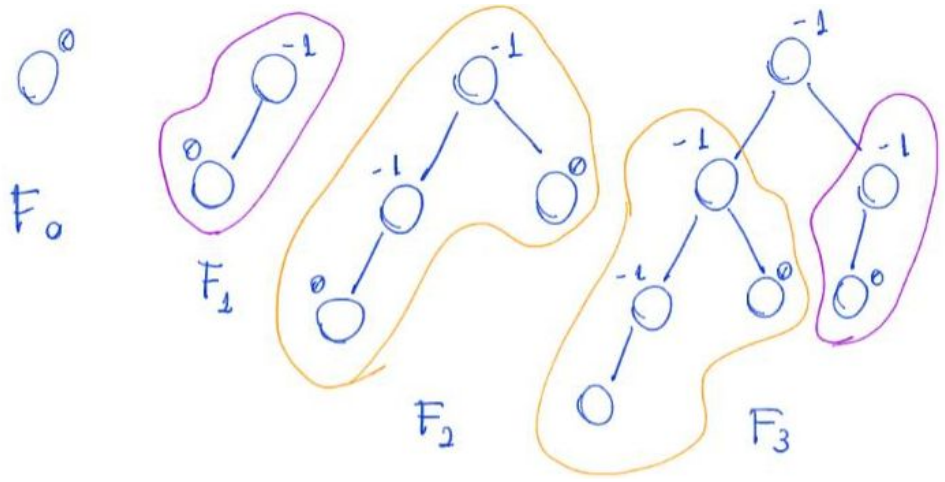
AED2 - Aula 04
Árvores AVL: altura máxima e remoção

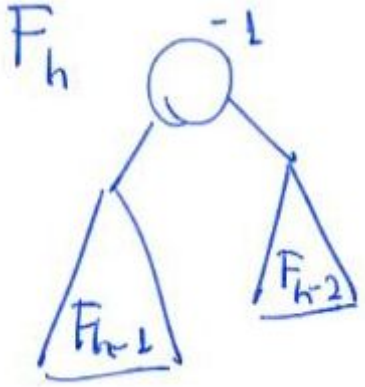
Altura máxima de árvores AVL

Quão desbalanceada pode ser uma árvore AVL?

Considere os seguintes exemplos:



Observe que a regra de formação dessas árvore é



- ou seja, F_h é composta por um nó raiz cujo
    - filho esquerdo é F_h-1
    - e o filho direito é F_h-2.

Note que, F_h é a árvore AVL de altura h com o menor número de nós possível.

- Isso porque, pensando recursivamente, tal árvore precisa ter:
    - uma subárvore com altura h - 1,
    - outra com altura h - 2,
    - e ambas as subárvores devem ter o menor número de nós possível.

Seja N(h) o número de nós da árvore F_h. Temos que

- N(h) = N(h - 1) + N(h - 2) + 1, para h >= 2

- N(0) = 1
- N(1) = 2

Vamos analisar o padrão desta recorrência

- N(0), N(1), N(2), N(3), N(4), N(5), N(6), ...
- 1, 2, 4, 7, 12, 20, 33, ...

E compará-la com o padrão da sequência de Fibonacci

- Fib(0), Fib(1), Fib(2), Fib(3), Fib(4), Fib(5), Fib(6), Fib(7), Fib(8), ...
- 1, 1, 2, 3, 5, 8, 13, 21, 34, …

Comparando as sequências podemos perceber que

- N(h) = Fib(h + 2) - 1,
    - o que pode ser provado usando indução matemática.

Mas, Fib(h + 2) >= 2^(h / 2).

- Para verificar isso, perceba que
    - Fib(h + 2) = Fib(h + 1) + Fib(h) >= Fib(h) + Fib(h) = 2 Fib(h).
- Ou seja, a cada dois incrementos no índice h
    - o valor na sequência de Fibonacci pelo menos dobra.

Portanto,

- N(h) = Fib(h + 2) - 1 >= 2^(h / 2) - 1
- h / 2 <= lg (N(h) + 1)
- h <= 2 lg (N(h) + 1)

Como o número de nós n de qualquer árvore AVL de altura h

- é maior ou igual a N(h), i.e., n >= N(h), temos
    - h <= 2 lg (N(h) + 1) <= 2 lg (n + 1),
- Ou seja, a altura de uma árvore AVL é O(log n).

Bônus:

- É possível fazer uma análise mais precisa
    - em que mostramos que Fib(h) >= 1,618^h,
        - valor que deriva da razão aurea.
- Usando esse limitante inferior mais preciso para Fib temos
    - N(h) = (Fib(h + 2) - 1) >= 1,618^(h + 2) - 1,
    - h + 2 <= lg (N(h) + 1) / lg 1,618 <= 1,44 lg (N(h) + 1).
- Portanto,
    - h <= 1,44 lg (n + 1) = O(log n).

Remoção em árvores AVL

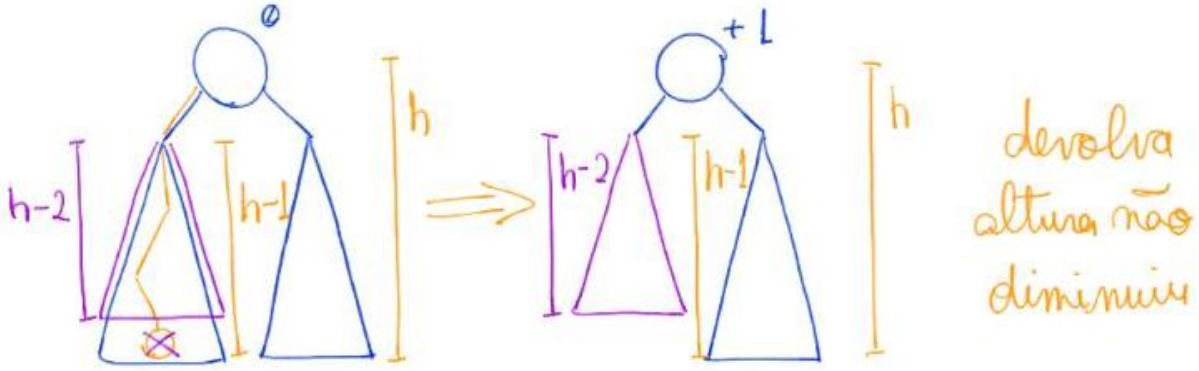
Similar aos casos da inserção, mas um tanto mais complexo.

Supomos que o algoritmo recursivo de remoção começa
- transformando a remoção de qualquer nó na remoção de uma folha,
    - como ocorre nas árvores binárias de busca comuns,
- e então analisamos o que precisa ser feito na volta da recursão
    - quando a altura de uma das subárvores diminui após uma remoção.

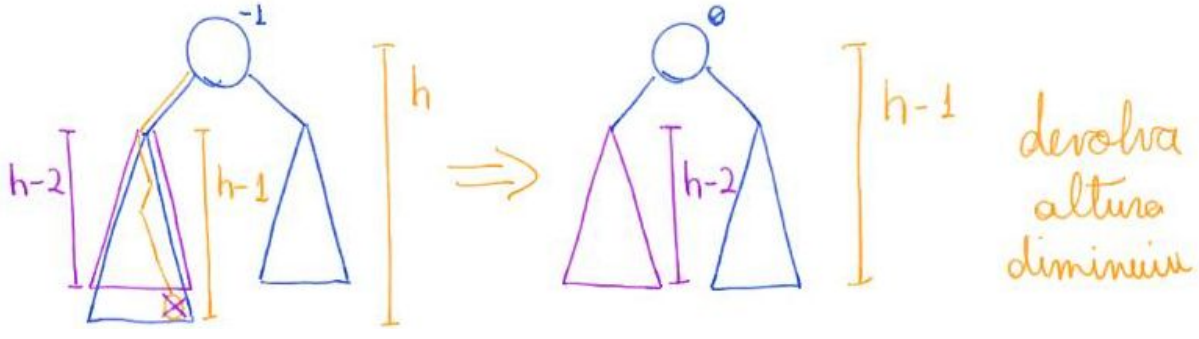
Caso 0: se a altura da subárvore em que ocorreu a remoção não diminuiu,
- o algoritmo não precisa realizar correções
- e devolve que a altura da sua árvore não diminuiu.

Caso 1: se a sua árvore era uma folha,
- remova a folha,
- e devolva que a altura diminuiu.

Caso 2: se a altura das duas subárvores era igual (i.e., balanceamento da raiz era 0) e a altura da subárvore em que ocorreu a remoção diminuiu,
- corrige o fator de balanceamento
- e devolve que a altura da sua árvore não diminuiu.

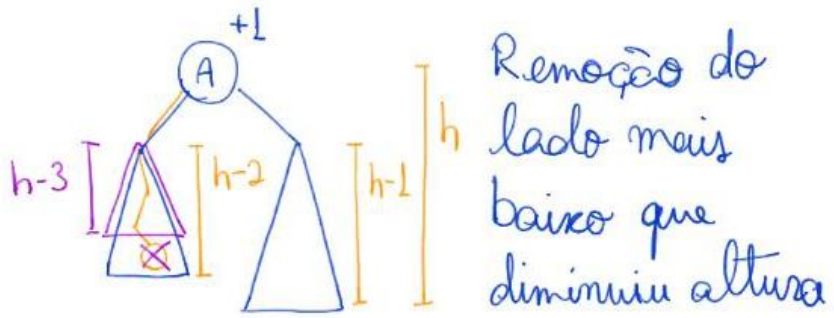


Caso 3: se removeu da subárvore mais alta e a altura desta diminuiu
- corrige o fator de balanceamento para 0
- e devolve que a altura da sua árvore diminuiu.

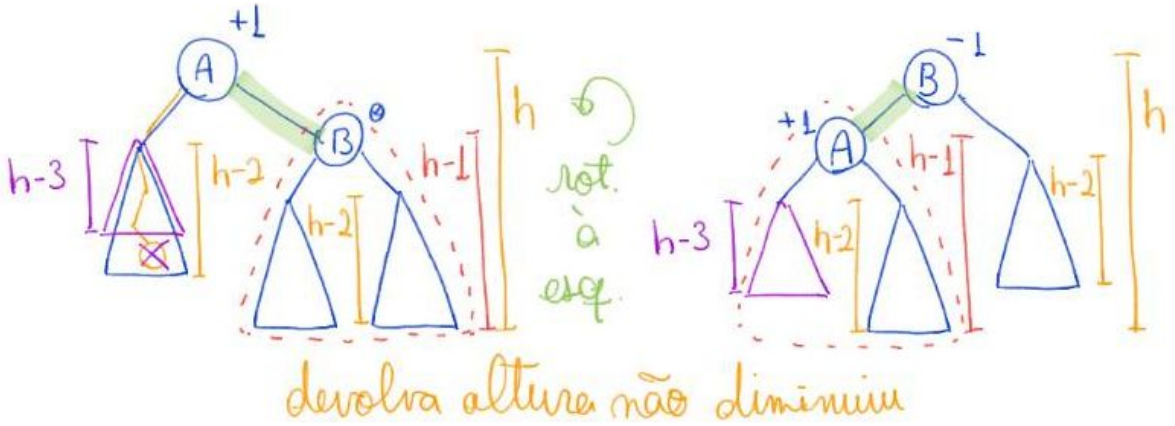


Caso 4: se removeu da subárvore mais baixa e a altura diminuiu

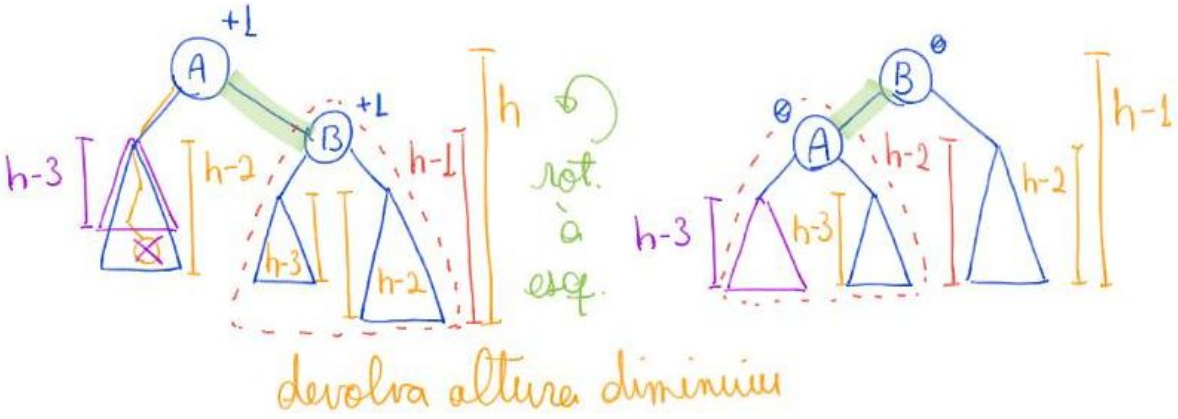
- é preciso realizar uma ou mais rotações para restaurar a propriedade AVL.



- Caso 4.1: fator de balanceamento 0 na raiz da subárvore mais alta.



- Caso 4.2 - fator de balanceamento +1 na raiz da subárvore mais alta.



- Caso 4.3 - fator de balanceamento -1 na raiz da subárvore mais alta.