Algoritmos e Estruturas de Dados II

Prof. Debora Medeiros

Árvores B – Parte III

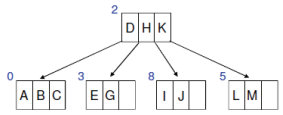
Algoritmos de Pesquisa e Inserção

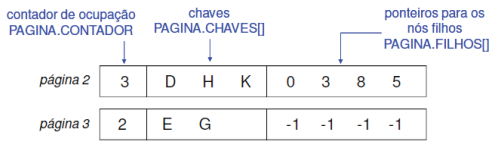
Adaptado dos Originais de:

Ricardo J. G. B. Campello
Thiago A. S. Pardo
Cristina D. A. Ciferri
Leandro C. Cintra
Maria Cristina F. de Oliveira

Revisão

- Arquivo da Árvore-B





Algoritmos

- Pesquisa
- função busca(RRN,
  chave,
  RRN_encontrado,
  pos_encontrada)
- ...

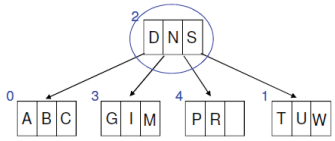
Algoritmo: Pesquisa

```
função busca(RRN,                  //página atual sendo pesquisada
             chave,                //chave sendo procurada
             RRN_encontrado,       //página que contém a chave
             pos_encontrada)       //posição da chave na página

se RRN=-1 então
    retorne FALSO                  //chave de busca não encontrada

senão
    leia página Pag identificada por RRN
    procure chave em Pag, fazendo POS igual a posição onde chave ocorre ou deveria ocorrer
    se chave encontrada então
        RRN_encontrado := RRN      //RRN atual contém a chave
        pos_encontrada := POS      //POS contém a posição da chave na página
        retorne VERDADEIRO
    //se chave não encontrada, recomeça-se  busca no filho apropriado
    senão retorne(busca(Pag.filhos[POS], chave, RRN_encontrado, pos_encontrada))
```

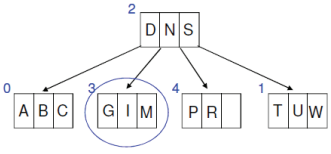

Exemplo



busca(2, K, RRN_encontrado, pos_encontrada)

Pag = D N S não existe → POS = 1

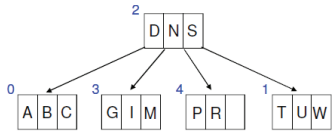
Exemplo



busca(Pag.filhos[1], K, RRN_encontrado, pos_encontrada)

Pag = G I M não existe → POS = 2

Exemplo

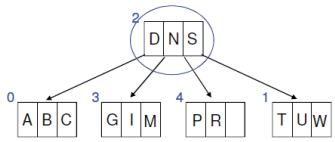


busca(Pag.filhos[2], K, RRN_encontrado, pos_encontrada)

Pag.filhos[2] = -1 → chave de busca não encontrada

retorna FALSO

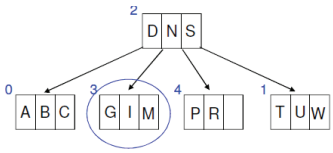
Exemplo



busca(2, M, RRN_encontrado, pos_encontrada)

Pag = D N S não existe → POS = 1

Exemplo



busca(Pag.filhos[1], M, RRN_encontrado, pos_encontrada)

Pag = G I M

chave de busca encontrada
pos_encontrada = 2
RRN_encontrado = 3
retorna VERDADEIRO

Algoritmos

- Inserção
  - ...

Algoritmo: Inserção

- Dois momentos
  - inicia-se com uma pesquisa que desce até o nível dos nós folhas
  - uma vez escolhido o nó folha no qual a nova chave deve ser inserida, os processos de **inserção**, **particionamento** (*split*) e **promoção** (*promotion*) propagam-se em direção à raiz
    - construção *bottom-up*

Algoritmo: Inserção

- Fases (procedimento recursivo)
  1. busca na página
     - pesquisa da página antes da chamada recursiva
  2. chamada recursiva
     - move a operação para os níveis inferiores da árvore
  3. inserção, *split* e *promotion*
     - executados após as chamadas recursivas
     - a propagação destes processos ocorre nos retornos das chamada recursivas

caminho inverso ao da pesquisa

Algoritmo: Inserção

- Parâmetros
  - Entrada...
  - Saída...

Algoritmo: Inserção

- Parâmetros
  - **RRN_atual**
    - RRN da página da árvore-B que está atualmente em uso (inicialmente, a raiz)
  - **chave**
    - a chave a ser inserida
  - **chave_promovida**
    - retorna a chave promovida, caso a inserção resulte no particionamento e na promoção da chave
  - **filho_dir_chave_promovida**
    - retorna o ponteiro para o filho direito de chave_promovida
      - quando ocorre um particionamento, não somente a chave promovida deve ser inserida em um nó de nível mais alto da árvore, mas também deve ser inserido o RRN da nova página criada no particionamento

Algoritmo: Inserção

- Valores de retorno
  - **há_promoção**
    - quando uma inserção é feita e uma chave é promovida condição de nó cheio (i.e., overflow)
  - **não_há_promoção**
    - quando uma inserção é feita e nenhuma chave é promovida nó com espaço livre
  - **erro**
    - quando uma chave sendo inserida já existe na árvore-B

Algoritmo: Inserção

- Variáveis locais/internas
  - Pagina
    - página atualmente examinada pela função
  - Pagina_nova
    - página nova resultante do particionamento
  - pos
    - posição em Pagina em que a chave já ocorre ou deveria ocorrer
  - chave_promovida_inferior
    - chave promovida do nível inferior para ser inserida em Pagina
  - filho_dir_chave_promovida_inferior
    - RRN promovido do nível inferior para ser inserido em Pagina
    - filho à direita chave_promovida_inferior

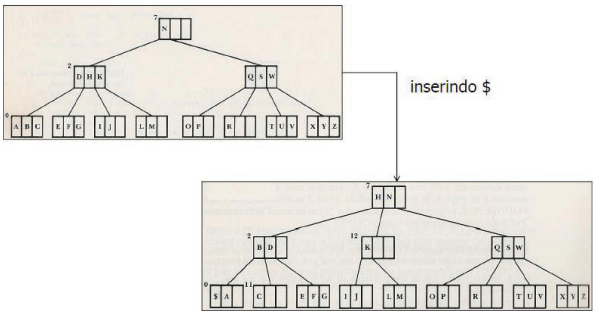
Algoritmo: Inserção

FUNÇÃO-INSERÇÃO(*RRN_atual*, *chave*, *chave_promovida*, *filho_dir_chave_promovida*)
se *RRN_atual* = -1
  chave_promovida ← *chave*;
  filho_dir_chave_promovida ← -1
  retorne HÁ_PROMOÇÃO
senão
  Pagina ← *página* [*RRN_atual*]
  procure por *chave* em *Pagina*
  pos ← posição em que *chave* ocorre ou deveria ocorrer em *Pagina*
se chave encontrada
  erro ← CHAVE JÁ EXISTE
  retorne *erro*
senão
...

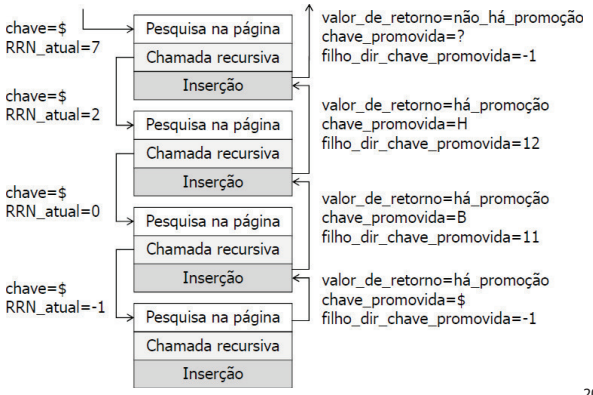
Algoritmo: Inserção

valor_de_retorno ← FUNÇÃO-INSERÇÃO(*Pagina.filhos*[*pos*], *chave*, *chave_promovida_inferior*, *filho_dir_chave_promovida_inferior*)
se *valor_de_retorno* = NÃO_HÁ_PROMOÇÃO **ou** *valor_de_retorno* = *erro*
  retorne valor_de_retorno
senão se há espaço em *Pagina* para *chave_promovida_inferior*
  inserir *chave_promovida_inferior* e *filho_dir_chave_promovida_inferior* em *Pagina*
  retorne NÃO_HÁ_PROMOÇÃO
senão
  SPLIT(*chave_promovida_inferior*, *filho_dir_chave_promovida_inferior*, *Pagina*, *chave_promovida*, *filho_dir_chave_promovida*, *Pagina_nova*);
  escreva Pagina no arquivo na posição RRN_atual
  escreva *Pagina_nova* no arquivo na posição *filho_dir_chave_promovida*
  retorne HÁ_PROMOÇÃO
fim-função

Exemplo: Inserção do $



Exemplo: Inserção do $



Bibliografia

- **M. J. Folk and B. Zoellick, *File Structures: A Conceptual Toolkit*, Addison Wesley, 1987.**