Algoritmos e Estruturas de Dados II

Prof. Ricardo J. G. B. Campello

Árvores B – Parte IV

Variantes

Adaptado e Estendido dos Originais de:

Leandro C. Cintra

Maria Cristina F. de Oliveira

Organização

- Inserção Revisitada
  - Revisão de Inserção em Árvores B
  - Inserção com Redistribuição
- Árvores B*
- Árvores B Virtuais
- Árvores B+ (próxima aula...)

2

Revisão de Inserção

1. Se árvore está vazia, crie a 1ª página (raiz), insira a chave e **FIM**

2. Senão, localize a página folha que deveria conter a chave

3. Se existe espaço, insira a chave, reordene a página e **FIM**

4. Senão (overflow):

4.1. Divida a página em duas (split) e redistribua as chaves entre elas

4.2. Se a pág. dividida era raiz, crie nova raiz como pai das duas resultantes

4.3. Promova a menor chave da pág. direita como separadora no nó pai

4.4. Se nó pai não sofreu overflow, **FIM**.

4.5. Senão, volte ao passo 4.1 para o nó pai.

3

Exemplo de Inserções

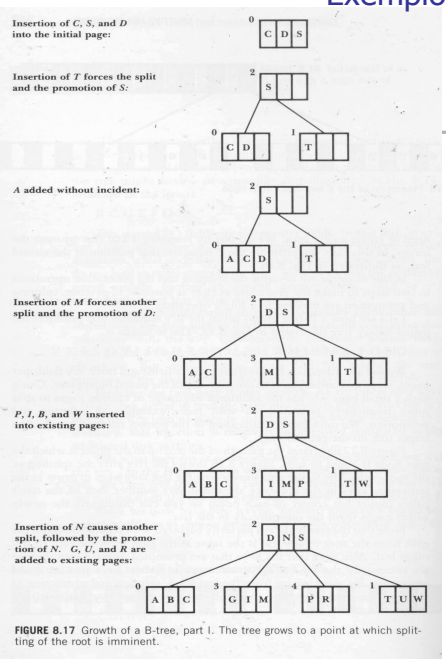


FIGURE 8.17 Growth of a B-tree, part I. The tree grows to a point at which splitting of the root is imminent.

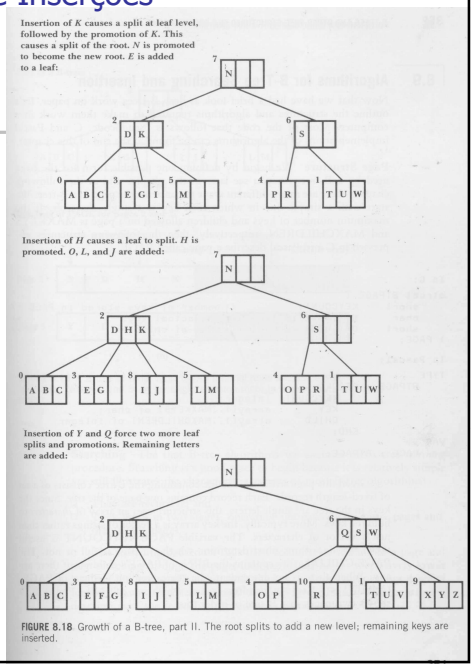


FIGURE 8.18 Growth of a B-tree, part II. The root splits to add a new level; remaining keys are inserted.

Revisão de Inserção

- Note que, sempre que ocorre um overflow (passo 4), o algoritmo substitui a página com problemas por duas novas páginas com metade de sua capacidade ocupada
- Logo, no **pior caso** tem-se que:
  - a utilização de espaço em uma Árvore B é em torno de **50%** !
- **Em média**, para árvores com **N** (no. de chaves) e **m** (ordem) grandes, tem-se que:
  - utilização de espaço se aproxima de um valor teórico ≈ **69%**
- Seria possível melhorar esse desempenho evitando ao máximo a criação de novas páginas ?

5

Inserção com Redistribuição

- Pode-se evitar ou ao menos adiar a criação de uma nova página utilizando uma operação já presente em Árvores B:
  - **Redistribuição** (para correção de underflow devido a eliminação)
- Redistribuição é uma idéia até então não explorada no algoritmo de inserção básico visto anteriormente
  - Ao invés de dividir uma página folha cheia em 2 páginas folha novas semi-vazias, pode-se optar por colocar a chave excedente (ou mais de uma!) em uma página irmã direta com espaço
  - Essa estratégia resulta em uma melhor utilização do espaço já alocado para a árvore

6

Inserção com Redistribuição

- Exemplo:
  - No quadro...

7

Inserção com Redistribuição

- Diferentemente da divisão e da concatenação, o efeito da redistribuição é **local**
  - Não existe propagação !
- Estudos empíricos indicam que a utilização de redistribuição durante inserção pode elevar a taxa de ocupação média de espaço para ≈ 85%
  - Resultados sugerem que qualquer aplicação séria de árvores B deve utilizar redistribuição durante inserção

8

Árvores B*

- Proposta por Knuth em 1973, trata-se de uma variação de Árvore B na qual cada nó tem, no mínimo, 2/3 do número máximo de chaves
  - Idéia é reduzir a subutilização de espaço das árvores

9

Árvores B*

- A construção destas árvores utiliza uma variação do processo de divisão (split)
  - assume que inserção com redistribuição é utilizada
    - logo, split é adiado até que páginas irmãs estejam cheias
  - realiza-se, então, a divisão do conteúdo de duas páginas cheias em 3 páginas (*two-to-three split*)

10

Árvores B* (split)

1. A pág. cheia da esquerda fornece 1/3 de suas maiores chaves para a nova página

2. A pág. cheia da direita fornece 1/3 de suas menores chaves para a nova página

3. A nova chave é inserida na página apropriada

3.1. Se na página esquerda, maior chave desta é deslocada para a nova

3.2. Se na página direita, menor chave desta é deslocada para a nova

4. Chave separadora do pai é rebaixada para a nova página

5. Nova página é reordenada

6. Menor e maior chaves da nova página são promovidas

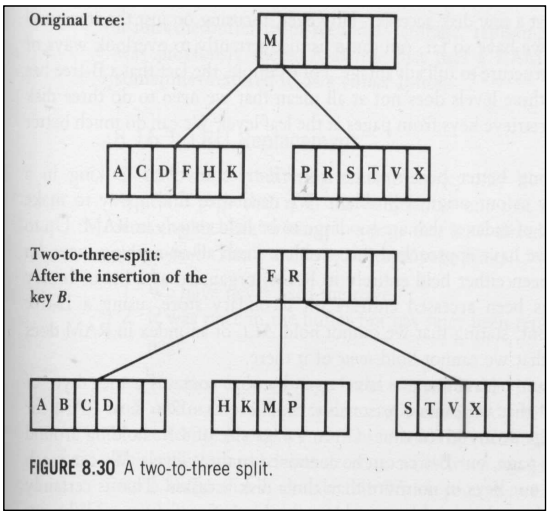


FIGURE 8.30 A two-to-three split.

Árvores B* (split)

- Deve-se tomar cuidado na implementação, uma vez que a raiz nunca tem página irmã
  - por definição não existe split 2-to-3 sem página irmã
  - logo, raiz requer tratamento especial
- Knuth propôs resolver o problema permitindo a raiz crescer além do tamanho normal das páginas
  - crescer até poder ser dividida em duas páginas aproximadamente 2/3 cheias, mais uma nova raiz
    - L/E da raiz pode demandar 2 acessos

12

Propriedades de Árvores B*

- Propriedades que diferem das B-Trees tradicionais:
  - Páginas têm no mínimo $\lfloor (2m-1)/3 \rfloor + 1$ descendentes
    - exceto a raiz (mín. 2) e as folhas (nenhum descendente)
    - exemplo: **m** = 8 $\Rightarrow$ no. mín. descendentes = 5
  - Uma página folha contém no mínimo $\lfloor (2m-1)/3 \rfloor$ e no máximo **m – 1** chaves
    - Exemplo: **m** = 8 $\Rightarrow$ no. chaves $\in$ [5,7]
- Claramente, essas propriedades demandam ajustes nos procedimentos de eliminação/redistribuição/concatenação

13

Árvores B Virtuais

- Árvores B são muito eficientes
  - mas podem ficar ainda melhores !
- Observe, por exemplo, que o fato da árvore ter profundidade 3 não implica em necessariamente fazer 3 acessos para recuperar uma página folha
  - O fato do índice não caber todo em RAM não significa que não se possa manter pelo menos parte dele...

14

Árvores B Virtuais

- Se raiz é sempre acessada, porque não mantê-la em RAM ?
- **Exemplo**
  - Considere uma árvore B com 3 níveis
    - Podemos atingir qualquer página com, no máximo, 3 acessos
  - Mantendo a raiz todo o tempo em memória primária, reduzimos o número máximo de acessos de 3 para 2 acessos
- Mas apenas com a raiz, ainda pode sobrar muita RAM !
  - Porque não utilizar essa memória também ?

15

Árvores B Virtuais

- Pode-se generalizar a idéia anterior:
  - ocupar toda a memória disponível com páginas
  - quando se precisa da página, ela pode já estar em RAM
- Se página demandada não estiver em RAM:
  - carregada-se em RAM, substituindo outra
- Tem-se assim uma espécie de cache de disco:
  - Buffer paginado em RAM denominado **Árvore B virtual**

16

Árvores B Virtuais

- **Questão**:
  - Quais páginas devem ser mantidas ou substituídas em RAM ?
    - Em outras palavras, qual a Política de Gerenciamento de Substituição ?
  - Política deve buscar maximizar eficiência do buffer mantendo em RAM páginas mais prováveis de serem requisitadas
- Política **LRU** (Last Recently Used):
  - substitui-se a página acessada menos recentemente
  - assume que acessos seguem um padrão de freqüência
    - padrão de probabilidades de acesso associadas às chaves

17

Árvores B Virtuais

- Política da **Altura**:
  - Colocar todos os níveis mais altos da árvore em RAM
  - Por exemplo, suponha hipoteticamente que:
    - temos um índice de 1Mb em uma árvore B com ~1.2Mb
      - árvore B com ~83% de utilização de espaço
    - cada página ocupa 4Kb $\Rightarrow$ árvore possui ~300 páginas
      - árvore B com raiz + 10 descendentes com ~30 descendentes cada
    - temos 256Kb de RAM disponível $\Rightarrow$ capacidade para 64 páginas
  - No exemplo acima, podemos colocar os dois primeiros níveis da árvore (11 páginas = 44Kb) em RAM e ainda sobra muita memória

18

Árvores B Virtuais

- Política da **Altura + LRU**:
  - No exemplo anterior, podemos utilizar a memória RAM que sobrou para implementar uma política LRU com páginas do 3º nível !
  - Teremos todas as páginas do 1º e 2º níveis da árvore permanentemente em RAM, além de 64 – 11 = 53 páginas do 3º e último nível em esquema rotativo
  - Esse esquema pode reduzir o no. esperado (médio) de acessos para menos de 1 !!!

19

Árvores B Virtuais

- Em Resumo:
  - Bufferização deve ser adotada em qualquer situação real de utilização de árvores B !

20

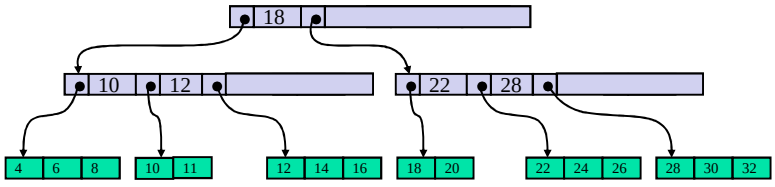
Alocação da Informação

- E a informação associada às chaves, onde fica?
  - informação = demais campos dos registros
  - normalmente fica em um arquivo separado (arquivo principal)
- Se a informação for mantida junto com a chave, elimina-se um acesso ao disco (ao arquivo principal)
  - árvore constitui o próprio arquivo principal
- Porém, perde-se no número de chaves que podem ser alocadas em uma página
  - reduz a ordem da árvore, aumentando a sua altura
  - compensa apenas se demais informações forem muito compactas
- Decisão deve ser baseada nas alturas máximas das árvores calculadas para ambos os casos

21

Árvores B+

- Árvores B+ também permitem acesso seqüencial ordenado ao arquivo principal
- Idéia é esboçada abaixo:



- Mais Detalhes: próxima aula...

22

Exercícios

- Capítulo 9 (Folk & Zoellick, 1987)
- Lista de Exercícios (CoTeia)
  - **Nota.** A lista faz referências à 2ª edição do livro de Folk & Zoellic.
    - FOLK, M. & ZOELLICK, B., *File Structures*, 2nd Edition, Addison-Wesley, 1992.

23

Bibliografia

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

24