Programación y Algoritmos
Tarea 07
Pablo Antonio Stack Sánchez

Problema (1)

Reglas:

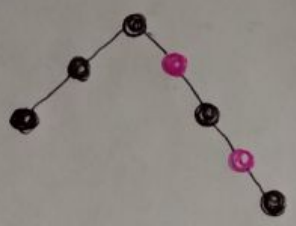
1:- Si un nodo es rojo sus dos hijos son negros

2:- Cada camino desde un nodo a sus hojas contiene el mismo número de nodos negros.

Sea h la altura del arbol, Por la regla 1 sabemos que no puede haber dos nodos rojos consecutivos y debido a que todos los caminos deben tener el mismo número de nodos negros, sabemos que al menos la mitad de los nodos de cualquier camino de la raiz a una hoja deben ser negros. Entonces la altura negra del arbol debe ser almenos $h/2$.

Ejemplo:

La h(b) más corta posible es cuando hay **h** nodos negros consecutivos y la h más grande es cuando se alternan los nodos rojos y negros.



$h(b) = 2$

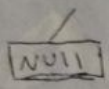
$h = 4$

$h(b) \geq \frac{h}{2}$

$2 \geq \frac{4}{2}$

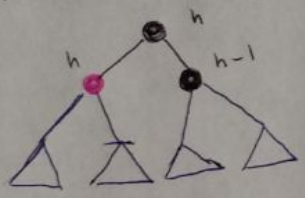
Problema (2)

Para una altura $h = 0$



si la altura es 0 hay solo una hoja y contiene por lo menos $2^0 - 1 = 0$ nodos internos.

ahora consideremos un nodo interno con dos hijos, sus hijos tienen altura negra $h(b)$ o $h(b)-1$, dependiendo si es un nodo negro o rojo.



Los triangulos representan subarboles, si la raiz tiene altura h, la altura del nodo negro es $h-1$ y del nodo rojo es h, para conservar las propiedades del arbol, dado que la altura de un hijo de un nodo es menor a la altura del padre. Podemos aplicar la hipotesis de inducción para concluir que cada hijo tiene por lo menos $2^{h(b)-1} - 1$ nodos internos. Entonces:

$$\left(2^{h(b)-1} - 1\right) + \left(2^{h(b)-1} - 1\right) + 1 = (2)\,2^{h(b)-1} - 1 = \boxed{2^{h(b)} - 1}$$

Problema (3)

dado que $h(b) \geq \frac{h}{2}$

$$n \geq 2^{\frac{h}{2}} - 1$$

moviendo el uno al lado izquierdo

$$n+1 \geq 2^{h/2}$$

aplicando $\log()$

$$\log(n+1) \geq \frac{h}{2}$$

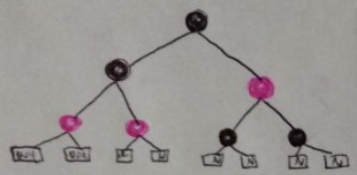
Por lo tanto

$$\boxed{h \leq 2 \log(n+1)}$$

Problema (4)

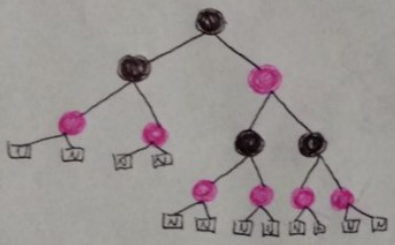
Dado el siguiente arbol valido:

$h(b) = 2$

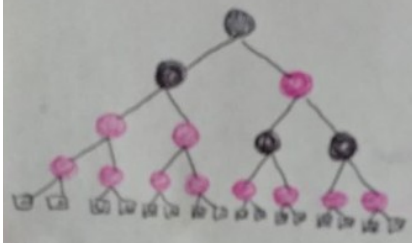


Ahora insertamos nodos rojos en los padres negros

$h(b) = 2$

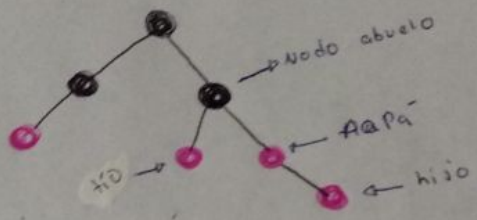


Vemos que continua siendo un arbol valido y la $h(b)$ no cambia.
Ahora, insertamos nodos rojos en padres rojos.

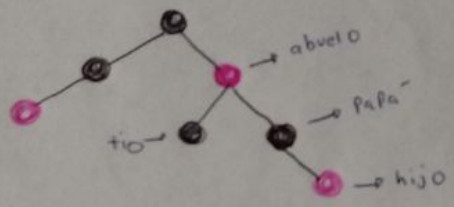


No es un árbol valido, ocurre una violación ya que no puede haber dos nodos rojos consecutivos.

Problema (5)



La altura negra hasta antes de la inserción del nodo que causa la violación es 2. ahora cambiamos de color los nodos abuelo, Papá y tío

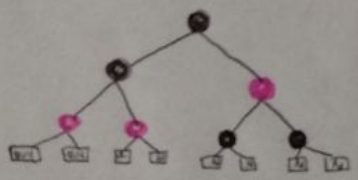


La altura negra no cambia y la complejidad es constante porque solo hay que cambiar el color de 3 nodos.

Problema (4)

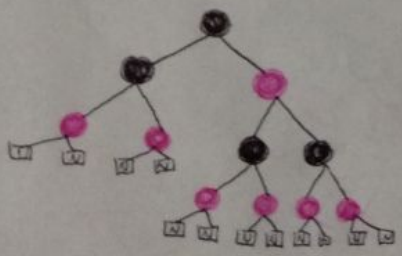
Dado el siguiente arbol valido:

$h(b) = 2$



ahora insertamos nodos rojos en los padres negros

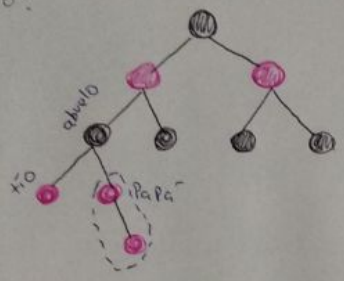
$h(b) = 2$



vemos que continua siendo un arbol valido y la $h(b)$ no cambia. Ahora, insertamos nodos rojos en padres rojos.

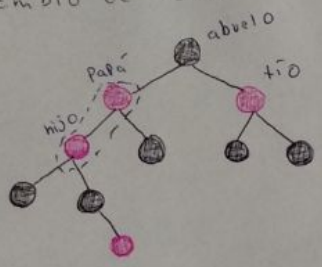
Problema (5)

Segundo caso:



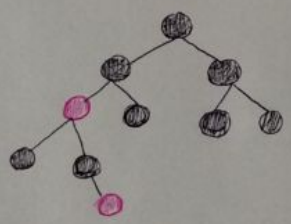
$h(b) = 2$

Aplicamos el cambio de color



$h(b) = 2$

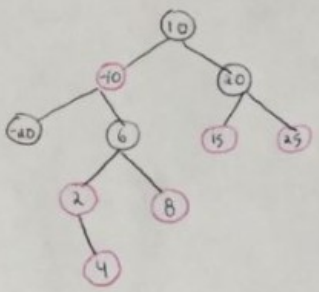
Volvemos a tener el mismo problema, así que aplicamos cambio de color.



$h(b) = 3$

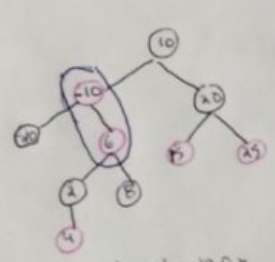
La raiz tiene que ser negra, por lo que la altura negra aumenta en uno. En el peor caso se tendría que recorrer la altura del arbol

Pregunta (6)



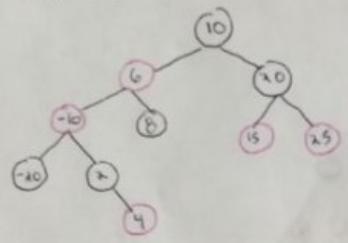
$h(b) = 2$

tenemos el mismo problema que en la pregunta 5, realizamos el cambio de color.

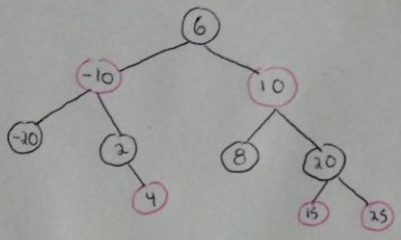


$h(b) = 2$

Esto lo podemos solucionar con una rotación a la izquierda



ahora rotamos a la derecha



$h(b) = 2$

Se soluciona el problema y ya tenemos un árbol valido. La altura negra no cambia. Debido a que máximo se ocupan 2 rotaciones para corregir el árbol la complejidad es constante.

Problema (7)

VER ARBOL_ROJONEGRO.C