

AEDS III

Ordenação Externa
Prof. Olga Goussevskaia

Ordenação Externa

- Exemplo de aplicação: MapReduce do Google
 - Framework de computação distribuída e paralela.
 - Realiza operações simples sobre entradas muito grandes de dados, onde a computação sobre uma parte não afeta a computação sobre a outra parte (admite paralelização trivial)

MapReduce do Google

- Exemplo de problema: contar quantas vezes cada URL foi acessada.
 - Entrada: vários arquivos de log de acessos
 - Estimativa do número de documentos indexados pelo Google em 2008: 1 trilhão (10^{12}).
- **Map**: dividir a entrada de dados e distribuir entre vários processadores (p_i). Retorno: lista de pares $\langle \text{URL}, \text{count}_i \rangle$
- **Reduce**: agregar as saídas da fase Map através de uma *ordenação externa* e emitir o resultado total $\langle \text{URL}, \text{count}_{\text{total}} \rangle$

Ordenação Externa

- A ordenação externa consiste em ordenar arquivos de tamanho maior que a memória interna disponível.
- Os métodos de ordenação externa são muito diferentes dos de ordenação interna.
- Na ordenação externa os algoritmos devem diminuir o número de acesso as unidades de memória externa.
- Nas memórias externas, os dados ficam em um arquivo seqüencial.
- Apenas um registro pode ser acessado em um dado momento. Essa é uma restrição forte se comparada com as possibilidades de acesso em um vetor.
- Logo, os métodos de ordenação interna são inadequados para ordenação externa.
- Técnicas de ordenação diferentes devem ser utilizadas.

Ordenação Externa

Fatores que determinam as diferenças das técnicas de ordenação externa:

1. Custo para acessar um item é algumas ordens de grandeza maior.
2. O custo principal na ordenação externa é relacionado a transferência de dados entre a memória interna e externa.
3. Existem restrições severas de acesso aos dados.
4. O desenvolvimento de métodos de ordenação externa é muito dependente do estado atual da tecnologia.
5. A variedade de tipos de unidades de memória externa torna os métodos dependentes de vários parâmetros.
6. Assim, apenas métodos gerais serão apresentados.

Ordenação Externa

- O método mais importante é o de ordenação por intercalação.
- Intercalar significa combinar dois ou mais blocos ordenados em um único bloco ordenado.
- A intercalação é utilizada como uma operação auxiliar na ordenação.
- Estratégia geral dos métodos de ordenação externa:
 1. Quebre o arquivo em blocos do tamanho da memória interna disponível.
 2. Ordene cada bloco na memória interna.
 3. Intercale os blocos ordenados, fazendo várias passadas sobre o arquivo.
 4. A cada passada são criados blocos ordenados cada vez maiores, até que todo o arquivo esteja ordenado.

Ordenação Externa

- Os algoritmos para ordenação externa devem reduzir o número de passadas sobre o arquivo.
- Uma boa medida de complexidade de um algoritmo de ordenação por intercalação é o número de vezes que um item é lido ou escrito na memória auxiliar.
- Os bons métodos de ordenação geralmente envolvem no total menos do que dez passadas sobre o arquivo.

Intercalação Balanceada de Vários Caminhos

- Considere um arquivo armazenado em uma fita de entrada:

I N T E R C A L A C A O B A L A N C E A D A

- Objetivo:
 - Ordenar os 22 registros e colocá-los em uma fita de saída.
- Os registros são lidos um após o outro.
- Considere uma memória interna com capacidade para para três registros.
- Considere que esteja disponível seis unidades de fita magnética.

Intercalação Balanceada de Vários Caminhos

- Fase de criação dos blocos ordenados:

fitas 1: *I N T A C O A D E*

fitas 2: *C E R A B L A*

fitas 3: *A A L A C N*

Intercalação Balanceada de Vários Caminhos

- Fase de intercalação - Primeira passada:
 1. O primeiro registro de cada fita é lido.
 2. Retire o registro contendo a menor chave.
 3. Armazene-o em uma fita de saída.
 4. Leia um novo registro da fita de onde o registro retirado é proveniente.
 5. Ao ler o terceiro registro de um dos blocos, sua fita fica inativa.
 6. A fita é reativada quando o terceiro registro das outras fitas forem lidos.

Intercalação Balanceada de Vários Caminhos

- Fase de intercalação - Primeira passada:
 7. Neste instante um bloco de nove registros ordenados foi formado na fita de saída.
 8. Repita o processo para os blocos restantes.
- Resultado da primeira passada da segunda etapa:

fita 4: *A A C E I L N R T*

fita 5: *A A A B C C L N O*

fita 6: *A A D E*

Intercalação Balanceada de Vários Caminhos

- Quantas passadas são necessárias para ordenar um arquivo de tamanho arbitrário?
 - Seja n , o número de registros do arquivo.
 - Suponha que cada registro ocupa m palavras na memória interna.
 - A primeira etapa produz n/m blocos ordenados.
 - Seja $P(n)$ o número de passadas para a fase de intercalação.
 - Seja f o número de caminhos intercalados.
 - Assim:

$$P(n) = \log_f \frac{n}{m}.$$

No exemplo acima, $n=22$, $m=3$ e $f=3$ temos:

$$P(n) = \log_3 \frac{22}{3} = 2.$$

Implementação por meio de Seleção por Substituição

- A implementação do método de intercalação balanceada pode ser realizada utilizando filas de prioridades.
- As duas fases do método podem ser implementadas de forma eficiente e elegante.
- Operações básicas para formar blocos ordenados:
 - Obter o menor dentre os registros presentes na memória interna.
 - Substituí-lo pelo próximo registro da fita de entrada.
- Estrutura ideal para implementar as operações: *heap*.
- Operação de substituição:
 - Retirar o menor item da fila de prioridades.
 - Colocar um novo item no seu lugar.
 - Reconstituir a propriedade do *heap*.

Implementação por meio de Seleção por Substituição

Algoritmo:

1. Inserir m elementos do arquivo na fila de prioridades.
2. Substituir o menor item da fila de prioridades pelo próximo item do arquivo.
3. Se o próximo item é menor do que o que saiu, então:
 - Considere-o membro do próximo bloco.
 - Trate-o como sendo maior do que todos os itens do bloco corrente.
4. Se um item marcado vai para o topo da fila de prioridades então:
 - O bloco corrente é encerrado.
 - Um novo bloco ordenado é iniciado.

Implementação por meio de Seleção por Substituição

Entra	1	2	3
<i>E</i>	<i>I</i>	<i>N</i>	<i>T</i>
<i>R</i>	<i>N</i>	<i>E*</i>	<i>T</i>
<i>C</i>	<i>R</i>	<i>E*</i>	<i>T</i>
<i>A</i>	<i>T</i>	<i>E*</i>	<i>C*</i>
<i>L</i>	<i>A*</i>	<i>E*</i>	<i>C*</i>
<i>A</i>	<i>C*</i>	<i>E*</i>	<i>L*</i>
<i>C</i>	<i>E*</i>	<i>A</i>	<i>L*</i>
<i>A</i>	<i>L*</i>	<i>A</i>	<i>C</i>
<i>O</i>	<i>A</i>	<i>A</i>	<i>C</i>
<i>B</i>	<i>A</i>	<i>O</i>	<i>C</i>
<i>A</i>	<i>B</i>	<i>O</i>	<i>C</i>

Entra	1	2	3
<i>L</i>	<i>C</i>	<i>O</i>	<i>A*</i>
<i>A</i>	<i>L</i>	<i>O</i>	<i>A*</i>
<i>N</i>	<i>O</i>	<i>A*</i>	<i>A*</i>
<i>C</i>	<i>A*</i>	<i>N*</i>	<i>A*</i>
<i>E</i>	<i>A*</i>	<i>N*</i>	<i>C*</i>
<i>A</i>	<i>C*</i>	<i>N*</i>	<i>E*</i>
<i>D</i>	<i>E*</i>	<i>N*</i>	<i>A</i>
<i>A</i>	<i>N*</i>	<i>D</i>	<i>A</i>
	<i>A</i>	<i>D</i>	<i>A</i>
	<i>A</i>	<i>D</i>	
	<i>D</i>		

- Primeira passada sobre o arquivo exemplo.
- Os asteriscos indicam quais chaves pertencem a blocos diferentes.

Implementação por meio de Seleção por Substituição

- Fase de intercalação dos blocos ordenados obtidos na primeira fase:
 - Operação básica: obter o menor item dentre os ainda não retirados dos f blocos a serem intercalados.

Algoritmo:

- Monte uma fila de prioridades de tamanho f .
- A partir de cada uma das f entradas:
 - Substitua o item no topo da fila de prioridades pelo próximo item do mesmo bloco do item que está sendo substituído.
 - Imprima em outra fita o elemento substituído.

Considerações Práticas

- Escolha da ordem de intercalação f :
 - Para fitas magnéticas:
 - * f deve ser igual ao número de unidades de fita disponíveis menos um.
 - * A fase de intercalação usa f fitas de entrada e uma fita de saída.
 - * O número de fitas de entrada deve ser no mínimo dois.
 - Para discos magnéticos:
 - * O mesmo raciocínio acima é válido.
 - * O acesso seqüencial é mais eficiente.
 - Sedegwick (1988) sugere considerar f grande o suficiente para completar a ordenação em poucos passos.
 - Porém, a melhor escolha para f depende de vários parâmetros relacionados com o sistema de computação disponível.

Ordenação Externa

- Como escolher f (número de caminhos)?
 - Ex: $n=200$ milhões, $m=1$ milhão, $P(n)=2$?
$$f^2 \geq \text{teto}(n/2m) \Rightarrow f \geq \sqrt{100} \Rightarrow f \geq 10$$
- E o número de fitas?
 - Intercalação balanceada: $2f$ (muito grande)
 - Alternativa: $f+1$
 - Usar uma única fita de saída
 - Custo: uma cópia adicional a cada passada da fase de intercalação!
 - O método da intercalação polifásica elimina este custo adicional (Cap. 4.2.4 Livro do Nívio Ziviani)

Quicksort Externo

- Quicksort interno (paradigma de divisão e conquista):
 - Escolher um elemento pivot p da lista (rand/mediana)
 - Particionamento: reordenar a lista, tal que todos os elementos $<p$ fiquem antes de p , e todos os elementos $>p$ fiquem depois de p . (se $=p$, qq lado)
 - Recursivamente ordenar as sub-listas $<p$ e $>p$.
- Quicksort externo (mesmo paradigma)
 - Igual ao quicksort interno, so que o pivot é substituído por um buffer de tamanho m ($m=O(\log n)$, $m \geq 3$)
 - É um algoritmo *in situ*, ou seja, não necessita de memória externa adicional.
 - Os n registros a serem ordenados estão em memória secundária de acesso randômico.

Quicksort Externo

- Foi proposto por Monard em 1980.
- Utiliza o paradigma de **divisão e conquista**.
- O algoritmo ordena *in situ* um arquivo $A = \{R_1, \dots, R_n\}$ de n registros.
- Os registros estão armazenados consecutivamente em memória secundária de acesso randômico.
- O algoritmo utiliza somente $O(\log n)$ unidades de memória interna e não é necessária nenhuma memória externa adicional.

Quicksort Externo

- Seja R_i , $1 \leq i \leq n$, o registro que se encontra na i -ésima posição de A .
- Algoritmo:
 1. Particionar A da seguinte forma:
$$\{R_1, \dots, R_i\} \leq R_{i+1} \leq R_{i+2} \leq \dots \leq R_{j-2} \leq R_{j-1} \leq \{R_j, \dots, R_n\},$$
 2. chamar recursivamente o algoritmo em cada um dos subarquivos $A_1 = \{R_1, \dots, R_i\}$ e $A_2 = \{R_j, \dots, R_n\}$.

Quicksort Externo

- Para o partionamento é utilizada uma área de armazenamento na memória interna.
- Tamanho da área: $\text{TamArea} = j - i - 1$, com $\text{TamArea} \geq 3$.
- Nas chamadas recursivas deve-se considerar que:
 - Primeiro deve ser ordenado o subarquivo de menor tamanho.
 - Condição para que, na média, $O(\log n)$ subarquivos tenham o processamento adiado.
 - Subarquivos vazios ou com um único registro são ignorados.
 - Caso o arquivo de entrada A possua no máximo TamArea registros, ele é ordenado em um único passo.

Quicksort Externo

1. Lê os primeiros $m/2$ e os últimos $m/2$ elementos para o buffer e ordena
2. Guarda $\max(\text{buffer})$ e $\min(\text{buffer})$ para evitar reordenar elementos no meio q já estão escritos
3. Lê o próximo elemento x do começo ou do fim, alternadamente, de forma a balancear a escrita
4. Se $x \leq \min(\text{buffer})$, escreve x no espaço liberado no começo do arquivo de entrada
Se $x \geq \max(\text{buffer})$, escreve x no fim do arquivo
Senão, escreve $\min(\text{buffer})$ ou $\max(\text{buffer})$ e coloca x no buffer
5. Ao terminar, escrever o conteúdo do buffer
6. Recursivamente, ordena(partição menor); ordena(partição restante)

Quicksort Externo

- Exemplo: ordenar $A\{5, 3, 10, 6, 1, 7, 4\}$, $m=3$
 - Lê $\{4,5,7\}$, ordene em memória: $B\{4,5,7\}$, $\text{min}=4, \text{max}=7$
 - Lê $\{3\}$: ($3 \leq \text{min}$) \Rightarrow escreve 3: $A\{\textcolor{red}{3}, \textcolor{green}{3}, 10, 6, 1, \textcolor{green}{7}, \textcolor{green}{4}\}$
 - Lê $\{1\}$: ($1 \leq \text{min}$) \Rightarrow escreve 1: $A\{\textcolor{red}{3}, \textcolor{red}{1}, 10, 6, \textcolor{green}{1}, \textcolor{green}{7}, \textcolor{green}{4}\}$
 - Lê $\{10\}$: ($10 \geq \text{max}$) \Rightarrow escreve 10: $A\{\textcolor{red}{3}, \textcolor{red}{1}, \textcolor{green}{10}, 6, \textcolor{green}{1}, \textcolor{green}{7}, \textcolor{red}{10}\}$
 - Lê $\{6\}$: ($\text{min} < 6 < \text{max}$) \Rightarrow escreve max (p/ balancear), coloca $\{6\}$ no buffer:
 $A\{\textcolor{red}{3}, \textcolor{red}{1}, \textcolor{green}{10}, \textcolor{green}{6}, \textcolor{green}{1}, \textcolor{red}{7}, \textcolor{red}{10}\}$, $B\{4,5,6\}$, $\text{min}=4, \text{max}=6$
 - Escreve o buffer no arquivo: $A\{3, 1, \textcolor{blue}{4}, \textcolor{blue}{5}, \textcolor{blue}{6}, 7, 10\}$
 - Recursão: ordena $A\{3, 1\}$; ordena $A\{7,10\}$;

Quicksort Externo

- Exemplo: ordenar $A\{5, 3, 10, 6, 1, 7, 4\}$, $m=3$
 1. Lê $\{5,4,3\}$, ordene em memória: $B\{3,4,5\}$, $\text{min}=3, \text{max}=5$
 2. Lê $\{7\}$: $(7 \geq \text{max}) \Rightarrow$ escreve 7: $A\{5, 3, 10, 6, 1, 7, 7\}$
 3. Lê $\{1\}$: $(1 \leq \text{min}) \Rightarrow$ escreve 1: $A\{1, 3, 10, 6, 1, 7, 7\}$
 4. Lê $\{10\}$: $(10 \geq \text{max}) \Rightarrow$ escreve 10: $A\{1, 3, 10, 6, 1, 10, 7\}$
 5. Lê $\{6\}$: $(6 \geq \text{max}) \Rightarrow$ escreve 6: $A\{1, 3, 10, 6, 6, 10, 7\}$
 11. Escreve o buffer no arquivo: $A\{1, 3, 4, 5, 6, 10, 7\}$
 12. Recursão: ordena $A\{1\}$; ordena $A\{6,10,7\}$;

Quicksort Externo: Análise

- Seja n o número de registros a serem ordenados.
- Seja e e b o tamanho do bloco de leitura ou gravação do Sistema operacional.
- Melhor caso: $O(\frac{n}{b})$
 - Por exemplo, ocorre quando o arquivo de entrada já está ordenado.
- Pior caso: $O(\frac{n^2}{\text{TamArea}})$
 - ocorre quando um dos arquivos retornados pelo procedimento Particao tem o maior tamanho possível e o outro é vazio.
 - A medida que n cresce, a probabilidade de ocorrência do pior caso tende a zero.
- Caso Médio: $O(\frac{n}{b} \log(\frac{n}{\text{TamArea}}))$
 - É o que tem a maior probabilidade de ocorrer.