MÁQUINA DE BUSCA

Bruno Rafael Martins Alves

João Vítor David Prates

Matheus Cascalho dos Santos

**Introdução**

Máquina de Busca é um software cujo funcionamento se baseia em uma interação indireta entre um usuário e um determinado banco de dados o qual, possui documentos diversos que podem ser acessados pelo usuário através de uma busca realizada por ele no software.

O intuito de Máquina de Busca é entregar ao usuário o documento que mais se aproxima da pesquisa realizada pelo mesmo. Cada palavra nos documentos do banco de dados é lida e depois guardada na forma de um vetor, de modo que, a direção e o módulo dependem de fatores como a quantidade de repetições em cada documento e a quantidade de documentos nos quais essa aparece. Esses vetores são determinantes para o funcionamento do software já que, esse realiza diversos cálculos com base nos vetores, e criam uma ordem de importância de cada palavra em cada arquivo para determinar quais palavras representam melhor cada documento.

**Implementação**

Máquina de Busca tem como característica o fato de ser muito modularizado, sendo cada módulo crucial para o funcionamento do mesmo.

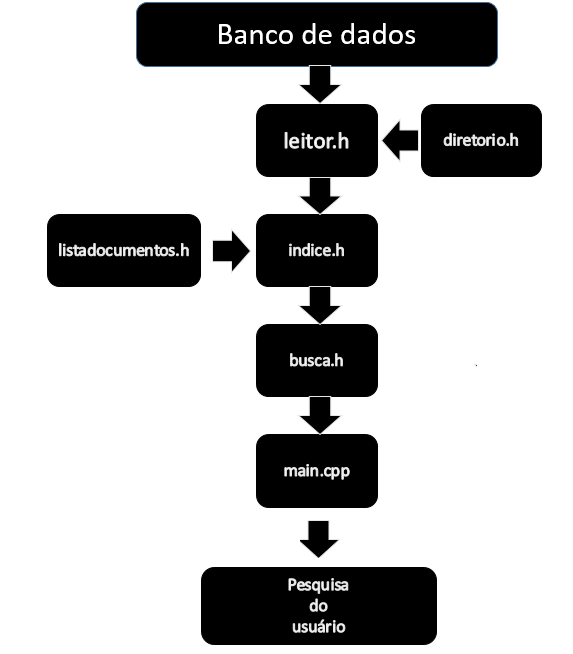
O software possui uma hierarquia que deve ser respeitada, iniciando-se com o módulo leitor.h e diretório.h. A função destes, de modo geral, é basicamente, ler os arquivos presentes no banco de dados e listar o conteúdo destes.

Ao prosseguir com o programa, os próximos módulos a serem utilizados são listadedocumentos.h e indice.h. O objetivo da união deles é criar um índice invertido, esse que tem como objetivo catalogar, em cada palavra do banco de dados, em quais documentos ela se fez presente, e quantas vezes se fez presente.

Feito o índice invertido, o próximo passo será entrar no módulo busca.h, cuja função é transformar cada palavra em um vetor o qual o módulo e a direção dependeriam, além de outros fatores, dos elementos do índice invertido. Com esses vetores, e a busca feita pelo usuário, podemos ranquear quais documentos são melhores representados pela palavra buscada.

O usuário do programa não tem acesso a nenhum desses passos, mas sim do que será executável no módulo main.cpp, onde ele poderá digitar sua busca e terá como retorno, apenas a ordem dos documentos representados por essa palavra.

Para uma melhor explanação da posição de cada módulo, podemos conferir o organograma abaixo, em que as setas representam a relação de dependência entre os elementos que compõe Máquina de Busca.



**diretorio.h**

Ao ser inicializado, um dos primeiros módulos a serem utilizados é o diretorio.h cujo principal objetivo é concatenar nomes de arquivos e seus respectivos diretórios para facilitar uma posterior leitura do banco de dados que acontecerá em um módulo diferente desse. O principal conteúdo de diretorio.h é a classe denominada Diretorio cujas variáveis representam os nomes dos documentos, seus respectivos diretórios e a união de ambas as coisas, sendo todas essas armazenadas em forma de strings. Além disso, a classe conta com métodos que permitem e facilitam a manipulação dessas variáveis.

O software possui um diretório padrão, que pode ser alterado, em que entrega o endereço de um arquivo texto denominado sumário o qual apresenta, como conteúdo, os nomes de todos os outros arquivos presentes no banco de dados. Esse sumário, para ser considerado padrão deve estar em uma pasta com o nome Diretorio no mesmo diretório do software.

**leitor.h**

O único módulo, diferente do main.cpp, que não apresenta uma classe específica é leitor.h. O objetivo deste é, intuitivamente, ler arquivos do banco de dados.

Para obter êxito na utilização desse módulo, é necessário entender bem a classe Diretorio, já que, leitor.h, em suas funções, utiliza-se e manipula bastante os diretórios. Em geral, a leitura desses altera listas, passadas comumente como parâmetro das funções, cujo conteúdo são, ora palavras lidas de um arquivo para uma posterior utilização em outros módulos, ora o nome dos arquivos do banco de dados lidos do sumário.

O leitor.h possui também a função transformaString que difere totalmente das demais funções presentes no módulo. O objetivo dessa função é padronizar as palavras que são lidas.

**listdocumentos.h**

O módulo listdocumentos.h possui uma classe chamada ListDocumentos cujo principal objetivo é, para cada documento, guardar, em forma de contador, a quantidade de vezes que uma determinada palavra apareceu no conteúdo do documento.

A classe ListDocumentos, apesar de muito importante para o funcionamento do software, atua como um complemento para a classe Indice.

Há o método numTotal que difere dos demais métodos da classe pelo motivo que ele não se aplica somente a um documento. O objetivo desse é retornar a quantidade de vezes que determinada palavra aparece em todo o banco de dados.

**indice.h**

O módulo indice.h possui como principal conteúdo a classe Indice. A principal função da classe é criar e guardar o índice invertido, sendo esse um map que recebe a palavra e a lista de documentos na qual essa palavra apareceu. O índice invertido será de suma importância para o software principalmente na parte do cálculo para ranquear as palavras que mais representam determinado documento.

Para utilizar a classe Indice, é imprescindível entender o funcionamento de ListDocumentos e de leitor.h. ListDocumentos está contido como um dos tipos construtores do índice invertido, sendo ele o responsável por determinar em quais documentos a palavra buscada aparece. Já o leitor.h possui funções que podem alterar a variável todosDocumentos\_ presente na classe Indice. Além disso, a função transformaString é elemento fundamental para padronizar as palavras e impedir erros no índice invertido.

**busca.h**

O módulo busca.h é o módulo final que o software vai utilizar. Ele possui uma classe denominada Busca que possui diversas funções.

A primeira função da classe é receber a busca feita pelo usuário e retorná-la como um vetor de modo que, quando forem feitos os cálculos para ranquear as palavras, todos os vetores já estejam formados.

A segunda função da classe é realizar os cálculos para o ranking cosseno. Para isso, a classe utiliza alguns métodos que geram valores de frequência do termo nos documentos, a importância do termo para cada documento, as coordenadas, seja do documento, seja da expressão de busca e a similaridade entre elas.

A terceira função da classe é retornar, ordenadamente, quais documentos são melhores representados pela busca feita pelo usuário.

**main.cpp**

**Conclusão**