Árvore Vermelho-Preta

Estrutura de Dados II

Jairo Francisco de Souza

Introdução

- As árvores Vermelho-preto são árvores binárias de busca
- Também conhecidas como Rubro-negras ou Red-Black Trees
- Foram inventadas por Bayer sob o nome “Árvores Binárias Simétricas” em 1972, 10 anos depois das árvores AVL

Introdução

- As árvores vermelho-preto possuem um bit extra para armazenar a cor de cada nó, que pode ser VERMELHO ou PRETO

- Além deste, cada nodo será composto ainda pelos seguintes campos:
  - valor (os “dados” do nodo)
  - fe (filho esquerdo)
  - fd (filho direito)
  - pai

Introdução

- Ela é complexa, mas tem um bom pior-caso de tempo de execução para suas operações e é eficiente na prática: pode-se buscar, inserir, e remover em tempo O(log n), onde n é o número total de elementos da árvore. De maneira simplificada, uma árvore rubro-negra é uma árvore de busca binária que insere e remove de forma inteligente, para assegurar que a árvore permaneça aproximadamente balanceada.

Introdução

- Uma árvore Vermelho-preto com n nós internos tem altura máxima de 2 log(n+1)

- Por serem “balanceadas” as árvores V-P possuem complexidade logarítmica em suas operações: O (log n)

Propriedades

1. Todo nó é vermelho ou preto
2. A raiz é preta
3. Toda folha (Nil) é preta
4. Se um nó é vermelho, então os seus filhos são pretos
5. Para cada nó, todos os caminhos do nó para folhas descendentes contém o mesmo número de nós PRETOS.

Quanto à raiz: Ser sempre preta é uma regra usada em algumas definições. Como a raiz pode sempre ser alterada de vermelho para preto, mas não sendo válido o oposto, esta regra tem pouco efeito na análise.

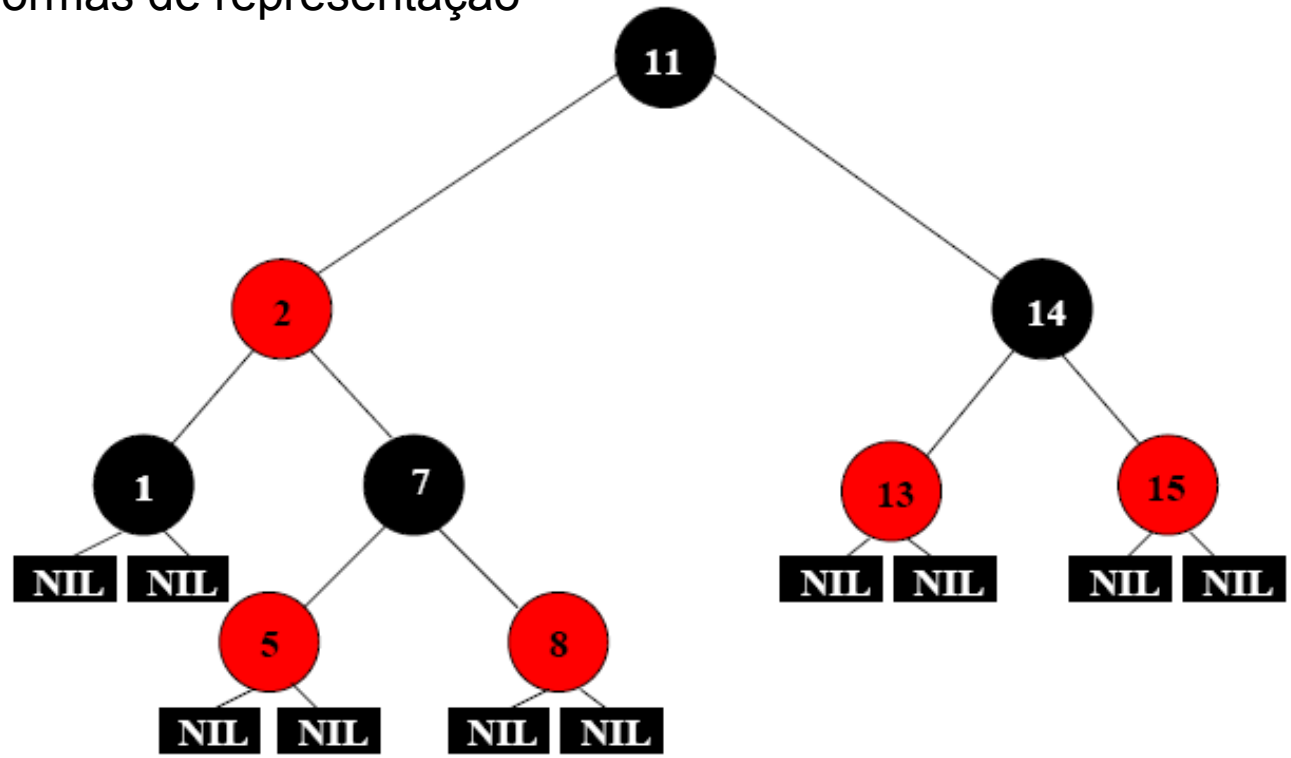
Propriedades

- Um nó que satisfaz os itens acima é denominado equilibrado, caso contrário é dito desequilibrado.
  - Em uma árvore rubro-negra todos os nós estão equilibrados

- Uma condição óbvia obtida das propriedades é que num caminho da raiz até uma sub-árvore vazia **não** pode existir dois nós rubros consecutivos

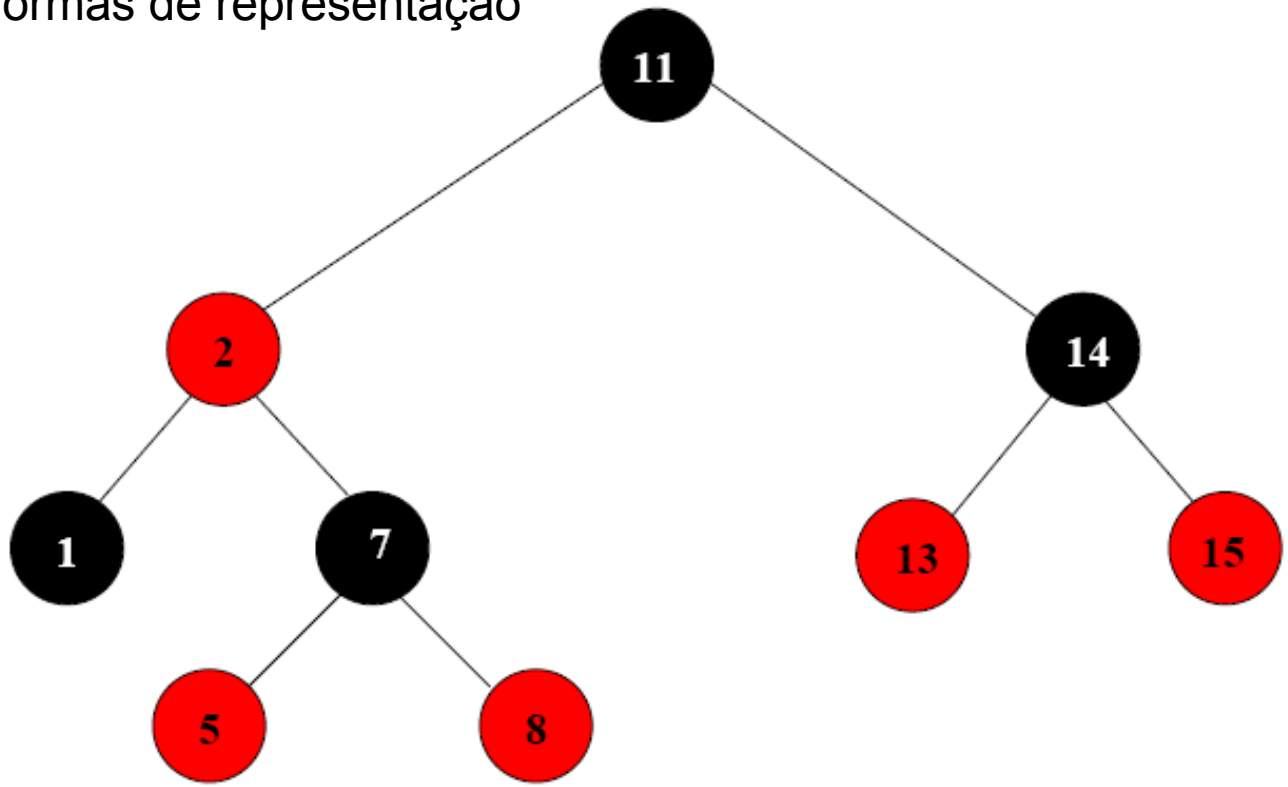
Propriedades

Formas de representação



Propriedades

Formas de representação



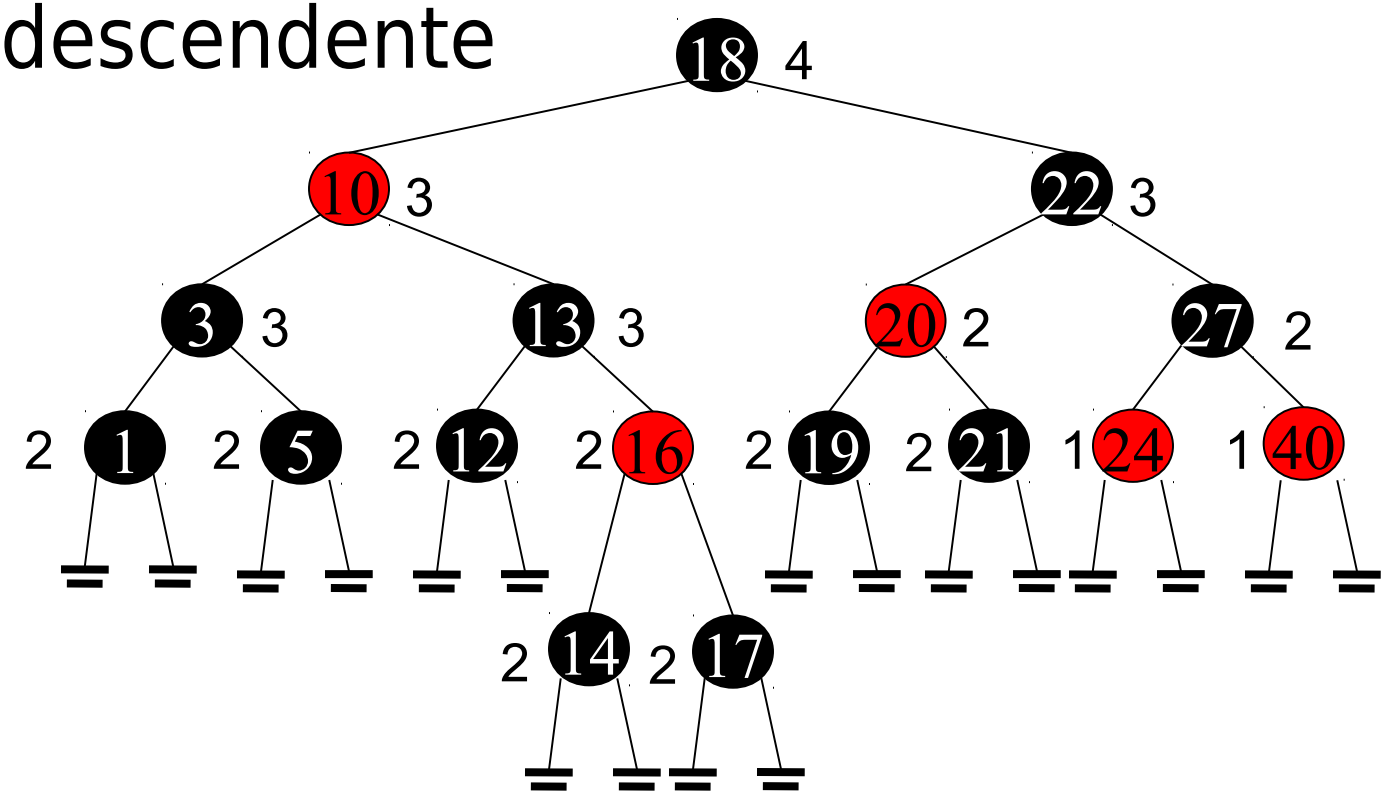
Propriedades

- Cada vez que uma operação for realizada na árvore, o conjunto de propriedades é testado

- Caso alguma não seja satisfeita, são realizadas rotações e/ou ajustes de cores, de forma que a árvore permaneça balanceada

Propriedades

- Altura negra: é número de nós negros encontrados até qualquer nó folha descendente

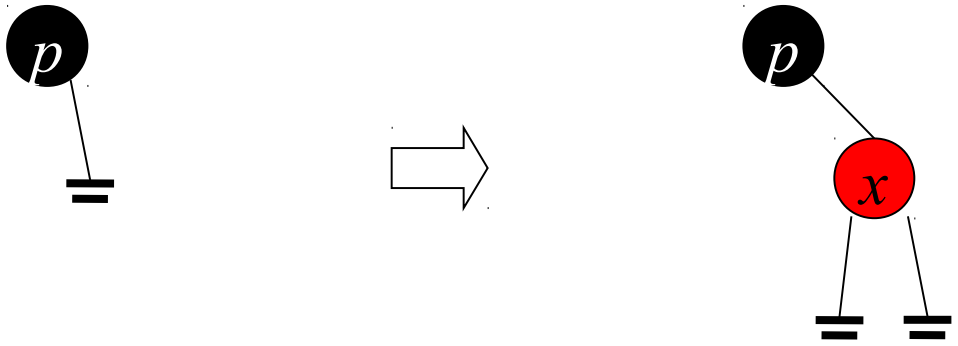


Inserção

- As operações Inserir e Remover são mais complicadas nas Árvores Rubro-Negras porque elas podem ferir alguma propriedade deste tipo de árvore.
- Como veremos, essas operações podem ser implementadas de forma bastante parecida com as respectivas operações nas Árvores Binárias de Busca, bastando apenas modificar as cores dos nós para que as propriedades de Árvores Rubro-Negras sejam satisfeitas.
- Como a inserção e a remoção propriamente ditas já foram vistas para Árvores Binárias de Busca, veremos apenas o que é necessário para acertar as cores da árvore.

Inserção

- Um nó é inserido sempre na cor vermelha, assim, não altera a altura negra da árvore
- Se o nodo fosse inserido na cor preta, invalidaria a propriedade 5, pois haveria um nodo preto a mais em um dos caminhos

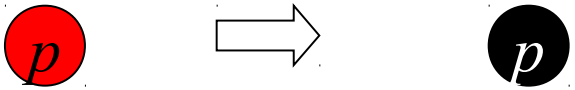


Inserção

- A operação de inserção em uma árvore rubro-negra começa por uma busca da posição onde o novo nodo deve ser inserido, partindo-se da raiz em direção aos nodos que possuam o valor mais próximo do qual vai ser inserido

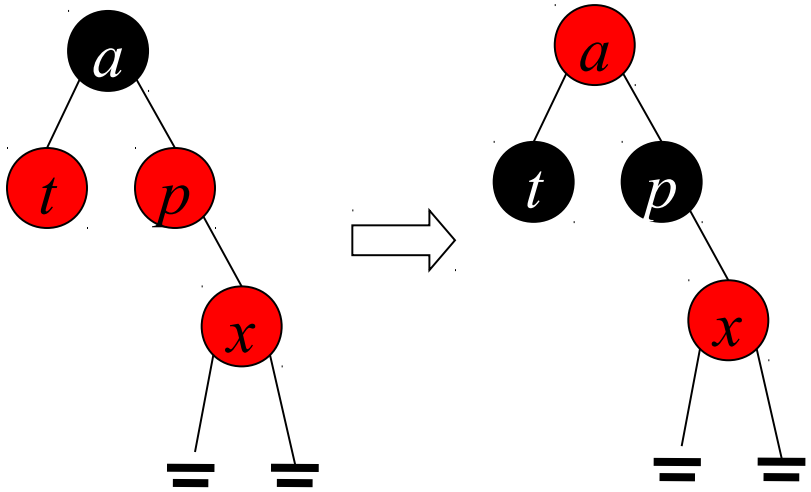
Inserção

- Caso 1: Caso esta inserção seja feita em uma árvore vazia, basta alterar a cor do nodo para preto, satisfazendo assim a propriedade 2



Exemplo

- Caso 2: Ao inserir x, se o tio de x é vermelho, é necessário fazer a recoloração de a, t e p

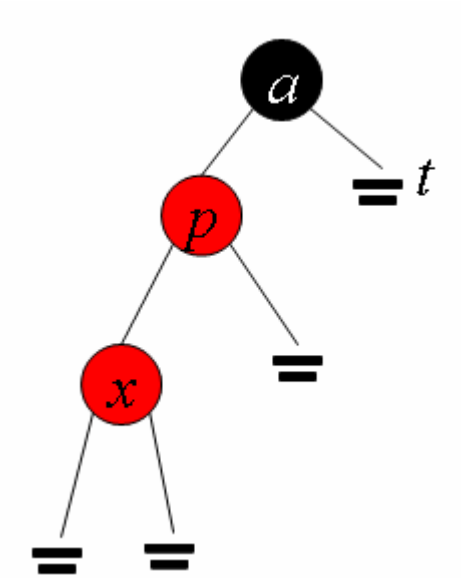


Obs1: Se o pai de *a* é vermelho, o rebalanceamento tem que ser feito novamente

Obs2: Se a é raiz, então ele deve ser preto!

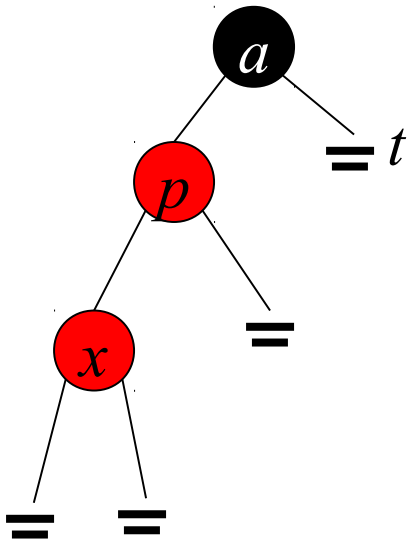
Exemplo

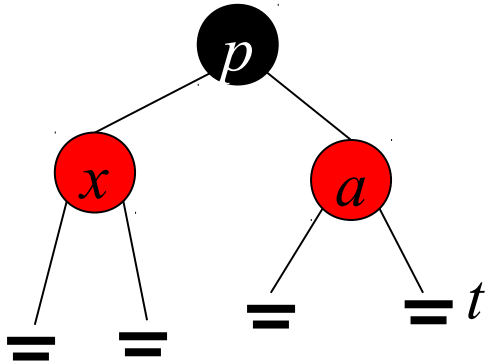
- Caso 3: Suponha que o tio do elemento inserido é preto. Neste caso, para manter o critério (4) é preciso fazer rotações envolvendo a, t, p e x.
  - Há 4 subcasos que correspondem às 4 rotações possíveis:



Exemplo

- Caso 3a: Rotação à Direita

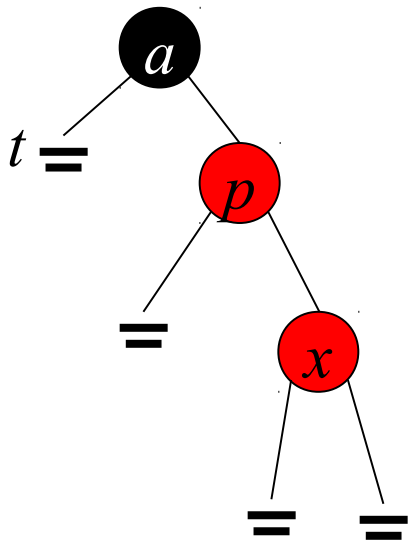


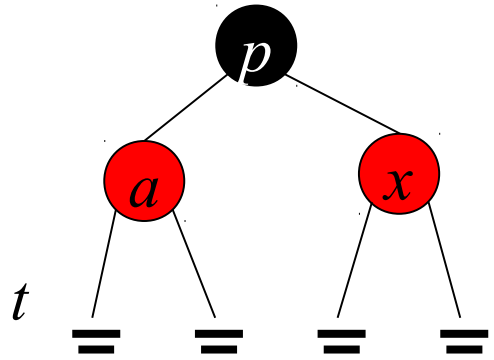


Recoloração de p e a

Exemplo

- Caso 3b: Rotação à Esquerda

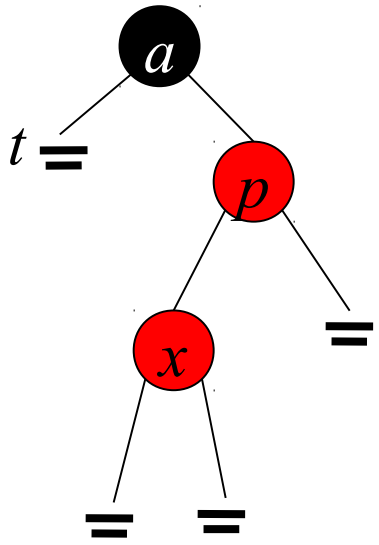


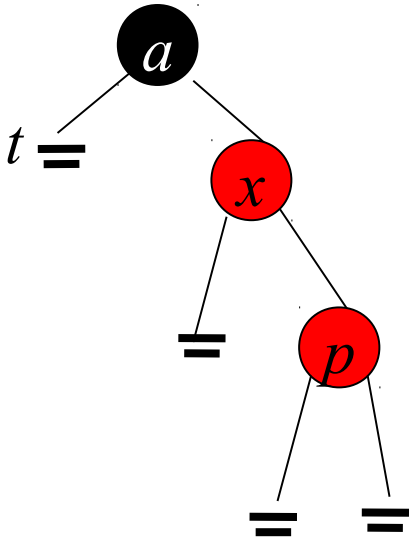


Recoloração de p e a

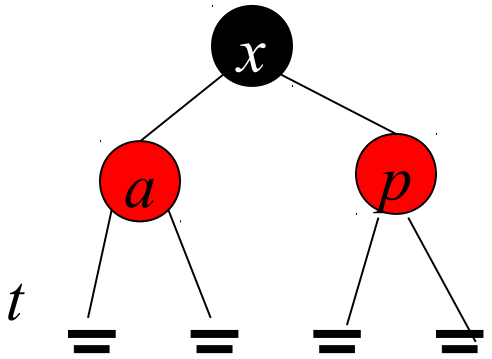
Exemplo

- Caso 3c: Rotação Dupla Esquerda
- Pode ser visto como um caso 3a seguido do caso 3b





Rotação simples à direita

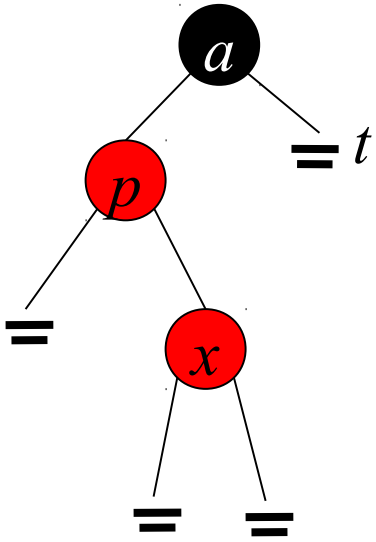


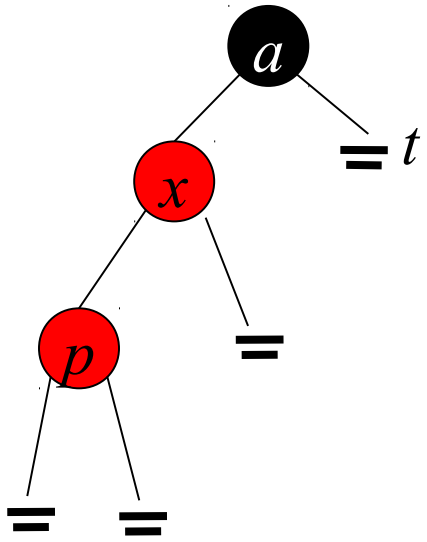
Rotação simples à esquerda

Recoloração de x e a

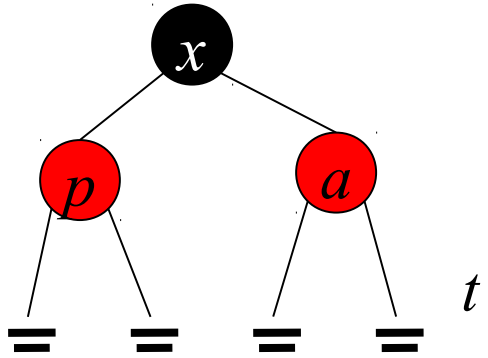
Exemplo

- Caso 3d: Rotação Dupla Direita
- Pode ser visto como um caso 3b seguido do caso 3a





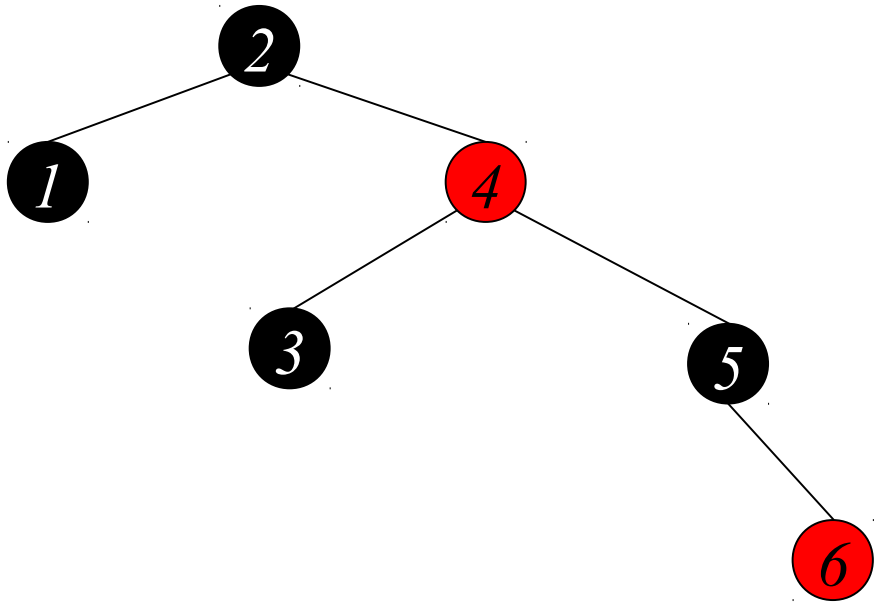
Rotação simples à esquerda



Rotação simples à direita
Recoloração de x e a

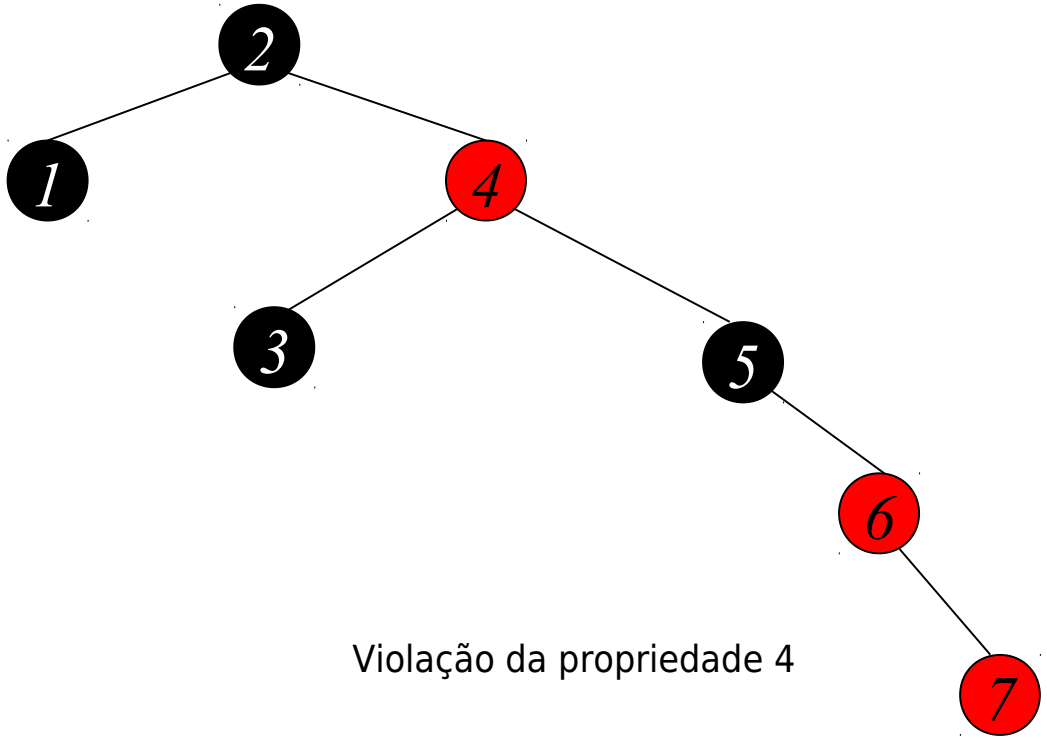
Exemplo

- Estado inicial da árvore



Exemplo

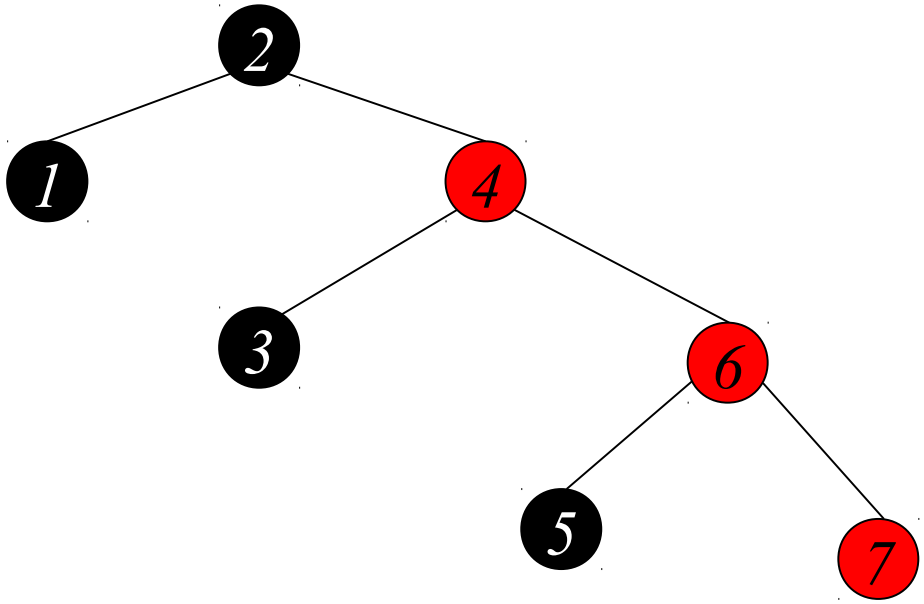
- Inserção do nodo 7



Violação da propriedade 4

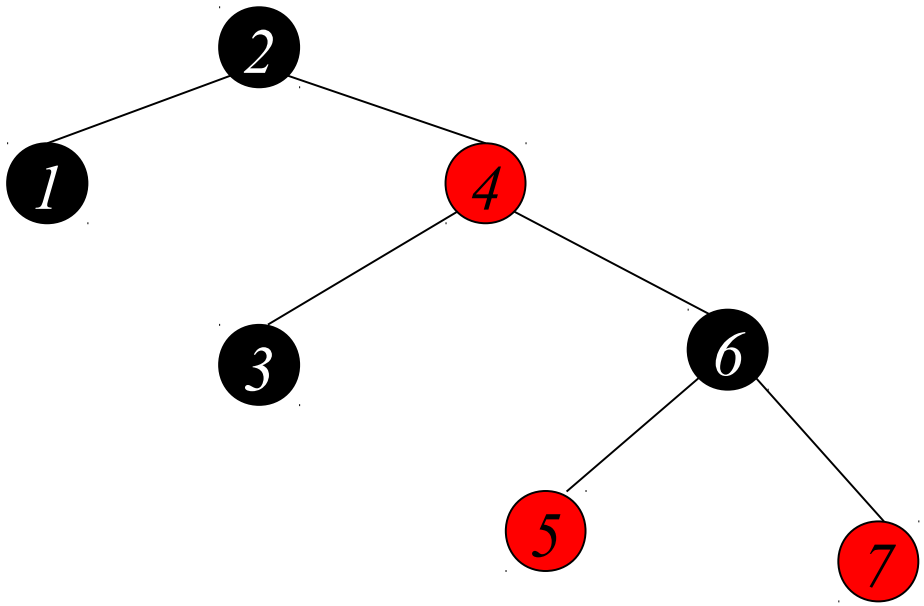
Exemplo

- Rotação à esquerda dos nodos 5,6 e 7



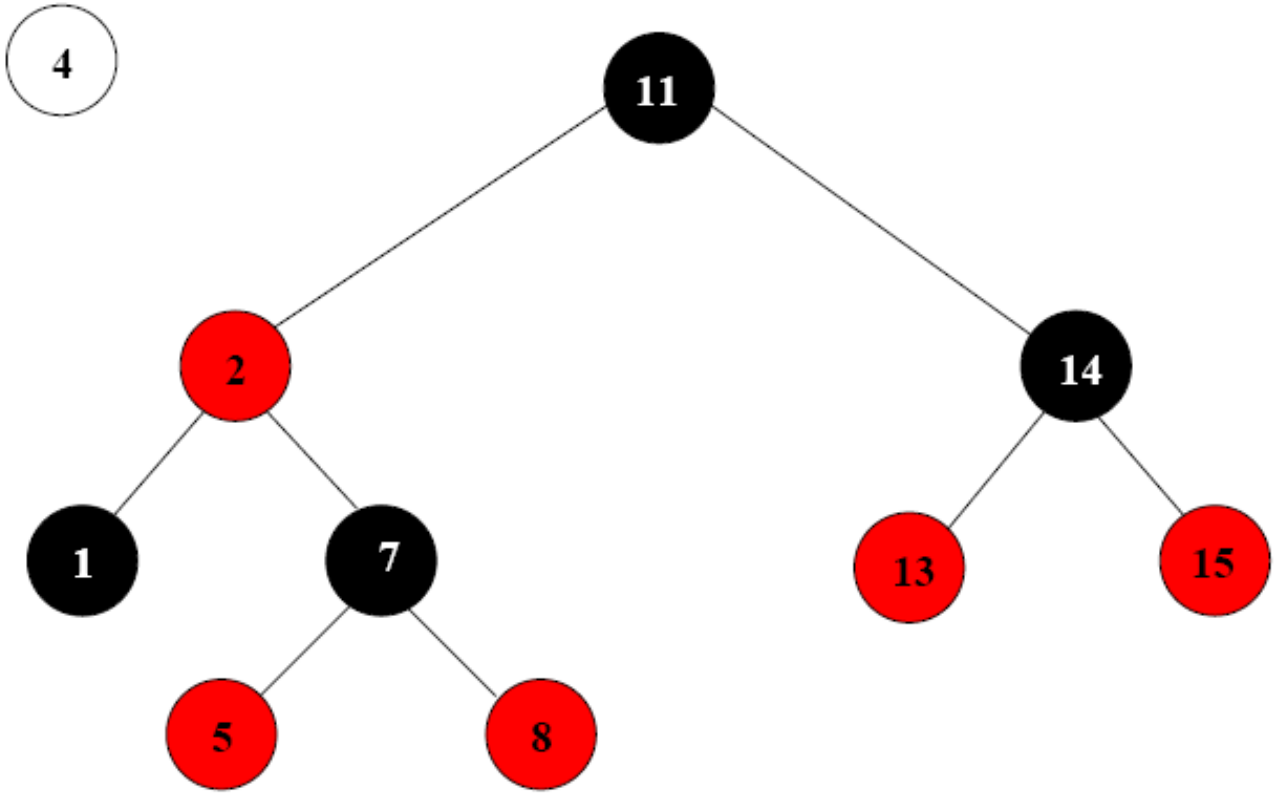
Exemplo

- Alteração de cor dos nodos 5 e 6

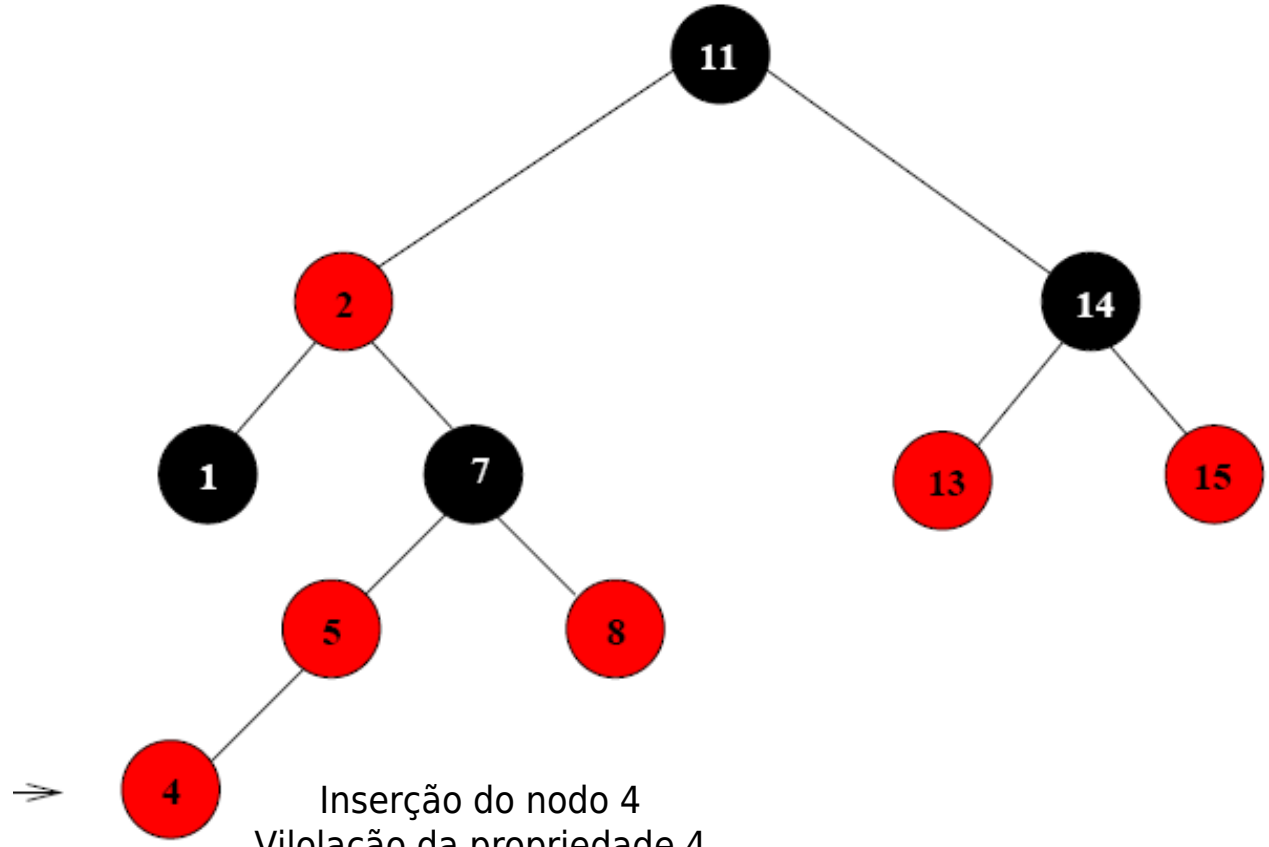


Exemplo 2

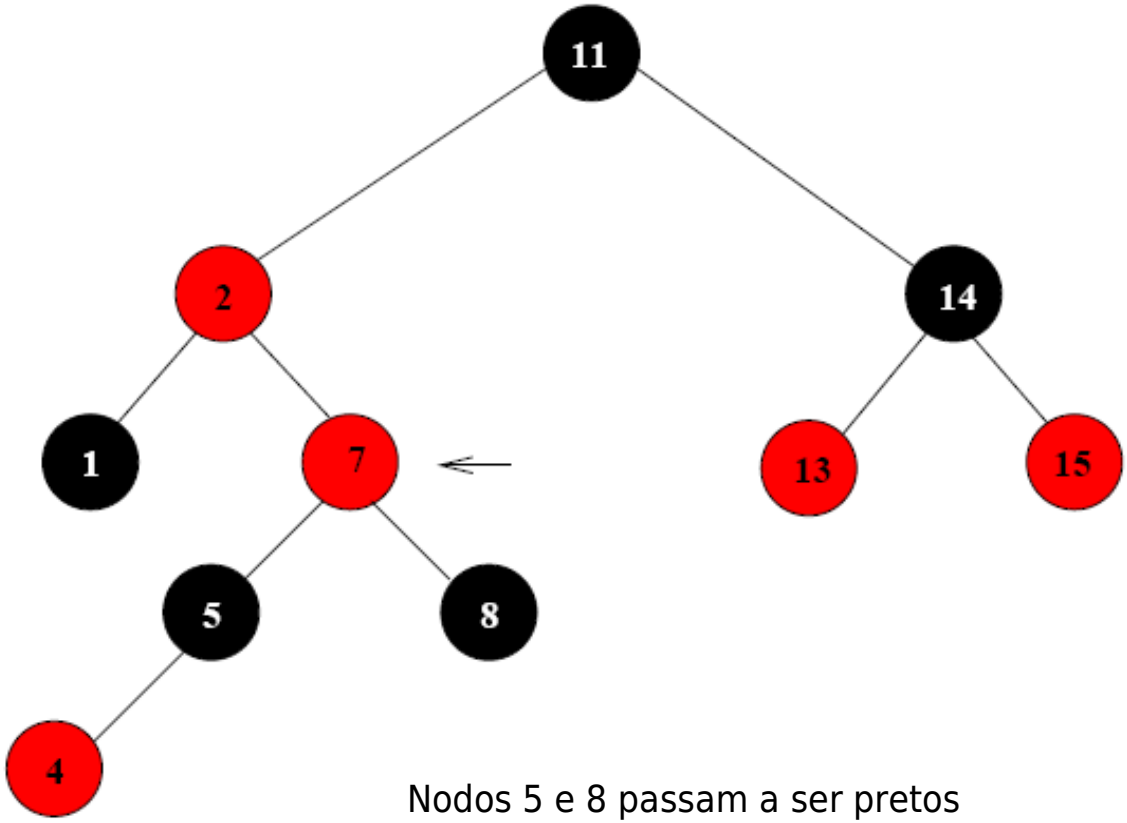
- Estado inicial da árvore



Caso 2: O tio do elemento inserido é VERMELHO

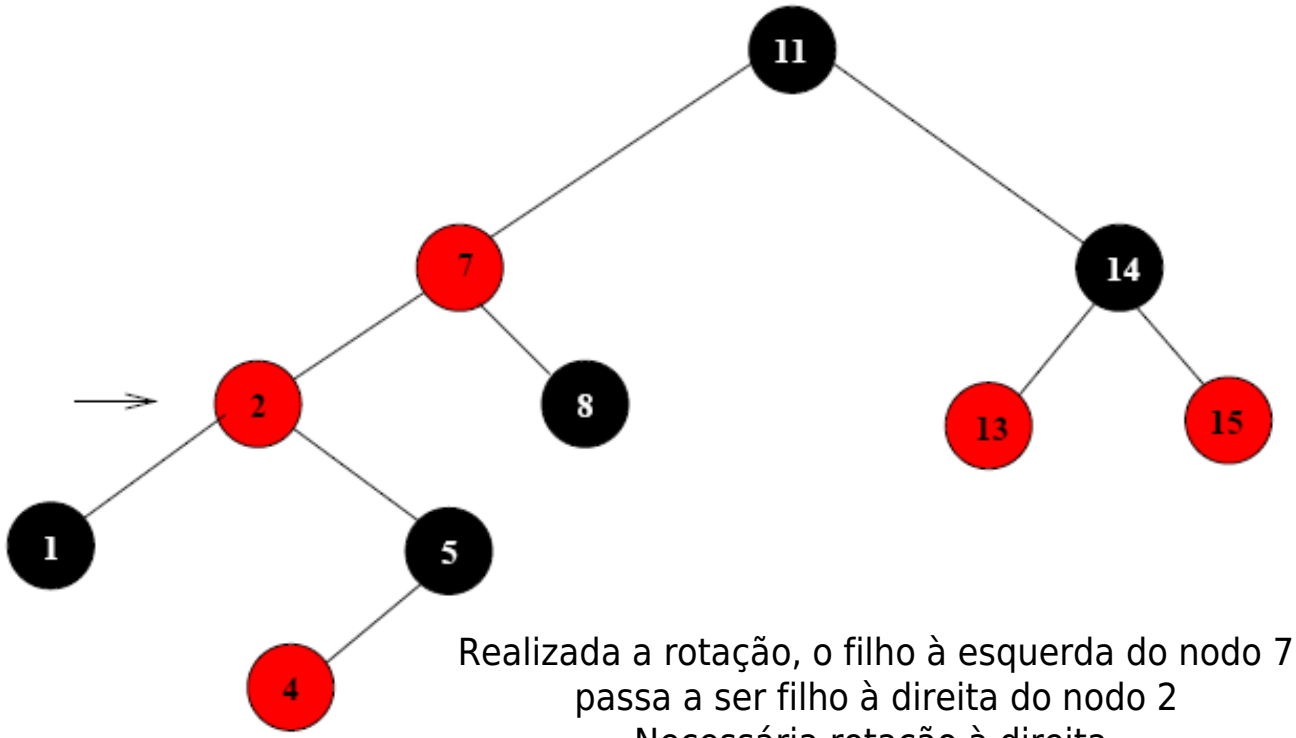


Caso 3b: O tio do elemento é PRETO e o elemento é filho à direita



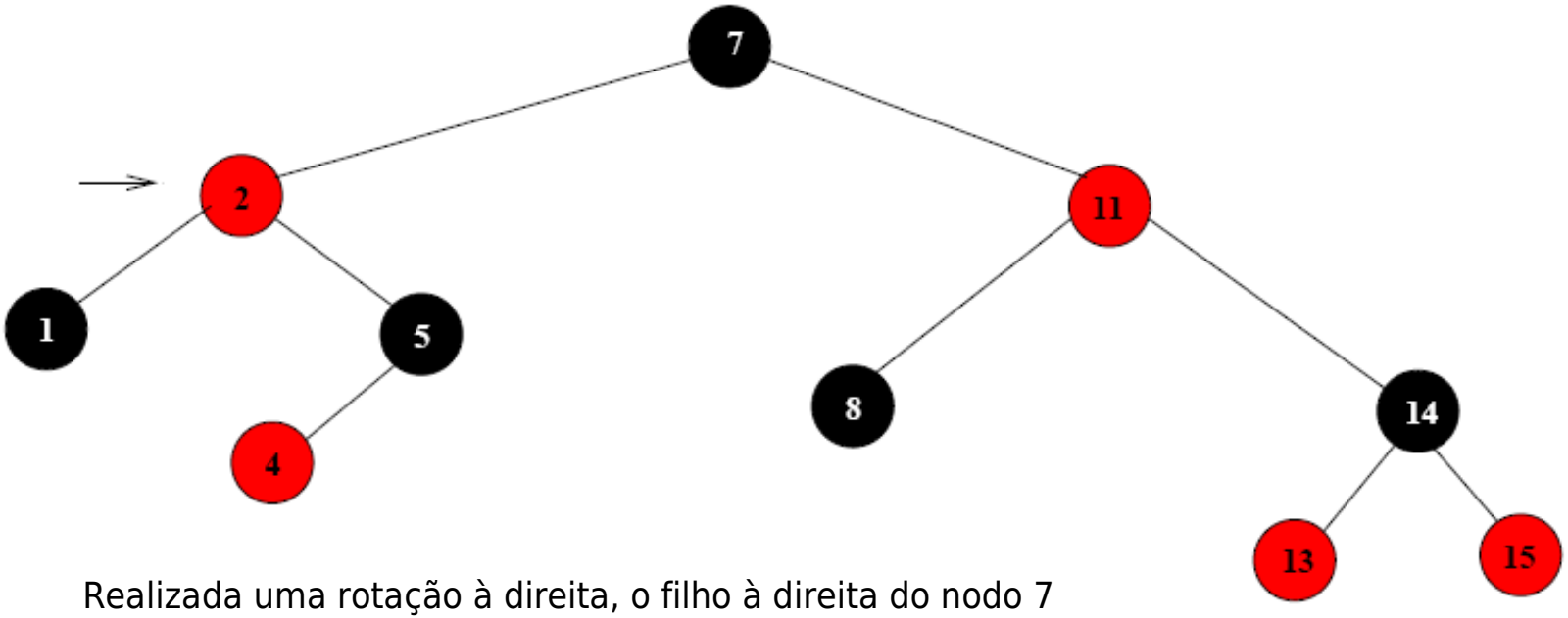
Nodos 5 e 8 passam a ser pretos
Outra violação da propriedade 4 entre os nodos 2 e 7
Necessária uma rotação à esquerda

Caso 3a: O tio do elemento é PRETO e o elemento é filho à esquerda



Realizada a rotação, o filho à esquerda do nodo 7 passa a ser filho à direita do nodo 2
Necessária rotação à direita.

Exemplo



Realizada uma rotação à direita, o filho à direita do nodo 7
passa a ser filho à esquerda do nodo 11
O nodo 7 é colorido de preto, é restaurada a propriedade 4
e nenhuma outra é violada

Aplicações

- A árvore VP, assim com a AVL, necessita de no máximo 1 rotação para balancear na inserção. Pode ter log(n) trocas de cores.
- Na remoção, a VP faz no máximo 1 rotação (simples ou dupla), enquanto a AVL pode fazer log(n) rotações.
- A árvore VP tem melhor pior caso que a árvore AVL.
  - Isso faz com que a árvore seja utilizada em aplicações de tempo real (críticas)
  - AVL é uma estrutura mais balanceada que a VP, o que leva a AVL a ser mais lenta na inserção e remoção, mas mais rápida na busca.
  - Pode ser usada para construir blocos em outras estruturas de dados que precisam de garantias no pior caso. Por exemplo: estruturas de dados em geometria computacional podem ser baseadas em árvores VP.

Exercício

- Inserir:
  - 41 – 38 – 31 – 12 – 19 – 8