Teoria dos Grafos – Período 2003.2
Representação de Árvores

Jorge César Abrantes de Figueiredo
16/02/2004

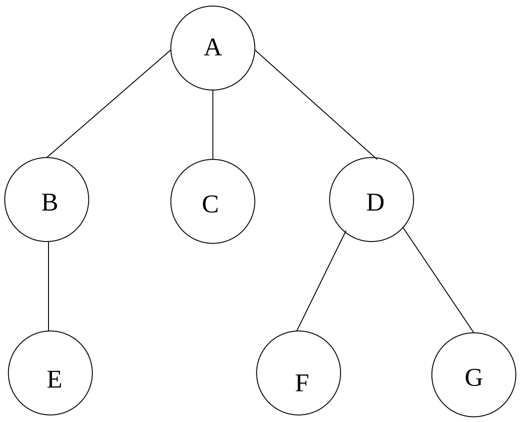
Árvores

Representação de Árvores

- Tipos de representação:
  - Representação por Adjacência
  - Representação Encadeada
  - Representação Alternativa

Representação por Adjacência

- Os nós são armazenados sequencialmente, de acordo com uma convenção pré-determinada
- Útil para armazenamento permanente (discos, fitas)
- Ineficiente para manipular árvores dinâmicas (inserções, remoções e localização de nós específicos)

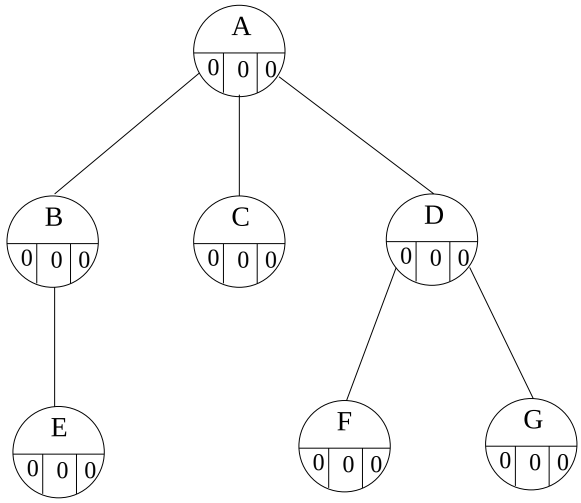


A	3	B	1	E	0	C	0	D	2	F	0	G	0

Representação por adjacência

Representação Encadeada de Árvores

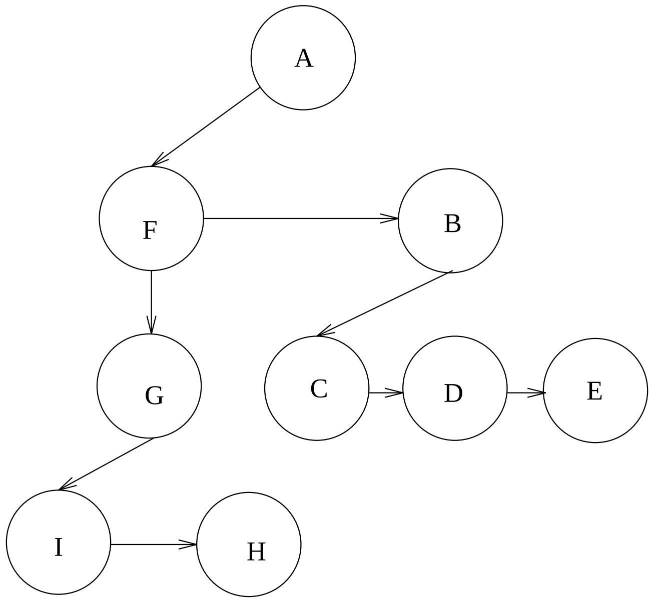
- Cada nó contém:
  - as informações do nó
  - referências as sub-árvores do nó (ponteiro para cada um de seus filhos)
- Desvantagem é o grande número de referências a sub-árvores nulas



Representação encadeada

Representação Alternativa

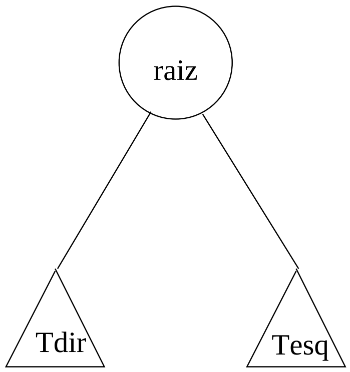
- Manter os filhos de um nó em uma lista encadeada de nós de árvores (primeiro filho/próximo irmão)
- Cada nó:
  - informação do nó
  - início da lista de filhos do nó



Representação alternativa

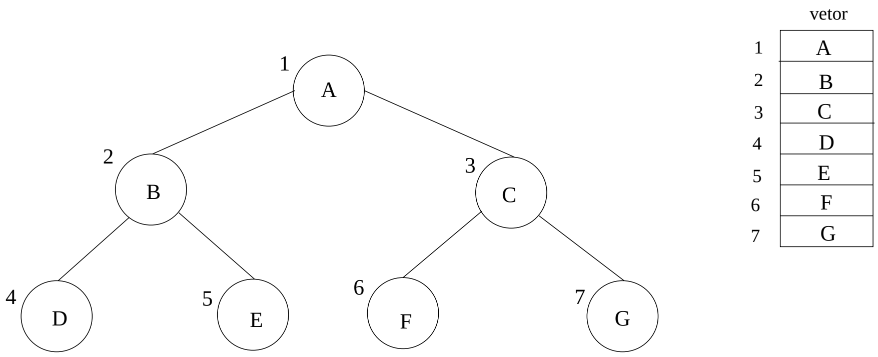
Árvores Binárias

- Uma árvore onde nenhum nó pode ter mais do que dois filhos (Grau ≤ 2)
- Uma raiz e duas sub-árvores (direita e esquerda) que podem ser vazias

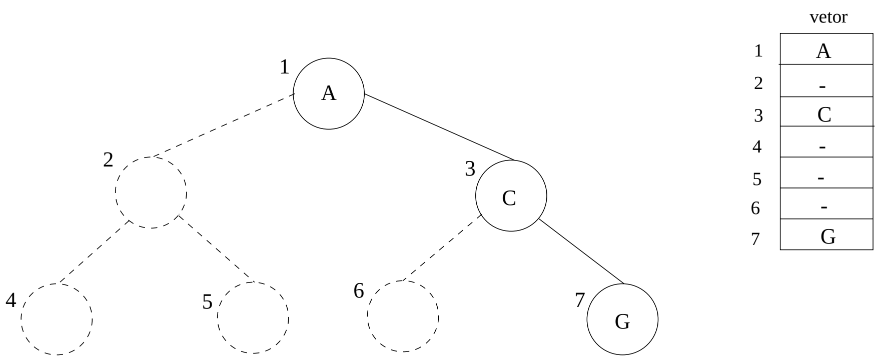


Árvore Binária

- O número máximo de nós do nível i é 2^i, $i \geq 0$
- O número máximo de nós é igual a $2^{h+1} - 1$, onde h é a altura da árvore
- Árvore binária completa: quando todo nó, com exceção das folhas, possui exatamente dois filhos
- Árvore binária incompleta: quando não é completa



Árvore binária completa



Árvore binária incompleta

Representação de Árvore Binárias

Representação Sequencial

- Os nós são numerados em uma certa ordem e armazenados em um vetor
- Considerando a árvore completa anterior:

	Vetor
1	A
2	B
3	C
4	D
5	E
6	F
7	G

Tabela 1: Representação sequencial - árvore completa

- Para recuperar:
  - $pai(i) = \lfloor i/2 \rfloor$, $i > 2$
  - $filho\_esq(i) = 2i$, $2i \leq n$
  - $filho\_dir(i) = 2i + 1$, $2i + 1 \leq n$
- Desvantagem na representação de árvores binárias incompletas com alto grau de incompletude

	Vetor
1	A
2	B
3	-
4	C
5	-
6	-
7	-

Tabela 2: Representação sequencial - árvore incompleta

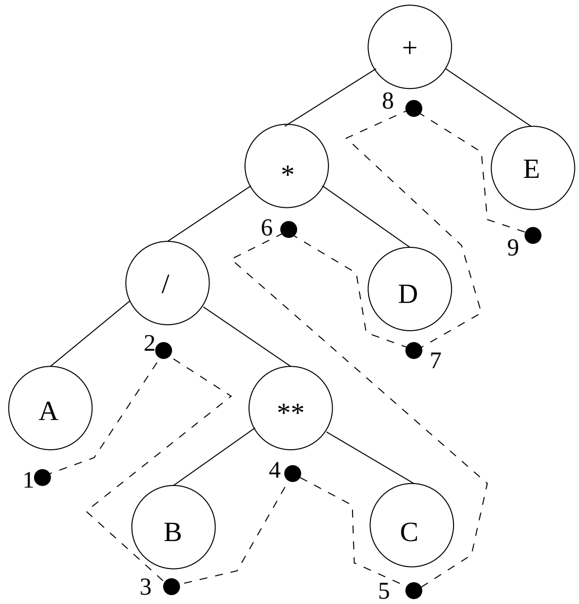
Representação Encadeada

- Utilizar ponteiros para os dois filhos
- A declaração dos nós das árvores é semelhante à estrutura de listas duplamente encadeadas
- Muitas funções aplicadas a listas encadeadas são também aplicadas em árvores (malloc, free)

Caminhamento em Árvores Binárias

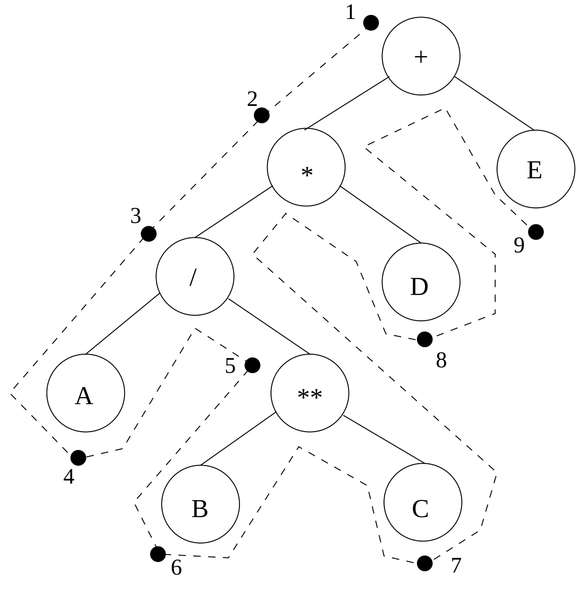
- A operação mais comum aplicada a uma árvore é o caminhamento sistemático em todos os seus nós
- Três formas de caminhamento:
  - Ordem Central: processa árvore à esquerda, depois o nó pai, depois a árvore à direita
  - Pré-Ordem: Processa o nó pai, depois a árvore à esquerda, depois a árvore à direita
  - Pós-Ordem: Processa a árvore à esquerda, depois a árvore à direita,depois o nó pai
- A utilização destes caminhamentos depende da aplicação

Caminhamento em Ordem Central



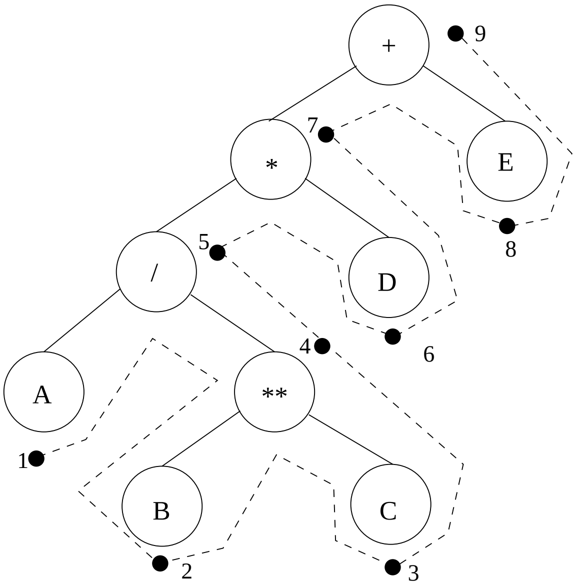
- Ordem Central: $A/B**C*D+E$ (notação infixa)

Caminhamento em Pré-Ordem



- Pré-Ordem: $+*/A**BCDE$ (notação pré-fixada)

Caminhamento em Pós-Ordem



- Pós-Ordem: $ABC**/D*E+$ (notação pós-fixada)
- Implementação de caminhamento em árvore pode ser:
  - recursiva
  - não recursiva

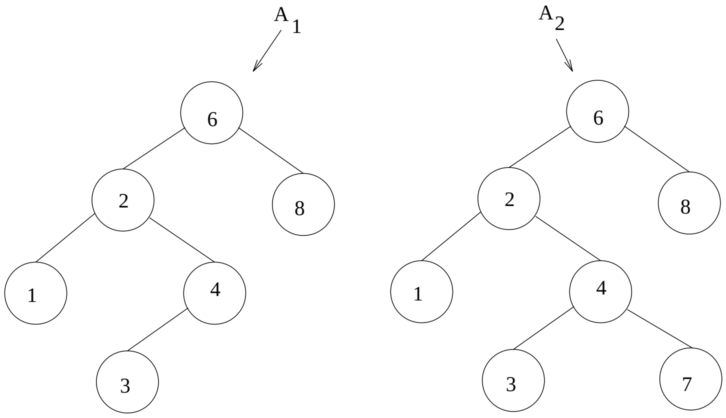
```
int ordem_central1(ARVORE *A)
/* rotina de caminhamento ordem central - versão recursiva */

{
    if(A != NULL){
        ordem_central1(A→esquerda);
        processa(A);
        ordem_entral1(A→direita);
        return Ok;
    }
    else
        return Erro;
};
```

Árvore Binária de Pesquisa

- Uma importante aplicação de árvores binárias é a pesquisa
- Cada nó da árvore tem associado uma chave de pesquisa

- Chaves de pesquisa são distintas
- Elementos da árvore são ordenados de alguma forma consistente, segundo as chaves de pesquisa
- Uma árvore binária de pesquisa é uma árvore binária onde, para todo nó (X) os valores de todas as chaves na sub-árvore esquerda são menores do que o valor da chave de X, e que os valores de todas as chaves na sub-árvore direita são maiores do que a chave de X
- Exemplo: A_1 é árvore binária de pesquisa. A_2 não é.

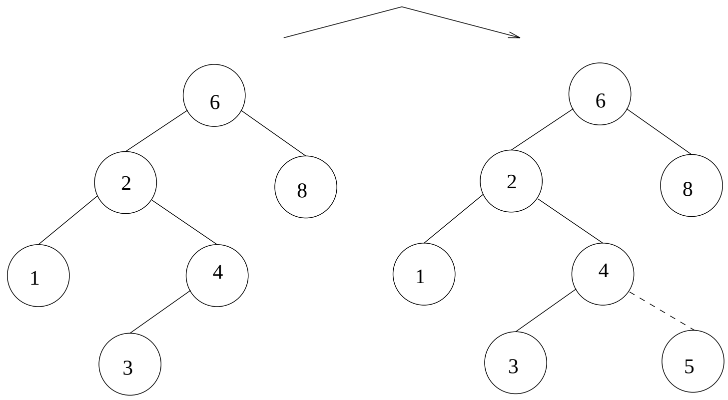


Exemplo de árvore binária de pesquisa

- Operações sobre árvores binárias de pesquisa:
  - **inserir** um nó na árvore
  - **remover** um nó da árvore
  - **localizar** um nó na árvore (pesquisa)

Inserção em Árvore Binária de Pesquisa

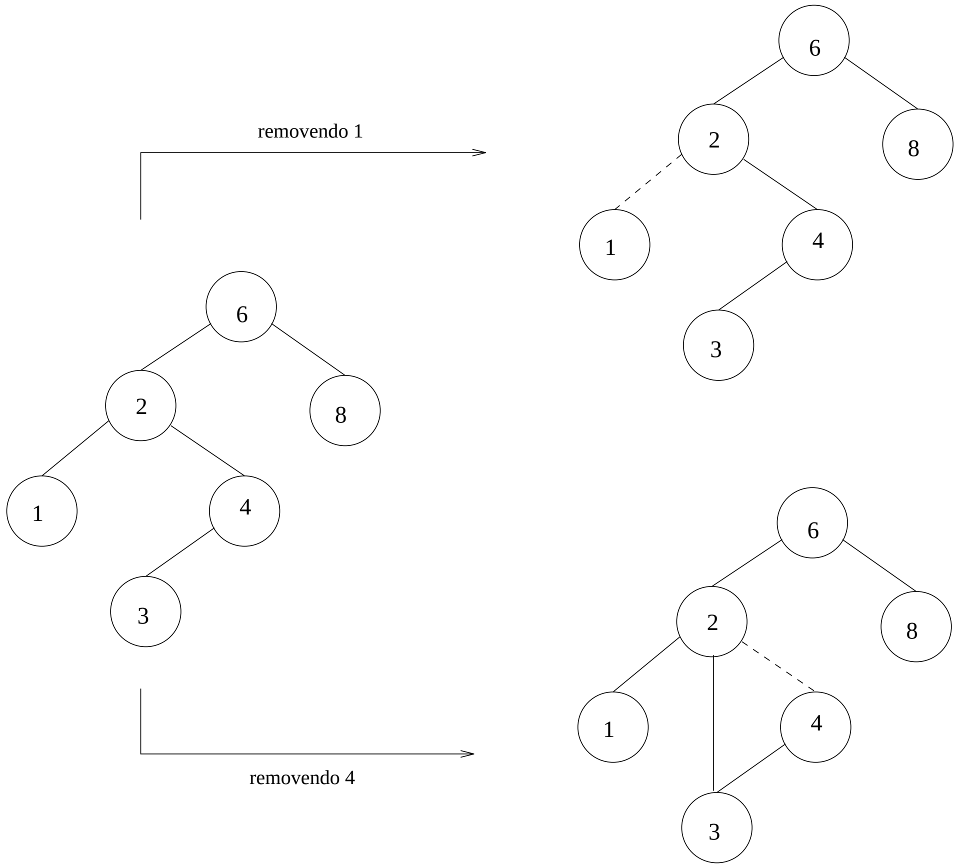
- Fazer como rotina localizar
- Se não achar chave, inserir no último nó
- Se chave duplicada, utilizar um campo extra para guardar número de ocorrências



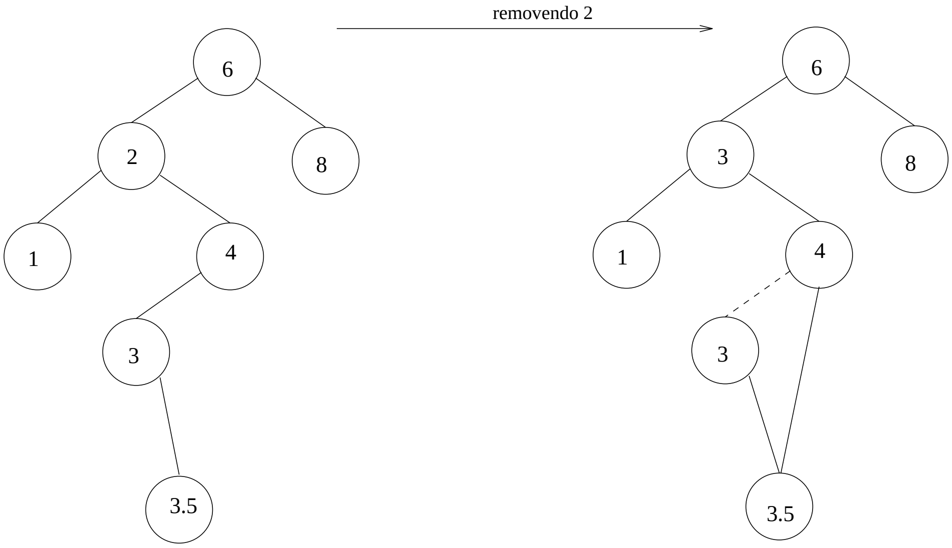
Inserção em árvore binária de pesquisa

Remoção em Árvore Binária de Pesquisa

- É a operação mais complexa
- Se nó é folha, remover imediatamente
- Se nó tem um filho, pode ser removido após seus pais ajustarem seus ponteiros excluindo o nó
- Quando o nó tem dois filhos, trocar a chave desse nó com a menor chave da sub-árvore direita e recursivamente remover aquele nó (que está vazio). Como o menor nó da sub-árvore direita não pode ter um filho à esquerda, a segunda remoção é simples



Remoção em árvore binária de pesquisa



Remoção de um nó com dois filhos

- Remoção e inserção pode desbalancear árvores binárias
- Uma árvore balanceada é uma árvore em que, para qualquer nó da árvore, a diferença máxima entre a altura de suas sub-árvores é 1
- Árvore AVL (Adelson-Velskii e Landis) é uma árvore binária de pesquisa com condição de balanceamento