Algoritmos 2 - Trabalho Prático 1 - Manipulação de sequências

Matheus Flávio Gonçalves Silva - 2020006850

Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte - MG - Brasil

matheusfgs@ufmg.br

1 Introdução

O problema proposto consiste na implementação do algoritmo de compressão LZ78 com a utilização de um dicionário e de uma árvore trie, que não precisa ser compacta, para armazenar os prefixos.

O trabalho consiste na implementação da função de compressão dos dados de um arquivo .txt qualquer de entrada e da descompressão dos dados compactados em um arquivo qualquer de saída. Todos os outputs possuem um nome padrão que é o nome original do arquivo de entrada, mas com a devida extensão, mas é possível alterar o output por meio da passagem de um argumento para execução do programa.

2 Instruções de Compilação e execução

A compilação e execução do programa é feita de acordo com o indicado na documentação de especificação do trabalho.

Compilação:

- Abra um terminal e acesse a pasta raiz do trabalho;
- Execute o comando

  ```
  make
  ```

  no terminal;
- Com esse comando, são criados alguns **".o"** intermediários e o arquivo **"tp01"** que é o programa compilado e pronto para execução.

Execução:

- Uma vez com o projeto compilado, abra um terminal e acesse a pasta raiz do trabalho;
- Execute o programa com um caso teste com o comando:

```
./tp01 -option "arquivo_entrada" -o "arquivo_saida"
```

sem aspas, em que **'-option'** é ou **-c"** ou **-x"** para compressão/descompressão, respectivamente. **'arquivo_entrada'** se refere à localização/nome de um arquivo de entrada, tanto para compressão quanto para descompressão. A última parte do comando é opicional, e serve para definir o nome do arquivo de saída tanto para compressão quanto para descompressão;

3 Implementação

O programa, bibliotecas e cabeçalhos utilizados foram desenvolvidos na linguagem C++, sendo utilizado o compilador G++ (com padrão de execução conforme dito na documentação do trabalho, a compilação é dada por meio do comando).

O desenvolvimento se deu em um notebook Asus X555UB-BRA-XX299T com processador Core i5 6200u e 8GB de RAM 1333Mhz LDDR3 com SSD de 240GB SanDisk em ambiente Linux Mint 21.1. Foi utilizado o Visual Studio Code como IDE para desenvolvimento.

3.1 Estruturas de Dados

Para a implementação do trabalho foram usados os **Tipos Abstratos de Dados (TADs) "Node"** e **"Trie"** para a criação da árvore de prefixos.

Cada uma possui os seguintes atributos:

Node:

```
char: caracter;
int: code;
std::unordered_map<char, Node*> children; // Armazena os filhos de determinado nó
```

Cada um dos atributos é utilizado para criar um elemento da árvore trie.

Trie:

```
Node* root;
int nextCode;
int insert(std::string word);// Insere uma nova palavra na árvore
int find(std::string word);// Procura se a palavra passada se encontra na árvore
```

3.2 Funções e Procedimentos

3.2.1 Arquivo node.cpp

- **Node(char caracter, int code)**: cria um novo nó

3.2.2 Arquivo trie.cpp

- **int Trie::insert(std::string word)**: Insere uma palvra à árvore. Sempre procura se os prefixos que formam a palavra já se encontram. Adiciona apenas o que previamente não estava na árvore.

- **int Trie::find(std::string word)**: Procura por uma palavra na árvore. Retorna código -1 caso a palavra não seja encontrada e retorna o código do nó caso seja o último nó da palavra.

3.2.3 Arquivo lz78.cpp

- **void compression(string filename, string output)**: Realiza a operação de compressão do texto passado como argumento. Utiliza ifstream e ofstream para leitura e escrita de arquivos. Faz uso de diversas funções padrões e da funções definidas anteriormente nos outros arquivos. Além disso, faz uso de variáveis de auxílio.
- **void decompression(string filename, string output)**: Realiza a operação de descompressão do texto passado como argumento. Utiliza ifstream e ofstream para leitura e escrita de arquivos. Faz uso de diversas funções padrões e da funções definidas anteriormente nos outros arquivos. Além disso, faz uso de variáveis de auxílio.

3.2.4 Arquivo main.cpp

- **opcaoInvalida()**: Imprime no console que foi passada uma configuração errada de entrada.
- **void compressionReport(string inputName, std::pair¡double,double¿values)**: Imprime no console que os valores teóricos de compressão segundo o método do LZ78. Não reflete o resultado real com tamanho de arquivos.

O programa principal cria uma série de variáveis de auxílio para a execução do algoritmo. É o responsável por definir qual operação será realizada e quais os nomes dos arquivos que serão manipulados. Além disso, imprime no terminal as informações a respeito da compressão dos arquivos, considerando que os bits têm tamanho 8 e que o tamanho da compressão é dado pela dimensão da árvore.

4 Análise de Complexidade

A complexidade média estimada da criação da Trie é O(M * N) para a compressão, em que M é o número de palavras do arquivo e N é o comprimento médio das palavras. Como os textos são textos reais, espera-se que esse tamanho médio em geral se aproxime à média de tamanho das palavras mais comumente utilizadas na língua em que foi escrito.. A complexidade do programa de descompressão varia linearmente com o tamanho do arquivo, dado que a reconstrução a partir do dicionário é feita com um vetor, de modo que inserções e consultas são em O(1), de modo que o tamanho do arquivo domina assintoticamente a operação pela repetição.

5 Análise de compressão

5.1 constituicao1988.txt:

O arquivo original tem tamanho em bits: 5214320

O arquivo comprimido tem tamanho em bits: 711088

A taxa de compressão é de 13.6372 %

5.2 dom_casmurro.txt:

O arquivo original tem tamanho em bits: 3276880

O arquivo comprimido tem tamanho em bits: 592576

A taxa de compressão é de 18.0835%

5.3 os_lusiadas.txt:

O arquivo original tem tamanho em bits: 2756304

O arquivo comprimido tem tamanho em bits: 519544

A taxa de compressão é de 18.8493%

5.4 1-LoremIpsum.txt:

O arquivo original tem tamanho em bits: 8160608

O arquivo comprimido tem tamanho em bits: 909440

A taxa de compressão é de 11.1443%

5.5 2-CatchingFire.txt:

O arquivo original tem tamanho em bits: 8160608

O arquivo comprimido tem tamanho em bits: 909440

A taxa de compressão é de 11.1443%

5.6 3-AMetamorfose.txt

O arquivo original tem tamanho em bits: 966328

O arquivo comprimido tem tamanho em bits: 191400

A taxa de compressão é de 19.8069%

5.7 4-PoemaCarlosDrummond.txt

O arquivo original tem tamanho em bits: 43592

O arquivo comprimido tem tamanho em bits: 13352

A taxa de compressão é de 30.6295%

5.8 5-ADivinaComedia.txt

O arquivo original tem tamanho em bits: 5923112

O arquivo comprimido tem tamanho em bits: 1.01318e+06

A taxa de compressão é de 17.1055

5.9 6-MemoriasPostumasDeBrasCubas.txt

O arquivo original tem tamanho em bits: 2979296

O arquivo comprimido tem tamanho em bits: 538456

A taxa de compressão é de 18.0733

5.10 7-OCortico.txt

O arquivo original tem tamanho em bits: 4016248

O arquivo comprimido tem tamanho em bits: 709232

A taxa de compressão é de 17.6591

5.11 8-TristeFimDePolicarpoQuaresma.txt

O arquivo original tem tamanho em bits: 3253912

O arquivo comprimido tem tamanho em bits: 601416

A taxa de compressão é de 18.4829

5.12 9-OMulato.txt

O arquivo original tem tamanho em bits: 4662848

O arquivo comprimido tem tamanho em bits: 801736

A taxa de compressão é de 17.1941

5.13 10-PequenoPrincipe.txt

O arquivo original tem tamanho em bits: 631584

O arquivo comprimido tem tamanho em bits: 130272

A taxa de compressão é de 20.6262

6 Conclusão

A compreensão e desenvolvimento do trabalho em si se deu com um pouco de dificuldade, sendo necessário extrapolar um pouco do horário para finalizar o relatório.

Além diss, em tamanho real, não teórico, o único arquivo em que a compressão se deu menor que o texto original foi o LoremIpsum. Creio que isso se deve ao fato de ser um texto com palavras de maior tamanho, de modo que o arquivo .z78 gerado conseguiu reduzir essa média.

Referências

VIMIEIRO, Renato Slides virtuais da disciplina de Algoritmos 2 (2023).

Disponibilizado via Teams. Departamento de Ciência da Computação. Universidade Federal de Minas Gerais. Belo Horizonte. Acesso diversas vezes.

FAROOQ, Omar"Std Filesystem C++"

Disponível em: <https://linuxhint.com/std-filesystem-cpp/. Acesso em 9 mai. 2023

Wikipedia "LZ78"

Disponível em: <https://pt.wikipedia.org/wiki/LZ78. Acesso em 9 mai. 2023