Compressão LZ77

Programação de Computadores I

Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

1 Introdução

O método de compressão LZ77 é nomeado desta forma devido aos seus criadores, Abraham Lempel e Jacob Ziv, e porque foi proposto em 1977.

Ele é um método amplamente utilizado em diversas ferramentas de compressão, como o 7ZIP e GZIP.

Ele funciona da seguinte forma: dada uma janela de tamanho k contendo os últimos símbolos codificados e os próximos símbolos, ele tenta achar a maior substring que inicia nesta janela e que é um prefixo dos próximos símbolos a serem codificados. Este prefixo então é representado através de uma tripla (x, y, c), indicando que a uma distância x para esquerda do início do prefixo, existe uma substring de tamanho y equivalente ao prefixo e que esta parte do prefixo equivalente à substring, é sucedida por um caractere c.

De maneira mais didática, tome o texto $T = aacaacabcabaaac$ e uma janela de tamanho 6, podemos codificá-lo como a Figura mostra abaixo.

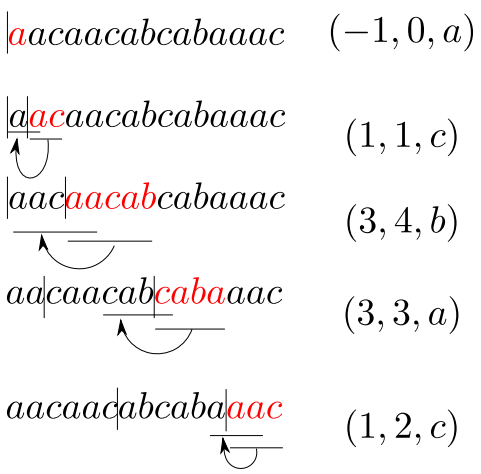

Figura 1: Codificação LZ77 para $T = aacaacabcabaaac$

A descompressão funciona de maneira inversa, substituindo as triplas pelas substrings envolvidas.

Existe um compromisso em relação ao tamanho da janela, quando maior a janela, melhor o resultado da compressão, mas mais lento será o processo de compressão e descompressão. Então este tamanho deve ser bem escolhido em vista da obtenção de uma boa relação de custo/benefício.

2 Descrição formal

Descrevendo o método mais formalmente. Suponha um texto $T[0, n-1]$ e que queremos codificar os próximos símbolos $T[i, n-1]$. A janela, por definição, consiste dos k símbolos anteriores a $T[i]$, isto é, $J = T[i-k, i-1]$.

Queremos procurar a maior substring que inicia na janela e que casa com um prefixo do texto. Em outras palavras, estamos interessados na substring de tamanho maximal ℓ, $T[j, j+\ell-1]$, que seja igual a $T[i, i+\ell-1]$, com $i-k \leq j \leq i-1$. Caso exista mais de uma substring nesta condição, escolhemos a que inicia mais a direita. Seja o índice x de onde

esta substring inicia, então, damos como saída para o compressor $(i-x, \ell, T[i+\ell])$. Caso não exista tal substring, o compressor dá como saída a tripla $(0, 0, T[i])$. Em qualquer caso, a janela é atualizada, passando a contemplar os últimos k símbolos codificados, isto é, os símbolos $T[i+\ell-k+1, i+\ell]$.

O caso inicial, em que nenhum símbolo ainda foi codificado e a janela é vazia, faz com que o compressor produza como saída $(-1, 0, T[0])$.

3 Objetivos

O objetivo deste trabalho é a implementação dos métodos de compressão e descompressão do LZ77 e a disponibilização de uma biblioteca, para que ela possa ser utilizado em programas de terceiros.

3.1 Especificação

Deverão ser disponibilizados dois módulos referentes à compressão e descompressão dos métodos LZ77. Cada módulo deverá possuir o arquivo cabeçalho e o arquivo de implementação. Os nomes dos arquivos devem ser:

- `lz77encode.h` e `lz77decode.c`, no módulo de compressão.
- `lz77decode.h` e `lz77decode.c`, no módulo de descompressão.

Obrigatoriamente, no módulo de compressão, deverá existir a função `encode` com a seguinte assinatura:

`compress(char* str,int k)`, em que `str` é a string a ser comprimida e `k` o tamanho da janela a ser utilizado.

Esta função deve imprimir o resultado da compressão na tela.

Similarmente, o módulo de descompressão deverá possuir a função `decompress(char* compressed_str`, em que `compressed_str` é a string a ser descomprimida. A função `decompress` deverá imprimir uma linha com a string descomprimida.

Funções auxiliares podem ser utilizadas em ambos os módulos, conforme a necessidade, contudo, o nome das funções principais de compressão e descompressão devem seguir as regras comentadas anteriormente.

3.2 Formato de Entrada e Saída

Ao receber uma *string* terminada com `\0` como entrada e um parâmetro k, a função `compress` deve escrever as triplas sem utilizar espaços para separá-las considerando k como o tamanho da janela. Abaixo uma série de exemplos demonstram a saída esperada.

- Exemplo 1:
  - Entrada: `ababcbababaaaaaa\0` e $k = 4$
  - Saída: `(-1,0,a)(0,0,b)(2,2,c)(4,3,a)(2,2,a)(1,4,\0)`
- Exemplo 2:

  - Entrada: `aacaacabcabaaac\0` e $k = 6$
  - Saída: `(-1,0,a)(1,1,c)(3,4,b)(3,3,a)(1,2,c)(0,0,\0)`

- Exemplo 3:
  - Entrada: `aaaaaaaaaaaaaaaaaaaaaaaaa\0` e $k = 5$
  - Saída: `(-1,0,a)(1,24,\0)`

Por sua vez, ao receber uma string contendo várias triplas, sem separação com espaços, e o parâmetro k que indica o tamanho da janela, a função `decompress` deverá imprimir, em uma única linha, a string descomprimida, como os exemplos abaixo:

- Exemplo 1:
  - Entrada: `(-1,0,a)(0,0,b)(2,2,c)(4,3,a)(2,2,a)(1,4,\0)`.
  - Saída: `ababcbababaaaaaa`.

- Exemplo 2:
  - Entrada: `(-1,0,a)(1,1,c)(3,4,b)(3,3,a)(1,2,c)(0,0,\0)`.
  - Saída: `aacaacabcabaaac`.

- Exemplo 3:
  - Entrada: `(-1,0,a)(1,24,\0)`.
  - Saída: `aaaaaaaaaaaaaaaaaaaaaaaaa`.

3.3 Documentação

O código deve estar bem documentando com comentários nos trechos cruciais para o entendimento do algoritmo. Além disso, um arquivo `Readme.txt` deve ser providenciado explicando como compilar a biblioteca disponibilizada.

3.4 Testes

Serão utilizados vários testes automatizados na biblioteca provida. Logo, é crucial que elas sigam rigidamente o formato de entrada e saída.

O ambiente de teste será um S.O `Linux` com compilador `GCC`.

4 Critérios de correção

Deve ser utilizada a linguagem de programação `C` para a implementação do compressor. A Tabela 1 expõe os critérios de avaliação.

Para validação da correção do algoritmo, testes automatizados serão realizados, então é **crucial** que a saída esteja conforme o especificado.

Serão descontados pontos dos códigos que não estiverem bem formatados.

Tabela 1: Tabela de pontuação

Critério	Nota
Correção da Implementação	90
Documentação	10
Total	100

5 Considerações

- Este trabalho pode ser feito em **dupla**.
- O trabalho que não compilar não será avaliado.
- Como a correção é automatizada, deverá ser impresso apenas o que a especificação pede, estilo URI/BOCA.
- Cópias e plágio serão avaliados automaticamente com nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do aluno através da sala de aula virtual da disciplina.