Professor
Marcio Feitosa

CURSO: Estruturas de Dados

TAD ÁRVORE BINÁRIA

Árvores binárias são estruturas de nós interligados em forma hierárquica, onde cada nó pode ter no máximo 2 filhos, intitulados filho esquerdo e filho direito. O objeto representante do nó deve ter pelo menos 2 ponteiros, um para cada filho. Em algumas implementações utiliza-se um terceiro ponteiro para o nó pai, mas é menos frequente, pois não costuma ser necessário fazer navegação reversa.

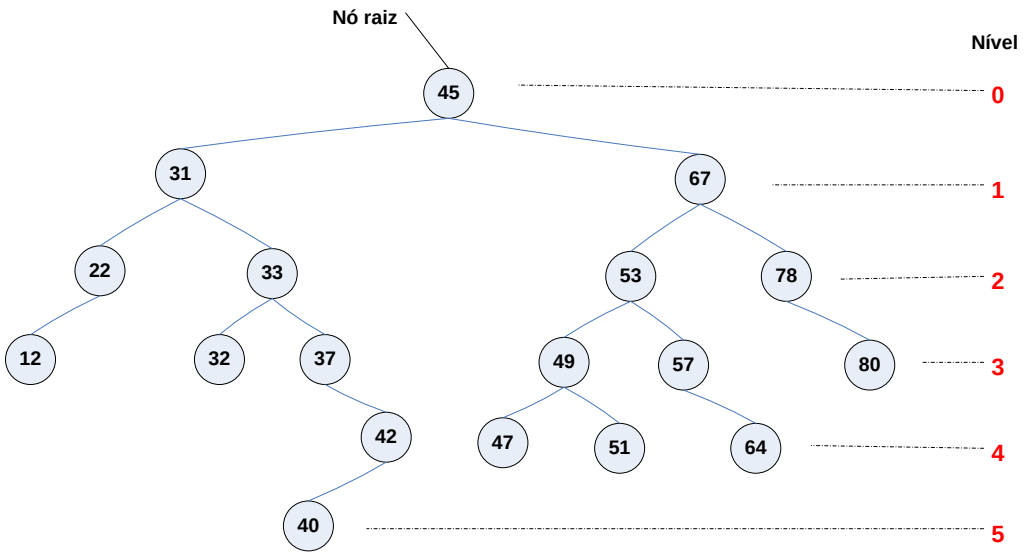


Figura 1

A árvore binária é bastante útil no armazenamento e recuperação de dados quando os dados seguem o seguinte critério de organização:

a) dados com valor menor que o dado de um determinado nó deverão estar em algum lugar da subárvore esquerda deste nó, e dados com valor maior, em algum lugar da subárvore direita;
b) não haja repetição de dados.

Seguindo este, a árvore é dita **"Árvore Binária de Busca"**.

Por exemplo, localizar o nó 53.

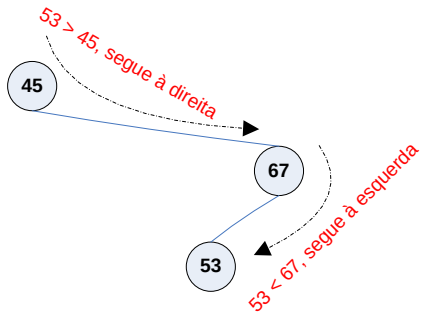


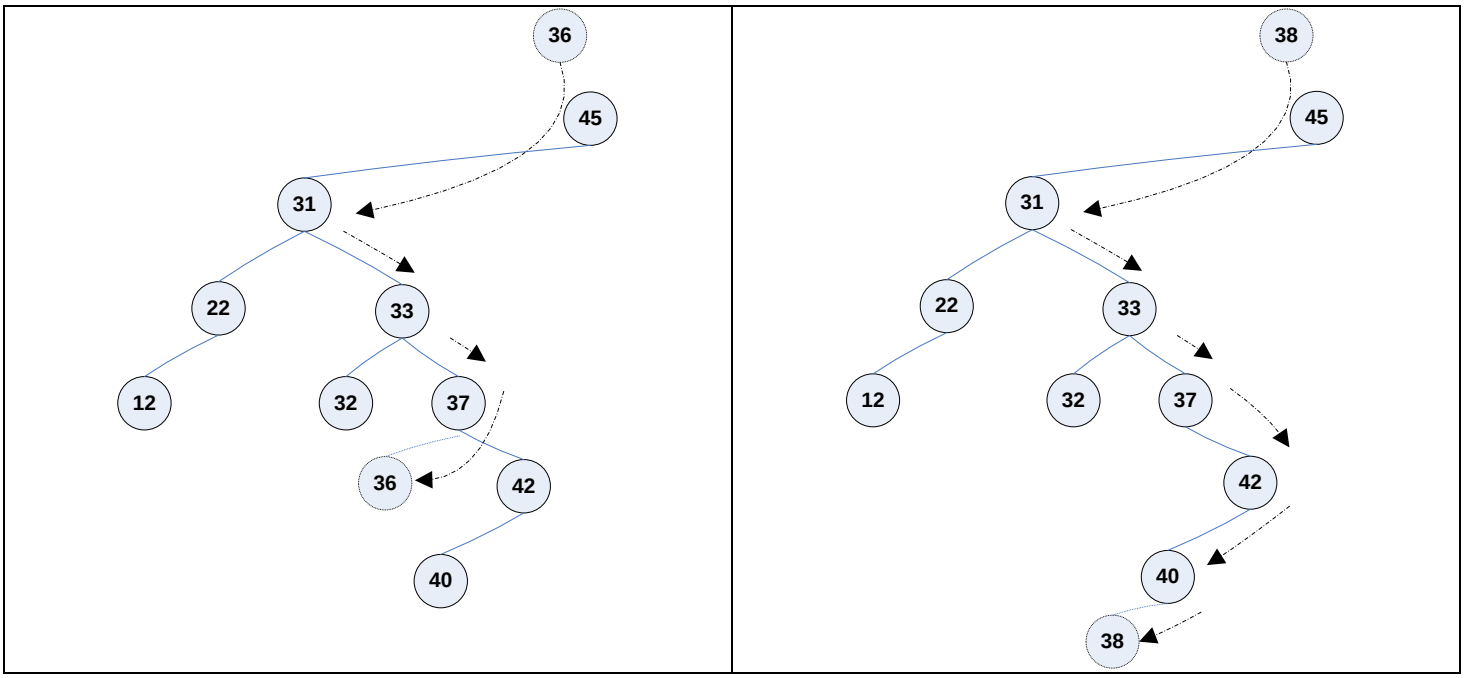
Figura 2

No exemplo da Figura 1 existem 18 elementos. Se esses elementos estivessem armazenados de forma sequencial como em um vetor ou lista encadeada, no pior caso seriam necessárias 18 verificações até achar o elemento procurado, ao passo que na árvore binária de busca seriam 6, ou seja, a quantidade de níveis da árvore (pior caso).

A vantagem da árvore aumenta exponencialmente à medida do aumento da quantidade de elementos. Com 32 níveis conseguimos armazenar próximo de 4.290.000.000 (4 bilhões) de elementos. Ou seja, no pior caso será necessário 32 passos para localizar um elemento, enquanto que em um vetor, ou lista, seriam mais de 4 bilhões.

Para inserir um elemento na árvore:

Para localizar a posição que o novo elemento deve ocupar, segue-se a regra do menor à esquerda e maior à direita. Exemplo: inserir os valores 36 e 38.



Professor
Marcio Feitosa

Para excluir um elemento da árvore:

A exclusão depende da quantidade de filhos do elemento a ser excluído.

Caso 1 - o elemento não tem filhos. Neste caso basta referenciar o ponteiro para o filho, no pai do elemento, para NULL e destruir o objeto[1] ou retorná-lo ao método chamador. Exemplo, eliminar o elemento 12.

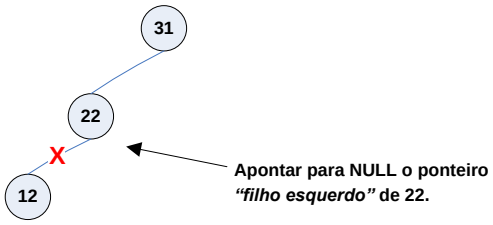


Figura 3

Caso 2 - o elemento tem 1 filho. Neste caso basta referenciar o ponteiro filho no nó pai do excluído para o filho do excluído. Ou seja, o filho do excluído vai para a posição deste. Por exemplo, eliminar o elemento 42.

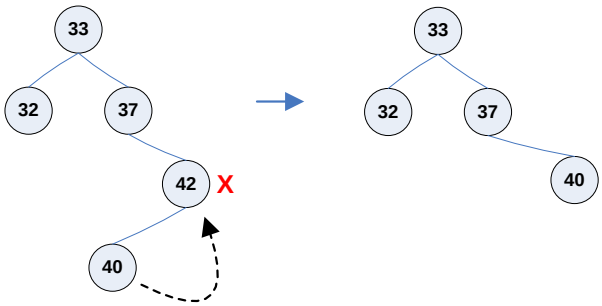


Figura 4

Caso 3 - o elemento tem 2 filhos. Este caso tem mais etapas. O roteiro deve ser o seguinte:

1. Localiza o elemento a ser excluído, E, e seu pai, pE.

Se for **filho esquerdo** de pE:

2. Localiza o nó de menor valor na subárvore direita de E. Este será o nó sucessor, S, e seu pai pS.
3. Aponta filho esquerdo de pE para S.
4. Aponta filho esquerdo de S para o filho esquerdo de E.
5. Caso $pS \neq E$
    a. Aponta o filho direito de S para filho direito de E
    b. Aponta filho esquerdo de pS para $NULL$.

[1] O método de destruição do objeto depende da linguagem de programação.

Se for **filho direito** de pE:

2. Localiza o nó de maior valor na subárvore esquerda de E. Este será o nó sucessor, S, e seu pai pS.
3. Aponta filho direito de pE para S.
4. Aponta filho direito de S para o filho direito de E.
5. Caso $pS \neq E$
    a. Aponta o filho esquerdo de S para filho esquerdo de E
    b. Aponta filho direito de pS para $NULL$.

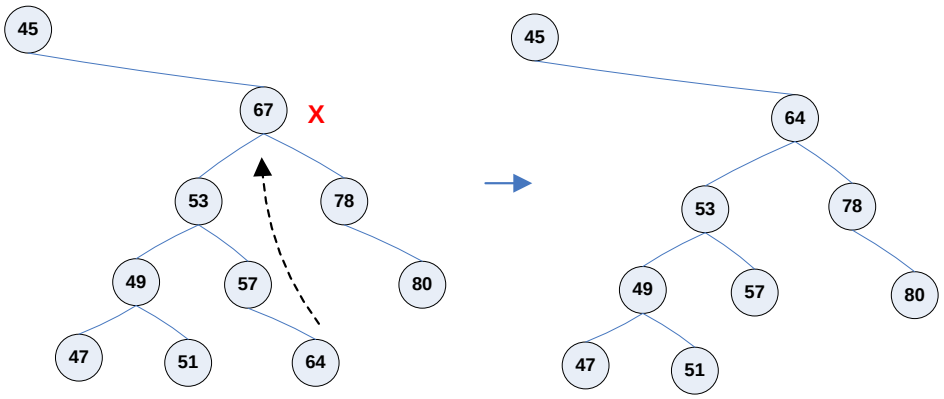


Figura 5

Na Figura 5 foi excluído o elemento 67. Como é filho direito, o maior elemento da sua subárvore esquerda será o seu sucessor (64).

Caso 4 - o nó excluído é o raiz.

- Se não tiver filhos, é caso único e a arvore passará a condição de vazia.
- Se tiver apenas 1 filho, este assumirá a posição de raiz.
- Se tiver os 2 filhos, o nó sucessor será (à escolher) o maior da subárvore esquerda ou o menor da subárvore direita.

Percurso em Pré-Ordem, em Ordem e em Pós-Ordem:

São métodos para percorrer todos os nós da árvore. São úteis quando se precisa atualizar algum dado em todos os nós e/ou se precisa percorrer em determinada ordem.

Esses métodos passam mais de uma vez pelo mesmo nó, pois se chegando a um nó folha, por exemplo, é necessário retroceder para percorrer outro ramo, mas a ação necessária sobre cada nó só é tomada uma única vez.

Pré-Ordem: neste método as raízes das subárvores são tratadas em primeiro lugar, em seguida o ramo à esquerda e depois à direita (a regra se mantém dentro de cada ramo, sub-ramo, etc.)

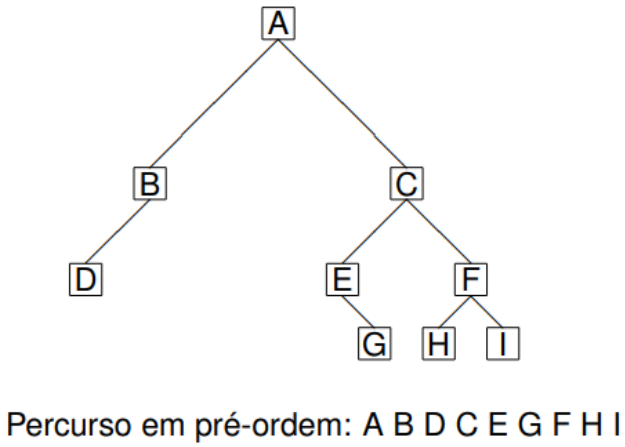


Figura 6 [2]

Em Ordem: Neste método é tratada primeiramente a subárvore esquerda, depois a raiz e finalmente a subárvore direita.

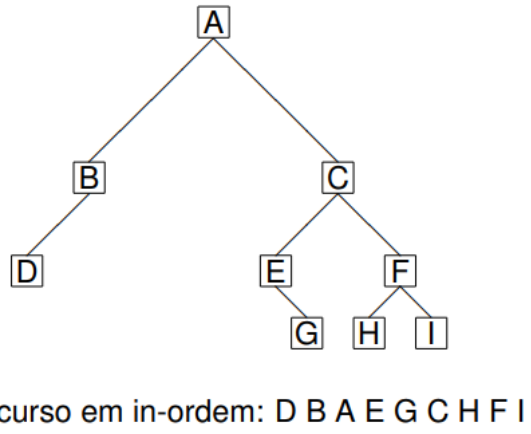


Figura 7

[2] As figuras ilustrativas dos percursos Pré-Ordem, Em Ordem e Pós-Ordem foram extraídas de: https://www.ime.usp.br/~song/mac5710/slides/05tree

Professor
Marcio Feitosa

Em Pós-Ordem: aqui primeiramente é tratada a subárvore esquerda, depois a subárvore direita e finalmente a raiz.

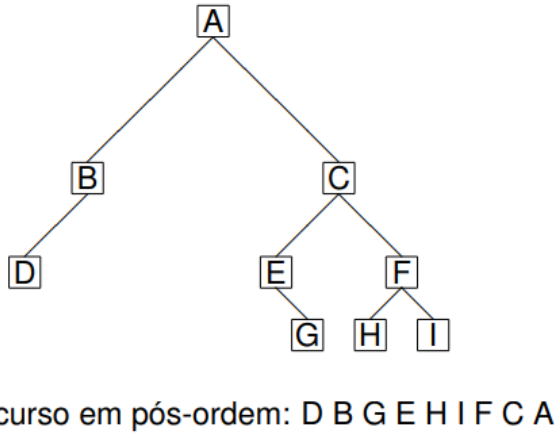


Figura 8

Pergunta: dadas 3 impressões de uma mesma árvore binária, Pré-Ordem, Ordem e Pós-Ordem, com qual delas seria mais fácil recuperar a árvore?

-o-o-o-o-o-o-o-o-o-o-o-o-