Universidade Federal da Paraíba [UFPb]
Introdução a Teoria da Informação
Joison Oliveira Pereira - 20170068553
Rebeca Raab Bias Ramos - 20170070453

Parte 1: Projeto ITI

Implementação de um compressor e descompressor usando o algoritmo LZW

O presente relatório tem o intuito de explanar sobre a implementação feita para a primeira parte do projeto da cadeira Ïntrodução a informação"

1. Linguagem, Algoritmo, e bibliotecas Base

Para a criação da aplicação usou-se a linguagem de programação *Python* devido a familiaridade dos desenvolvedores com a linguagem e também pela amplitude de ferramentas que facilitam a implementação do algoritmo. O LZW *(Lempel-Ziv-Welch)* foi escolhido para realização da compressão e descompressão dos arquivos originais fornecidos pelo professor da disciplina que servem de entrada para o *software*. Além disso, no desenvolvimento, não foram utilizadas nenhuma biblioteca como base para o algoritmo apenas as que auxiliaram na geração de gráficos, manipulações de variáveis de tempo e barras de progresso.

2. Testes de Compressão e Descompressão

A execução do algoritmo nos testes, utilizando os arquivos (**corpus.txt**) e (**disco.mp4**), se mostrou correta, apresentando resultados positivos para as entradas. Funcionando tanto para caracteres podendo haver acentuações ou não, ocorrendo de forma similar para letras maiúsculas, minúscula e caracteres especiais.

Mesmo após a compressão e descompressão, o arquivo de texto mostra as informações corretas e o arquivo de vídeo ainda continua tocável, o que indica que a implementação executa de maneira correta, isso é mais notável observando os arquivos $saidaDEScompressao\backslash saidaDescomprimidaX$.

No que se trata das comparações ao arquivo original usou-se o programa *diff* da plataforma link para o site para realizar esta comparação. Que indicou que os arquivos Original e Comprimido são iguais, como mostra a figura 1.

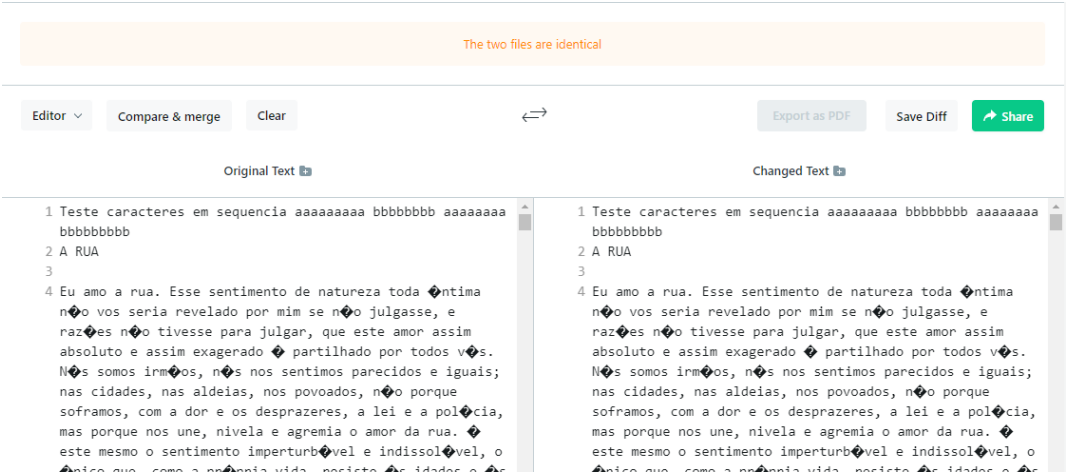

Figura 1: Comparação do arquivo descomprimido de corpus16 com arquivo original.

3. Valores de K's solicitados

Para o algoritmos LZW, foi pedido que o K fosse de 9 a 16. E de fato isso aconteceu na implementação e impressão. Em todas as iterações do algoritmo o K assumiu valores adequados e compatíveis com o esperado, para cada valor de K foi um arquivo dentro da pasta **saidaCompressao** e da pasta **saidaDEScompressao**.

```
100%|██████████| 15637138/15637138 [00:17<00:00, 881731.34it/s]
K = 9  ||  Tamanho: 15637138 -> 9046854
========================================================================
100%|██████████| 15637138/15637138 [00:16<00:00, 960521.25it/s]
K = 10  ||  Tamanho: 15637138 -> 7338381
========================================================================
100%|██████████| 15637138/15637138 [00:16<00:00, 955453.83it/s]
K = 11  ||  Tamanho: 15637138 -> 6134026
========================================================================
100%|██████████| 15637138/15637138 [00:15<00:00, 988548.86it/s]
K = 12  ||  Tamanho: 15637138 -> 5330170
========================================================================
100%|██████████| 15637138/15637138 [00:15<00:00, 998156.14it/s]
K = 13  ||  Tamanho: 15637138 -> 4712040
========================================================================
100%|██████████| 15637138/15637138 [00:15<00:00, 1017617.95it/s]
K = 14  ||  Tamanho: 15637138 -> 4199068
========================================================================
100%|██████████| 15637138/15637138 [00:15<00:00, 977522.49it/s]
K = 15  ||  Tamanho: 15637138 -> 3766604
========================================================================
100%|██████████| 15637138/15637138 [00:16<00:00, 951055.99it/s]
K = 16  ||  Tamanho: 15637138 -> 3396767
========================================================================
```

Figura 2: Logs de COMPRESSÃO do arquivo corpus.txt.

```
100%|██████████| 9046854/9046854 [00:09<00:00, 911428.29it/s]
K = 9  ||  Tamanho: 9046854 -> 15637138
Tempo 16.941451 segundos

========================================================================
100%|██████████| 7338381/7338381 [00:07<00:00, 955655.53it/s]
K = 10  ||  Tamanho: 7338381 -> 15637138
Tempo 13.096710 segundos

========================================================================
100%|██████████| 6134026/6134026 [00:06<00:00, 920137.62it/s]
K = 11  ||  Tamanho: 6134026 -> 15637138
Tempo 11.167416 segundos

========================================================================
100%|██████████| 5330170/5330170 [00:05<00:00, 913810.41it/s]
K = 12  ||  Tamanho: 5330170 -> 15637138
Tempo 9.761962 segundos

========================================================================
100%|██████████| 4712040/4712040 [00:05<00:00, 861184.25it/s]
K = 13  ||  Tamanho: 4712040 -> 15637138
Tempo 8.966078 segundos

========================================================================
100%|██████████| 4199068/4199068 [00:05<00:00, 837292.24it/s]
K = 14  ||  Tamanho: 4199068 -> 15637138
Tempo 8.113122 segundos

========================================================================
100%|██████████| 3766604/3766604 [00:04<00:00, 791325.44it/s]
K = 15  ||  Tamanho: 3766604 -> 15637138
Tempo 7.725772 segundos

========================================================================
100%|██████████| 3396767/3396767 [00:04<00:00, 750477.95it/s]
K = 16  ||  Tamanho: 3396767 -> 15637138
Tempo 7.076141 segundos

------------------------------------------------------------------------
```

Figura 3: Logs de DESCOMPRESSÃO do arquivo corpus.txt.

```
100%|██████████| 2111047/2111047 [00:03<00:00, 591329.66it/s]
K = 9  ||  Tamanho: 2111047 -> 2097699
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 599743.44it/s]
K = 10  ||  Tamanho: 2111047 -> 2090702
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 598840.74it/s]
K = 11  ||  Tamanho: 2111047 -> 2081530
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 604196.30it/s]
K = 12  ||  Tamanho: 2111047 -> 2069378
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 606601.84it/s]
K = 13  ||  Tamanho: 2111047 -> 1959288
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 610026.53it/s]
K = 14  ||  Tamanho: 2111047 -> 1772621
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 612301.77it/s]
K = 15  ||  Tamanho: 2111047 -> 1520385
========================================================================
100%|██████████| 2111047/2111047 [00:03<00:00, 625515.34it/s]
K = 16  ||  Tamanho: 2111047 -> 1289578
========================================================================
```

Figura 4: Logs de COMPRESSÃO do arquivo disco.mp4.

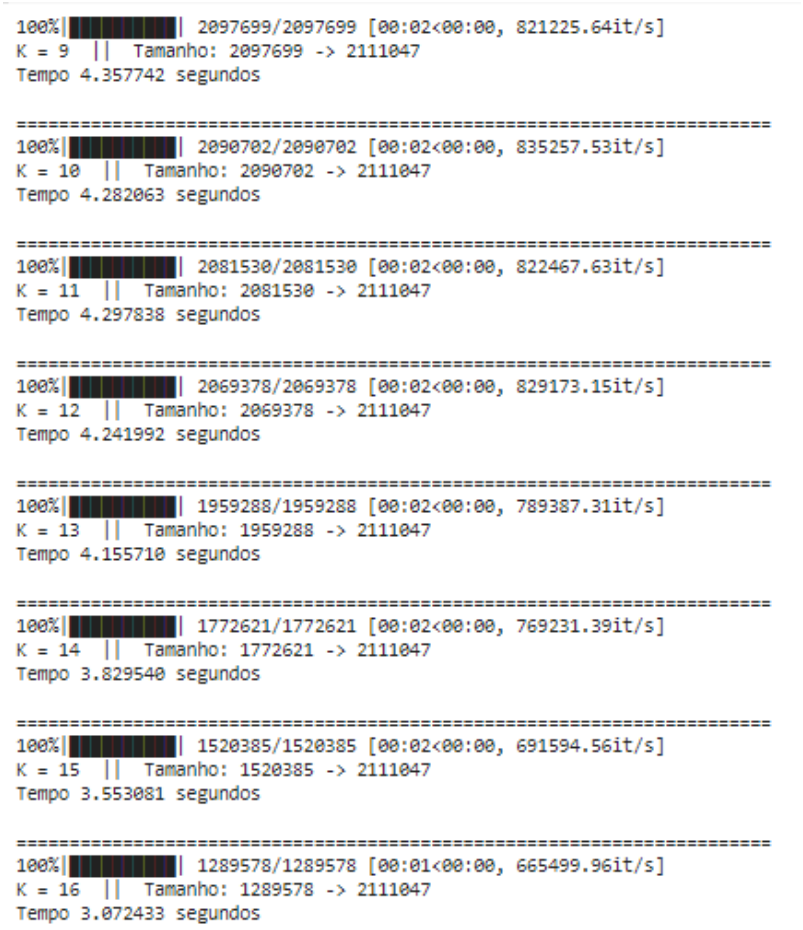

Figura 5: Logs de DESCOMPRESSÃO do arquivo disco.mp4.

4. Armazenamento de K bits no arquivo e técnicas utilizadas

O armazenamento ocorreu de forma correta no arquivo, assim como pode-se conferir nas figuras mostradas acima. Para isso, foram armazenados os índices dos dicionário em dois bytes.

5. Curva RC x K e Tempo x K

Também foi plotado os gráficos para **rc**, **rcIdeal** e de **tempo**. Obtendo resultados satisfatórios para ambos arquivos de entrada. Os cálculos de foram obtidos através das seguintes expressões em *Python*, usando a função *len* para usar o tamanhos dos arquivos no cálculo:

$$rc.append(8/(len(arqComprimido) * 16/len(arqOriginal)))$$

$$rcIdeal.append(8/(len(arqComprimido) * K/len(arqOriginal)))$$

Como mostra as figuras 2, 3 para o arquivo de entrada corpus.txt e as figuras 4, 5 para o arquivo de entrada disco.mp4.

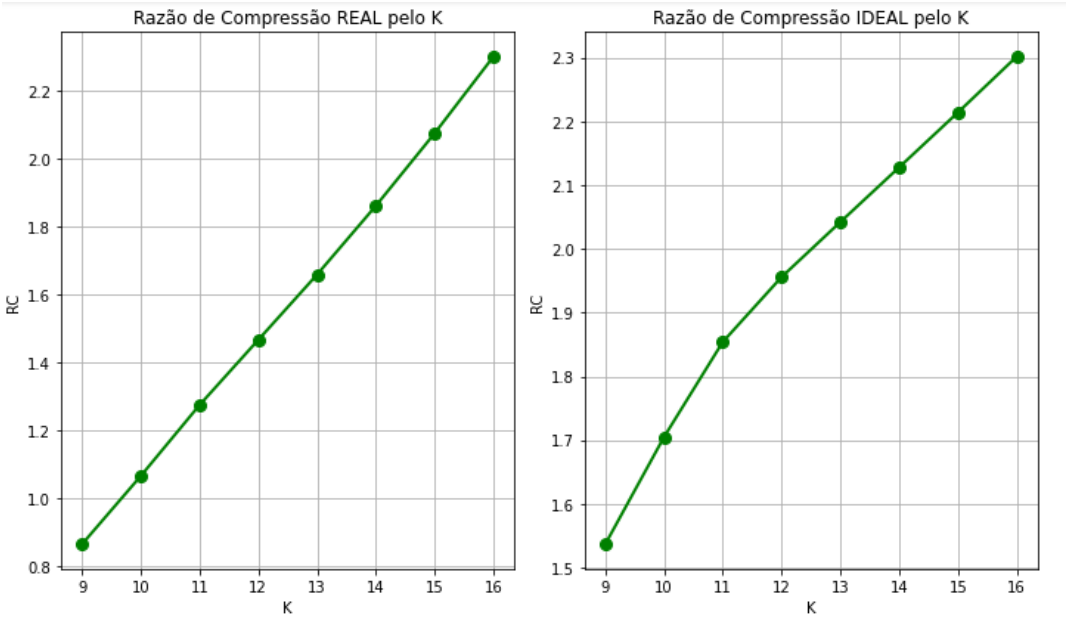

Figura 6: Comparação gráficos de razão de compressão REAL e IDEAL arquivo corpus.txt.

Figura 7: Gráfico de tempo arquivo corpus.txt.

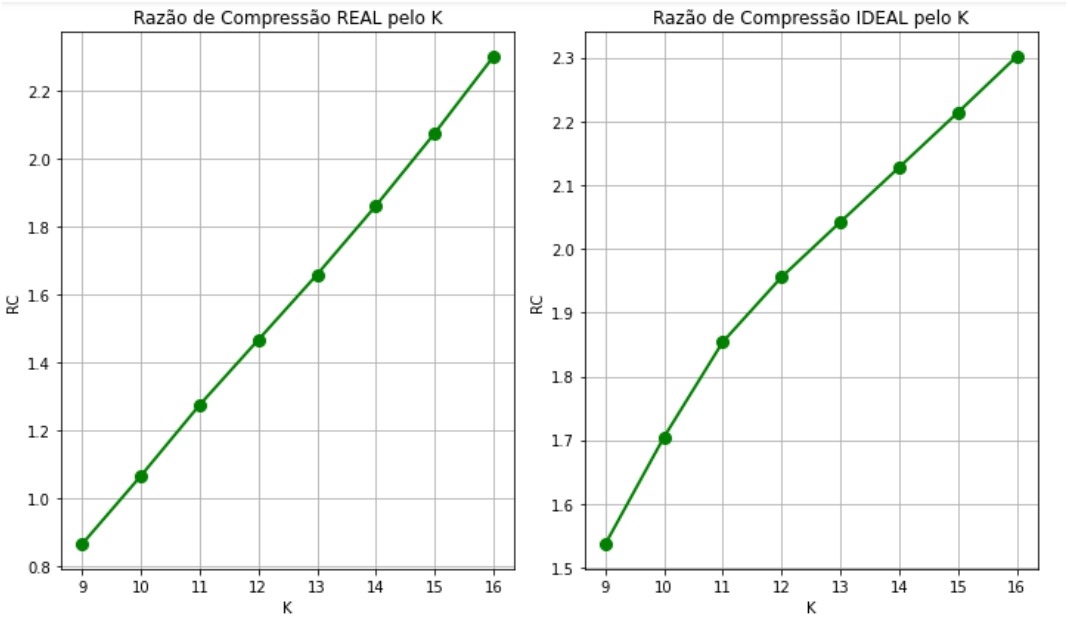

Figura 8: Comparação gráficos de razão de compressão REAL e IDEAL arquivo disco.mp4.

Figura 9: Gráfico de tempo arquivo disco.mp4.

6. Tamanho máximo

Dicionário ficou estático depois que atingiu o tamanho máximo. O tamanho máximo foi no K = 16.