Árvores binárias

Alessandro

1 Árvore binária

Uma **árvore binária** é uma árvore ordenada com as seguintes propriedades:

1. Todos os nós têm no máximo dois filhos.
2. Cada nó filho é rotulado como sendo **filho da direita** ou um **filho da esquerda**.
3. O filho da esquerda precede o filho da direita na ordenação dos filhos de um nó.

A subárvore enraizada no filho da direita ou no filho da esquerda de um nó interno v é chamada de **subárvore direita** ou **subárvore esquerda** de v, respectivamente. Uma árvore binária é **própria** se cada nó tem zero ou dois filhos. Alguns autores também se referem a estas árvores, como árvores binárias **cheias**. Logo, em uma árvore binária própria todo nó interno tem exatamente dois filhos. Uma árvore binária que não é própria é dita **imprópria**.

Uma **árvore binária** é chamada de **árvore de pesquisa binária** quando:

1. Os **valores** em **qualquer subárvore esquerda** são **menores** que o valor em seu **nó-pai**.
2. Os **valores** em qualquer **subárvore direta** são **maiores** que o valor em seu **nó-pai**.

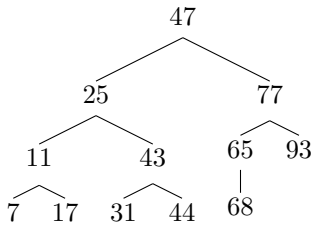


Figura 1: Árvore de pesquisa binária

Uma árvore binária pode ser definida como sendo:

1. Uma **árvore vazia**; ou
2. Um nó **raiz** tendo duas subárvores, identificadas como a **subárvore da direita** e a **subárvore da esquerda**.

Considerando que precisamos armazenar n nós em uma árvore binária, a sua altura ou número de níveis é $h_{max} = n$, onde n é a quantidade de elementos ou Nós do conjunto árvore.

A **altura máxima** de uma árvore será igual a n se e somente se a árvore for **degenerada** e tiver **filhos** em uma **única direção** (Figura 2).

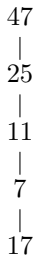


Figura 2: Exemplo de árvore degenerada. A altura de T é igual a n, onde n é o número de nós de T. $h_{max} = 5$

1.1 Profundidade

A **profundidade** de uma **árvore binária** significa o nível máximo de qualquer folha na árvore. Isso equivale ao tamanho do percurso mais distante da raiz até qualquer folha.

Se uma árvore binária tem m nós no nível l, então ela terá no máximo $2m$ nós no nível $l + 1$. Como uma árvore binária pode conter no **máximo um nó nível 0(raiz)**, ela poderá conter no máximo 2^1 nós no nível l.

Um árvore é estritamente binária quando todo nó que **não** é **folha** tiver uma **subárvore esquerda** e **direita não vazias**.

Uma árvore binária com n folhas contém sempre $2n - 1$ nós. Logo, a altura mínima é $h_{min} = (\log_2 n) + 1$.

1.2 Altura de uma Árvore binária

Uma **árvore binária** é dita **cheia**, ou **completa**, se todos os seus **nós internos** têm **duas subárvores associadas** e todos os nós **folhas** estão no **último nível**.

Nesse tipo de árvore, temos um nó no nível 0, dois nós no nível 1, quatro nós no nível 2, oito nós no nível 3, e assim sucessivamente.

A altura de uma árvore é uma medida importante na avaliação da eficiência com que visitamos os nós de uma árvore.

A altura indica o esforço computacional necessário para alcançar qualquer nó na árvore.

2 Percurso em Árvores Binárias

Muitas operações em árvores binárias envolvem o percurso de todas as subárvores, com a execução de alguma ação de tratamento em cada nó, de forma que é comum percorrer uma árvore em uma das seguintes ordens:

1. **pré-ordem**: trata **raiz**, percorre **esquerda** (*left*), percorre **direita** (*right*).
2. **ordem simétrica**: percorre **esquerda** (*left*), trata **raiz**, percorre **direita** (*right*).
3. **pós-ordem**: percorre **esquerda** (*left*), percorre **direita** (*right*), trata **raiz**.

Nas **árvores binárias de busca**, a ordem **importante** é a **ordem simétrica**.

Exemplo: Suponha que precisamos descobrir números repetidos em uma lista não-ordenada de números.

1. Uma possibilidade é comparar cada novo número com todos os números já lidos.

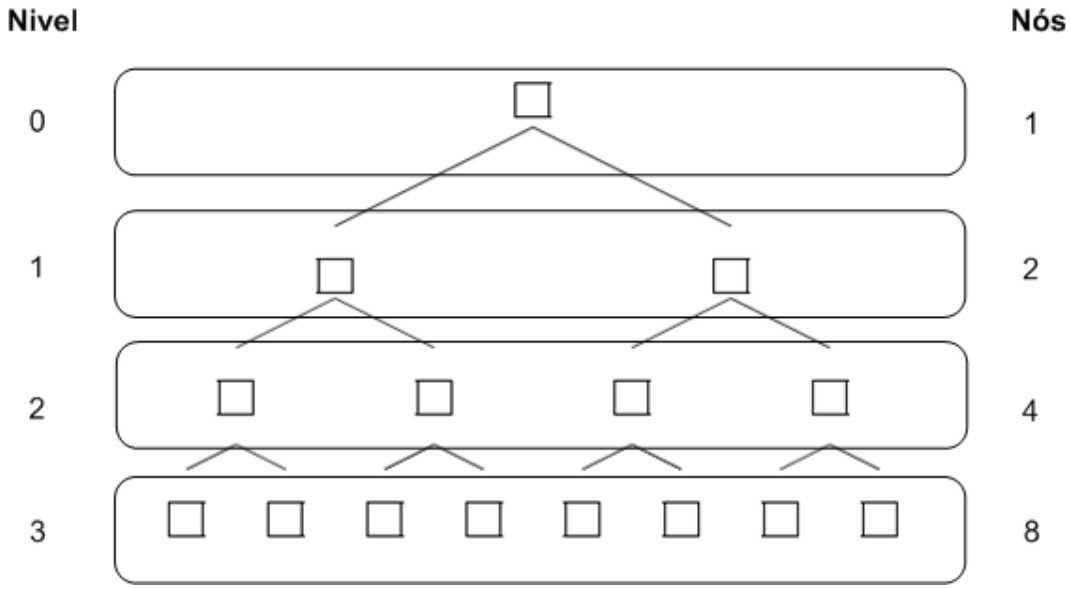


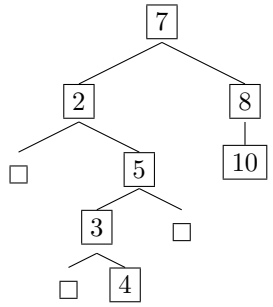
Figura 3: Número máximo de nós nos níveis de uma árvore binária

2. Manter uma lista ordenada dos números e a cada número lido fazer uma busca na lista.
3. Uma forma eficiente é usar uma **árvore binária** para manter os números. Dessa forma, o primeiro número é colocado na **raiz** da árvore. Cada novo número informado é comparado com o elemento raiz, caso seja **igual**, trata-se de uma **repetição**, partirmos para ler outro número. Se é **menor** repetimos o processo com a árvore da esquerda e se **maior** com a árvore da direita. Este processo continua até que uma repetição seja encontrada ou uma árvore vazia é achada. Neste caso, o número é inserido na posição devida na árvore.

Seja o conjunto v fornecido pelo usuário.

7 8 2 5 8 3 5 10 4

Inserindo cada valor em um árvore binária teremos a seguinte representação da árvore, temos:



A partir da árvore construída, podemos obter os valores da árvore em ordem crescente percorrendo os nós em ordem simétrica. O algoritmo 1 apresenta o pseudo-código pata obter os elementos de uma árvore binária T.

Algoritmo 1: simétrico(no)

```
1 nos ← {}
2 se no <> nulo então
3 |   nos ⋃ simetrico(left(no)) ⋃ no
4 |   nos ⋃ simetrico(right(no))
```

```
1 public List<No<E>> ascOrder(No<E> no) {
    List<No<E>> nodes = new ArrayList<No<E>>();
3    if (no != null){
       nodes.addAll(ascOrder(no.left()));
5      nodes.add(no);
       nodes.addAll(ascOrder(no.right()));
7    }
     return nodes;
9 }
```