



Universidad
Tecnológica
de Pereira

CLASE 33 PRESENCIAL MEDIADA POR HERRAMIENTAS DE VIRTUALIDAD
CAPÍTULO: ÁRBOLES ROJINEGROS
CURSO DE ESTRUCTURA DE DATOS - Código IS304 - Grupo 01

Programa de Ingeniería de Sistemas y Computación
Profesor Hugo Humberto Morales Peña
Viernes 12 de Noviembre de 2021

Borrado

El procedimiento RB-DELETE es una pequeña modificación del procedimiento TREE-DELETE. Después de borrar un nodo, es llamado el procedimiento auxiliar RB-DELETE-FIXUP que ajusta los colores de los nodos y realiza tantas rotaciones como sean necesarias para restaurar las propiedades de árboles rojinegros.

Algoritmo RB-DELETE

```
function RB-DELETE(  $T, z$  )  
1  if  $left[z] == NILleaf$  or  $right[z] == NILleaf$  then  
2       $y \leftarrow z$   
3  else  
4       $y \leftarrow TREE-SUCCESSOR( z )$   
5  if  $left[y] \neq NILleaf$  then  
6       $x \leftarrow left[y]$   
7  else  
8       $x \leftarrow right[y]$   
9   $p[x] \leftarrow p[y]$   
10 if  $p[y] == NILleaf$  then  
11      $T \leftarrow x$   
12 else  
13     if  $y == left[p[y]]$  then  
14          $left[p[y]] \leftarrow x$   
15     else  
16          $right[p[y]] \leftarrow x$   
17 if  $y \neq z$  then
```

```

18      $key[z] \leftarrow key[y]$ 
19     Copy all information fields from  $y$  to  $z$ 
20   if  $color[y] == \text{BLACK}$  then
21      $T \leftarrow \text{RB-DELETE-FIXUP}(T, x)$ 
22   if  $y == p[left[y]]$  then
23      $\text{FREE}(left[y])$ 
24   if  $y == p[right[y]]$  then
25      $\text{FREE}(right[y])$ 
26    $\text{FREE}(y)$ 
27   return  $T$ 

```

Hay 3 diferencias entre los procedimientos TREE-DELETE y RB-DELETE. La primera, todas las referencias a NIL en TREE-DELETE son reemplazadas por las referencias al nodo NILleaf en RB-DELETE. Segunda, la prueba de que si x es NIL en la línea 9 de TREE-DELETE es removida, y la asignación $p[x] \leftarrow p[y]$ es realizada sin condiciones en la línea 9 de RB-DELETE. Por lo tanto, si x es el nodo NILleaf, el apuntador a su padre, apunta al padre del nodo y . Tercera, un llamado a RB-DELETE-FIXUP es realizado en las líneas 20-21 si y es negro. Si y es rojo, las propiedades se mantienen cuando y es borrado, debido a las siguientes razones:

- Ninguna altura negra en el árbol ha cambiado.
- No hay nodos rojos adyacentes, y
- Desde que y no sea la raíz, si este es rojo, la raíz permanecerá negra.

El nodo x utilizado en el llamado a RB-DELETE-FIXUP es uno de dos nodos: o el nodo que era hijo único de y antes de que y fuese borrado si y tuviese un hijo que no fuese el nodo NILleaf, o si y no tiene hijos, x es el nodo NILleaf. En éste último caso, la asignación incondicional en la línea 9 garantiza que el padre de x es ahora el nodo que antes fue el padre de y .

Algoritmo RB-DELETE-FIXUP

```

function RB-DELETE-FIXUP(  $T, x$  )
1   while  $x \neq T$  and  $color[x] == \text{BLACK}$  do
2     if  $x == left[p[x]]$  then
3        $w \leftarrow right[p[x]]$ 
4       if  $color[w] == \text{RED}$  then
5          $color[w] \leftarrow \text{BLACK}$                                 ▷ Case 1
6          $color[p[x]] \leftarrow \text{RED}$                             ▷ Case 1
7          $T \leftarrow \text{LEFT-ROTATE}(T, p[x])$                   ▷ Case 1
8          $w \leftarrow right[p[x]]$                                 ▷ Case 1
9       if  $color[left[w]] == \text{BLACK}$  and  $color[right[w]] == \text{BLACK}$  then
10         $color[w] \leftarrow \text{RED}$                                 ▷ Case 2

```

```

11          $x \leftarrow p[x]$                                  $\triangleright$  Case 2
12     else
13         if  $color[right[w]] == \text{BLACK}$  then
14              $color[left[w]] \leftarrow \text{BLACK}$                  $\triangleright$  Case 3
15              $color[w] \leftarrow \text{RED}$                          $\triangleright$  Case 3
16              $T \leftarrow \text{RIGHT-ROTATE}( T, w )$              $\triangleright$  Case 3
17              $w \leftarrow right[p[x]]$                          $\triangleright$  Case 3
18              $color[w] \leftarrow color[p[x]]$                  $\triangleright$  Case 4
19              $color[p[x]] \leftarrow \text{BLACK}$                  $\triangleright$  Case 4
20              $color[right[w]] \leftarrow \text{BLACK}$              $\triangleright$  Case 4
21              $T \leftarrow \text{LEFT-ROTATE}( T, p[x] )$            $\triangleright$  Case 4
22              $x \leftarrow T$                                  $\triangleright$  Case 4
23     else
24          $w \leftarrow left[p[x]]$ 
25         if  $color[w] == \text{RED}$  then
26              $color[w] \leftarrow \text{BLACK}$                      $\triangleright$  Case 1
27              $color[p[x]] \leftarrow \text{RED}$                      $\triangleright$  Case 1
28              $T \leftarrow \text{RIGHT-ROTATE}( T, p[x] )$            $\triangleright$  Case 1
29              $w \leftarrow left[p[x]]$                          $\triangleright$  Case 1
30         if  $color[right[w]] == \text{BLACK}$  and  $color[left[w]] == \text{BLACK}$  then
31              $color[w] \leftarrow \text{RED}$                          $\triangleright$  Case 2
32              $x \leftarrow p[x]$                                  $\triangleright$  Case 2
33     else
34         if  $color[left[w]] == \text{BLACK}$  then
35              $color[right[w]] \leftarrow \text{BLACK}$                  $\triangleright$  Case 3
36              $color[w] \leftarrow \text{RED}$                          $\triangleright$  Case 3
37              $T \leftarrow \text{LEFT-ROTATE}( T, w )$              $\triangleright$  Case 3
38              $w \leftarrow left[p[x]]$                          $\triangleright$  Case 3
39              $color[w] \leftarrow color[p[x]]$                  $\triangleright$  Case 4
40              $color[p[x]] \leftarrow \text{BLACK}$                  $\triangleright$  Case 4
41              $color[left[w]] \leftarrow \text{BLACK}$              $\triangleright$  Case 4
42              $T \leftarrow \text{RIGHT-ROTATE}( T, p[x] )$            $\triangleright$  Case 4
43              $x \leftarrow T$                                  $\triangleright$  Case 4
44      $color[x] \leftarrow \text{BLACK}$ 

```

45 **return** T

Si el nodo a borrar y del árbol en RB-DELETE es negro, 3 problemas se pueden generar. Primero, si y es la raíz y un hijo rojo de y se vuelve la nueva raíz, entonces se está violado la propiedad 2. Segundo, si x y $p[y]$ (el cual ahora es $p[x]$) son rojos, entonces se está violado la propiedad 4. Tercero, remover a y causa que cualquier camino que previamente haya contenido a y tenga un nodo menos de color negro. Por lo tanto, la propiedad 5 se ha violado por algún ancestro de y en el árbol.

Ejemplo 1:

Para el árbol rojinegro de la Figura 1 borrar el nodo de información 1.

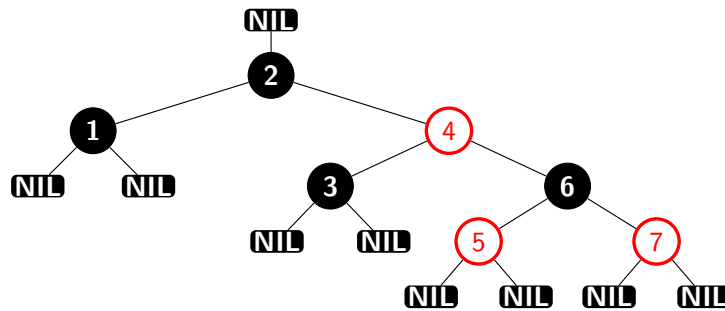


Figura 1: Árbol Rojinegro el cual tiene almacenados los números del 1 al 7.

Después de borrar el número 1 se obtiene el árbol rojinegro de la Figura 3.

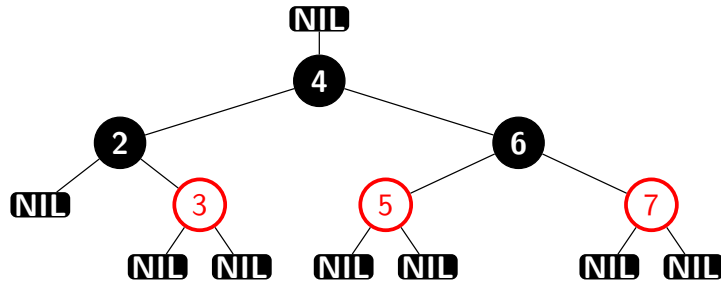


Figura 2: Árbol Rojinegro que se obtiene como resultado después de borrar el número 1.

Ejercicio 2:

Para el árbol rojinegro de la Figura 3 borrar el 2 (nodo que es la raíz del árbol).

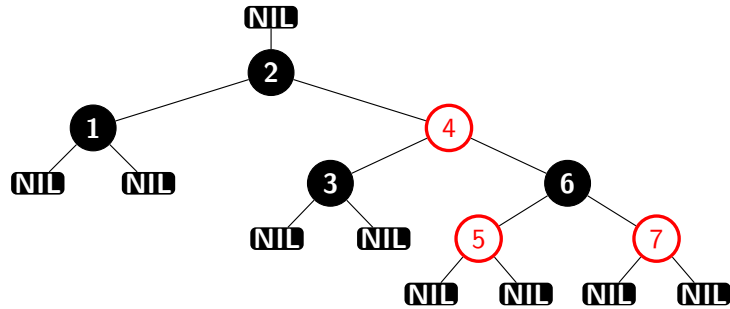


Figura 3: Árbol Rojinegro el cual tiene almacenados los números del 1 al 7.

Ejercicio 3:

Sobre el árbol rojinegro de la Figura 4 borrar el nodo de información 47.

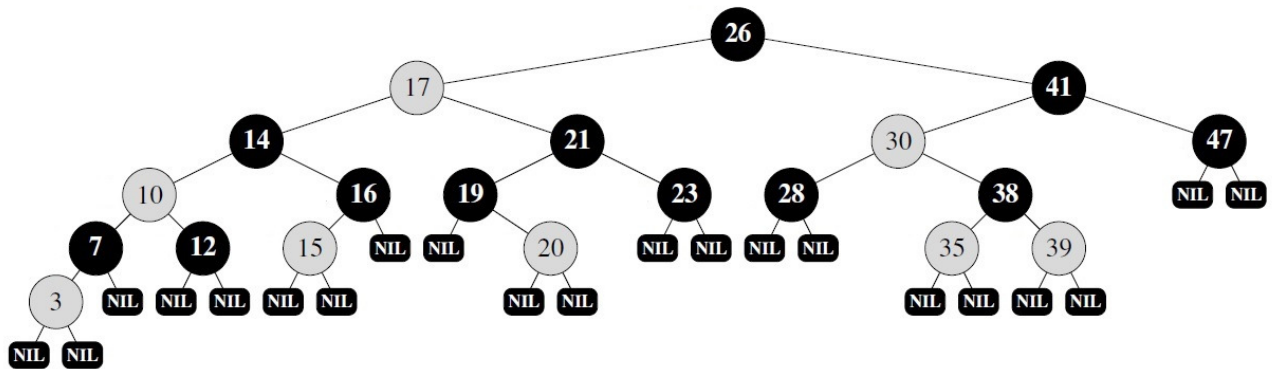


Figura 4: Árbol rojinegro sobre el cual se pide borrar el nodo de información 47.

Ejercicio 4:

Sobre el árbol rojinegro de la Figura 5 borrar el nodo raíz (nodo de información 26).

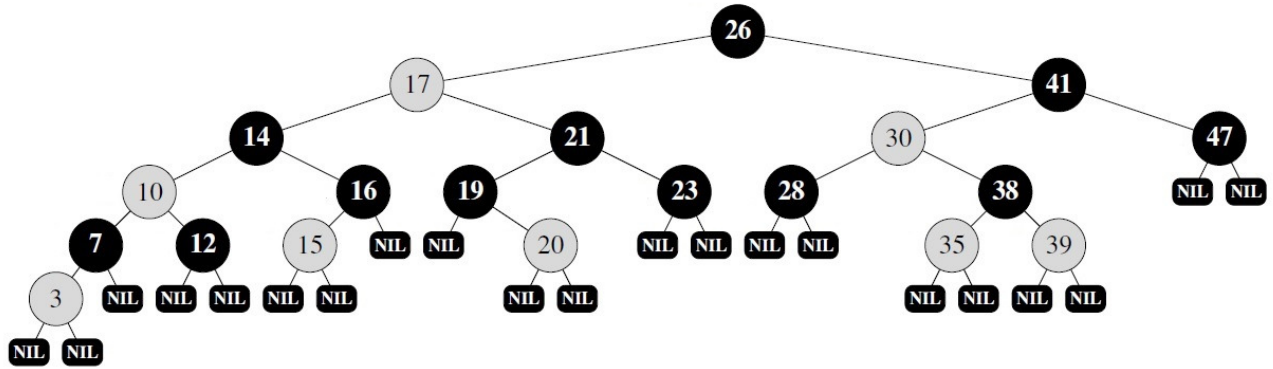


Figura 5: Árbol rojinegro sobre el cual se pide borrar la información del nodo que se encuentra en la raíz.

Asistentes a clase

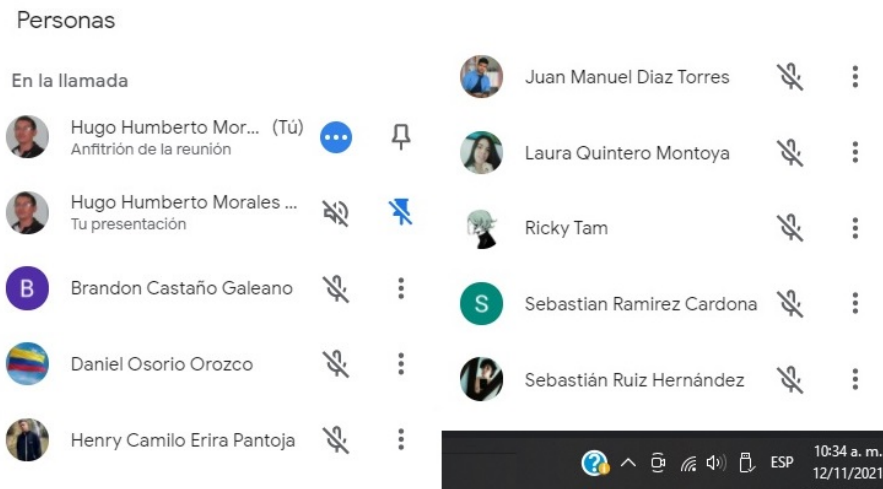


Figura 6: Asistentes a la Clase 33 en el Google Meet (Noviembre 12, 2021).