peor caso

P(**in** A: `arreglo(nat)`)) $\rightarrow$ res : arreglo(nat)

```
n ← tam(a)                                                        ▷ Θ(1)
M ← 0                                                             ▷ Θ(1)
for i ← 0 to n − 1 do                                             ▷ Θ(n)
    if A[i] ≥ n then                                              ▷ Θ(1)
        A[i] ← 0                                                  ▷ Θ(1)
    else
        M ← max(M, A[i])                                          ▷ Θ(1)
    end if
end for
B ← nuevo arreglo(nat) indexado desde 0 hasta M inclusive, inicializado en 0   ▷ Θ(M)
for i ← 0 to M do                                                 ▷ Θ(M)
    for j ← i to M do                                             ▷ Θ(M)
        B[A[i]] ← 1 + B[A[i]] + B[A[j]]                           ▷ Θ(1)
    end for
end for
return B
```

Justificación: Las primeras dos líneas son asignaciones de enteros, $\Theta(1)$. La guarda del if es una comparación de enteros, la instrucción siguiente es una asignación de enteros (recordar que acceder a una posición de un arreglo toma tiempo constante) y el bloque del else implica evaluar el máximo entre dos enteros y asignarlo a un variable, todo esto tambíen es $\Theta(1)$. La inicialización de B toma tiempo lineal en función de su tamaño, M. Si A tiene un elemento igual a n-1, estaremos en el peor caso y M tomará su mayor valor posible: n-1. El en el mejor caso, todos los elementos son 0 o mayores o iguales a n, y M valdrá 0. En el peor cado reemplazo $\Theta(M)$ por $\Theta(n)$ y en el mejor, por $\Theta(1)$

$T_{mejor}(n) = \Theta(1) + \Theta(1) + \sum_{i=0}^{n} \Theta(1) + \Theta(1) + \Theta(1)$
$= \Theta(1) + n\,\Theta(1)$
$= \Theta(n)$

$T_{peor}(n) = \Theta(1) + \Theta(1) + \sum_{i=0}^{n} \Theta(1) + \Theta(n) + \sum_{i=0}^{n} \sum_{j=i}^{n} \Theta(1)$
$= \Theta(n) + \Theta(1) \sum_{i=0}^{n} (n - i)$
$= \Theta(n) + \Theta(1)(\sum_{i=0}^{n} n - \sum_{i=0}^{n} i)$
$= \Theta(n) + \Theta(1)(n^2 - \frac{n(n+1)}{2})$
$n^2 - \frac{n(n+1)}{2} = \frac{n^2}{2} - \frac{n}{2} \in \Theta(n^2)$ (*)
$= \Theta(n) + \Theta(n^2)$
$= \Theta(n^2)$

(*)$\frac{n^2}{2} - \frac{n}{2} \in O(n^2)$ trivialmente, ya que la primer función es siempre menor
$\frac{n^2}{2} - \frac{n}{2} \in \Omega(n^2) \Leftrightarrow \exists k, n_0 \geq 0 / n \geq n_0 \Rightarrow \frac{n^2}{2} - \frac{n}{2} \geq k\, n^2$
$\frac{n^2}{2} - \frac{n}{2} \geq k\, n^2 \Rightarrow \frac{1}{2} - \frac{1}{2n} \geq k$ Tomemos $k = \frac{1}{4}$
$\frac{1}{2} - \frac{1}{2n} \geq \frac{1}{4} \Rightarrow \frac{1}{4} \geq \frac{1}{2n} \Rightarrow 2 \leq n$ Entonces alcanza con tomar $n_0 = 2$

2. Invariante de representación y función de abstracción

(a) Escribir en castellano el invariante de representación.

1. e.inactivasVacías y e.inactivasNoVacías no tienen pestañas en común
2. el contenido de las pestañas en e.inactivasNoVacías no puede ser la secuencia vacía
3. e.seleccionada no está en e.inactivasVacías ni en e.inactivasNoVacías
4. los números de las pestañas deben ser los números del 0 al total de pestañas

(b) Escribir formalmente el invariante de representación.

(1) $(\forall t$: tupla $\langle nro$: nat,$contenido$: string$\rangle$, n: nat$)$
$((t \in e.inactivasNoVacias \land n \in e.inactivasVacias) \Rightarrow$
$n \neq t.nro)$

(2) $(\forall t$: tupla $\langle nro$: nat,$contenido$: string$\rangle)$
$(t \in e.inactivasNoVacias \Rightarrow \neq \neg vacia?(t.contenido))$

(3) $(\forall t$: tupla $\langle nro$: nat,$contenido$: string$\rangle$, n: nat$)$
$((t \in e.inactivasNoVacias \land n \in e.inactivasVacias) \Rightarrow$
$t.nro \neq e.seleccionada \land n \neq e.seleccionada)$

(4) $(\forall i$: nat$)(0 \leq i \leq \#e.inactivasVacias + \#e.inactivasNoVacias + 1 \Rightarrow$
$(i = e.seleccionada \lor i \in e.inactivasVacias \lor$
$(\exists t$: tupla $\langle nro$: nat,$contenido$: string$\rangle)$
$(t \in e.inactivasNoVacias \land t.nro = i))$

Rep : $\widehat{estr}$ $\longrightarrow$ bool

Rep(e) $\equiv$ true $\Longleftrightarrow$ (1) $\land$ (2) $\land$ (3) $\land_L$ (4)

(c) Escribir formalmente la función de abstracción.

Abs : $\widehat{estr}$ e $\longrightarrow$ editor {Rep(e)}

Abs(e) $=_{\text{obs}}$ ed: editor | $\#pestañas(ed) = \#e.inactivasVacias + \#e.inactivasNoVacias +$
$1 \land$
$seleccionada?(ed, e.seleccionada) \land$
$(\forall t$: tupla $\langle nro$: nat,$contenido$: string$\rangle$, n: nat$)$
$((n \in e.inactivasVacias \Rightarrow vacia?(texto(ed, n))) \land$
$(t \in e.inactivasNoVacias \Rightarrow texto(ed, t.nro) = t.contenido)) \land$

$texto(ed, e.seleccionada) = e.anteriores \& e.posteriores \land$

$posicionCursor(ed) = long(e.anteriores)$