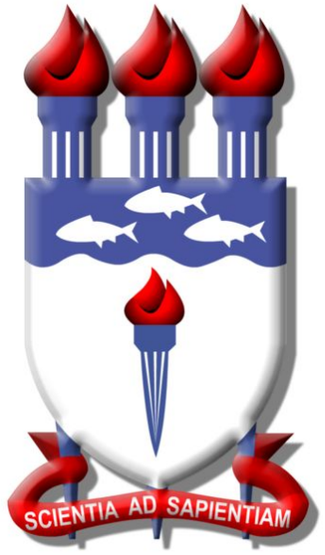


Árvore B

John Lucas, Lael Santa Rosa, Pamela Medeiros, Yuri Dimitri

https://github.com/laellsr/huffman

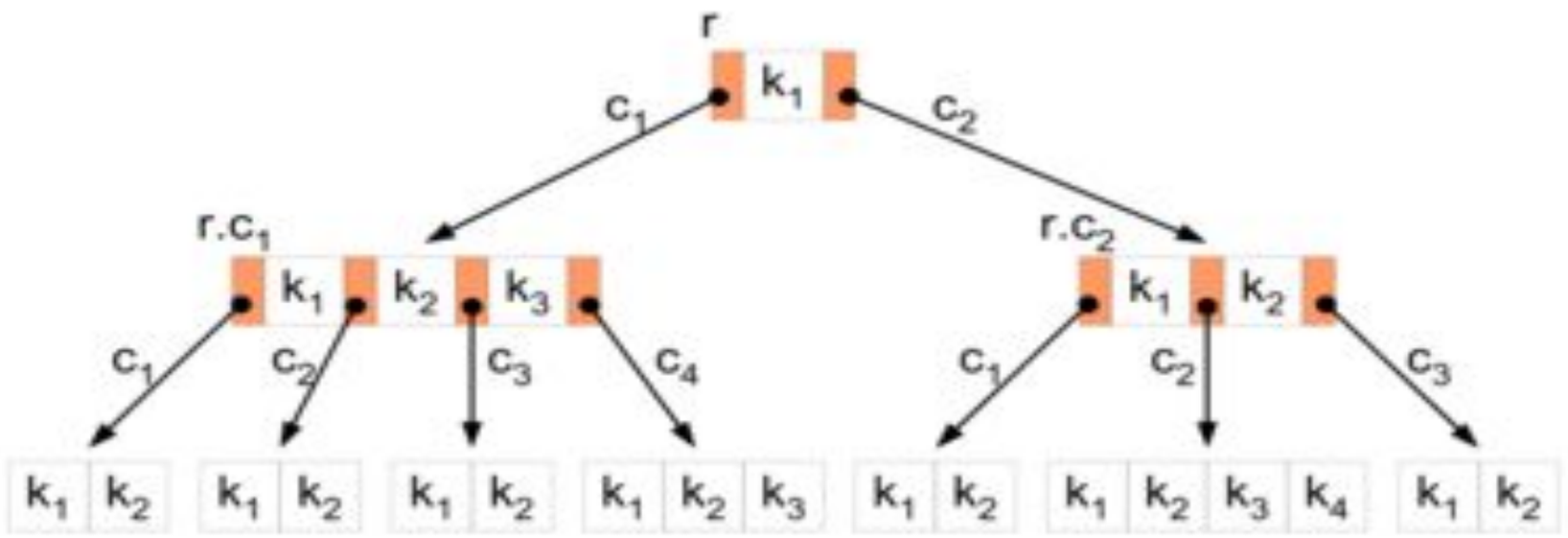
Motivação

- Necessidade de trabalhar com grande quantidade de dados mantendo buscas, inserções e remoções rápidas.

Árvore B

- A Árvore B é uma estrutura de dados em árvore auto-balanceada. Faz parte do grupo de árvores de pesquisa.

- Foi projetada para funcionar em memórias secundárias, guardando uma grande quantidade de dados, tendo entrada, saída e busca em tempo logarítmico.



Tempo de Execução

- Busca: O(log n)
- Inserção: O(log n)
- Remoção: O(log n)

Definições

- Nó/página
  - Sequência ordenada de chaves
    - Número de ponteiros para páginas-filho = número de chaves+1

- Ordem
  - É o número máximo de ponteiros que podem ser armazenados em uma página (Knuth , 1973)
  - Número mínimo de chaves que podem estar em uma página da árvore (Bayer&McGreight,1972)

- Nó folha
  - Nível mais baixo das chaves (Bayer&McGreight,1972)

Structs

```
#define DEGREE 8
typedef struct b_tree b_tree;
struct b_tree
{
    int keys[DEGREE-1];
    b_tree *childs[DEGREE];
    int num_keys;  //contador de chaves em uso na página
    short is_leaf;
};
```

OBS.: Note que estamos usando uma das possíveis definições de ordem, no caso a definição de Donald Knuth, 1972.

Busca em Árvore B

```
b_tree* b_tree_search(b_tree *bt, int element, int *index)
{
    if (bt == NULL) return NULL;

  int pos = binary_search(bt->keys, bt->num_keys, element);

  if(pos < bt->num_keys && bt->keys[pos] == element){
      *index = pos;
      return bt;
  } else if(bt->is_leaf == 1){
      return NULL;
  } else {
        return b_tree_search(bt->childs[pos],element,index);
  }
}
```

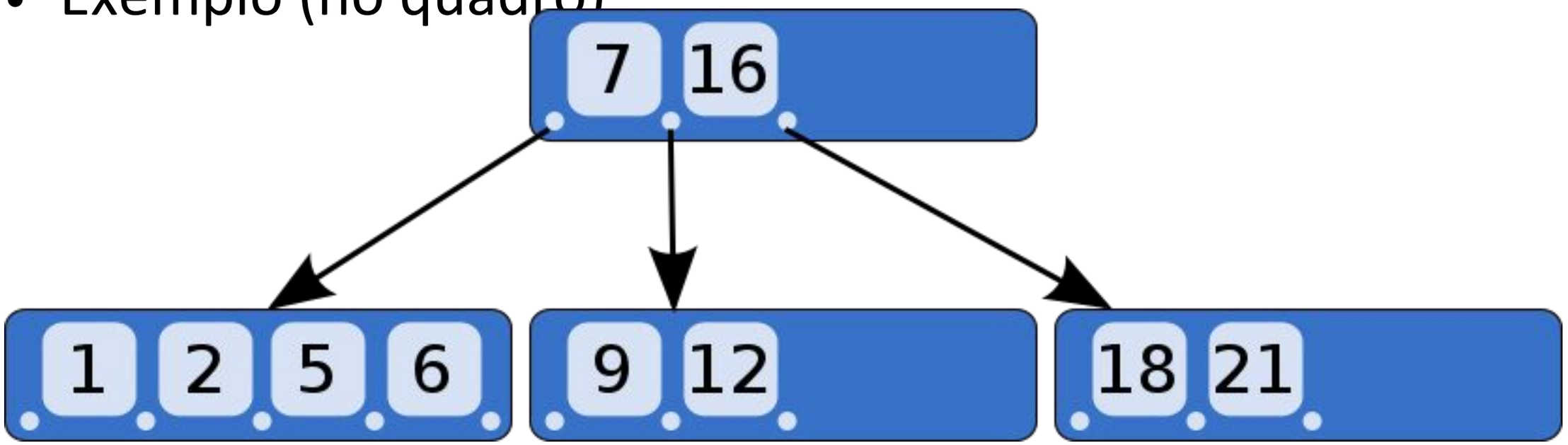
```
int binary_search(int *v, int size, int element)
{
    int begin = 0;
    int end = size - 1;
    int middle;

    while(begin <= end){

       middle = (begin + end) / 2;
       if(v[middle] < element){
          begin = middle + 1;
       } else if(v[middle] > element){
          end = middle - 1;
       } else {
          return middle;
       }
    }
    return begin;
}
```

Animação

- O seguinte site mostra o funcionamento completo de uma Árvore B: https://www.cs.usfca.edu/~galles/visualization/BTree.html
- Exemplo (no quadro)



De volta à Motivação…

- Como a ideia principal das **árvores B** é trabalhar com dispositivos de memória secundária, quanto menos acessos a disco a estrutura de dados proporcionar, melhor será o desempenho do sistema.