TRABALHO PRÁTICO 5:
Localidades de Referência Temporal e Espacial

Pedro Araujo Pires

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)

ppires@dcc.ufmg.br

1. INTRODUÇÃO

Neste trabalho foi feita uma análise em um log de acessos a vídeos, caracterizando suas localidades de referência temporal e espacial.

A transmissão de um vídeo atráves da internet é feita através de *chunks*. Cada *chunk* representa uma porção do vídeo que é identificada sequencialmente. Durante a transmissão de um vídeo, os *chunks* podem ser vistos em sequência (ver o vídeo do início ao fim), ou podem ser vistos em uma ordem diferente (avançar ou retroceder o vídeo). Para fazer o cálculo da localidade de referência temporal, é calculada a distância de pilha entre acessos a um mesmo *chunk*, e para a localidade espacial é calculada a distância entre *chunks* de um mesmo vídeos.

2. SOLUÇÃO PROPOSTA

Para fazer o cálculo das localidades de referência temporal, foi calculada a distância de pilha entre acessos a um mesmo *chunk*. Para fazer esse cálculo, os *chunks* são colocados e uma pilha no primeiro acesso. A partir daí, sempre que um *chunk* for acessado, ele é movido para o topo da pilha, e a distância de subida é armazenada (quantidade *chunks* que o *chunk* sendo acessado teve de ”pular”para chegar à primeira posição da pilha).

Para o cálculo das localidades de referência espacial, foi calculada a distância na memória entre dois *chunks* de um mesmo vídeo. Para isso, sempre que um *chunk* é acessado, é calculada a diferença entre a posição deste *chunk*, e o último *chunk* acessado do mesmo vídeo.

2.1. Estruturas de dados

2.1.1. Video:

Armazena informações sobre um vídeo.

1: Video
id;
último *chunk* acessado;
array as localidades de referência espacial do vídeo;
tamanho do array;

2.1.2. VideoChunk:

Armazena informações sobre um *chunk* de vídeo.

2: VideoChunk
id do vídeo;
index do *chunk*;
popularidade;

2.1.3. PilhaTemporal:

Estrutura de dados usada para fazer o cálculo das localidades de referência temporal.

3: PilhaTemporal
lista de acessos aos *chunks*;
array com as distâncias de pilha;
tamanho do *array*;

Todas as estruturas de dados foram alocadas dinamicamente. O programa `valgrind` foi utilizado para verificar se toda a memória utilizada pelo programa foi desalocada corretamente. A saída do programa foi:

```
HEAP SUMMARY:
    in use at exit: 0 bytes in 0 blocks
  total heap usage: 43 allocs, 43 frees, 1,772 bytes allocated

All heap blocks were freed -- no leaks are possible
```

2.2. Decisões de Implementação e Análise de Complexidade

Para fazer o cálculo das localidades de referência temporal, foi utilizada uma lista encadeada, onde cada item da lista é um *chunk* de vídeo. Sempre que é lido do log de entrada um novo acesso a um *chunk*, a lista é percorrida procurando por esse *chunk*. Se ele não estiver na lista, ele é inserido na primeira posição da lista. Se ele já estiver na lista, o *chunk* é enviado para a primeira posição da lista, e é calculada a distância de subida (quantas posições na lista o *chunk* subiu). Esse distância de subida é então armazenada em um *array*.

Durante a fase de pesquisa do *chunk* na lista, no pior caso a lista será percorrida completamente. Para enviar um item para o início da lista, somente é necessário algumas manipulações de ponteiros, e a operação é feita em $O(1)$. A operação de armazenar um novo valor de distância de pilha também é feita em $O(1)$, pois o tamanho do *array* é sempre conhecido. Logo, sendo C o número total de *chunks* presentes no log, e N o número de entradas no log, o algoritmo de cálculo das localidades de referência temporal possui complexidade de tempo $O(CN)$.

Para a complexidade de espaço, é necessário armazenar todos os *chunks* em uma lista, e armazenar todas as distância de pilha em um *array*. A lista possui uma entrada para cada *chunk*, e o *array* possui uma entrada para cada linha do log. Logo, a complexidade de espaço é $O(CN)$.

Para fazer o cálculo das localidades de referência espacial de cada vídeo, assim como suas respectivas popularidades, foi utilizada uma lista encadeada, onde cada item da lista é um vídeo. Sempre que é lida uma linha do log de acessos, a lista é percorrida

procurando pelo vídeo. Quando ele é encontrado, é calculada a diferença entre o *chunk* atual e o último *chunk* acessado, e essa diferença é armazenada em um *array*. O contador de popularidade do vídeo também é incrementado.

Durante a fase de pesquisa do vídeo, no pior caso a lista será percorrida completamente. O cálculo da diferença entre os acessos a *chunks* de um vídeo, e a inserção de um novo valor no *array* com as localidades de referência são feitos em $O(1)$. Como essas operações são feitas uma vez para cada linha lida do log, a complexidade de tempo é $O(VN)$, onde V é o total de vídeos no log, e N é o número de linhas no log.

Para a complexidade de espaço, é necessário armazenar todos os vídeos em uma lista. Para cada vídeo existe um *array* com as localidades de referência espacial do vídeo. Este *array* possui uma entrada para cada *chunk* do vídeo que foi acessado. Logo, a complexidade de espaço é $O(VN)$.

Apesar de no arquivo `PedroAraujo.txt` os vídeos possuírem ids de 1 a 10, e todos os vídeos possuírem *chunks* com índices de 0 a 19, não existe nada que garanta que esses números serão assim em todos os logs. Logo, ao invés de armazenar os *chunks* em um array, utilizando o próprio índice do *chunk* para acessar o array, foi utilizada a lista encadeada. Devido a isso, as complexidades de tempo sempre são uma função do total de *chunks* ou do total de vídeos. Se fosse utilizado um *array* ao invés da lista encadeada, as complexidades de tempo seriam somente uma função do total de linhas no log.

Devido à quantidade de dados gerados, foi utilizada a GNU Scientific Library para fazer os cálculos estatísticos necessários.

3. IMPLEMENTAÇÃO

3.1. Código

3.1.1. Arquivos .c

- **io.c:** Define as funções relacionadas à parte de entrada e saída do programa.
- **espacial.c:** Define as funções relacionadas ao cálculo das localidades de referência espacial.
- **temporal.c:** Define as funções relacionadas ao cálculo das localidades de referência temporal.
- **lista.c:** Define as funções relacionadas à lista encadeada.
- **video.c:** Define as funções relacionadas aos vídeos e aos *chunks*.
- **statistics.c:** Define as funções relacionadas ao cálculo das estatísticas relacionadas às localidades de referência.
- **main.c:** Define a função main do programa.

3.1.2. Arquivos .h

- **io.h:** Define os cabeçalhos das funções relacionadas à parte de entrada e saída do programa.
- **espacial.h:** Define os cabeçalhos das funções relacionadas à localidade de referência espacial.

- **temporal.h:** Define os cabeçalhos das funções relacionadas à localidade de referência temporal, e a struct $PilhaTemporal$.
- **video.h:** Define os cabeçalhos das funções relacionadas as vídeos e aos *chunks*, e as structs $Video$ e $VideoChunk$.
- **statistics.h:** Define os cabeçalhos das funções relacionadas ao cálculodas estatísticas.
- **lista.h:** Define os cabeçalhos das funções sobre a lista encadeada.
- **boolean.h:** Define o tipo de dado boolean.

3.2. Compilação

O programa deve ser compilado através do comando `make`. Esse comando irá compilar todos os arquivos, e gerar um executável chamado `tp5`.

Também é possível compilar o programa para que ele gere as estatísticas e os gráficos apresentados neste relatório. Para isso, deve-se utilizar o comando `make stats`. Porém, para que a compilação e a execução ocorram sem erros, é necessário que a **GNU Scientific Library** e programa **gnuplot** estejam instalados no sistema.

3.3. Execução

A execução do programa recebe como parâmetro somente o nome do arquivo que contém os dados de entrada. O comando para a execução do programa é da forma:

```
./tp5 <arquivo de entrada>
```

3.3.1. Formato da entrada

O arquivo de entrada é um log de acessos a vídeos, onde cada linha do log possui três números inteiros. O primeiro indica qual o vídeo sendo acessado, o segundo é o índice do *chunk*, e o terceiro o tamanho do *chunk*.

```
3 9 50
3 15 63
6 14 62
1 0 45
1 1 37
1 2 80
```

3.3.2. Formato da saída

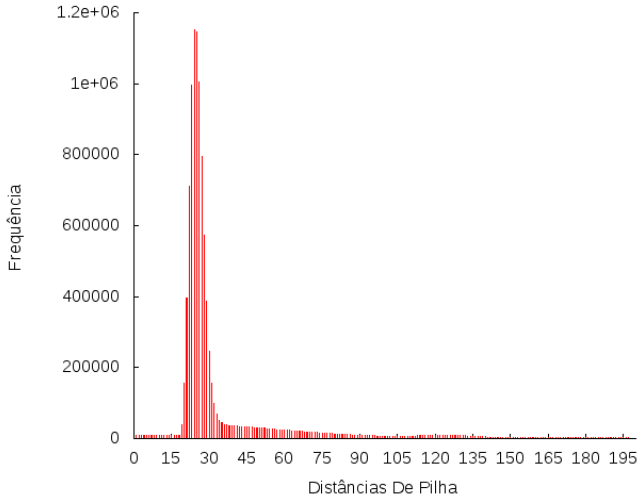
A saída do programa consiste em três arquivos: `localidadetemporal.txt`, `localidadeespacial.txt` e `popularidade.txt`. O arquivo `localidadetemporal.txt` possui vários números separados por vírgula, onde cada número é uma distância de pilha. Essas distâncias de pilha foram calculadas considerando a ordem dos acessos presente no log. O arquivo `localidadeespacial.txt` possui o mesmo formato, onde os números representam as localidades de referência espacial de todos os vídeos, em sequência, concatenadas. O último arquivo, `popularidade.txt`, possui a popularidade de todos os vídeos.

Quando o programa for compilado com a opção de gerar as estatísticas, será gerado o arquivo `estatisticas.txt`, que contém todas as estísticas geradas. Os gráficos das estatísticas também serão gerados como imagens no formato PNG.

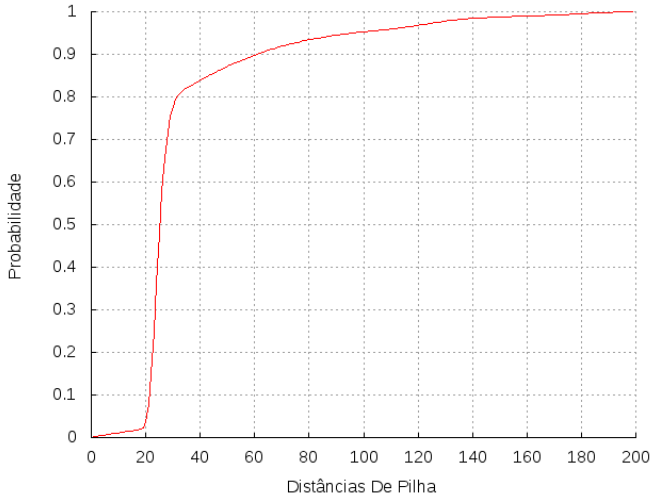
4. ANÁLISE ESTATÍSTICA

4.1. Localidade de Referência Temporal

Para analisar a localidade de referência temporal dos acessos aos *chunks* de vídeo, foram calculadas as distâncias de pilha. A figura abaixo mostra o histograma das distâncias de pilha.



Essas distâncias de pilha possuem uma média de 34.70, com desvio padrão 27.02. Considerando a quatidade de dados utilizados para calcular essa média, pode-se concluir que esta média é bem representativa, pois o desvio padrão corresponde somente a 13.5% do intervalo. A partir do histograma, é possível plotar a função de distribuição de probabilidade cumulativa (CDF), conforme mostra a gráfico abaixo.



Pela CDF podemos ver, por exemplo, que a probabilidade da distância de pilha ser menor ou igual a 60 é de 90%.

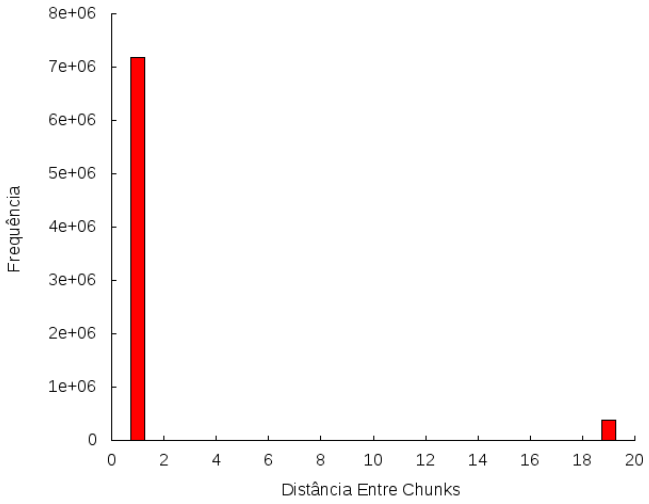
4.2. Localidade de Referência Espacial

Para calcular as localidades de referência espacial de cada vídeo, foi calculada a diferença dos índices de dois *chunks* acessados em sequência, de um mesmo vídeo. A Tablea 1 mostra as médias, desvio padrão e coeficiente de variação desses valores para cada vídeo.

# vídeo	Média	Desvio Padrão	Coef. de Variação
1	1.90	3.92	2.06
2	6.64	4.72	0.71
3	6.65	4.73	0.71
4	6.66	4.72	0.71
5	1.90	3.92	2.06
6	6.65	4.71	0.71
7	6.63	4.73	0.71
8	6.69	4.74	0.71
9	1.90	3.92	2.06
10	6.63	4.73	0.71

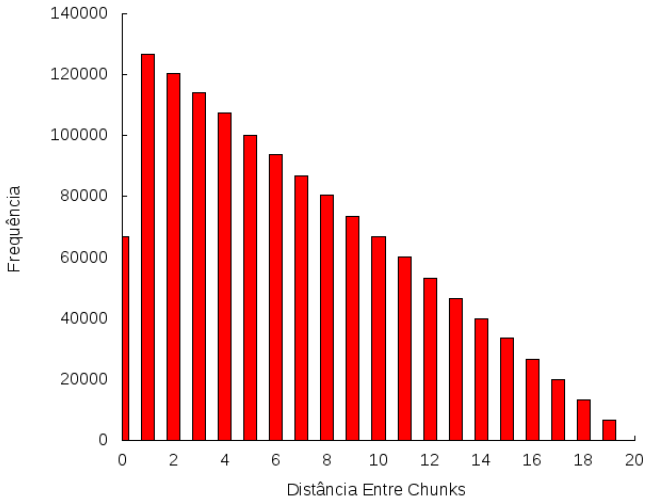
Tabela 1. Média das distâncias entre *chunks* dos vídeos

Pela tabela podemos ver que existem duas classes de vídeos. A primeira classe, composta pelos vídeos 1, 5 e 9, possui um histograma das distâncias entre *chunks* como o mostrado na figura abaixo.



O histograma indica que os *chunks* desses vídeos foram acessados em sequência, ou então somente o primeiro e último *chunks* foram vistos. Esse fato reflete na média e no desvio padrão dessas medidas. Como os pontos estão situados nos extremos do intervalo, a média é pouco representativa, como indica o desvio padrão.

Na segunda classe de vídeos os acessos foram mais ”desorganizados”, de forma que na maioria das vezes os vídeos não foram vistos em sequência. Esse comportamento pode ser observado no histograma abaixo:



Apesar de o desvio padrão desta classe de vídeos ser maior do que o da outra classe, o coeficiente de variação nos indica que a média de distância destes acessos é mais representativa do que a média de acessos do outra classe.

Todos os cálculos apresentados nesta seção foram feitos utilizando a **GNU Scientific Library**, disponível em `http://www.gnu.org/software/gsl/`. Os gráficos foram gerados pelo aplicativo **gnuplot**, disponível em `http://www.gnuplot.info/`.

5. CONCLUSÃO

Neste trabalho foram implementados algoritmos para calcular as localidades de referência temporal e espacial de um log de acessos a vídeos. Também foram feitas análises estatísticas sobre os dados gerados, possibilitando um entendimento melhor das localidades de referência. A partir destas estísticas, é possível ver que as localidades de referência temporal e espacial desempenham um papel importante na computação, pois mesmo com um conjunto de dados muito grande, as distâncias de acesso temporal e espacial são relativamente pequenas.