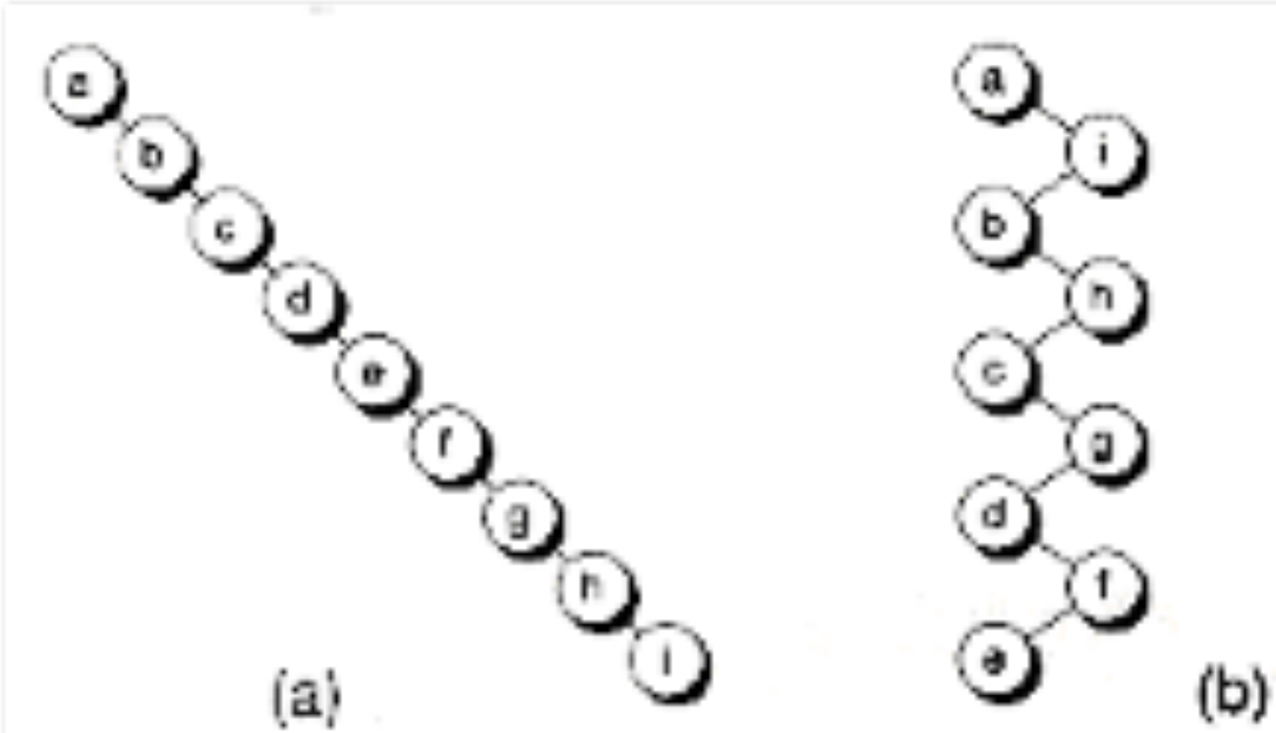
CMP1054
Estruturas de Dados I

Árvores Red-Black

Prof. Dr. José Olimpio Ferreira

Árvores degeneradas.

Figure 11.1 Degenerate trees.



Introdução.

- Inventada em 1972, 10 anos depois da AVL por Rudolf Bayer, sob o nome B-árvores binárias simétricas.
- Seu nome atual foi adotado em 1978 devido a Leo J. Guibas and Robert Sedgewick.
- Árvores vermelho-preto (do inglês Red-Black trees).

Introdução.

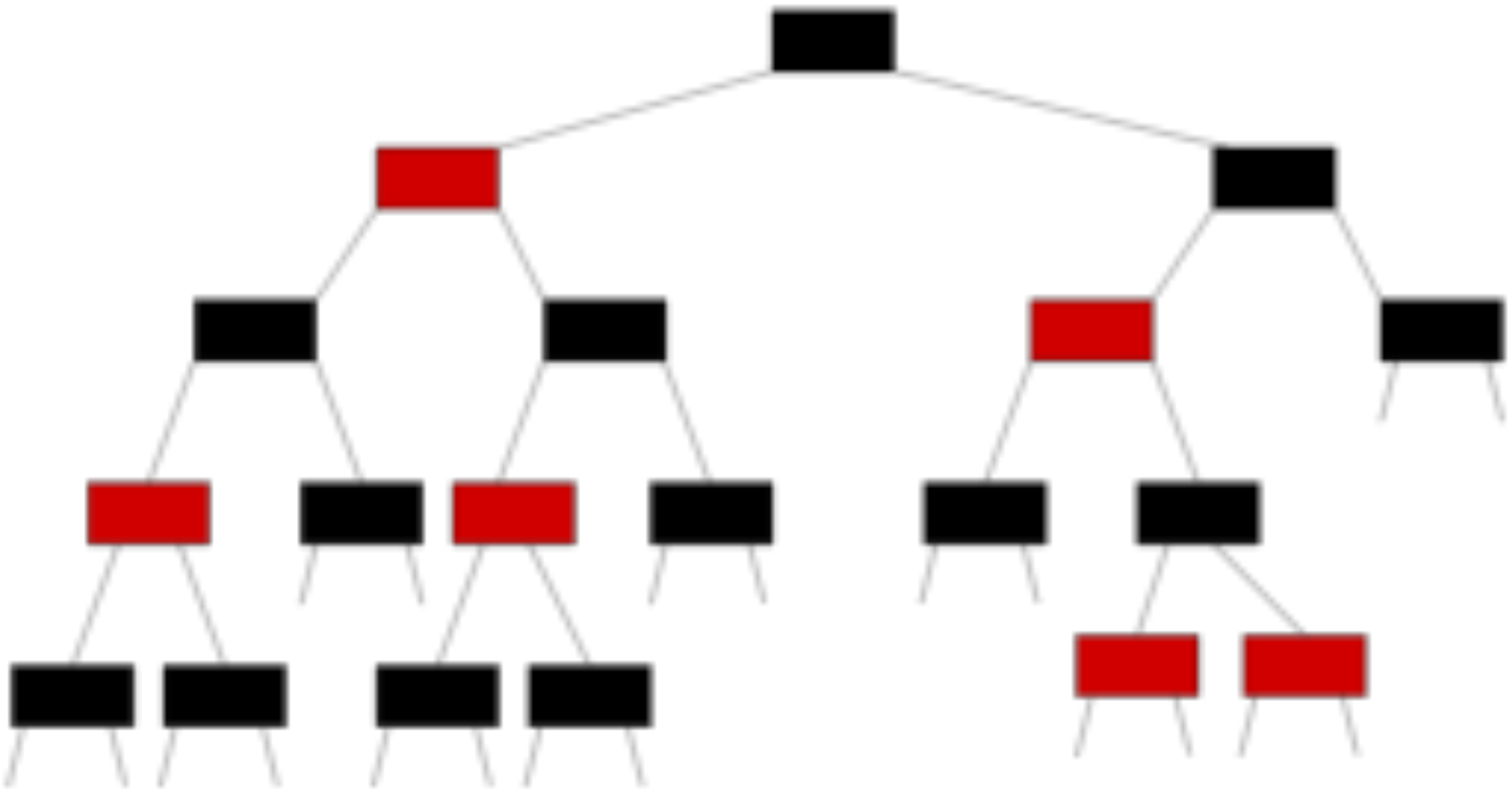
- A árvore red-black tem esse nome devido a “coloração” de seus nós.
- Uma árvore red-black (ARB) é uma árvore binária de busca com um campo adicional que armazena se o nó é vermelho ou preto.
- O fato de um nó ser vermelho ou preto é usado como fator de balanceamento da ARB

Definição.

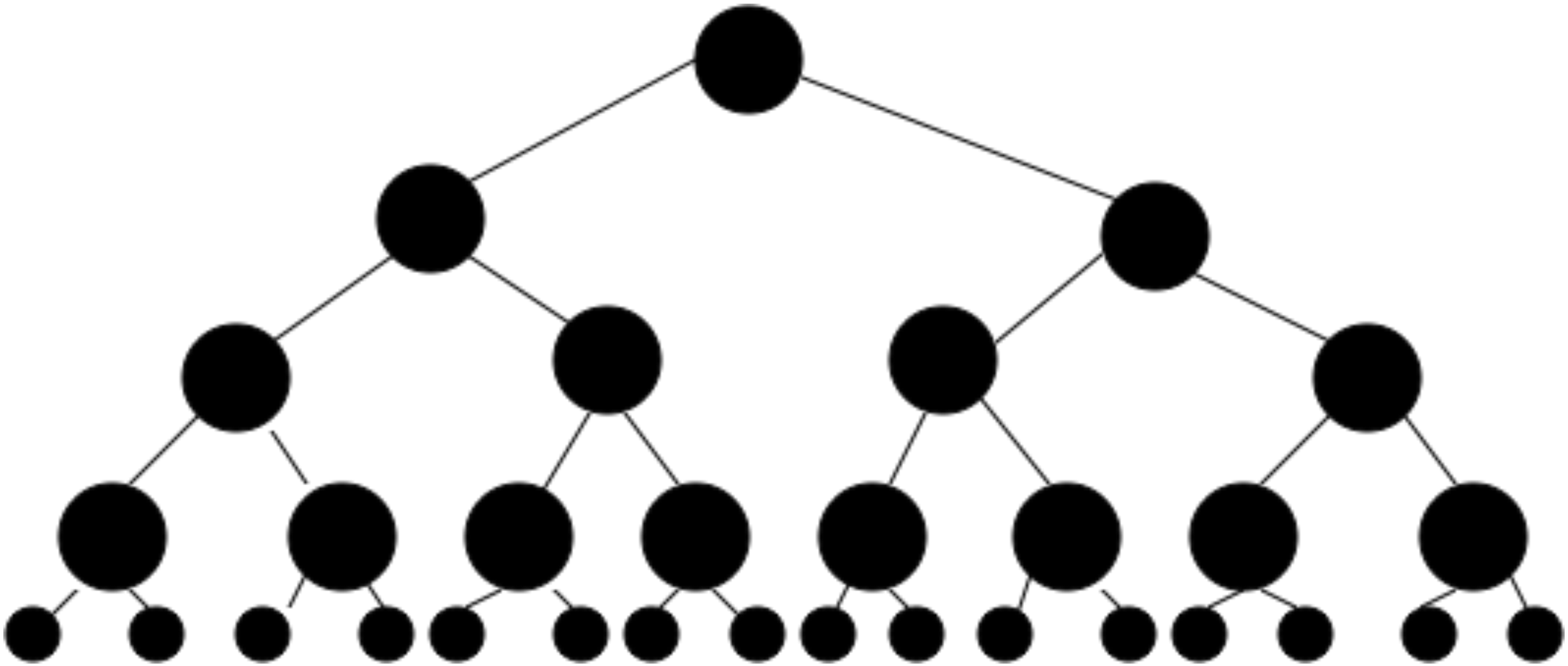
- Uma árvore de pesquisa binária é uma árvore vermelho-preto se satisfazer as seguintes propriedades:

1. Todo nó ou é vermelho ou é preto
2. A raiz é preta
3. Todas os nós nulos (folhas) são pretos
4. Se um nó é vermelho então seus dois filhos são pretos
5. Cada nó tem a mesma altura preta
    - Todos os caminhos simples de um nó para qualquer folha contém a mesma quantidade de nós pretos

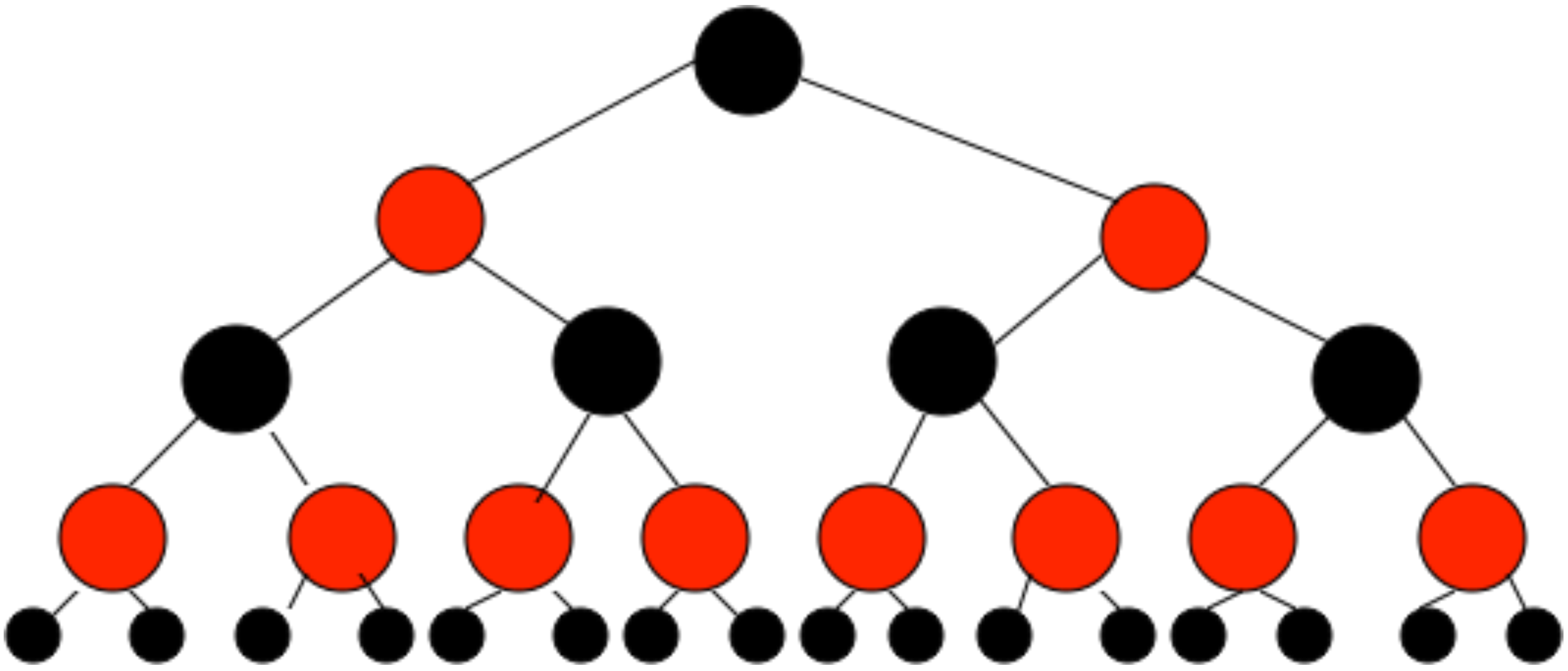
Exemplo de ABP red-black.



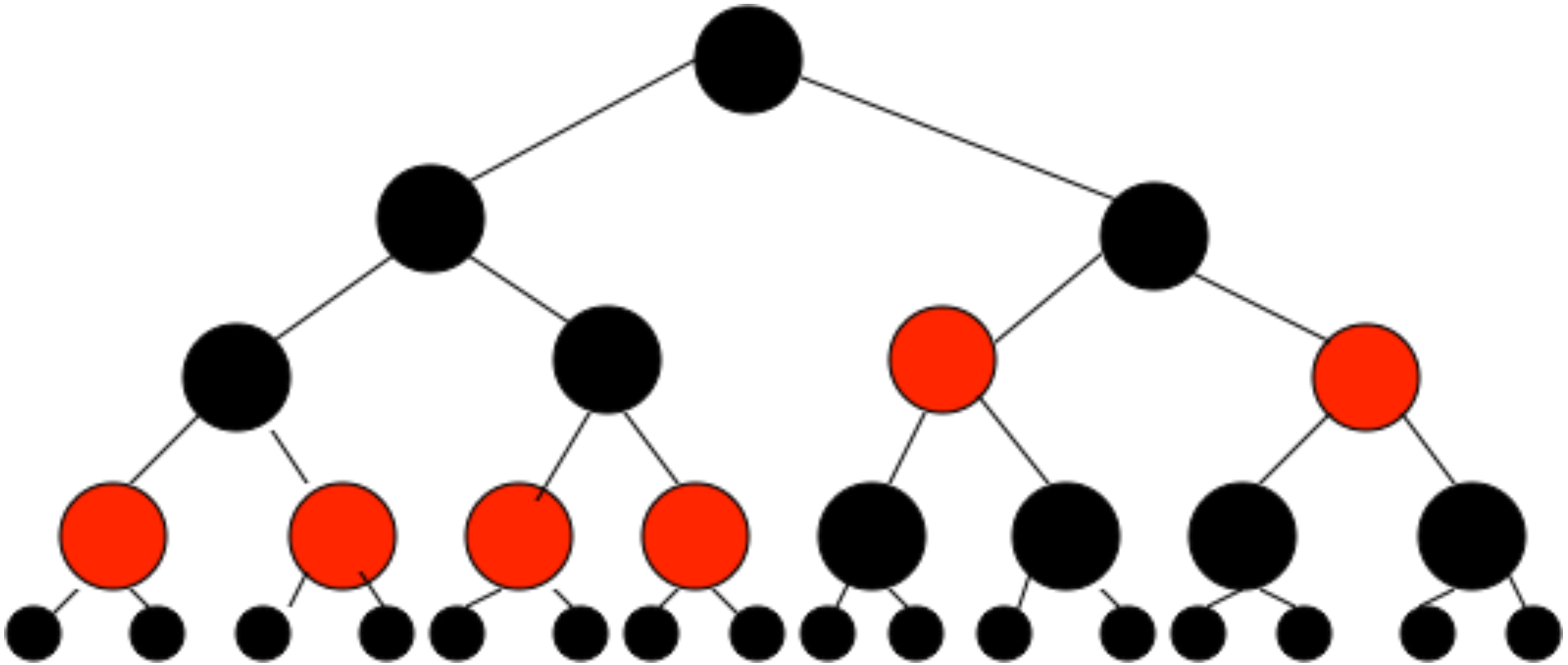
Exemplo de ABP red-black..



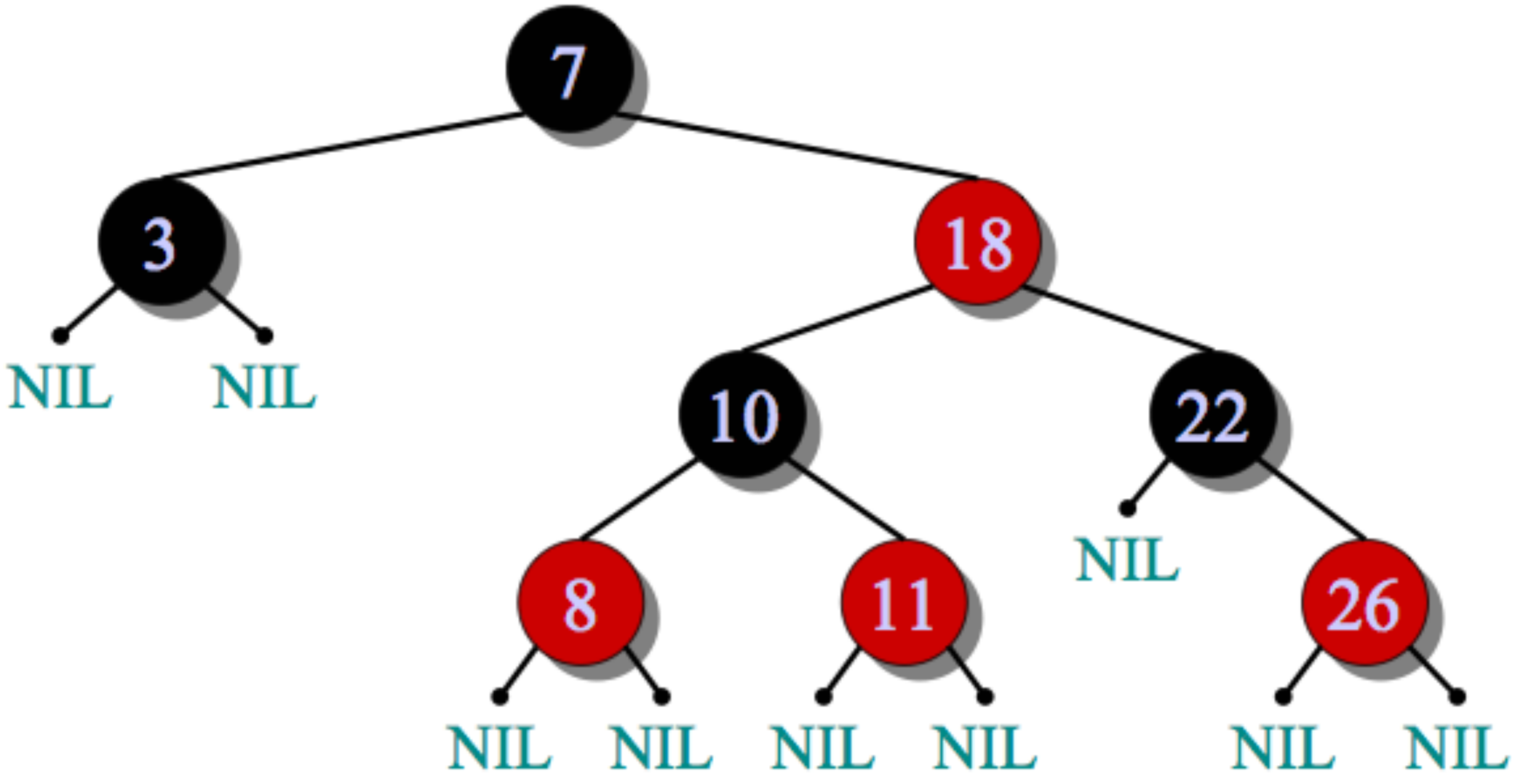
Exemplo de ABP red-black.



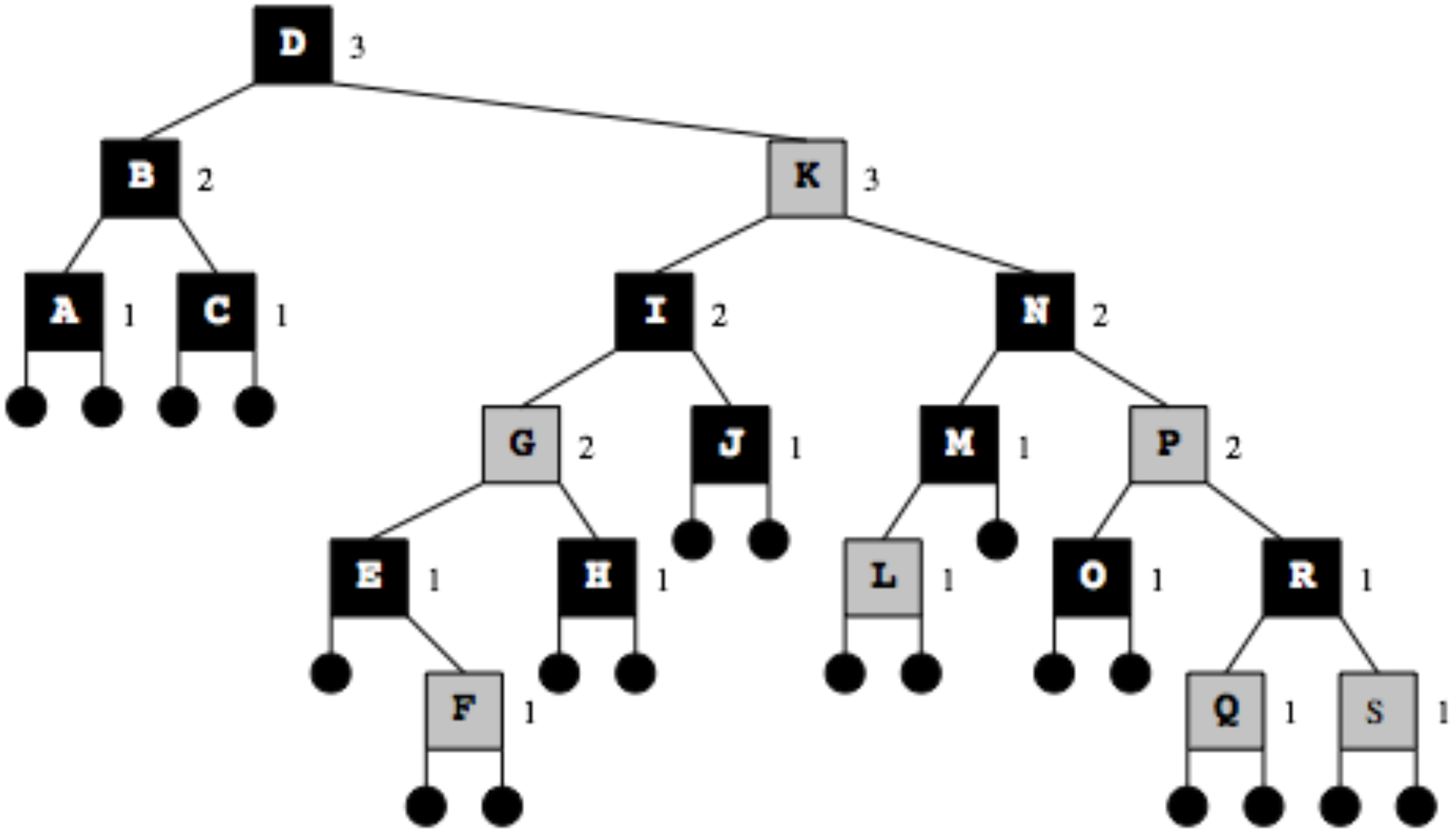
Exemplo de ABP red-black.



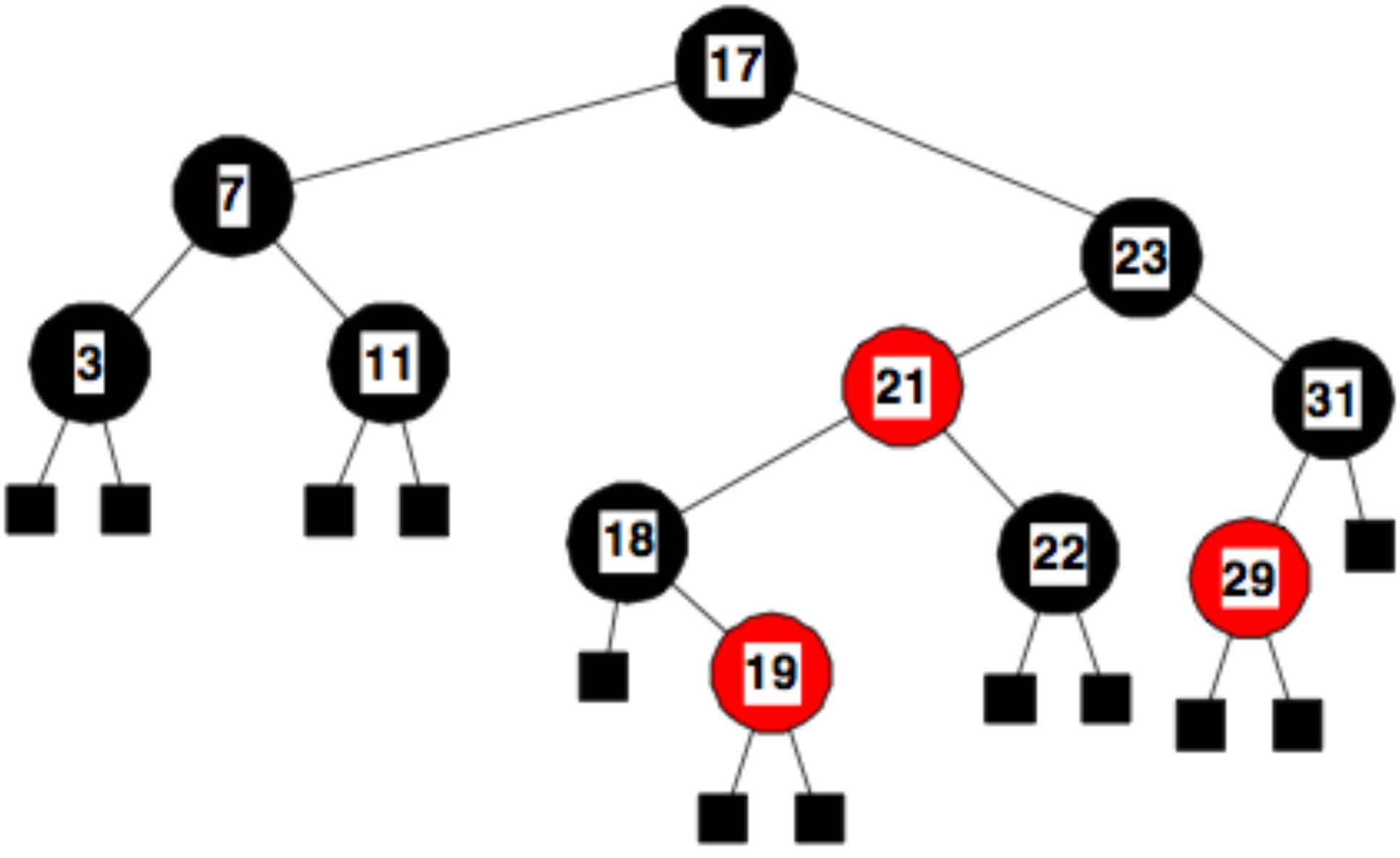
Exemplo de ABP red-black.



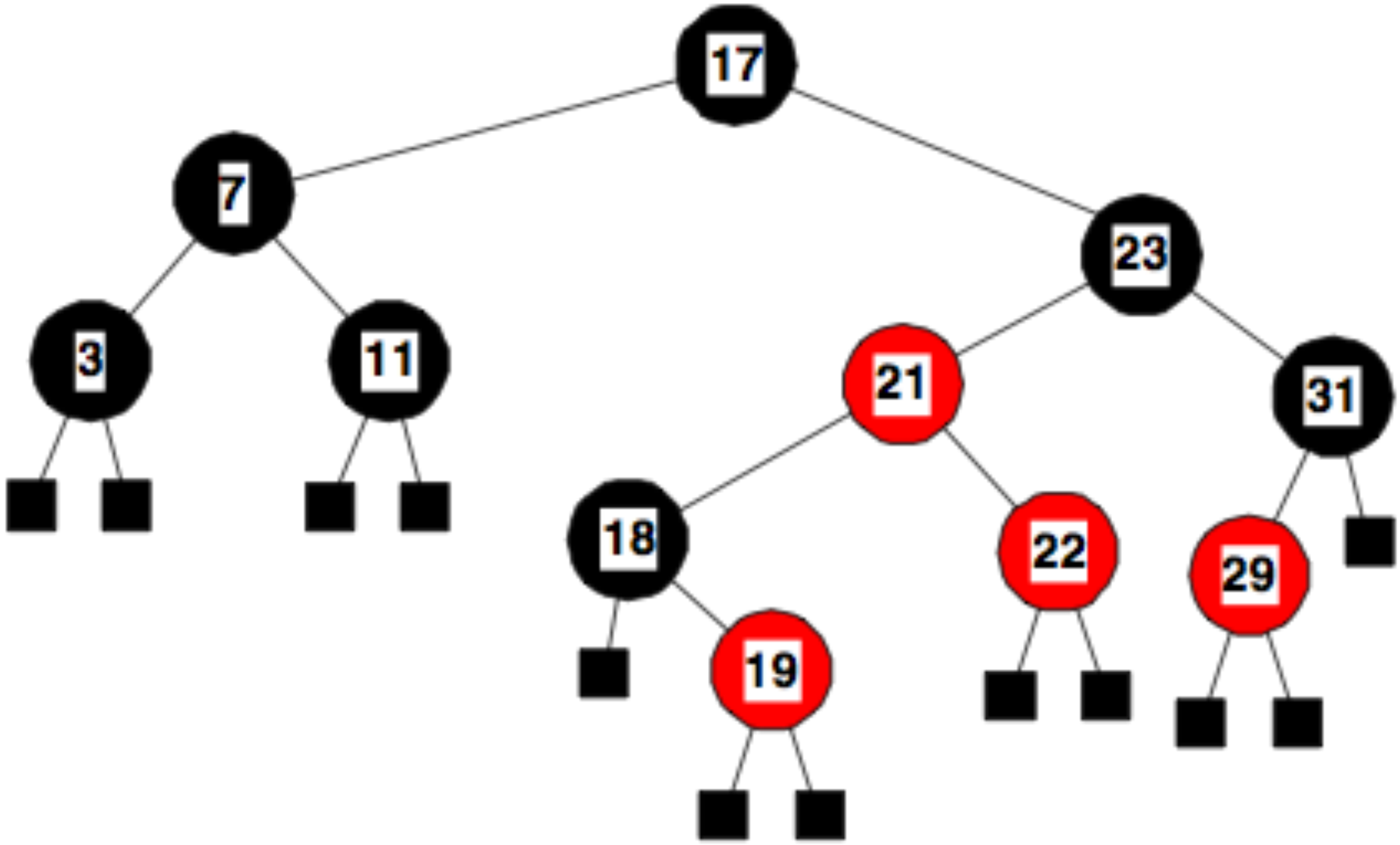
Exemplo de ABP red-black.



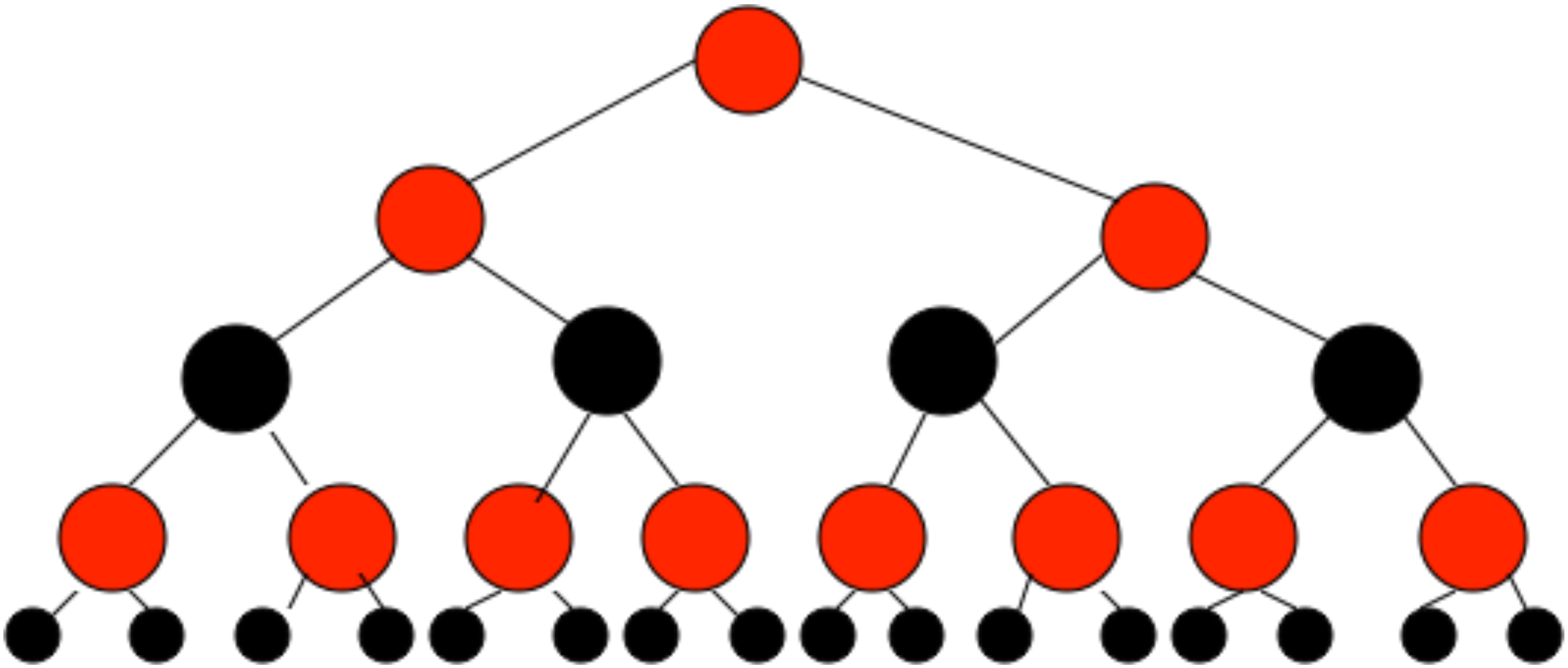
Exemplo de ABP red-black..



Não é Red-Black.



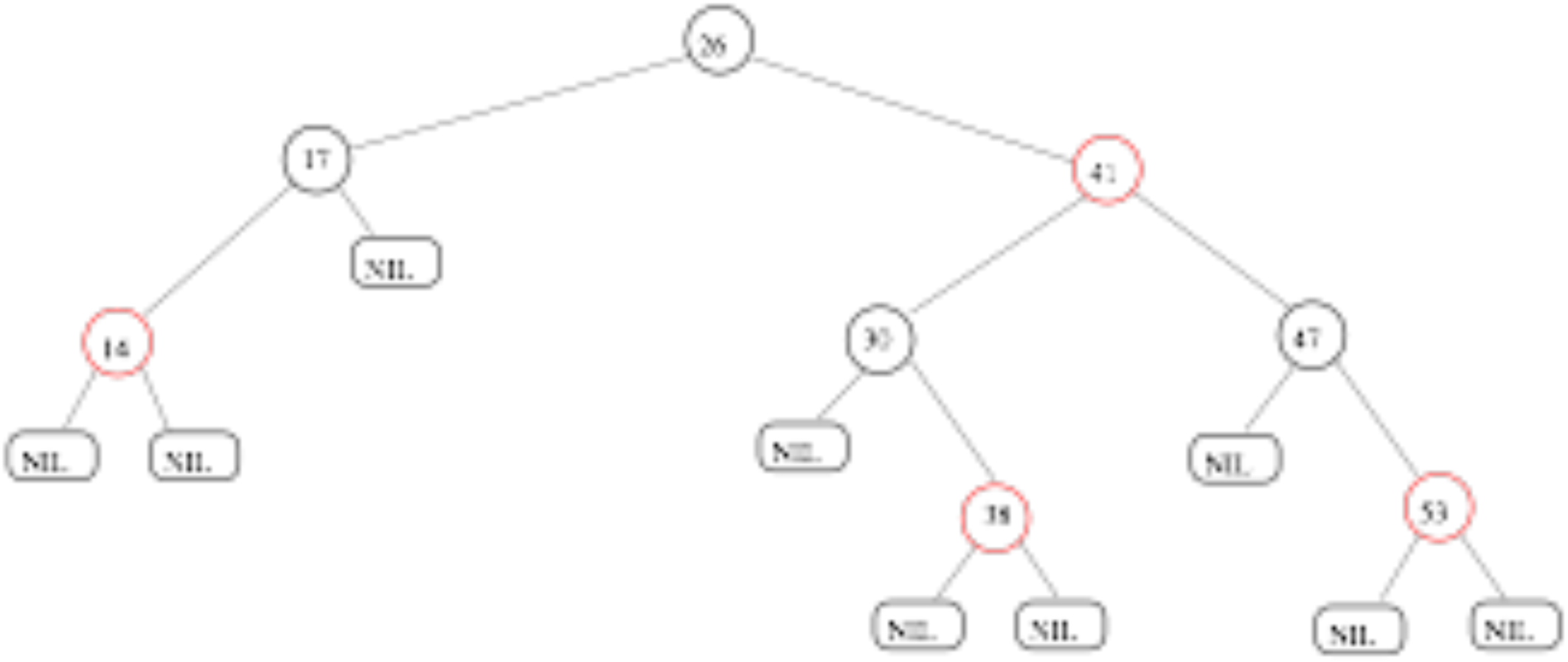
Não é Red-Black.



Observações.

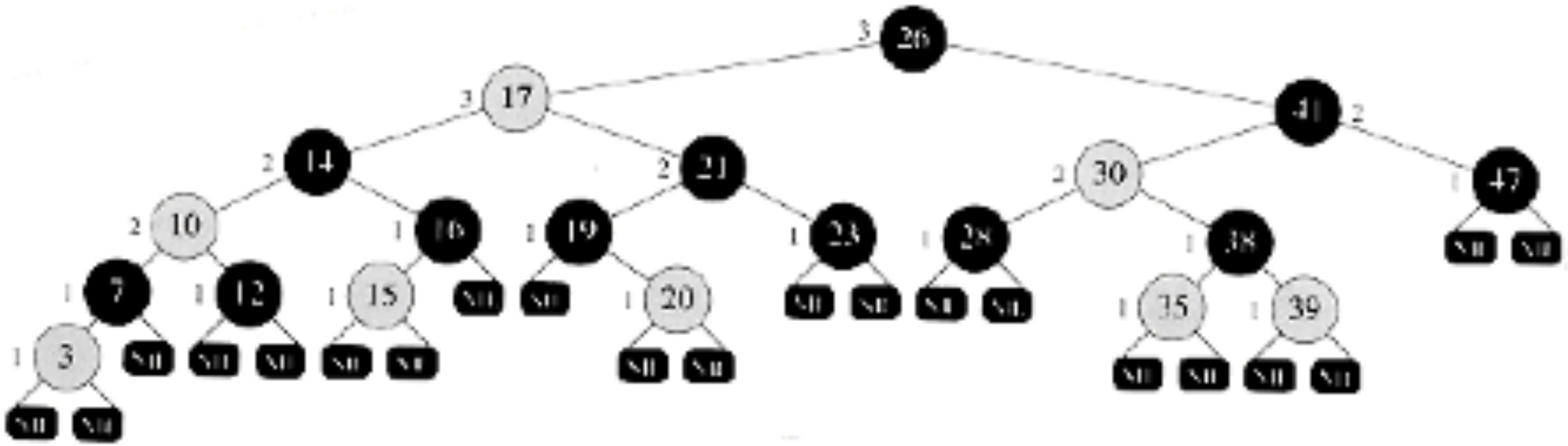
- Árvore Red-Black (**ARB**) é uma árvore de pesquisa binária com um campo extra por nó para armazenamento de sua cor (Vermelho ou Preto).
- Pode ser gasto apenas um bit para armazenar esta informação.
- Se um filho ou um pai de um nó não existe, o ponteiro correspondente contém o endereço do nó nulo (Preto).
- Nós nulos → nós externos (folhas) ou o pai do nó raiz.
- Nós normais (com chave) → nós internos da árvore.

Exemplo de ABP red-black.



NIL[T]: Sentinela para todas as folhas e pai da raiz

Exemplo de ABP red-black.



Conceitos.

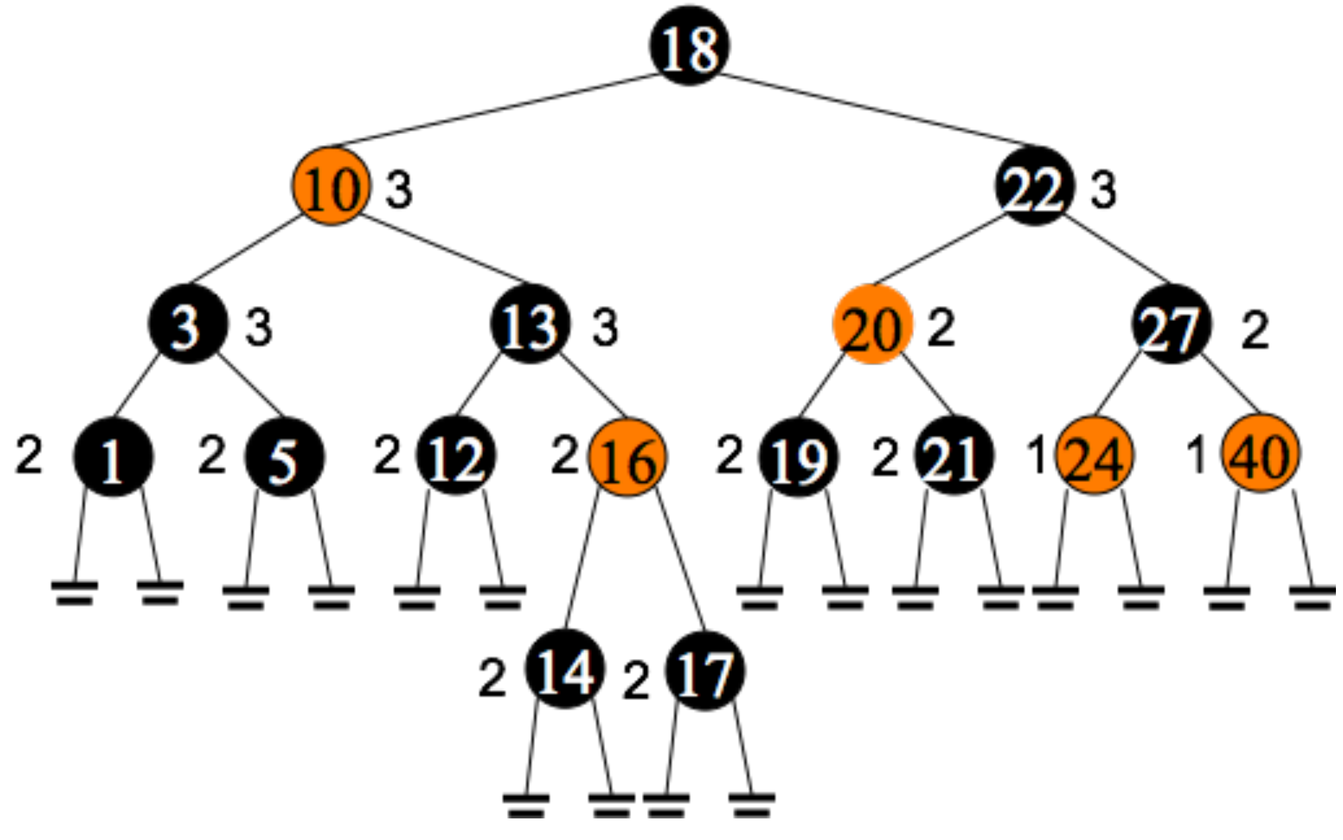
- A altura de qualquer árvore vermelho-preta é logarítmica no número de chaves armazenadas
- A busca nas árvores vermelho-preta tem complexidade logarítmica.
- Uma **ARB** impede que uma subárvore fique com mais do que o dobro da altura da outra subárvore de um nó.

Conceitos.

- A maneira como os nós podem ser coloridos é restringida.
  - Para garantir que nenhum dos caminhos da raiz até cada folha seja mais que duas vezes o comprimento de qualquer outro caminho.
- Todos os nós têm a mesma altura preta.
- Altura-Preta de um nó é a quantidade de nós pretos deste nó até uma folha (nulo) de qualquer caminho descendente.
- A árvore red-black (**ARB**) é aproximadamente balanceada.

Altura Negra

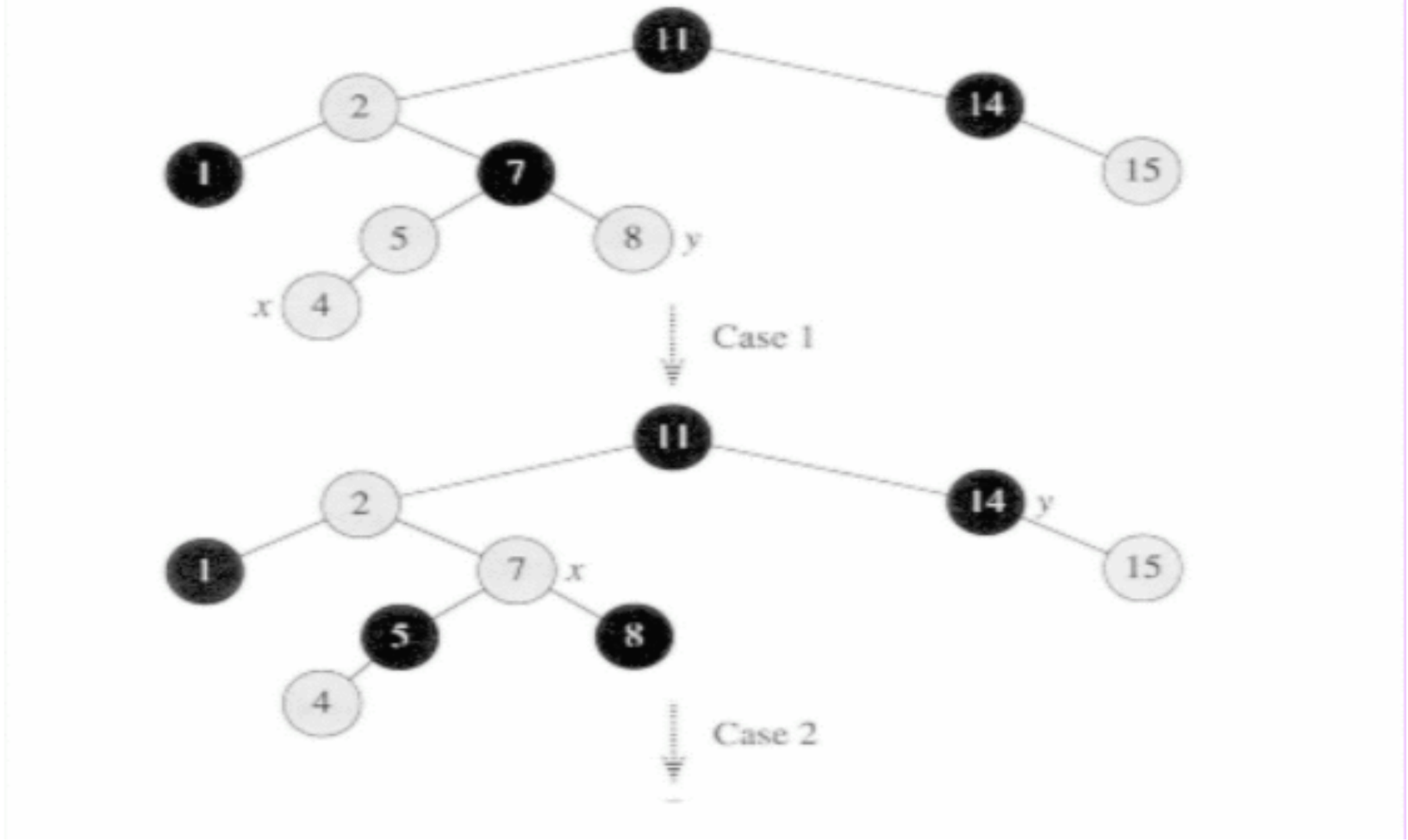
- Altura negra de um nó
  - Número de nós negros encontrados até qualquer nó folha (nulo) descendente



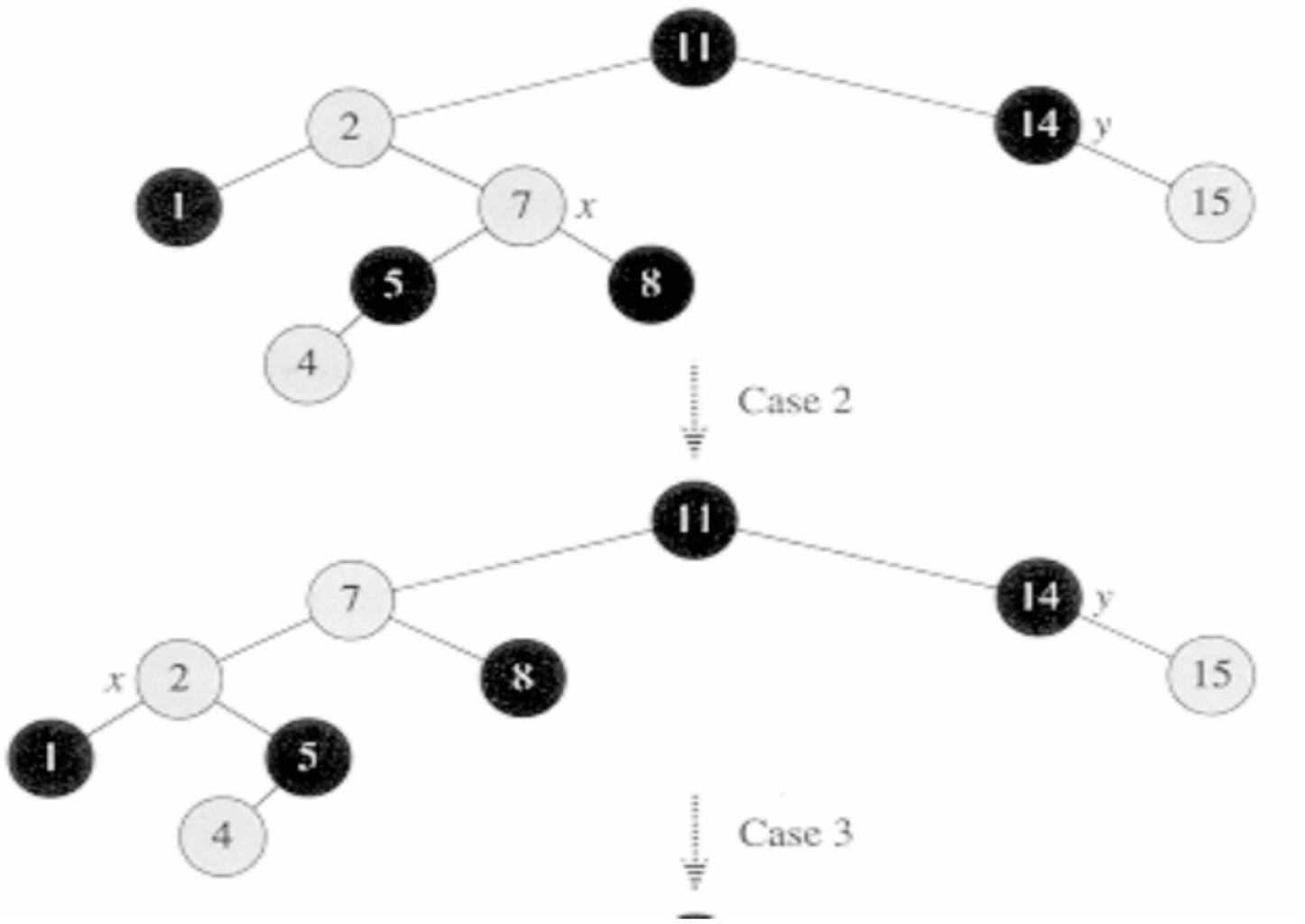
Inserção.

- A inserção de um novo nó (apontado por) X em uma árvore red-black é feita da seguinte forma:
  - Passo 1 → Insere-se o nó X como se faz normalmente em uma árvore binária de pesquisa.
  - Passo 2 → Colore-se o nó X de vermelho (para não ferir a propriedade 5 (altura preta)).
  - Passo 3 → Verifica-se se o mesmo está ferindo a propriedade 4 (se um nó é vermelho então seus dois filhos são pretos).
    - Caso afirmativo ---> aplica-se as soluções expressas nos três casos distintos detalhados a seguir.
  - Passo 4 → Colore-se a raiz de preto.

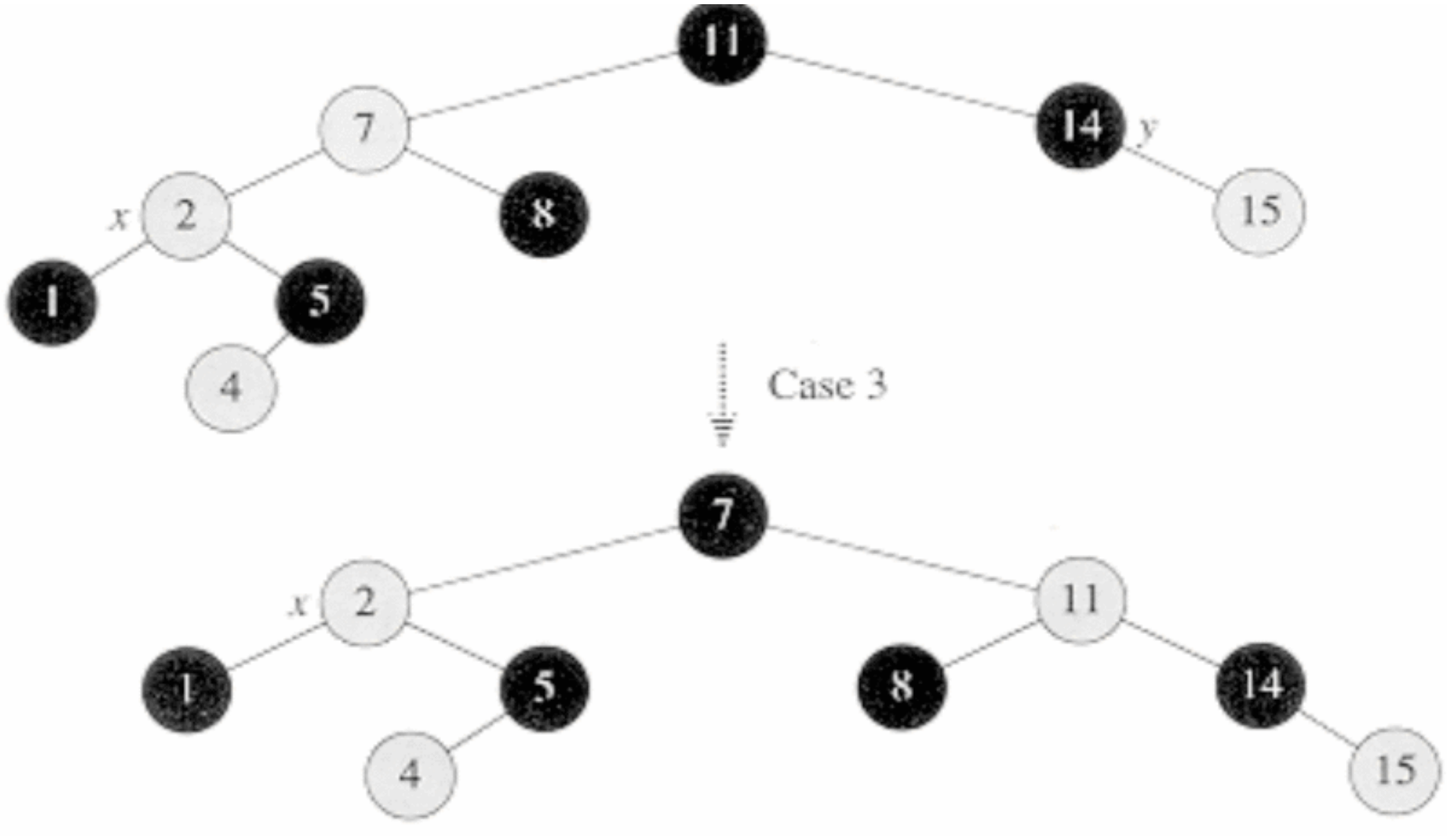
- Caso 1: O tio (Y) de X é vermelho
  - Solução: Troca-se a cor do pai e do tio de X para preta, a cor do avô para vermelha e transfere-se o ponteiro X para o seu avô, voltando-se ao início do passo 3.



- Caso 2: O tio (Y) de X é preto e X é o sobrinho mais próximo
  - Solução: Transfere-se o ponteiro X para seu pai e faz-se uma rotação no novo X para o lado oposto ao antigo X, transformando em Caso 3.



- Caso 3: O tio de X é preto e X é seu sobrinho mais distante
  - Solução: Troca-se a cor do pai de X para preta, a cor do avô para vermelha e é feita uma rotação neste último (avô) para o lado oposto a X, resolvendo-se o problema.



```
RB-INSERT-FIXUP(T, z)
 1  while cor[p[z]] = VERMELHO
 2        do if p[z] = esquerda[p[p[z]]]
 3              then y ← direita[p[p[z]]]
 4                   if cor[y] ← VERMELHO
 5                     then cor[p[z]] ← PRETO                    ▷ Caso 1
 6                          cor[y] ← PRETO                       ▷ Caso 1
 7                          cor[p[p[x]]] ← VERMELHO              ▷ Caso 1
 8                               z ← p[p[z]]                     ▷ Caso 1
 9                     else  if z = direita[p[z]]
10                               then z ← p[z]                   ▷ Caso 2
11                                    LEFT-ROTATE(T, z)          ▷ Caso 2
12                          cor[p[z]] ← PRETO                    ▷ Caso 3
13                               cor[p[p[z]]] ← VERMELHO         ▷ Caso 3
14                          RIGHT-ROTATE(T, p[p[z]])             ▷ Caso 3
15           else (igual a cláusula then
                          com "direita" e "esquerda" trocadas)
16  cor[raiz[T]] ← PRETO
```

Exercício

- Simule a inserção de nós numa árvore red-black vazia até completar 5 níveis. Pense os valores para as chaves aleatoriamente.
- Use uma ferramenta de desenho --> LibreOffice Draw.