**T1: ALGORITMO DE ORDENAÇÃO EXTERNA**

jlsoares@inf.ufsm.br

# Introdução

Este trabalho tem por objetivo implementar um algoritmo de ordenação externa de textos grandes, com uma memória de hardware limitada. No meu algoritmo será utilizada linguagem de programação C++, e a técnica de divisão e conquista para manipular o arquivo e o método MergeSort para a ordenação interna dos itens com tamanho de memória definidos, e a Intercalação Balanceada de vários conjuntos ordenados de dados em arquivos para trabalhar com a relação de memórias interna e externa, gerando um arquivo de saída com as palavras ordenadas após todo o processo de ordenação.

# Algoritmo

## – Funcionamento

No trabalho temos o seguinte desafio ordenar um arquivo texto muito grande que não cabe em memória interna para a execução da ordenação do mesmo, sabendo disso o princípio do algoritmo baseia-se em ler sequencialmente o arquivo e dividi-lo em pequenos conjuntos e ordenar cada uma dessas partes isoladamente, definidas por um tamanho fixo no programa.

Após as ordenações, o conteúdo estará dividido entre os arquivos temporários chamados em meu programa de “run\_” e inicia-se o processo de intercalação entre os arquivos ‘run\_’ até gerar o arquivo de saída do qual o usuário nomeou.

O processo de divisão dos arquivos ocorre da seguinte forma: abre arquivo de entrada calcula o tamanho de cada palavra se cabe na memória armazena até completar o espaço em memória e chama a função para ordenar

Após as ordenações, o conteúdo estará dividido entre os arquivos temporários e inicia-se o processo de intercalação entre os arquivos, no meu processo de intercalação será usada a balanceada de vários caminhos na primeira execução teremos pequenos conjuntos ordenados divididos nos arquivos e se intercala os elementos de cada conjunto correspondente de cada arquivo para um único arquivo saída, assim esses conjuntos de cada arquivo se juntam e formam um conjunto maior de dados ordenados

Após todos os conjuntos serem intercalados e divididos entre os novos arquivos, os arquivos ‘run\_’ serão sobrescritos com a nova intercalação dos arquivos anteriores, e assim gerando conjuntos internos ordenados ainda maiores. Esse processo de intercalação entre esses dois grupos de arquivos temporários é recursivo, até que o conjunto gerado seja do mesmo tamanho do arquivo de entrada e fique em um único arquivo de saída.

Após o arquivo de saída ser gerado, todos os arquivos temporários usado no processo de intercalação são excluídos.

O número de ciclos de intercalação necessárias entre esses arquivos temporários é calculado previamente pelo programa com base no número de conjuntos ordenados em memória interna, para assim ter melhor controle das execuções.

**2.2 – Implementação**

Primeira etapa:

Logo no início do programa temos uma mensagem se o usuário deseja criar um arquivo genérico para ordenação ou ele deseja ordenar um arquivo próprio (sobre este arquivo ele deve estar na mesma localização do programa para poder ser aberto), caso o usuário deseje gerar um arquivo no próprio programa era feito um arquivo de tamanho dado, caso contrário o usuário dará o arquivo de entrada e nomeara o arquivo de saída sempre incluindo a extensão deles.

Segunda etapa:

Será pedido o tamanho de cada pagina a ser lido (tamanho de página é a mesma coisa que tamanho de memória que tem) este tamanho já vem definido como 4KB, mas também poderá ser modificado pelo próprio usuário desde que respeite as seguintes restrições: A memória não pode ser muito pequena se não ira criar e abrir muitos arquivo causando erro na execução.

Terceira etapa:

É pedido o número de blocos de memória (número de arquivos para a execução).

Quarta etapa:

É obtido o tamanho do arquivo de entrada e calcula o número de execuções necessária para separar o arquivo em tamanhos que não estourem a capacidade de memória definida pelo usuário. É também gerado os nomes dos arquivos que serão criados posteriormente.

Quinta etapa:

Lê o arquivo de entrada e o separa em arquivos de tamanhos correspondentes pelo de memória.

Sexta etapa:

Ordena os pequenos arquivos com o algoritmo merging e faz a intercalação destes arquivos ate gerar o arquivo de saída;

# Aplicações

Este método de ordenação tem inúmeras aplicações entre elas, melhorar a busca, seleção de dados em grandes arquivos, ordenar catálogos, banco de dados, registros e cadastros dentro de uma empresa, ordenar arquivos, melhorar o gerenciamento de dados.

**Referências**

https://en.wikipedia.org/wiki/External\_sorting

http://wiki.icmc.usp.br/images/1/1e/SCC0203-1o-2012-15.OrdenacaoExterna.pdf

http://www.decom.ufop.br/guilherme/BCC203/geral/ed2\_ordenacao-externa.pdf

http://www.showme.com/sh/?h=0a0OGGW

https://pt.wikipedia.org/wiki/Merge\_sort

http://www.cos.ufrj.br/~rfarias/cos121/aula\_07.html

https://www.youtube.com/watch?v=PKCMMSXQyJE

https://www.geeksforgeeks.org/merge-sort/