Algoritmo Função de Espalhamento

Gabriela Gomes dos Santos e Pedro Felipe Gonçalves de Arruda

February 2, 2017

Abstract

Este trabalho tem por objetivo apresentar um algoritmo, que, gera uma imagem em formato ppm, na qual demonstra a diferença entre o espalhamento de duas funções hash distintas.

1 Introdução

No quarto semestre de Engenharia da Computação, nos foi proposto o seguinte exercício: "Utilizando a biblioteca hash.h, desenvolvida em sala de aula, crie um algoritmo, no qual, gere um vetor de 1024 posições, através deste, crie uma imagem ppm que demonstre o espalhamento de duas funções, ou seja, uma matriz 32X32. Para testar a função, o programa deve abrir dois arquivos txt diferentes, uma lista de palavras em português e a outra em inglês. Apresentar a diferença entre as funções e listas através das imagens geradas."

Utilizamos como base o conceito de lista através da biblioteca hash, uma estrutura não indexada.

As listas de palavras foram inseridas no programa separadamente, palavra por palavra. Cada palavra foi enviada para a biblioteca hash, na qual contém uma função de espalhamento, que determina sua posição no vetor.

Ao calcular a posição do vetor, na qual será inserida a palavra, pode ocorrer uma colisão, ou seja, diferentes palavras podem ocupar a mesma posição.

Nosso programa é escrito na linguagem C, contendo quatro funções principais. Vetor: abre o arquivo .txt e monta o vetor, Max: procura o maior valor do vetor, graphicGerationF1: gera a imagem, Color: gera a cor.

2 Criação da Imagem

2.1 Função vetor

Para a criação da imagem o programa primeiramente cria um vetor que será utilizado para identificar as colisões.

Isso é feito pela função vetor, que inicializa o hash da biblioteca. Em seguida abre o arquivo com as palavras e manda-as para a biblioteca uma a uma juntamente com a função de espalhamento, que são representadas pelos os números 1 e 2, sendo 1 a função com melhor espalhamento.

```
void vetor(char filename[SIZE_CHAR],int option){
        int i=0;

        dictionaryHashLinked hash;
        dInit(&hash);

        FILE *file = fopen(filename, "r");

        if(file == NULL){
                printf("Error file doesn't exist\n");
        }else{
                char word[SIZE_CHAR];
                while(fgets(word, SIZE_CHAR, file) != NULL){
                        i = put(&hash, word, i, option);
                        vetorP1[i]++;
                }
        }
        fclose(file);

}
```

2.2 Função max

Após a criação do vetor, é usado a função Max que varre o vetor em busca do maior do valor, que será usado como base para o valor da cor da imagem.

```
  void max(){
        int i=0;
        maxV = vetorP1[0];

        for(i = 1; i< MAX_VETOR; i++){
                if(maxV < vetorP1[i])
                    maxV = vetorP1[i];
        }
}
```

2.3 Função graphicGeration

Em seguida a função graphicGeration que em a imagem com base no vetor montado, sendo que para o nível das cores usa outra função.

```
  void graphicGeration(char imagename[SIZE_CHAR]){        // gera a imagem
        int cor[PIXEL];
        int i, j, k, x, y, z, b;
        char name[256];
        k=0;


    strcpy(name,imagename);
    strcat(name,".ppm\0");
```

```
        FILE *ppm = fopen(name, "w+"); //cria arquivo
        if(ppm == NULL){
                printf("Error create file\n");

        }else{
                fprintf(ppm, "%s\n","P3" );
                fprintf(ppm, "%d  %d\n", PIXEL * MULTIPLI, PIXEL * MULTIPLI);
                fprintf(ppm, "%d\n",255);

                for(i = 0; i < PIXEL; i++){
                        for (j = 0; j < PIXEL; j++, k++){
                                cor[j] = Color(vetorP1[k]);        // cor
                        }
                        for(x = 0; x < MULTIPLI; x++){
                                for (y = 0; y < PIXEL; y++){
                                        for (int z = 0; z < MULTIPLI; z++){
                                        fprintf(ppm, "%d  %d  %d ", 255, cor[y], 255);
                                        }
                                }
                                fprintf(ppm, "\n");
                        }
                }
         }
         fclose(ppm);
}
```

2.4 Função Color

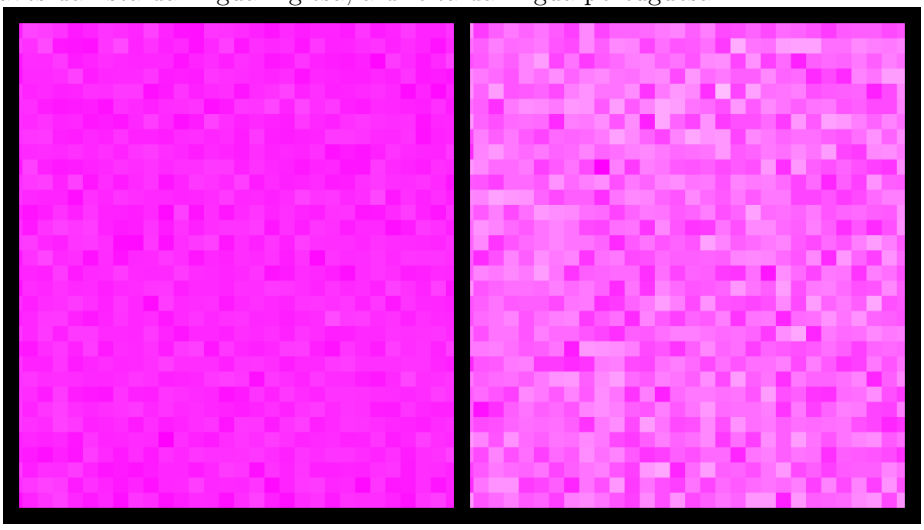
A função color recebe o valor do vetor e calcula o valor respectivo da sua cor

```
        int Color(int vValor){                // cor
            return 255 -  ((255*vValor)/maxV);
        }
```

3 Diferença de Espalhamento

A seguir estão as imagens geradas pelo nosso programa. Primeiramente utilizando a função 1 = key[i]*((i+3)/(2+i)), onde i varia de acordo com a letra da palavra atual. A imagem da esquerda demonstra o espalhamento da lista de palavras em inglês, a direita da lista de palavras em português.

A segunda função é = key[i]*(i+1) ,foi criada com o objetivo de melhorar o espalhamento, pois na função 1 as colisões ficaram concentradas em lugares diferentes, como foi possível observar através das imagens anteriores. Foram geradas as seguintes imagens, utilizando a nova função. A esquerda gerada através da lista da língua inglesa, a direita da língua portuguesa.



4 Conclusão

O trabalho nos permitiu visualizar o espalhamentos das diferentes funções, através das imagens ppm, possibilitando assim maior compreenção da matéria. A diferença entre as imagens das listas de palavras, está na quantidade de colisões, esta sendo maior na lista de palvras em inglês.

Por fim, foi possível observar que a primeira função testada, obteve um espalhamento centralizado em diferentes lugares. Todavia a segunda função, demonstrou um espalhamento com maior divisão na matriz.