1.Introdução

Árvores Vermelho-Preto são um tipo de árvore de busca binária balanceada que usa um conjunto de regras para manter o equilíbrio, garantindo complexidade de tempo logarítmica para operações como inserção, exclusão e busca, independentemente do formato inicial da árvore. Árvores Vermelho-Preto são autobalanceadas, usando um esquema simples de codificação de cores para ajustar a árvore após cada modificação.

Uma árvore rubro-preta é uma árvore de busca binária autobalanceada onde cada nó tem um atributo adicional: uma cor, que pode ser vermelha ou preta. O objetivo principal dessas árvores é manter o equilíbrio durante inserções e exclusões, garantindo recuperação e manipulação eficientes de dados.

Uma Árvore Rubro-Preta tem as seguintes propriedades:

- **Cor do Nó:** Cada nó é vermelho ou preto.
- **Propriedade da Raiz:** A raiz da árvore é sempre preta.
- **Propriedade Vermelha:** Nós vermelhos não podem ter filhos vermelhos (não há dois nós vermelhos consecutivos em nenhum caminho).
- **Propriedade Preta:** Cada caminho de um nó para seus nós nulos descendentes (folhas) tem o mesmo número de nós pretos.
- **Propriedade da Folha:** Todas as folhas (nós NIL) são pretas.

A maioria das operações BST (binary search tree) leva tempo O(h) onde h é a altura do BST. O custo dessas operações pode se tornar O(n) para uma árvore binária enviesada. Se garantirmos que a altura da árvore permaneça O(log n) após cada inserção e exclusão, podemos garantir um limite superior de O(log n) para todas essas operações. A altura de uma árvore Red-Black é sempre O(log n) onde n é o número de nós na árvore.

Operações	Complexidade de tempo
Busca	**O(log n)**
Inserção	**O(log n)**
Remoção	**O(log n)**

As árvores AVL são mais equilibradas em comparação com as Árvores Rubro-Pretas, mas podem causar mais rotações durante a inserção e exclusão. Então, se sua aplicação envolve inserções e exclusões frequentes, então as árvores Rubro-Pretas devem ser preferidas. E se as inserções e exclusões forem menos

frequentes e a busca for uma operação mais frequente, então a árvore AVL deve ser preferida em relação à Árvore Rubro-Preta.

Principais pontos de uma árvore vermelha-preta:

- A altura preta da árvore rubro-preta é o número de nós pretos em um caminho do nó raiz para um nó folha. Os nós folha também são contados como nós pretos. Então, uma árvore rubro-negra de altura h tem altura preta >= h/2.
- A altura de uma árvore rubro-negra com n nós é h<= 2 log2(n + 1).
- Todas as folhas (NIL) são pretas.
- A profundidade preta de um nó é definida como o número de nós pretos da raiz para aquele nó, ou seja, o número de ancestrais pretos.

1.1 Inserção em árvores vermelho-preta

Após inserir o novo nó como um nó vermelho, podemos encontrar vários casos dependendo das cores do pai e do tio do nó (o irmão do pai):

- **Caso 1 - Tio é vermelho:** Recolorir o pai e o tio para preto, e o avô para vermelho. Então, suba na árvore para verificar se há mais violações.
- **Caso 2 - Tio é preto:** O nó é um filho direito: Execute uma rotação para a esquerda no pai; O nó é um filho esquerdo: Execute uma rotação para a direita no avô e recolorir apropriadamente.

1.2 Busca em árvores vermelho-preta

A busca por um nó em uma Árvore Vermelha-Preta é similar à busca em uma Árvore de Busca Binária (BST) padrão. A operação de busca segue um caminho direto da raiz até uma folha, comparando o valor alvo com o valor do nó atual e movendo para a esquerda ou direita de acordo.

1.3 Remoção

A exclusão de um nó de uma Árvore Rubro-Preta também envolve um processo de duas etapas: executar a exclusão do BST, seguida da correção de quaisquer violações que surjam.

- **Exclusão de BST:** Remova o nó usando regras BST padrão.
- **Corrigir Double Black:** Se um nó preto for excluído, uma condição de “double black” pode surgir, o que requer correções específicas.

Quando um nó preto é excluído, lidamos com o problema do preto duplo com base na cor do irmão e nas cores de seus filhos:

- **Caso 1:** Irmão é Vermelho: Gire o pai e recolora o irmão e o pai.

- **Caso 2:** Irmão é Preto: Os filhos dos irmãos são pretos - recolorir o irmão e propagar o preto duplo para cima; Pelo menos um dos filhos dos irmãos é vermelho - Se o filho distante do irmão for vermelho, execute uma rotação no pai e no irmão e recolora apropriadamente; Se o filho próximo do irmão for vermelho - Gire o irmão e seu filho e então manipule como acima.

1.4 Rotação em árvores vermelho-preta

As rotações são operações fundamentais para manter a estrutura equilibrada de uma Árvore Rubro-Negra (RBT). Elas ajudam a preservar as propriedades da árvore, garantindo que o caminho mais longo da raiz até qualquer folha não seja mais do que o dobro do comprimento do caminho mais curto. As rotações vêm em dois tipos: rotações à esquerda e rotações à direita.

1.4.1 Rotação para a esquerda

Uma rotação à esquerda no nó *x*x move *x*x para baixo, para a esquerda, e seu filho direito *y*y para cima, para tomar o lugar de *x*x. O passo a passo é:

1. Defina y para ser o filho direito de x.
2. Mova a subárvore esquerda de y para a subárvore direita de x.
3. Atualize o pai de x e y.
4. Atualize o pai de x para apontar para y em vez de x.
5. Defina o filho esquerdo de y para x.
6. Atualize o pai de x para y.

1.4.2 Rotação para a direita

Uma rotação à direita no nó *x*x move *x*x para baixo para a direita e seu filho esquerdo *y*y para cima para tomar o lugar de *x*x. O passo a passo é:

1. Defina y para ser o filho esquerdo de x.
2. Mova a subárvore direita de y para a subárvore esquerda de x.
3. Atualize o pai de x e y.
4. Atualize o pai de x para apontar para y em vez de x.
5. Defina o filho direito de y para x.
6. Atualize o pai de x para y.

Rotações em Árvores Rubro-Negras são tipicamente realizadas durante inserções e exclusões para manter as propriedades da árvore.

Quando um novo nó é inserido, ele é sempre colorido de vermelho. Isso pode criar violações das propriedades da Árvore Vermelho-Preto, especificamente:

- A raiz deve ser preta.
- Nós vermelhos não podem ter filhos vermelhos.

Caso de inserção: Recolorindo e Propagando para Cima

Se o pai e o tio do novo nó forem ambos vermelhos, recolora o pai e o tio para preto, e o avô para vermelho. Então, recursivamente aplique a correção ao avô.

Caso de inserção: Rotação e Recoloração

- Se o tio do novo nó for preto e o novo nó for o filho direito de um filho esquerdo (ou vice-versa), execute uma rotação para mover o novo nó para cima e alinhá-lo.
- Se o tio do novo nó for preto e o novo nó for o filho esquerdo de um filho esquerdo (ou direito de um direito), execute uma rotação e recolorir o pai e o avô para corrigir a violação.

Após a exclusão, a árvore pode precisar de correção para restaurar propriedades:

- Quando um nó preto é removido, ou um nó vermelho é substituído por um nó preto, uma situação de duplo preto pode surgir.

Caso de remoção: O Irmão é vermelho

- Recolorir o irmão e o pai, e executar uma rotação

Caso de remoção: Irmão é preto com filhos pretos

- Recolorir o irmão para vermelho e mover o problema para o pai

Caso de remoção: O irmão é preto com pelo menos um filho vermelho

- Gire e recolorir para corrigir o problema do preto duplo

2.Aplicações

- **Implementando mapas e conjuntos:** Árvores Vermelhas e Pretas são frequentemente usadas para implementar mapas e conjuntos, onde busca, inserção e exclusão eficientes são cruciais.
- **Filas de prioridade:** Árvores Vermelhas e Pretas podem ser usadas para implementar filas de prioridade, onde elementos são ordenados com base em sua prioridade.
- **Sistemas de arquivos:** Árvores Vermelhas e Pretas são usadas em alguns sistemas de arquivos para gerenciar estruturas de arquivos e diretórios.
- **Bancos de dados na memória:** Árvores Vermelhas e Pretas são algumas vezes usadas em bancos de dados na memória para armazenar e recuperar dados de forma eficiente.

- **Gráficos e desenvolvimento de jogos:** Árvores Vermelhas e Pretas podem ser usadas em gráficos e desenvolvimento de jogos para tarefas como detecção de colisões e busca de caminhos.

3.Vantagens

- **Balanceado:** Árvores Rubro-Negras são autobalanceadas, o que significa que elas mantêm automaticamente um equilíbrio entre as alturas das subárvores esquerda e direita. Isso garante que as operações de busca, inserção e exclusão levem tempo O(log n) no pior caso.
- **Busca, inserção e exclusão eficientes:** Devido à sua estrutura balanceada, Árvores Rubro-Negras oferecem operações eficientes. Busca, inserção e exclusão levam tempo O(log n) no pior caso.
- **Simples de implementar:** As regras para manter as propriedades da Árvore Rubro-Negra são relativamente simples e diretas de implementar.
- **Amplamente utilizadas:** Árvores Rubro-Negras são uma escolha popular para implementar várias estruturas de dados, como mapas, conjuntos e filas de prioridade.

4.Desvantagens

- **Mais complexo do que outras árvores balanceadas:** Comparado a árvores balanceadas mais simples, como árvores AVL, as Red-Black Trees têm regras de inserção e exclusão mais complexas.
- **Sobrecarga constante:** Manter as propriedades da Red-Black Tree adiciona uma pequena sobrecarga a cada operação de inserção e exclusão.
- **Não é ideal para todos os casos de uso:** Embora eficiente para a maioria das operações, as Red-Black Trees podem não ser a melhor escolha para aplicativos em que inserções e exclusões frequentes são necessárias, pois a sobrecarga constante pode se tornar significativa.