Universidade Federal de Itajubá - UNIFEI - Campus Itabira
Engenharia de Computação
Estrutura de Dados II - ECO029

ANDRÉ 'ZANV' VIANA - 25037
LUCAS SAMUEL VIEIRA - 24072
WILLIAM A. PRADO - 24015

RELATÓRIO - TRABALHO PRÁTICO I - MÁQUINA DE BUSCA

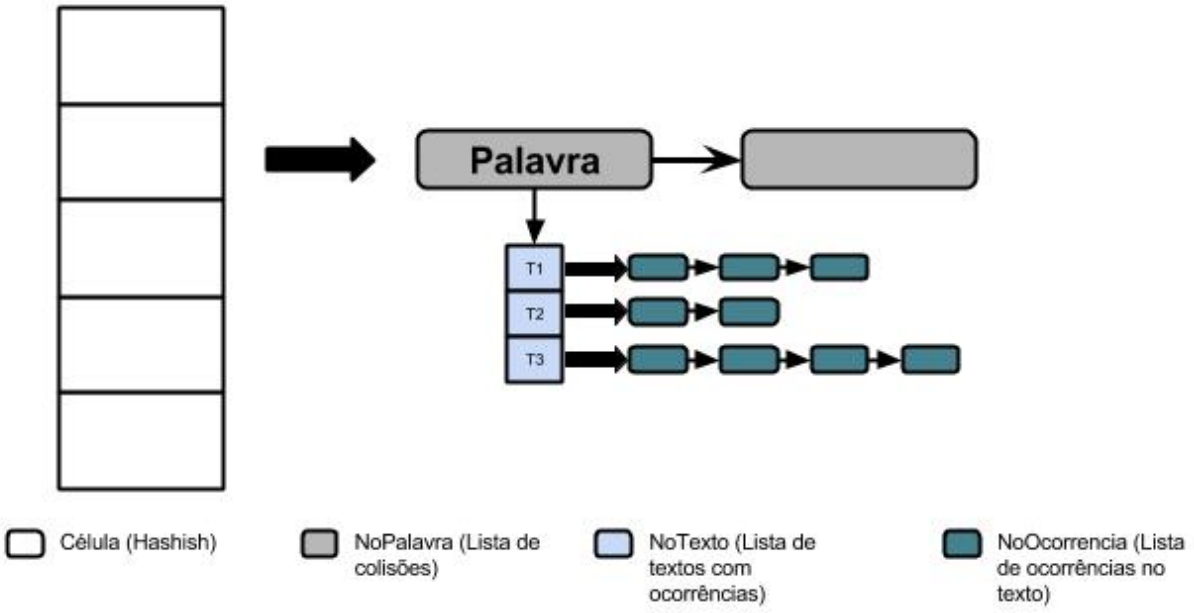
ITABIRA
2014

Descrição do trabalho

O projeto teve como objetivo a criação de um programa capaz de realizar pesquisas por palavras e frases em arquivos de texto pré determinados, usando para o tal o método de Hashing, com tratamento de colisão. As palavras e frases encontradas na busca devem são impressas juntamente com o nome do arquivo em que se encontram e o contexto.

Estruturas de Dados Utilizadas

Para a realização das pesquisas, é utilizado o método de Hashing, com listas dinâmicas para o tratamento de colisão. A estrutura completa possui um vetor [Hashish] de listas cujos nós [NoPalavra] contém sub-listas de nós de arquivo [NoTexto] que por sua vez contém mais uma série de sub-listas, cujos nós [NoOcorrencia] marcam as posições das ocorrências de determinada palavra em determinado arquivo.



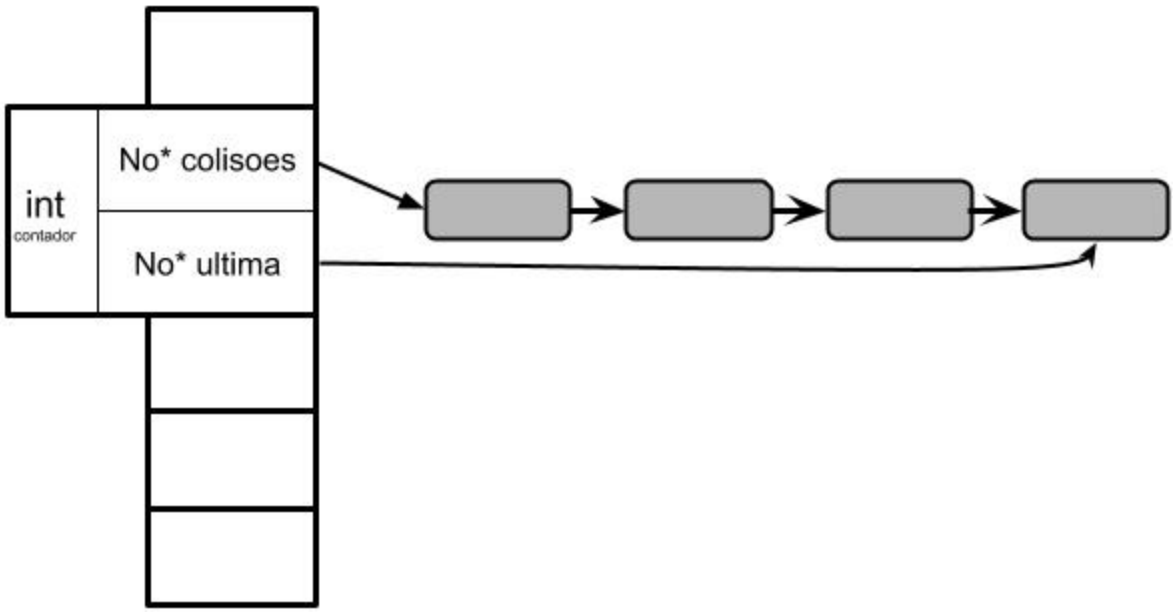
Vetor Hash (Hashish)

Para o Hashing, é utilizado um vetor de 10^3 posições. O índice de cada palavra é definido pela função:

$$\text{Índice} = \left(\sum_{i=1}^{n}(c_i * 2^i)\right) \mod 10^3$$

Onde ‘n’ é o número de caracteres na string, ‘i’ é a posição do caractere a ser processado incrementada em 1, e ‘c’, o valor int do caractere a ser processado.

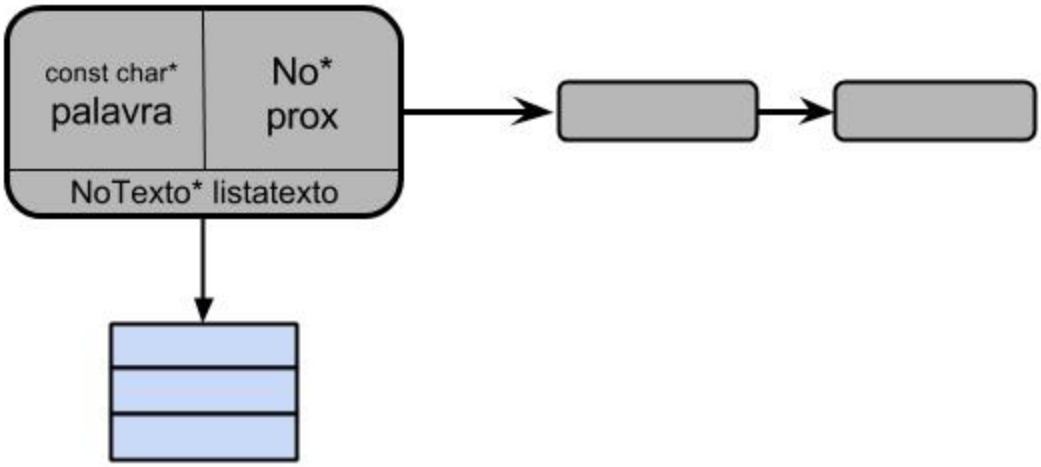
Cada célula do vetor possui dois ponteiros e um contador. O primeiro dos ponteiros aponta para o primeiro nó da lista de colisões, o segundo, para o último nó desta mesma lista; o contador se mantém atualizado com o número de colisões no índice.



Listas

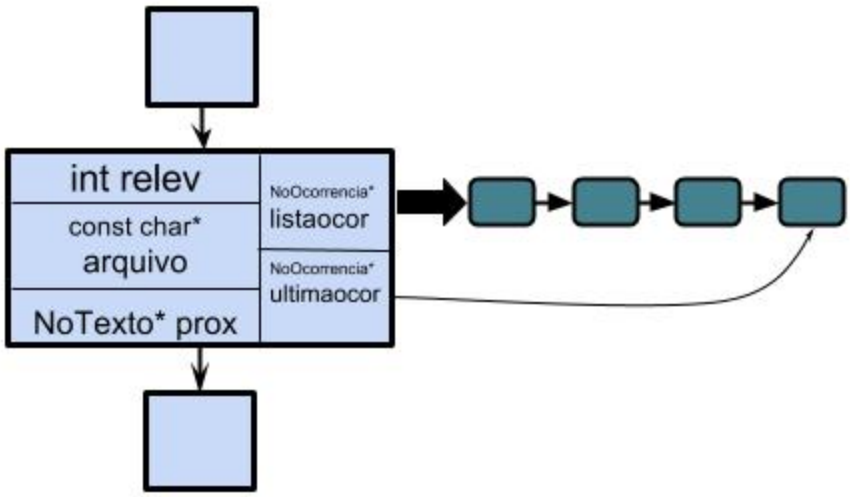
Lista de Colisões

A lista usada no tratamento de colisões é dinâmica e seus nós usam como chave as palavras cujas ocorrências estão marcadas. Cada nó desta lista contém um ponteiro para o próximo nó, um ponteiro para a lista de textos nos quais há ocorrências da palavra, e uma string contendo a palavra.



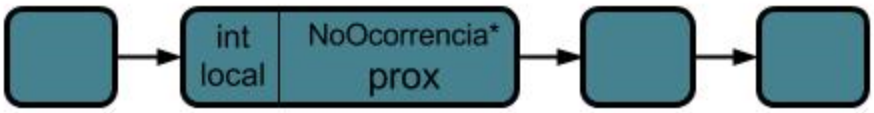
Sub-lista de Textos

A sub-lista de textos encontra-se sob a lista de colisões, e contém todos os textos nos quais há ao menos uma ocorrência da palavra em questão. Cada nó desta lista contém um ponteiro para o próximo nó, um ponteiro para a lista de ocorrências, um ponteiro auxiliar para a última ocorrência, a ser usado em adições, e uma string contendo o nome do arquivo de texto, e um contador de relevância.



Sub-lista de Ocorrências

Por fim, a sub-lista de ocorrências, sob a sub-lista de textos, contém a posição de cada ocorrência da palavra em questão no texto em questão. Cada nó da lista possui apenas um ponteiro para o próximo nó e uma variável int que marca a quantos caracteres há entre o começo do texto e a ocorrência.



Implementação

A implementação do programa foi dividida em três partes: Text Pool, Hash e Pesquisa.

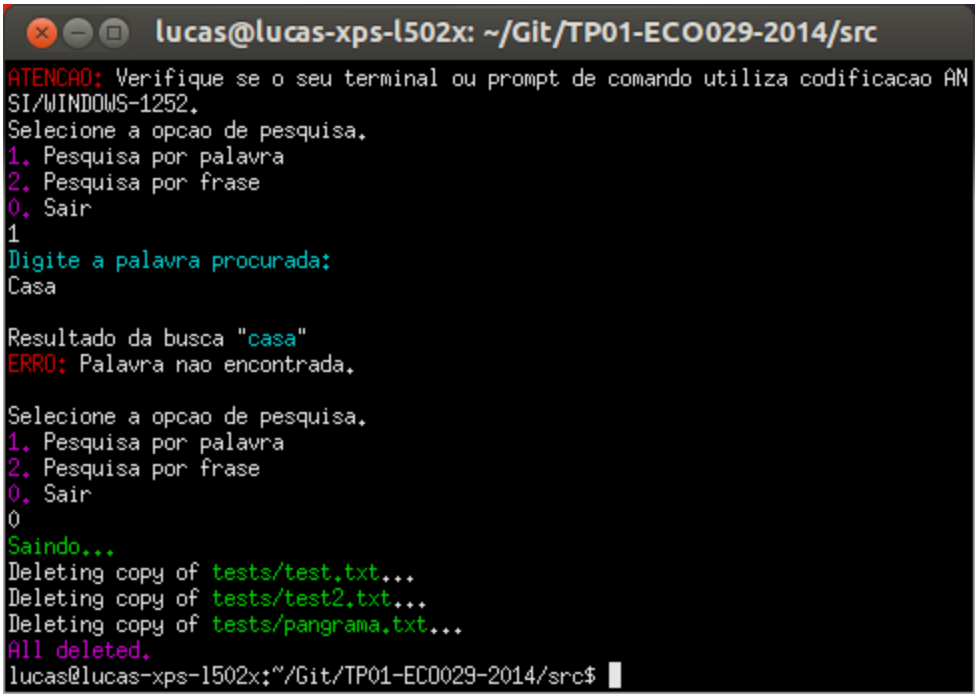
A classe Text Pool guarda os textos indicados pelo arquivos robots.txt, nos quais ocorrerão as pesquisas. Suas principais funções são usadas para encontrar, processar, guardar e resgatar os textos a serem usados.

Sob Hash, econtram-se as estruturas utilizadas no programa e as funções que preenchem e navegam por tais estruturas. O vetor Hashish e suas listas e sub-listas são preenchidos com as palavras e ocorrências dos textos durante a inicialização do programa.

Por fim, sob Pesquisa encontram-se as funções responsáveis pela interface e controle do programa; notavelmente as funções responsáveis por imprimir os resultados da pesquisa.

Testes

Busca por Palavra



Conclusão

Ao fim do projeto, foi concluído que máquinas de busca simples são menos complexas do que normalmente se espera, porém demandam consideravel planejamento para que funcionem de forma correta e estável.