Algoritmo e Estrutura de Dados II
COM-112

Árvore Rubro-Negra

Vanessa Souza

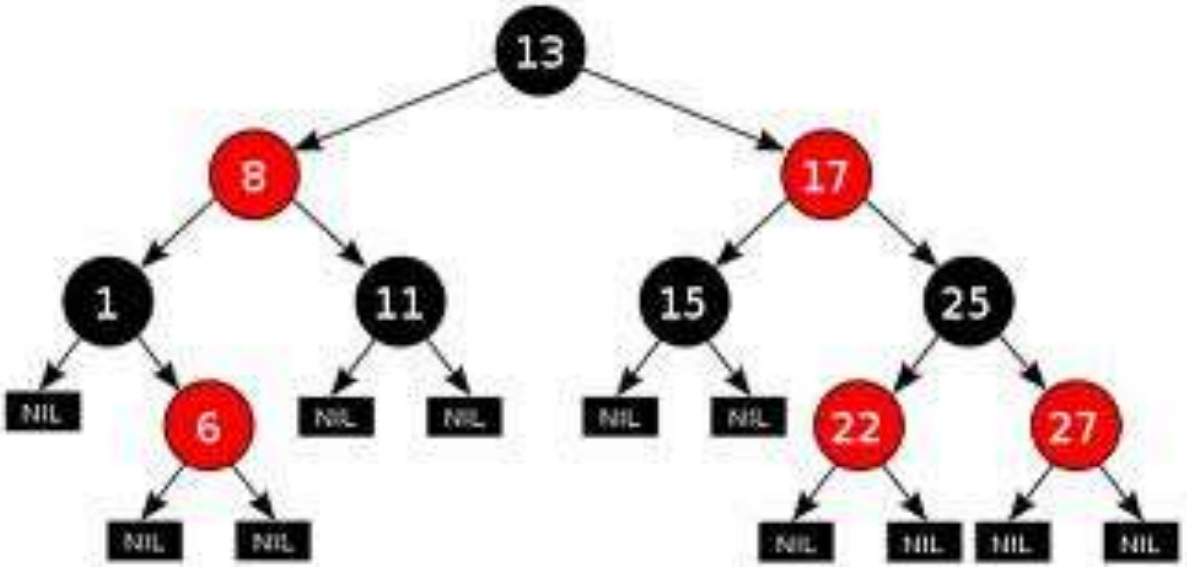
Universidade Federal de Itajubá – Instituto de Matemática e Computação

Árvores Binárias Balanceadas

- Como vimos, árvores de pesquisa que são ‘balanceadas’ garantem que as operações básicas de conjuntos dinâmicos (busca, inserção, remoção, maior, menor) demorem o tempo de O(log n) no pior caso.

Árvores Binárias Balanceadas

- Árvores **vermelho-preto** é um dos muitos algoritmos existentes para garantir o balanceamento de uma árvore de pesquisa binária.
  - Rubro-negra
  - *Red-Black trees*



Introdução

- Inventada em 1972, 10 anos depois da AVL por Rudolf Bayer, sob o nome B-árvores binárias simétricas.

- Adquirindo em 1978 seu atual nome, por Leo J. Guibas and Robert Sedgewick.

Árvore Vermelho-Preto

- A árvore Vermelho-Preto tem esse nome devido a "coloração" de seus nós.

- Cada nó possui um atributo extra que indica sua cor: **vermelho** ou **preto**.

- O fato de um nó ser vermelho ou preto é usado como fator de balanceamento.

Árvores Binárias Balanceadas

- Árvores vermelho-preto são boas árvores de pesquisa, pois pode-se mostrar que sua altura é, no máximo, igual a $2 * \log_2(n+1)$.
  - Ver *Cormen et al.; Algoritmos : teoria e prática. Rio de Janeiro : Elsevier, 2002.*

Árvore Vermelho-Preto

- As árvores vermelho-preto são um tipo de árvore binária de pesquisa aproximadamente balanceada.

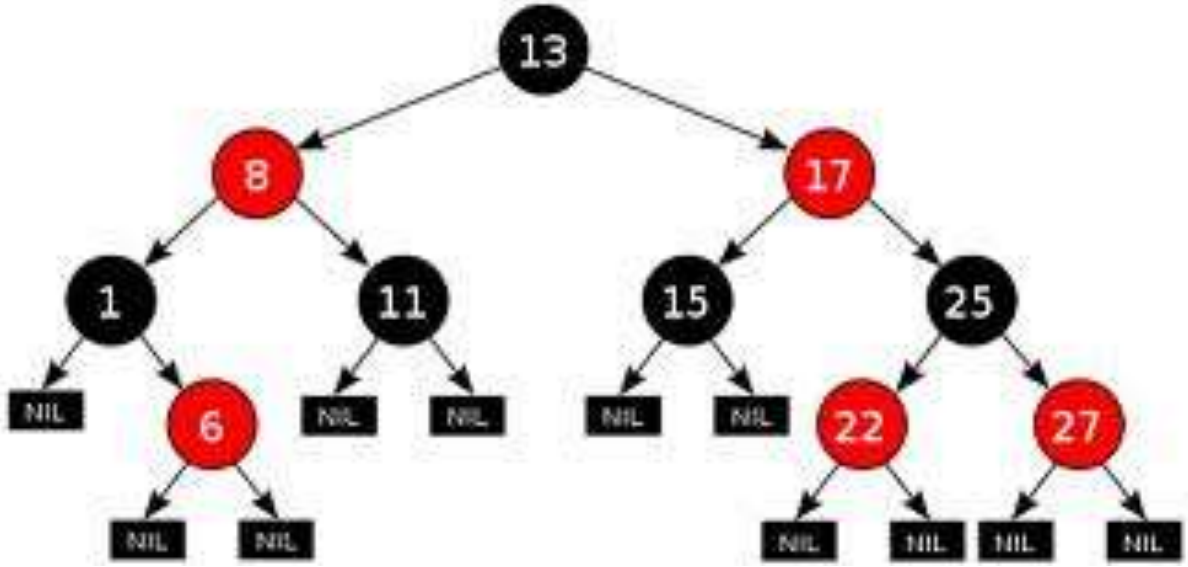
- Restringindo o modo como os nós podem ser coloridos em qualquer caminho desde a raiz até uma folha, as árvores vermelho-preto asseguram que nenhum desses caminhos será maior que duas vezes o comprimento de qualquer outro.

PROPRIEDADES

Árvore Vermelho-Preto

- Uma árvore binária de pesquisa é uma árvore vermelho-preto se satisfaz as seguintes condições:
  1. Todo nó é vermelho ou preto;
  2. A raiz da árvore é preta;
  3. Se um nó é vermelho, então seus filhos são pretos;
  4. Para cada nó, todos os caminhos até as folhas contêm o mesmo número de nós pretos.

Árvore Vermelho-Preto

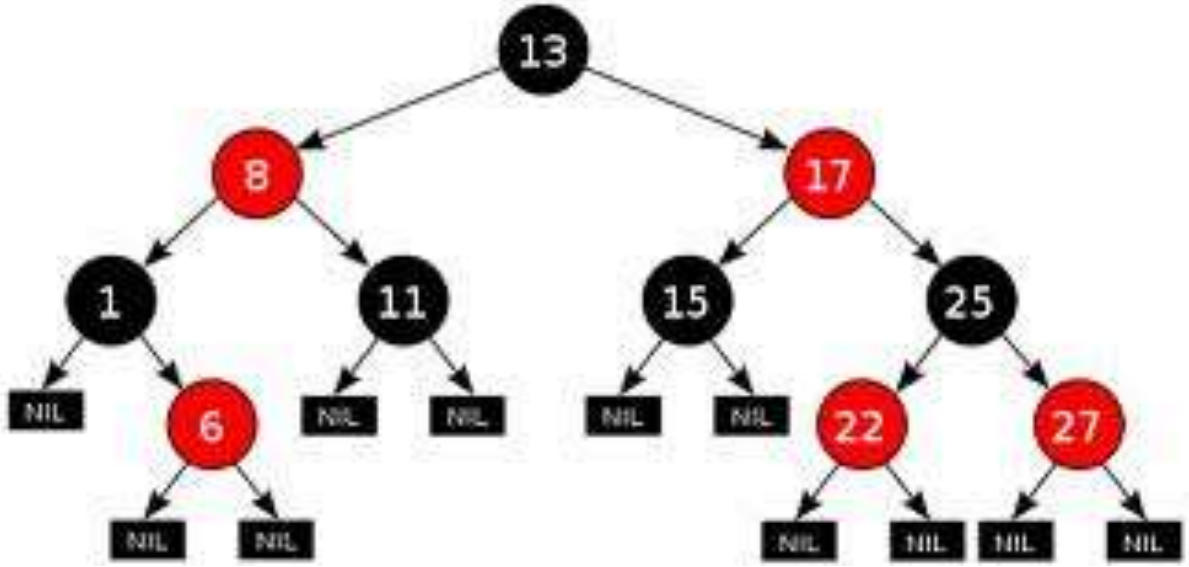


1. Todo nó é vermelho ou preto;
2. A raiz da árvore é preta;
3. Se um nó é vermelho, então seus filhos são pretos;
4. Para cada nó, todos os caminhos até as folhas contêm o mesmo número de nós pretos.

Árvore Vermelho-Preto

- Altura de Preto
  - Número de nós pretos em qualquer caminho desde um nó x, sem incluir esse nó, até uma folha.
  - A altura de preto de uma árvore vermelho-preto é a altura de preto da raiz.

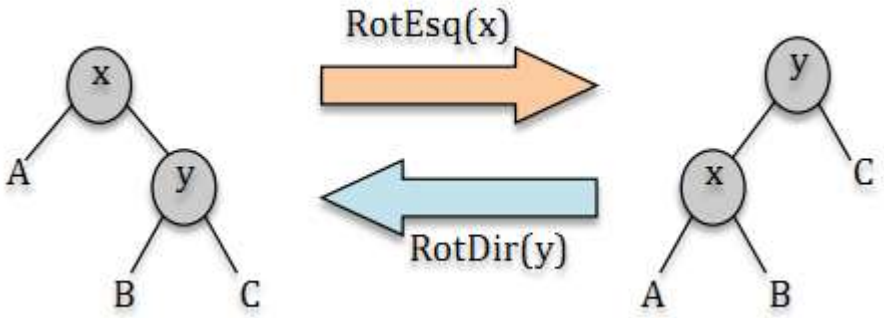
Árvore Vermelho-Preto



1. Altura de preto da árvore
2. Altura de preto do nó 8

Árvore Vermelho-Preto

- A inclusão e remoção de nós em árvores vermelho-preto, embora possam ser feitas em tempo O(log n), podem violar as propriedades da árvore. Para restabelecer as propriedades vermelho-preto podem ser usadas as operações de rotação à esquerda e à direita.



Inserção

Árvore Vermelho-Preto – Inserção

- Algoritmo:
  - O nó é avaliado considerando a árvore como uma árvore binária de pesquisa comum.
  - O novo nó é colorido de vermelho
  - Chama-se um procedimento para restaurar as propriedades vermelho-preto da árvore.

Árvore Vermelho-Preto – Inserção

- Todo nó a ser avaliado por convenção é vermelho(pois se fosse preto não seguiria a propriedade 4).

- Se após a inserção for quebrada qualquer propriedade da AVP devem ser feitas rotações e/ou inversão de cores dos nós para que sejam satisfeitas todas as propriedades .

- As regras de inserção levam em consideração a cor do **TIO** (o outro filho do pai do nó que recebeu o novo nó) do nó avaliado.

Árvore Vermelho-Preto – Inserção

- **Vamos pensar...**

- Quando um novo nó é avaliado na árvore, quais propriedades podem ser quebradas?
  1. Todo nó é vermelho ou preto;
  2. A raiz da árvore é preta;
  3. Se um nó é vermelho, então seus filhos (caso existam) são pretos;
  4. Para cada nó, todos os caminhos até as folhas contêm o mesmo número de nós pretos.

Árvore Vermelho-Preto – Inserção

- Quando um novo nó é avaliado na árvore, quais propriedades podem ser quebradas?
  1. Todo nó é vermelho ou preto;
  2. A raiz da árvore é preta;
  3. Se um nó é vermelho, então seus filhos (caso existam) são pretos;
  4. Para cada nó, todos os caminhos até as folhas contêm o mesmo número de nós pretos.

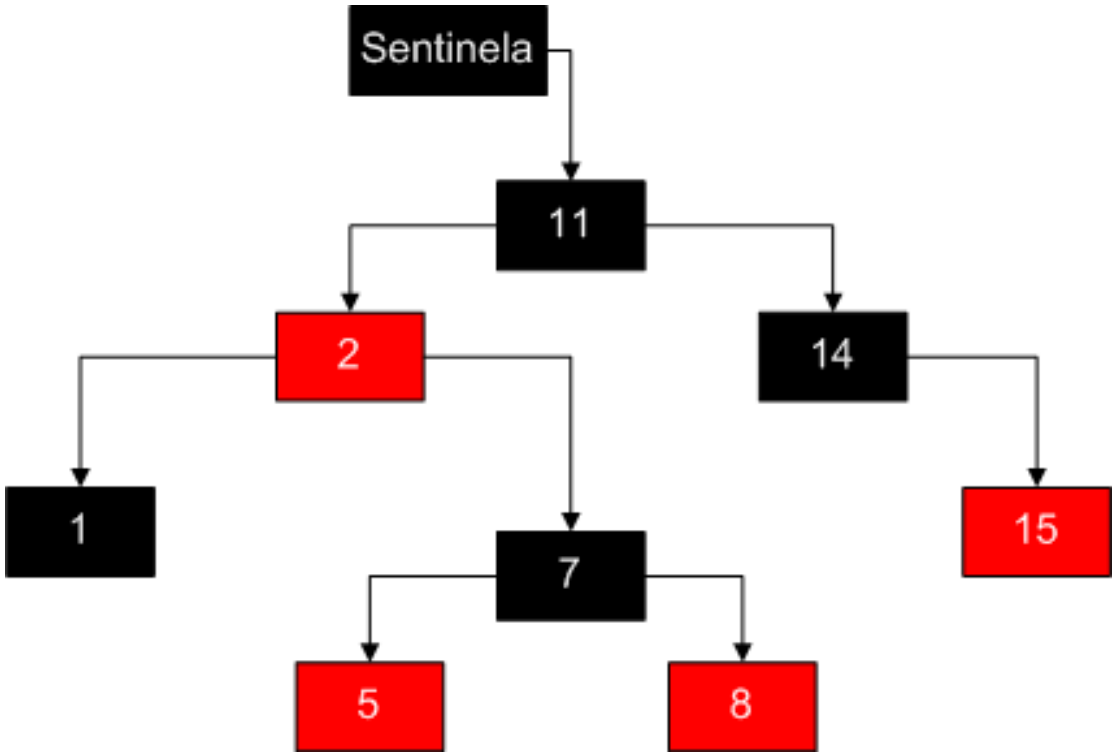
- Certamente continuam válidas:
  - A propriedade 1 (pois todos os nós continuam sendo vermelhos ou pretos);
  - A propriedade 4 (pois o número de nós pretos não se altera com a inclusão de um novo nó vermelho).

Árvore Vermelho-Preto – Inserção

- Quando um novo nó é avaliado na árvore, quais propriedades podem ser quebradas?
  1. Todo nó é vermelho ou preto;
  2. A raiz da árvore é preta;
  3. Se um nó é vermelho, então seus filhos (caso existam) são pretos;
  4. Para cada nó, todos os caminhos até as folhas contêm o mesmo número de nós pretos.

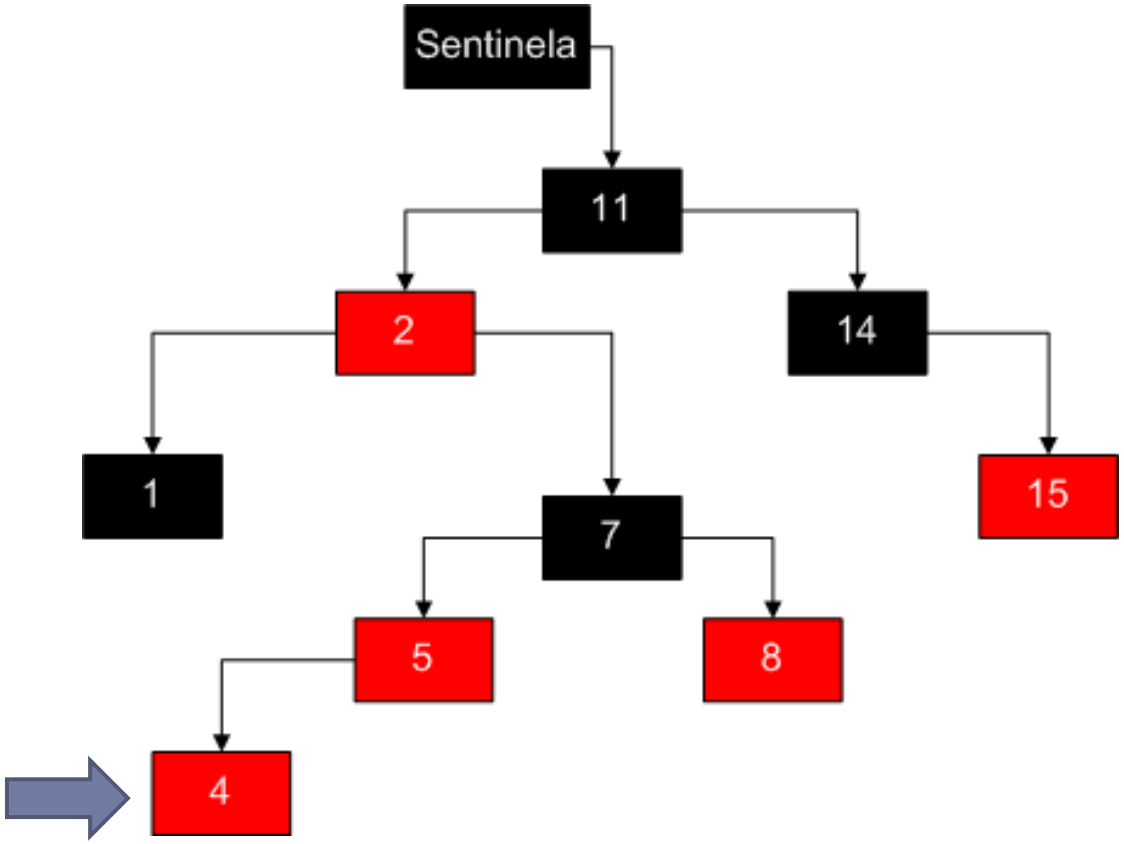
- Portanto, podem ser violadas as propriedades 2 (a raiz não pode ser vermelha) e 3 (um nó vermelho não pode ter um filho vermelho).
  - A propriedade 2 será violada se o novo nó passar a ser a raiz da árvore.
  - A propriedade 3 será violada se o pai do novo nó for vermelho.

Árvore Vermelho-Preto – Inserção



Inserir elemento 4

Árvore Vermelho-Preto – Inserção



A inserção quebrou a propriedade 3 da árvore vermelho-preto

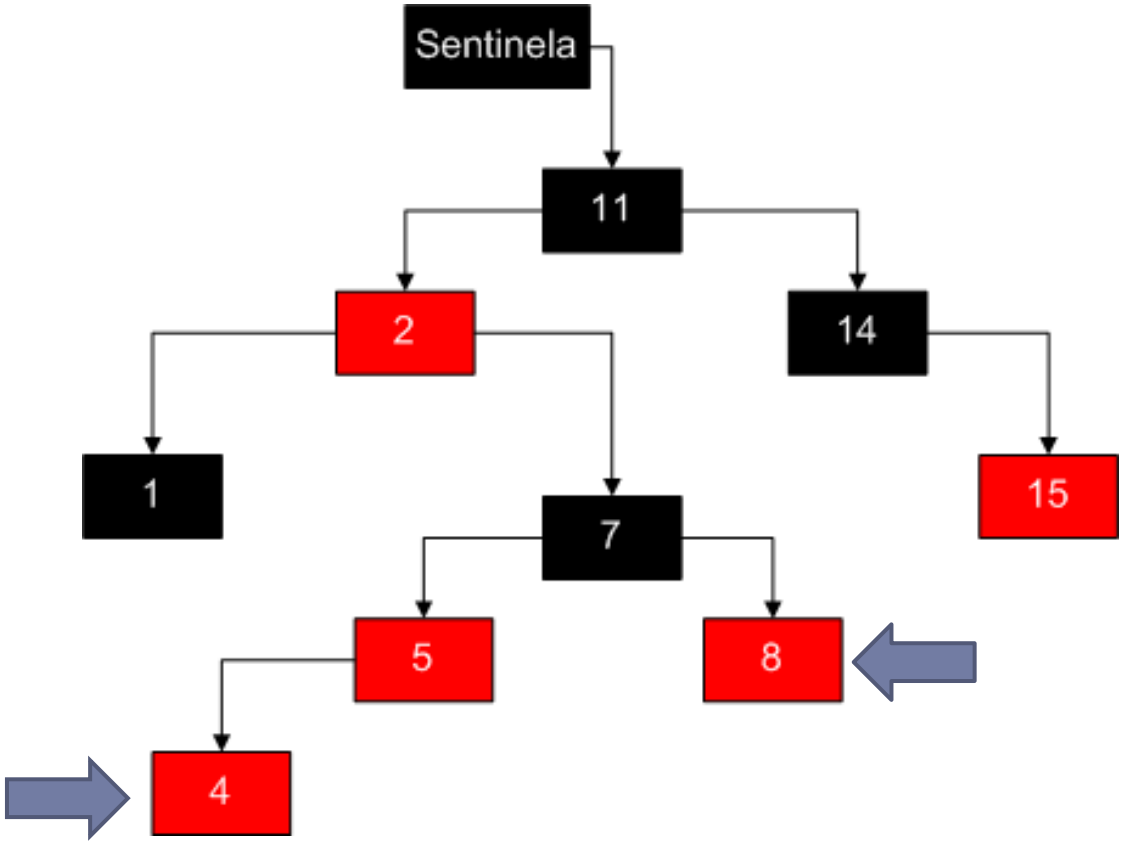
Árvore Vermelho-Preto – Inserção

- Existem três casos para corrigir as cores após uma inserção:
  - Caso 1: O tio do elemento avaliado é VERMELHO.
  - Caso 2: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da direita.
  - Caso 3: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da esquerda.
- Na verdade são 6, porque a solução depende do lado em que o nó foi avaliado.

Árvore Vermelho-Preto – Inserção

- Existem três casos para corrigir as cores após uma inserção:
  - **Caso 1: O tio do elemento avaliado é VERMELHO.**
  - Caso 2: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da direita.
  - Caso 3: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da esquerda.

Árvore Vermelho-Preto – Inserção

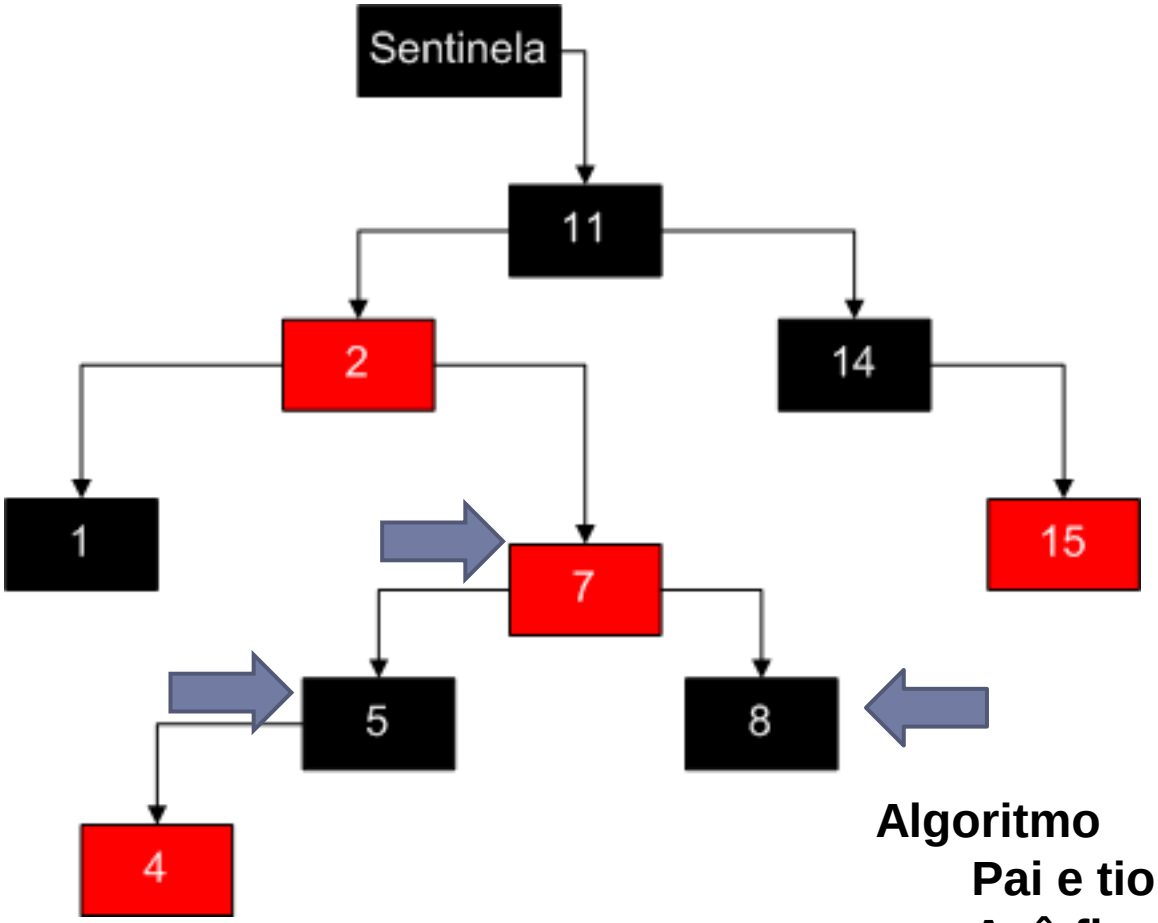


O tio do elemento avaliado é vermelho

Árvore Vermelho-Preto – Inserção

- Existem três casos para corrigir as cores após uma inserção:
  - **Caso 1: O tio do elemento avaliado é VERMELHO.**
  - **Algoritmo**
    - **Pai e tio ficam pretos**
    - **Avô fica vermelho**
    - **Avalia o avô (recursivamente)**

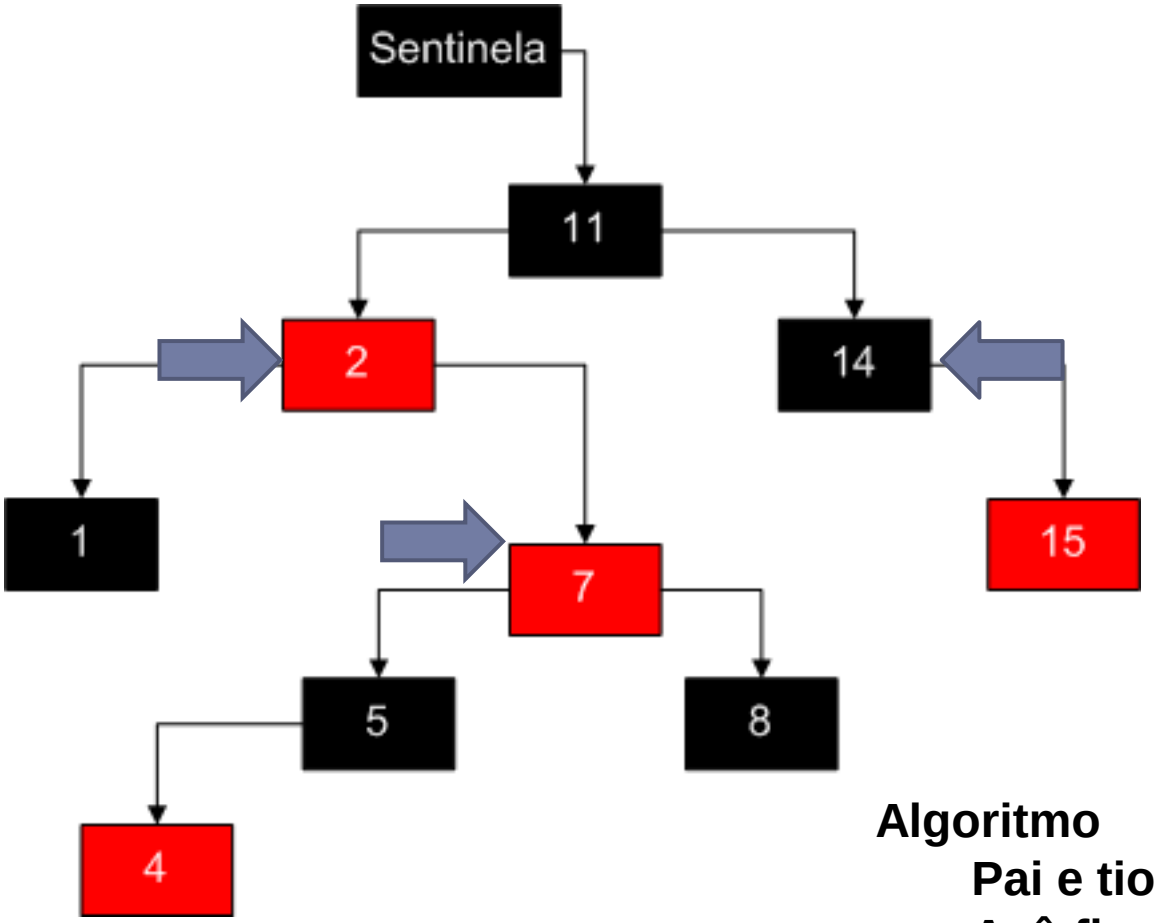
Árvore Vermelho-Preto – Inserção



Algoritmo

- **Pai e tio ficam pretos**
- **Avô fica vermelho**
- **Avalia o avô (recursivamente)**

Árvore Vermelho-Preto – Inserção



Algoritmo

- **Pai e tio ficam pretos**
- **Avô fica vermelho**
- **Avalia o avô (recursivamente)**

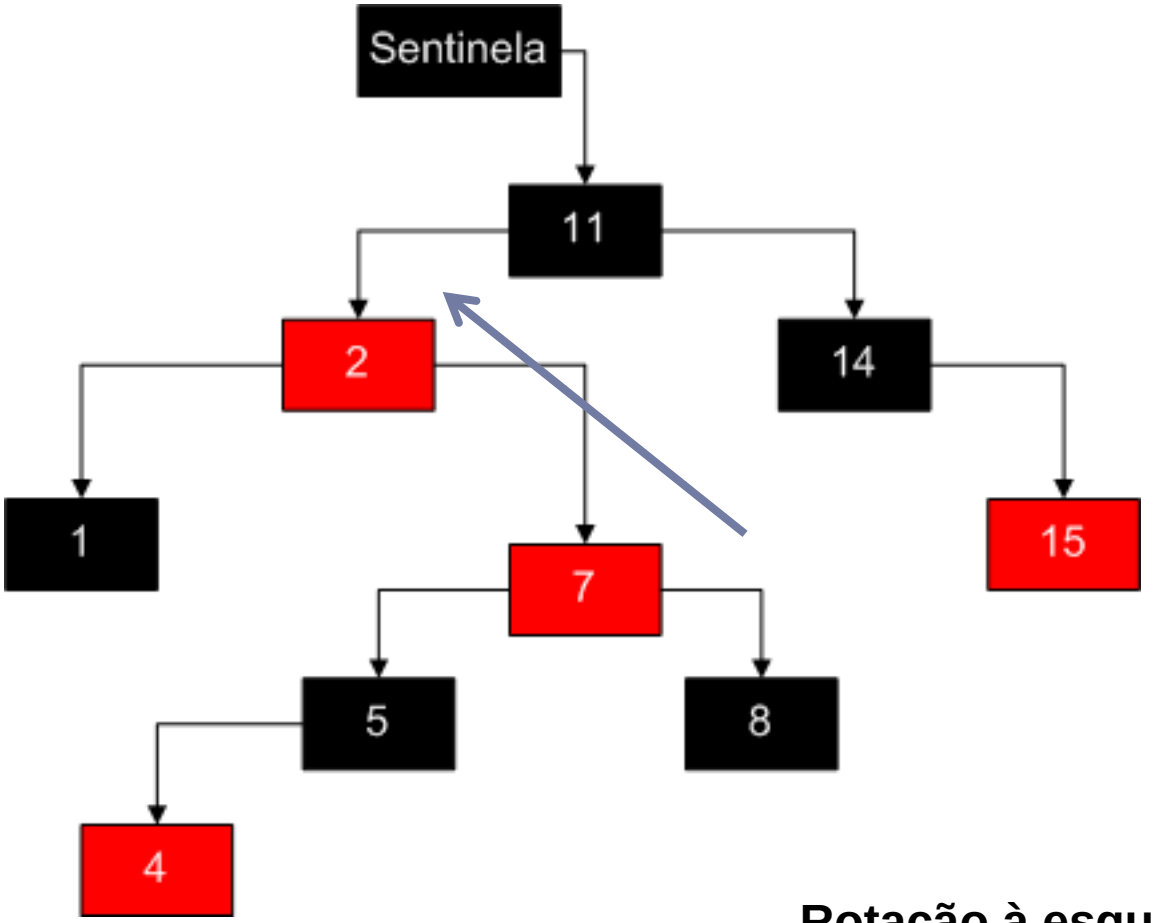
Árvore Vermelho-Preto – Inserção

- Existem três casos para corrigir as cores após uma inserção:
  - Caso 1: O tio do elemento avaliado é VERMELHO.
  - **Caso 2: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da direita.**
  - Caso 3: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da esquerda.

Árvore Vermelho-Preto – Inserção

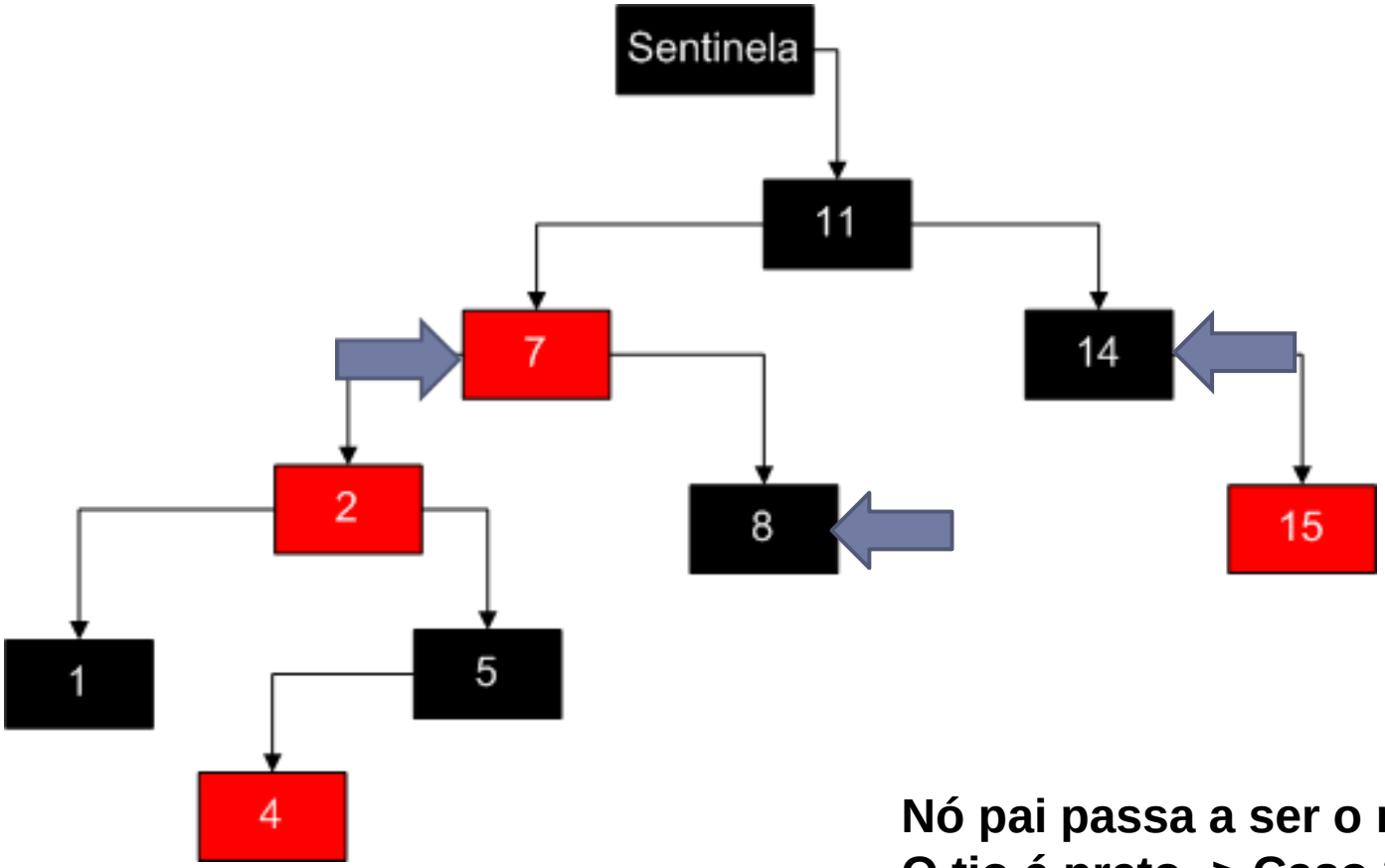
- Existem três casos para corrigir as cores após uma inserção:
  - **<u>Caso 2:</u> O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da direita.**
  - **Algoritmo**
    - **Rotação à esquerda no pai do nó (que passa a ser o nó avaliado)**
      - **Leva ao caso 3**

Árvore Vermelho-Preto – Inserção



Rotação à esquerda no nó pai

Árvore Vermelho-Preto – Inserção

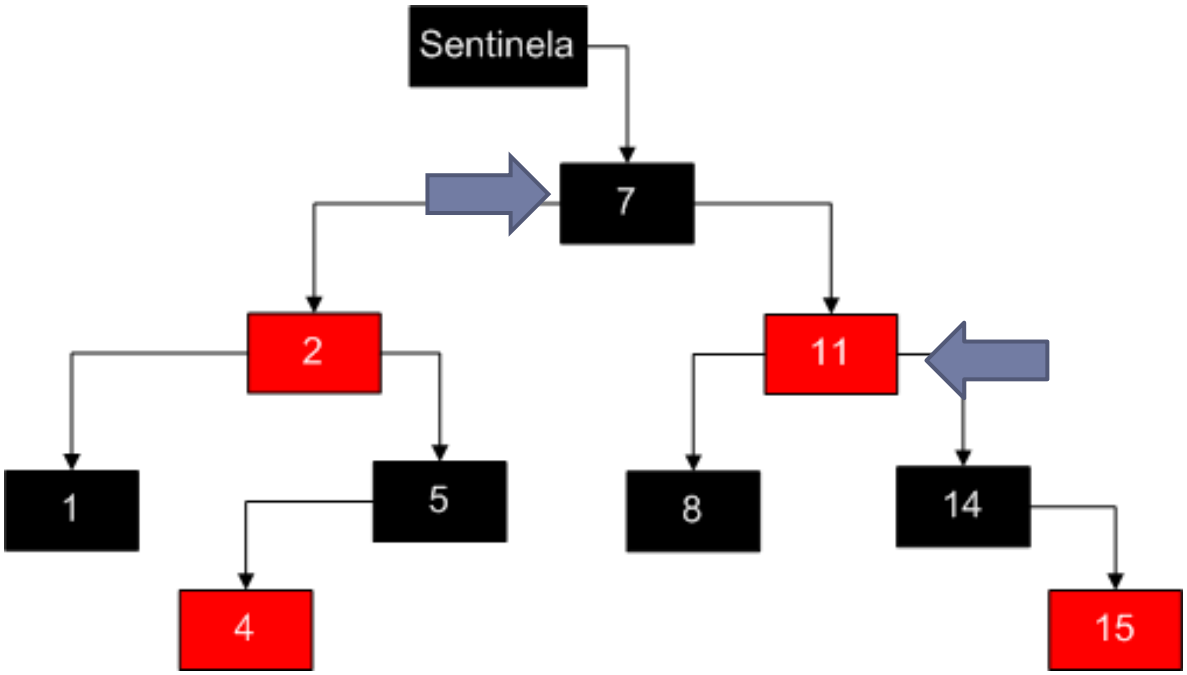


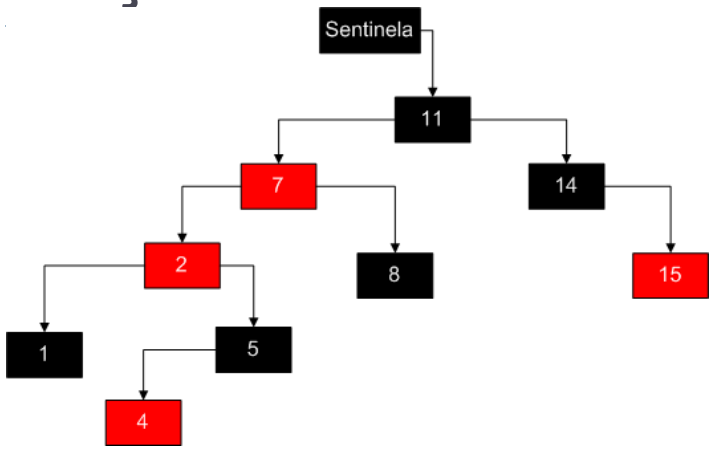
Nó pai passa a ser o nó avaliado
O tio é preto -> Caso 3

Árvore Vermelho-Preto – Inserção

- Existem três casos para corrigir as cores após uma inserção:
  - **<u>Caso 3:</u> O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da esquerda.**
  - **Algoritmo**
    - **Pai fica preto**
    - **Avô fica vermelho**
    - **Rotação à direita no avô**

Árvore Vermelho-Preto – Inserção





Pai fica preto
Avô fica vermelho
Rotação à direita no avô

Árvore Vermelho-Preto – Inserção

- Algoritmo:
  - O nó é avaliado considerando a árvore como uma árvore binária de pesquisa comum.
  - O novo nó é colorido de vermelho
  - **Chama-se um procedimento para restaurar as propriedades vermelho-preto da árvore.**

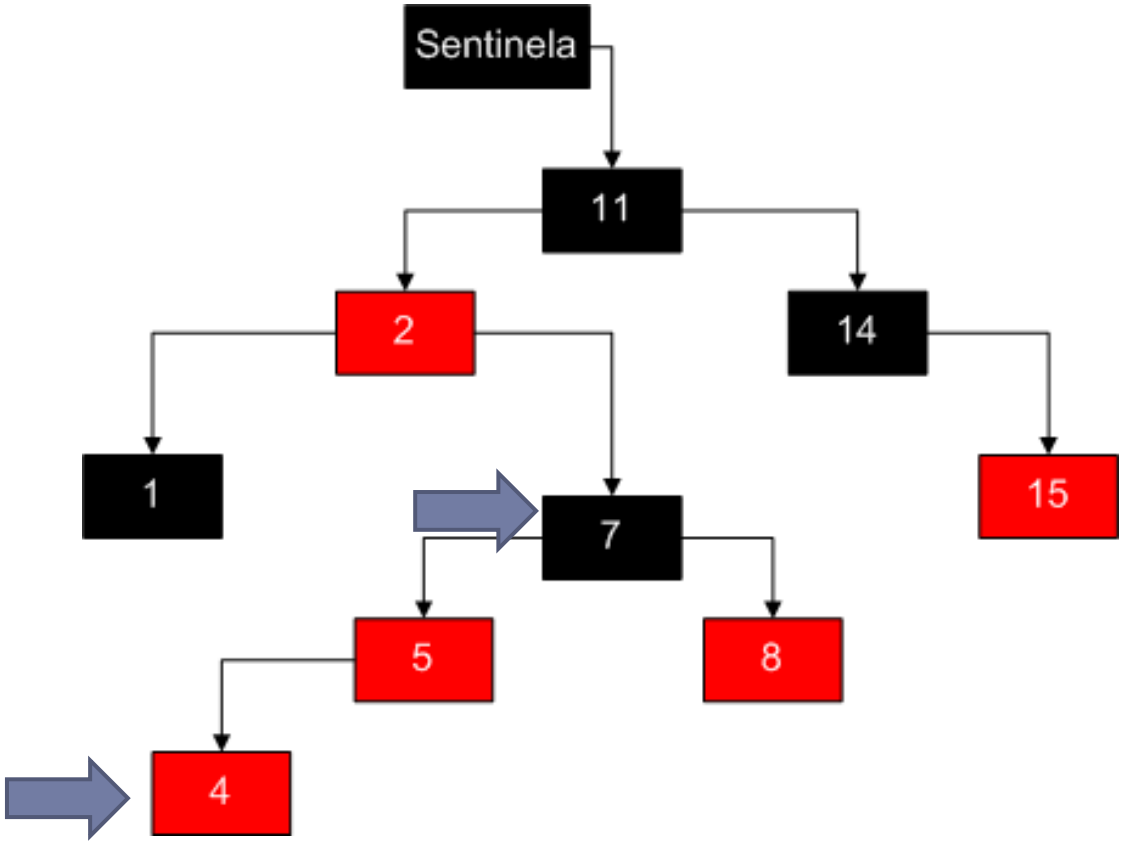
Árvore Vermelho-Preto – Inserção

- Existem três casos para corrigir as cores após uma inserção:
  - Caso 1: O tio do elemento avaliado é VERMELHO.
  - Caso 2: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da direita.
  - Caso 3: O tio do elemento avaliado é PRETO e o elemento avaliado é um filho da esquerda.
- No final a raiz da árvore é sempre colorida de preto para validar a propriedade 2 (A raiz da árvore é sempre preta).

Árvore Vermelho-Preto – Inserção

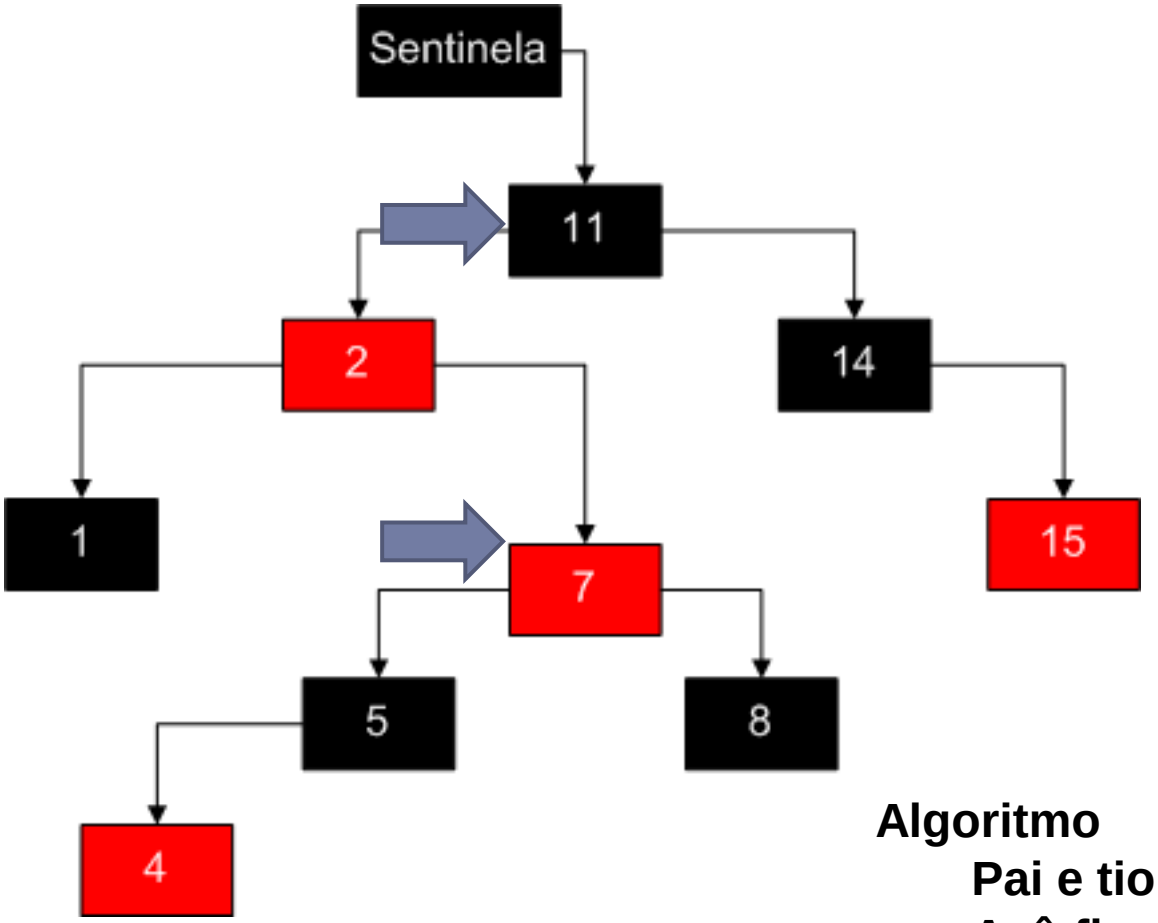
- Reparem que em todos os casos vistos anteriormente, o pai do nó avaliado é filho da esquerda de seu avô.

Árvore Vermelho-Preto – Inserção



A inserção quebrou a propriedade 3 da árvore vermelho-preto

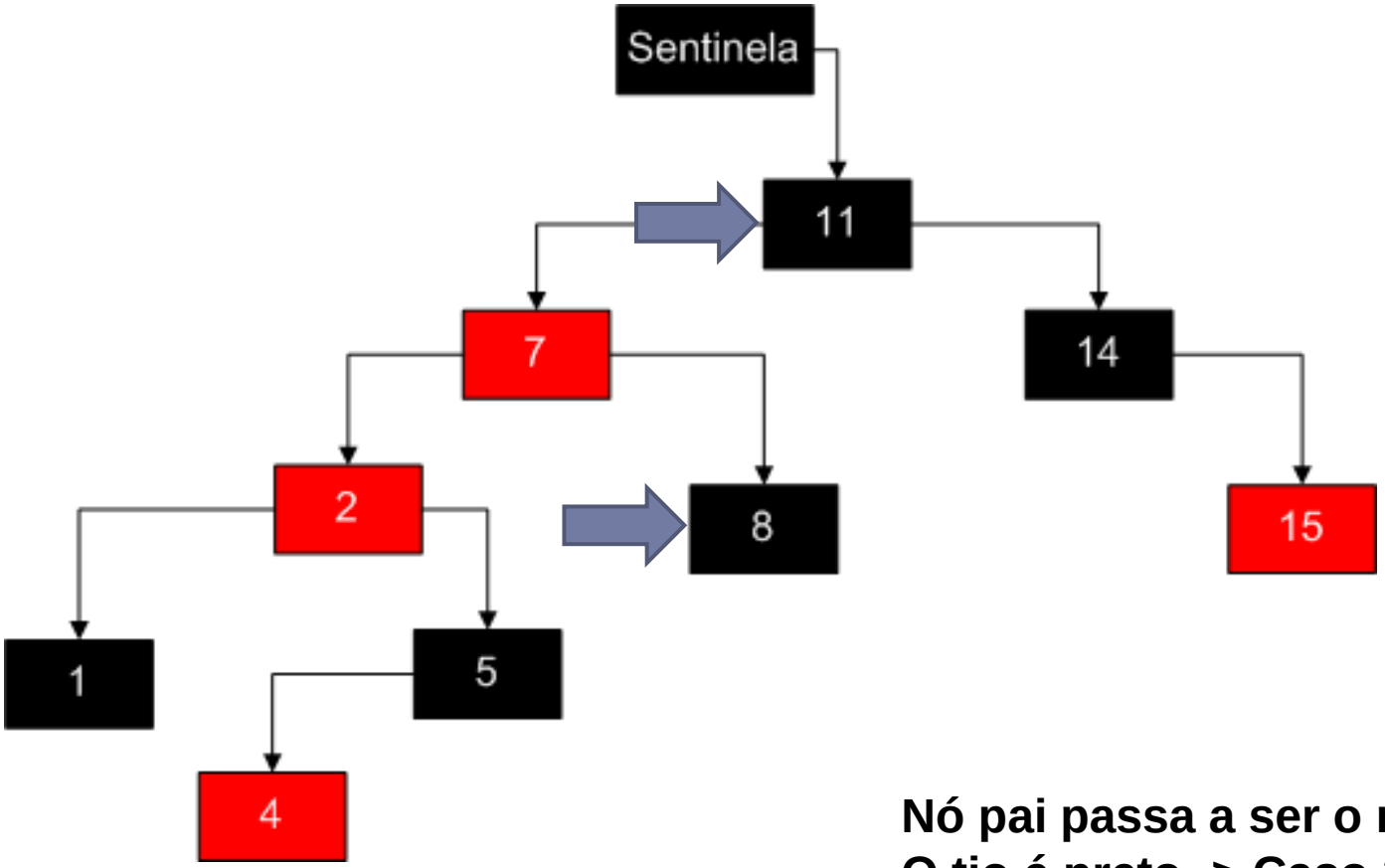
Árvore Vermelho-Preto – Inserção



Algoritmo

Pai e tio ficam pretos
Avô fica vermelho
Avalia o avô (recursivamente)

Árvore Vermelho-Preto – Inserção



Nó pai passa a ser o nó avaliado
O tio é preto -> Caso 3

Árvore Vermelho-Preto – Inserção

- Reparem que em todos os casos vistos anteriormente, o pai do nó avaliado é filho da esquerda de seu avô.
  - Se o nó avaliado estiver à direita do avô, o algoritmo é o mesmo, porém espelhado, com esquerda e direita trocadas.

Árvore Vermelho-Preto – Inserção

- Resumo : Para z = esquerda(avô)

Caso	Cor do Tio	Filho	Ações
1	vermelho	Direita ou Esquerda	Colore o pai de preto Colore o tio de preto Colore o avô de vermelho z = avô – faz nova avaliação
2	preto	direita	z = pai Rotação à esquerda (z) Vira Caso 3
3	preto	esquerda	Colore o pai de preto Colore o avô de vermelho Rotação à direita no avô
1,2,3			Colore a raiz de preto

COPIAR!!!!

Árvore Vermelho-Preto – Inserção

- Resumo : Para z = direita(avô)

Caso	Cor do Tio	Filho	Ações
1	vermelho	Direita ou Esquerda	Colore o pai de preto Colore o tio de preto Colore o avô de vermelho z = avô – faz nova avaliação
2	preto	esquerda	z = pai Rotação à direita (z) Vira Caso 3
3	preto	direita	Colore o pai de preto Colore o avô de vermelho Rotação à esquerda no avô
1,2,3			Colore a raiz de preto

Comparação com a AVL

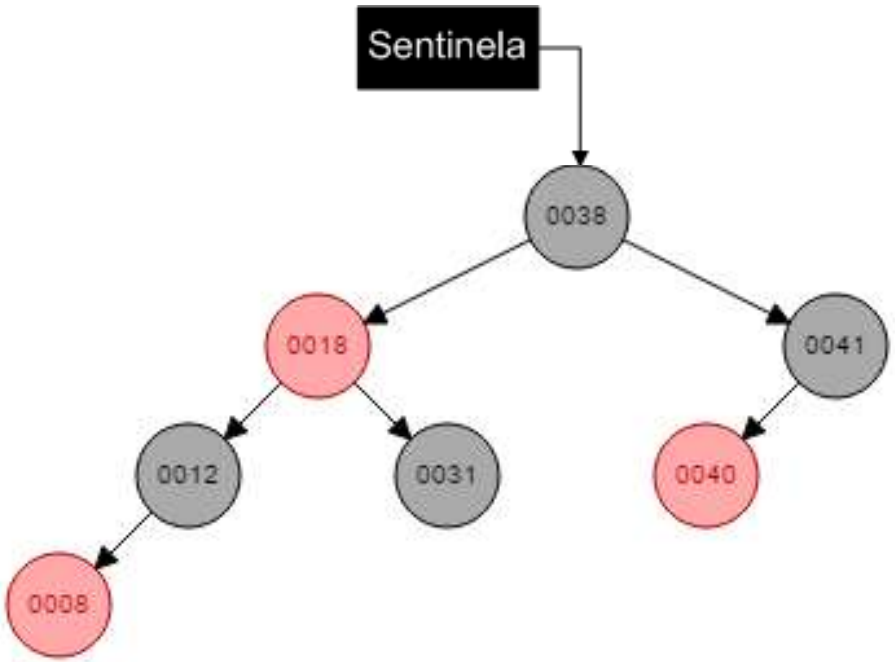
- Árvores AVL:
  - primeira árvore binária de busca com balanceamento proposta por Adel'son-Vel'skii e Landis em 1962;
  - altura: entre $\log_2(n+1)$ e $1.4404*\log_2(n+2)-0.328$, portanto, O(logn);
- Árvores rubro-negras:
  - proposta por Guibas e Sedgewick em 1978;
  - altura: $2*\log_2(n+1)$, portanto, O(log n);
- Comparação:
  - árvores AVL são mais rigidamente balanceadas que árvores rubro-negras, levando a inserção e remoção mais lentas, porém recuperação (busca) mais rápida;

Exercício de Fixação

- Mostre a árvore Rubro-Negra que resulta após a inserção bem-sucedida das chaves 41, 38, 31, 12, 18, 8 e 40 em uma árvore Rubro-Negra inicialmente vazia.

Exercício de Fixação

- Mostre a árvore Rubro-Negra que resulta após a inserção bem-sucedida das chaves 41, 38, 31, 12, 18, 8 e 40 em uma árvore Rubro-Negra inicialmente vazia.

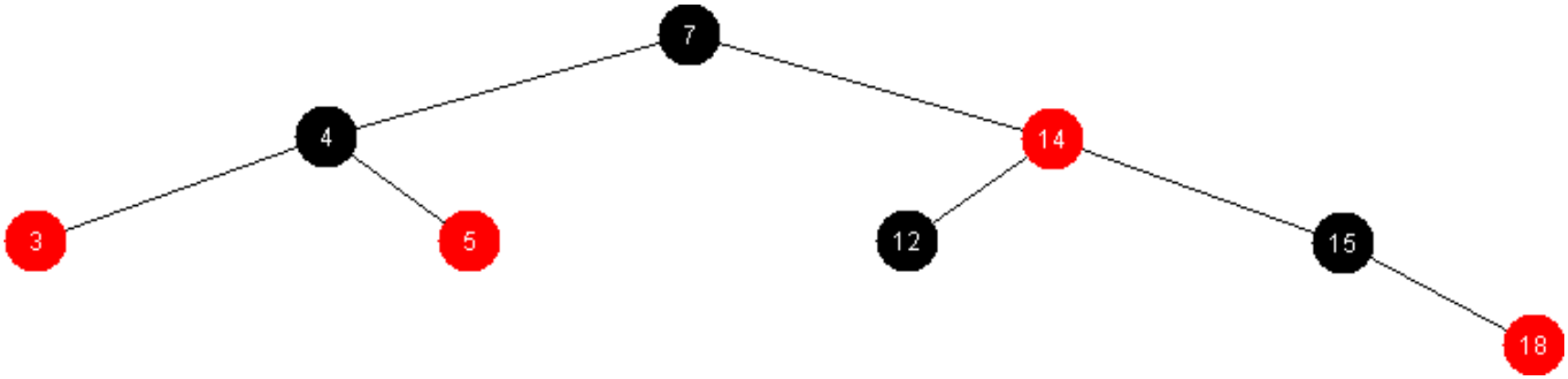


Exercício de Fixação

- Insira em uma árvore Rubro-Negra, itens com as chaves: 4 – 7– 12 – 15 – 3 – 5 – 14 – 18 (nesta ordem). Desenhe a árvore resultante da inserção, sendo que uma nova árvore deve ser desenhada quando houver uma rotação ou troca de cores. (Atenção: verifique a necessidade de rotação e/ou troca de cores a cada inserção)

Exercício de Fixação

- Insira em uma árvore Rubro-Negra, itens com as chaves: 4 – 7– 12 – 15 – 3 – 5 – 14 – 18 (nesta ordem). Desenhe a árvore resultante da inserção, sendo que uma nova árvore deve ser desenhada quando houver uma rotação ou troca de cores. (Atenção: verifique a necessidade de rotação e/ou troca de cores a cada inserção)



Para Casa

- Para cada uma das afirmações sobre árvores rubro-negras, determine se é verdadeira ou falsa. Se você achar que é verdadeira, forneça uma justificativa. Se você achar que é falsa, forneça um contra-exemplo.

  a) Um sub-árvore de uma árvore rubro-negra é também uma árvore rubro-negra.

  b) Toda árvore AVL é também uma árvore Rubro-Negra.

  c) Toda árvore rubro-negra é também uma árvore AVL.

- *Entregar na próxima quinta-feira (18/05)*
- *Em dupla*

Para Casa

- Para a próxima aula
  - *Estudar o Capítulo 13.3 do livro do Cormen, especialmente o algoritmo RB-INSERT-FIXUP.*