Árboles rojo-negro

Guillermo Palma

Universidad Simón Bolívar
Departamento de Computación y T.I.

CI-2612: Algoritmos y Estructuras II

Plan

Sobre los árboles rojo-negro

- Son árboles binarios con un campo adicional en los nodos que es un color rojo o negro
- Se tienen reglas que indican como el árbol debe ser coloreado
- Las reglas de coloración aseguran que en el árbol no hay ningún camino que sea el doble de otro
- El árbol es balanceado
- Se garantiza que las operaciones sobre el árbol se ejecutan en tiempo $O(\log n)$
- Hay un centinela que apunta a todas las hojas NIL del árbol y a la raíz

Ejemplo de un árbol rojo-negro I

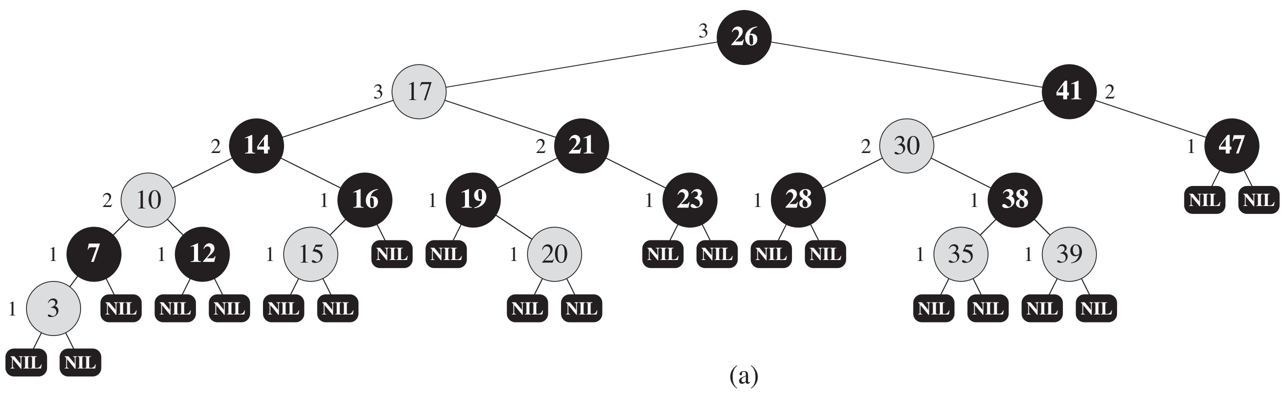


Figura: Árbol rojo-negro con todas las hojas NIL coloreadas de negro. Fuente [1]

Ejemplo de un árbol rojo-negro II

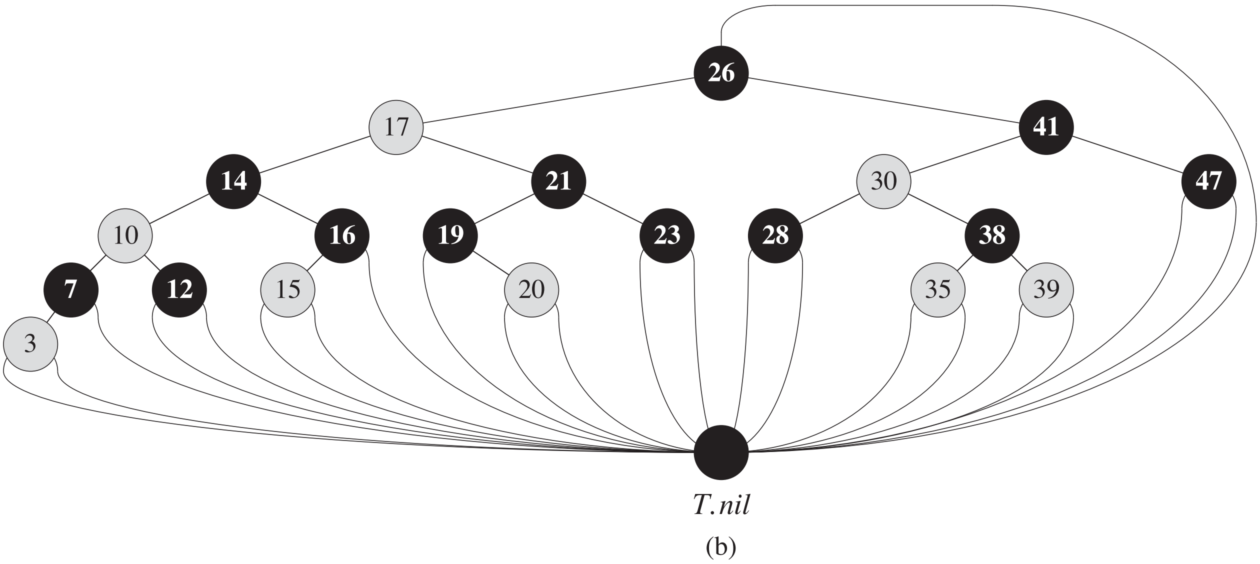


Figura: Árbol rojo-negro las hojas NIL son reemplazadas por un centinela. Fuente [1]

Ejemplo de un árbol rojo-negro III

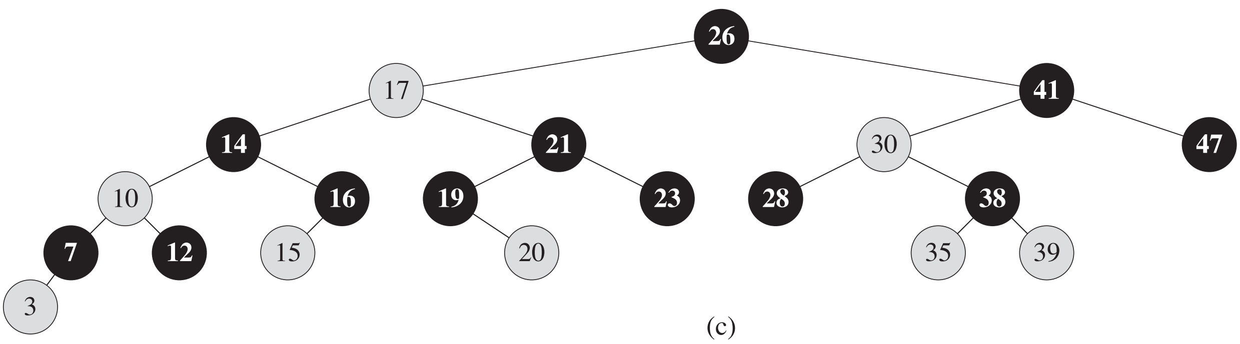


Figura: Árbol rojo-negro donde las hojas son omitidas. Fuente [1]

Propiedades de los árboles rojo-negro

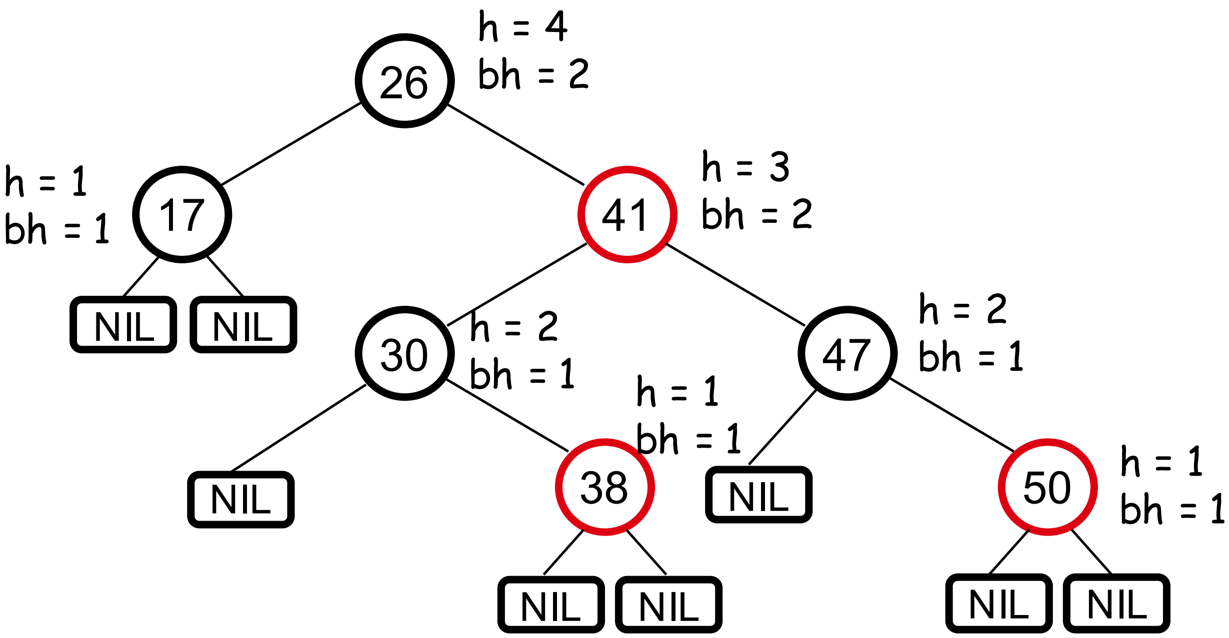
Todo árbol rojo-negro debe satisfacer las siguientes propiedades:

1. Cada nodo es coloreado de rojo o negro
2. La raíz de color negro
3. Cada hoja (NIL) es de color negro
4. Si un nodo es de color rojo, entonces sus hijos son de color negro
5. Para cada nodo, todos los caminos desde el nodo hasta sus hojas, contienen el mismo número de nodos de color negro

Black-height en un árbol rojo-negro

- **Height**: número de lados del camino las largo desde un nodo x hasta una hoja
- **Black-height**: número de nodos de color negro desde un nodo x hasta la hoja, sin incluir x

Ejemplo de un Black-Height en un árbol rojo-negro



Observaciones sobre los árboles rojo-negro

Observación 1

Dado cualquier nodo x con altura $h(x)$, se cumple que $bh(x) \geq \frac{h(x)}{2}$

Observación 2

Dado cualquier subárbol cuya raíz sea el nodo x, se cumple que el subárbol contiene al menos $2^{bh(x)} - 1$ nodos internos

Observaciones sobre los árboles rojo-negro

Lema

Un árbol rojo-negro con n nodos internos tiene una altura de a lo sumo $2\log(n+1)$

Operaciones sobre los árboles rojo-negro

- Las operaciones sobre los árboles binarios, MINIMUM, MAXIMUM, SUCCESSOR, PREDECESSOR, y SEARCH se pueden aplicar a árboles rojo-negro **sin modificación**. El tiempo de ejecución en árboles rojo-negro es $O(\log n)$
- Las operaciones sobre los árboles binarios, TREE-INSERT y TREE-DELETE se pueden aplicar a árboles rojo-negro **con modificaciones** para garantizar que se cumplan las propiedades de los mismos. El tiempo de ejecución en árboles rojo-negro es $O(\log n)$

Rotaciones sobre los árboles rojo-negro

- Operaciones para restaurar las propiedades de los árboles rojo-negro
- Se pueden colorear los nodos de otro color
- Se cambia las estructura de los subárboles
- Se mantiene las estructura y propiedades de los árboles binarios

Rotación izquierda

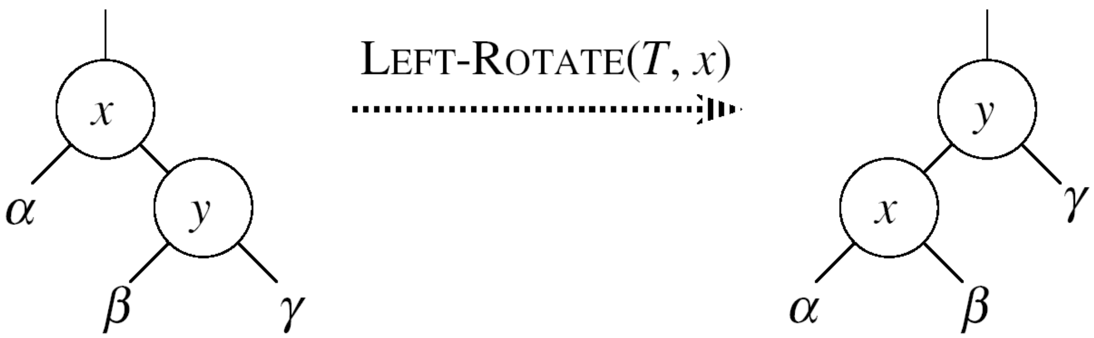


Figura: El hijo derecho *x* de *y* no es *NIL*. Fuente [1]

Ejemplo de rotación izquierda

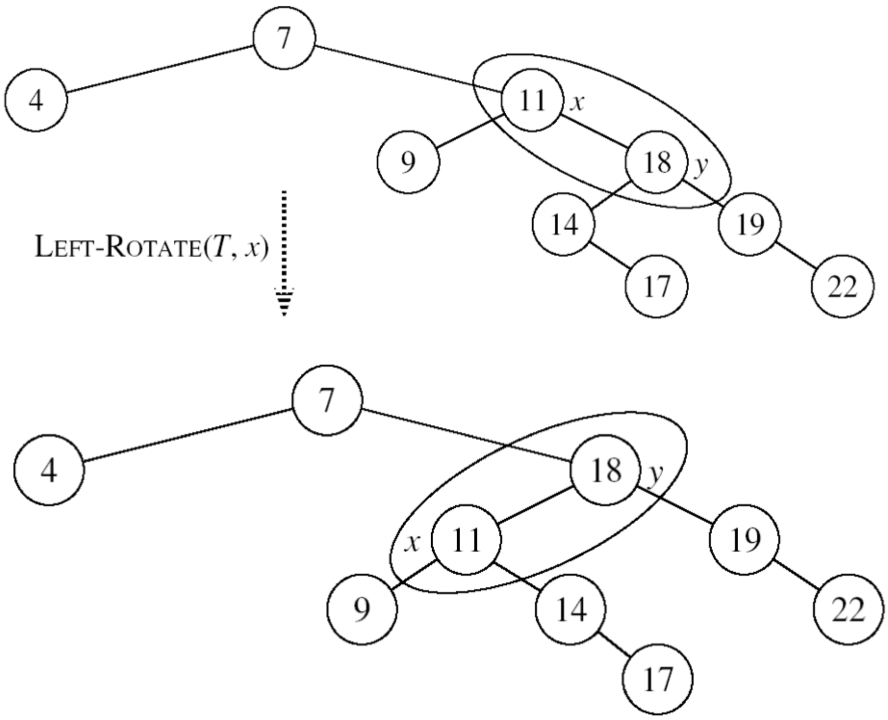


Figura: Fuente [1]

Rotación derecha

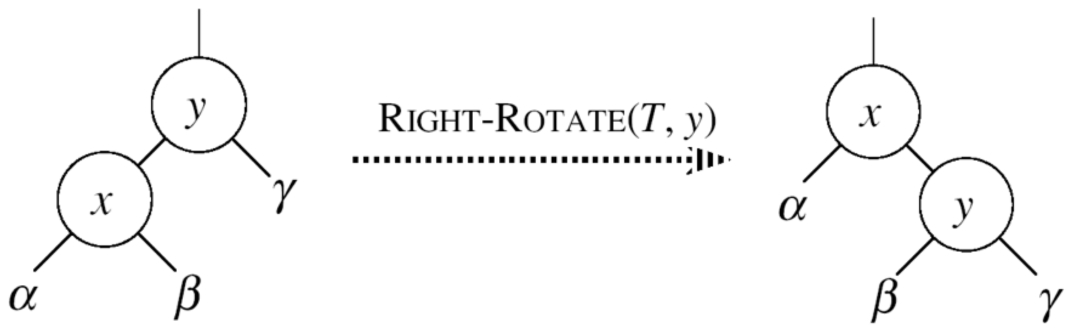


Figura: El hijo izquierdo *x* de *y* no es *NIL*. Fuente [1]

Inserción en los árboles rojo-negro

- Se quiere insertar un nodo con clave z en un árbol rojo-negro
- Se inserta el nodo con clave z usando como en la inserción en los árboles binarios
- Se colorea el nodo con clave z de rojo
- Se restaura las propiedades de los árboles rojo-negro con un procedimiento (RB-INSERT-FIXUP)

Ejemplo de inserción

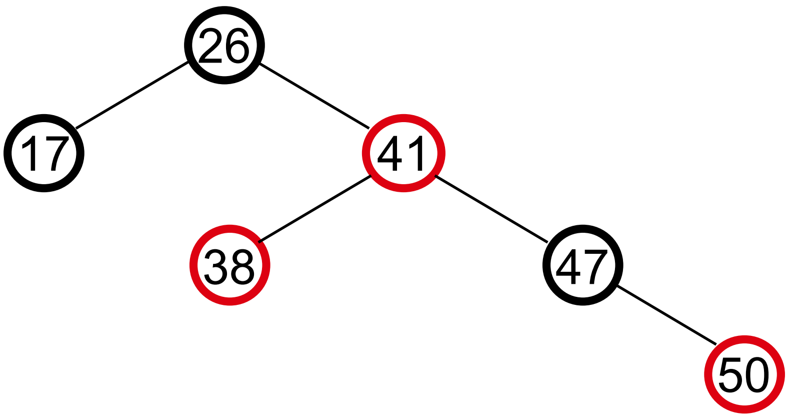


Figura: Se inserta el nodo con clave 38 en el árbol

RB-INSERT-FIXUP caso 1

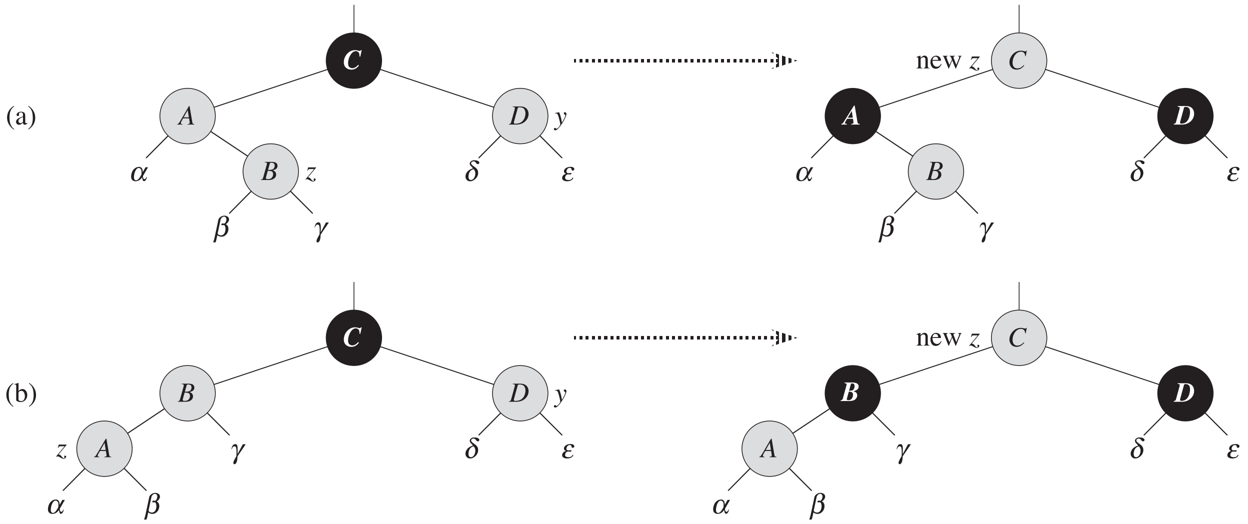


Figura: El tío y de z es un nodo de color rojo. Fuente [1]

Inserción en los árboles rojo-negro

RB-INSERT-FIXUP caso 2 y caso 3

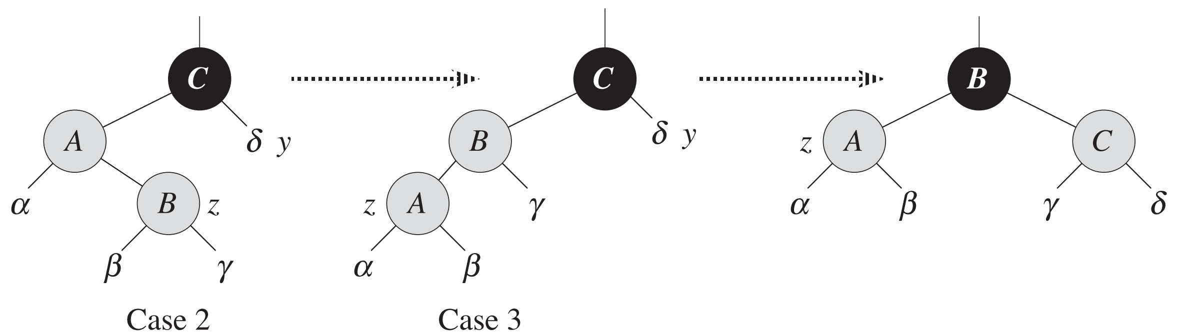


Figura: Caso 2:El tío y de z es un nodo de color negro y z es un hijo derecho.
Caso 3:El tío y de z es un nodo de color negro y z es un hijo izquierdo
Fuente [1]

Ejemplo completo de inserción

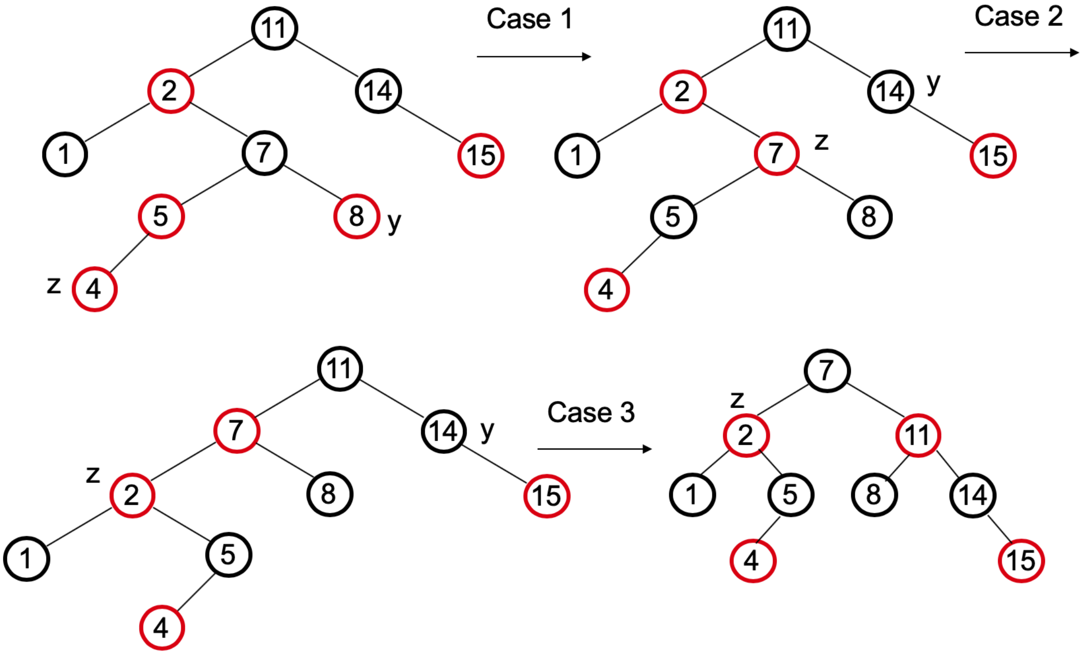


Figura: Insertando un nodo con clave igual a 4. Fuente [1]

Análisis de las operaciones árboles rojo-negro

- Los árboles rojo-negro garantizan que dado n nodos, entonces la altura h del árbol es $O(\log n)$

Teorema

El tiempo de las operaciones de BÚSQUEDA, MÍNIMO, MÁXIMO, SUCESOR, PREDECESOR e INSERCIÓN en los árboles rojo-negro son $O(h)$

Referencias

T. Cormen, C. Leirserson, R. Rivest, and C. Stein.
Introduction to Algorithms.
McGraw Hill, 3ra edition, 2009.