Algoritmos e Estruturas de Dados II

Prof. Ricardo J. G. B. Campello

Indexação de Arquivos I:

Índices Simples

Adaptado e Estendido dos Originais de:

Leandro C. Cintra

Maria Cristina F. de Oliveira

Índice

- Mecanismo para localizar informações via chave
  - mapeamento chave $\rightarrow$ localização da informação
  - por exemplo, índice de um livro
- No caso de arquivos:
  - permite localizar registros rapidamente
  - evita ter que reorganizar o arquivo de dados conforme este for modificado
    - faça uma analogia com um texto...

Arquivo de Índice

- Um **Arquivo de Índice**:
  - Impõe ordem a um arquivo de dados sem precisar rearranjar o arquivo em si
  - Permite acesso a registros via chave sem precisar varrer o arquivo de dados
  - Permite várias visões diferentes de um mesmo arquivo de dados
    - acesso por múltiplas chaves

3

Arquivo de Índice

- Estudaremos inicialmente arquivos com **Índices Simples**:
  - Estrutura de dados linear
    - lista de pares (chave , localização)
- Posteriormente no curso veremos índices com EDs mais sofisticadas
  - Por exemplo, árvores

4

Arquivo de Índice

- Exemplo Prático (Arquivo de Músicas)
  - Registros de tamanho variável com:
    - **ID Number:** Número de identificação
    - **Title**: Título
    - **Composer**: Compositor(es)
    - **Artist**: Artista(s)
    - **Label**: Rótulo (código da gravadora)
  - Chave primária:
    - Combinação de **Label** e **ID Number**

5

Arquivo de Índice

Record address	Label	ID number	Title	Composer(s)	Artist(s)
17	LON	2312	Romeo and Juliet	Prokofiev	Maazel
62	RCA	2626	Quartet in C Sharp Minor	Beethoven	Julliard
117	WAR	23699	Touchstone	Corea	Corea
152	ANG	3795	Symphony No. 9	Beethoven	Giulini
196	COL	38358	Nebraska	Springsteen	Springsteen
241	DG	18807	Symphony No. 9	Beethoven	Karajan
285	MER	75016	Coq d'Or Suite	Rimsky-Korsakov	Leinsdorf
338	COL	31809	Symphony No. 9	Dvorak	Bernstein
382	DG	139201	Violin Concerto	Beethoven	Ferras
427	FF	245	Good News	Sweet Honey in the Rock	Sweet Honey in the Rock

Figure 7.2 Contents of sample recording file.

Arquivo de Índice

Index

Key	*Reference field*
ANG3795	152
COL31809	338
COL38358	196
DG139201	382
DG18807	241
FF245	427
LON2312	17
MER75016	285
RCA2626	62
WAR23699	117

Recording file

Address of record	*Actual data record*
17	LON | 2312 | Romeo and Juliet | Prokofiev | ...
62	RCA | 2626| Quartet in C Sharp Minor | Beethoven | ...
117	WAR | 23699 | Touchstone | Corea | ...
152	ANG | 3795 | Symphony No. 9 | Beethoven | ...
196	COL | 38358 | Nebraska | Springsteen | ...
241	DG | 18807 | Symphony No. 9 | Beethoven | ...
285	MER | 75016 | Coq d'Or Suite | Rimsky-Korsakov | ...
338	COL | 31809 | Symphony No. 9 | Dvorak | ...
382	DG | 139201 | Violin Concerto | Beethoven | ...
427	FF | 245 | Good News |Sweet Honey in the Rock | ...

Figure 7.3 Index of the sample recording file.

Arquivo de Índice

- Cada par (chave , localização) é um registro
  - implementação eficiente usa **registros de tamanho fixo**
    - chave e localização (byte offset) como campos de tamanho fixo
    - pode eventualmente conter outros campos
      - p. ex. tamanho do registro no arquivo de dados
- Em geral, mantido ordenado
  - com registros de tamanho fixo, permite busca binária (BB)
- Menor e mais simples que o arquivo de dados original
  - muitas vezes cabe todo em memória primária!

8

Arquivo de Índice

- O Arquivo de Dados, em contraste...
  - em geral, muito maior que o arquivo de índices
  - em geral, possui registros de tamanho variável
  - em geral, "organizado" segundo a ordem de entrada dos registros
    - ***entry sequenced file***

9

Arquivos de Índice Moderados

- A manutenção e busca de registros no arquivo de dados será muito mais eficiente se o arquivo de índice puder ser carregado e manipulado em RAM
  - Isso é possível em muitos casos, quando o arquivo de índice possui tamanho "moderado"

10

Arquivos de Índice Moderados

- Exemplo:
  - Arquivo de dados com 10^6 registros de ~1Kb em média
    - deve ser indexado até byte offset ~ $10^6 \times 1000$ = 1 Bilhão
    - 4 bytes são mais que suficientes para representar esse offset
  - Arquivo de índice com registros de 24 bytes
    - 4 bytes para o byte offset + 20 bytes para a chave
      - CPF, por exemplo, requer apenas 11 bytes na maior representação
    - Com 10^6 registros, arquivo de índice ocupa 24×10^6 = **24Mb**

11

Operações Básicas

- Para arquivos de índice que cabem em RAM:
  - Carrega-se todo o índice em um vetor
  - **Busca**
    - Depois de carregado o índice, qualquer registro é localizado e recuperado em RAM com **O(1) acessos** externos
      - qq. consulta será O(1), ao preço fixo dos acessos para ler todo o índice
    - Em RAM, localização da chave no índice é muito rápida
      - Se índice não estiver ordenado, busca é seqüencial
      - Mas normalmente mantém-se o índice ordenado, para permitir BB

12

Operações Básicas

- Para arquivos de índice que cabem em RAM:
  - Carrega-se todo o índice em um vetor
  - **Inserção**
    - novo registro é inserido no final do arquivo de dados ou segundo uma política do tipo first-fit ou worst-fit
    - um registro associado é também inserido no índice
      - contém a chave e o byte offset do novo registro no arquivo de dados
      - se índice está em vetor ordenado, inserção demanda deslocamentos
        - mas em RAM, isso não demanda qualquer acesso

Operações Básicas

- Para arquivos de índice que cabem em RAM:
  - Carrega-se todo o índice em um vetor
  - **Remoção**
    - registro é removido do arquivo de dados segundo alguma política de marcação de registros removidos (p. ex. first-fit)
    - o registro associado deve também ser removido do índice
      - deslocamentos ou marcação da célula correspondente do vetor
        - não demanda qualquer acesso

Operações Básicas

- Para arquivos de índice que cabem em RAM:
  - Carrega-se todo o índice em um vetor
  - **Atualização**
    - Altera-se o registro no arquivo de dados
    - Se atualização mudou o valor da chave (remoção + inserção):
      - altera-se o registro no vetor de índices em RAM
        - chave e, eventualmente, byte offset (porquê ???)
      - reordena-se o vetor de índices
    - Se atualização não mudou o valor da chave:
      - se tamanho do registro não aumenta, nada muda no índice
      - caso contrário, muda-se apenas o byte offset no índice

15

Operações Básicas

- Para arquivos de índice que cabem em RAM:
  - Ao final de uma seção de operações
    - deve-se atualizar o arquivo de índice no disco
      - caso sua cópia em memória tenha sido alterada
  - É imperativo que o programa se proteja contra índices desatualizados
    - queda de energia
    - crashs do sistema (software ou do hardware)
    - CRTL + C ...

16

Prevenção de Índices Desatualizados

- Deve haver um mecanismo que permita saber se o índice está atualizado em relação ao arquivo de dados
- Possibilidade:
  - Um *flag* de status é setado no arquivo índice mantido em disco assim que a sua cópia na memória é alterada
  - Esse *flag* pode ser mantido no registro cabeçalho do arquivo índice, e atualizado sempre que o índice é reescrito no disco
- Se um programa detecta que o índice está desatualizado, uma função é ativada que reconstrói o índice a partir do arquivo de dados

17

Exercícios

- Elabore um arquivo com pelo menos 10 registros de clientes de um banco, contendo necessariamente o CPF de cada cliente como um dos campos
  - Mostre esse arquivo com os campos separados por delimitadores e registros com indicadores de tamanho (armazenados em ASCII – 2 bytes)
  - Assuma que o seu arquivo será precedido de um registro de cabeçalho de 20 bytes; então calcule o byte offset de início de cada um dos demais registros do arquivo.
  - Com os byte offsets em mãos, mostre o arquivo de índice do seu arquivo de dados de clientes, usando o CPF como chave primária. Use registros de índice de 15 bytes, 11 para a chave e 4 para o byte offset (ambos armazenados em ASCII).

18

Exercícios

- Remova alguns registros do arquivo do exercício anterior e mostre como fica o arquivo de dados e o arquivo de índice após essas remoções, considerando que, no primeiro, os registros removidos são encadeados em uma lista de slots vagos disponíveis, enquanto que, no segundo, os registros removidos são apenas marcados para serem eventualmente preenchidos durante os deslocamentos ocasionados por uma inserção.
- Faça a inserção de alguns novos registros no arquivo resultante do exercício anterior tentando aproveitar slots vagos tanto no arquivo de dados (diretamente, através da lista de disponíveis) como no arquivo de índices (indiretamente, para interromper uma seqüência de deslocamentos até o final do arquivo).

19

Outros Exercícios

- Capítulo 7 (Folk & Zoellick, 1987)
- Lista de Exercícios (CoTeia)
  - **Nota.** A lista faz referências à 2ª edição do livro de Folk & Zoellic. Nesse caso, o capítulo de indexação é o Capítulo 6
    - FOLK, M. & ZOELLICK, B., *File Structures*, 2nd Edition, Addison-Wesley, 1992.

20

Bibliografia

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

21