Atividade Avaliativa 3

Cauã Borges (834437)

29 de junho de 2025

1 Questão 1: O Enigma da Árvore Proibida (Árvore Rubro-Negra)

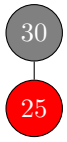
Parte A: A Árvore Esquecida

Construção da Árvore Rubro-Negra inserindo os valores na ordem dada: 30, 25, 28, 35, 32, 40, 45, 50, 55, 60. As regras de balanceamento foram seguidas, e cada passo é explicado com as operações de recoloração e rotação necessárias.

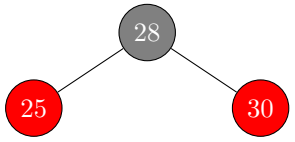
Passo a passo da construção

1. **Inserir 30:** O nó 30 é inserido como raiz. Por ser a raiz, ele é colorido de PRETO.

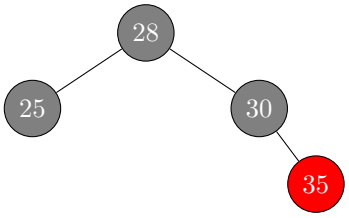
2. **Inserir 25:** O nó 25 é inserido à esquerda de 30 e colorido de VERMELHO.



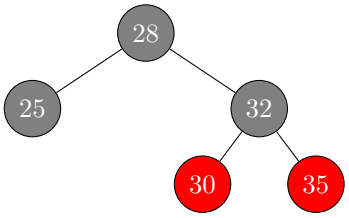
3. **Inserir 28:** O nó 28 é inserido à direita de 25 e colorido de VERMELHO. O pai (25) é VERMELHO e o avô (30) é PRETO. O tio (nulo) é PRETO. Caso 3: Rotação Dupla à Esquerda-Direita e Recoloração.
   - Rotação à Esquerda em 25 (28 se torna filho esquerdo de 30).
   - Rotação à Direita em 30 (28 se torna a nova raiz).
   - Recoloração: 28 (PRETO), 25 (VERMELHO), 30 (VERMELHO).



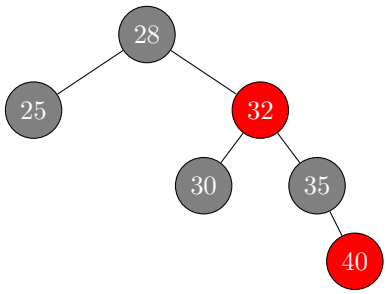
4. **Inserir 35:** O nó 35 é inserido à direita de 30 e colorido de VERMELHO. O pai (30) é VERMELHO e o avô (28) é PRETO. O tio (25) é VERMELHO. Caso 1: Recoloração.
    - Recoloração: 28 (VERMELHO), 25 (PRETO), 30 (PRETO). A raiz (28) é recolorida para PRETO.



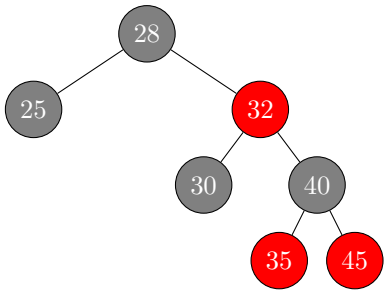
5. **Inserir 32:** O nó 32 é inserido à esquerda de 35 e colorido de VERMELHO. O pai (35) é VERMELHO e o avô (30) é PRETO. O tio (nulo) é PRETO. Caso 3: Rotação Dupla à Direita-Esquerda e Recoloração.
    - Rotação à Direita em 35 (32 se torna filho direito de 30).
    - Rotação à Esquerda em 30 (32 se torna filho direito de 28).
    - Recoloração: 32 (PRETO), 30 (VERMELHO), 35 (VERMELHO).



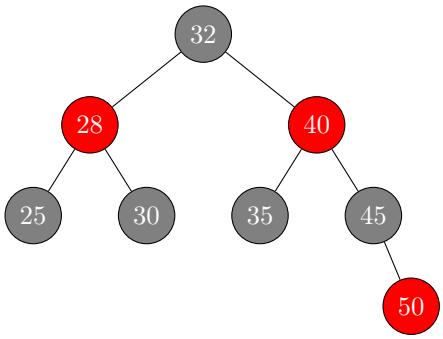
6. **Inserir 40:** O nó 40 é inserido à direita de 35 e colorido de VERMELHO. O pai (35) é VERMELHO e o avô (32) é PRETO. O tio (30) é VERMELHO. Caso 1: Recoloração.
    - Recoloração: 32 (VERMELHO), 30 (PRETO), 35 (PRETO). O nó 32 é filho VERMELHO de 28 (PRETO). Não há violação.



7. **Inserir 45:** O nó 45 é inserido à direita de 40 e colorido de VERMELHO. O pai (40) é VERMELHO e o avô (35) é PRETO. O tio (nulo) é PRETO. Caso 3: Rotação à Esquerda e Recolocação.
   - Rotação à Esquerda em 35 (40 se torna filho direito de 32).
   - Recoloração: 40 (PRETO), 35 (VERMELHO), 45 (VERMELHO).

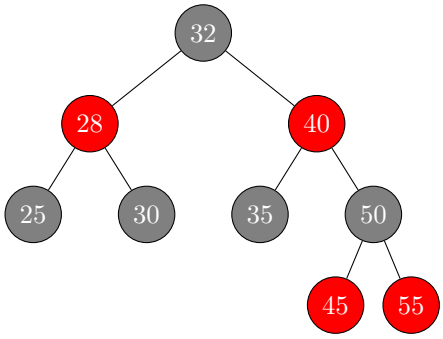


8. **Inserir 50:** O nó 50 é inserido à direita de 45 e colorido de VERMELHO. O pai (45) é vermelho,tio é vermelho.
   - Recoloração: 40(vermelho), 35 e 45 (pretos)
   - 40 e 32 ambos vermelhos
   - Rotação à Esquerda (28 se torna filho a esquerda de 32 e 40 filho à direita)
   - Recoloração: 28(vermelho), 32(preto)



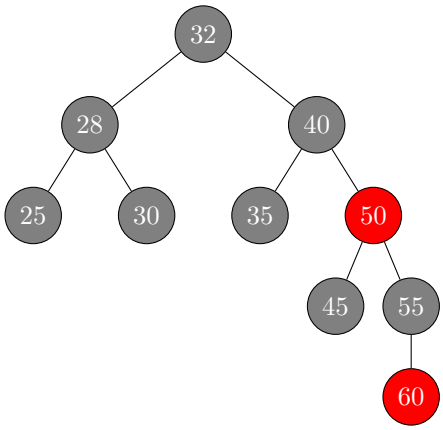
9. **Inserir 55:** O nó 55 é inserido à direita de 50 e colorido de VERMELHO. O pai (50) é VERMELHO e o avô (45) é PRETO.

   - Rotação à Esquerda em 45(50 vira filho de 40, 45 filho de 50 à esquerda e 55 filho à direita)
   - Recoloração: 50(preto), 45(vermelho)

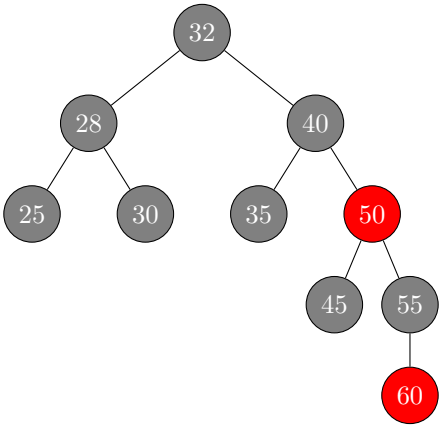


10. **Inserir 60:** O nó 60 é inserido à direita de 55 e colorido de VERMELHO. O pai (55) é VERMELHO e o avô (50) é PRETO. O tio (45) é VERMELHO.

    - Recoloração: 45(preto), 55(preto), 50(vermelho)
    - 50 e 40 vermelhos, 28 vermelho
    - Recoloração: 28(preto), 40(preto), 32(vermelho)
    - Raiz vermelha.
    - Recoloração: 32(preto).



Configuração Final da RB-tree (Parte A)

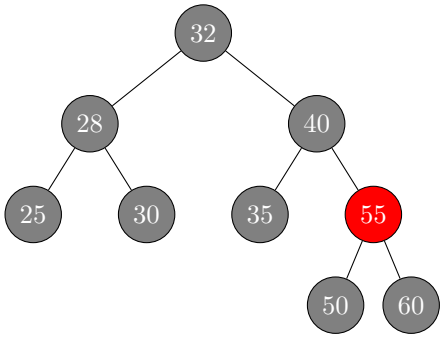


Parte B: O Exílio do Traidor

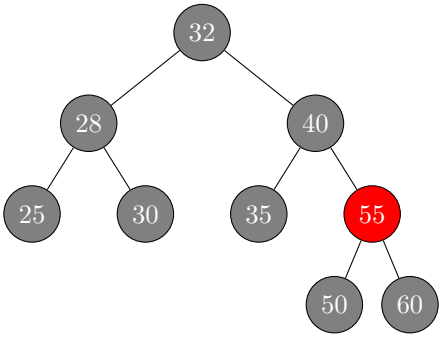
Remoção do nó 45 da Árvore Rubro-Negra.

Passo a passo da remoção do nó 45

1. **Remover 45:** O nó 45 é um nó PRETO sem filhos. Remover 45 e colocar null leaf, essa folha tem irmão preto e sobrinho vermelho, rotação à esquerda colocando 55 como filho de 40, 50 como filho à esquerda de 55 e 60 filho à direita. Remover null leaf e recoloração 55(vermelho), 50 e 60 (pretos)



Configuração Final da RB-tree (Parte B)



2 Questão 2: Sistema de Busca em Biblioteca com Skip List

Parte A e B: Construção da Skip List e Operações

https://onlinegdb.com/YpxuRPa7X

Parte C: Análise de Complexidade

Para mostrar matematicamente que a complexidade de tempo da busca em Skip Lists é $O(\log n)$ e que a complexidade de espaço é $O(n)$, assumimos que $d = 2$ (probabilidade $p = 0.5$ de um nó ser promovido para o próximo nível).

Complexidade de Tempo ($O(\log n)$)

A busca em uma Skip List começa no nível mais alto e desce para níveis inferiores conforme necessário. Em cada nível, a busca avança horizontalmente até encontrar um nó maior ou o fim do nível. A altura esperada de uma Skip List com n elementos é $O(\log n)$. Isso ocorre porque, em média, metade dos nós em um nível são promovidos para o nível acima. Assim, o número de nós em cada nível diminui exponencialmente. A busca em cada nível é linear no número de nós visitados nesse nível. Como o número total de nós visitados em todos os níveis é proporcional à altura da Skip List, a complexidade de tempo esperada para busca (e também para inserção e remoção) é $O(\log n)$.

Formalmente, se temos n elementos, o número de níveis L é aproximadamente $\log_{1/p} n$. Para $p = 0.5$, $L \approx \log_2 n$. Em cada nível, o número esperado de comparações é uma constante. Portanto, o tempo total de busca é $O(L) = O(\log n)$.

Complexidade de Espaço ($O(n)$)

Para a complexidade de espaço, consideramos o número total de ponteiros na Skip List. No nível 0, temos n nós, cada um com um ponteiro. No nível 1,

esperamos ter $n \cdot p$ nós. No nível 2, $n \cdot p^2$ nós, e assim por diante. A soma total de nós em todos os níveis é uma série geométrica:

$$N_{total} = n + np + np^2 + \cdots + np^{L-1} = n \sum_{i=0}^{L-1} p^i$$

Para $p < 1$, esta soma converge para $n \cdot \frac{1}{1-p}$. Para $p = 0.5$, a soma é $n \cdot \frac{1}{1-0.5} = n \cdot 2 = 2n$. Isso significa que o número total de nós (e, consequentemente, o número total de ponteiros) é proporcional a n. Portanto, a complexidade de espaço esperada da Skip List é $O(n)$.

3 Questão 3: O Cofre dos Códigos Perdidos (Tabela Hash)

Parte A, B e C: Construção da Tabela Hash (Sondagem Linear)

https://onlinegdb.com/gyeAF5spyf

O código Python para a Tabela Hash com sondagem linear é apresentado abaixo. A função de espalhamento utilizada é $h(k) = k \pmod{M}$, com $M = 17$. Os valores inseridos são: 25, 47, 98, 13, 52, 75, 67, 32, 81, 11, 89, 55, 29, 39, 42.

O número total de colisões na Tabela Hash redimensionada foi de 5. Isso demonstra a eficácia do redimensionamento na redução de colisões, pois a taxa de ocupação é reduzida e os elementos são redistribuídos.

Parte D: Garantias teóricas (resolução manual)

Para mostrar matematicamente que o número esperado de sondagens em uma busca terminada com sucesso em uma Tabela Hash de endereçamento aberto é no máximo $\frac{1}{\alpha} \log \frac{1}{1-\alpha}$, assumindo hashing simples uniforme, onde α denota a taxa de ocupação. A busca pela chave k é equivalente à sequência de sondagens usada na inserção da chave k.

Se k é a $(i+1)$-ésima chave a ser inserida em T, então o fator de carga nesse momento é:

$$\alpha = \frac{i}{m}$$

o que implica que o número esperado de sondagens é no máximo:

$$E[X] \leq \frac{1}{1-\alpha} = \frac{1}{1-\frac{i}{m}} = \frac{m}{m-i}$$

Tomando a média sobre todas as n chaves da tabela:

$$\frac{1}{n} \sum_{i=0}^{n-1} \frac{m}{m-i} = \frac{m}{n} \sum_{i=0}^{n-1} \frac{1}{m-i} = \frac{1}{\alpha} \sum_{i=0}^{n-1} \frac{1}{m-i}$$

Por uma substituição de variáveis, seja $k = m - i$. Assim, temos:

$i = 0 \rightarrow k = m$

$i = n - 1 \rightarrow k = m - (n - 1)$

Dessa forma, o somatório pode ser expresso como:

$$\frac{1}{\alpha n} \sum_{k=m-(n+1)}^{m} \frac{1}{k}$$

Pode-se mostrar que para funções monotonicamente decrescentes, temos:

$$\sum_{k=m}^{n} f(k) \leq \int_{m-1}^{n} f(x)\, dx$$

Logo, podemos escrever:

$$\frac{1}{\alpha} \sum_{k=m-(n+1)}^{m} \frac{1}{k} \leq \frac{1}{\alpha} \int_{m-n}^{m} \frac{1}{x}\, dx = \frac{1}{\alpha} \ln x|_{m-n}^{m} = \frac{1}{\alpha}[\ln m - \ln m - n] = \frac{1}{\alpha} \ln \frac{m}{m-n}$$

Dividindo o número rador e o denominador do logaritmo por m, finalmente chegamos em:

$$\frac{1}{\alpha} \ln \frac{1}{1-\alpha}$$

4 Questão 4: Sobre o Filtro de Bloom

Parte A: Número ótimo k de funções hash

Mostre matematicamente que o número ótimo k de funções hash é dado por:

$$k = \frac{m}{n} \ln 2$$

onde n denota o número de chaves e m denota o número de slots do filtro (tamanho).

Demonstração:

A probabilidade de um bit permanecer 0 após a inserção de uma chave é $(1 - \frac{1}{m})^k$. Após a inserção de n chaves, a probabilidade de um bit permanecer 0 é $(1 - \frac{1}{m})^{kn}$.

A probabilidade de um falso positivo (ϵ) é a probabilidade de que um elemento que não está no conjunto seja testado como presente. Isso ocorre se todos os k bits correspondentes a esse elemento forem 1. A probabilidade de um bit ser 1 é $1 - (1 - \frac{1}{m})^{kn}$. Assim, a probabilidade de falso positivo é:

$$\epsilon \approx (1 - e^{-kn/m})^k$$

Para minimizar ϵ, tomamos a derivada de $\ln \epsilon$ em relação a k e igualamos a zero. Simplificando a expressão para ϵ e usando a aproximação $1 - x \approx e^{-x}$ para pequenos x, temos:

$$\epsilon \approx (1 - e^{-kn/m})^k$$

Para encontrar o k ótimo, minimizamos essa expressão. O valor de k que minimiza ϵ é obtido quando $e^{-kn/m} = 1/2$. Resolvendo para k:

$$-\frac{kn}{m} = \ln(1/2)$$
$$-\frac{kn}{m} = -\ln 2$$
$$k = \frac{m}{n} \ln 2$$

Parte B: Tamanho ótimo m do filtro de Bloom

Mostre matematicamente que o tamanho ótimo m do filtro de Bloom é dado por:

$$m = -\frac{n \ln \epsilon}{(\ln 2)^2}$$

onde ϵ denota a probabilidade de falso positivo e n denota o número de chaves.

Demonstração:

Partindo da fórmula da probabilidade de falso positivo:

$$\epsilon \approx (1 - e^{-kn/m})^k$$

Substituindo o k ótimo encontrado na Parte A ($k = \frac{m}{n} \ln 2$):

$$\epsilon \approx (1 - e^{-(\frac{m}{n} \ln 2)n/m})^k$$
$$\epsilon \approx (1 - e^{-\ln 2})^k$$
$$\epsilon \approx (1 - 1/2)^k$$
$$\epsilon \approx (1/2)^k$$

Tomando o logaritmo natural de ambos os lados:

$$\ln \epsilon = k \ln(1/2)$$
$$\ln \epsilon = -k \ln 2$$
$$k = -\frac{\ln \epsilon}{\ln 2}$$

Agora, igualamos as duas expressões para k:

$$\frac{m}{n} \ln 2 = -\frac{\ln \epsilon}{\ln 2}$$

Resolvendo para m:

$$m = -\frac{n \ln \epsilon}{(\ln 2)^2}$$