UNIVERSIDADE FEDERAL DO PIAUÍ - CAMPUS SHNB

SISTEMAS DE INFORMAÇÃO

ALGORITMOS E PROGRAMAÇÃO II

Rai Damasio Leal

José Mateus

Lucas Emanuel

Trabalho Prático

1 Introdução

Com a constante correria do dia a dia, muitas pessoas não possuem tempo para acompanhar notícias de diversos portais de comunicação. Diante desse cenário, os sumarizadores de texto se tornaram uma ferramenta útil para mostrar de forma sucinta e criativa o conteúdo de um texto, permitindo que o usuário rapidamente verifique se o conteúdo lhe interessa ou não. Com isso em mente, este trabalho prático tem como objetivo desenvolver um programa sumarizador de conteúdo básico, que funcione via console do computador.

O programa receberá como entrada um arquivo de texto (puro) ou o endereço de uma página web e exibirá na tela um sumário do conteúdo, que consiste na lista das n ***palavras*** mais frequentes, iniciando da palavra mais frequente para a menos frequente. Para implementação do programa, será utilizada a linguagem C e o código deverá ser compilado e executado em terminal do sistema Linux ou Windows.

A execução do programa será feita via linha de comando, com os parâmetros necessários para indicar o tipo de conteúdo, o caminho do arquivo com stopwords e o caminho do arquivo de texto de saída. Vale ressaltar que arquivo de stopwords contém palavras "irrelevantes" que devem ser desconsideradas pelo programa.

Por fim, a saída do programa deverá ser no formato palavra-valor, em ordem decrescente de frequência das palavras. Com isso, espera-se que este trabalho prático proporcione uma solução simples e eficaz para a visualização rápida do conteúdo de um texto, contribuindo para otimização do tempo das pessoas em seu dia a dia.

2 Solução Proposta

A solução que proponho é um programa sumarizador de conteúdo em linguagem C, que funcionará através do console do computador. O programa receberá como entrada um arquivo de texto sem formatação e apresentará na tela um sumário do conteúdo, que será uma lista das n palavras mais frequentes no texto, começando pela palavra mais frequente e terminando com a menos frequente. O usuário poderá especificar o valor de ‘n’ para definir o número de palavras a serem exibidas.
O programa também terá já definido um arquivo de texto contendo uma lista de stopwords, que são palavras irrelevantes que não devem ser consideradas na contagem da frequência das palavras. Dessa forma, o programa será capaz de ignorar palavras comuns, como "o", "a" e "de", que não adicionam informações importantes ao texto.
Por fim, o usuário também poderá especificar um caminho para um arquivo de texto de saída, onde o sumário será armazenado. A saída será apresentada no formato palavra-valor, onde cada palavra será seguida pelo número de vezes que ela aparece no texto, em ordem decrescente de frequência.

Em resumo, a solução proposta consiste em um programa sumarizador de conteúdo em linguagem C, que recebe um arquivo de texto de entrada, um arquivo de stopwords e um caminho para um arquivo de saída, e gera um sumário do conteúdo com as n palavras mais frequentes, ignorando as stopwords.

A organização do programa foi utilizado 4 funções para resolver o problema, onde cada uma se mostrou importante em sua área, são elas:

contar_ocorrencias: essa função é responsável pela formatação do texto, ela serve para contar o número de ocorrências do texto, recebe 3 parâmetros o primeiro é o arquivo onde o texto está localizado o segundo parâmetro é um vetor onde as palavras do arquivo está e o terceiro parâmetro é o tamanho do vetor.

Ordenar: essa função é responsável por ordenar o texto em ordem decrescente, recebe 3 parâmetros, o primeiro é um vetor onde está armazenado as ocorrências das palavras o segundo parâmetro está o tamanho do vetor de palavras e o terceiro parâmetro é o próprio vetor de palavras.

ve_quantidade: esse função é responsável por imprimir em um arquivo o sumário de palavras final, recebe 4 parametros, o primeiro é o arquivo onde deve ser armazenado o texto, o segundo é o vetor onde esta as palavras, o terceiro é a quantidade de palavras que o usuário deseja salvar e por fim o último parametro é o vetor onde está armazenado as ocorrencias das palavras.

descopactar_stopwords: essa função é responsável por tirar as palavras stopwords e colocar em um vetor para ser comparadas com o vetor de palavras ela não recebe parametro, é chamada em dentro da função de contar_ocorrências para ser comparadas com o texto de palavras.

Primeiramente utilizamos a diretiva #define, seguidas por variáveis globais, protótipos de funções, a manipulação de arquivos, vetores, matrizes, e apontadores de memória (Ponteiros). Utilizamos o método bolha na ordenação das frases.

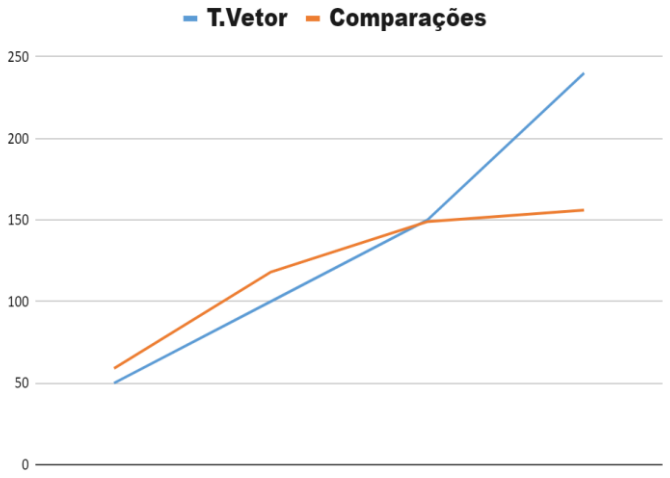
3 Análise de Desempenho

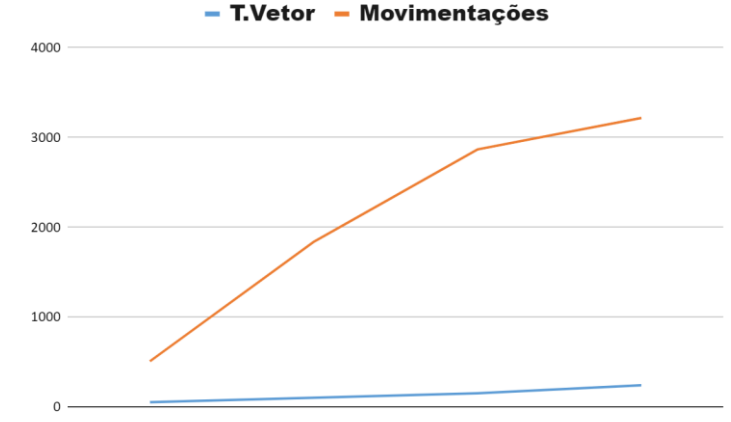
Nesta seção apresentamos uma análise detalhada do desempenho de nosso código e destacamos os principais indicadores de desempenho que foram avaliados.

A função analisada foi a de ordenação, pois é a função que é executada mais vezes no algoritmo. O método escolhido foi o de ordenação da inserção que teve como objetivo avaliar a

quantidade de comparações e movimentações que o algoritmo fez para ordenar o sumário de palavras como mostra o gráfico a seguir.

Os indicadores utilizados foram o tamanho do vetor de palavras, o valor total de comparações e o valor total de movimentações. Cada um desses indicadores é importante para avaliar o desempenho geral do algoritmo.





I

A partir do gráfico foi percebido que, o número de movimentações é superior ao de comparações, ou seja, para ordenar as palavras pelo número de ocorrências em ordem decrescente o algoritmo faz mais movimentações do que comparações.

Outro ponto analisado foi qual função do código mais se repete, pelas análises feitas foi concluído que todas as funções usadas só se repetem apenas uma vez a cada execução. O trecho que mais se repete é quando o arquivo que contém o texto abre e é passado para um vetor pois é passado palavra por palavra.

Algumas limitações do código que foram encontradas foi que palavras acentuadas não são exibidas corretamente no arquivo. O código não tem um controle adequado de erros para abrir ou fechar arquivos, o que pode levar a erros se houver problemas ao acessar ou gravar em arquivos.

Em resumo, a análise de desempenho apresentada mostrou que o algoritmo de ordenação da inserção utilizado para ordenar o sumário de palavras pode apresentar um bom desempenho,

principalmente em relação ao número de comparações realizadas. Além disso, foram identificadas limitações relacionadas à leitura de arquivos, tratamento de erros e exibição de caracteres especiais. É importante considerar outras alternativas de algoritmos de ordenação para melhorar o desempenho do código e incorporar um tratamento de erros robusto para garantir a estabilidade e confiabilidade do programa.

4 Conclusão

Por conclusão, foi proposto um sumarizador de textos, sendo passados 5 parâmetros para entrada no próprio sistema operacional, os mesmos foram divididos em, arquivos de texto puro, o caminho para o mesmo, o tamanho, o caminho para o arquivo de stopwords, o caminho para a saída, em suma o programa receberá como entrada um conteúdo de texto e exibirá na tela um sumário do conteúdo. A solução proposta para a resolução do trabalho primeiramente foi dividir os parâmetros em funções para serem mais fáceis de se manipular, em seguida salvamos os stopwords em um arquivo .txt comum, onde dentro da função descompactar_stopwords() lemos esses arquivos com as stopwords e armazenamos em um vetor de strings 'Palavras_stopwords', e o objetivo da função é verificar se todas as palavras foram lidas ou o vetor 'palavras_stopwords' chegaram ao tamanho máximo. Falando por alto um pouco das outras funções implementadas, a função ordernar(), utilizou o método inserção na hora da ordenação, ordenação essa decrescente. Em seguida a função contar_ocorrencias(), conta as ocorrências de uma palavra, onde nela verificamos se é ou não stopwords, ou se ela já foi contabilizada antes.

E por conclusão desse tópico a função ve_quantidade(), se responsabiliza por imprimir a palavra e a ocorrência, ou seja, quantas vezes ela se repetiu ao decorrer do arquivo.

Por aspectos positivos, tivemos êxito em 90% do que foi solicitado, devemos entrelaçar isso com os aspectos negativos, não conseguimos chamá-las dentro do powershell, sugerida pelo professor.

Em paralelo a isso, em um futuro projeto, poderíamos habilitar a visibilidade pelo power shell, e a criação de um website para a visualização do mesmo.