Estruturas de Dados II

Árvores AVL - Implementação

Prof[a]. Juliana de Santi
Prof. Rodrigo Minetto
Universidade Tecnológica Federal do Paraná
Material compilado de: Cormen, Notas de aula IC-UNICAMP e IME-USP

Sumário

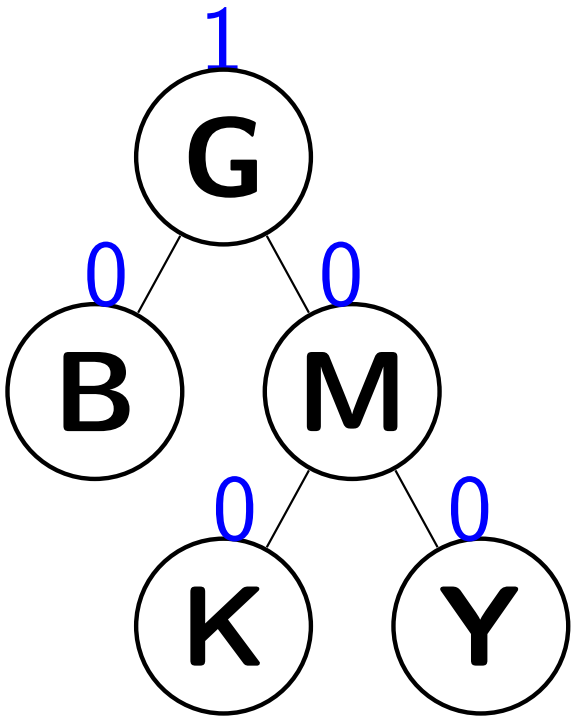
Árvore AVL - TAD

Uma árvore **AVL** de caracteres pode ser definida pela seguinte **estrutura**:

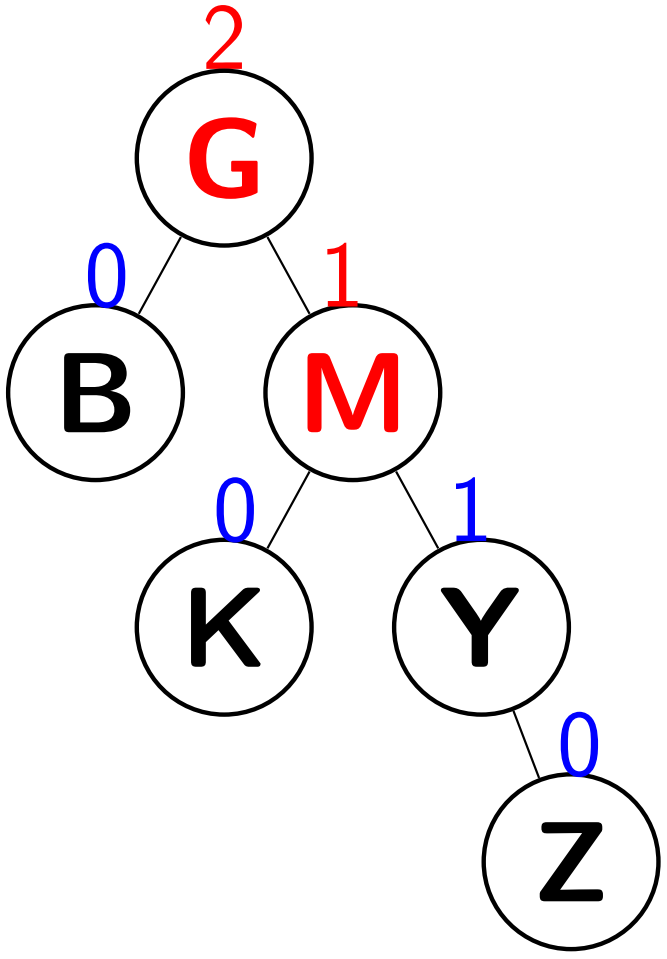
```
typedef struct node {
    char chave;
    int altura;
    struct node* esq;
    struct node* dir;
} No, Arvore;
```

Sumário

Inserção do nó Z

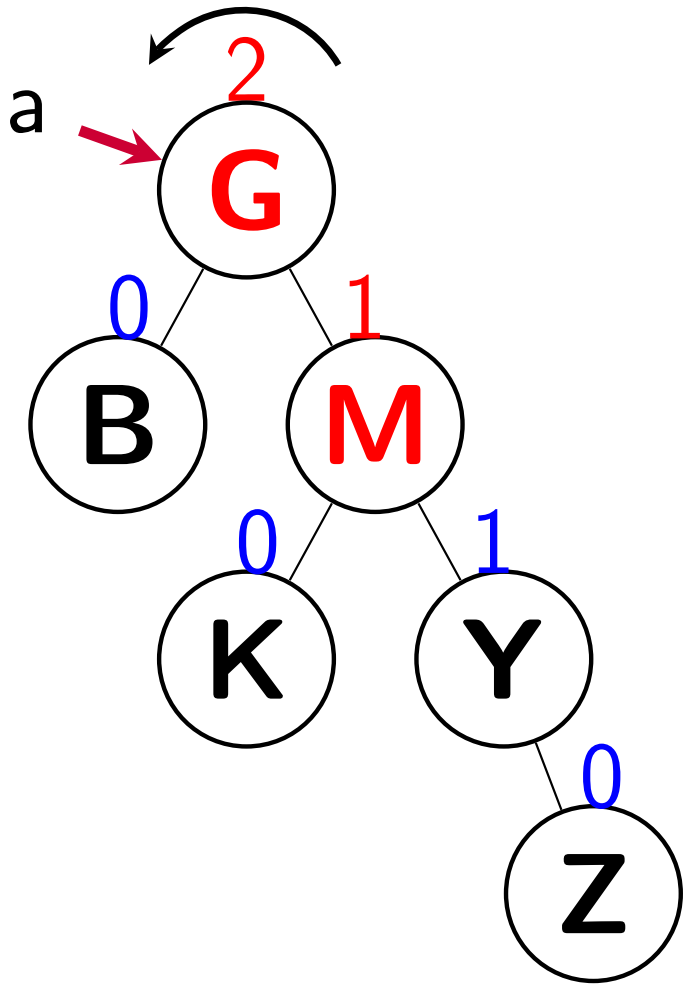


Inserção do nó Z



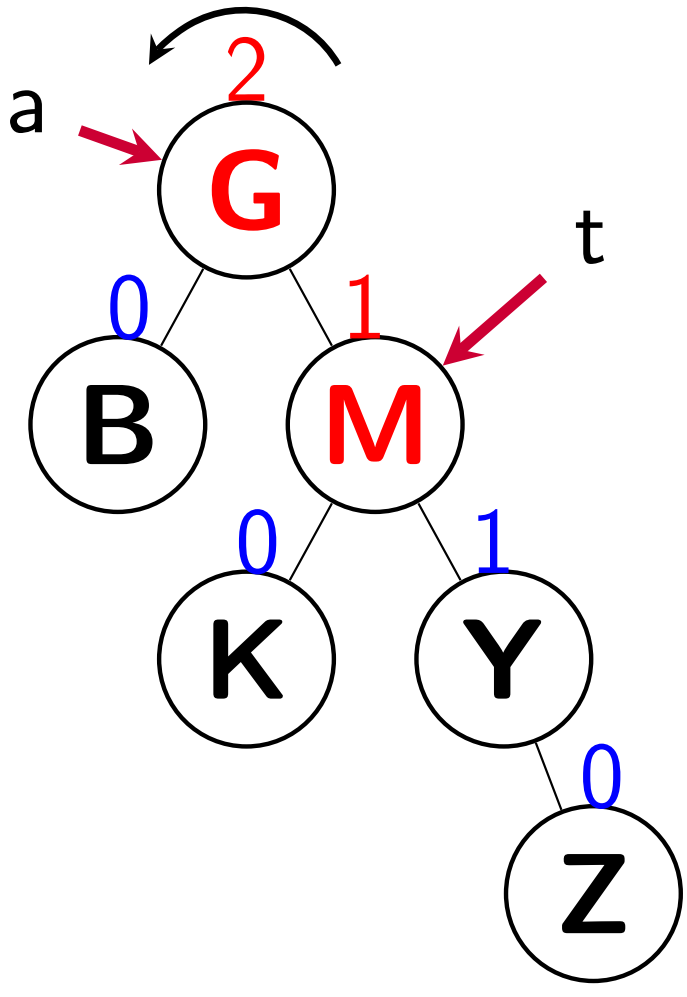
Árvore AVL - Rotação simples à esquerda

```
Arvore* Rotacao_simples_esq (Arvore *a) {
    No *t = a→dir;
    a→dir = t→esq;
    t→esq = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



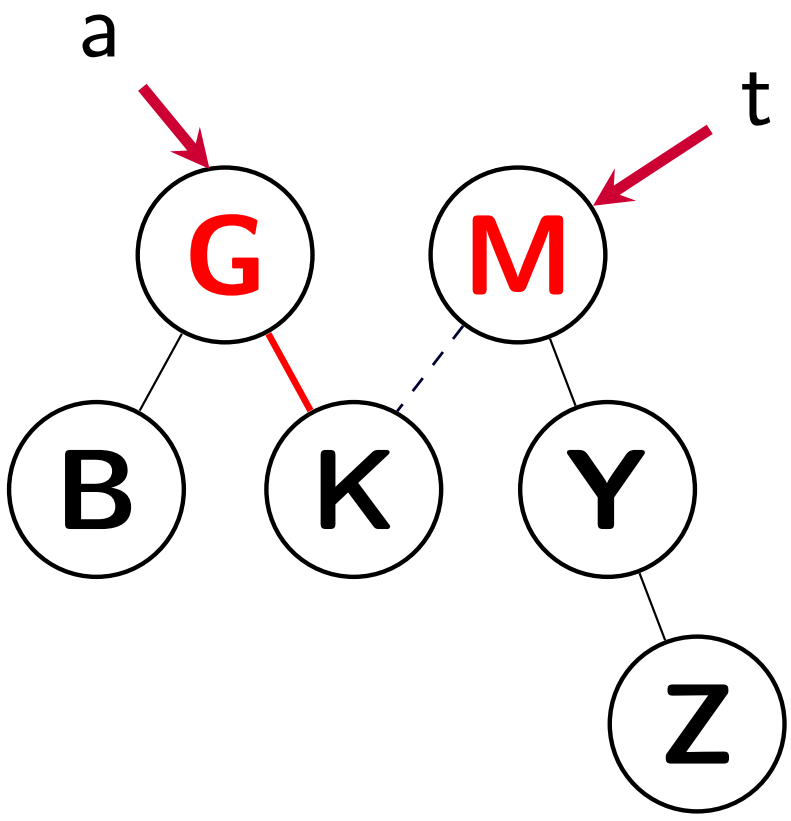
Árvore AVL - Rotação simples à esquerda

```
Arvore* Rotacao_simples_esq (Arvore *a) {
    No *t = a→dir;
    a→dir = t→esq;
    t→esq = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



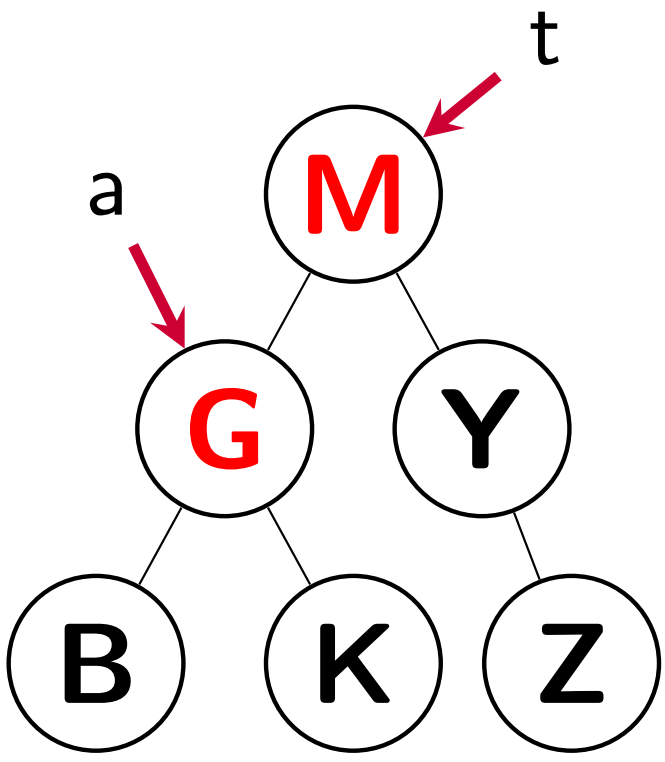
Árvore AVL - Rotação simples à esquerda

```
Arvore* Rotacao_simples_esq (Arvore *a) {
    No *t = a→dir;
    a→dir = t→esq;
    t→esq = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



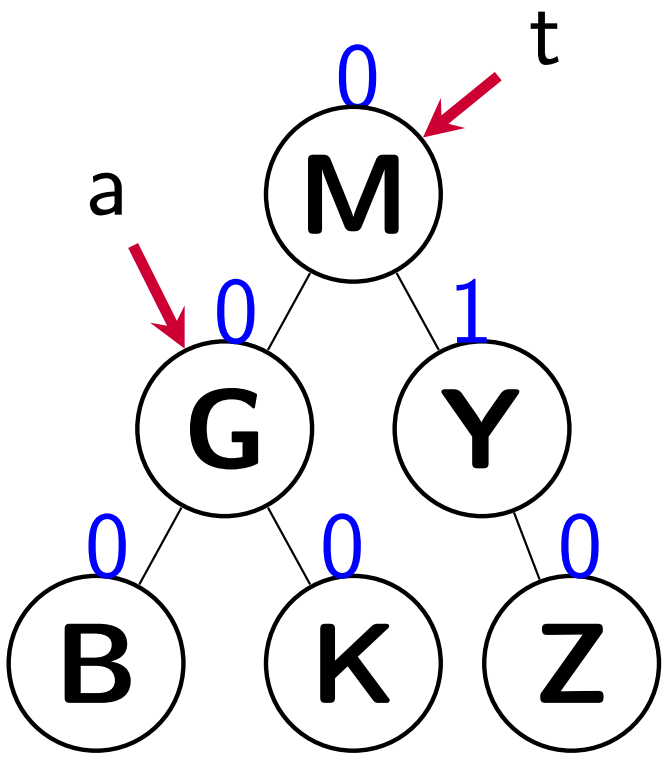
Árvore AVL - Rotação simples à esquerda

```
Arvore* Rotacao_simples_esq (Arvore *a) {
    No *t = a→dir;
    a→dir = t→esq;
    t→esq = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



Árvore AVL - Rotação simples à esquerda

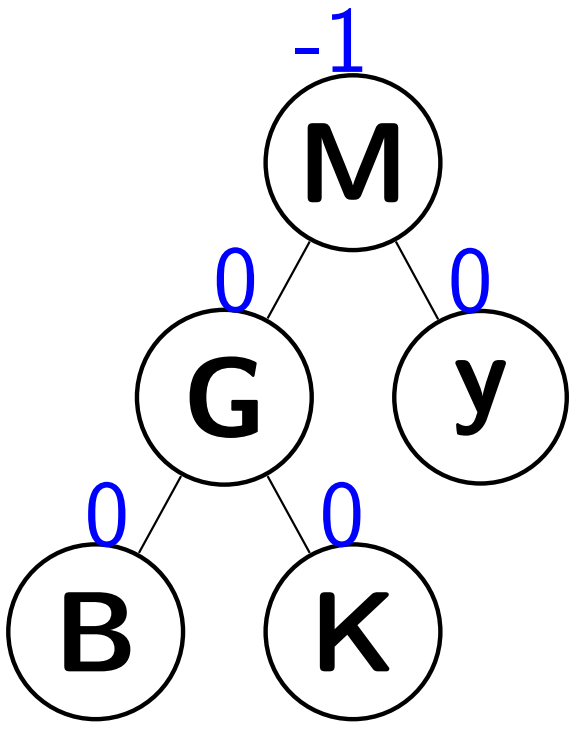
```
Arvore* Rotacao_simples_esq (Arvore *a) {
    No *t = a→dir;
    a→dir = t→esq;
    t→esq = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



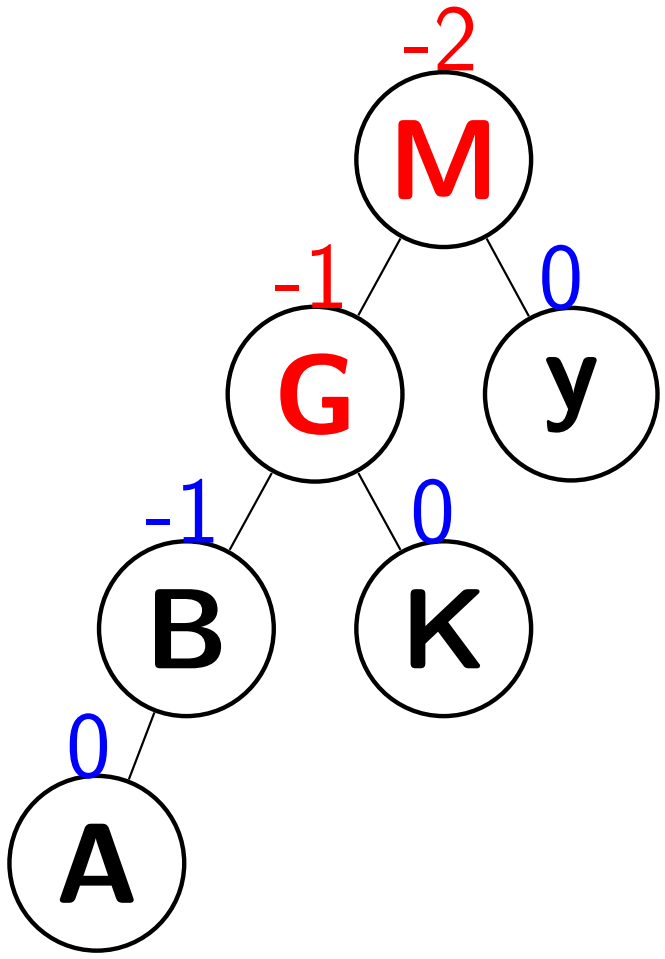

```
int Atualizar (Arvore *a) {
    return (maior(Altura(a→esq), Altura(a→dir)) + 1);
}
```

Sumário

Inserção do nó A

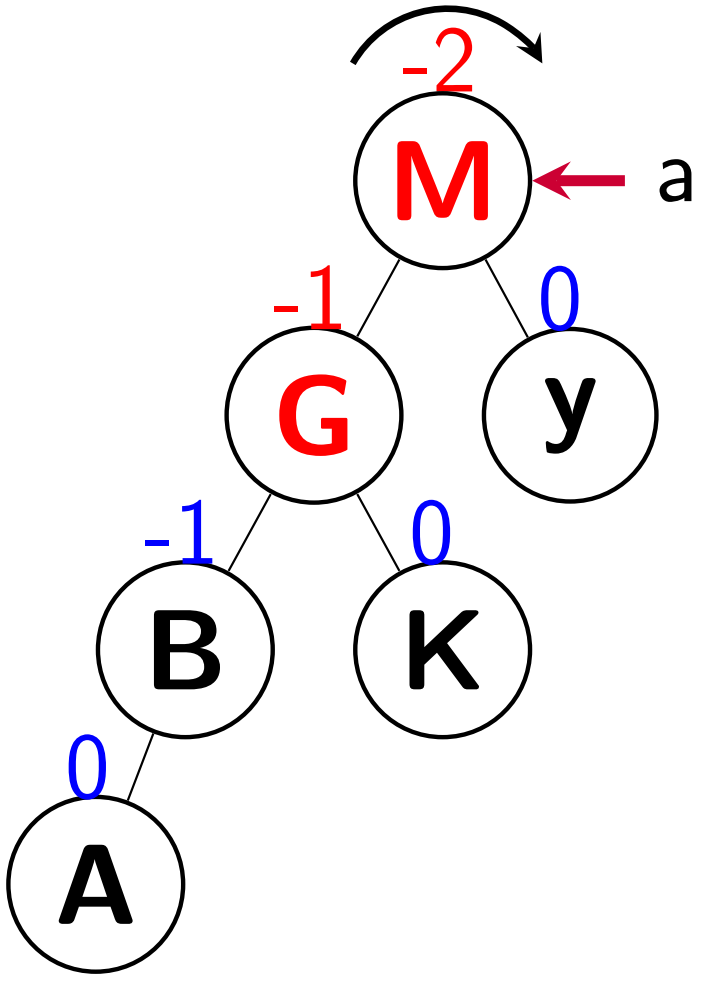


Inserção do nó A



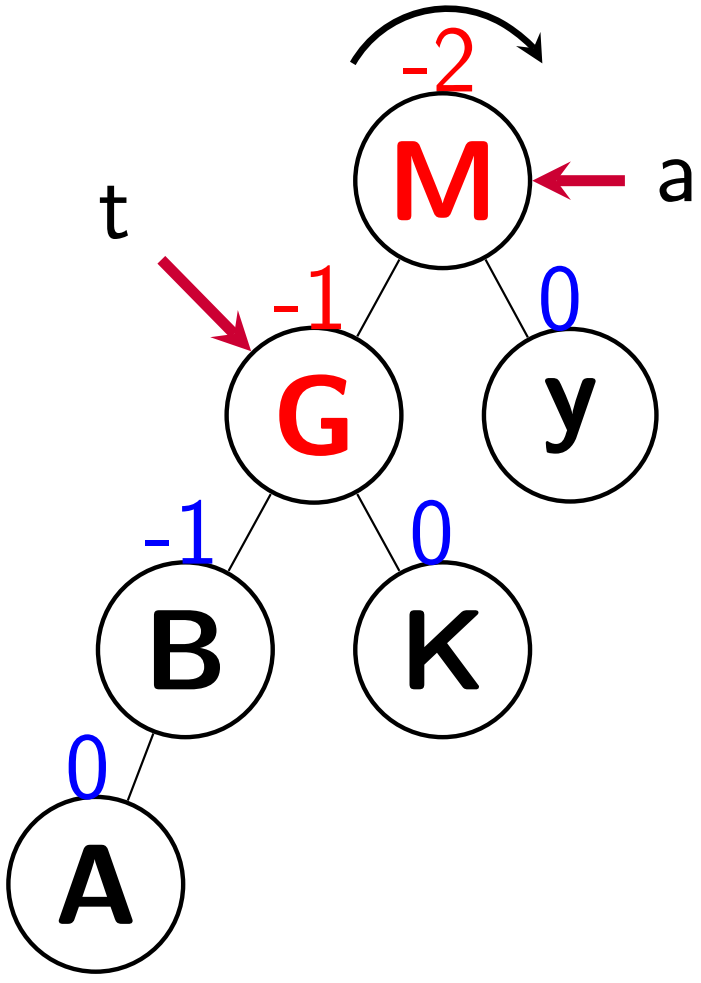
Árvore AVL - Rotação simples à direita

```
Arvore* Rotacao_simples_dir (Arvore *a) {
    No *t = a→esq;
    a→esq = t→dir;
    t→dir = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



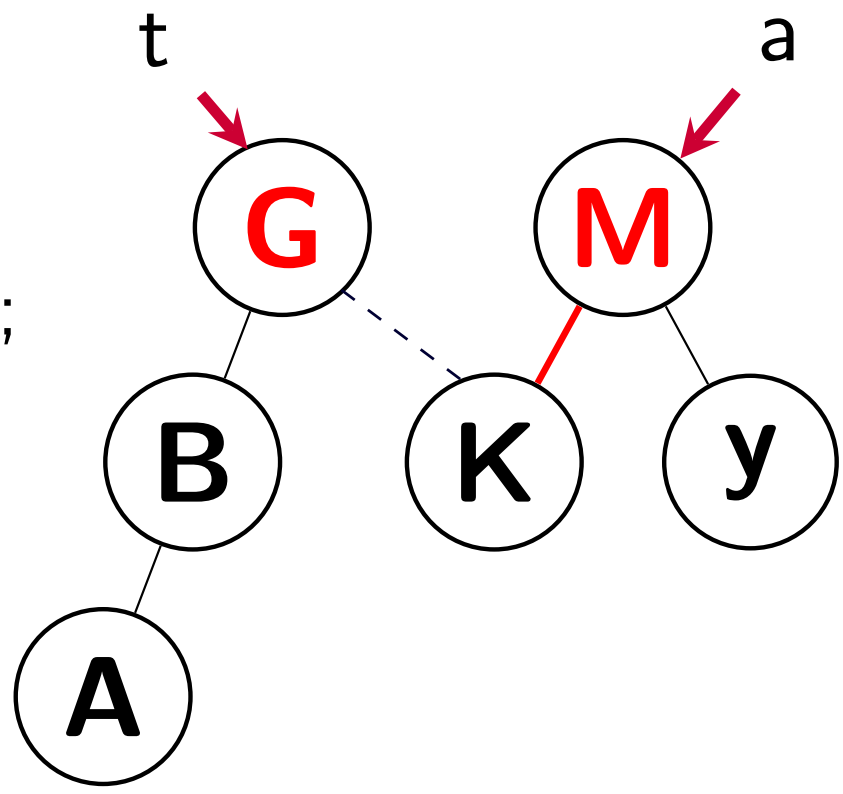
Árvore AVL - Rotação simples à direita

```
Arvore* Rotacao_simples_dir (Arvore *a) {
    No *t = a→esq;
    a→esq = t→dir;
    t→dir = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



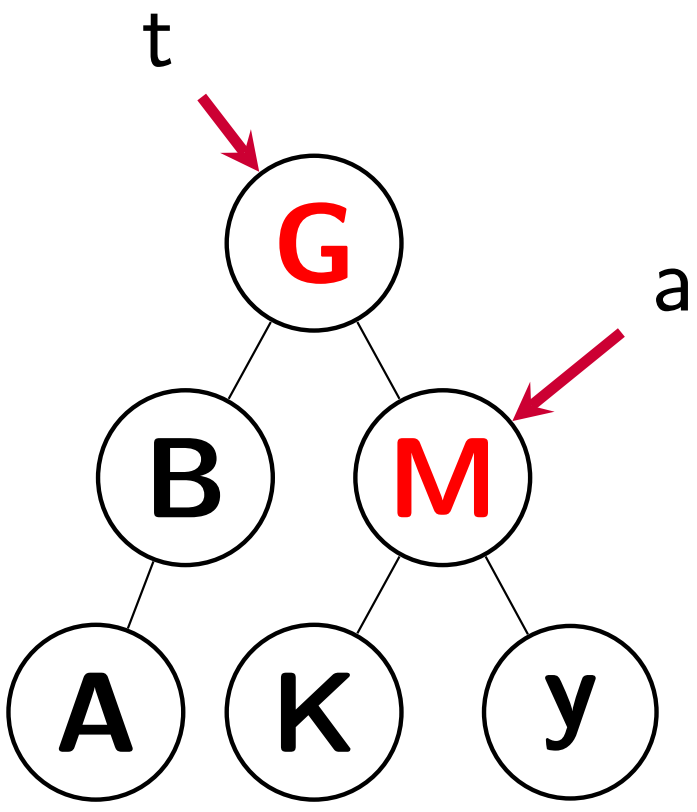
Árvore AVL - Rotação simples à direita

```
Arvore* Rotacao_simples_dir (Arvore *a) {
    No *t = a→esq;
    a→esq = t→dir;
    t→dir = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



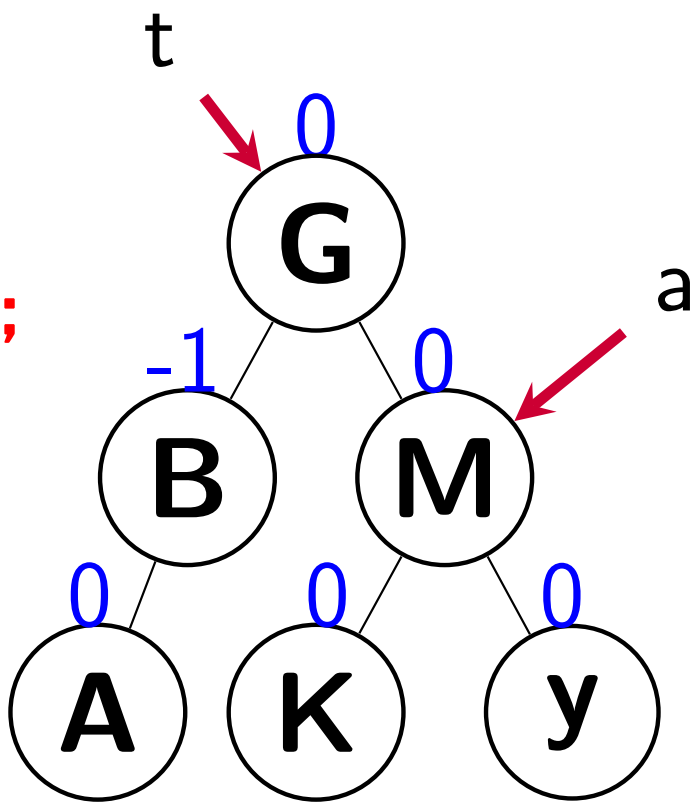
Árvore AVL - Rotação simples à direita

```
Arvore* Rotacao_simples_dir (Arvore *a) {
    No *t = a→esq;
    a→esq = t→dir;
    t→dir = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```



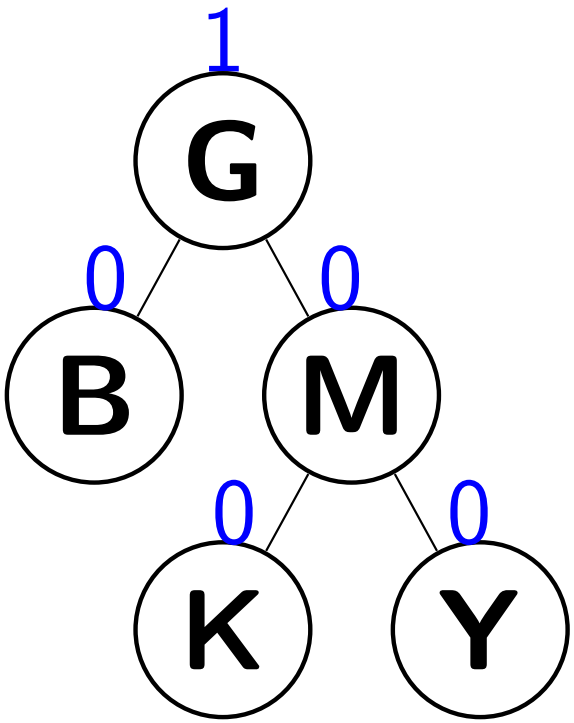
Árvore AVL - Rotação simples à direita

```
Arvore* Rotacao_simples_dir (Arvore *a) {
    No *t = a→esq;
    a→esq = t→dir;
    t→dir = a;
    a→altura = Atualizar(a);
    t→altura = Atualizar(t);
    return t;
}
```

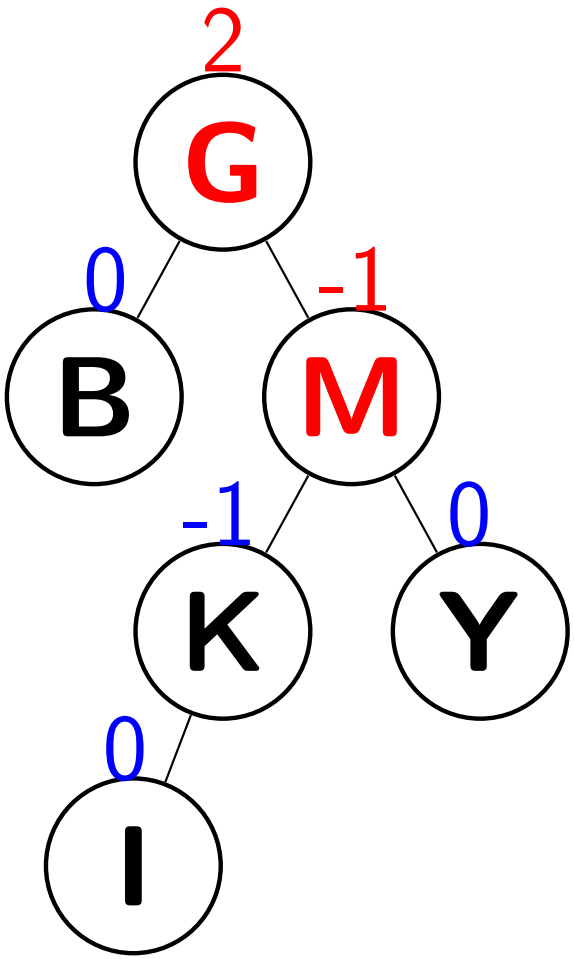


Sumário

Inserção do nó I

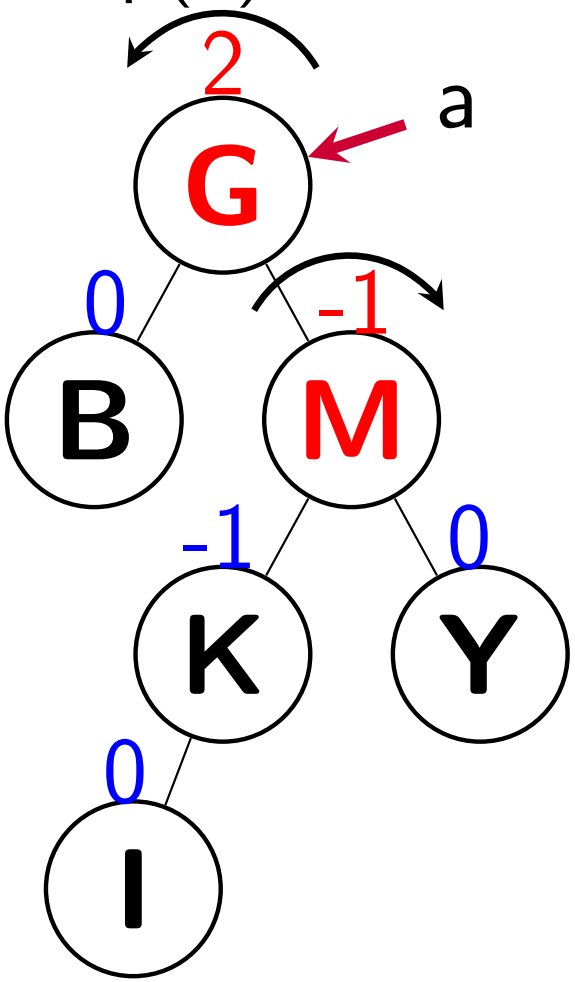


Inserção do nó I



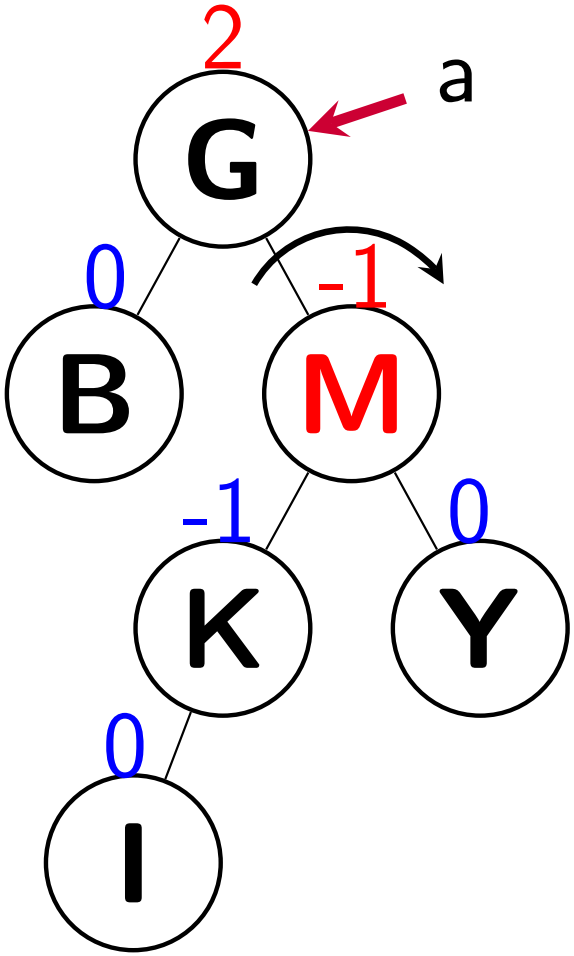
Árvore AVL - Rotação dupla à esquerda

```
Arvore* Rotacao_dupla_esq (Arvore *a) {
    a→dir = Rotacao_simples_dir (a→dir);
    return Rotacao_simples_esq (a);
}
```



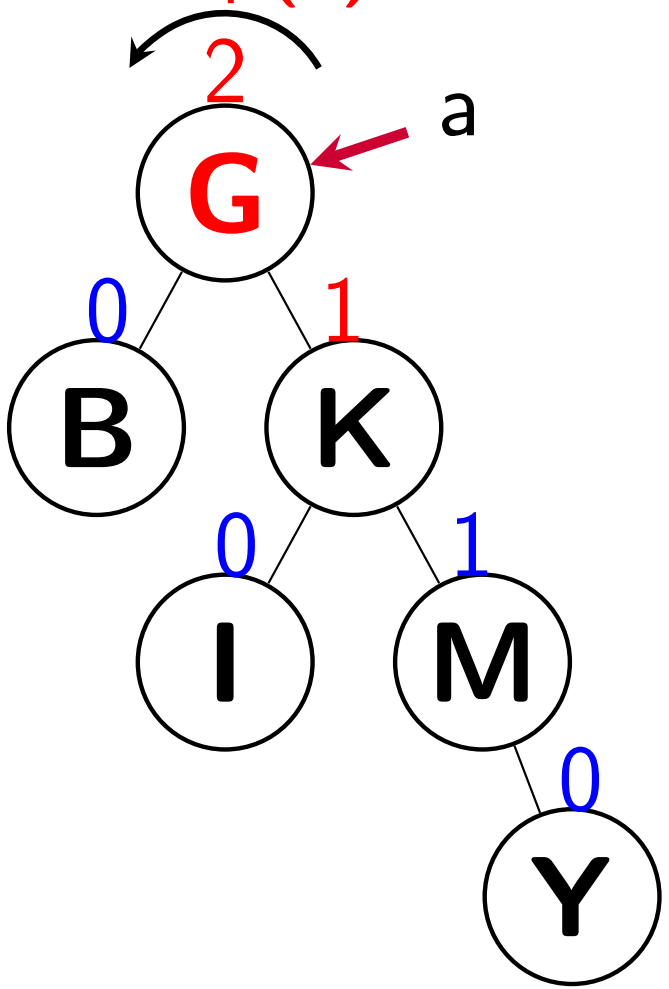
Árvore AVL - Rotação dupla à esquerda

```
Arvore* Rotacao_dupla_esq (Arvore *a) {
    a->dir = Rotacao_simples_dir (a->dir);
    return Rotacao_simples_esq (a);
}
```



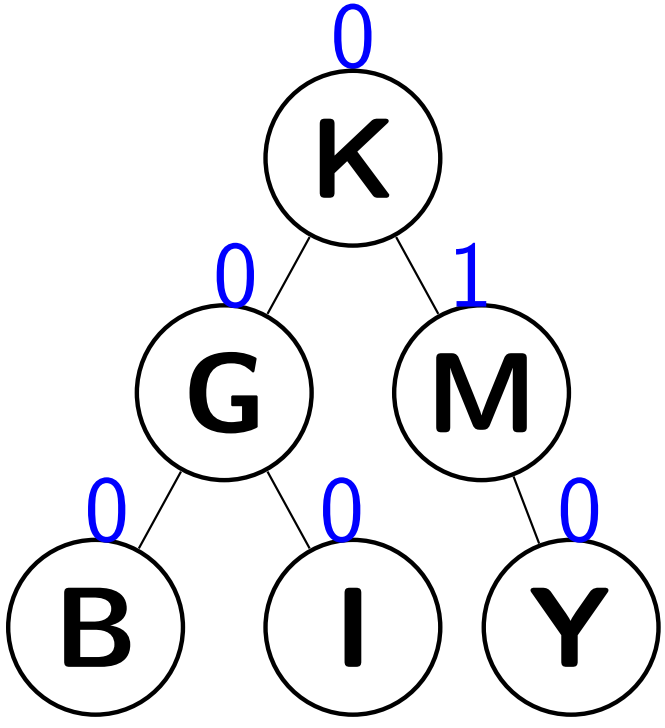
Árvore AVL - Rotação dupla à esquerda

```
Arvore* Rotacao_dupla_esq (Arvore *a) {
    a->dir = Rotacao_simples_dir (a->dir);
    return Rotacao_simples_esq (a);
}
```



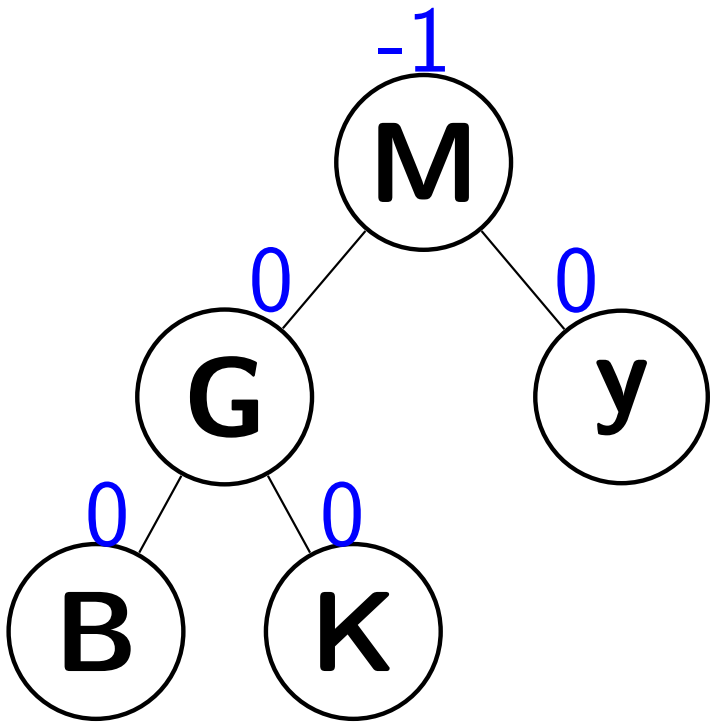
Árvore AVL - Rotação dupla à esquerda

```
Arvore* Rotacao_dupla_esq (Arvore *a) {
    a→dir = Rotacao_simples_dir (a→dir);
    return Rotacao_simples_esq (a);
}
```

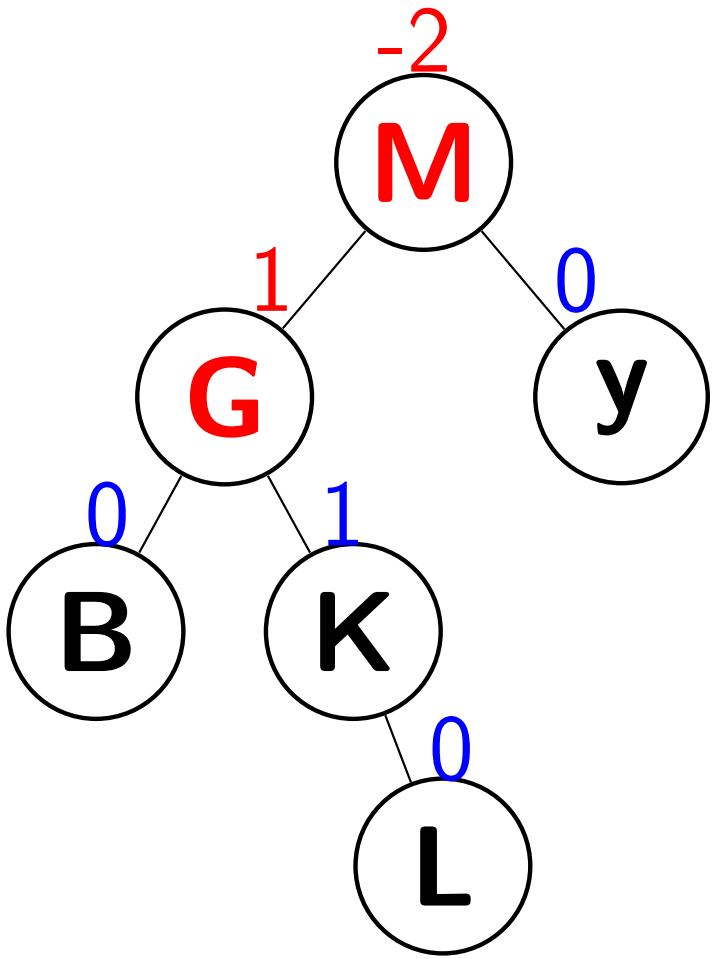


Sumário

Inserção do nó L

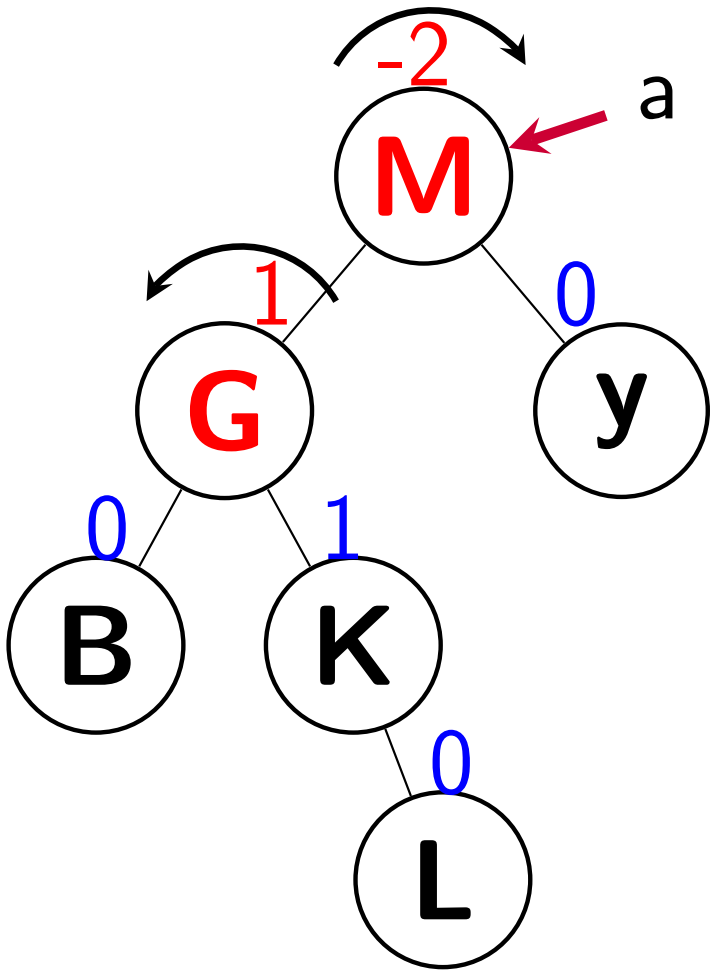


Inserção do nó L



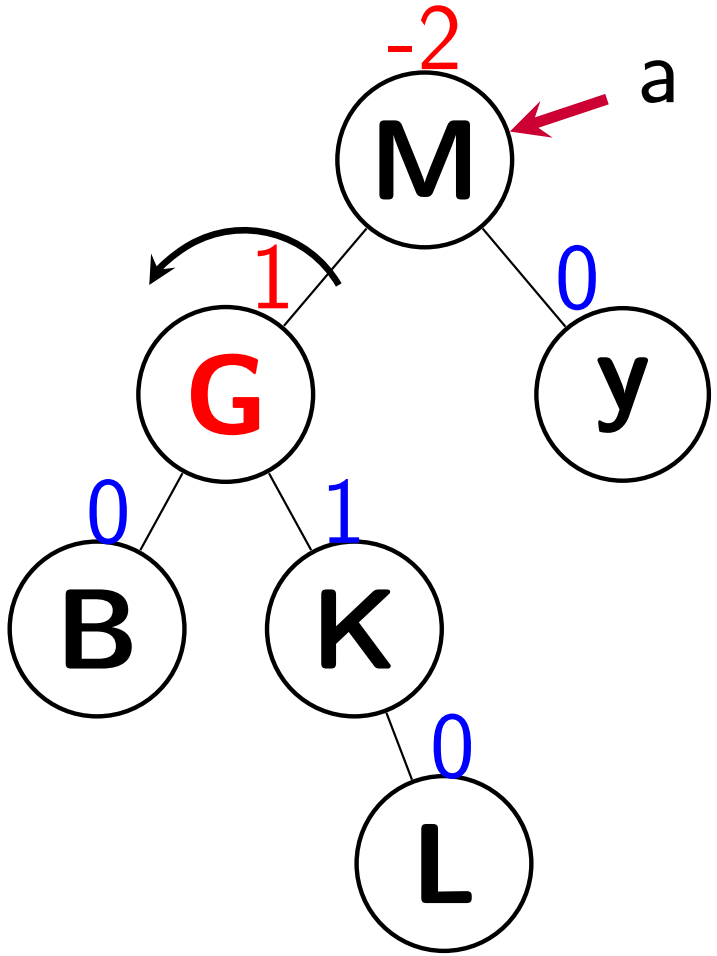
Árvore AVL - Rotação dupla à direita

```
Arvore* Rotacao_dupla_dir (Arvore *a) {
    a→esq = Rotacao_simples_esq (a→esq);
    return Rotacao_simples_dir (a);
}
```



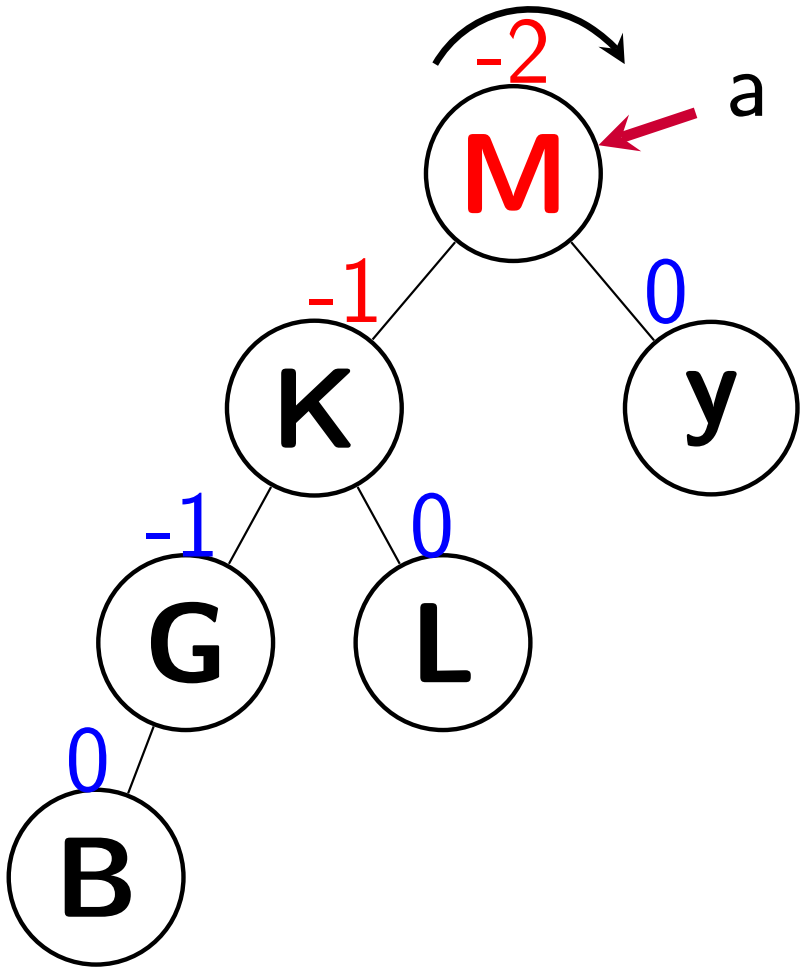
Árvore AVL - Rotação dupla à direita

```
Arvore* Rotacao_dupla_dir (Arvore *a) {
    a->esq = Rotacao_simples_esq (a->esq);
    return Rotacao_simples_dir (a);
}
```



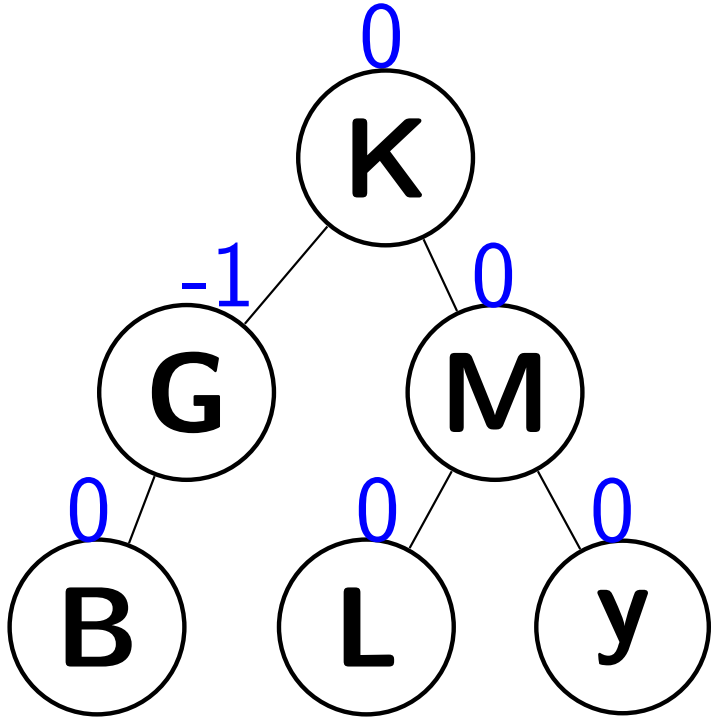
Árvore AVL - Rotação dupla à direita

```
Arvore* Rotacao_dupla_dir (Arvore *a) {
    a->esq = Rotacao_simples_esq (a->esq);
    return Rotacao_simples_dir (a);
}
```



Árvore AVL - Rotação dupla à direita

```
Arvore* Rotacao_dupla_dir (Arvore *a) {
    a→esq = Rotacao_simples_esq (a→esq);
    return Rotacao_simples_dir (a);
}
```



Sumário

Árvore AVL - Atualizar fator de balanceamento à esquerda

```
Arvore* Atualizar_fb_esq (Arvore *a) {
    a→altura = Atualizar(a);
    if (Balanceamento(a) == -2) {
        if (Balanceamento(a→esq) ≤ 0) {
            a = Rotacao_simples_dir (a);
        }
        else {
            a = Rotacao_dupla_dir (a);
        }
    }
    return a;
}
```

Sumário

Árvore AVL - Atualizar fator de balanceamento à direita

```
Arvore* Atualizar_fb_dir (Arvore *a) {
    a→altura = Atualizar(a);
    if (Balanceamento(a) == +2) {
        if (Balanceamento(a→dir) ≥ 0) {
            a = Rotacao_simples_esq (a);
        }
        else {
            a = Rotacao_dupla_esq (a);
        }
    }
    return a;
}
```

Sumário

Árvore AVL - Fator de balanceamento (funções auxiliares)

```
int Altura (Arvore *a) {
    return (a == NULL ? -1 : a->altura);
}

int Balanceamento (Arvore *a) {
    return (Altura(a->dir) - Altura(a->esq));
}

int Atualizar (Arvore *a) {
    return (maior(Altura(a->esq), Altura(a->dir)) + 1);
}
```

Sumário

Árvore AVL - Inserção

```
Arvore* Inserir (Arvore *a, char chave) {
  if (a == NULL)
      a = (No*)malloc(sizeof(No));
      a→chave = chave;
      a→altura = 0;
      a→esq = a→dir = NULL;
  else if (chave < a→chave)
      a→esq = Inserir (a→esq, chave);
      a = Atualizar_fb_esq (a);
  else
      a→dir = Inserir (a→dir, chave);
      a = Atualizar_fb_dir (a);
  return a;
```