

TCC341 - Trabalho 04

Introdução

O código de Hamming é um código de bloco linear desenvolvido por Richard Hamming em 1950 utilizado no processamento de sinal e nas telecomunicações. A sua utilização permite a transferência de dados de forma segura e eficiente já que possibilita a identificação e correção de erros ou perdas de bits durante a comunicação. Normalmente é utilizado o sistema Hamming(7,4), no qual cada 4 bits de dados recebe 3 bits de paridade, possibilitando a detecção de erros de até 2 bits e a correção de até 1 bit.



Objetivo

Nesse trabalho você deve implementar uma função hash utilizando o código de Hamming e mostrar como ele não é uma boa função de hash criptográfica por apresentar uma alta taxa de colisão.

O código de Hamming adiciona novos bits, chamados nits de paridade, a uma sequência já existente de bits. Seja k o número de bits que iremos adicionar, ou seja, o tamanho do bits de paridade, temos que a mensagem

original deve ter $n = 2^k - 1 - k$ bits. Assim para uma mensagem de tamanho 11 adicionaremos 4 bits de paridades, pois $7 = 2^4 - 1 - 4 = 16 - 5 = 11$. A mensagem final terá, portanto, 15 bits (os originais mais o bits de paridade).

Os bits de paridade são adicionados ao bits originais em posições cujo índices são potências de 2 (começando o índice na posição 1). Assim o bit de paridade p_0 será adicionado na posição $2^0 = 1$, o bit de paridade p_1 será adicionado na posição $2^1 = 2$, e o bit de paridade p_i será adicionado na posição 2^i da sequência de bits originais.

O valor de cada bit é dado pela soma de alguns bits da sequência original em módulo 2. Assim, o bit de paridade p_0 será a soma dos elementos 1 a 1, sempre pulando 1 elemento a partir dele, ou seja, será a soma dos elementos nas posições 1, 3, 5, 7, 9, 11, etc. O bit de paridade p_1 será a soma dos elementos 2 a 2, sempre pulando 2 elementos a partir dele, ou seja, será a soma dos elementos nas posições 2, 3, 6, 7, 10, 11, etc. O bit de paridade p_2 será a soma dos elementos 4 a 4, sempre pulando 4 elementos a partir dele, ou seja, será a soma dos elementos nas posições 4, 5, 6, 7, 12, 13, 14, 15 etc. Assim, o bit de paridade p_i será a soma dos elementos 2^i a 2^i , sempre pulando 2^i elementos a partir dele.

Assim para construirmos o código de Hamming do número 1 0 0 1 1 0 1 0 0 0 0 fazemos o seguinte:

- Adicionamos aos bits os valores dos bits de paridade p_i na posição 2^i :
- $p_0 \ p_1 \ 1 \ p_2 \ 0 \ 0 \ 1 \ p_3 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0$
- Para o bit de paridade p_0 olharemos os valores dos bits de 1 em 1, pulando 1 valor, a partir dele:
- **$p_0 \ p_1 \ 1 \ p_2 \ 0 \ 0 \ 1 \ p_3 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0$**
- O bit de paridade p_0 terá o valor da soma dos bits em módulo 2:
- $p_0 = 1+0+1+1+1+0+0 = 4 = 0 \text{ mod } 2$
- Para o bit de paridade p_1 olharemos os valores dos bits de 2 em 2, pulando 2 valores, a partir dele:
- $p_0 \ p_1 \ 1 \ p_2 \ 0 \ 0 \ 1 \ p_3 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0$
- O bit de paridade p_1 terá o valor da soma dos bits em módulo 2:
- $p_1 = 1+0+1+0+1+0+0 = 3 = 1 \text{ mod } 2$

- Para o bit de paridade p_2 olharemos os valores dos bits de 4 em 4, pulando 4 valores, a partir dele:
- $p_0 p_1 1 p_2 0 0 1 p_3 1 0 1 0 0 0 0$
- O bit de paridade p_2 terá o valor da soma dos bits em módulo 2:
- $p_2 = 0+0+1+0+0+0+0 = 1 = 1 \bmod 2$
- Para o bit de paridade p_3 olharemos os valores dos bits de 8 em 8, pulando 8 valores, a partir dele:
- $p_0 p_1 1 p_2 0 0 1 p_3 1 0 1 0 0 0 0$
- O bit de paridade p_3 terá o valor da soma dos bits em módulo 2:
- $p_3 = 1+0+1+0+0+0+0+0 = 2 = 0 \bmod 2$

Assim o código de Hamming do número 1 0 0 1 1 0 1 0 0 0 0 é obtido simplesmente substituindo os bits de paridade nas suas posições corretas e será 0 1 1 1 0 0 1 0 1 0 1 0 0 0 0 já que seus bits de paridade são 0 1 1 0.

Entrada e Saída

Você deve ler dois inteiros, k e d nessa ordem, onde k é o número de bits de paridade e d é o número de valores de hash a serem calculados. Seguido disso, haverá d linhas, cada uma com n bits (valores somente 0 e 1) separados por espaços, onde $n = 2^k - 1 - k$. O valor de k será no máximo 10 e o valor de d no máximo 1024.

Após a leitura de cada sequência de bits, você deve imprimi-la na tela (SEM espaços entre os bits) seguido na mesma linha, separado por um hífen, de seu valor em hash, ou seja, de seus bits de paridade segundo o código de Hamming. Finalmente, você deve no final, imprimir a porcentagem de colisão, ou seja, o número de valores de hash (ou paridades) que possuem algum cópia na solução com precisão de duas casa decimais. Esse valor deve ser dado em porcentagem pelo valor de d , ou seja, o valor de hashes calculados. Nunca uma mesma sequência será dada na entrada.

Exemplos

Nos exemplos abaixo a cor azul representa a entrada que deve ser lida pelo seu programa e a cor vermelha, a saída que deve ser gerada pelo seu programa na saída padrão.

Exemplo 1:

4 1

1 0 0 1 1 0 1 0 0 0 0

10011010000 - 0110

O numero de colisoes foi 0.00%

Exemplo 2:

4 2

0 1 0 0 1 1 0 1 0 0 0

01001101000 - 0101

1 0 0 1 1 0 1 0 0 0 0

10011010000 - 0110

O numero de colisoes foi 0.00%

Exemplo 3:

4 9

0 1 0 0 1 1 0 1 0 0 0

01001101000 - 0101

1 0 0 1 1 0 1 0 0 0 0

10011010000 - 0110

0 1 0 1 0 1 0 1 0 1 0

01010101010 - 0101

1 1 1 1 1 1 1 1 1 1

11111111111 - 1111

0 0 0 0 0 0 0 0 0 0

00000000000 - 0000

1 0 1 0 1 0 1 0 1 0 1

10101010101 - 1010

1 1 1 0 0 0 1 1 1 0 0

11100011100 - 0101

1 1 0 0 1 1 0 0 1 1 0

11001100110 - 0110

0 0 1 1 0 0 1 1 0 0 1

00110011001 - 1001

O número de colisões foi 55.56%