Documentação EP4 -Localização de Palavras II

Fellipe Souto Sampaio[1]

MAC 0323 Estrutura de Dados
Prof. Dr. Yoshiharu Kohayakawa

Instituto de Matemática e Estatística - IME USP
Rua do Matão 1010
05311-970 Cidade Universitária, São Paulo - SP

[1]Número USP: 7990422 e-mail: fellipe.sampaio@usp.com

1 Introdução

Este relatório serve como descrição para o exercício programa 4 e a forma como sua solução foi implementada. O algorítmo utilizado no EP4 para implementação das tabelas de hashing foi baseado nas notas de aula do professor Yoshiharu e nas notas online do professor Sedgewick.

2 client

Este é o arquivo principal do programa, o cliente. Através dele são feitas as chamadas para os outros módulos que constroem as tabelas e buscam as palavras no texto.

2.1 int main

Parâmetros de entrada t: -farquivodeentrada.txt.
Saída: 0 se a execução ocorrer sem falha.
O programa recebe como entrada o nome do livro processado pelo CoreNLP e o abre. No main é inicializada as tabelas que implementam o funcionamento do programa. Após a leitura de todas palavras o usuário pode entrar com palavras para realizar buscas, as seguintes opções foram implementadas

-e Imprime todas as sentenças que ocorrem a palavra λ.

-ev Imprime todas as sentenças que ocorrem a palavra λ, o número da sentença e a quantidade de tokens.

-a Imprime todas as sentenças que ocorrem o lema λ e todas suas variações.

-av Imprime todas as sentenças que ocorrem o lema λ e todas suas variações, juntamente com o número da sentença e a quantidade de tokens.

-aV Imprime as sentenças notadas de um determinado lema λ.

3 interface

Esta interface funciona como ponte entre o cliente, as tabelas de hashing e o livro processado, por meio dele o cliente pode processar um livro e procurar palavras nas tableas, recebendo como produto final a localização de palavras e lemas ao longo do texto.

4 readBook

Interface que processa o livro fornecido pelo usuário. Cada sentença anotada é separada e processada individualmente, como se fosse uma linha individual. Dentro dessa linha separa-se a senteça, os tokens e o número da senteças, que são armazenados na estrutura de dados bookSentence. O restante da senteça anotada (informação gerada pelo coreNLP) é filtrado pela função do C strtok que transforma cada item fechado por chaves da sentença em um token individual, fornecendo como saída um par palavra e lema. Esse par é enviado para ambas as tabelas para que sejam inseridos. Ao fim dessas operações alguns ponteiros para o buffer de dados são criados, para fácil impressão da senteça notada.

5 handlingWord e handlingLemma

São duas estruturas de dados que fazem a comunicação entre o módulo interface e a implementação das tabelas de Hashing. Através destas interfaces inserir novos itens em qualquer uma das tabelas é possível sem que seja preciso um prévio conhecimento de qual das duas resoluções está sendo empregada. Itens repetidos tem um tratamento especial , para que não haja ambiguidade na busca.

6 Funções de Hashing

Neste EP temos 4 tabelas de hashing, implementadas de duas formas diferentes:

- wordEnc e lemmaEnc, tabelas de hashing dinâmicas com resolução de colisão por encadeamento.
- wordProb e lemmaProb, tabelas de hashing dinâmicas com resolução de colisão por probing linear.

As tabelas se baseiam na teoria vista em aula, a maior diferença reside na inserção e no tráfego através dos elementos da tabela, em sumo há uma pequena diferença na estrutura de dados de word e lemma. Estas estruturas são as mesmas utilizadas no EP3, EnglishWord e EnglishLemma, para palavras e lemas respectivamente. Os campos e as diferenças são as seguintes:

word - A palavra ou lema, dependendo da estrutura.

sentenceArray - Em Englishword guarda uma copia do lema da palavra word, em Englishlemma guarda um array de strings nos quais todas são derivações do lema.

sentenceLocal - Em Englishword guarda o número da sentença em que word aparece e na primeira posição a quantidade de elementos que existem no vetor. Em Englishlemma guarda a quantidade de elementros que existem em sentenceArray.

As estruturas de dados tem funcionamento independente de qual tipo de hashing está sendo aplicado, e para isso estão declaradas dentro do arquivo header *wordDefs* e *lemmaDefs*. Semelhante abstração acontece em **fundamentalHash**, função que encapsula algumas funções vitais e comuns para todas as quatro tabelas criadas. Isso permitiu uma grande independência entre cliente-interface, facilitando a modelagem do problema. Ainda em **fundamentalHash** está definida a função de hashing utilizada por todas as tabelas e também uma tabela de primos até $2^{32} - 1$, útil na manipulação do hashing dinâmico e controle do fator de carregamento.