Unidade VIII:
Balanceamento de Árvores Binárias

Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação

Balanceamento de Árvores

- Qual é a vantagem de uma árvore estar balanceada?

Balanceamento de Árvores

- Qual é a vantagem de uma árvore estar balanceada?
  - Resposta: eficiência em termos de pesquisa, inserção e remoção
- Inicialmente, toda árvore é balanceada e elas podem desbalancear após as operações de inserção e remoção

Ideia Básica do Balanceamento de Árvores

- As árvores desbalanceadas para a esquerda devem ser rotacionadas para a direita e as para a direita, para a esquerda

Tipos de Rotação

- Rotação simples à esquerda
- Rotação simples à direita
- Rotação dupla direita – esquerda
- Rotação dupla esquerda - direita

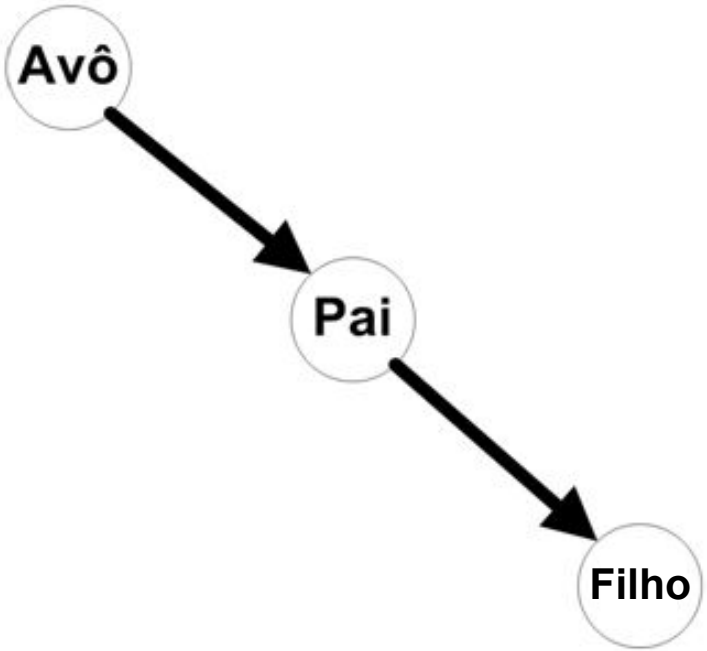
Tipos de Rotação

- Rotação simples à esquerda
- Rotação simples à direita
- Rotação dupla direita – esquerda
- Rotação dupla esquerda - direita

Quando usar cada uma delas e como fazer ?

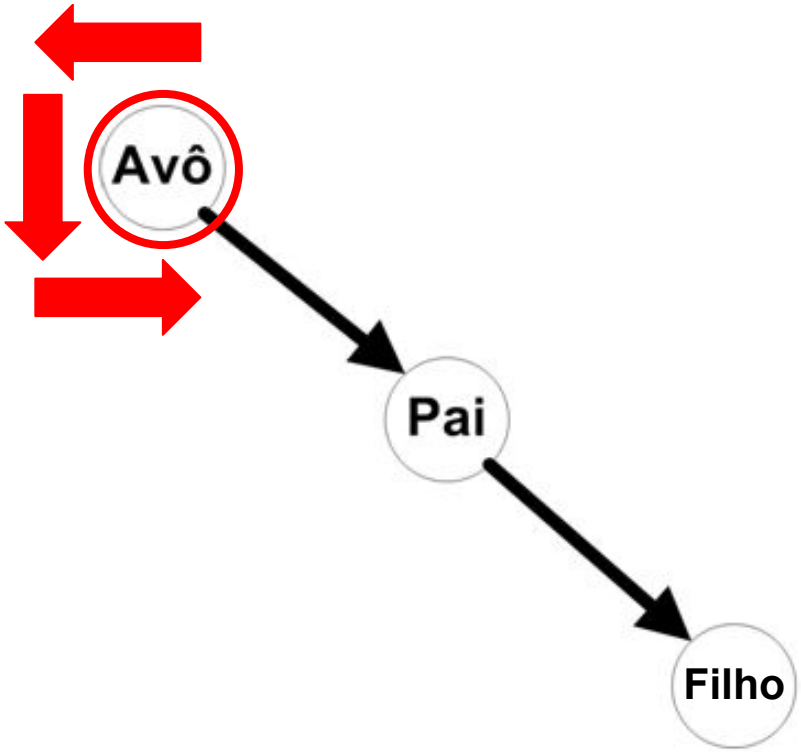
Rotação Simples à Esquerda

- Usada em subárvores em que o pai e o filho estão desbalanceados para a direita



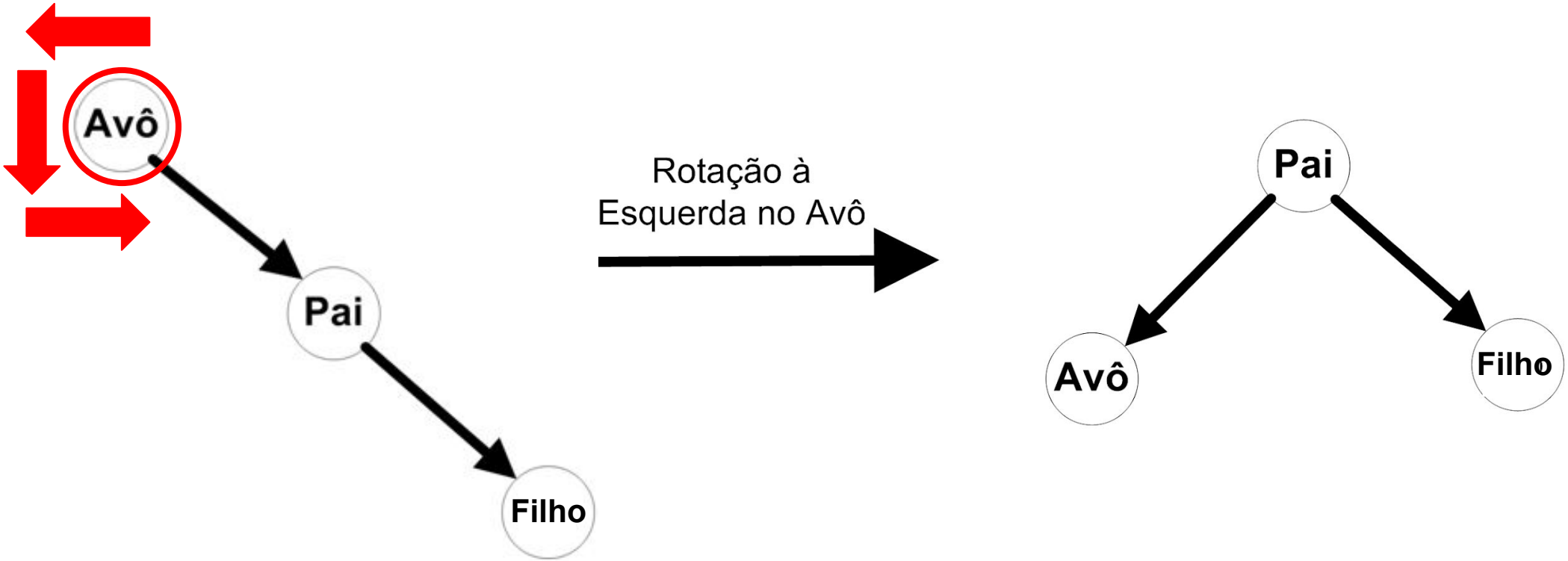
Rotação Simples à Esquerda

- Usada em subárvores em que o pai e o filho estão desbalanceados para a direita

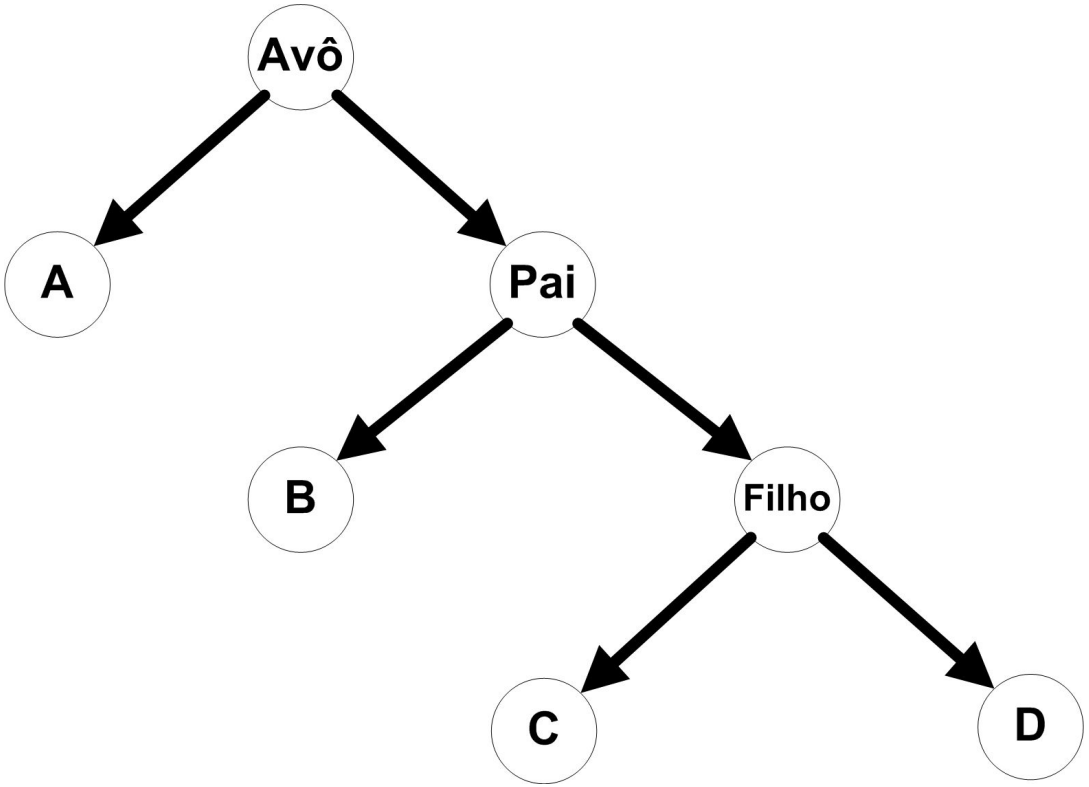


Rotação Simples à Esquerda

- Usada em subárvores em que o pai e o filho estão desbalanceados para a direita



Implementação da Rotação à Esquerda

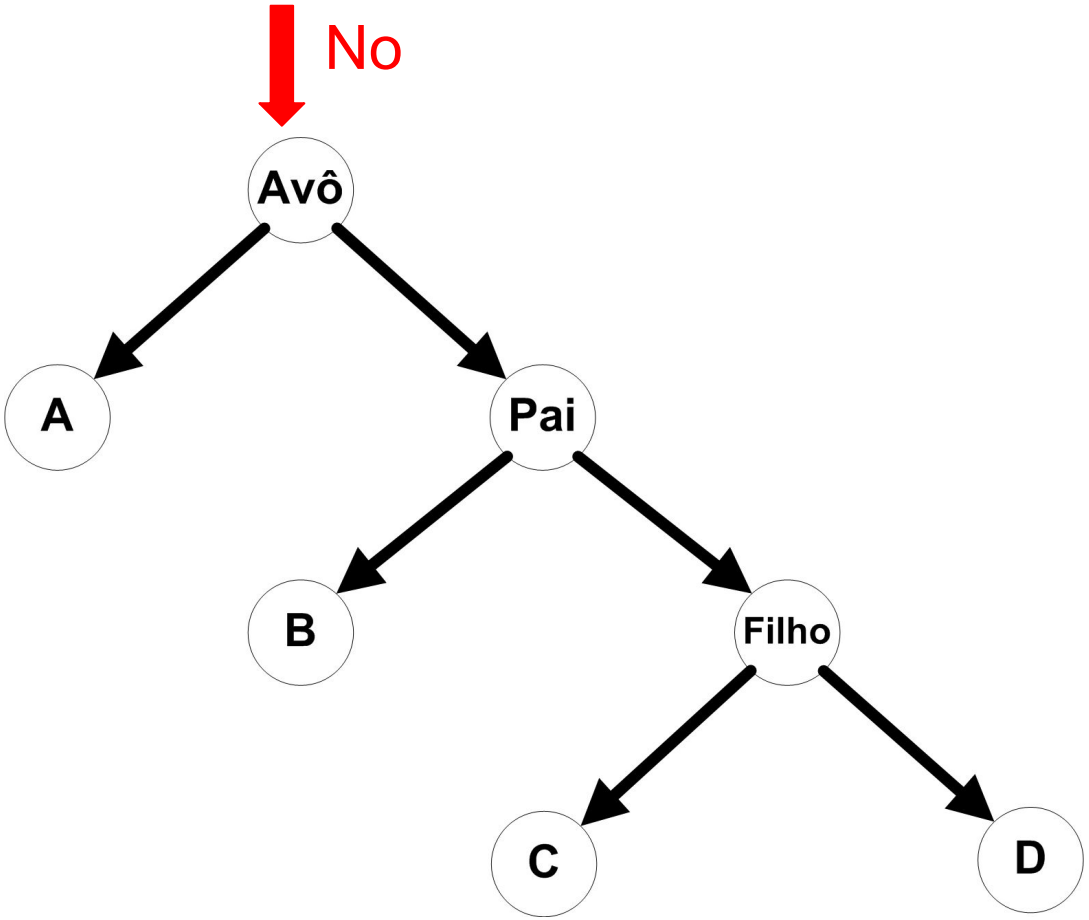



```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

Implementação da Rotação à Esquerda

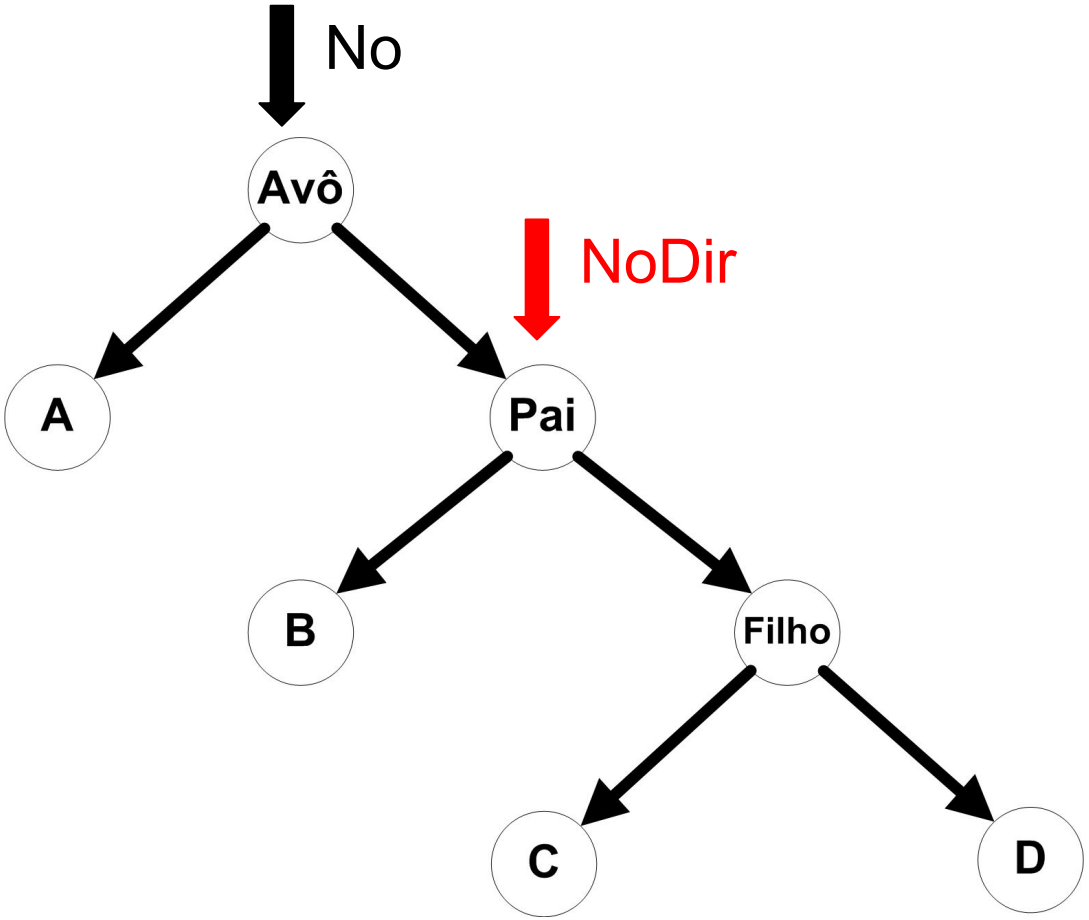



```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

Implementação da Rotação à Esquerda

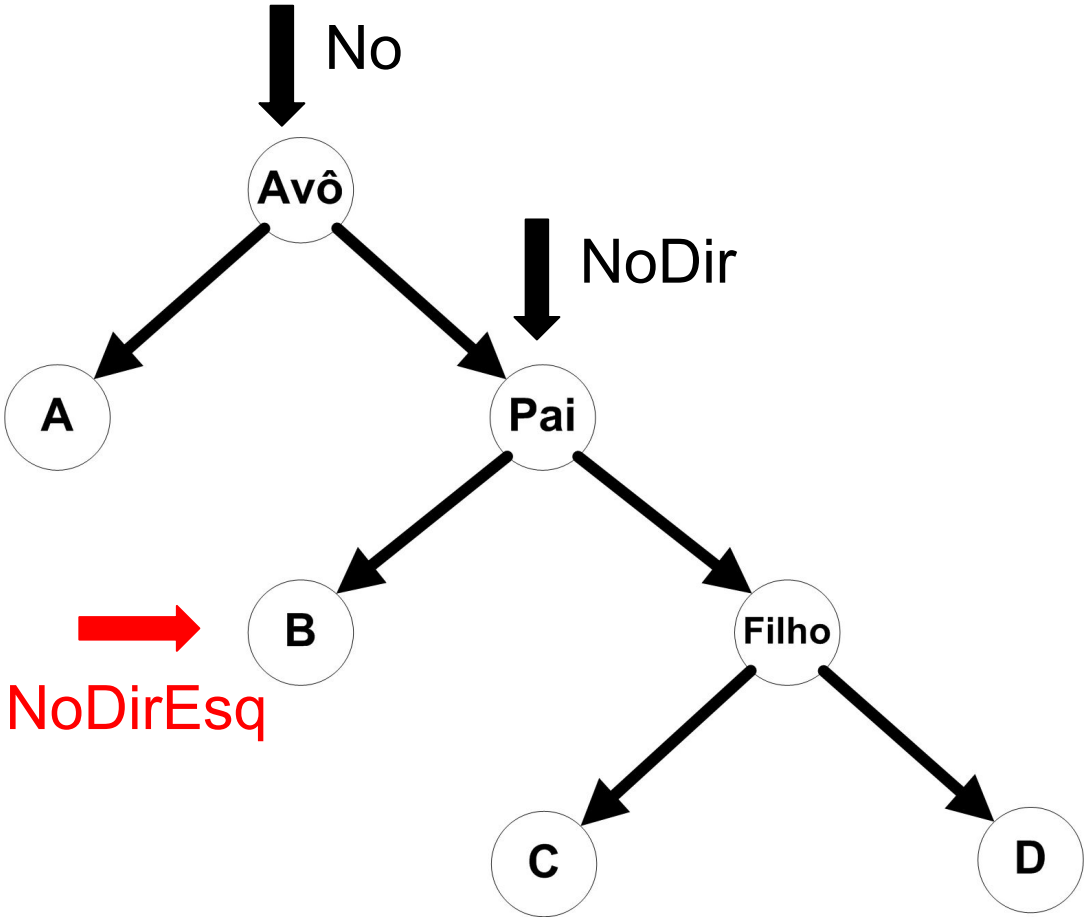



```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

Implementação da Rotação à Esquerda

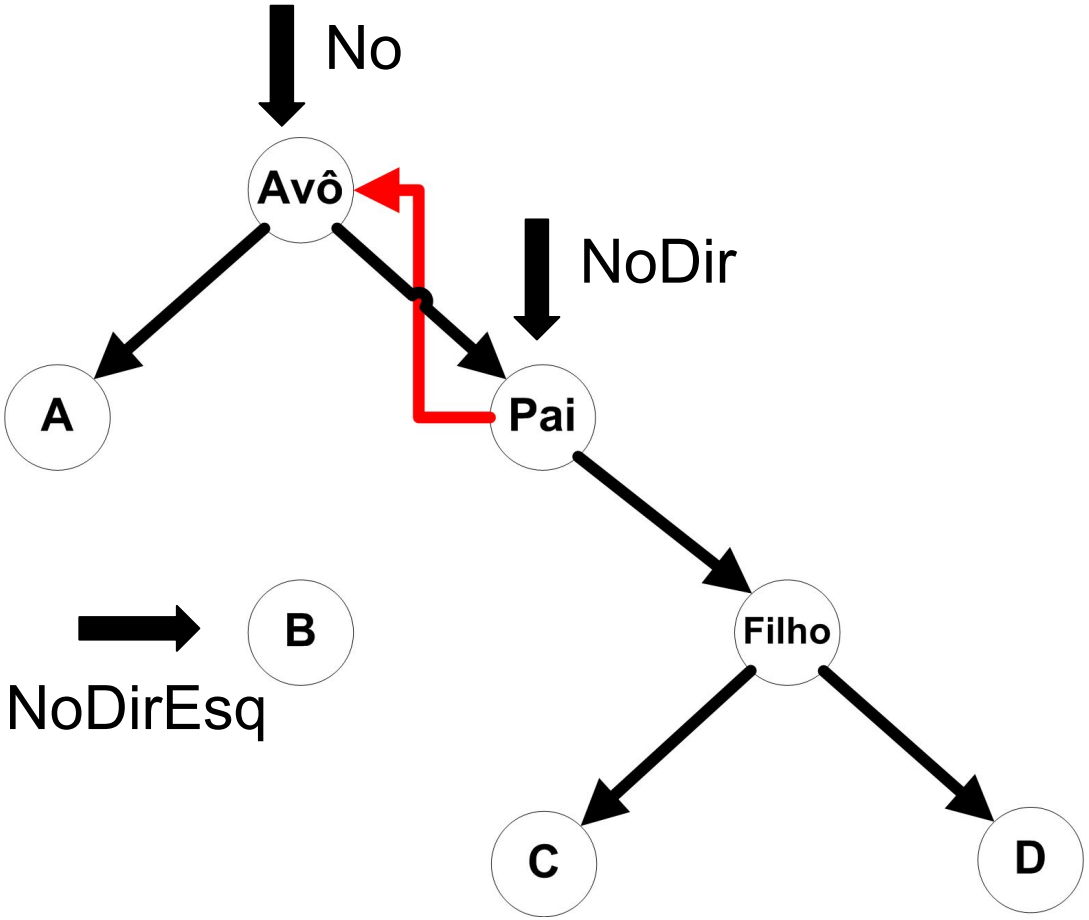



```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

Implementação da Rotação à Esquerda

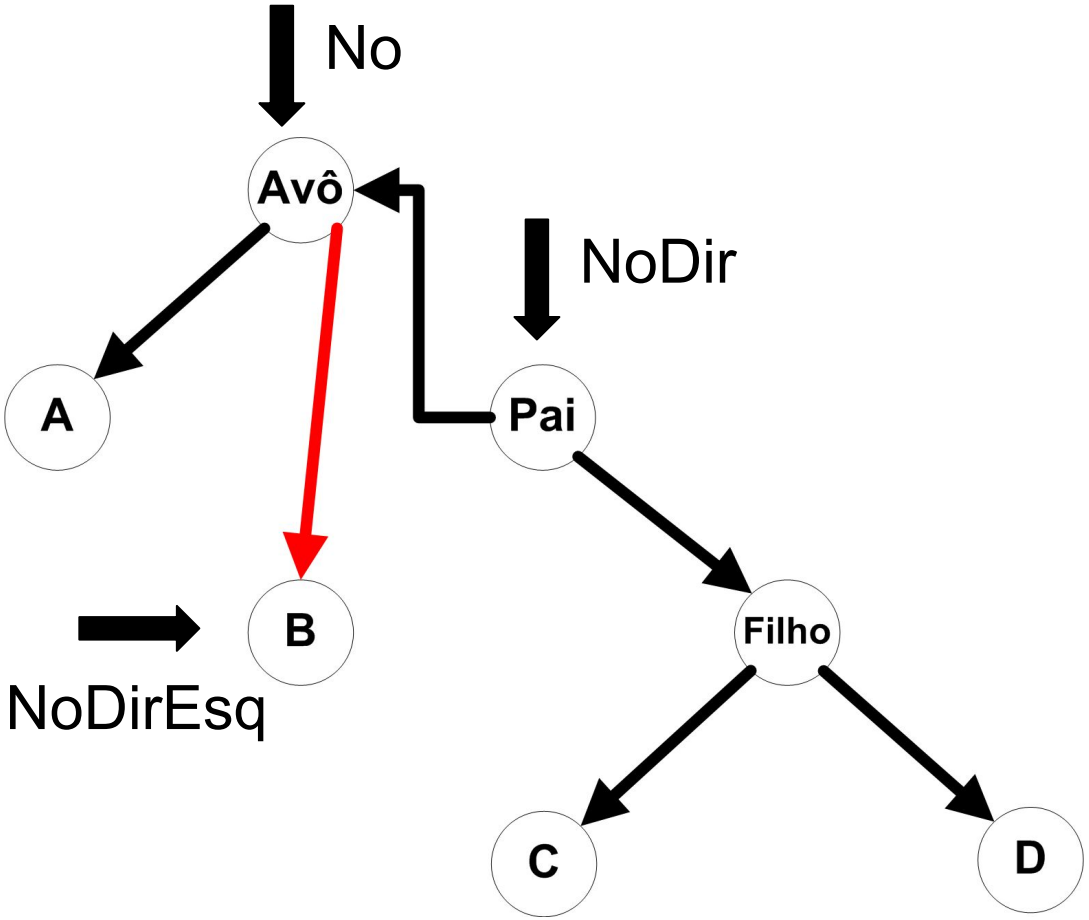



```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

Implementação da Rotação à Esquerda

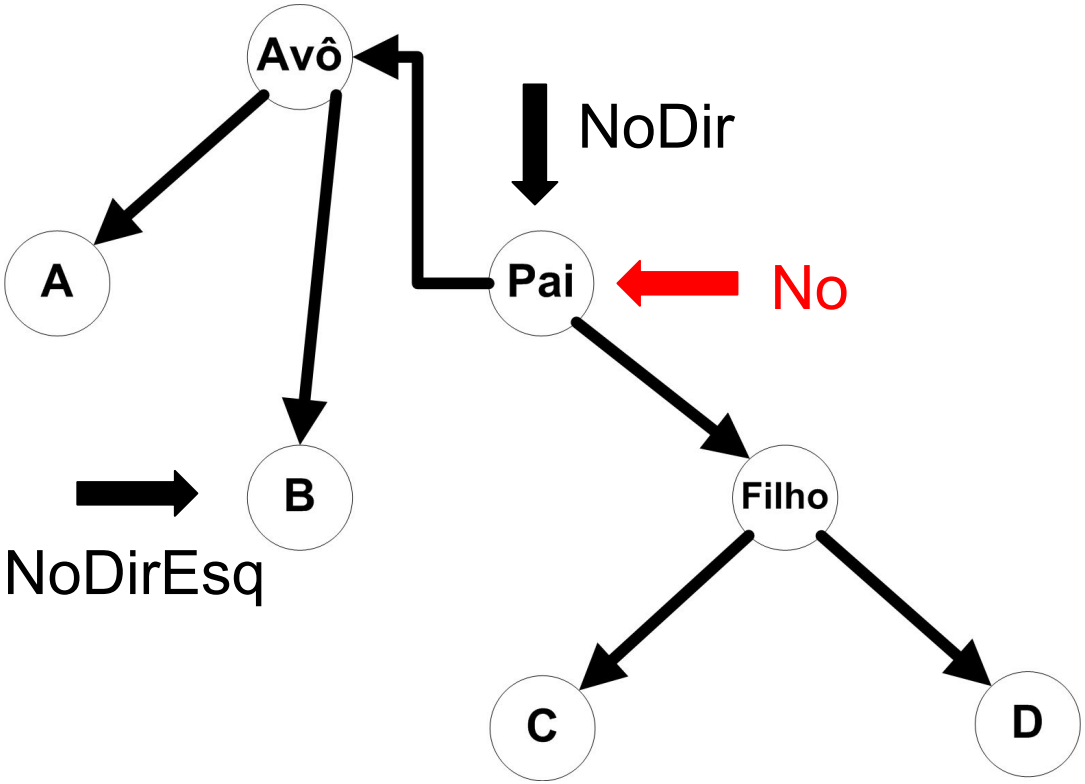



```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

Implementação da Rotação à Esquerda



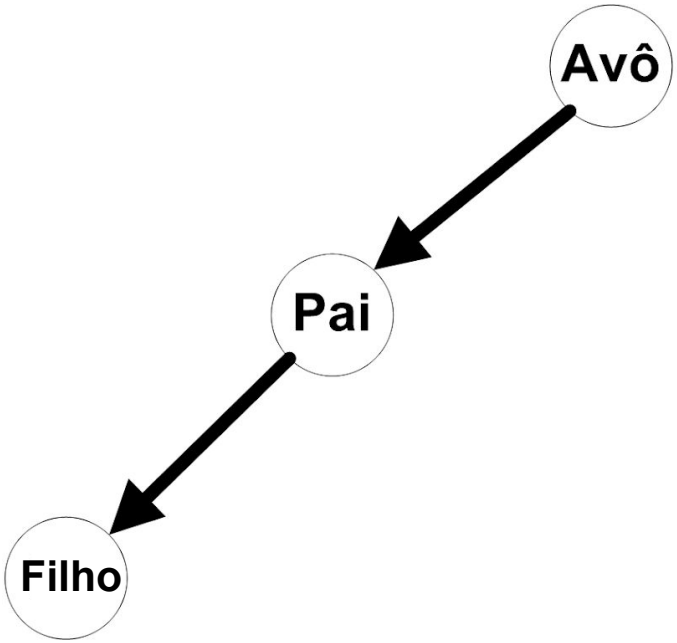

```
No rotacionarEsq (No no) {
    No noDir = no.dir;
    No noDirEsq = noDir.esq;

    noDir.esq = no;
    no.dir = noDirEsq;

    return noDir;
}
```

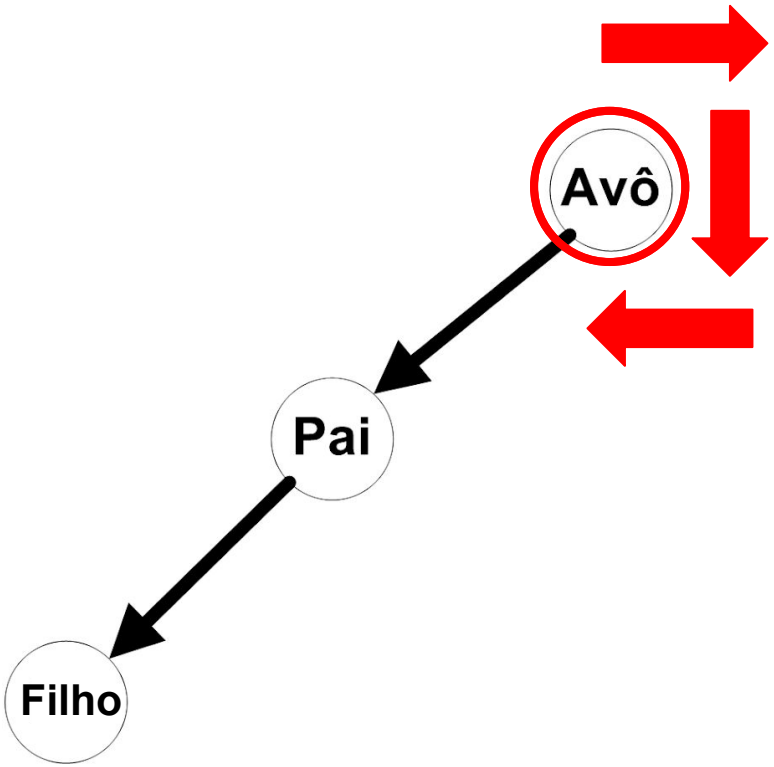
Rotação Simples à Direita

- Usada em subárvores em que o pai e o filho estão desbalanceados para a esquerda



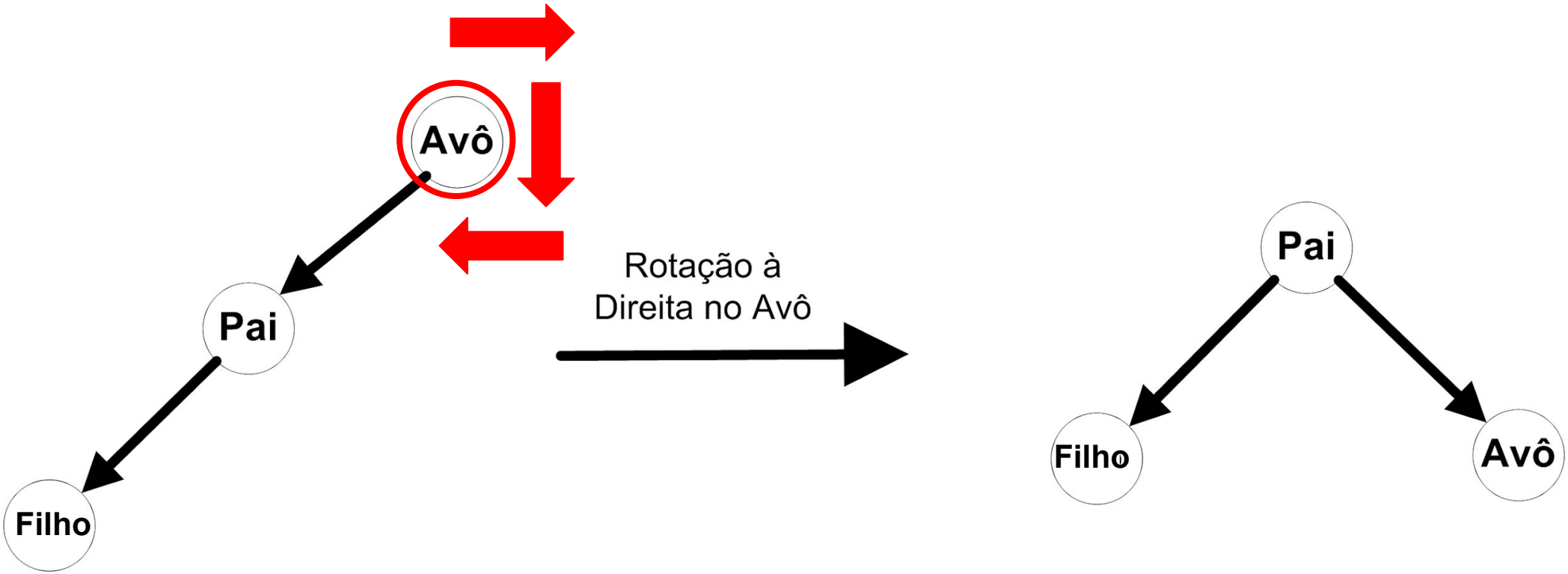
Rotação Simples à Direita

- Usada em subárvores em que o pai e o filho estão desbalanceados para a esquerda

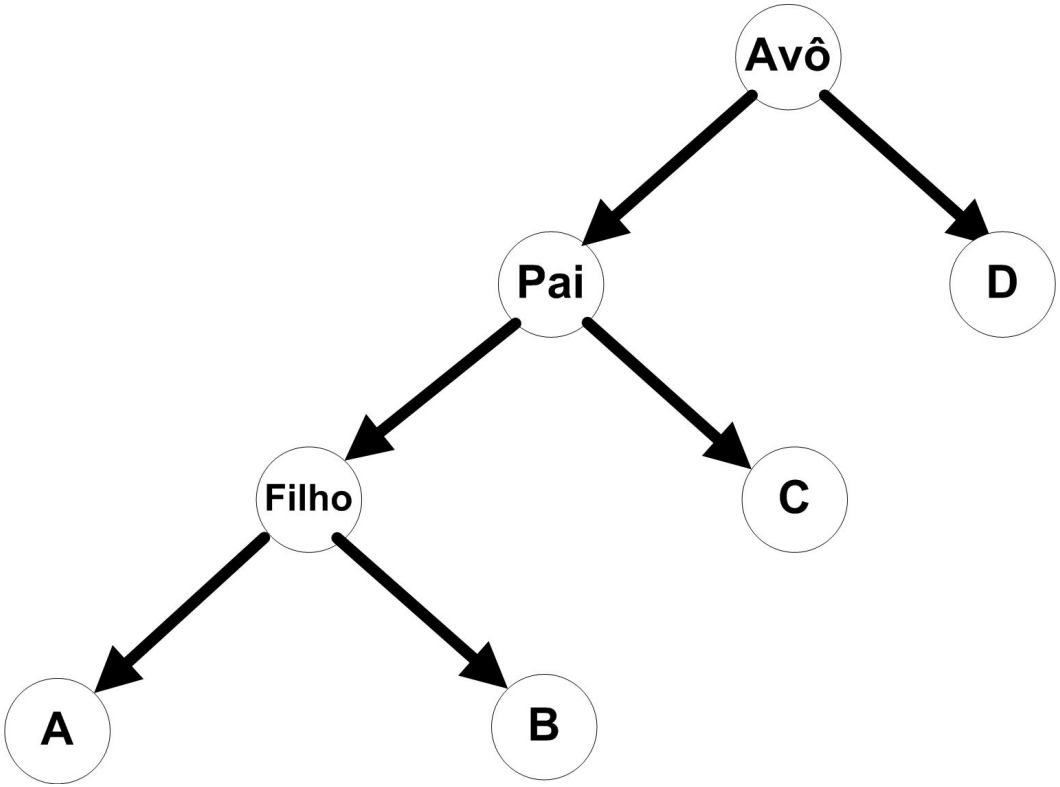


Rotação Simples à Direita

- Usada em subárvores em que o pai e o filho estão desbalanceados para a esquerda



Implementação da Rotação à Direita

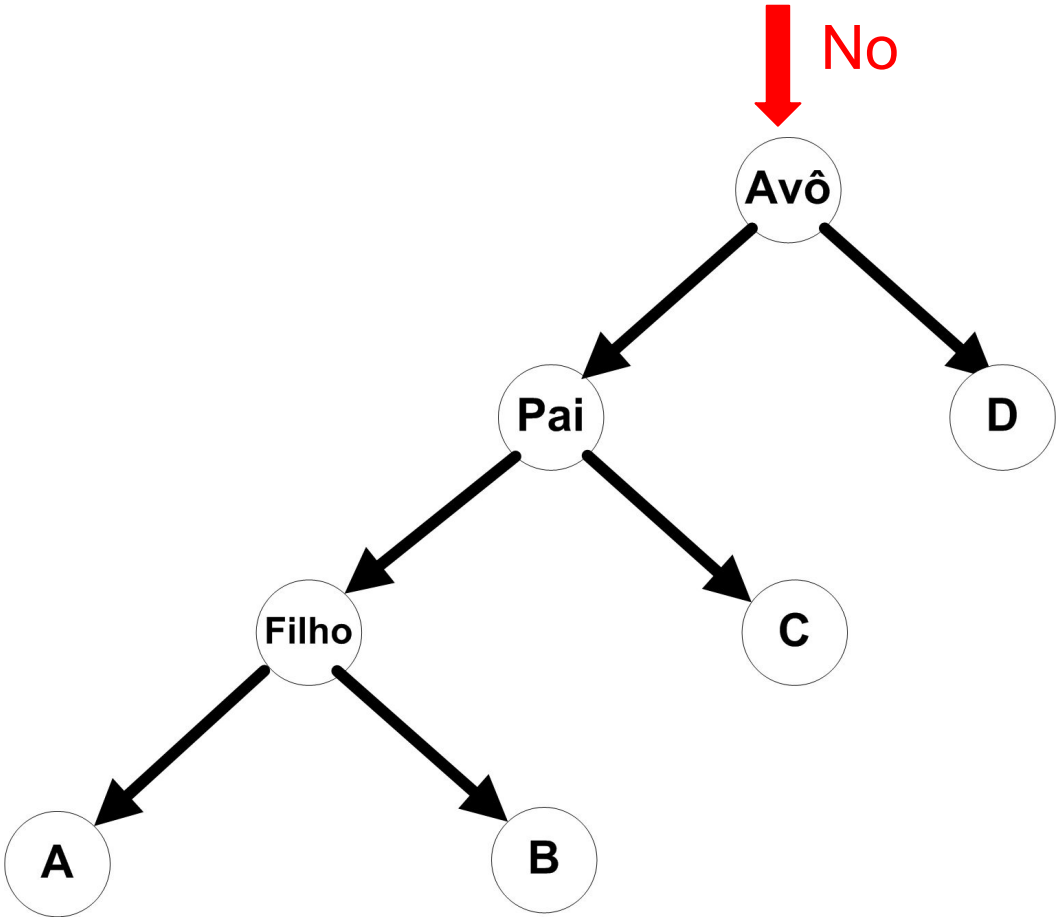



```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

Implementação da Rotação à Direita

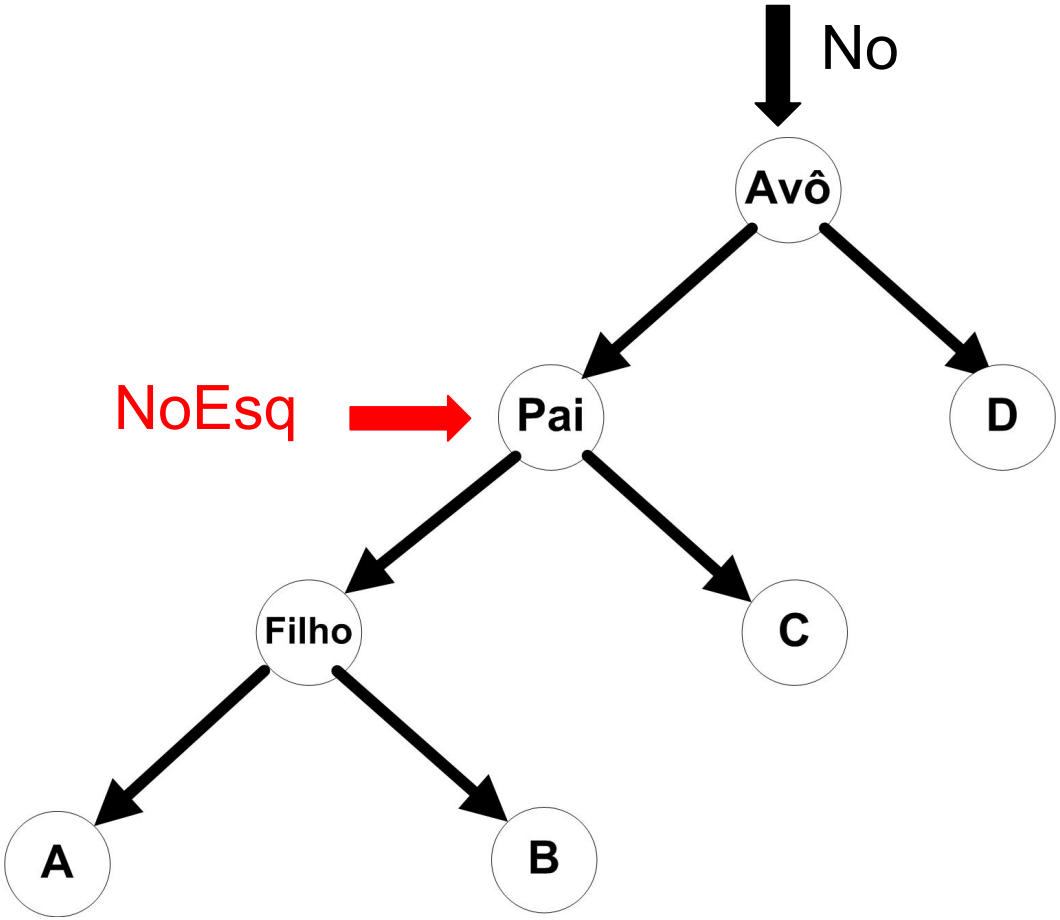



```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

Implementação da Rotação à Direita

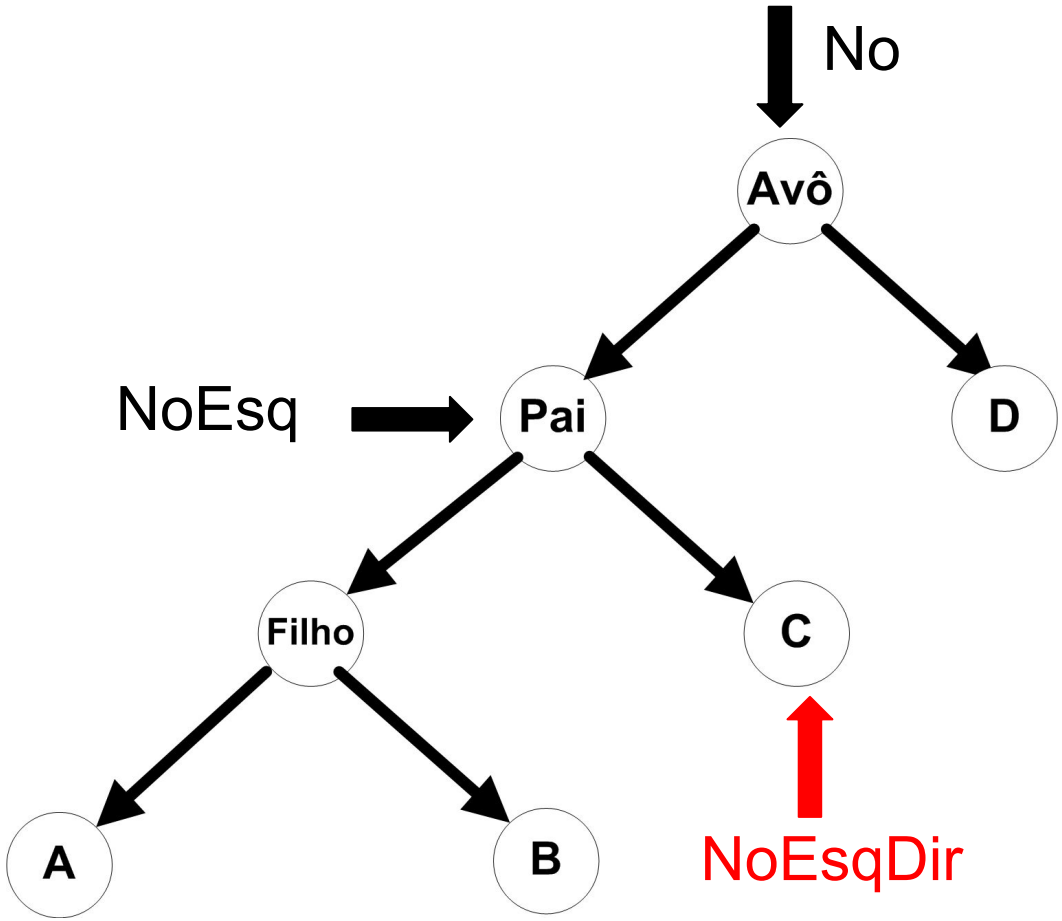



```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

Implementação da Rotação à Direita

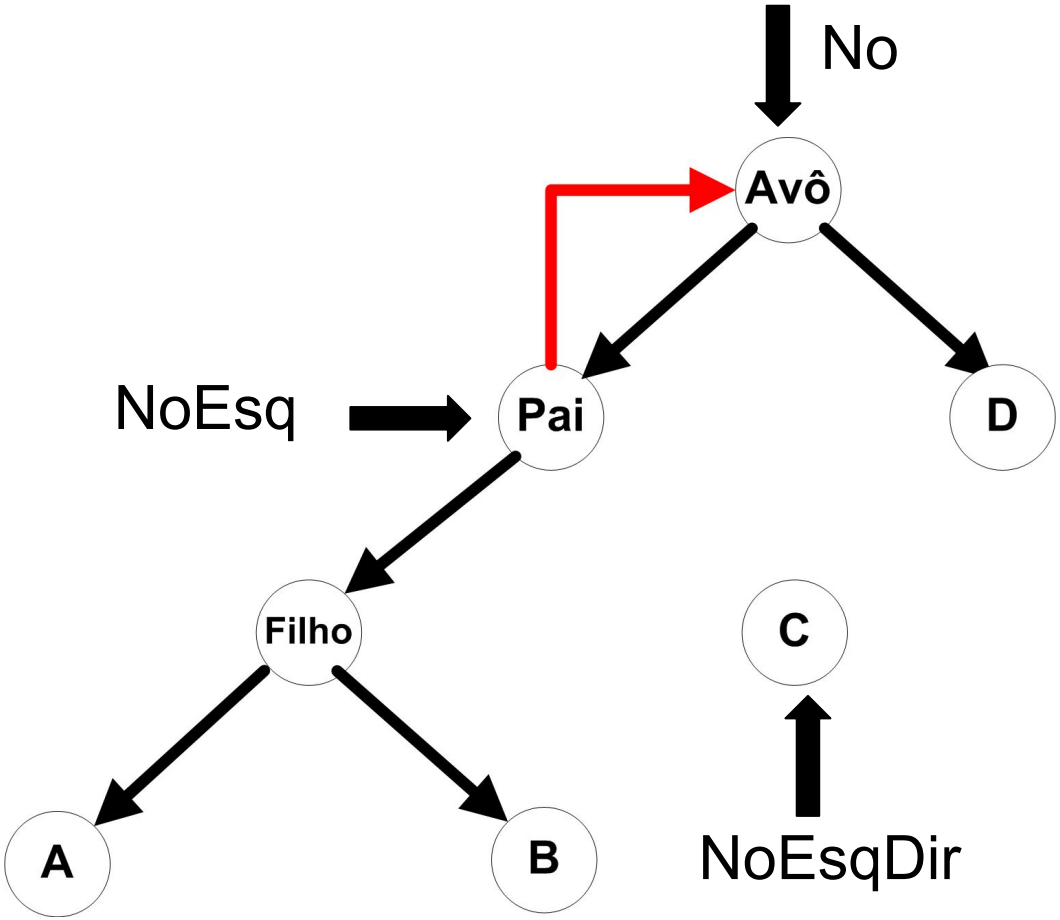



```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

Implementação da Rotação à Direita

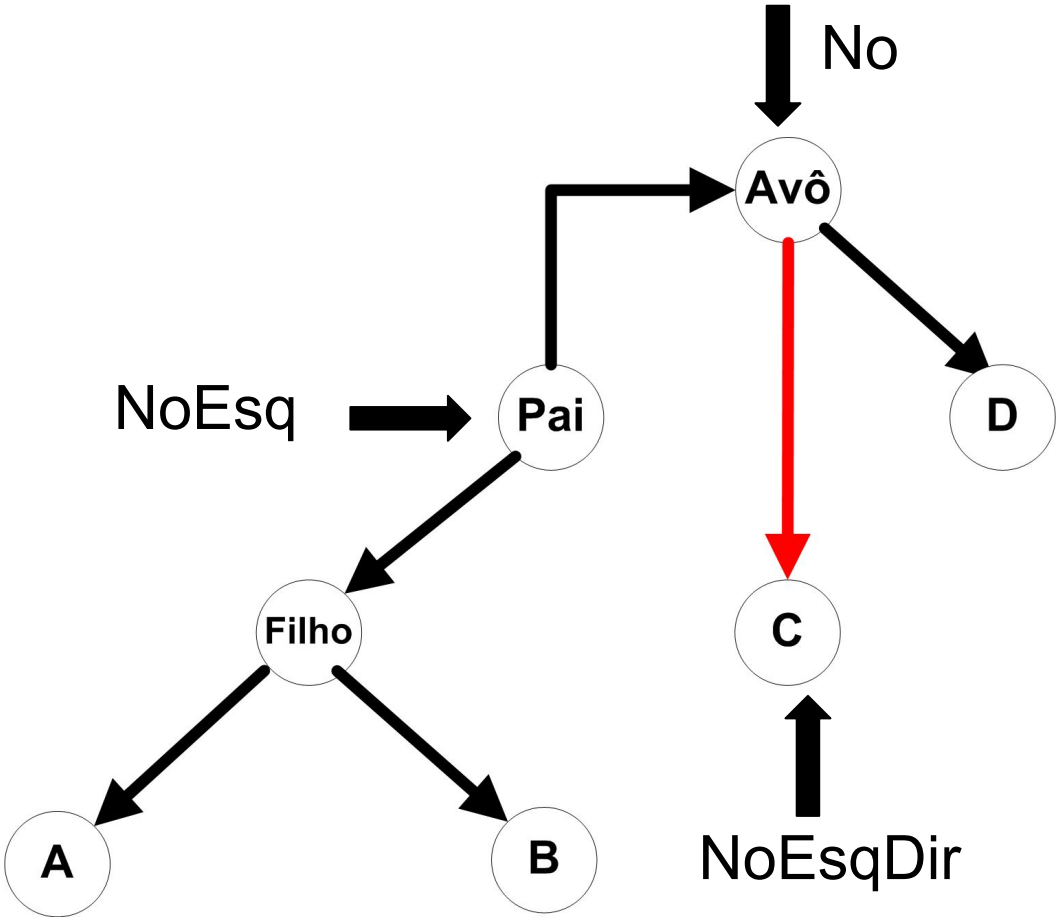



```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

Implementação da Rotação à Direita

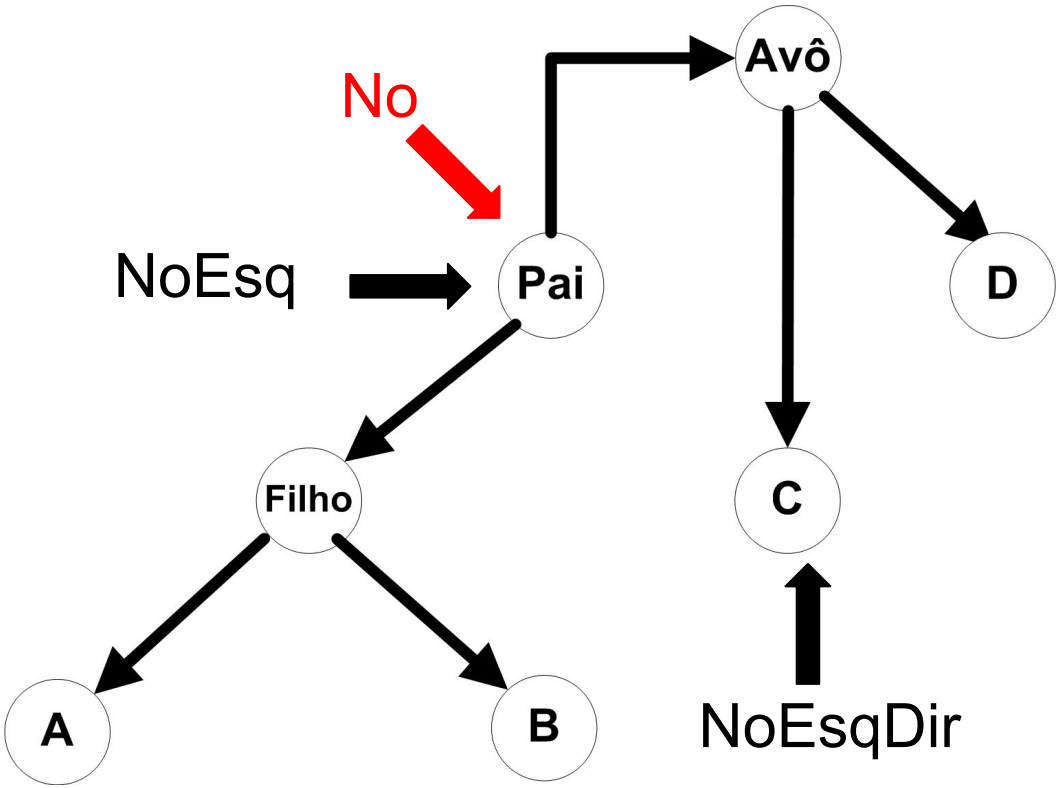



```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

Implementação da Rotação à Direita



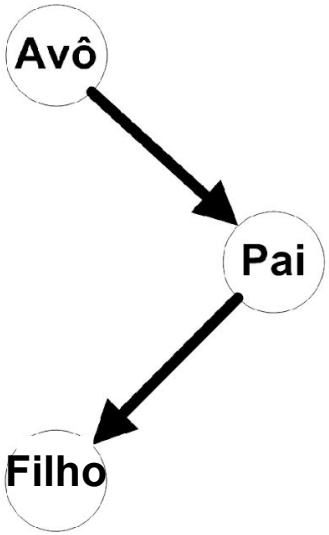

```
No rotacionarDir (No no) {
    No noEsq = no.esq;
    No noEsqDir = noEsq.dir;

    noEsq.dir = no;
    no.esq = noEsqDir;

    return noEsq;
}
```

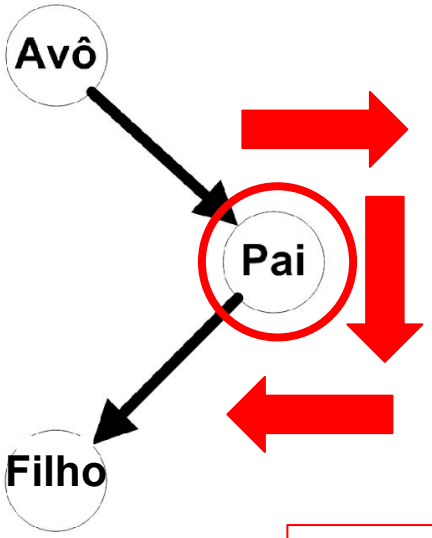
Rotação Dupla Direita – Esquerda

- Usada em subárvores em que o pai está desbalanceado para a direita e o filho para a esquerda



Rotação Dupla Direita – Esquerda

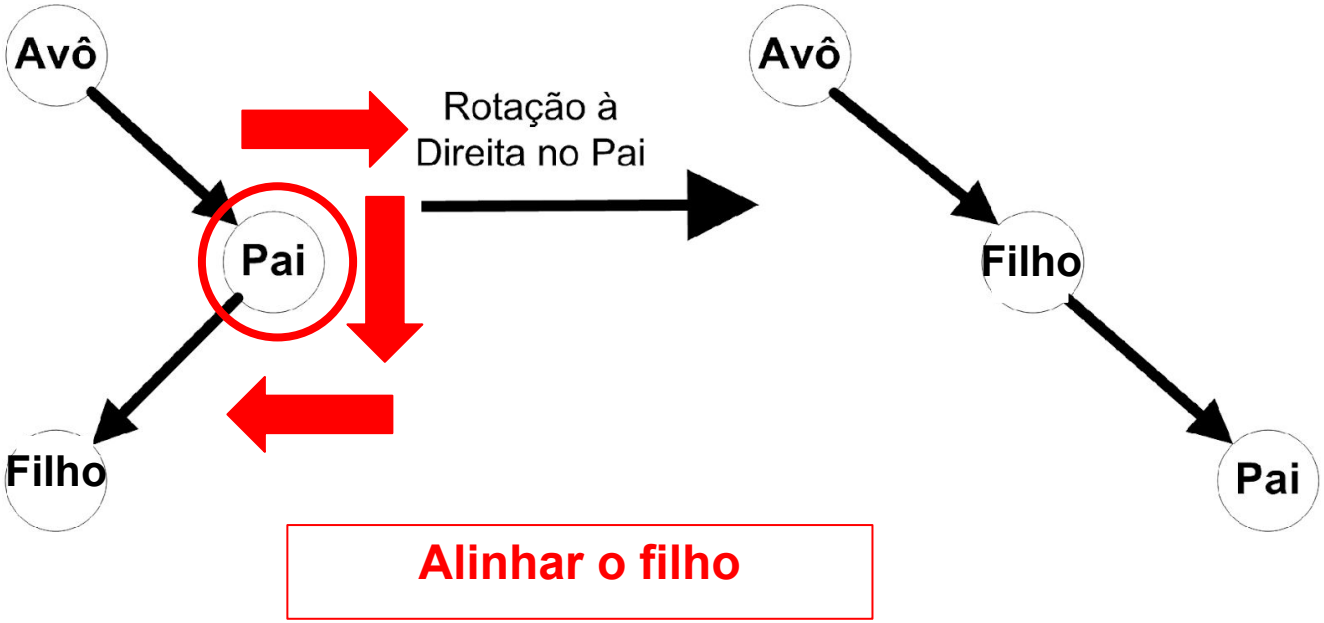
- Usada em subárvores em que o pai está desbalanceado para a direita e o filho para a esquerda



Alinhar o filho

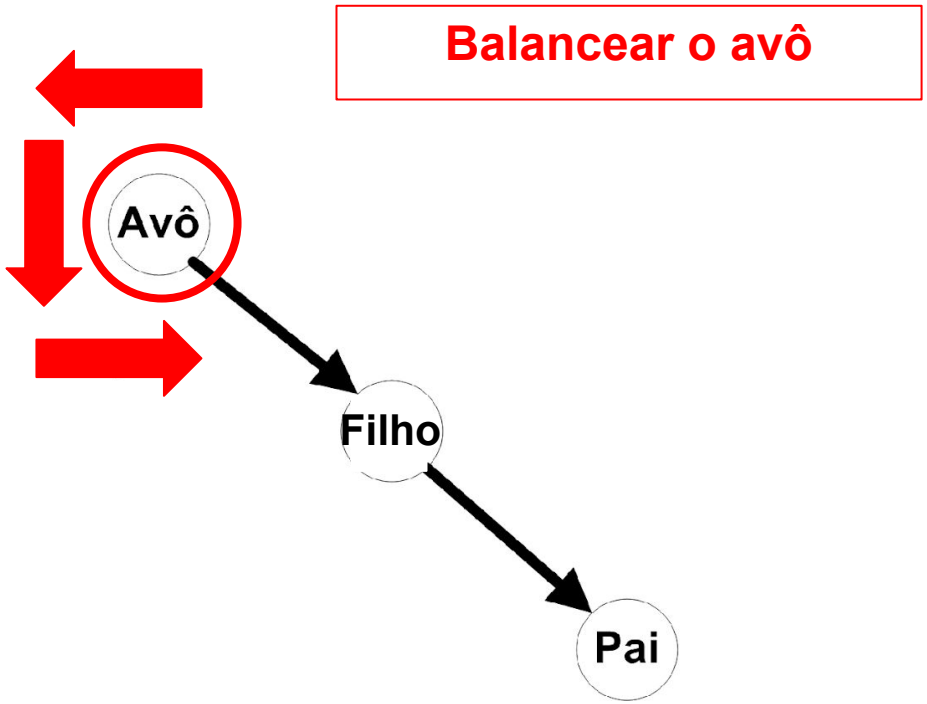
Rotação Dupla Direita – Esquerda

- Usada em subárvores em que o pai está desbalanceado para a direita e o filho para a esquerda



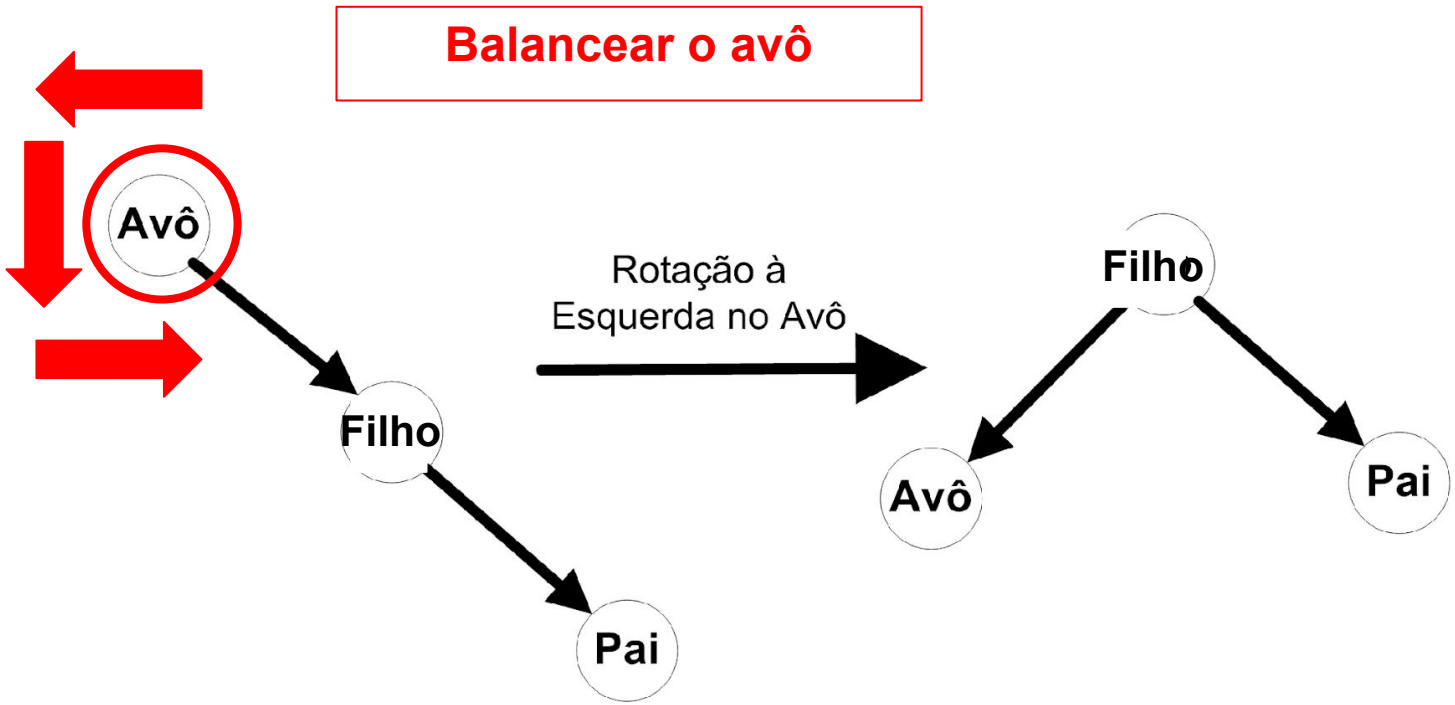
Rotação Dupla Direita – Esquerda

- Usada em subárvores em que o pai está desbalanceado para a direita e o filho para a esquerda



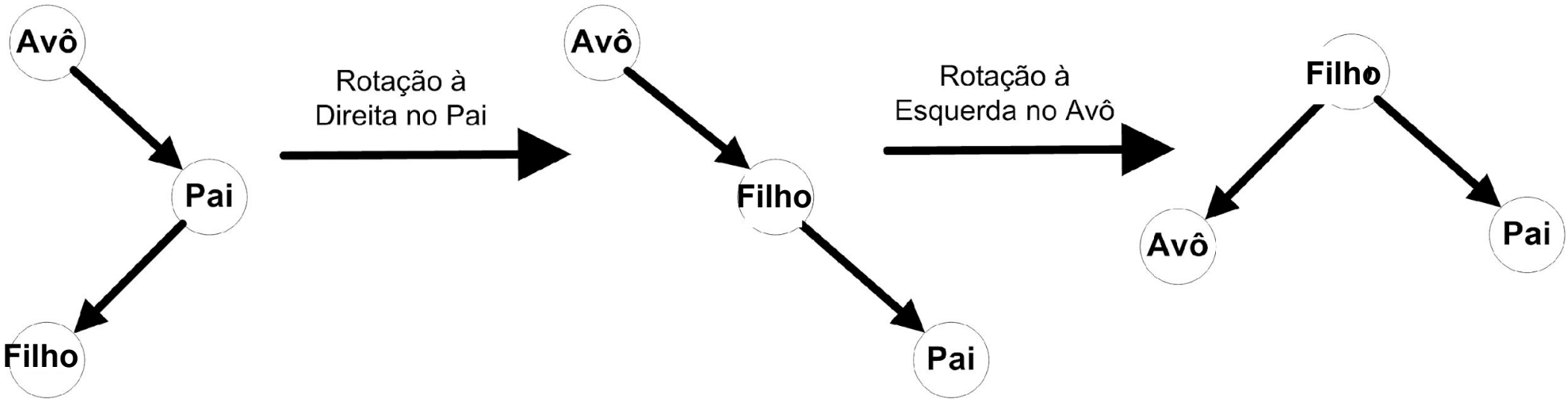
Rotação Dupla Direita – Esquerda

- Usada em subárvores em que o pai está desbalanceado para a direita e o filho para a esquerda



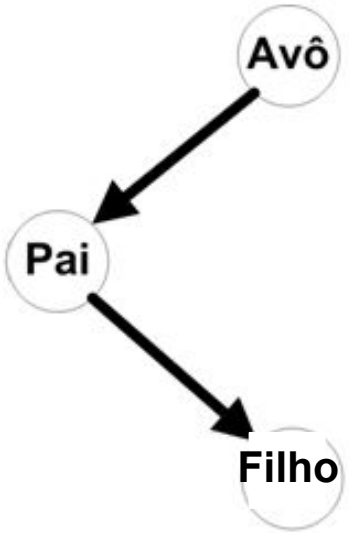
Implementação da Rotação à Direita - Esquerda

```
No rotacionarDirEsq (No no) {
    no.dir = rotacionarDir (no.dir);
    return rotacionarEsq(no);
}
```



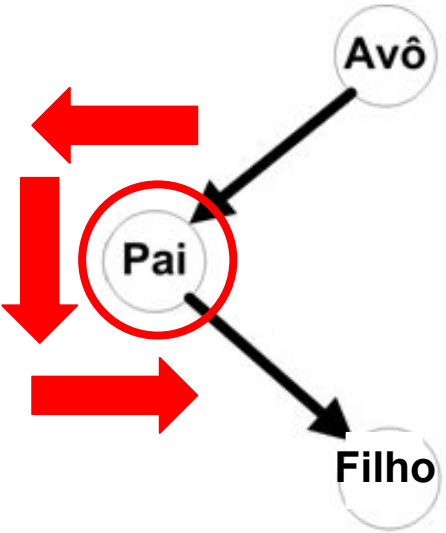
Rotação Dupla Esquerda – Direita

- Usada em subárvores em que o pai está desbalanceado para a esquerda e o filho para a direita



Rotação Dupla Esquerda – Direita

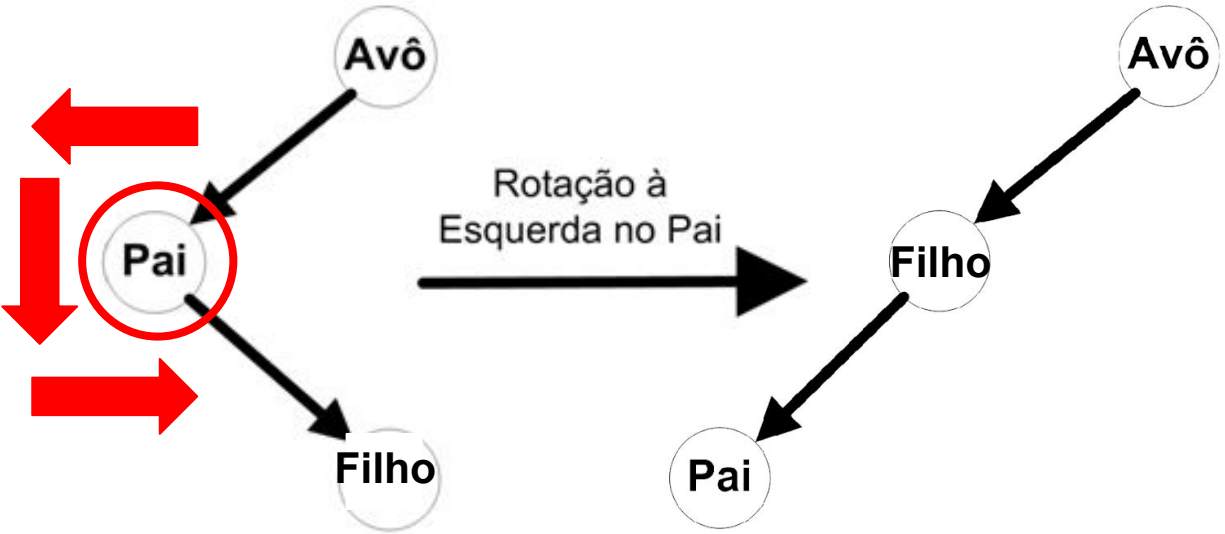
- Usada em subárvores em que o pai está desbalanceado para a esquerda e o filho para a direita



Alinhar o filho

Rotação Dupla Esquerda – Direita

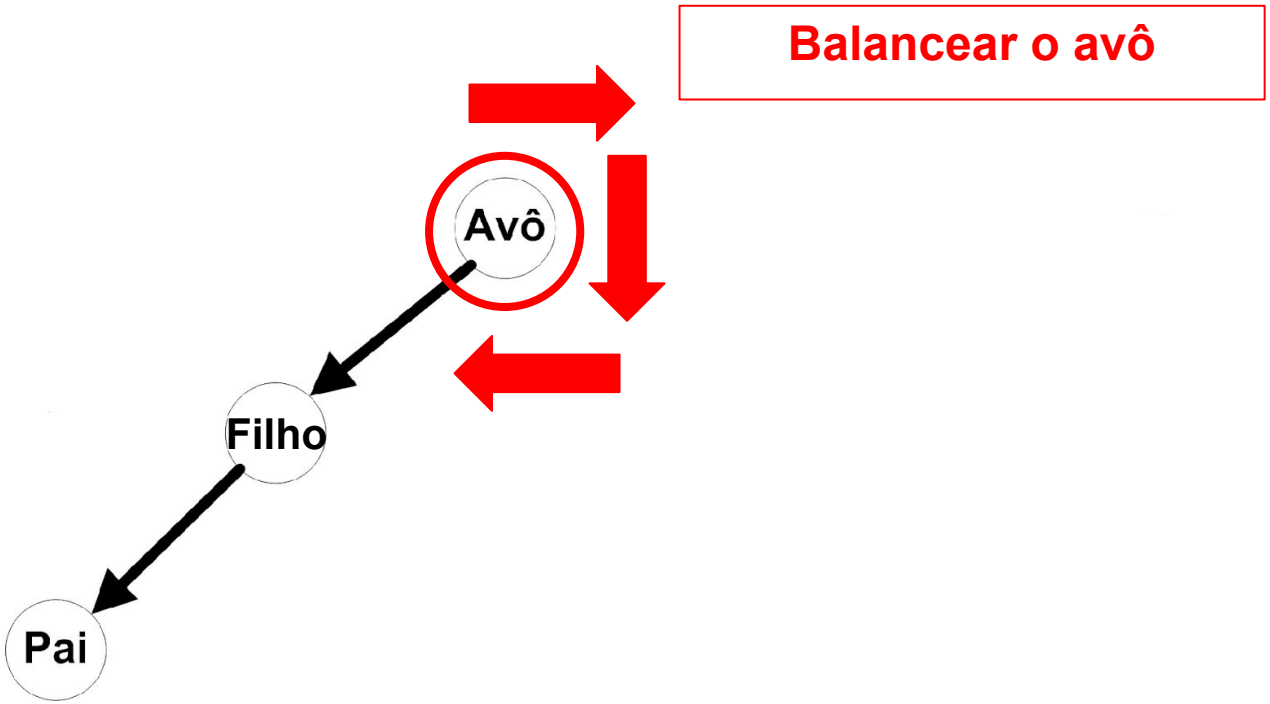
- Usada em subárvores em que o pai está desbalanceado para a esquerda e o filho para a direita



Alinhar o filho

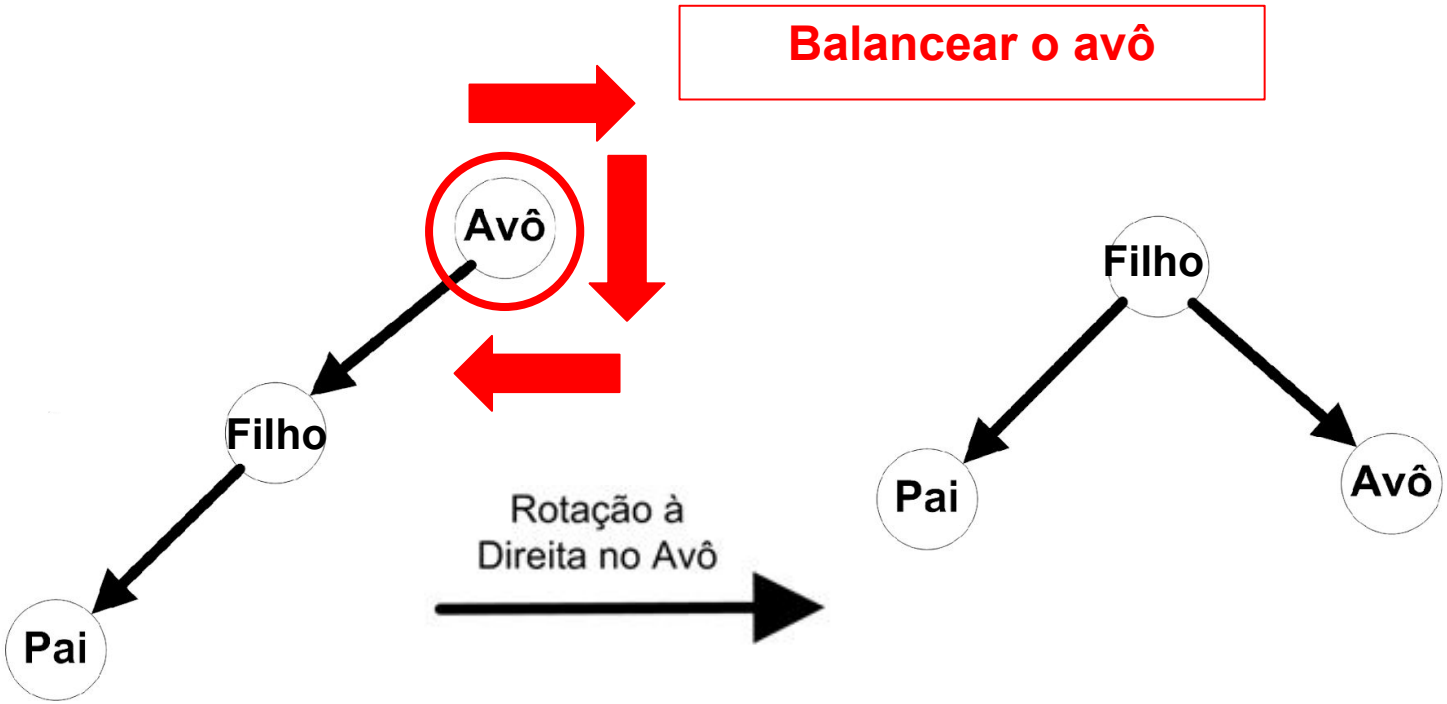
Rotação Dupla Esquerda – Direita

- Usada em subárvores em que o pai está desbalanceado para a esquerda e o filho para a direita



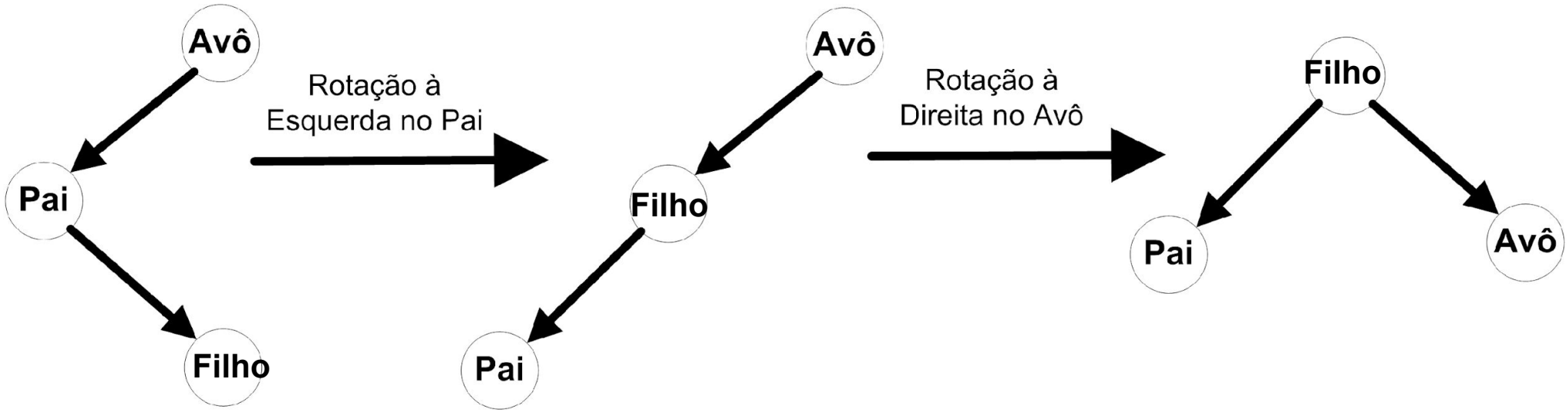
Rotação Dupla Esquerda – Direita

- Usada em subárvores em que o pai está desbalanceado para a esquerda e o filho para a direita



Implementação da Rotação à Esquerda - Direita

```
No rotacionarEsqDir (No no) {
    no.esq = rotacionarEsq (no.esq);
    return rotacionarDir(no);
}
```



Balanceamento de Árvores

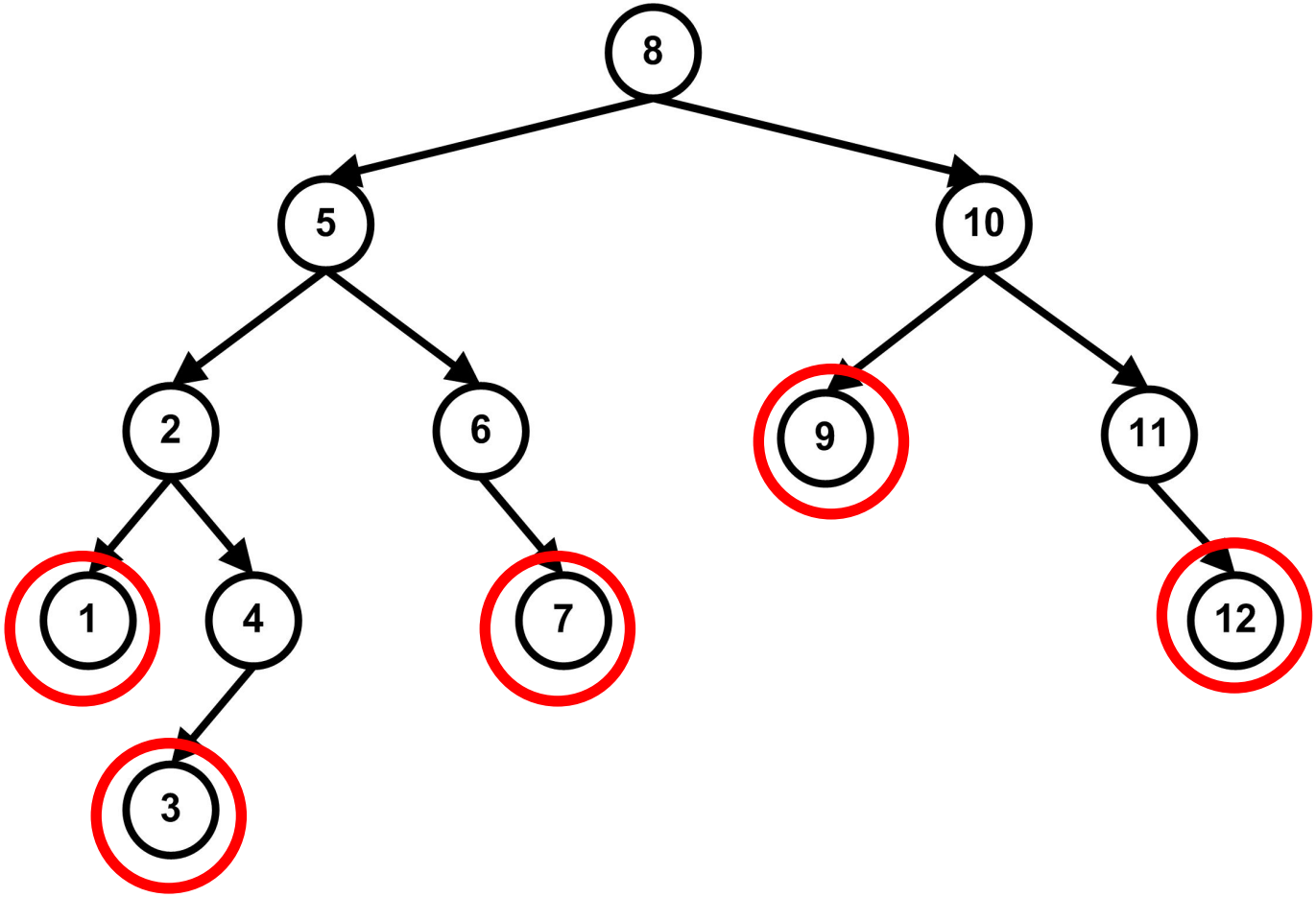
- Qual é o custo para se manter uma árvore balanceada?

- Na prática, não existe “muita” diferença entre árvores balanceadas ou praticamente balanceadas

- Algumas árvores balanceadas como a AVL e a Alvinegra permitem árvores praticamente balanceadas

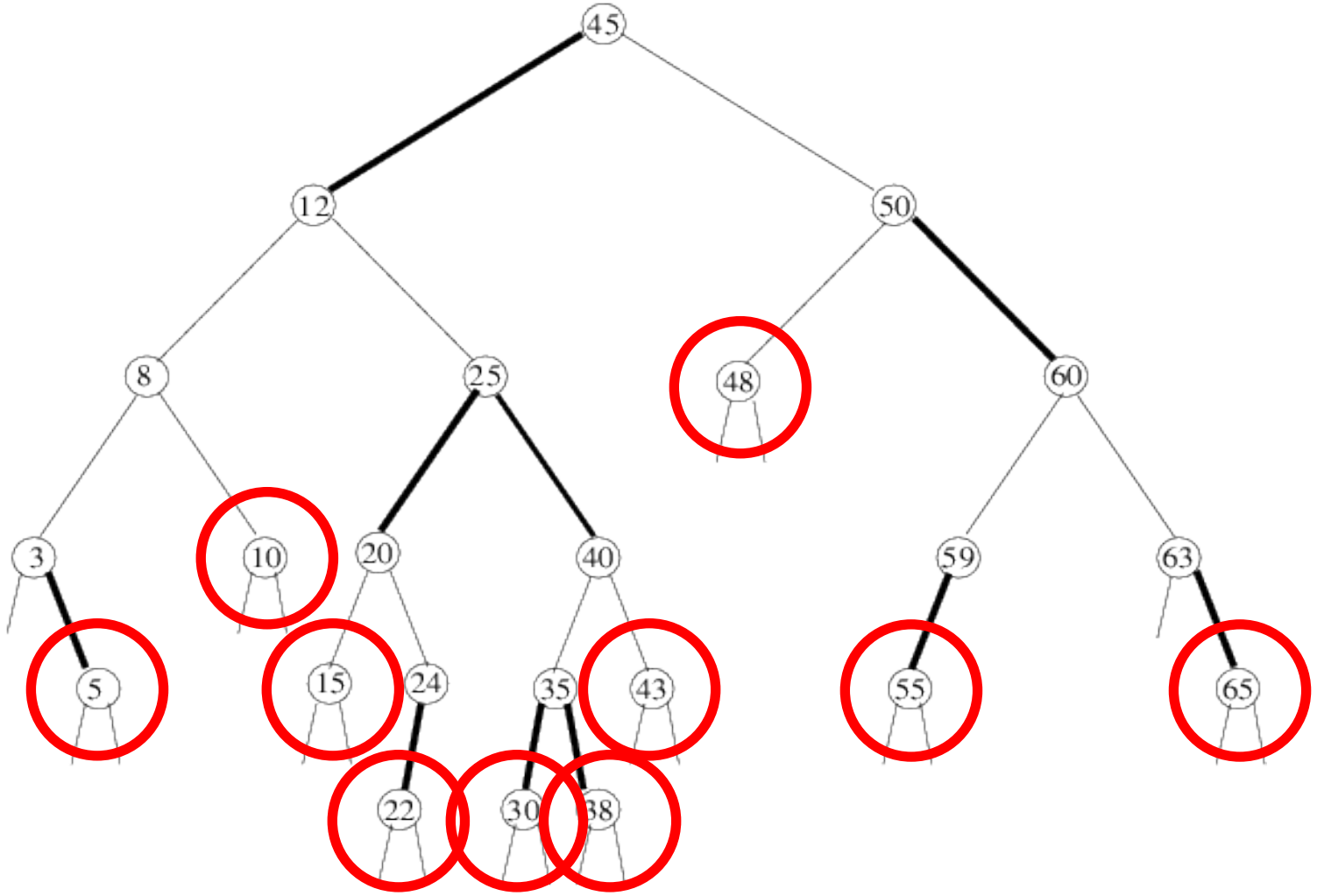
Balanceamento de Árvores

- Exemplo de árvore AVL em que as folhas ocupam mais de dois níveis



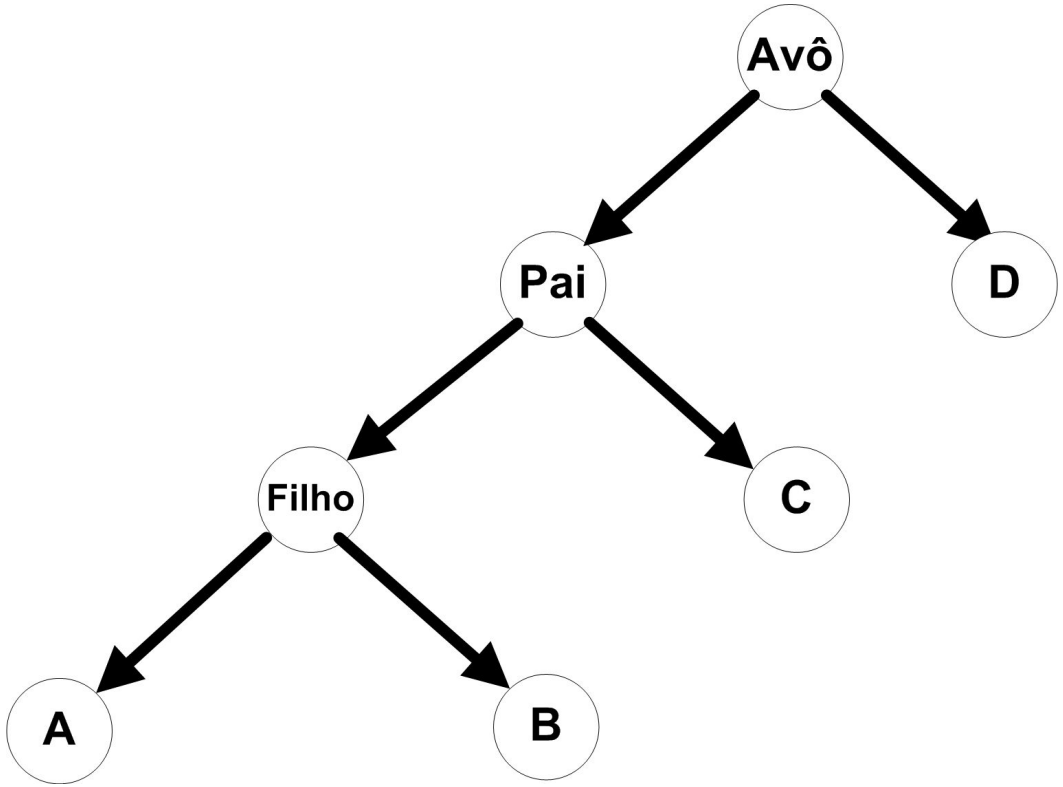
Balanceamento de Árvores

- Exemplo de árvore Alvinegra em que as folhas ocupam mais de dois níveis



Exercício Resolvido 1

- Leia e insira os números em uma árvore binária não balanceada (unidade anterior) para que ela fique como a figura abaixo. Em seguida, execute o caminhar pré. Finalmente, efetue uma rotação à **direita** no nó avô para balancear nossa árvore e execute o caminhar pré novamente.

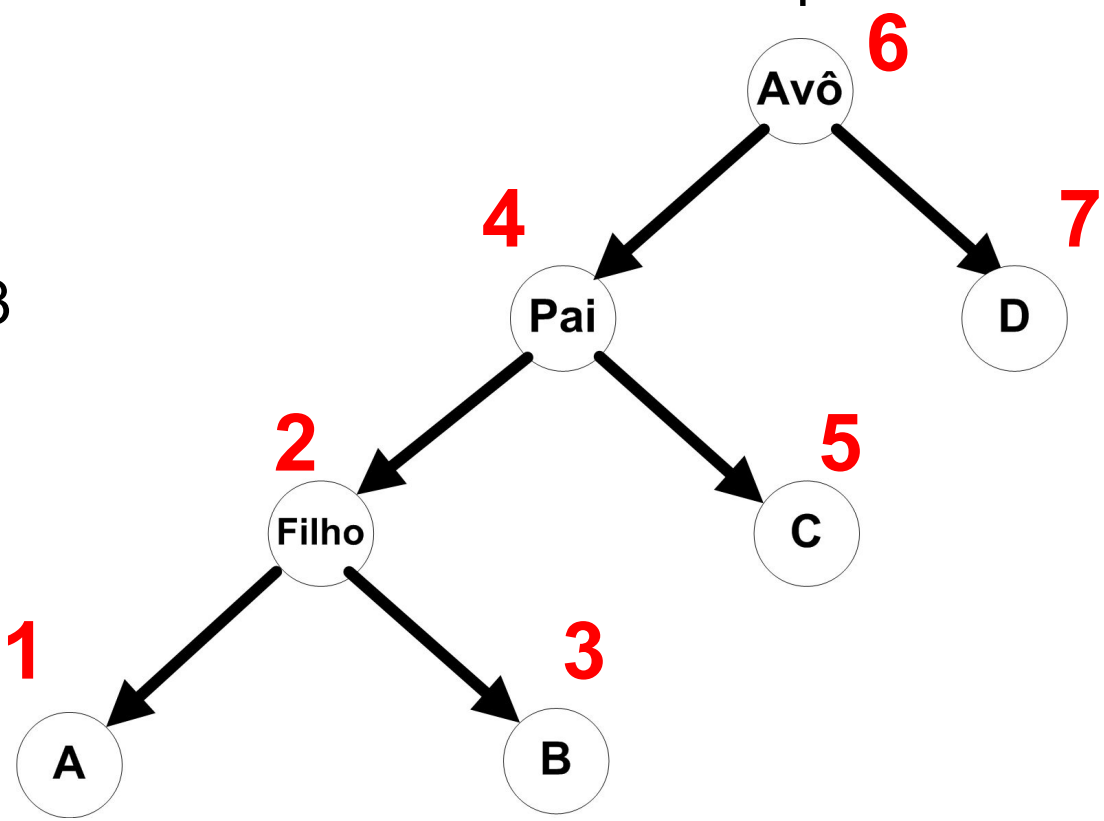


Exercício Resolvido 1

- Leia e insira os números em uma árvore binária não balanceada (unidade anterior) para que ela fique como a figura abaixo. Em seguida, execute o caminhar pré. Finalmente, efetue uma rotação à **direita** no nó avô para balancear nossa árvore e execute o caminhar pré novamente.

Sequência:

6, 4, 7, 2, 5, 1 e 3

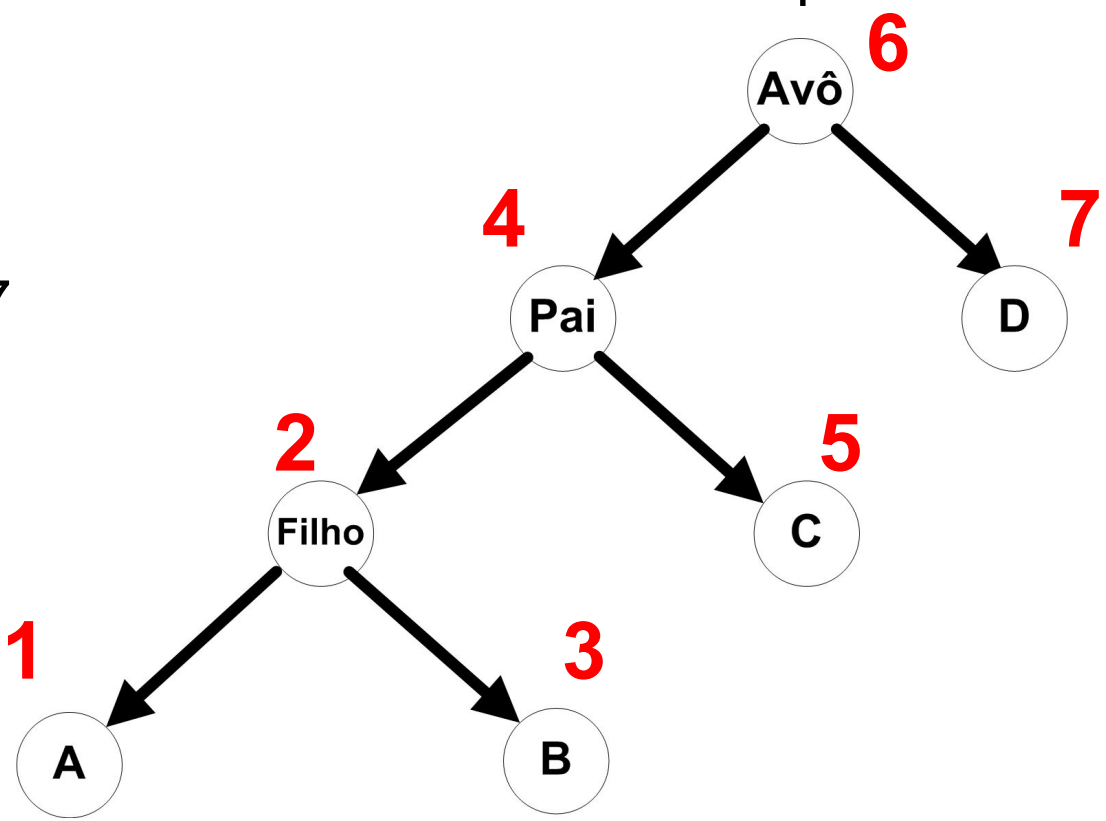


Exercício Resolvido 1

- Leia e insira os números em uma árvore binária não balanceada (unidade anterior) para que ela fique como a figura abaixo. Em seguida, execute o caminhar pré. Finalmente, efetue uma rotação à **direita** no nó avô para balancear nossa árvore e execute o caminhar pré novamente.

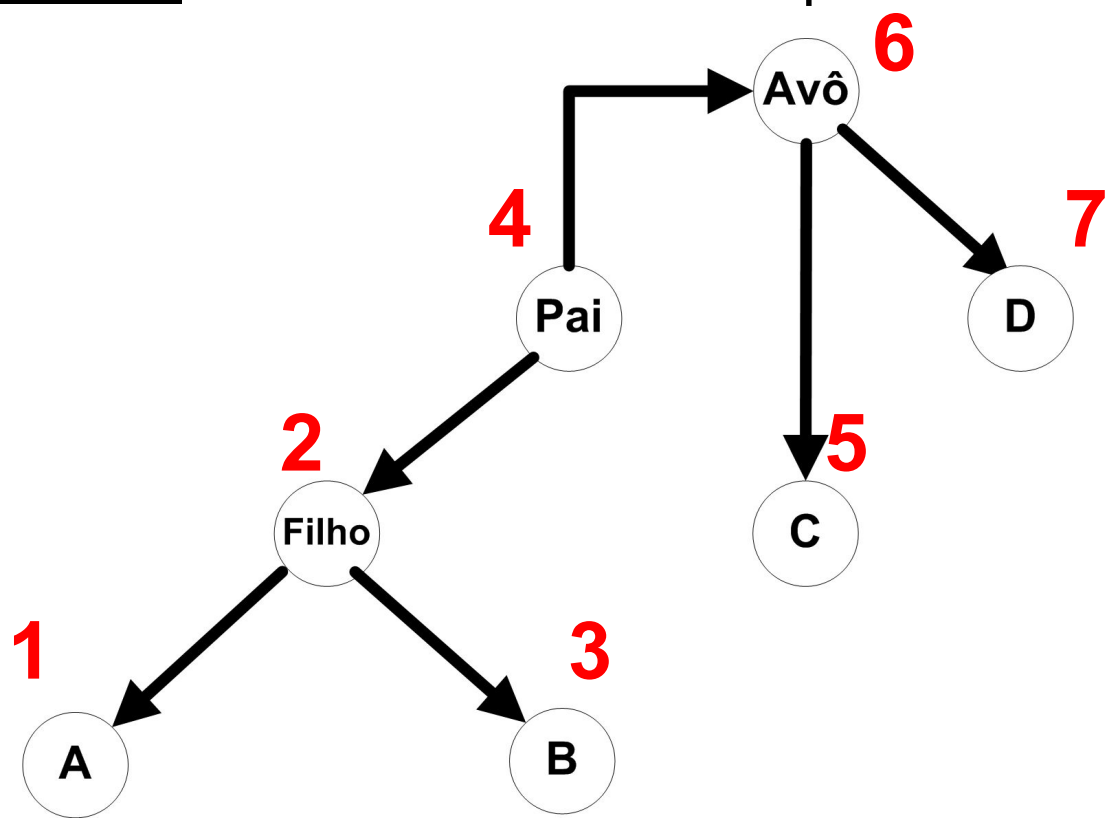
Caminhar pré:

6, 4, 2, 1, 3, 5 e 7



Exercício Resolvido 1

- Leia e insira os números em uma árvore binária não balanceada (unidade anterior) para que ela fique como a figura abaixo. Em seguida, execute o caminhar pré. Finalmente, efetue uma rotação à **direita** no nó avô para balancear nossa árvore e execute o caminhar pré novamente.

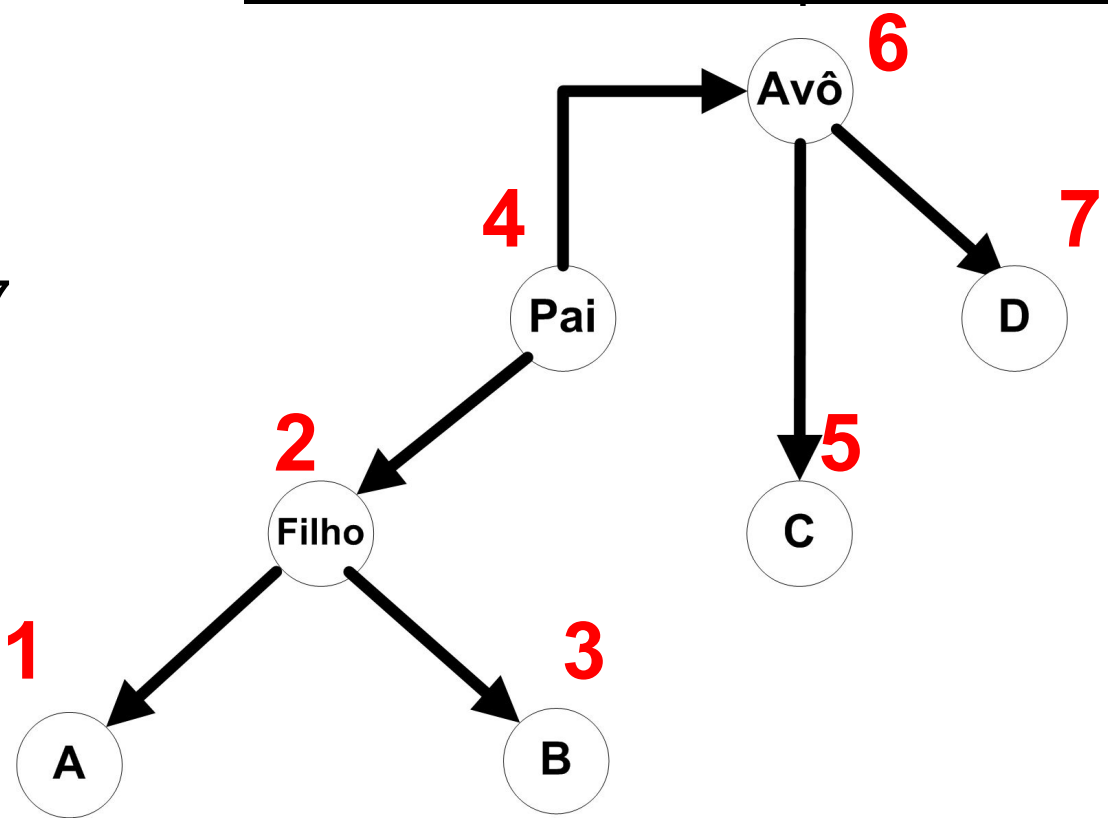


Exercício Resolvido 1

- Leia e insira os números em uma árvore binária não balanceada (unidade anterior) para que ela fique como a figura abaixo. Em seguida, execute o caminhar pré. Finalmente, efetue uma rotação à **direita** no nó avô para balancear nossa árvore e execute o caminhar pré novamente.

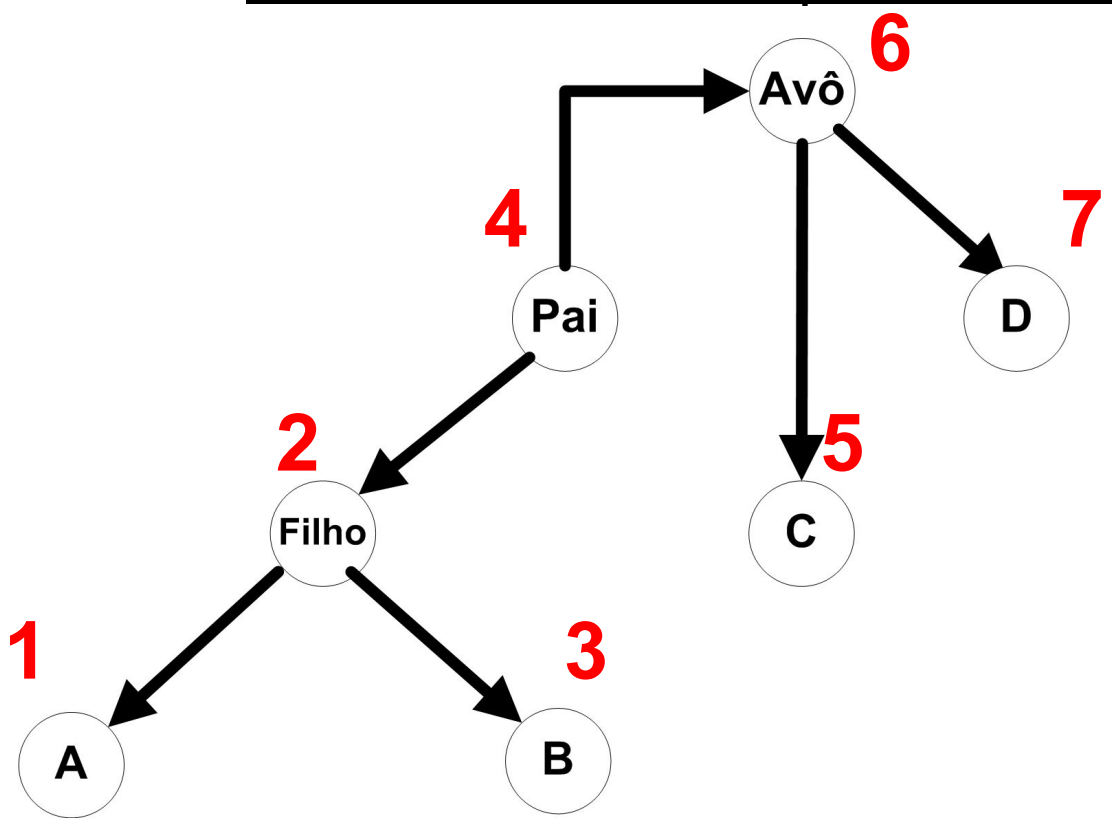
Caminhar pré:

4, 2, 1, 3, 6, 5 e 7



Exercício Resolvido 1

- Leia e insira os números em uma árvore binária não balanceada (unidade anterior) para que ela fique como a figura abaixo. Em seguida, execute o caminhar pré. Finalmente, efetue uma rotação à **direita** no nó avô para balancear nossa árvore e execute o caminhar pré novamente.



Exercício 1

```
void exer3(){
   int a, b, c;
   a = lerInt();
   b = lerInt();
   c = lerInt();

   inserir(a);
   inserir(b);
   inserir(c);



}
```

ros em uma árvore binária não balanceada (unidade

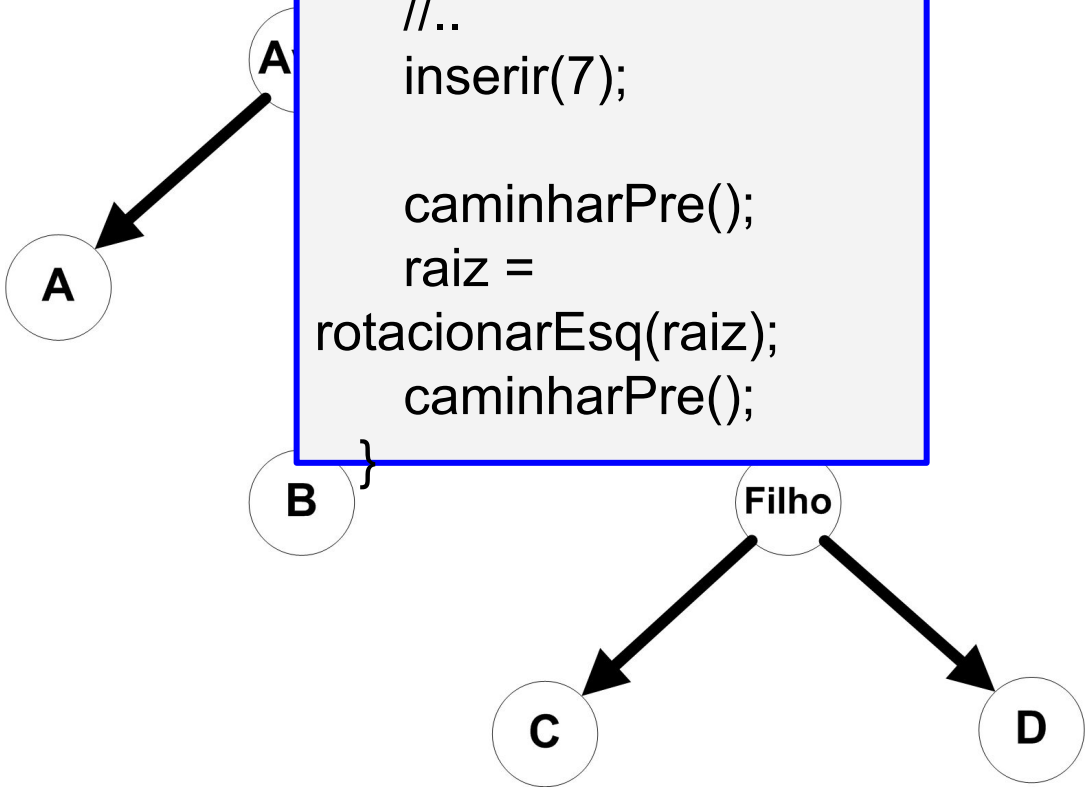
fique como a figura abaixo. Em seguida, execute o

nte, efetue uma rotação à **esquerda** no avô para

balancear nossa árvore e exe ovamente.

```
void exerResol2(){
   inserir(2);
   //..
   inserir(7);

   caminharPre();
   raiz =
rotacionarEsq(raiz);
   caminharPre();
}
```



Exercício Resolvido 2

- Leia e insira três números inteiros em uma árvore binária não balanceada (unidade anterior). Em seguida, usando os quatro tipos de rotação aprendidas nesta unidade, garanta que a árvore tenha apenas dois níveis.

Exercício Resolvido 2

- Leia e insira três números inteiros em uma árvore binária não balanceada (unidade anterior). Em seguida, usando os quatro tipos de rotação aprendidas nesta unidade, garanta que a árvore tenha apenas dois níveis.

Para responder esta pergunta, precisamos saber quantas árvores distintas podemos fazer com três elementos

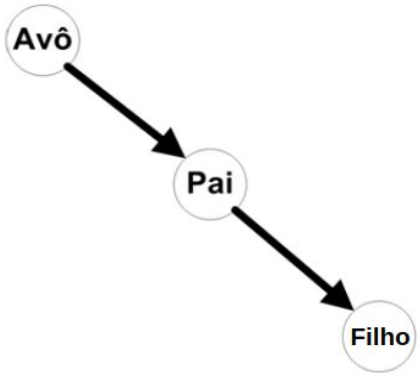
Da matemática, em combinações, temos $\underset{3}{\_\_\_}\ \underset{2}{\_\_\_}\ \underset{1}{\_\_\_}$

1, 2, 3 -- 1, 3, 2 --
2, 1, 3 -- 2, 3, 1 --
3, 1, 2 -- 3, 2, 1

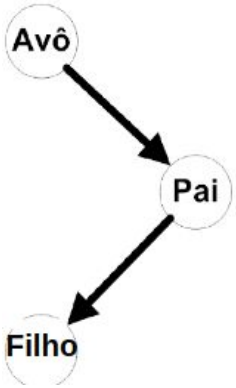
Exercício Resolvido 2

- Leia e insira três números inteiros em uma árvore binária não balanceada (unidade anterior). Em seguida, usando os quatro tipos de rotação aprendidas nesta unidade, garanta que a árvore tenha apenas dois níveis.

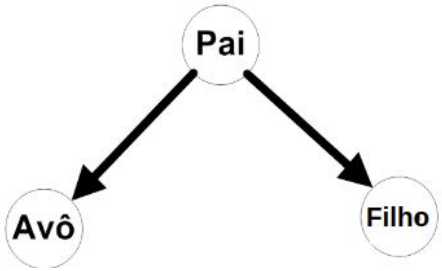
Árvores possíveis



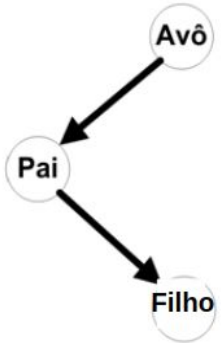
1, 2, 3



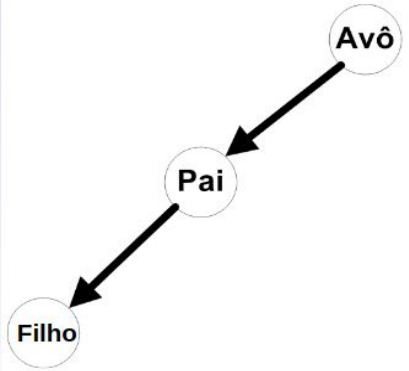
1, 3, 2



2, __, __



3, 1, 2



3, 2, 1

Exercício Resolvido 2

- Leia e insira três números inteiros em uma árvore binária não balanceada (unidade
aprendi
íveis.

```
void balancear (){
    if(raiz.esq != null && raiz.dir != null){ //casos [2,1,3] e [2,3,1]
            /*****************************/
    } else if (raiz.dir != null){
      if (raiz.dir.dir != null){ //caso [1,2,3]
          /*****************************/
      } else { // caso [1,3,2]
          /*****************************/
      }
    } else { // if(raiz.esq != null)
      if (raiz.esq.dir != null){ //caso [3, 1, 2]
          /*****************************/
      } else { //caso [3,2,1]
          /*****************************/
      }
    }
  }
```

Exercício Resolvido 2

- Leia e insira três números inteiros em uma árvore binária não balanceada (unidade aprendi íveis.

```
void balancear (){
    if(raiz.esq != null && raiz.dir != null){ //casos [2,1,3] e [2,3,1]
            System.out.println("Árvore balanceada");
    } else if (raiz.dir != null){
      if (raiz.dir.dir != null){ //caso [1,2,3]
            raiz = rotacionarEsq(raiz);
      } else { // caso [1,3,2]
            raiz = rotacionarDirEsq(raiz);
      }
    } else { // if(raiz.esq != null)
      if (raiz.esq.dir != null){ //caso [3, 1, 2]
            raiz = rotacionarEsqDir(raiz);
      } else { //caso [3,2,1]
            raiz = rotacionarDir(raiz);
      }
    }
  }
```