Árvores Rubro-Negras

Iarley Souza

April 2023

1 Questão 1

Para encontrarmos o maior número de nós internos em uma árvore Rubro-Negra com altura negra k, devemos levar em consideração a propriedade que mantém o mesmo número de nós pretos em qualquer caminho de um nó p até suas folhas descendentes. A fim de obter o máximo de nós internos dentro do intervalo de k, devemos optar pelos maiores caminhos possíveis entre p e suas folhas, isto é, devemos optar por camimhos intercalados por nós pretos e vermelhos. Como cada nó vermelho obrigatoriamente deve possuir 2 filhos pretos, e como o resultado será uma árvore completa com todas suas folhas no nível mais profundo, obtemos 2^{2k}. Subtraindo do nó raiz obtemos:

$$2^{2k} - 1$$

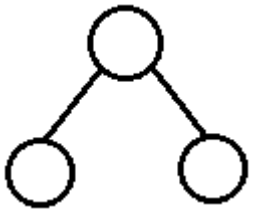
Para encontrarmos o mínimo de nós possível, consideramos as mesmas condições anteriores, com a exceção de que teremos somente nós pretos, a fim de obter os menores caminhos. Por este motivo, não precisamos multiplicar k por dois. Com isso temos:

$$2^k - 1$$

2 Questão 2

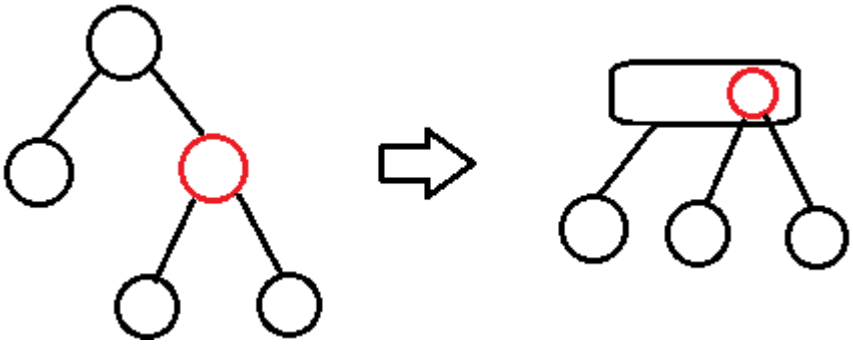
Tomando por base a definição de grau de um nó como sendo o número de filhos que o mesmo possui, devemos considerar os seguintes casos para um nó v preto:

- Caso 1: v não possui nenhum filho vermelho.



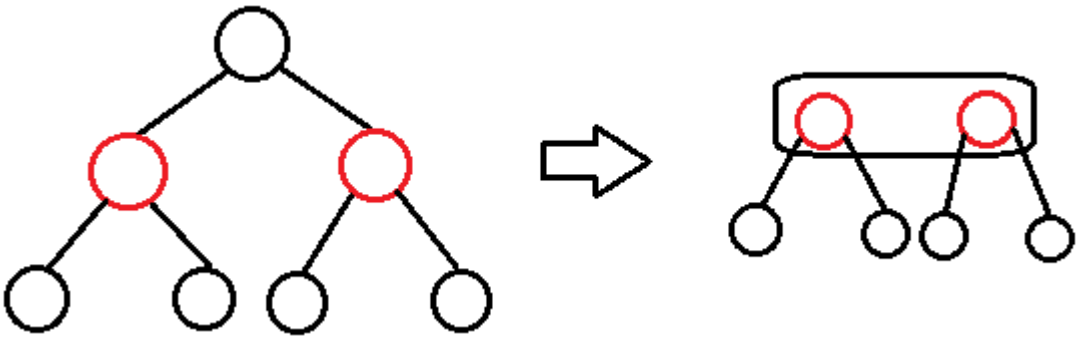
Neste caso, v não há nenhum nó vermelho para absorver, portanto não herdará nenhum nó a mais como filho, logo, $grau(v) = 2$.

- Caso 2: v possui apenas um filho vermelho.



v absorve seu único filho vermelho e herda seus dois netos pretos. Como v possui um outro filho preto, v torna-se $grau(v) = 3$.

- Caso 3: Os dois filhos de v são vermelhos.



Neste último caso, v absorve seus dois filhos vermelhos. Como em uma rubro-negra, todo nó vermelho deve possuir obrigatoriamente dois filhos pretos, v passa a ter $grau(v) = 4$.

Quanto a profundidade das folhas, as únicas que terão suas profundidades alteradas serão aquelas descendentes do nó vermelho absorvido. Isso ocorre pelo fato da absorção dos nós vermelhos "encurtar" os diferentes caminhos que passam por estes. Neste caso, a profundidade das folhas afetadas pode diminuir uma unidade em até n vezes, sendo n o número de nós vermelhos no caminho da folha até a raíz.

3 Questão 3

A ideia consiste em assegurar todas as cinco propriedades de uma árvore Rubro-Negra:

1. Todo nó da árvore é vermelho ou preto.
2. A raíz é preta.
3. Toda folha é preta.
4. Se um nó é vermelho, então ambos os filhos são pretos.
5. Para todo nó, todos os caminhos do nó até as folhas descendentes contém o mesmo número de nós pretos.

Para isso, devemos considerar os seguintes casos ao analisar um nó p qualquer:

- Caso 1: Se p é a raíz, deve ser pintado de preto (propriedade 2).
- Caso 2: Se $height(p)$ for ímpar ou, $height(p) < height(v)$, sendo v seu irmão, nulo ou não, p é pintado de preto (propriedades 3 e 4).
- Caso contrário: Pintamos p de vermelho.
- Consideramos atribuir aos nós somente as cores preto e vermelho (propriedade 1).

O Caso 2 garante que nenhuma folha seja vermelha, pois toda folha possui $h = 1$, ímpar. Ele também assegura que todo nó a ser pintado de vermelho possua h par, garantindo que seus filhos sejam pretos, pois a altura deles será $h - 1$.

Por fim, as chamadas recursivas para estes casos de teste nos nós garantirá a última propriedade (propriedade 5).

Para realizar esta tarefa utilizaremos 3 funções:

- *paint_it_black*, que pinta um nó de preto e chama outra função para pintar seus filhos.
- *paint_it_red*, que pinta um nó de vermelho e chama outra função para pintar seus filhos.
- *paint_children*, recebe os filhos de um nó e decide a cor que estes terão.

Aqui está a implementação:

AVL - Rubro-Negra

```
void paint_it_black(Node* node) {
    node->color = 0; // 0 representa a cor preta
    if (node->height != 1) { // node é um nó interno, não uma folha
        paint_children(node->left, node->right);
    }
}

void paint_it_red(Node* node) {
    node->color = 1; // 1 representa a cor vermelha
    paint_children(node->left, node->right);
}

void paint_children(Node* a, Node* b) {
    if (a != nullptr) {
        if (height(a) < height(b) || height(a) % 2 == 1) {
            paint_it_black(a);
        } else {
            paint_it_red(a);
        }
    }
    if (b != nullptr) {
        if (height(b) < height(a) || height(b) % 2 == 1) {
            paint_it_black(b);
        } else {
            paint_it_red(b);
        }
    }
}
```

Note que para garantir que o nó raíz seja pintado de preto, devemos começar a transformação utilizando a função *paint_it_black*, passando *root*.