Estructuras Discretas
Tarea 11
Fecha de entrega: jueves 7 de diciembre de 2023

Profesor: Nestaly Marín Nevárez
Ayudantes de teoría: Eduardo Pereyra Zamudio
Ricardo López Villafán
Ayudantes de laboratorio: Edgar Mendoza León
David Valencia Rodríguez

Resuelve de manera limpia y ordenada los siguientes ejercicios. Indica claramente el número de pregunta que se esta resolviendo.

1. Define recursivamente una función *toma* que reciba un número n y una lista ℓ, y que regrese la lista que contiene los primeros n elementos de ℓ. Toma en cuenta el caso en el que la longitud de la lista sea menor a n. Por ejemplo: *1 punto*
$$toma\,(3,\,[a,\,b,\,c,\,d,\,e]) = [a,\,b,\,c]$$
$$toma\,(3,\,[a,\,b]) = [a,\,b]$$

2. Define recursivamente una función que reciba un árbol binario definido como $T = tree\,(T_1,\,c,\,T_2)$ y regrese una lista que contenga las etiquetas de los nodos en el camino desde la raíz hasta la hoja más a la derecha de T. *2 puntos*

3. Sea $sumimp(n)$ la función especificada como $sumimp(n) = 1 + 3 + 5 + \cdots + (2n + 1)$. *2 puntos*
   a) Define una implementación recursiva $f(n)$ para la función $simp(n)$.
   b) Demuestra que $f(n) = (n + 1)^2$.

4. Sea A una fórmula proposicional cuyos únicos conectivos son $\wedge$, $\vee$ y $\neg$. La fórmula dual de A, denotada por A_D, se construye intercambiando an A cada $\wedge$ por un $\vee$, cada $\vee$ por un $\wedge$, y cada constante o variable p por su negación $\neg p$. *3 puntos*
   a) Define una función recursiva $dual(A)$ que reciba una fórmula A y regrese su dual A_D.
   b) Demuestra que $dual(A) \equiv \neg A$ utilizando inducción sobre fórmulas.

5. Demuestra que el número máximo de hojas en un árbol binario T de altura n es 2^{n-1} y que el máximo número de nodos internos de T es $2^{n-1} - 1$. (La *altura* de un árbol T está definida como la máxima distancia desde la raíz de T hacia cualquiera de sus hojas, mientras que los *nodos internos* de T son todos aquellos que no sean una hoja.) *2 puntos*

1. Define recursivamente una función *toma* que reciba un número n y una lista ℓ, y que regrese la lista que contiene los primeros n elementos de ℓ. Toma en cuenta el caso en el que la longitud de la lista sea menor a n. Por ejemplo: *1 punto*

$$toma\,(3,\, [a,\, b,\, c,\, d,\, e]) = [a,\, b,\, c]$$

$$toma\,(3,\, [a,\, b]) = [a,\, b]$$

$$\left.\begin{array}{l} 1)\ \bullet\ toma\,(n, [\,]) = [\,] \\ 2)\ \bullet\ toma\,(0, \ell) = [\,] \end{array}\right\}$$ tenemos un caso base para cada parámetro

3) $\bullet\ toma\,(n, (a:\ell)) = [a] \sqcup toma\,(n-1, \ell)$

Ejemplo:

$\bullet\ toma\,(3, (a:(b:(c:(d:(e:[\,])))))) =$

$= [a] \sqcup toma\,(2, (b:(c:(d:(e:[\,])))))$

$= [a] \sqcup [b] \sqcup toma\,(1, (c:(d:(e:[\,]))))$

$= [a] \sqcup [b] \sqcup [c] \sqcup toma\,(0, (d:(e:[\,])))$

$= [a] \sqcup [b] \sqcup [c] \sqcup [\,]$

$= [a, b, c]$ (aplicando directamente la operación de concatenación $\sqcup$)

2. Define recursivamente una función que reciba un árbol binario definido como $T = tree(T_1, c, T_2)$ y regrese una lista que contenga las etiquetas de los nodos en el camino desde la raíz hasta la hoja más a la derecha de T. *2 puntos*

Llamaremos a nuestra función "hder":

1) • $hder(void) = [\,]$

2) • $hder(tree(T_1, c, T_2)) = [c] \sqcup hder(T_2)$

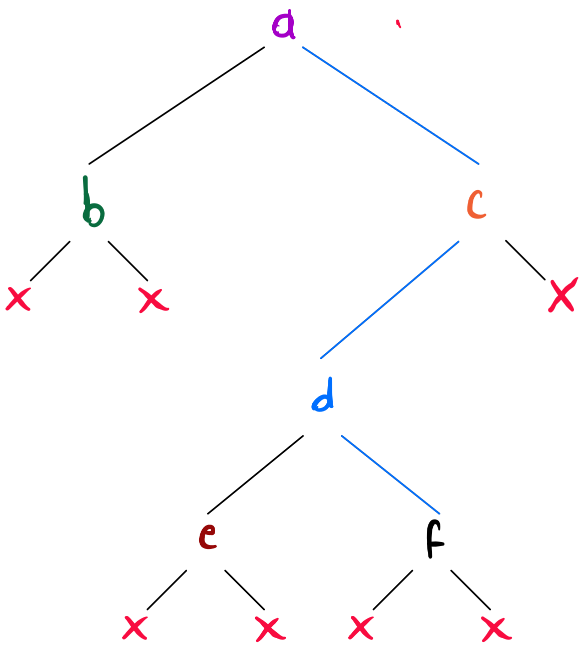
3) • $hder(tree(T_1, c, void)) = [c] \sqcup hder(T_1)$

Ejemplo.

$T = tree(tree(void, b, void), a,$

$tree(tree(tree(void, e, void), d, tree(void, f, void)),$

$c, void))$



$hder(tree(tree(void, b, void), a,$
$tree(tree(tree(void, e, void), d, tree(void, f, void)),$
$c, void)))$

$= [a] \sqcup hder(tree(tree(tree(void, e, void),$
$d, tree(void, f, void)), c, void)))$

$= [a] \sqcup [c] \sqcup hder($
$tree(tree(void, e, void), d, tree(void, f, void)))$

$= [a] \sqcup [c] \sqcup [d] \sqcup hder(tree(void, f, void))$

$= [a] \sqcup [c] \sqcup [d] \sqcup [f] \sqcup hder(void) = [a] \sqcup [c] \sqcup [d] \sqcup [f] \sqcup [\,] \ldots$

3. Sea $sumimp(n)$ la función especificada como $sumimp(n) = 1 + 3 + 5 + \cdots + (2n + 1)$. *2 puntos*

   a) Define una implementación recursiva $f(n)$ para la función $simp(n)$.

   b) Demuestra que $f(n) = (n + 1)^2$.

a) $\cdot f(0) = 1$ caso base

$\cdot f(n) = (2n+1) + f(n-1)$ caso recursivo

b) P.D.: Para todo natural n, $f(n) = (n+1)^2$

Base. $P(0)$. P.D.: $f(0) = (0+1)^2$

$f(0) = 1 = 1^2 = (0+1)^2$

H.I.: Supongamos que $f(n) = (n+1)^2$ para algún $n \in \mathbb{N}$.

P.I.: $P(n+1)$. P.D.: $f(n+1) = (n+2)^2$.

$$
\begin{aligned}
f(n+1) &= 2(n+1) + 1 + f(n) && \text{def. sumimp} \\
&= 2n + 3 + f(n) \\
&= 2n + 3 + (n+1)^2 && \text{H.I} \\
&= 2n + 3 + n^2 + 2n + 1 \\
&= n^2 + 4n + 4 \\
&= (n+2)^2
\end{aligned}
$$

4. Sea A una fórmula proposicional cuyos únicos conectivos son $\wedge$, $\vee$ y $\neg$. La fórmula dual de A, denotada por A_D, se construye intercambiando an A cada $\wedge$ por un $\vee$, cada $\vee$ por un $\wedge$, y cada constante o variable p por su negación $\neg p$. *3 puntos*

   a) Define una función recursiva $dual(A)$ que reciba una fórmula A y regrese su dual A_D.

   b) Demuestra que $dual(A) \equiv \neg A$ utilizando inducción sobre fórmulas.

a) 1). $dual(p) = \neg p$ 4). $dual(\neg A) = \neg\, dual(A)$

2). $dual(true) = false$ 5). $dual(A \wedge B) = dual(A) \vee dual(B)$

3). $dual(false) = true$ 6). $dual(A \vee B) = dual(A) \wedge dual(B)$

b) Demostración por inducción en A.

Base: A es una fórmula atómica p. P.D.: $dual(p) = \neg p$.

$$comp(p) = \neg p \quad \text{por 1)}$$
$$\equiv \neg p \quad \text{reflexividad de } \equiv$$

-La demostración es análoga para $A = true$ y $A = false$.

H.I.: Supongamos que $comp(A) \equiv \neg A$ y $comp(B) \equiv \neg B$ para dos fórmulas proposicionales A y B.

P.I. - Probaremos la propiedad para $\neg A$ y $A \wedge B$. El caso $A \vee B$ es similar a $A \wedge B$.

1) $\text{dual}(\neg A) \equiv \neg\, \text{dual}(A)$ por 4)

$\equiv \neg\neg A$ H.I.

$\equiv A$ doble negación

2) $\text{dual}(A \wedge B) \equiv \text{dual}(A) \vee \text{dual}(B)$ por 5)

$\equiv \neg A \vee \neg B$ H.I.

$\equiv \neg(A \vee B)$ De Morgan

- Concluimos que $\text{dual}(A) \equiv \neg A$ para cualquier fórmula A. ∎

5. Demuestra que el número máximo de hojas en un árbol binario T de altura n es 2^{n-1} y que el máximo número de nodos internos de T es $2^{n-1}-1$. (La *altura* de un árbol T está definida como la máxima distancia desde la raíz de T hacia cualquiera de sus hojas, mientras que los *nodos internos* de T son todos aquellos que no sean una hoja.) *2 puntos*

Demostración por inducción en T.

- definimos la función $hoja(c) = tree(void, c, void)$ que, dada una etiqueta c, regresa una hoja etiquetada.

Base.- $T = hoja(c)$

- T tiene altura 1, tiene $1 = 2^0$ hojas y $0 = 2^0 - 1$ nodos internos.

H.I.: Sean T_1 y T_2 árboles binarios de altura n_1 y n_2 respectivamente. Supongamos que T_1 tiene a lo más 2^{n_1-1} hojas y a lo más $2^{n_1-1}-1$ nodos internos, y que T_2 tiene a lo más 2^{n_2-1} hojas y a lo más $2^{n_2-1}-1$ nodos internos.

P.I.- La altura de $T = tree(T_1, c, T_2)$ está definida como $m = 1 + \max\{n_1, n_2\}$. Queremos demostrar que T tiene 2^{m-1} hojas y $2^{m-1}-1$ nodos internos como máximo.

(Versión 1)

- Notemos que cualquier hoja de T_1 o T_2 sigue siendo una hoja en T. Lo mismo ocurre para los nodos internos de T_1 y T_2. Además, la raíz etiquetada como c es un nodo interno de T.

$\therefore$ Por H.I., T tiene a lo más $h_T = 2^{n_1-1} + 2^{n_2-1}$ hojas y a lo más $ni_T = 2^{n_1-1} - 1 + 2^{n_2-1} - 1 + 1$ nodos internos.

- Como $m = 1 + \text{máx}\{n_1, n_2\}$, se cumple que $n_1 \leq m-1$ y $n_2 \leq m-1$. Por lo tanto: el número de hojas de T es

- $h_T = 2^{n_1-1} + 2^{n_2-1} \leq 2^{m-2} + 2^{m-2} = 2^{m-1}$

y el número de nodos internos de T es.

- $ni_T = 2^{n_1-1} + 2^{n_2-1} - 1 \leq 2^{m-2} + 2^{m-2} - 1 = 2^{m-1} - 1$

(Versión 2)

Podemos definir dos funciones recursivas para contar el número de hojas y nodos internos de un árbol binario.

\# hojas

- $h(\text{leaf}(c)) = 1$ • $h(\text{tree}(T_1, c, T_2)) = h(T_1) + h(T_2)$

nodos internos

- $ni(leaf(c)) = 0$ • $ni(tree(T_1, c, T_2)) = 1 + ni(T_1) + ni(T_2)$

- Luego, usamos estas definiciones para $T = tree(T_1, c, T_2)$:

  - $h(tree(T_1, c, T_2)) = h(T_1) + h(T_2)$ def #hojas

$$\leq 2^{n_1-1} + 2^{n_2-1} \quad \text{H.I.}$$

$$\leq 2^{m-2} + 2^{m-2} \quad m = 1 + \max\{n_1, n_2\}$$

$$= 2^{m-1}$$

  - $ni(tree(T_1, c, T_2)) = 1 + ni(T_1) + ni(T_2)$ def #ni

$$\leq 1 + 2^{n_1-1} - 1 + 2^{n_2-1} - 1 \quad \text{H.I.}$$

$$= 2^{n_1-1} + 2^{n_2-1} - 1$$

$$\leq 2^{m-2} + 2^{m-2} - 1 \quad m = 1 + \max\{n_1, n_2\}$$

$$= 2^{m-1} + 1$$

■