"Meça sete vezes, corte uma" (Provérbio Russo).

AVL

Paulo Ricardo Lisboa de Almeida

DSBD

UFPR

O problema

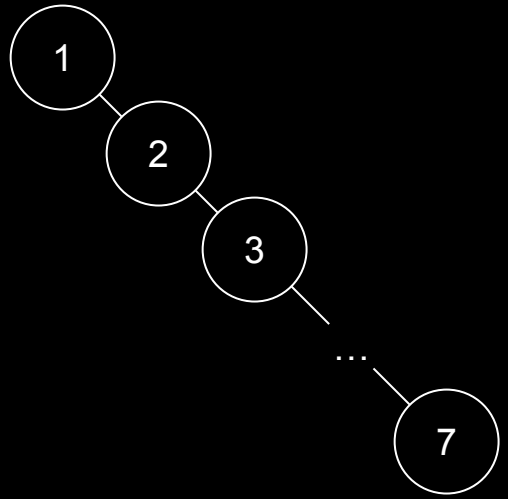
Uma árvore de busca binária pode ter custo O(n) se estiver desbalanceada.

Exemplo: incluir os valores 1, 2, 3, 4, 5, 6 e 7 em uma BST.

O problema

Uma árvore de busca binária pode ter custo O(n) se estiver desbalanceada.

Exemplo: incluir os valores 1, 2, 3, 4, 5, 6 e 7 em uma BST.



Desejamos árvores que sejam capazes de se balancear automaticamente.

Balanceada

Árvore balanceada: A altura da árvore *T* tem a mesma ordem de grandeza de uma árvore completa.

Qual a altura?

Balanceada

Árvore balanceada: A altura da árvore *T* tem a mesma ordem de grandeza de uma árvore completa.

O(log$_2$ n).

A propriedade se estende às suas subárvores.

Toda subárvore de *T* com *m* nodos possui altura *O(log$_2$ m)*.

AVL

Uma árvore AVL é um tipo de árvore de busca binária balanceada.

Proposta por **A**delson-**V**elsky e **L**andis.

An algorithm for the organization of information
by G. M. Adel'son-Vel'skii and E. M. Landis
Soviet Mathematics Doklady, 3, 1259-1263, 1962
http://monet.skku.ac.kr
Networkging Lab.

Thus (5.1) is not true for this function for $l = 1$ and $r = 1$.

Received 22/MAR/62

BIBLIOGRAPHY

[1] C. Miranda, *Equazioni alle derivate parziali di tipo ellittico*, Springer, Berlin, 1955.
[2] S. M. Nikol'skiĭ, Sibirsk. Mat. Ž. 1 (1960), 78.

Translated by:
F. M. Goodspeed

AN ALGORITHM FOR THE ORGANIZATION OF INFORMATION

G. M. ADEL'SON-VEL'SKIĬ AND E. M. LANDIS

In the present article we discuss the organization of information contained in the cells of an automatic calculating machine. A three-address machine will be used for this study.

Statement of the problem. The information enters a machine in sequence from a certain reserve. The information element is contained in a group of cells which are arranged one after the other. A certain number (the information estimate), which is different for different elements, is contained in the information element. The information must be organized in the memory of the machine in such a way that at any moment a very large number of operations is not required to scan the information with the given evaluation and to record the new information element.

An algorithm is proposed in which both the search and the recording are carried out in $C \lg N$ operations, where N is the number of information elements which have entered at a given moment.

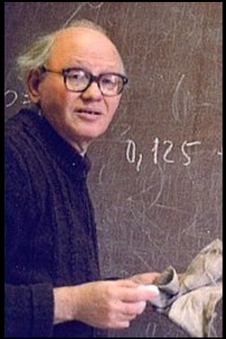
A part of the memory of the machine is set aside to store the incoming information. The information elements are arranged there in their order of entry. Moreover, in another part of the memory a "reference board" [1] is formed, each cell of which corresponds to one of the information elements.

The reference board is a dyadic tree (Figure 1a): each of its cells has no more than one left cell, and no more than one right cell subordinated to it. Direct subordination induces subordination (partial ordering). In addition, for each cell of the tree, all the cells which are subordinate to a left (right) directly subordinate cell, will be arranged further to the left (right) than the given cell. Moreover, we assume that there is a cell (the head) to which all the others are subordinate. By transitivity, the conception "further to the left" and "further to the right" extends to the aggregate of all the cell pairs, and this aggregate becomes ordered. Thus, a given order of cells in a reference board should coincide with the order of arrangement of the estimates of the corresponding information elements (to be specific, we shall consider the estimates as increasing from left to right).

In the first address of each cell of the reference board, a place is indicated where the corresponding information element is located. The addresses of the cells of the reference board, which are directly subordinate on the left and right respectively to the given cell, are located in the second and third addresses. If a cell has no directly subordinate cells on either side, then there is zero in the corresponding address. The head address is stored in a certain fixed cell l.

Let us call the sequence of the cells of the tree a *chain* in which each previous cell is directly

Adelson-Velskii, M., & Landis, E. M. *An algorithm for the organization of information.* Doklady Akademii Nauk SSSR, 146: 263–266. 1963.



Georgy Adelson-Velsky
08/01/1922 - 26/04/2014

Cientista da Computação Russo.
- Árvore AVL

en.wikipedia.org/wiki/Georgy_Adelson-Velsky

Yevgeniy Landis
06/10/1921 - 12/12/1997

Cientista da Computação Russo.
- Árvore AVL

en.wikipedia.org/wiki/Evgenii_Landis

Propriedade da AVL

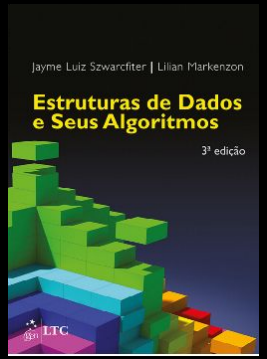
Propriedade da AVL: Uma árvore binária T é AVL se $\forall$ nó de T, as alturas de suas duas subárvores, esquerda e direita, diferem em módulo de até uma unidade.

Um nodo que respeita essa propriedade é dito **regulado**.

Caso contrário, é **desregulado**.

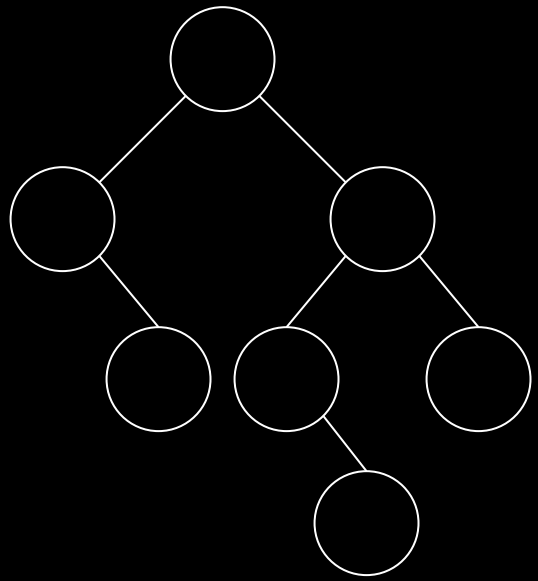
A árvore é balanceada.

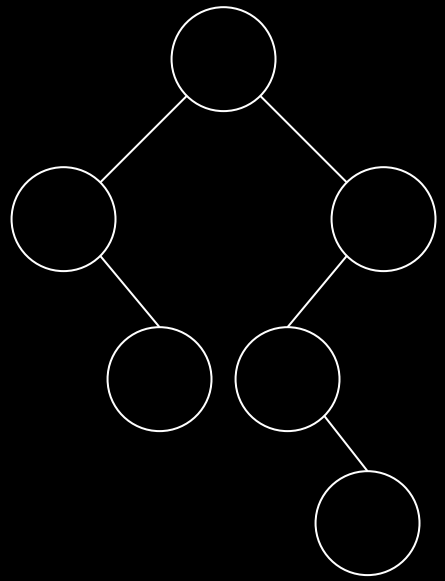
Veja a prova em Szwarcfiter e Markenzon (2010).



Pergunta

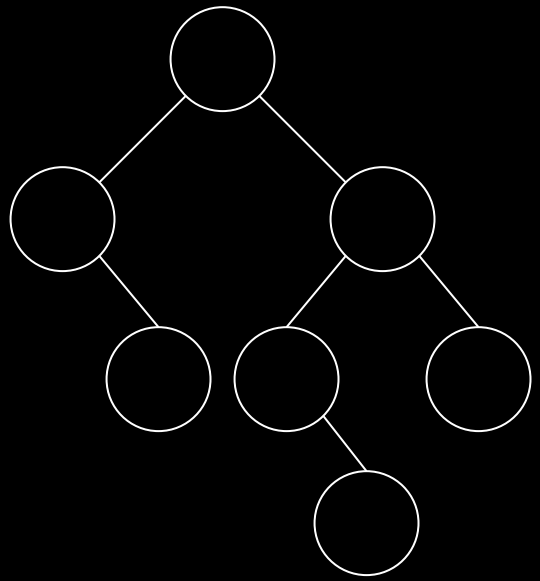
As árvores a seguir são AVLs?



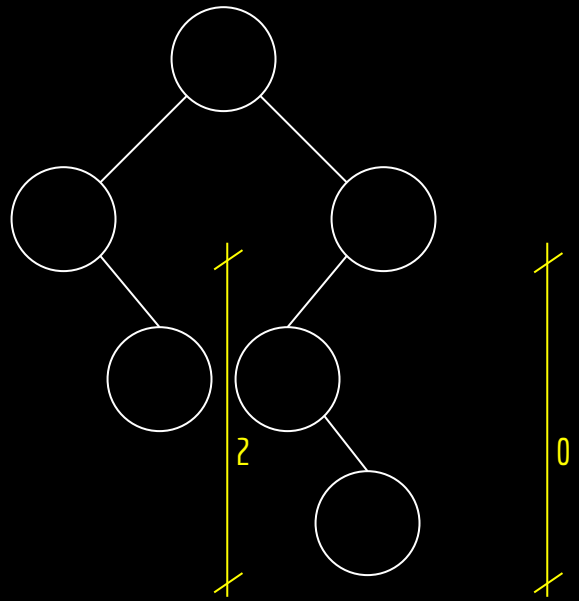


Pergunta

As árvores a seguir são AVLs?



AVL.



Não é AVL.

Inclusão

Seja T uma árvore AVL.

Ao realizar uma inclusão, determinada subárvore pode ficar desregulada.

Necessário regular novamente para que a árvore permaneça balanceada.

Casos

Considerando uma subárvore p de T, demos dois casos ao inserir um filho direito de p.

Caso 1: inserir um nodo na subárvore direita do filho direito de p.

Caso 2: inserir um nodo na subárvore esquerda do filho direito de p.

Casos

Considerando uma subárvore p de T, demos dois casos ao inserir um filho direito de p.

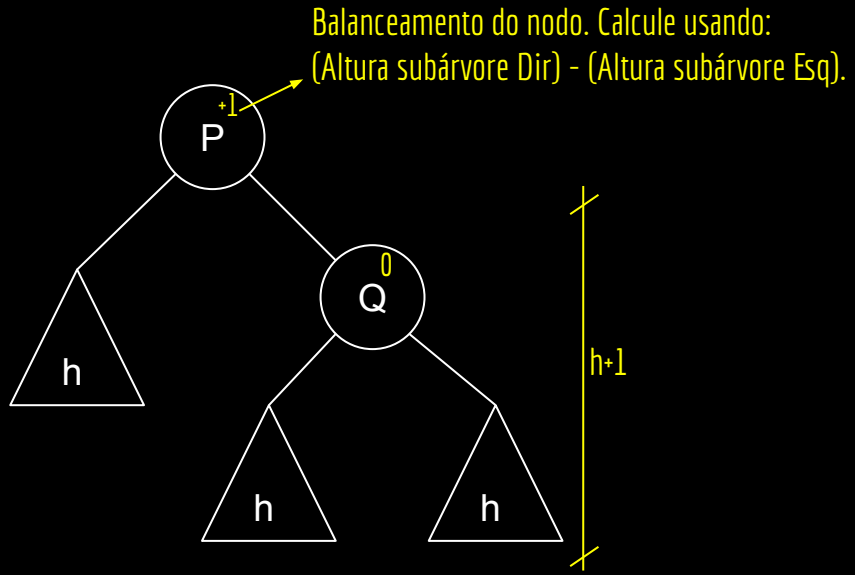
- Caso 1: inserir um nodo na subárvore direita do filho direito de p.
- Caso 2: inserir um nodo na subárvore esquerda do filho direito de p.

Temos mais dois casos simétricos ao inserir um filho esquerdo de p.

- Fica como exercício definir esses casos.

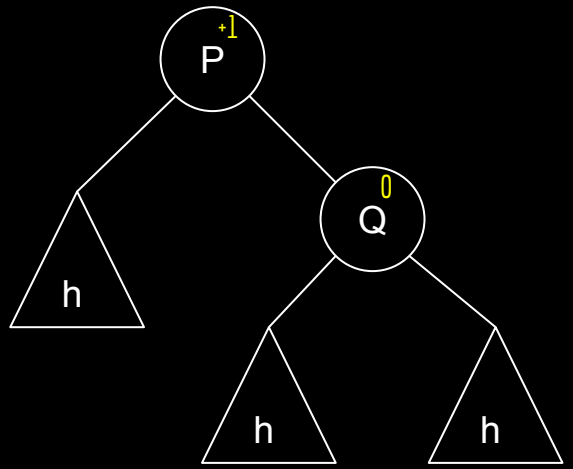
Novo nodo na subárvore direita do filho direito

Antes da inserção.

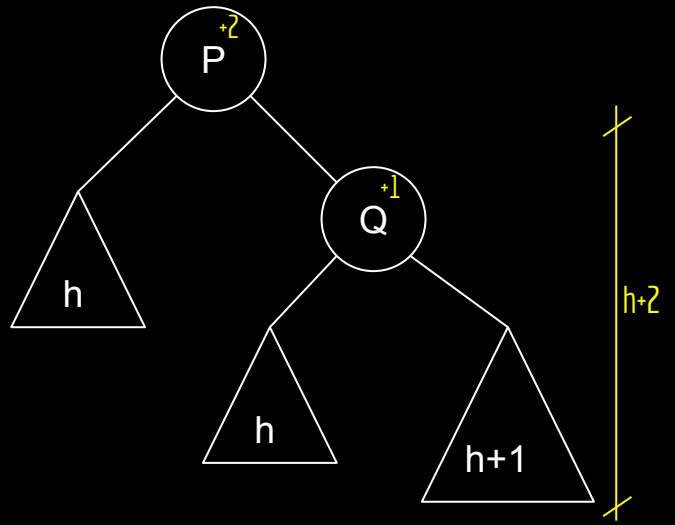


Novo nodo na subárvore direita do filho direito

Antes da inserção.

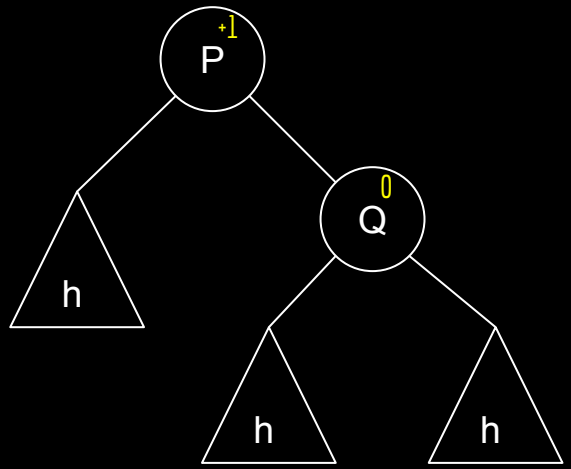


Depois da inserção.

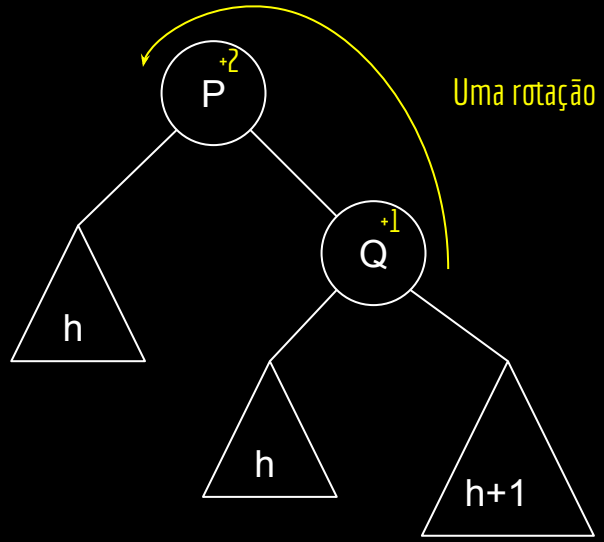


Novo nodo na subárvore direita do filho direito

Antes da inserção.



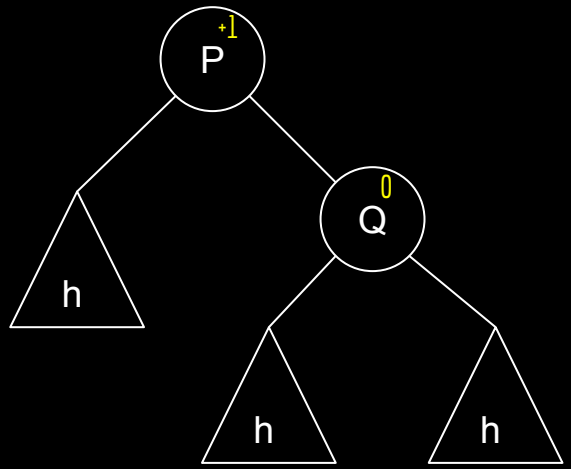
Depois da inserção.



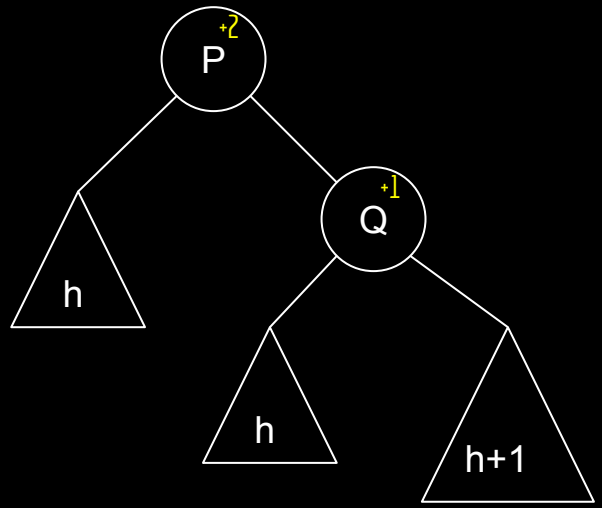
Uma rotação para a esquerda.

Novo nodo na subárvore direita do filho direito

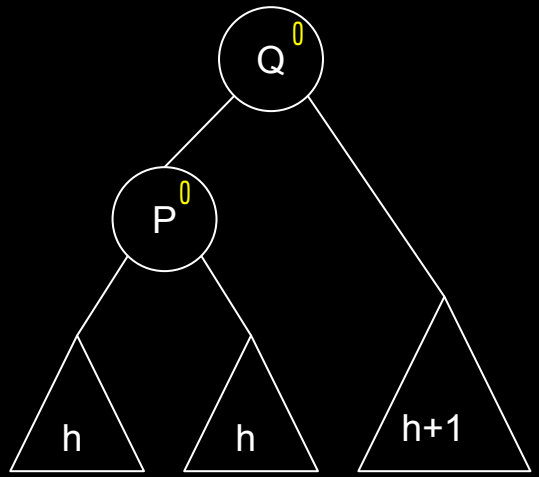
Antes da inserção.



Depois da inserção.

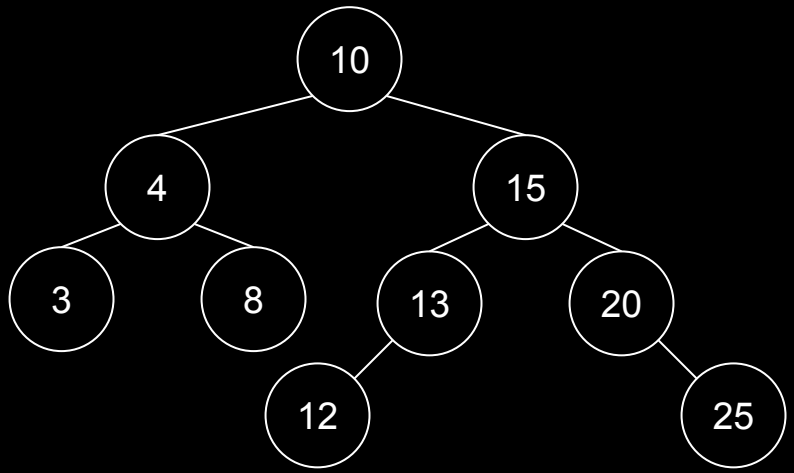


Depois da rotação.



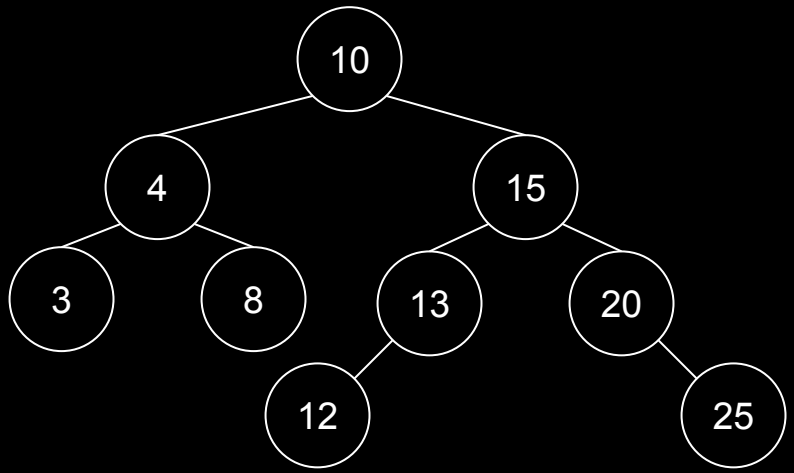
Faça você mesmo

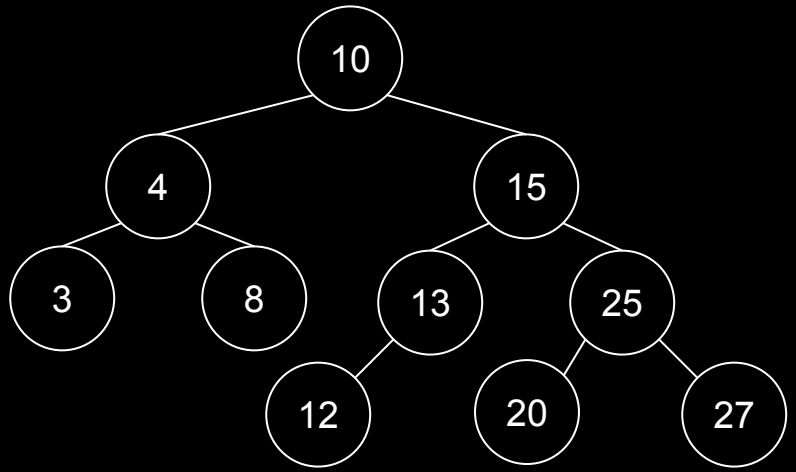
Insira um nodo de chave 27 na árvore a seguir. Mantenha a propriedade da AVL.



Faça você mesmo

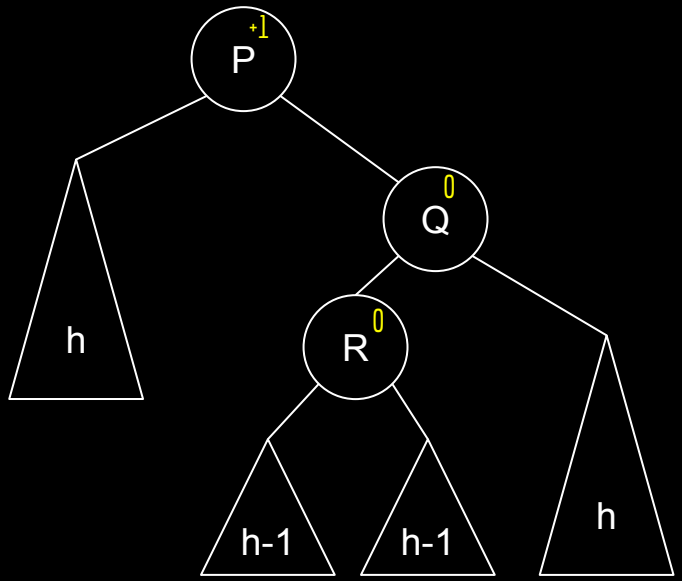
Insira um nodo de chave 27 na árvore a seguir. Mantenha a propriedade da AVL.





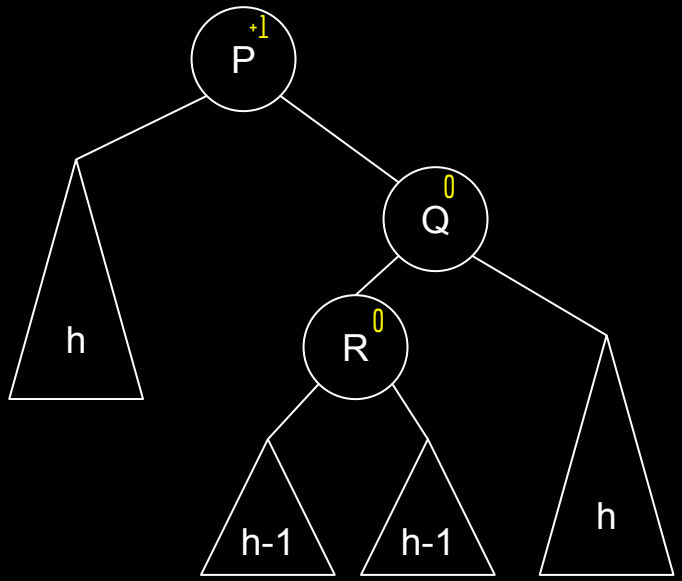
Novo nodo na subárvore esquerda do filho direito

Antes da inserção.

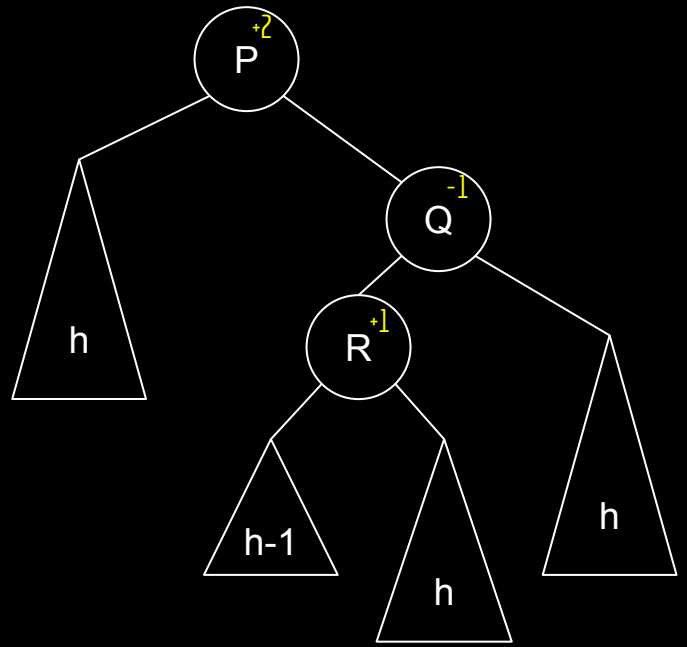


Novo nodo na subárvore esquerda do filho direito

Antes da inserção.



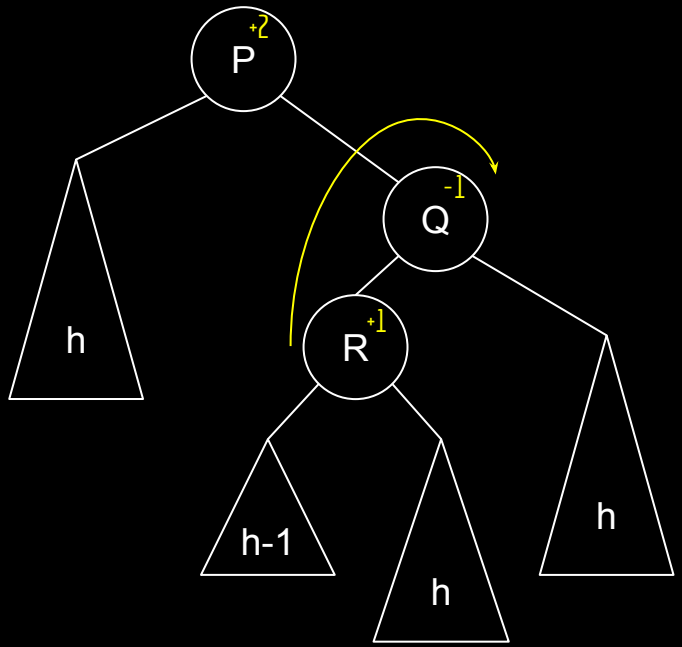
Depois da inserção.



Rotação Dupla Direita

Primeiro uma rotação para direita entre R e Q.

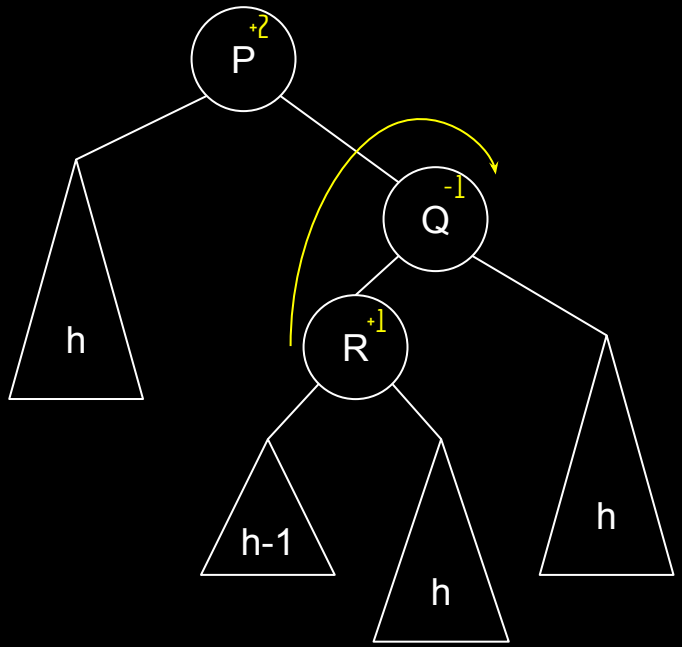
Depois da inserção.



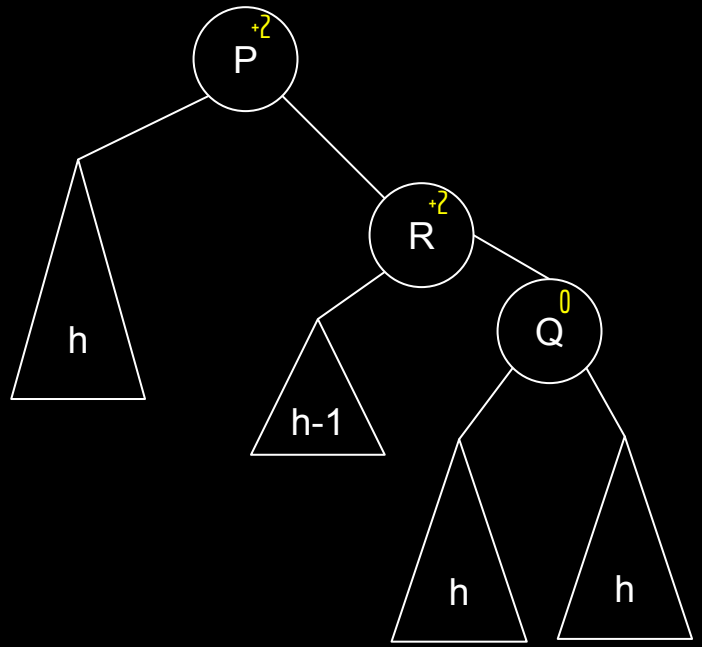
Rotação Dupla Direita

Primeiro uma rotação para direita entre R e Q.

Depois da inserção.

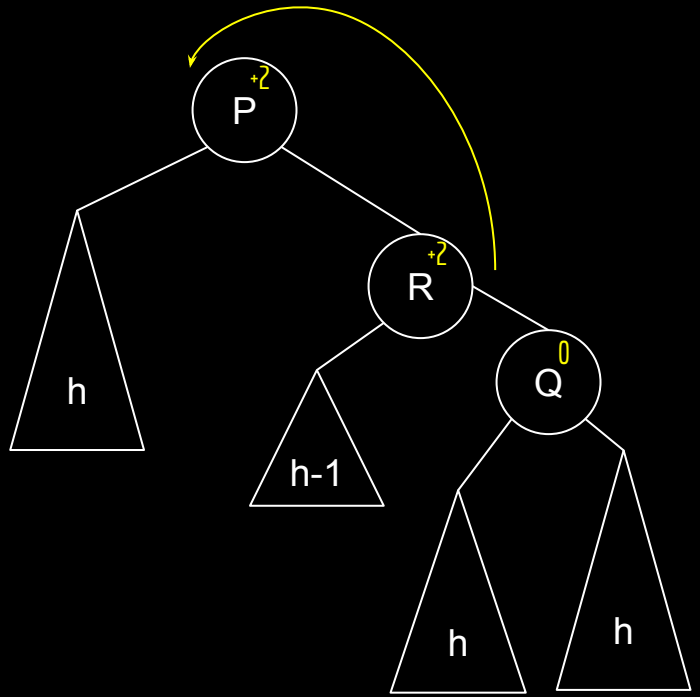


Primeira Rotação.



Rotação Dupla Direita

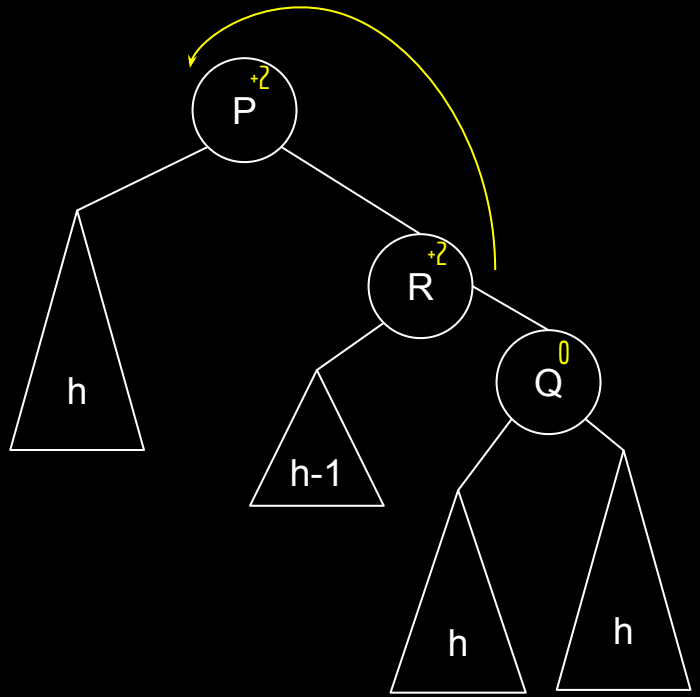
Depois uma rotação para a esquerda entre R e P.

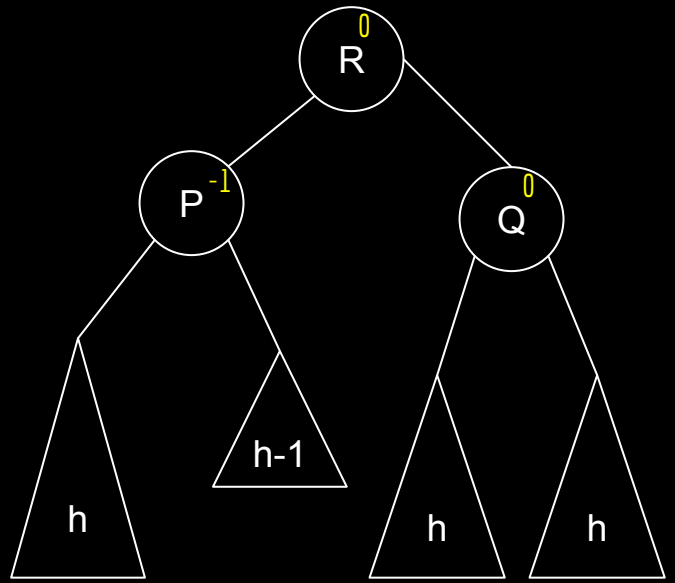


Rotação Dupla Direita

Depois uma rotação para a esquerda entre R e P.

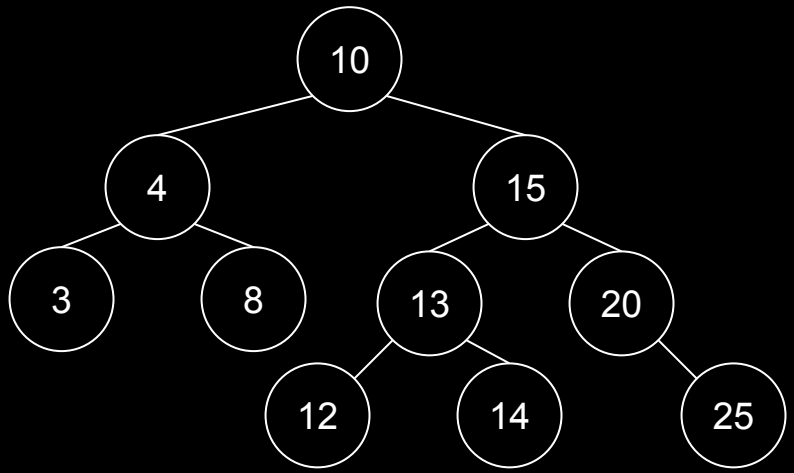
Primeira Rotação.





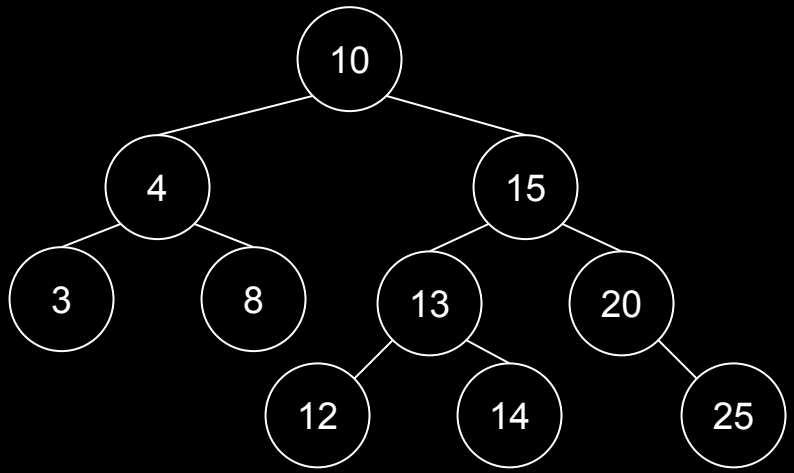
Faça você mesmo

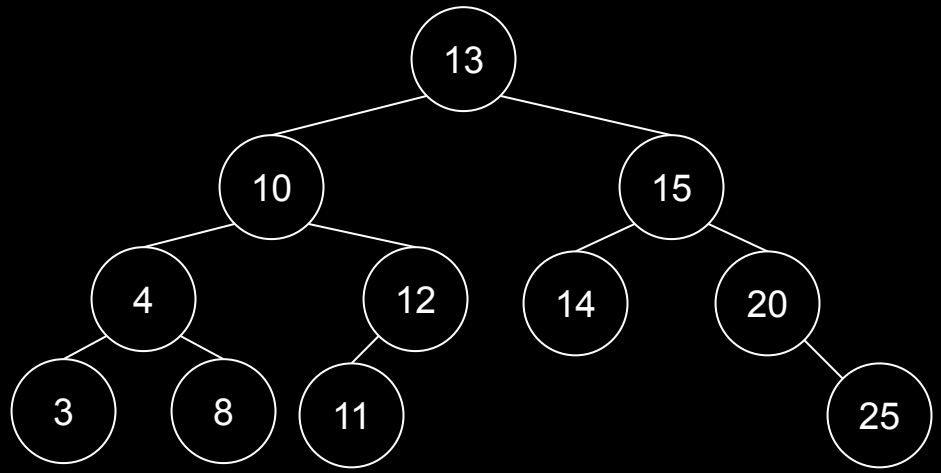
Insira um nodo de chave 11 na árvore a seguir. Mantenha a propriedade da AVL.



Faça você mesmo

Insira um nodo de chave 11 na árvore a seguir. Mantenha a propriedade da AVL.





Dica

A Rotação foi feita em dois passos para facilitar o entendimento.

Podemos fazer em um único passo.

Veja na literatura.

Está desbalanceado?

Ao inserir um nodo, a árvore pode ou não se tornar desbalanceada.

Como encontrar P e Q (se existirem)?

Está desbalanceado?

```
função atualizarBalanco(q)
entrada: nodo q inserido na árvore
saída: a árvore AVL balanceada.

pai = q.pai
se q é filho esquerdo de p
     p.balanco--
senão
     p.balanco++
enquanto p não é raiz e p.balanco ≠ ± 2
     q = p
     p = p.pai
     se q.balanco é 0
          retorne
     se q é um filho esquerdo de p
          p.balanco--
     senão
          p.balanco++
se p->balanco == ± 2
     rebalancear a subárvore com raiz em P
```

Excluir

Para excluir um item:

1. Exclua o item normalmente usando, por exemplo, o algoritmo usado para árvores de busca binária.
    a. Cuidado para atualizar os balanços dos nodos transplantados.
2. Atualize o balanço de todos os pais a partir do nodo excluído.
    a. Diferente da inserção, ao excluir um nodo podemos desbalancear a árvore toda, até a raiz. É necessário verificar todos os pais.

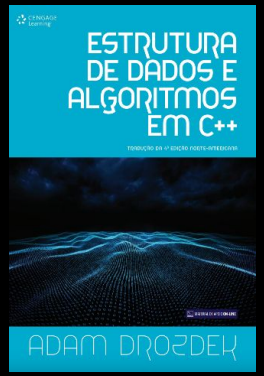
Custo

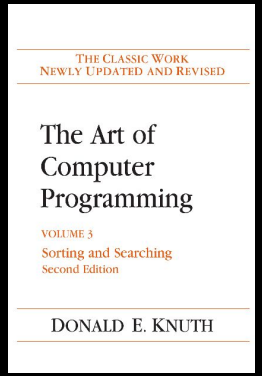
Como uma AVL aceita que uma subárvore tenha uma diferença de altura entre suas subárvores esquerda e direita de -1 ou +1, a árvore não é perfeitamente balanceada.

O custo ainda é $O(\log_2 n)$.

No entanto, a árvore pode precisar de 44% mais comparações em suas operações do que uma árvore perfeitamente balanceada.

Veja uma discussão em Drozdek (2016) e em Knuth (1998).





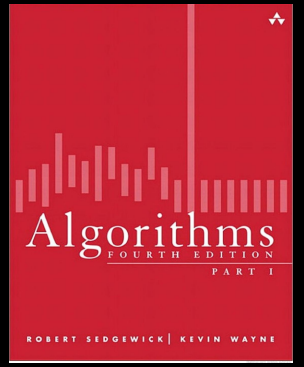
Exercícios

1. Responda e justifique:
    a. Toda árvore completa é uma AVL?
    b. Toda árvore AVL é completa?
2. Verifique que todas operações de rotação mantém a propriedade de uma árvore de busca binária.

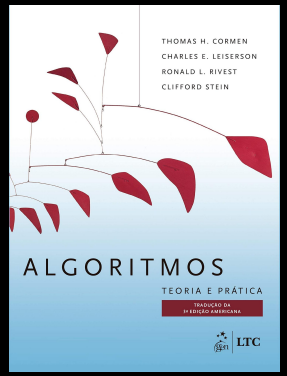
Referências

Estrutura de Dados e Algoritmos em C++. A. Drozdek. 4a ed. 2016.

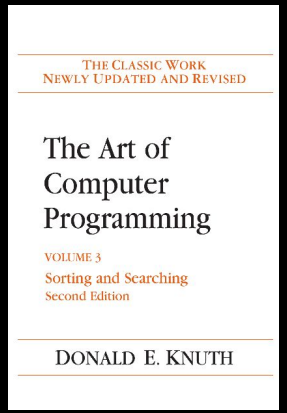
R. Sedgewick, K. Wayne. Algorithms Part I. 4a ed. 2011.



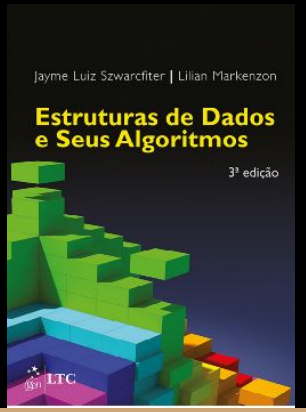
T. Cormen, C. Leiserson, R. Rivest, C. Stein. Algoritmos: Teoria e Prática. 3a ed. 2012.



Knuth, D. The Art of Computer Programming: Volume 3: Sorting and Searching. 1998.



Szwarcfiter, Markenzon, L. Estruturas de dados e seus algoritmos.2010.



Licença