Técnicas de Programação Avançada

Série 1 - Ordenação Externa

Conteúdo

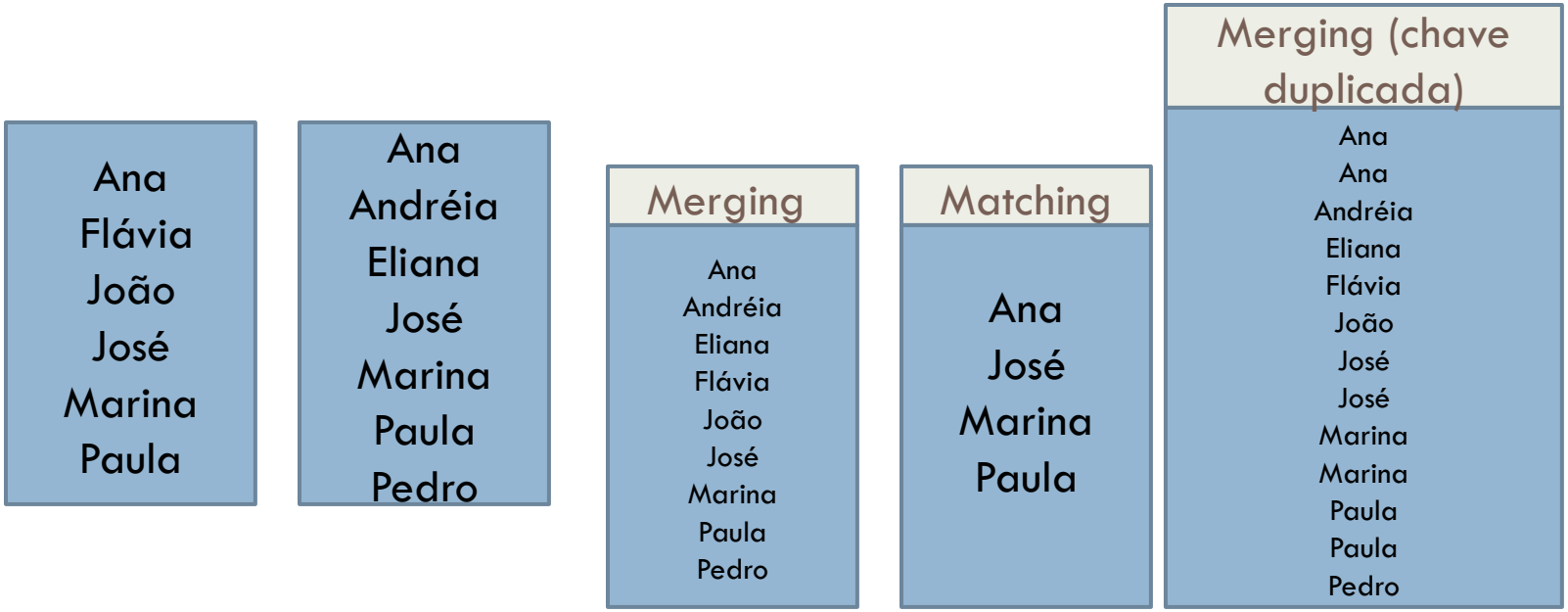
- Motivação
- Processamento Cosequencial
- Merging
- Matching
- Ordenação Externa Via Multi-Way Merging

Motivação

- Como ordenar um arquivo de 40 ou 100 Giga bytes?
- Quais os limites da ordenação interna?
- Onde estão os gargalos da ordenação Externa?

Processamento Coseqüencial

- Processamento coordenado (simultâneo) de duas ou mais "listas" de entradas seqüenciais, produzindo uma única lista como saída
- Exemplos típicos:
  - *Merging* (união) ou *Matching* (intersecção) de dois ou mais conjuntos de registros mantidos em arquivos separados e ordenados por chave



Operações Coseqüenciais

Merging:

- lê uma entrada de cada lista/arquivo e as compara
- se ambas são iguais
  - copia a entrada na lista/arquivo de saída e avança para a próxima entrada em cada lista
- se uma das entradas é menor:
  - copia essa entrada na saída e lê a próxima entrada da respectiva lista
- retorna ao segundo passo até que ambas as listas terminem

Operações Cosequenciais

Matching:

- lê uma entrada de cada lista/arquivo e as compara
- se ambas são iguais,
  - copia a entrada na lista/arquivo de saída e avança para a próxima entrada em cada lista
- se uma das entradas é menor:
  - lê a próxima entrada da respectiva lista
- retorna ao segundo passo até que uma das listas termine

Operações Cosequenciais

Multi-Way:

- Operações coseqüenciais, tais como merging e matching, não se restringem em duas listas (2-way)
- Versões k-way são obtidas como generalizações de 2-way
- Muda apenas a comparação:
  - Avança-se para a próxima entrada em toda lista cuja entrada corrente for mínima dentre todas as entradas correntes

Exercício

1. Escreva um algoritmo/programa (em C ou Java) que realize a leitura de n vetores, com para n ={2, 3} de inteiros ordenados e gere três vetores de saída equivalentes a um matching, a um merging (união), e um merging com repetição.
2. Realize um teste com três vetores de entrada gerados aleatoriamente. Gere os maiores vetores que você conseguir manter em memória primária. Meça os tempos de execução de suas operações com estes vetores.

Ordenação Externa

Ordenação Externa via Muti-Way Merging

- Pode-se utilizar uma modificação da operação coseqüencial de merging multi-way para ordenar um arquivo grande em disco: Merging como soma (chaves repetidas) e não como união.

Idéia:

- Carrega-se toda a RAM disponível com parte do arquivo
- Ordena-se os registros em RAM com um algoritmo de ordenação interna
- Escreve-se os registros ordenados em um arquivo separado
- Repete-se os passos acima até encerrar o arquivo original
- Se a RAM disponível comporta (1/k * no. de regs. do arq. original),

  então ter-se-ão k arquivos ordenados
- Aplica-se então multi-way merging nos arquivos ordenados

Ordenação Externa

- Multi-Way Merging nos Arquivos Ordenados:
- Para maximizar a eficiência da operação de merging, opera-se com L/E de blocos em RAM
- A RAM disponível é sub-dividida em k buffers de entrada: 1 buffer para cada um dos k arquivos ordenados
- A "RAM disponível" já desconta uma porção reservada para buffer de saída
- Os buffers são preenchidos com 1/k regs. dos respectivos arquivos
- O Merging é realizado em RAM
- Cada buffer de entrada é recarregado sempre que vazio
- Como cada arquivo ordenado é exatamente do tamanho da RAM disponível, cada buffer é carregado k vezes
- O Buffer de saída é descarregado no arquivo de saída sempre que cheio

Ordenação Externa

- Ex:
- A: 10 4 20 3 11 2 9 0 1 8 5 7 19 13 12 6
- Tam. Memória disponível: 4 (já descontado a memória do buffer de saida)
- A1: 10 4 20 3 → 3 4 10 20
- A2: 11 2 9 0 → 0 2 9 11
- A3: 1 8 5 7 → 1 5 7 8
- A4: 19 13 12 6 → 6 12 13 19

Ordenação Externa

- Exemplo (40-way):
- Arquivo com 40Gb (40.000.000 de registros de 1Kb)
- 1GB de RAM disponível
- RAM comporta 1.000.000 de registros
- Com 40 ordenações em RAM produz-se 40 arquivos ordenados
- Cada arquivo com 1.000.000 de registros do arquivo original
- Com merging 40-way produz-se um único arquivo ordenado
- RAM é dividida em 40 blocos de 1/40 Gb → cada bloco com 25M.

Ordenação Externa

Desempenho

- O multi-way merging do exemplo anterior necessita, no mínimo, 1600 seeks no disco:
  - Mesmo que cada bloco de registros possa ser lido com um único seek, serão necessários $k = 40$ seeks para cada arquivo
  - Como são 40 arquivos, tem-se ao menos $40 \times 40 = 1600$ seeks
  - Fora os seeks para escrita do arquivo de saída

Ordenação Externa

- Estratégias para Melhorar o Desempenho:
  - Aumento do tamanho e redução da quantidade de arquivos ordenados para serem fundidos:
    - Merging em múltiplos passos

Ordenação Externa

- Merging em Múltiplos Passos:
  - No exemplo anterior, ao invés de fazer o merging dos 40 arquivos, seria vantajoso fazer o merging em múltiplas etapas ? Por exemplo:
    - 1ª etapa: 5 mergings de 8 arquivos cada, produzindo 5 arquivos:
      - cada merging demanda $8 \times 8 = 64$ seeks (o bloco agora é de 1/8 Gbytes)
      - total de $64 \times 5 = 320$ seeks
    - 2ª etapa: 1 merging dos 5 arquivos, produzindo 1 arquivo
      - RAM disponível comporta 1/8 de cada arquivo (cada arquivo de 8Gbytes)
      - logo, cada um dos 5 blocos em RAM comporta 1/40 de cada arquivo (1/5 de 1/8 de cada arquivo)
      - 40 seeks por arquivo $\Rightarrow$ total de 200 seeks

Ordenação Externa

Merging em Múltiplos Passos:

- A estratégia de 2 passos no exemplo resultou em um total de 320+200 = 520 seeks, contra 1600 da estratégia anterior.
- No entanto, esses valores são apenas limitantes inferiores:
  - Consideram que um bloco de registros sempre pode ser lido em um único seek, independente de seu tamanho
  - Mesmo com essa simplificação, em cada caso os seeks estarão associados à transmissão de quantidades diferentes de registros:
  - As análises anteriores ignoram o tempo de transmissão de registros, dentre vários outros aspectos que dependem de hardware e software
  - A Escolha de uma determinada estratégia em um projeto deve considerar em detalhes todos esses aspectos...

Ordenação Externa

- As análises dos métodos de ordenação tradicionais se preocupam basicamente com o tempo de execução dos algoritmos
  - Ordem computacional é estimada em função da quantidade de operações (comparações, trocas, etc) feitas utilizando a memória principal (primária) da máquina
  - Modelo de ordenação interna
- Quando é preciso ordenar uma base de dados muito grande, que não cabe na memória principal, um outro modelo faz-se necessário
  - No modelo de ordenação externa assume-se que os dados devem ser recuperados a partir de dispositivos externos
  - ordenação em memória secundária

Ordenação Externa

Complexidade de Métodos de ordenação Interna

	Complexidade
Inserção	$O(n^2)$
Seleção	$O(n^2)$
Shellsort	$O(n \log n)$
Quicksort	$O(n \log n)$
Heapsort	$O(n \log n)$

Ordenação Externa

- Como o acesso à memória secundária é muito mais lento, a maior preocupação passa a ser minimizar a quantidade de leituras e escritas nos respectivos dispositivos
- Uma dificuldade é que o projeto e análise dos métodos de ordenação externa dependem fortemente do estado da tecnologia
  - Por exemplo, o acesso a dados em fitas magnéticas é seqüencial, mais lento, enquanto em discos tem-se o acesso direto
  - Nesses últimos, no entanto, tem-se o tempo de localização de trilha (seek time) e de setor/cluster (latency time), que por sua vez dependem da velocidade de rotação do disco, da estrutura de dados utilizada para armazenamento, etc.

Ordenação Externa

- Usualmente utiliza-se um modelo simplificado, que abstrai ao máximo os detalhes tecnológicos
- Leva-se em conta a quantidade de operações envolvendo a transferência de blocos de registros entre as memórias primária e secundária

Ordenação Externa

- O modelo de ordenação externa assume que o hardware e o S.O. são dados (e.g. tamanho dos blocos)
- Assume-se usualmente que os blocos contêm múltiplos registros
- Assume-se usualmente que os registros possuem tamanho fixo
  - Caso contrário as análises ganham um caráter de "estimativa média"

Ordenação Externa

- Ordenação externa via multi-way merging:
  - Sabemos que é possível ordenar um arquivo grande em disco separando-o em k arquivos ordenados em RAM e fazendo a fusão intercalada (merging) desses arquivos
- Essa abordagem, porém, possui uma limitação:
  - A quantidade k e o tamanho dos arquivos são determinados pela memória primária disponível e pelo tamanho do arquivo a ser ordenado
    - Fusão pode ter que lidar com diversos arquivos de tamanho reduzido e necessariamente ordenados
    - Pode inviabilizar a paralelização de L/E em múltiplos dispositivos.

Ordenação Externa

- Seria possível fazer a fusão ordenada de um número arbitrário de arquivos de tamanho arbitrário, não ordenados e possivelmente armazenados em diferentes dispositivos externos? SIM.
  - São necessárias múltiplas passagens coseqüenciais pelos arquivos.
  - Algoritmo é denominado Merge-Sort Externo

Ordenação Externa

- Exemplo do algoritmo operando com 4 arquivos:
  - Assume-se que todos os 4 arquivos são armazenados em um único disco.
  - Os registros são lidos de 2 arquivos de origem e reescritos de forma parcialmente ordenada em 2 arquivos de destino.
- Os arquivos de origem e destino se alternam nas sucessivas iterações do algoritmo.
- Utiliza-se o conceito de rodada (run):
  - Subconjuntos ordenados de registros

Ordenação Externa

- Inicialmente divide-se o arquivo original em dois arquivos f1 e f2 ditos de origem, com as seguintes propriedades:
  - O número de rodadas de f1 e f2 (incluindo eventual cauda) difere em no máximo 1
  - No máximo um dentre f1 e f2 possui uma cauda
  - Aquele com cauda possui pelo menos tantas rodadas quanto o outro
- Uma cauda é uma rodada com número incompleto de registros
  - f1: 7 15 29 | 8 11 13 | 16 22 31 | 5 12
  - f2: 8 19 54 | 4 20 33 | 00 10 62 |

Ordenação Externa

- Em princípio vamos tratar os arquivos como estando organizados em rodadas de tamanho unitário:
  - Iniciado:
    - f1 : 28 03 93 10 54 65 30 90 10 69 08 22
    - f2 : 31 05 96 40 85 09 39 13 08 77 10
- Inicia-se então a leitura de blocos de registros dos arquivos
  - Por hora considera-se que conjuntos de registros são lidos sequencialmente de ambos os arquivos e intercalados
    - Algoritmo merging por rodadas
  - Isso significa **que cada rodada de um arquivo é fundida com a rodada correspondente do outro arquivo**, formando uma rodada com o dobro do tamanho

Ordenação Externa

- Ao final de cada passagem pelos arquivos, tem-se os arquivos ditos de destino, g1 e g2 , organizados em rodadas com o dobro do tamanho dos arquivos de origem:
  - 1a Passagem:
    - g1: 28 31 | 93 96 | 54 85 | 30 39 | 08 10 | 08 10
    - g2: 03 05 | 10 40 | 09 65 | 13 90 | 69 77 | 22

- Esses arquivos tornam-se então os arquivos de origem e o processo se repete:
  - 2a Passagem:
    - g1: 03 05 28 31 | 09 54 65 85 | 08 10 69 77
    - g2: 10 40 93 96 | 13 30 39 90 | 08 10 22

Ordenação Externa

- 3a Passagem

g1: 03 05 10 28 31 40 93 96 | 08 08 10 10 22 69 77

g2: 09 13 30 39 54 65 85 90 |

- 4a Passagem

G1: 03 05 09 10 13 28 30 31 39 40 54 65 85 90 93 96

g2: 08 08 10 10 22 69 77

- 5ª passagem

03 05 08 08 09 10 10 10 13 22 28 30 31 39 40 54 65 69 77 85 90 93 96

Ordenação Externa

- Número de passagens?
- Como o tamanho das rodadas dobram a cada passagem, tem-se que após i passagens o tamanho da rodada é k = 2^i e que quando k >= n (onde n é a quantidade total de registros a serem ordenados) tem-se:
  - O número de passagens necessárias é portanto tal que 2^i>=n
  - Logo, qualquer i >=log n número de passagens são suficientes:
  - Ou seja, não mais que Teto[log n] passagens bastam.
- Como são n registros e a fusão se dá pela comparação de pares de chaves em tempo constante, a complexidade do algoritmo em termos de números de comparações é O(n log n)
- A mesma que o Merge-Sort recursivo tradicional (ordenação interna)

Referências

- http://wiki.icmc.usp.br
- Projeto e Análise de Algoritmos – Nívio Ziviani.