Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2133 — Estructuras de datos y algoritmos — 2020-2

Ayudantía 5

Pregunta 1

Respecto a los árboles rojo y negro:

a) Justifica que la rama más larga del árbol tiene a lo más el doble de nodos que la rama más corta. Se entiende por rama, la ruta de la raíz a una hoja.

b) Supón que insertamos un nodo x, y luego lo eliminamos inmediatamente. ¿Es el árbol resultante el mismo que el inicial? Justifica.

c) Considera un árbol rojo-negro formado mediante n inserciones. Justifica que si $n > 1$, entonces el árbol tiene al menos un nodo rojo.

Pregunta 2

Los árboles 2-3 son árboles de búsqueda en que los nodos tienen ya sea una clave y dos hijos, o bien dos claves y tres hijos; y todas las hojas del árbol (que se exceptúan de la regla anterior porque no tienen hijos) están a la misma profundidad. Teniendo presente estas propiedades, responde:

a) ¿Cuál es la altura máxima que puede tener un árbol 2-3 con n elementos? ¿Y la mínima? ¿Cómo es la estructura del árbol cuando ocurre cada uno de estos casos?

b) Queremos insertar una clave x en un árbol 2-3 T de altura h, que tiene n claves. ¿Qué debe cumplirse para que esta inserción aumente la altura de T? ¿Para qué valores de n está garantizado que **sí** aumentará la altura? ¿Para qué valores de n está garantizado que **no** aumentará la altura?

c) Determina un orden en que hay que insertar las claves 1, 3, 5, 8, 13, 18, 19 y 24 en un árbol 2-3 inicialmente vacío para que el resultado sea un árbol de altura 1, es decir, una raíz y sus hijos.

Pregunta 3: Propuesta

Considere un árbol rojo negro que corresponda a un árbol 2-3, escriba un algoritmo que siga el procedimiento de inserción en el árbol rojo negro de manera que corresponda al procedimiento que seguiría en en el árbol 2-3.

Solución

Pregunta 1

a)

En primer lugar, por la propiedad 4 de ARN vista en clases ("La cantidad de nodos negros camino a cada hoja debe ser la misma") podemos afirmar que la rama más corta tiene la misma cantidad de nodos negros que la más larga. Llamaremos a esta cantidad "**n**". La rama más corta posible tendrá solo los nodos negros (será de n nodos).

De esta forma, la rama más larga será de $n + r$ nodos.

Tomando en cuenta la propiedad 2 y 3 vista en clases ("La raíz del árbol es negraz "Si un nodo es rojo, sus hijos deben ser negros", respectivamente). Y además si consideramos las hojas nulas como negros, sabemos que la cantidad de nodos rojos es menor a la de negros, de hecho, por cada rojo existirá su hijo negro ($r <= n - 1$ más la raíz $\Rightarrow r <= n$).

Dado esto, la rama más larga posible tendrá $n + r$ nodos, donde $r = n$. Por lo tanto tendrá $2n$ nodos en total, el doble de los n posibles en la rama más corta. Entonces, podemos afirmar que la rama más larga de un árbol RN tiene a los más el doble de nodos que la más corta.

b)

No necesariamente, esto lo podemos demostrar con un contra ejemplo,

Si insertamos las claves 4, 7, 10, 23, 5 tenemos un árbol de la siguiente manera,

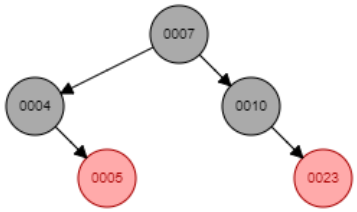
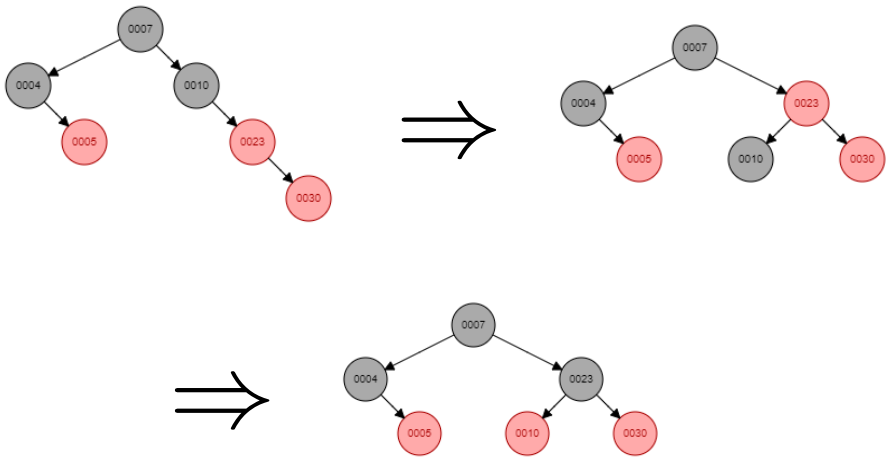


Figura 1: Árbol Rojo y Negro.

Ahora bien, si quisieramos insertar algún nodo mayor a 5 o mayor a 23 tendríamos que realizar una rotación y cambios de colores para seguir cumpliendo las propiedades de los ARN. (Recordar que al insertar un nodo siempre es de color rojo al inicio.)

Para seguir con el ejemplo, insertaremos el 30,



Ahora bien, eliminaremos el nodo recién insertado, lo cual no va a producir ninguna rotación o cambio de color, ya que sin este el árbol sigue cumpliendo todas las propiedades,

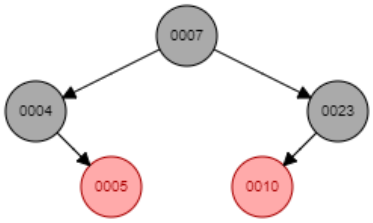


Figura 2: Árbol Rojo y Negro final.

Al comparar las figuras 1 y 2 podemos ver claramente que no son el mismo árbol. Esto sucede básicamente porque si es que no se cumplea alguna regla de los RN al insertar un nodo, entonces se debe reorganizar el mismo de tal modo que cambie su estructura para así cumplir las propiedades.

c)

Para responder esta pregunta notamos que cuando se inserta un nodo a un árbol rojo negre, se inserta con color rojo y luego se le cambia el color en caso de haber un problema. Tomando en cuenta esto notemos dos casos antes de una inserción en un árbol de $n \geq 1$:

1. El árbol no tiene un nodo rojo.
2. El árbol tiene un nodo rojo.

En el primer caso, al insertar un nodo no va a haber ningún problema con las 4 propiedades, por lo que se mantendrá rojo.
En el segundo caso, al insertar un nodo rojo se pueden hacer solo cambios de color y rotaciones en caso de que se inserte debajo de un nodo rojo. En ninguno de estos dos casos eliminamos todos los nodos rojos por lo que siguen habiendo nodos rojos en el árbol final.

Pregunta 2

a)

La altura será mayor mientras menos elementos tenga el áarbol por nivel, y viceversa. Siguiendo esta lóogica la altura seráa mayor cuando todos los nodos sean nodos 2, será menor cuando todos sus nodos sean nodos 3.

Figura 3: Comparación de estructura de árboles 2-3

Viendo la figura a), podemos notar que la cantidad de elementos para el k-ésimo nivel es de 2 notar también, que la cantidad de elementos acumulados hasta el k-ésimo nivel es $2k-1$. Análogamente para b), la cantidad de elementos para el k-ésimo nivel es $2 \cdot 3^{k-1}$, y la cantidad de elementos acumulados hasta el k-ésimo nivel es 3^{k-1}. Si despejamos la altura en cada una, obtenemos para la altura mínima:

$$h = \log_3(n+1)$$

Y para altura máxima.

$$h = \log_2(n+1)$$

Cabe mencionar que estos casos son cuando el árbol esta completamente lleno. Para alcanzar un nivel k de altura, debe exceder necesariamente los $k-1$ niveles anteriores, y no superar los k niveles. Por lo tanto, la altura mínima en función de la cantidad de elementos quedaría expresada como:

$$h = \lceil \log_3(n+1) \rceil$$

En el caso de la altura máxima, como las hojas deben estar a la misma altura, al anadir un valor extra, el árbol se rebalancea, manteniendo su altura, por lo que quedaría expresada como:

$$h = \lfloor \log_2(n+1) \rfloor$$

En los casos que $n > 0$. Si $n = 0$ en realidad no existiría estructura de datos, por lo que las fóormulas pierden sentido.

b)

En primer lugar, llamaremos al "camino" de un arbol T a los nodos que hay que recorrer desde la raíz para llegar a un nodo hoja. De esta forma, para que una inserción provoque el aumento de altura de un árbol T se debe cumplir que, en el camino a la hoja donde se insertará la nueva clave, todos los nodos sean nodos 3 (es decir, que tengan 2 claves). Así, se hará un split desde la hoja donde insertamos hasta la raíz, aumentando finalmente la altura.

Ahora evaluaremos dos casos en dónde siempre aumenta la altura, y dónde nunca aumenta la altura para una altura h con n claves.

Garantía de que aumenta la altura: El arbol T debe tener solo nodos 3. Notemos que si pasa lo contrario (al menos uno no es un nodo 3), entonces podemos buscar el camino en donde podamos insertar de tal forma que este nodo pase a ser nodo 3, lo que haría que no aumente la altura. Teniendo esto, la cantidad de claves en función de la altura h es:

$$\text{claves} = \sum_{i=1}^{h} 2 \cdot (3^{i-1})$$

dado que en un nivel i hay 3^{i-1} nodos, con cada uno 2 claves. Resolviendo obtenemos.

$$\text{claves} = 3^h - 1$$

Serie geométrica:

$$\sum_{k=0}^{n} ar^k = a\frac{1-r^{n+1}}{1-r}$$

Garantía de que no aumenta la altura: El árbol T no debe tener ningún camino que tenga únicamente nodos 3. Notemos por la definición del comienzo que la condición para que aumente la altura es que agreguemos una clave en la hoja que tenga un camino con solo nodos 3, por lo que debemos aseguranos que no exista tal camino. Luego, para garantizar esto debemos contar la cantudad de claves para que no exista un árbol con dicho camino.
La menor cantidad de claves con la que podría aumentar la altura es tal que todos sean nodos 2 excepto un camino en donde sean nodos 3. Teniendo esto, si tomamos esa cantidad de claves - 1 no podremos tener un camino con nodos 3, y garantizaremos desde esa cantidad hacia abajo que no aumente la altura.

Un árbol con las características antes mencionadas tendría la siguiente estructura.

- nivel 1 $\rightarrow$ 2 claves (nodo 3)
- nivel 2 $\rightarrow$ 4 claves (nodo 3 + 2 nodos 2)
- nivel 3 $\rightarrow$ 8 claves (nodo 3 + 6 nodos 2)

  $\vdots$
- nivel h $\rightarrow$ 2^h claves (nodo 3 + 2^h - 2 nodos 2)

Si sumamos obtenemos:

$$\text{claves} = \sum_{i=1}^{h} 2^i - 1$$

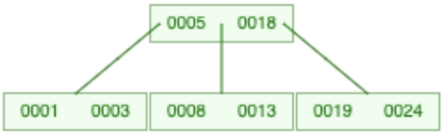
Resolviendo obtenemos:

$$\text{claves} = 2^i - 2$$

Luegos, para garantizar que la inserción en el árbol T no tendrá efectos en la altura h, la cantidad de claves debe ser menos a $2^{h+1} - 2$. Además, para la factibilidad de un árbol 2-3, se debe cumplir que las claves sean mayores a $2^h - 1$. que es la cantidad mínima de claves que un árbol 2-3 de altura h puede tener.

c)

Para esta pregunta notamos que hay solo una configuración final para el árbol 2-3, entonces podemos intentar llegar a esta configuración con una secuencia de pasos. La unica configuración posible para este árbol es:



Entonces, intentemos crear los 4 nodos. Para esto primero insertemos 1, 5 y 8:

Y esto se convierte a:

Ahora podemos intentar crear el tercer nodo, para esto queremos que el nodo de la derecha se separe en 3, y que el elemento 18 suba. Para eso insertemos el nodo 18 y 19. Esto resulta en:

Y se convierte en:

Luego, solo falta agregar los nodos que faltan en cualquier orden: 3, 13 y 24 para llegar a la solución