Trabalho Prático 1 de Algoritmos 2

Nome: Artur Gaspar da Silva

Explicando o que foi feito:

O intuito deste trabalho foi implementar um algoritmo de compressão de arquivo que utiliza Tries, baseado no método LZ78. Para mais detalhes da especificação do trabalho, leia o arquivo “tp1.txt”. Nesse relatório será explicado em maior detalhe as escolhas de implementação do algoritmo, como compilar e como executar o programa, além de uma brevíssima descrição dos arquivos presentes aqui e uma descrição do formato do arquivo comprimido.

Escolhas de implementação:

Escolha da Trie Simples (ao invés da compacta):

Escolhemos implementar uma trie simples (ao invés de uma trie compacta), pois a trie compacta não apresentaria benefícios para esse trabalho: se fóssemos pegar todos os prefixos ou todos os sufixos do texto, a trie compacta teria complexidade assintótica de espaço linear, enquanto a trie padrão teria complexidade quadrática. Mas esse não é o caso, aqui a trie simples já é suficiente.

Escolha do limite da quantidade de códigos

Aqui foi decidido que o texto não teria mais que 2^{30}-1 caracteres, e portanto o arquivo comprimido não teria mais que essa quantidade de códigos (afinal no máximo adicionamos um código por caractere lido, nunca mais que isso). Note que 2^{30}-1 caracteres seriam mais de 1Gb de arquivo, mesmo com um algoritmo linear no tamanho da entrada é esperado que isso demore mais de um minuto num computador comum.

Escolha da representação de códigos e caracteres.

A fim de economizar espaço, foi tomada a decisão de usar o mínimo possível de bits para representar um caractere, e o mínimo possível de bits para representar um código. Ou seja, se temos 4 caracteres diferentes por exemplo, cada um será representado por 2 bits apenas. Se tivermos 5 códigos, cada um será representado por 3 bits (pois 3 é o menor inteiro tal que $2^3 >= 5$).

Brevíssima descrição dos arquivos presentes:

Pra todo “$i” inteiro entre 01 e 10 (incluindo 01 e 10), “teste$i.txt” é um dos casos de teste padrão pedidos. Veja a seção “Resultados” pra uma melhor descrição do que é cada um.

Para todo “$i” inteiro igual a 1 ou 2, “bonus$i.txt” é um caso de teste bônus, que eu achei interessante. Veja a seção “Resultados” pra uma melhor descrição do que é cada um.

O arquivo “trie_utils.cpp” implementa a classe da trie com todas suas funcionalidades, e algumas utilidades como “imprimir erro de leitura de arquivo”.

O arquivo “d_compressor.cpp” implementa o algoritmo principal, utilizando-se das funcionalidades implementadas em “trie_utils.cpp”.

Como compilar o programa:

Com os arquivos “d_compressor.cpp” e “trie_utils.cpp” no mesmo diretório do terminal, utilize o comando “g++ -O3 d_compressor.cpp -o d_compressor.out”.

Como executar o programa:

Compressão

./<programa> -c <arquivo_entrada> [-o <arquivo_saida>]

Descompressão

./<programa> -x <arquivo_entrada> [-o <arquivo_saida>]

A especificação do arquivo de saída é opcional. Caso ela não seja dada, o nome do arquivo de saída para compressão deve ser o nome original do arquivo acrescido da extensão z78. No caso da descompressão, será o nome do arquivo original, suprimida a extensão atual e acrescentada a extensão .txt.

Resultados:

Casos de teste:

teste01.txt: historinhas escritas por mim sobre um mundo de RPG on-line que eu estava inventando quando era mais novo. Taxa de compressão: arquivo comprimido com (34 026)/(61 621) = 55.22% do tamanho original, o original é (61 621)/(34 026) = 1.81 vezes maior.

teste02.txt: mais algumas historinhas escritas por mim quando eu era mais novo, de personagens que eu inventei pro mundo do teste01. Taxa de compressão: arquivo comprimido com (3 790)/(5 532) = 68,51% do tamanho original, o original é (5 532)/(3 790) = 1.46 vezes maior.

teste03.txt: resumo que eu escrevi quando era mais novo das histórias do teste01 e teste02. Taxa de compressão: arquivo comprimido com (6 800)/(10 617) = 64.04% do tamanho original, o original é (10 617)/(6 800) = 1.56 vezes maior.

teste04.txt: segunda versão do resumo do teste03, com algumas partes a mais na história. Taxa de compressão: arquivo comprimido com (9 546)/(15 630) = 61.07% do tamanho original, o original é (15 630)/(9 546) = 1.64 vezes maior.

teste05.txt: argumentos escritos por eu mais novo para tentar convencer as outras pessoas que ciravam coisas pro servidor de Minecraft de RPG a deixar essas histórias que eu inventei serem “oficiais”. Taxa de compressão: arquivo comprimido com (3 599)/(5 094) = 70.65% do tamanho original, o original é (5 094)/(3 599) = 1.41 vezes maior.

teste06.txt: Romeu e Julieta, de Willian Sheakspare (https://www.gutenberg.org/cache/epub/1513/pg1513.txt). Taxa de compressão: arquivo comprimido com (95 210)/(163 624) = 58,19% do tamanho original, o original é (163 624)/(95 210) = 1,72 vezes maior.

teste07.txt: The Critique Of Pure Reason, de Immanuel Kant (https://www.gutenberg.org/files/4280/4280-0.txt). Taxa de compressão: arquivo comprimido com (542 615)/(1 294 029) = 41.93% do tamanho original, o original é (1 294 029)/(542 615) = 2.38 vezes maior.

teste08.txt: The call of Cthulhu, de Howard Philips Lovecraft (https://www.gutenberg.org/cache/epub/68283/pg68283.txt). Taxa de compressão: arquivo comprimido com (54 668)/(90 450) = 60.44% do tamanho original, o original é (90 450)/(54 668) = 1.65 vezes maior.

teste09.txt: 山水情, de autor anônimo (https://www.gutenberg.org/files/25146/25146-0.txt). Taxa de compressão: arquivo comprimido com (217 862)/(339 002) = 64.27% do tamanho original, o original é (339 002)/(217 862) = 1.56 vezes maior.

teste10.txt: Also sprach Zarathustra: Ein Buch für Alle und Keinen, de Friedrich Wilhelm Nietzsche (https://www.gutenberg.org/cache/epub/7205/pg7205.txt). Taxa de compressão: arquivo comprimido com (286 784)/(551 325) = 52.02% do tamanho original, o original é (551 325)/(286 784) = 1.92 vezes maior.

Bônus (caso que comprime muito bem e caso que comprime muito mal):

bonus1.txt: “42” repetido um milhão menos uma vezes. Taxa de compressão: arquivo comprimido com (4 603) /(1 999 998) = 0.23% do tamanho original, o original é (1 999 998)/(4 603) = 434,50 vezes maior.

bonus2.txt: 1 999 998 bytes gerados aleatoriamente. Taxa de compressão: o tamanho do arquivo original é (2 397 289)/(1 999 998) = 1.20 vezes maior, o original tem (1 999 998)/(2 397 289) = 83.43% do tamanho original. O algoritmo na verdade piora o resultado para textos completamente aleatórios.