Projeto: Search Engine

Cristiana Nogueira, Hanna Rodrigues e Marcos Antonio

30 de abril de 2020

Planejamento de atividades

- Detalhamento do projeto;
- Pré-processamento;
- Implementação da estrutura de dados e algoritmos de pesquisa;
- Interface Web (desafio extra).

Pré processamento do Corpus

- Limpeza dos dados, verificação de erros nos arquivos, tokenização;
- Correção de erros nas tags html;
- Organização dos textos de acordo com a estrutura pensada;

Estrutura de Dados : Trie

- Também é conhecida como árvore de prefixos;
- Cada nó é um caractere do alfabeto escolhido que pode ser por exemplo o alfabeto ASCII;
- Cada caminho da árvore percorrendo da raiz à uma folha representa uma string diferente.

Estrutura de Dados : Trie

- Dadas as definições do que é uma trie, quais as vantagens sobre outras opções de estruturas de dados?
- O número maxímo de filhos de um nó é igual ao tamanho do alfabeto.
- O tempo que precisamos para procurar uma palavra é proporcional ao tamanho da palavra, não importa quantas palavras existem na árvore;
- O número de nós visitados para buscar ou inserir uma chave de comprimento W em uma trie é no máximo 1 + W.

Execução da busca

- Busca apenas a 1º ocorrência no texto, já que não estamos fazendo um ranking de relevância das páginas;
- Como funcionalidade adicional será implementada sugestão de palavras. Podemos fazer uma análise ortográfica na palavra ou conjunto de palavras buscadas, de modo que se houver um erro de escrita podemos sugerir palavras próximas;
- Uma das opções é aplicar um algoritmo de distância e retornar a busca para a palavra sugerida. A distância Levenshtein ou distância de edição pode ser uma implementação interessante para o caso em que o erro se encontra no início da palavra. Assim, sugerimos a palavra com distância miníma;
- Além disso, também queremos que a busca seja ordenada pelo título dos textos.

Ideias para otimização do tempo de execução e memória

Uma vez que todas as strings presentes no corpus são inseridas na árvore, é possível fazer uma compactação usando um tipo de trie chamada Radix-trie.
A radix-trie é uma solução de otimização de memória para a árvore de prefixos, a qual condensa em um mesmo nó partes em comum de um prefixo;

Serialização

O Serialização é o processo de fazer uma tradução de uma estrutura de dados para um formato que pode ser armazenado ou transmitido e resconstruído novamente em um ambiente computacional diferente. Assim, é possível salvar o projeto processado para que ele seja reconstruído em novo ambiente sem que seja preciso refazer todo o processamento.

Bibliotecas auxiliares

- Fstream (leitura de arquivos),
- iomanip (formatação de saída),
- Sugestão: Django (Framework para desenvolvimento web com python);
- Boost (conexão com python e serialização)

Referências

- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-851-advanced-data-structures-spring-2012/lecture-videos/session-16-strings/
- https://www.ime.usp.br/ pf/estruturas-de-dados/aulas/tries.html
- https://en.wikipedia.org/wiki/Serialization
- `https://en.wikipedia.org/wiki/Radix_tree`
- http://people.cs.ksu.edu/ rhowell/DataStructures/trees/tries/intro.html

Obrigada!
Para dúvidas, sugestões:

Cristiana: cristiana.couto@fgv.edu.br

Hanna: hanna.rodrigues.ismart@gmail.com

Marcos: marcosantonioalves358@gmail.com

Repositório do projeto:
`https://github.com/Cristiananc/Search_Engine`