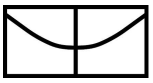

Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Projeto de Disciplina

Projeto e Análise de Algoritmos

Brasília, DF
2021

1 - Integrantes

- Carlos Eduardo Taborda Lottermann - 18/004159
- Gabriel Delolmo Erhardt - 17/0142612
- Guilherme Braga Pinto - 17/0162290
- Rafael - 17/0043827

2 - Tecnologias usadas

No momento, cogita-se usar Python usando Scrapy para implementar o web crawler e Ruby on Rails para criar o servidor com o qual o usuário irá interagir e que irá acessar o mecanismo de busca.

O indexador e buscador será feito usando C++, até o momento não se planeja usar nenhuma biblioteca para eles.

3 - Algoritmos

Um std::unordered_map da STL do C++ será usado para relacionar índices a links dos sites. Sites são adicionados uma única vez, sendo feita uma verificação no mapa antes da indexação.

Um std::unordered_map será usado para auxiliar na indexação, mapeando palavras em listas de índices de sites que as contêm. A quantidade de elementos no mapa será denotada por N e a quantidade de elementos numa lista será dada por L.

Indexer

O algoritmo indexador irá listar as palavras usadas em cada página (talvez com um limite mínimo de letras para que a palavra seja considerada) e irá adicionar a página no final das listas referentes a cada palavra encontrada - criando-as quando necessário.

A complexidade de inserção do mapa usado é em média constante, mas pode chegar a ser linear na quantidade de elementos no mapa. A inserção na lista é constante.

No pior caso, todas as P palavras lidas serão inseridas em listas novas que serão inseridas no mapa, com complexidade N referente à quantidade de listas já presentes no mapa. Logo a inserção será P*N.

Searcher

O algoritmo de pesquisa para uma palavra irá recuperar a lista referente à palavra.

O custo de encontrar um elemento no mapa é constante na média e no pior caso linear na quantidade de elementos no mapa. Logo a complexidade de encontrar uma lista de sites que contêm uma palavra é constante na média e N no pior caso.

Para pesquisas de mais palavras, as listas serão recuperadas e serão feitos tratamentos para combiná-las.

No caso de uma pesquisa OR basta juntar as listas, ação com complexidade constante resulta em complexidade constante na média e N no pior caso.

No caso de pesquisas usando AND ou exclusão de palavras com ‘-’, será necessário buscar as listas e então iterar sobre elas. Como as listas são ordenadas por ordem de chegada a complexidade é linear na soma dos tamanhos das listas em ambos os casos: itera-se sobre as listas até serem encontrados elementos iguais que serão considerados ou excluídos da pesquisa.

A complexidade dessa operação será a soma do tamanho das listas referentes às palavras usadas. A pesquisa com AND e ‘-’ resulta, então, em complexidade 2*N + 2*L.

No caso de pesquisas por strings, será feita uma pesquisa AND usando as palavras da string, seguida por uma verificação se a string está presente de fato na página. O custo dessa verificação dependerá da quantidade de palavras P contidas no site. Para uma string com S palavras a complexidade será, então, S*N (busca das listas) + 2*L*(S-1) (iterações sobre as listas) + S*N*P (verificação se string exata existe). Onde o pior caso se dá quando as listas referentes a cada palavra tem os mesmos elementos.