

Fundamentos de Arquivos

Adaptado dos Originais de:

Leandro C. Cintra

Maria Cristina F. de Oliveira

Arquivos

- Informação mantida em memória secundária (externa)
 - HD
 - CD
 - DVD
 - Memórias Flash
 - Fitas magnéticas
 - ...

2

Memória Secundária x Principal

- Tempo de Acesso
 - Ordem de grandeza da diferença entre os tempos de acesso pode ser de milhares
 - Por exemplo, discos (HDs, ...) podem ser milhares de vezes mais lentos que memória RAM

3

Discos X Memória Principal

- Capacidade de Armazenamento
 - Disco – alta, a um custo relativamente baixo
 - RAM – limitada pelo custo e espaço
- Tipo de Armazenamento
 - Disco – não volátil
 - RAM – volátil

4

Memória Secundária x Principal

- Em resumo:
 - acesso a dispositivo secundário é custoso (lento) !
- Logo:
 - número de acessos deve ser minimizado
- Para tanto:
 - Estruturas de organização de informação
 - Estruturas de Arquivos (File Structures)

5

Organização de Arquivos

- Meta:
 - minimizar as desvantagens do uso da memória externa
- Objetivo:
 - minimizar o tempo total de acesso ao dispositivo de armazenamento externo
 - de forma independente da tecnologia:

Tempo de Acesso = no. de acessos * tempo de 1 acesso

6

Memória Secundária x Principal

- Estruturas de dados eficientes em memória principal não o são em memória externa
 - Eficiência presume que toda a informação é instantaneamente acessível na memória se o respectivo endereço for conhecido
 - Tempo de acesso desprezível
 - Gargalos são as operações lógicas, aritméticas, ...

7

Dispositivo Externo como Gargalo

- Muitos processos são "disk-bounded", isto é, CPU e rede têm que esperar pelos dados do dispositivo externo:
 - Tais dispositivos (p. ex. discos) tipicamente são muito mais lentos que CPU e rede

8

Técnicas p/ Minimizar o Problema

- **Multi-Programação:**
 - CPU trabalha em outro processo enquanto aguarda o dispositivo externo
- **Striping:**
 - dados repartidos em vários *drives* (paralelismo)
- **RAID (*Redundant Array of Inexpensive Disks*):**
 - tecnologia baseada em striping
 - <http://linas.org/linux/raid.html>

9

Técnicas p/ Minimizar o Problema

- **Disk Cache:**
 - blocos de memória RAM configurados para conter páginas de dados do dispositivo externo
 - cache é verificado primeiro. Se a informação desejada não é encontrada, um acesso ao dispositivo externo é realizado, e o novo conteúdo é carregado no cache
- **Organização de Arquivos !**

10

Arquivo Físico e Arquivo Lógico

- **Arquivo Físico:**
 - seqüência de bytes armazenados no disp. externo
 - armazenamento em geral não é fisicamente seqüencial
- **Arquivo Lógico:**
 - arquivo como visto pelo aplicativo que o acessa
 - freqüentemente visão é seqüencial
- **Associação Arquivos Físico – Lógico:**
 - iniciada pelo aplicativo, gerenciada pelo S.O.

11

Arquivo Físico e Arquivo Lógico

- **Arquivo Físico:**
 - conjunto de bytes no dispositivo externo
 - geralmente agrupados em setores e clusters de dados
 - gerenciado pelo sistema operacional
- **Arquivo Lógico:**
 - modo como a *linguagem de programação* ou *aplicativo* enxerga os dados
 - seqüência de bytes eventualmente organizados em *registros* ou outra *estrutura lógica*

12

Associação Arquivo Físico – Lógico

- Em Pascal:
file arq; ou **var** arq: **text**;
assign(arq, 'meu_arq.dat');
- Em C: (associa e abre – nesse exemplo para escrita)
file *pt_arq;
if ((pt_arq=**fopen**("meu_arq.dat", "w")) == **NULL**)
 printf("erro...") // p. ex. disco cheio ou protegido
else ...

13

Revisão Arquivos em C

- pt_arq = **fopen**("filename", "flags")
 - filename: nome do arquivo a ser aberto
 - flags: controla o modo de abertura
 - **r**: abre arquivo texto somente para leitura
 - **w**: cria arq. texto só para escrita (se já existe, é apagado)
 - **a**: anexa (append) ou cria arquivo texto só para escrita
 - **r+**: abre arquivo texto para leitura / escrita
 - **w+**: cria arq. texto para leitura / escrita (se já existe, é apagado)
 - **a+**: anexa ou cria arquivo texto para leitura / escrita

14

Revisão Arquivos em C

- pt_arq = **fopen**("filename", "flags")
 - filename: nome do arquivo a ser aberto
 - flags: controla o modo de abertura
 - **rb**: abre arquivo binário somente para leitura
 - **wb**: cria arq. binário só para escrita (se já existe, é apagado)
 - **ab**: anexa (append) ou cria arquivo binário só para escrita
 - **r+b**: abre arquivo binário para leitura / escrita
 - **w+b**: cria arq. bin. para leitura / escrita (se já existe, é apagado)
 - **a+b**: anexa ou cria arquivo binário para leitura / escrita

15

Fechamento de Arquivos

- Em C: **fclose**(pt_arq)
 - Descarrega conteúdo do *buffer*, garantindo que todas as informações sejam atualizadas e salvas
 - Encerra a associação entre arquivos lógico e físico
- Notas:
 - Em geral, S.O. fecha o arquivo se o programa não o fizer (prevenindo contra perdas de informação)
 - **fflush**(pt_arq) força descarregar o *buffer* sem precisar fechar o arquivo, prevenindo interrupções

16

Revisão Arquivos em C

- Funções de Leitura / Escrita
 - **fread()**
 - **fwrite()**
 - dados lidos/escritos como registros ou blocos de bytes
 - exemplo: **fwrite**(&v[0], sizeof(**int**), 10, pt_arq);
 - **fgetc()**
 - **fputc()**
 - dados lidos/escritos um caractere por vez

17

Revisão Arquivos em C

- Funções de leitura / escrita
 - **fgets()**
 - **fputs()**
 - dados lidos/escritos como *strings*
 - **fscanf()**
 - **fprintf()**
 - dados lidos/escritos de modo formatado

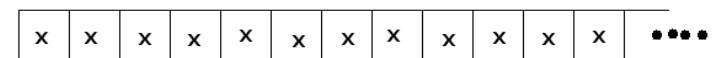
18

Revisão Arquivos em C

- Controle de final de arquivo:
 - Via retornos de funções de leitura/escrita:
 - **EOF** (tipo especial): **fgetc**, **fscanf**, ...
 - Ponteiro **NULL**: **fgets**
 - No. de *itens* lidos/escritos: **fread**, **fwrite**
 - Via **feof**(ponteiro)
 - função com retorno lógico

19

O Ponteiro no Arquivo Lógico



↑
Ponteiro indicando a posição atual
x - um byte qualquer

20



Acesso Seqüencial vs Aleatório

- **Leitura Seqüencial:**
 - ponteiro de leitura avança byte a byte (ou por blocos), a partir de uma posição inicial
- **Acesso Aleatório (Direto - Seeking):**
 - acesso envolve o posicionamento do ponteiro em um byte ou registro arbitrário

21



Revisão Arquivos em C

- Seeking:
 - mover o ponteiro para uma certa posição no arquivo
 - seeking em C
 - `bol = fseek(pt_arq, byte-offset, origin)`
 - `bol`: retorno lógico (0 ou 1 - mal ou bem sucedido)
 - `pt_arq`: ponteiro arquivo
 - `byte-offset`: deslocamento, em bytes, a partir de *origin*
 - `origin`:
 - **SEEK_SET** (tipo especial – início do arquivo)
 - **SEEK_CUR** (tipo especial – posição atual)
 - **SEEK_END** (tipo especial – fim do arquivo)

22



Bibliografia

- **M. J. Folk and B. Zoellick, *File Structures: A Conceptual Toolkit*, Addison Wesley, 1987.**
- **Schildt, H. "C Completo e Total", 3a. Edição, Pearson, 1997.**

23