

Algoritmos e Estruturas de Dados

qua 18 dez 2024 19:00:24 -03

Vamos definir nossa linguagem para trabalhar com árvores nesta aula. Dado um nó x de nossa árvore,

- $x.left$ é o filho esquerdo de x . Se ele não existir seu valor é NIL
- $x.right$ é o filho direito de x . Se ele não existir seu valor é NIL
- $x.key$ é o valor que x representa na árvore.

Árvores são estruturas naturalmente recursivas. Se tomarmos um nó x qualquer de uma árvore podemos falar em uma *subárvore enraizada em x* . Assim, podemos definir uma árvore como:

- um nó raiz r com uma chave $r.key$
- um apontador para uma subárvore esquerda $r.left$
- um apontador para uma subárvore direita $r.right$

A figura (fonte abaixo é um bom resumo do vocabulário usado em árvores.

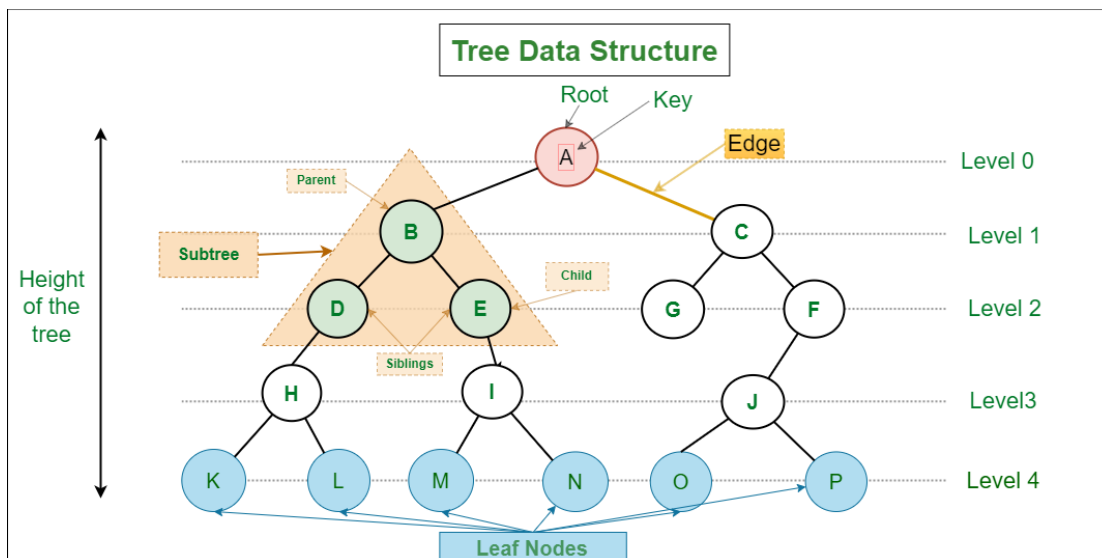


Figure 1: Fonte: <https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/>

Uma *Árvore de Busca Binária* é uma árvore especial que guarda seus elementos em ordem crescente de *key*. Para isso ela tem a seguinte propriedade:

Propriedade básica da ABB

Para todo nó x em uma *ABB*:

- $x.key \geq l.key$ para todo nó l na subárvore esquerda de x
- $x.key \leq r.key$ para todo nó r na subárvore direita de x

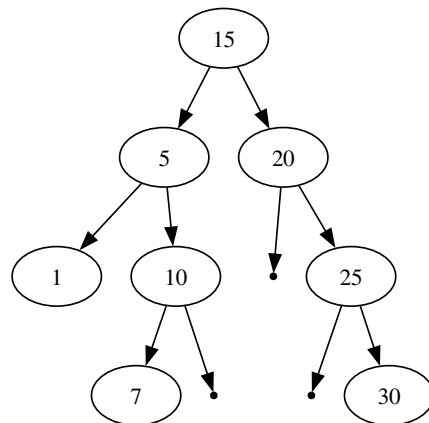
Se uma árvore não respeitar as propriedades acima então ela não é uma *ABB*.

Isso facilita muito checarmos se um elemento está na árvore.

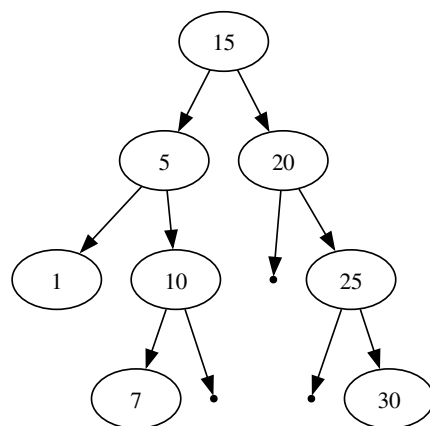
Buscando por elementos

A busca (*query*) é a operação mais básica de uma *ABB* e envolve dizer se existe um elemento com chave k na árvore. É também um ótimo lugar para começarmos a entender como aplicar a *Propriedade básica da ABB*.

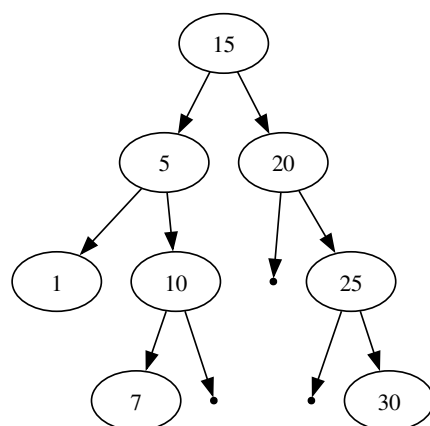
Faremos a busca do valor **7** na árvore abaixo. Desenhe em cima da árvore as decisões tomadas em cada nó, começando na raiz e descendo até encontrar o valor **7**.



Agora faça a busca pelo valor **27**. Desenhe em cima da árvore as decisões tomadas em cada nó. O que acontece quando chegamos em um nó raiz?



Agora faça a busca pelo valor **13**. Desenhe em cima da árvore as decisões tomadas em cada nó. Como descobrimos que esse valor **não** está na árvore?



Vamos agora formalizar essas simulações no algoritmo `QUERY(R, K)` que busca o valor `K` na árvore enraizada em `R`.

Versão iterativa:

Versão recursiva:

Implementação

Todos exercícios do PrairieLearn usam o seguinte `struct`. Ele é basicamente idêntico à maneira que escrevemos nosso pseudo-código.

```
typedef struct _TreeNode {
    struct _TreeNode *left, *right;
    int key;
} TreeNode;
```

Validar uma *ABB*

Vamos começar vendo alguns exemplos de árvores que não são *ABB*.

!!! exercise choice

Para qual nó da árvore abaixo a propriedade básica da *ABB* não vale?

```
{% set graph %}
digraph G {
  node [fillcolor="#d9dfff" color="#d9dfff" style="filled"]
  15 -> 14
  15 -> 21
  14 -> 10
  14 -> 16
  21 -> 17
  21 -> null1
  # sdlfk
  null1 [shape=point]
  null2 [shape=point]
  null3 [shape=point]
}
{% endset %}
<graphviz-graph graph='{{ graph }}'></graphviz-graph>
```

```
- [x] 15
- [ ] 14
- [ ] 10
- [ ] 16
- [ ] 21
- [ ] 17
```

!!! exercise choice Para qual nó da árvore abaixo a propriedade básica da *ABB* não vale?

```
{% set graph %}
digraph G {
  node [fillcolor="#d9dfff" color="#d9dfff" style="filled"]
  17 -> 14
  17 -> 22
  14 -> 10
  14 -> 16
  22 -> 21
  22 -> null1
  21 -> 18
  21 -> 23
  # sdlfk
  null1 [shape=point]
}
{% endset %}
<graphviz-graph graph='{{ graph }}'></graphviz-graph>
```

```
- [ ] 17
- [ ] 14
- [x] 22
```

-
- [] 10
 - [] 16
 - [] 21
 - [] 18
 - [] 23

Percebemos que um algoritmo para isso precisa de várias partes!

1. precisamos checar se a propriedade é válida para todos os nós
2. para checar a propriedade precisamos percorrer toda a subárvore esquerda e toda a sub-árvore direita.

Esses dois passos tem algo em comum: eles envolvem visitar todos os nós da árvore.

!!! exercise long Escreva abaixo um rascunho de um algoritmo para visitar todos os nós de uma árvore.

!!! answer

Será discutido na próxima aula.

Com base nesse algoritmo, faça a implementação no PrairieLearn do algoritmo para checar se uma árvore é válida.

Estudo extra

Os seguintes exercícios do leetcode são interessantes e relacionados ao assunto atual.

1. Soma de caminho até folha
2. Soma K
3. Altura mínima