



ESTRUTURAS DE DADOS E ALGORITMOS

Aula 13: Pesquisa e Operações em Árvores

Filipe Magno Gouveia Quintal
Faculdade de Ciências Exactas e Engenharias




2023/2024



1

PESQUISA

- Procurar informação é uma das nossas atividades quotidianas:
 - Números em listas telefónicas;
 - Palavras em dicionários;
 - Referências bibliográficas
 - ...googling
 - Já estudamos a pesquisa em vetores...
 - Como é que as BST podem melhorar a pesquisa?



Estruturas de Dados e Algoritmos


2

PESQUISA SEQUENCIAL EM VECTORES

- É uma das formas mais simples de procurar um elemento num vector

```
bool pesquisa(int a[], int size, int item ){
    for ( int index = 0 ; index < size ; index++ )
        if ( item == a[index] )
            return true;
    return false;
}
```

- **Eficiência**
 - Melhor caso $O(1)$
 - Pior Caso $O(n)$
 - Caso Médio $O(n)$




Estruturas de Dados e Algoritmos

3

PESQUISA SEQUENCIAL EM VECTORES

- A pesquisa num vector pode ser mais eficiente se este estiver ordenado
- **Pesquisa Binária**
 - É considerado em primeiro lugar o elemento que se encontra no **meio do vector**. Se a sua chave for menor do que a do elemento pretendido, repete-se o processo no **lado esquerdo**, senão, repete-se o processo no **lado direito**.
 - Termina quando encontrar o elemento pretendido ou quando já não for possível a divisão por 2.
 - Note-se que em cada passo a pesquisa binária reduz o número de elementos para metade (daí o seu nome).



Estruturas de Dados e Algoritmos

4

PESQUISA BINÁRIA

```
int search (int a[], int v, int l, int r) {
    while (r >= l) {
        int m = (l+r)/2 ;
        if (v == a[m])
            return m;
        if (v < a[m])
            r = m-1;
        else
            l = m+1 ;
    }
    return -1 ;
}
```



Estruturas de Dados e Algoritmos

5

PESQUISA BINÁRIA – EFICIÊNCIA

- Melhor Caso: $O(1)$
- Pior Caso: $O(\log n)$
- Caso Médio: $O(\log n)$

<https://blog.pentec.com/binary-vs-linear-search-animated-gifs/>

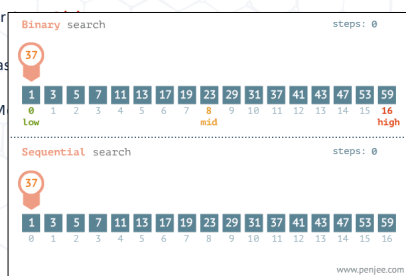


Estruturas de Dados e Algoritmos

6

PESQUISA BINÁRIA – EFICIÊNCIA

- Melhor
- Pior Cas
- Caso M



<https://blog.pentec.com/binary-vs-linear-search-animated-gifs/>



Estruturas de Dados e Algoritmos

7

PESQUISA BINÁRIA – PESQUISA / PROCURA

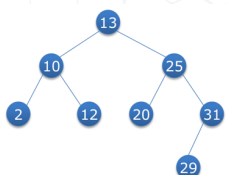
- Como já foi discutido...
- Uma árvore de pesquisa binária... é uma árvore binária ordenada pelo seguinte critério:
 - O valor de qualquer nó da subárvore esquerda é **menor ou igual** ao valor da raiz;
 - O valor de qualquer nó da subárvore direita é **maior** do que o valor da raiz;
 - As subárvores esquerda e direita são árvores de pesquisa.



Estruturas de Dados e Algoritmos

8

PESQUISA BINÁRIA – PESQUISA / PROCURA



Estruturas de Dados e Algoritmos

9

PESQUISA BINÁRIA –IMPLEMENTAÇÃO

Podemos definir um nó de uma BST como:

```
struct nodo {
    int item;
    nodo * esquerda;
    nodo * direita;
}
```

Valor do nó

Apontador para a sub-árvore esquerda

Apontador para a sub-árvore direita

Estruturas de Dados e Algoritmos

10

PESQUISA BINÁRIA – PESQUISA / PROCURA - IMPLEMENTAÇÃO

Inserção iterativa

```
nodo* inserir(nodo* raiz, int item ){
    nodo* aux = raiz;
    nodo* prev = null;
    if (raiz == NULL )
        aux = novoNodo( item );
    else{
        while (raiz != NULL ){
            prev = raiz ;
            raiz = (raiz->item > item ? raiz->esq : raiz->dir );
        }
        if ( prev->item < item )
            prev->dir = novoNodo( item );
        else
            prev->esq = novoNodo( item );
    }
    return aux;
}
```

11

PESQUISA BINÁRIA – PESQUISA / PROCURA - IMPLEMENTAÇÃO

Inserção recursiva

```
nodo* inserirNodo(struct nodo* nodo, int num){
    if (nodo == NULL) {
        return novoNodo(num);
    }else{
        if (num <= nodo->dados)
            nodo->esquerda = inserirNodo(nodo->esquerda, num);
        else
            nodo->direita = inserirNodo(nodo->direita, num);
    }
    return nodo;
}
```

Estruturas de Dados e Algoritmos

12

PESQUISA BINÁRIA – PESQUISA / PROCURA - IMPLEMENTAÇÃO**Pesquisa recursiva**

```
bool PesquisaBST(nodo* raiz, int item ){
    if ( raiz == NULL )
        return false;
    if ( item == raiz->item )
        return true;
    else if ( item < raiz->item )
        return PesquisaBST( raiz->esquerda, item );
    else
        return PesquisaBST( raiz->direita, item );
}
```



Estruturas de Dados e Algoritmos

13

ÁRVORE DE PESQUISA BINÁRIA – NOVO NODO

```
nodo * novoNodo(int num) {
    nodo* novo = new nodo;
    novo -> dados = num;
    novo -> esquerda = NULL;
    novo -> direita = NULL;
    return novo;
}
```



Estruturas de Dados e Algoritmos

14

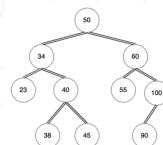
REMOÇÃO EM ÁRVORES DE PESQUISA BINÁRIAS

Estruturas de Dados e Algoritmos

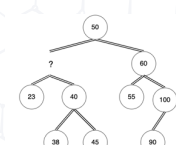
15

REMOÇÃO EM ÁRVORES DE PESQUISA BINÁRIAS

- Como??
- Tal como a inserção a remoção numa BST tem de respeitar as regras das árvores
- ... Valores menores do que a raiz à esquerda, valores maiores à direita



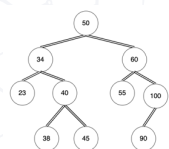
Remover o 34?



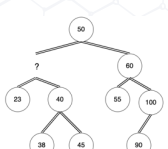
Estruturas de Dados e Algoritmos

16

REMOÇÃO EM ÁRVORES DE PESQUISA BINÁRIAS



Remover o 34?



- Qual o valor que a colocar em '?' 🤔
- O 40, 23, 38?
- O 45?

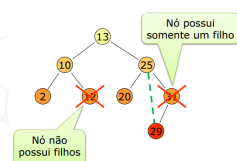


Estruturas de Dados e Algoritmos

17

REMOÇÃO EM ÁRVORES DE PESQUISA BINÁRIAS

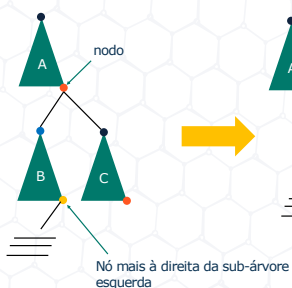
- O nó não tem filhos, ou seja, é uma folha
- O nó tem um único filho (subárvore)
- O nó tem dois filhos (subárvores)
- Dois métodos:
 - Remoção por **fusão**
 - Remoção por **cópia**



Estruturas de Dados e Algoritmos

18

REMOÇÃO – POR FUSÃO

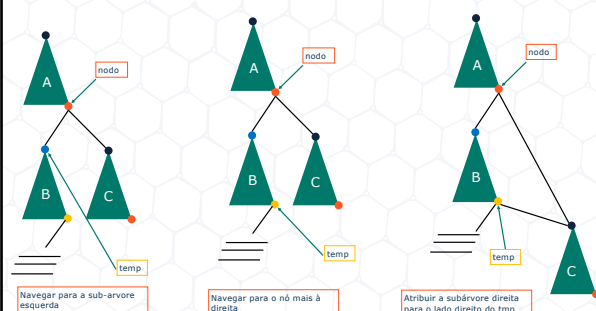


Estruturas de Dados e Algoritmos

19

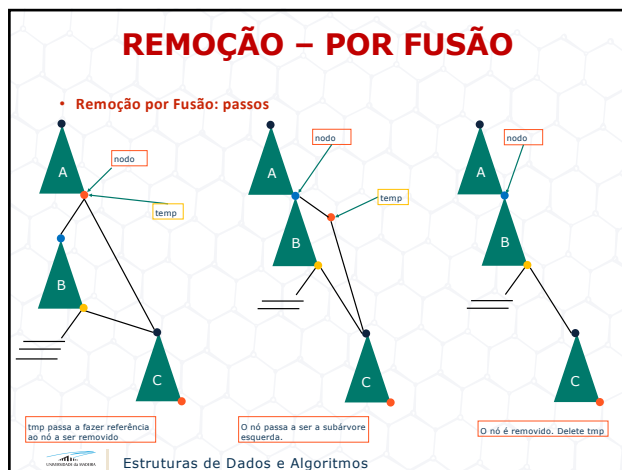
REMOÇÃO – POR FUSÃO

- Remoção por Fusão: passos

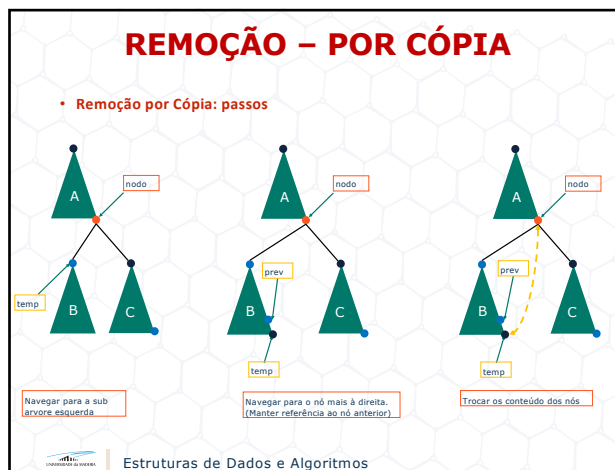


Estruturas de Dados e Algoritmos

20



21



22



23



24

