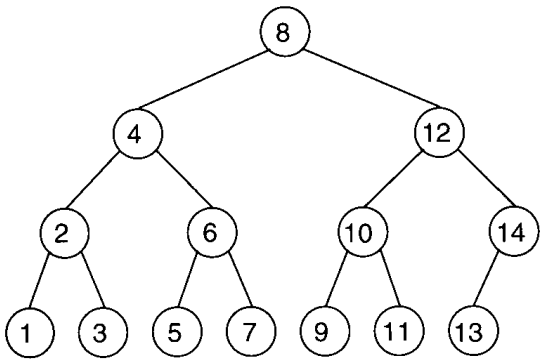
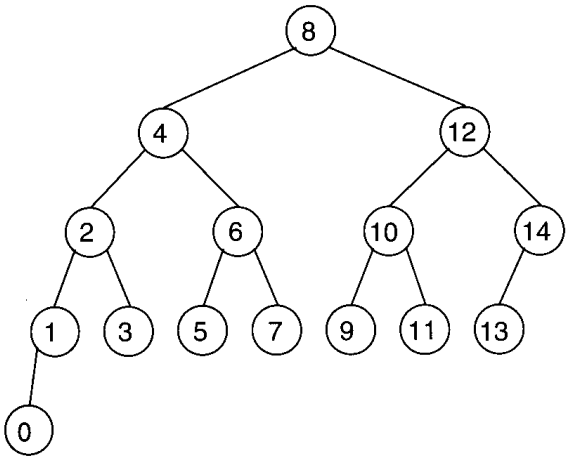
Árvore balanceada: se a árvore possui n nós, sua altura deve ser $O(log\, n)$. A altura de uma subárvore com m nós deve ser $O(log\, m)$.

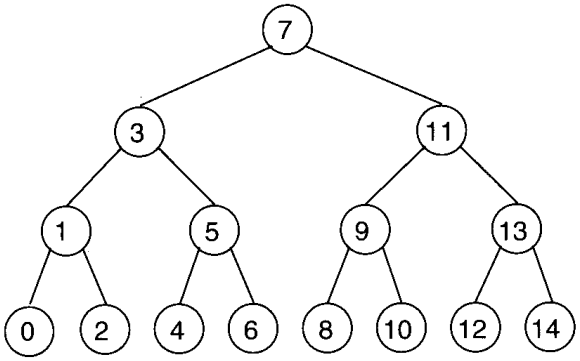
Inserir um nó de forma a manter uma ABB completa não é eficiente. Exemplo extraído de Szwarcfiter & Markenzon, pp.129.



(a)



(b)



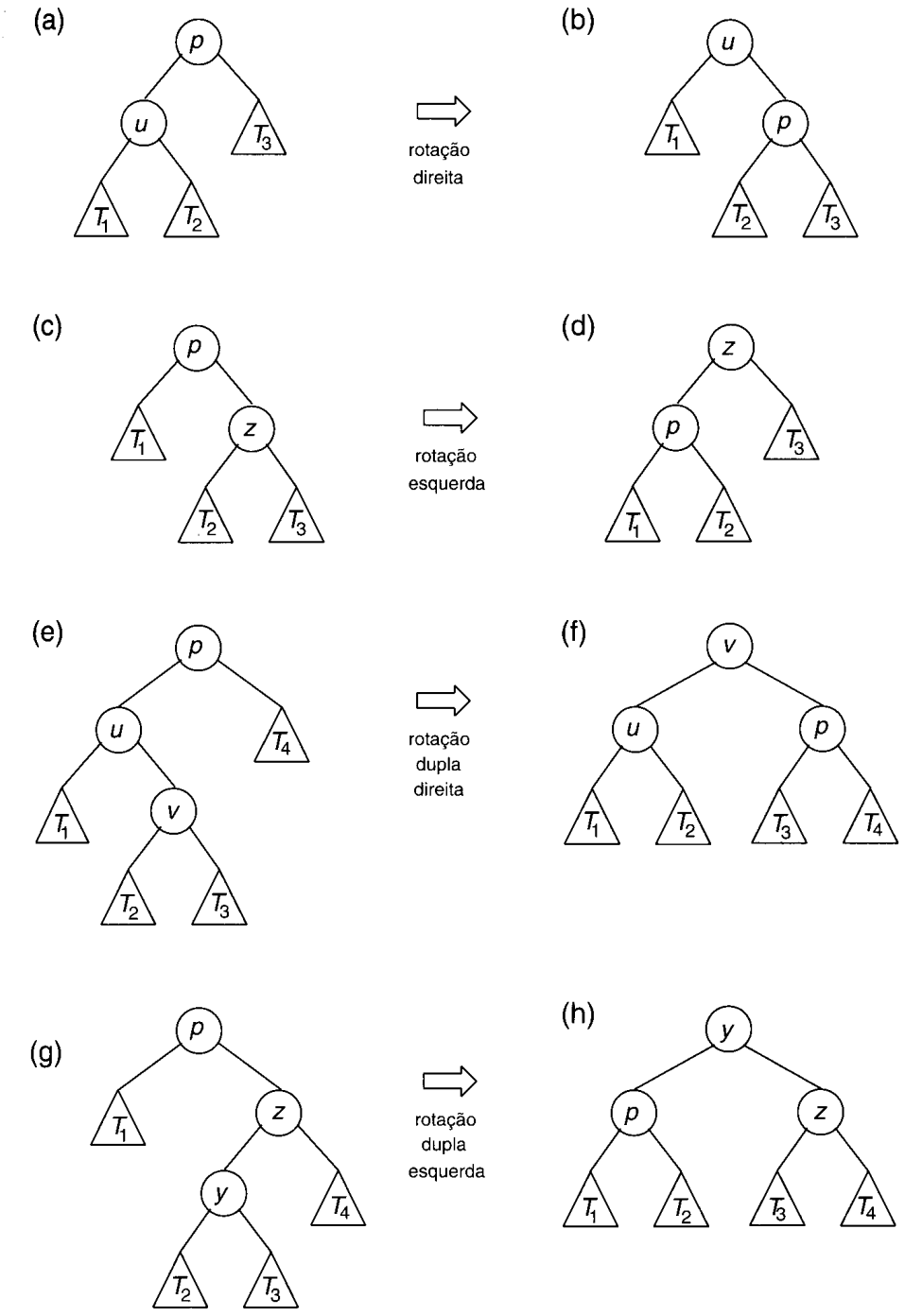
(c)

Árvores AVL: uma árvore binária T é uma árvore AVL se, para qualquer nó de T, as alturas de suas duas subárvores, esquerda e direita, diferem em módulo de até uma unidade.

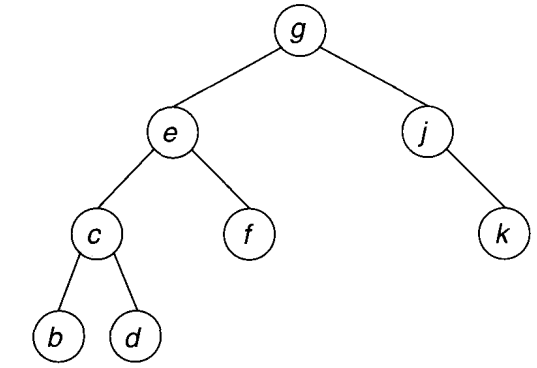
Pode-se mostrar que toda árvore AVL é balanceada. Ou seja, se ela tiver n nós, então sua altura é $O(log\, n)$.

Operações de inserção e remoção tendem a destruir a característica de balanceamento de uma árvore. Quando as subárvores de um nó tem diferença de altura superior a 1, o nó está desregulado. Para regular o nó, realizam-se operações de rotação.

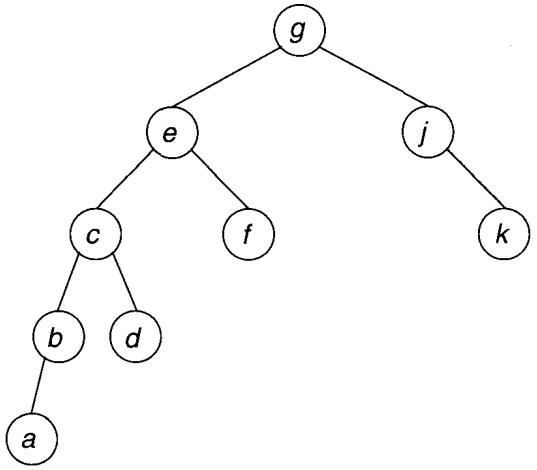
Tipos de rotações (Szwarcfiter & Markenzon, pp.135)



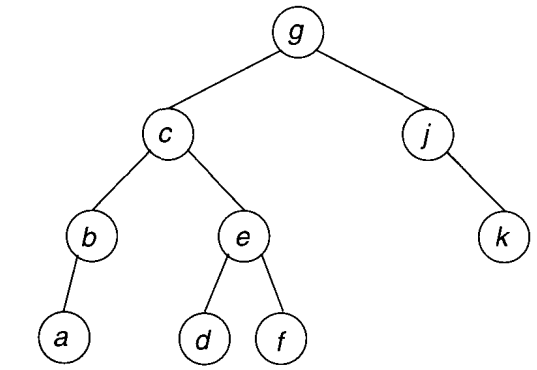
Inserção de um nó, seguido de regulação (Szwarcfiter & Markenzon, pp.138)



(a)



(b)



(c)

Algoritmo de inclusão (Szwarcfiter & Markenzon, pp.141)

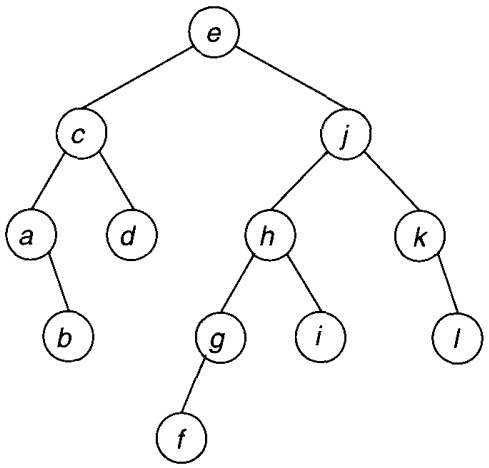
algoritmo 5.1: Busca e inserção em árvore AVL

```
procedimento inicio-no(pt)
    ocupar(pt)
    pt↑.esq := λ;   pt↑.dir := λ
    pt↑.chave := x;   pt↑.bal := 0

procedimento caso1(pt, h)
    ptu := pt↑.esq
    se ptu↑.bal = −1 então
            pt↑.esq := ptu↑.dir;   ptu↑.dir := pt
            pt↑.bal := 0;   pt := ptu
    senão ptv := ptu↑.dir
            ptu↑.dir := ptv↑.esq;   ptv↑.esq := ptu
            pt↑.esq := ptv↑.dir;   ptv↑.dir := pt
            se ptv↑.bal = −1 então pt↑.bal := 1 senão pt↑.bal := 0
            se ptv↑.bal = 1 então ptu↑.bal := −1 senão ptu↑.bal := 0
            pt := ptv
    pt↑.bal := 0;   h := F
```

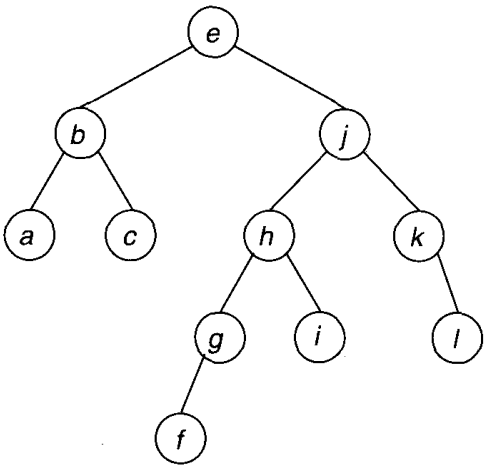
Algoritmo de inclusão, continuação (Szwarcfiter & Markenzon, pp.142)

```
procedimento caso2(pt, h)
    ptu := pt ↑ .dir
    se ptu ↑ .bal = 1 então
            pt ↑ .dir := ptu ↑ .esq;    ptu ↑ .esq := pt
            pt ↑ .bal := 0;    pt := ptu
    senão ptv := ptu ↑ .esq
            ptu ↑ .esq := ptv ↑ .dir;    ptv ↑ .dir := ptu
            pt ↑ .dir := ptv ↑ .esq;    ptv ↑ .esq := pt
            se ptv ↑ .bal = 1 então pt ↑ .bal := −1 senão pt ↑ .bal := 0
            se ptv ↑ .bal = −1 então ptu ↑ .bal := 1 senão ptu ↑ .bal := 0
            pt := ptv
    pt ↑ .bal := 0;    h := F

procedimento ins-AVL(x, pt, h)
    se pt = λ então
            inicio-no(pt)
            h := V
    senão se x = pt ↑ .chave então pare
            se x < pt ↑ .chave então
                    ins-AVL(x, pt ↑ .esq, h)
                    se h então
                            caso pt ↑ .bal seja
                             1:    pt ↑ .bal := 0;    h := F
                             0:    pt ↑ .bal := −1
                            -1:    caso1(pt, h)          % rebalanceamento
            senão ins-AVL(x, pt ↑ .dir, h)
                    se h então
                            caso pt ↑ .bal seja
                            -1:    pt ↑ .bal := 0;    h := F
                             0:    pt ↑ .bal := 1
                             1:    caso2(pt, h)          % rebalanceamento
```

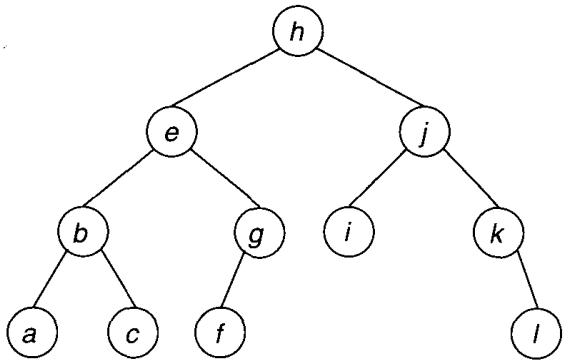
Remoção de um nó, seguido de regulação (Szwarcfiter & Markenzon, pp.143)



(a)



(b)



(c)