Ordenando arquivos grandes

Programação II – Engenharia de Telecomunicações

Prof. Emerson Ribeiro de Mello

mello@ifsc.edu.br

INSTITUTO FEDERAL
Santa Catarina
Campus São José

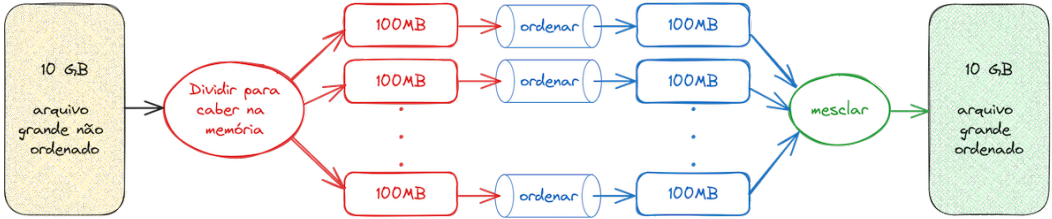
Licenciamento

Problema

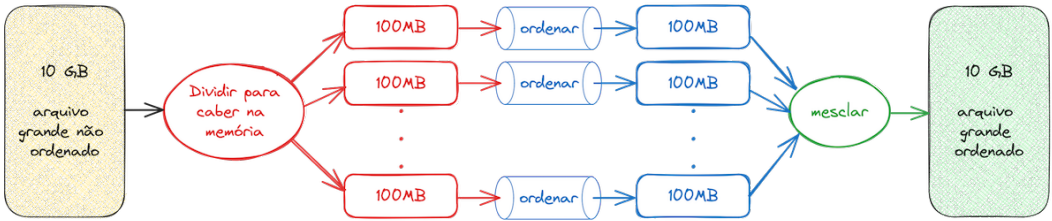
Em um computador com `4GB` de memória RAM, como ordenar um arquivo de texto com tamanho de `10GB`, sendo que em cada linha tem apenas um número inteiro?

- Teria como carregar todas as linhas do arquivo (memória secundária) para a memória RAM (memória principal?)
  - Não!

Como ordenar arquivos grandes



Como ordenar arquivos grandes



- Qual tamanho do bloco seria mais adequado?
  - 10MB? 100MB? 200MB? 1GB? 2GB?
- Qual algoritmo de ordenação usar?
  - Troca e seleção? (i.e. *bubble* ou *selection*)
  - Divisão e conquista? (i.e. *merge* ou *quick*)

Algoritmos de ordenação externos

K-way merge sort

- Algoritmo de mesclagem que combina $\mathtt{k}$ listas ordenadas em uma única lista ordenada
  - O *merge sort* que vimos na aula anterior $k = 2$
- n é o total de elementos, e o tamanho da saída é a soma do tamanho das k listas. Assim, $k \leq n$

Algoritmo

Arquivo de 900MB usando blocos de tamanho 100MB na RAM

1. Leia 100 MB do arquivo, armazene na memória RAM e ordene (i.e. *quicksort*)
2. Salve em arquivo temporário o bloco ordenado no passo anterior
3. Repita os passos 1 e 2 até terminar o arquivo original ($900MB/100MB = 9$ blocos)
4. Leia os primeiro 10 MB de cada arquivo na RAM e reserve 90MB para *buffer* de saída ($9 \times 10MB$ de entrada $+90MB$ de saída)
5. Execute o algoritmo *9-way merge* sobre os 10MB de cada bloco e salve no *buffer* de saída. Quando terminar, descarregue o conteúdo do *buffer* no disco e repita o passo anterior até chegar no final de cada bloco

Aumentando um passo de mesclagem

Para melhorar eficiência no acesso ao disco

- O algoritmo anterior é composto por dois passos: ordenar e mesclar
- Se o arquivo for muito grande e o bloco pequeno, teremos um k muito grande, resultando em um algoritmo ineficiente
  - Usar dois passos de mesclagem seria uma alternativa melhor

Exemplo: Arquivo de 50GB e blocos de 100MB

1. Crie 500×100 blocos ordenados
2. Faça a primeira mesclagem combinando $25 \times 100MB$ blocos por vez, resultando em $20 \times 2.5GB$ blocos ordenados
3. Faça a segunda mesclagem combinando os $20 \times 2.5GB$ blocos em um arquivo de $50GB$ ordenado

Exercício 1

- Faça um programa que gere um arquivo texto com n linhas
- Cada linha deve armazenar um número inteiro de 0 até `INTMAX_MAX`[1]

```c
#include<stdio.h>
#include<stdint.h>
#include<limits.h>

int main(void){
    printf("Maior número int: %d\n", INT_MAX);          // 2147483647
    printf("Maior número int64_t: %ld\n", INTMAX_MAX); // 9223372036854775807
    return 0;
}
```

Código: Material de apoio

[1] O padrão C99 provê a biblioteca `stdint.h` que define macros e tipos conhecidos independente da arquitetura de máquina (32bits, 64bits, etc) Veja mais em https://cplusplus.com/reference/cstdint

Exercício 2

- Faça um programa que seja capaz de ordenar arquivos textos, com um número inteiro por linha, de tamanho arbitrário (i.e. `10GB`)

Curiosidades

- Código fonte[2] do pacote *GNU core utilities*, que contém o utilitário `sort`[3]
  - Faz uso do *k-way merge sort*
- Programação concorrente (*multithread*) podem tornar o processo de ordenação mais rápido, pois aproveitaria melhor os múltiplos núcleos dos processadores (CPU) modernos
  - Esse é um assunto que será abordado na disciplina de Sistemas Operacionais

[2] https://www.gnu.org/software/coreutils/#source
[3] https://man7.org/linux/man-pages/man1/sort.1.html

Leitura recomendada

Slides baseados em

- **External sorting**
  - https://en.wikipedia.org/wiki/External_sorting
- **K-way merge sort**
  - https://en.wikipedia.org/wiki/K-way_merge_algorithm