Fontes de Informação e Codificadores

1 Descrição Geral

Este trabalho consiste em estudar as fontes de informação e suas formas de codificação. Para isso será estudado o algoritmo de codificação Lempel-Ziv-Welch.

Além do estudo, o algoritmo de Lempel-Ziv-Welch para codificação de informações deve ser implementado utilizando a linguagem de programação C ou C++.

Os arquivos fonte do programa devem ser escritos de tal forma que possam ser compilados tanto em distribuições Linux quanto em Windows. O compilador a ser usado deve ser o GCC.

O trabalho pode ser realizado em duplas, para possibilitar a discussão das questões relacionadas aos algoritmos e para possibilitar a implementação dentro do prazo limite para a entrega da tarefa.

2 Descrição do Trabalho

Cada grupo da disciplina é responsável por implementar tanto o codificador quando o decodificador Lempel-Ziv-Welch.

Na Figura 1 estão representados os elementos que formam o sistema de comunicação de dados que será utilizado no trabalho.

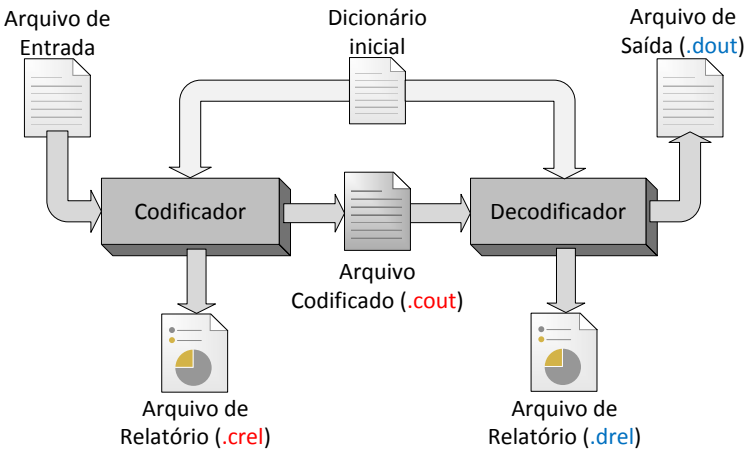


Figura 1: Elementos do sistema de comunicação de dados utilizado no trabalho.

2.1 Codificador Lempel-Ziv-Welch

O codificador é o programa que recebe dois arquivos de entrada: o arquivo que será codificado (**arquivo de entrada**) e um **dicionário**; e gera dois outros arquivos: um **arquivo codificado** e outro com um **relatório** do processamento.

O **arquivo de entrada** original deve ser composto por caracteres ASCII e pelo caractere de terminação de arquivo EOF.

O arquivo de **dicionário** contém uma lista de (*atributo*:*valor*), onde o "atributo" é um caractere válido e "valor" é o código numérico associado e esse caractere.

O **arquivo codificado** consiste do texto do arquivo original codificado segundo o algoritmo de Lempel-Ziv-Welch.

O arquivo de **relatório**, por sua vez, deve conter informações sobre as operações realizadas. Devem estar presentes as seguintes informações: dicionário de codificação gerado, nome e tamanho do arquivo de entrada e nome e tamanho do arquivo codificado.

O nome do arquivo de entrada podem ser passados por parâmetro para o programa codificador, na linha de comando. O **arquivo codificado** assim como o **arquivo de relatório** devem ser nomeados de acordo com o nome do arquivo de entrada, mudando-se o *file-type*. Portanto, respectivamente, devem ser chamados de <nome do arquivo de entrada>.cout e <nome do arquivo de entrada>.crel.

2.2 Decodificador Lempel-Ziv-Welch

O decodificador é o programa que recebe como entrada o **arquivo codificado** e o mesmo **dicionário** utilizado na codificação.

Como saída deverá gerar dois outros arquivos: um **arquivo decodificado** e um arquivo com o **relatório** do processamento.

Como resultado da operação do decodificador, o arquivo decodificado deve conter o mesmo conteúdo do arquivo de entrada utilizado no processo de codificação.

O arquivo de relatório do decodificador deve conter informações sobre o processamento realizado, bem como o dicionário de decodificação gerado, nome e tamanho do arquivo de entrada e nome e tamanho do arquivo decodificado.

Da mesma forma que para o codificador, os arquivos de entrada podem ser passados como parâmetros. O arquivo de saída decodificado e o arquivo de relatório devem ser nomeados como, respectivamente: <nome do arquivo de entrada>.dout e <nome do arquivo de entrada>.drel.

3 O algoritmo do codificador e do decodificador

Cada módulo (codificador e decodificador) do Lempel-Ziv-Welch deve ser implementado como um programa individual e independente.

Detalhes sobre os algoritmos de codificação e decodificação estão descritos no texto que acompanha a tarefa. As Figuras 2(a) e 2(b), no final deste documento, apresentam um fluxograma simplificado dos algoritmos de codificação e decodificação, respectivamente.

4 Formato dos arquivos

4.1 Formato do arquivo de dicionário

Para que os algoritmos de codificação e decodificação funcionem corretamente, os dicionários iniciais do codificador e do codificador devem ser idênticos.

Um exemplo de dicionário inicial é dado na Tabela 2, no final deste documento. Esse dicionário contém apenas os caracteres básicos minúsculos. Um dicionário completo deve incluir também as letras maiúsculas, acentos, caracteres acentuados (minúsculos e maiúsculos), etc. Para a realização deste trabalho não é necessário utilizar um dicionário completo. É suficiente um dicionário com todos os caracteres da Tabela 2, incluindo nele também o ".", "," e o " ", desde que os arquivos de entrada utilizados pelo codificador e decodificador contenham somente estes caracteres.

O arquivo com o dicionário inicial deve ser composto por uma lista de (*atributo:valor*), onde *atributo* é um caractere e *valor* é o código associado ao caractere. Por exemplo, o dicionário da Tabela 2 será representado no arquivo como:

```
a:1
b:2
c:3
d:4
.
.
```

```
.
y:25
z:26
```

4.2 Formato do arquivo codificado

O arquivo codificado deve obedecer ao formato da Tabela 1.

Campo	Tamanho	Descrição
gID	1 byte	Identificador do grupo que gerou o arquivo. Byte entre 0x01 e 0x09.
Telm	1 byte	Quantidade de bits utilizados para codificar os caracteres do arquivo original.
Dados	Variável	Dados da mensagem original codificados.

Tabela 1: Campos do formato de arquivo de codificação.

Um exemplo de resultado de codificação é dado a seguir:

```
00000000 00000100 0001 1111 1001 1000 0101 0101 0100 1111 0110 1001 1010 ...
```

Idealmente, por uma questão de compactação, o arquivo codificado deveria ser binário. Entretanto, para fins dessa tarefa, o arquivo codificado poderá ter seu campo de "Dados" formado por caracteres ASCII "0" e "1".

Dessa forma, o exemplo anterior teria dois bytes para identificação do grupo (valor 0x00) e da quantidade de bits usadas na codificação (0x04), e seria seguida por uma sequência de caracteres "0" e "1", da seguinte forma:

```
00011111100110000101010101001111011010011010 ...
```

Mas **CUIDADO!** ao calcular o tamanho do arquivo codificado, pois na parte de dados cada "bit" será representado por um "byte".

4.3 Formato do arquivo de relatório do codificador

O formato é livre, desde que contenha as seguintes informações:

1) tabela com o dicionário de codificação gerado para o arquivo de entrada; 2) dados dos arquivos de entrada e de saída (nome e tamanho em bytes); 3) a entropia do alfabeto usado; 4) a quantidade média de informação recebida por símbolo da mensagem (nesse caso, a mensagem é todo o arquivo de entrada); 5) o percentual de compactação do arquivo codificado em relação ao arquivo de entrada.

4.4 Formato do arquivo decodificado

Este arquivo deverá conter as informações decodificadas a partir do arquivo codificado de entrada. O conteúdo desse arquivo deverá ser o mesmo do arquivo usado como entrada no codificador.

4.5 Formato do arquivo de relatório do decodificador

O arquivo de relatório do decodificador deverá conter:

1) os dados dos arquivos de entrada e de saída (nome e tamanho em bytes) 2) a entropia do alfabeto usado; 3) a quantidade média de informação recebida por símbolo da mensagem (nesse caso, a mensagem é todo o arquivo de entrada); 4) o percentual de compactação do arquivo codificado de entrada em relação ao original.

5 Realização dos testes

Os grupos são responsáveis por escolher um arquivo de entrada. Sugere-se a utilização de texto reais, como por exemplo notícias, letras de músicas, trechos de livros, etc. Outra alternativa é utilizar geradores de textos aleatórios disponíveis na Internet.

Deverão ser realizadas medições da "transmissão/recepção" utilizando arquivos com tamanhos variando entre 2^k bytes, com $k = \{6, 7, 8, 9, 10, 11, 12\}$. Para cada tamanho deverão ser utilizados 10 arquivos distintos, de modo que o resultado seja estatisticamente confiável.

6 O que deve ser entregue

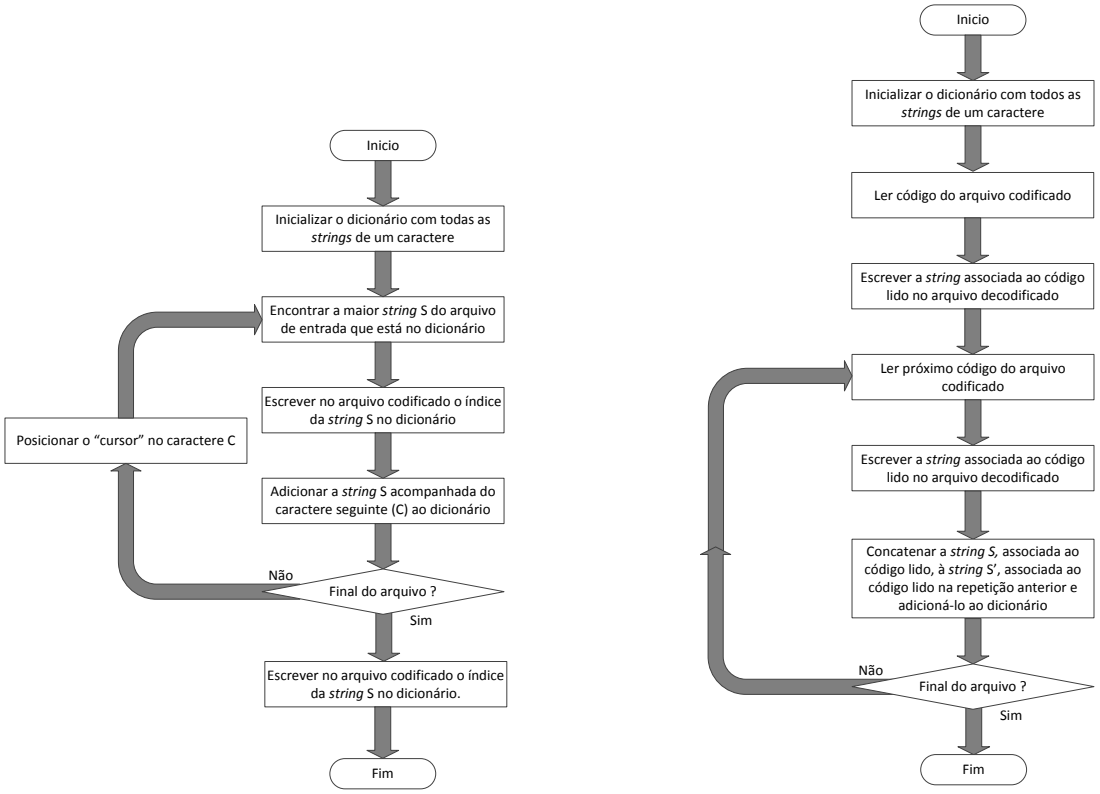
É obrigatória a entrega dos seguintes arquivos:

- Arquivos fontes desenvolvidos pelo grupo (arquivos de cabeçalhos e implementações) com a implementação do algoritmo Lempel-Ziv-Welch . Os arquivos devem ser livres de qualquer tipo de erro de compilação.
- Arquivos usados como entrada para o codificador **originais** , arquivo de **dicionário inicial** e os arquivos **resultantes do processo de codificação**.
- Arquivo no formado PDF com as **medições realizadas**, **documentação** sucinta do projeto, **ferramentas** e **procedimentos** necessários para compilar e *linkar* os arquivos objeto e descrição da estrutura de diretórios utilizada (que deve ser mantido no arquivo entregue).

Todos os arquivos devem ser entregues em um arquivo compactado em formato .zip ou .tar.

7 Considerações Finais

- **Todos** os arquivos solicitados devem ser entregues. A falta de algum deles incorrerá em redução da nota final do trabalho.
- É permitido aos grupos trocarem **ideias**, não implementações.
- Os trabalhos devem ser entregues até a data prevista. Trabalhos com até 1 semana de atraso, concorrem a 75% da nota. Trabalhos com até 2 semanas de atraso concorrem à 50% da nota.
- Trabalhos entregues além de duas semanas receberão nota zero.



(a) (b)

Figura 2: Fluxogramas simplificados do algoritmo Lempel-Ziv-Welch. Em 2(a) o codificador e em 2(b) o decodificador.

Caractere	Código	Caractere	Código
a	1	n	14
b	2	o	15
c	3	p	16
d	4	q	17
e	5	r	18
f	6	s	19
g	7	t	20
h	8	u	21
i	9	v	22
j	10	w	23
k	11	x	24
l	12	y	25
m	13	z	26

Tabela 2: Exemplo de dicionário inicial.