**Algoritmo de Ordenação Externa**

**Eduardo Mueller Nedel e Pedro Henrique Vestena Rossato**

emnedel@inf.ufsm.br e phrossato@inf.ufsm.br

1. **Introdução**

O algoritmo consiste em uma ordenação externa, ele foi implementando pensando que o usuário tem um limite de memória, sendo assim, o algoritmo irá tentar trabalhar com esta. Esta ordenação poderá ser feita de modo “rápido” mesmo com pouca memória RAM disponível, o que é um facilitador para empresas que utilizam de computadores antigos.

Em um contexto atual, este programa pode não ser muito útil, já que a maioria dos computadores tem bastante memória disponível, além disso, pode-se usar outros métodos de ordenação, que podem ser mais demorados, para contornar o problema da memória. Claro, que isto demandará mais processamento, porém, atualmente, os processadores têm grande capacidade.

1. **Algoritmo**

Falamos do algoritmo. Não esqueça das referências abaixo sobre fontes de onde tirou as informações. Olhe os exemplos.

O algoritmo foi feito na linguagem de programação C++. As bibliotecas iostream e ifstream necessárias para ler e escrever em um arquivo, string e vector para armazenar os dados, algorithm para utilizar o sort e a cmath para usar ceil(), que arredonda um numero para mais.

Ordem do algoritmo

{ Bibliotecas }

{ Funções : int num\_char (vector <string> palavras), verifica quantos caracteres contém em um vector de string}

Main :

Abertura do arquivo de entrada, caso este não exista, o programa será finalizado, caso contrário, seguirá as etapas a seguir.

Pergunta ao usuário, que deve informar quantos bits o programa poderá utilizar.

O conteúdo do txt é lido e separado em blocos (outros arquivos com no máximo x bits, digitados pelo usuário). Ele será separado usando um loop (até que o ponteiro do arquivo chegue ao fim deste) que irá pegando as palavras, armazenando na memória e as escrevendo, já em ordem de dicionário, em novos arquivos quando a memória quase estourar. Após isso, será verificado se o vector tem alguma palavra, e se tiver, esta será adicionada em arquivos.

Feita esta etapa de separação, chega a vez de reunirmos os arquivos novamente, o programa, através de dois for, irá ler e ordenar em um novo arquivo, dois arquivos, até que todos arquivos tenham sido ordenados. Isso será feito até que restem um arquivo, que irá conter a entrada ordenada em ordem de dicionário.

Após ordenado, colocamos a respostas em um arquivo denominado Respostas.txt, excluímos todos os arquivos temporários e terminamos o programa.

1. **Aplicações**

O programa, por exemplo, pode ser aplicado a sistemas de arquivos de fichário, o qual pode ser ordenado por nome, reunindo todos os nomes de fichas em um arquivo txt, facilitando a organização.

**Referências**

C++ Reference, http://www.cplusplus.com/reference/

Stack Overflow, https://pt.stackoverflow.com/

Marcos Kutova (2013) “Intercalação Balanceada”, http://www.showme.com/sh/?h=0a0OGGW

Linguagem C Programação Descomplicada (2014), “[ED] Aula 66 - Ordenação externa”, https://www.youtube.com/watch?v=sVGbj1zgvWQ

Guilherme Tavares de Assis, “ Ordenação Externa”, http://www.decom.ufop.br/guilherme/BCC203/geral/ed2\_ordenacao-externa.pdf