Trabalho 2:
Simulação da Camada de Enlace

João Victor Alves dos Santos, 18/0033760
Pedro Lísias Viana Arcoverde Alves, 19/0036559
Valentin Ferreira Paes, 18/0037901

[1]Dep. Ciência da Computação – Universidade de Brasília (UnB)
CIC0235 - TELEINFORMÁTICA E REDES 1 - T01 (2022.2 - 24M12)
Prof. Marcelo Antonio Marotta

1. Introdução Teórica

A camada de enlace de dados é uma parte fundamental da comunicação de rede e desempenha um papel crucial na entrega confiável de pacotes de dados. Neste relatório, vamos apresentar um simulador de camada de enlace, que é uma ferramenta para simular e testar o comportamento da camada de enlace em diferentes cenários de rede.

Através da utilização deste simulador, será possível experimentar e avaliar diferentes protocolos e mecanismos da camada de enlace, como os protocolos de Contagem de caracteres, Inserção de bytes ou caracteres, Bit de paridade par, CRC e Hamming.

1.1. Contagem de caracteres

O protocolo de Contagem de caracteres (Character Counting Protocol, em inglês) é um protocolo de comunicação simples que é utilizado para garantir que todos os dados sejam transmitidos corretamente de um dispositivo para outro. O protocolo funciona adicionando um número de caracteres, conhecido como a contagem de caracteres, à mensagem antes de enviá-la. O dispositivo receptor então verifica a contagem de caracteres recebida contra a mensagem para garantir que todos os dados tenham sido transmitidos corretamente. A vantagem deste protocolo é que ele é fácil de implementar e pode ser usado em qualquer tipo de conexão de comunicação, desde uma rede local até uma conexão de longa distância. Além disso, o protocolo de contagem de caracteres é escalável, o que significa que ele pode ser facilmente ajustado para atender às necessidades de uma rede em constante mudança.

Em resumo, o protocolo de contagem de caracteres é uma ferramenta simples e eficaz para garantir a integridade dos dados na comunicação de rede. Ele é uma opção viável para pequenas redes ou comunicações de curta distância, onde a segurança é menos crítica, mas a integridade dos dados é ainda importante.

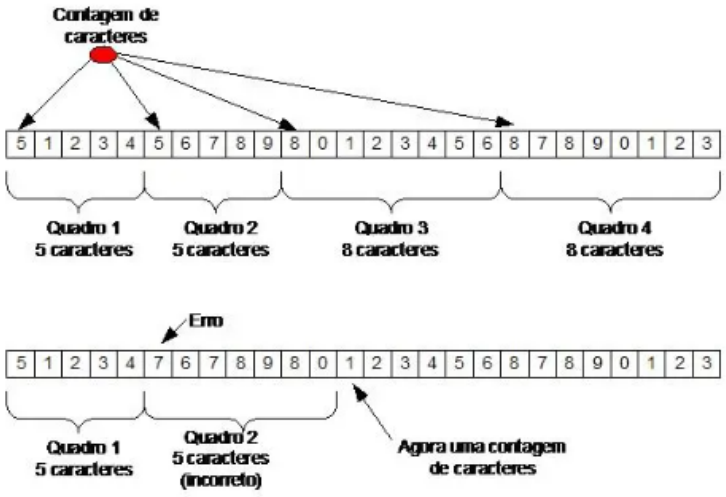

Figure 1. Contagem de caracteres

1.2. Inserção de bytes ou caracteres

O protocolo de Inserção de Bytes ou Caracteres é um protocolo de comunicação simples utilizado para garantir a integridade dos dados durante a transmissão. O protocolo funciona adicionando um número de bytes ou caracteres aos dados antes de enviá-los, conhecidos como bytes ou caracteres de inserção. O dispositivo receptor então verifica os bytes ou caracteres de inserção recebidos contra os dados para garantir que todos os dados tenham sido transmitidos corretamente. A vantagem deste protocolo é que ele é simples de implementar e pode ser usado em qualquer tipo de conexão de comunicação, desde uma rede local até uma conexão de longa distância. Além disso, o protocolo de inserção de bytes ou caracteres é escalável, o que significa que ele pode ser facilmente ajustado para atender às necessidades de uma rede em constante mudança.

Em resumo, o protocolo de inserção de bytes ou caracteres é uma ferramenta eficaz para garantir a integridade dos dados durante a comunicação de rede. Ele é uma opção viável para pequenas redes ou comunicações de curta distância, onde a segurança é menos crítica, mas a integridade dos dados é ainda importante.

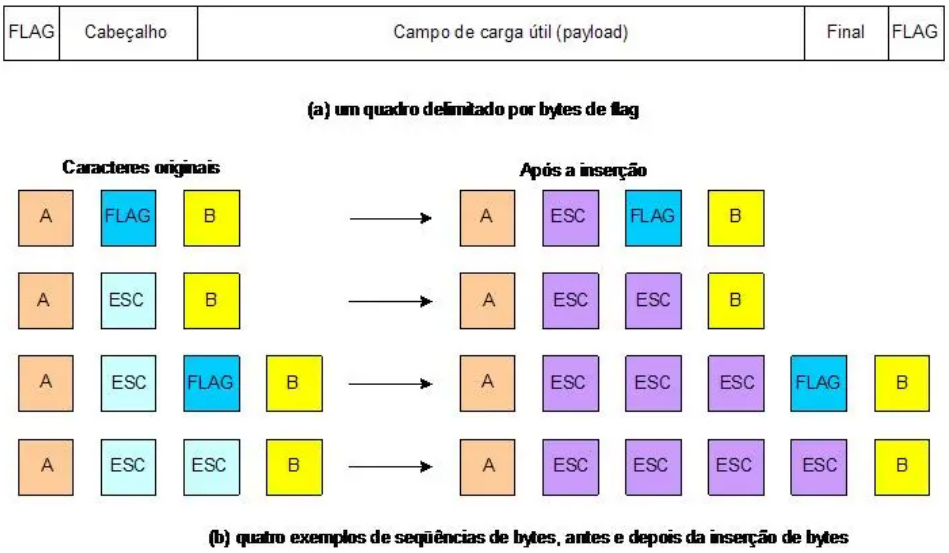

Figure 2. Inserção de bytes ou caracteres

1.3. Bit de paridade par

O protocolo de Bit de Paridade Par é um mecanismo de verificação de erro simples utilizado na comunicação de dados. O objetivo desse protocolo é detectar erros na transmissão de dados. Para isso, ele adiciona um bit adicional à mensagem original, conhecido como bit de paridade, antes de transmitir a mensagem. O bit de paridade é calculado de tal forma que o número total de bits ”1” na mensagem incluindo o bit de paridade seja sempre par. Quando a mensagem é recebida, o receptor verifica o bit de paridade para garantir que a mensagem não tenha sido corrompida durante a transmissão. A vantagem do protocolo de bit de paridade par é sua simplicidade e eficiência na detecção de erros simples, como um bit corrompido. No entanto, ele não é capaz de detectar ou corrigir erros mais complexos, como a perda de múltiplos bits.

Em resumo, o protocolo de bit de paridade par é uma solução simples e eficaz para garantir a integridade dos dados durante a transmissão, especialmente em situações onde a taxa de erro é baixa. No entanto, ele não é uma solução robusta para todas as situações e outros protocolos de verificação de erro, como o cálculo de soma de verificação, podem ser necessários em situações onde a integridade dos dados é crítica.

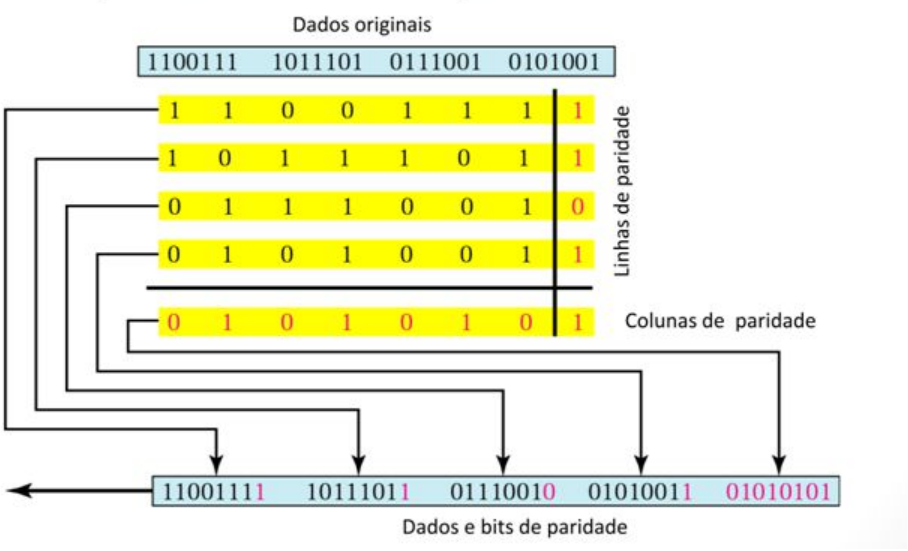

Figure 3. Bit de paridade par

1.4. CRC

O protocolo de Somas de Verificação de Código de Redundância Cíclica (CRC) é um mecanismo de verificação de erro amplamente utilizado em comunicações de dados. Ele é projetado para detectar erros na transmissão de dados, mesmo em situações onde a taxa de erro é alta. O protocolo de CRC funciona calculando um valor de soma de verificação para a mensagem original e adicionando esse valor aos dados antes de transmiti-los. O receptor então verifica a soma de verificação recebida contra a mensagem original para garantir que não houve erros na transmissão. A vantagem do protocolo de CRC é sua capacidade de detectar uma ampla gama de erros, incluindo erros de bit único, perda de bits e erros de transmissão de múltiplos bits. Além disso, ele é escalável e pode ser facilmente ajustado para atender às necessidades de uma rede em constante mudança.

Em resumo, o protocolo de CRC é uma solução robusta e eficaz para garantir a integridade dos dados durante a comunicação de rede. Ele é amplamente utilizado em aplicações críticas, como sistemas de comunicação de dados militares e de saúde, devido à sua capacidade de detectar erros com alta taxa de precisão.

Código CRC	Polinômio Gerador	Aplicações	Detecção de erros
CRC-16	$X^{16} + x^{15} + x^2 + 1$	Em sistemas síncronos, possuidores de um comprimento de caractere igual a 8 bits.	Detecta até 16 erros simultâneos e 99% dos casos de erros simultâneos maiores que 16.
CRC-12	$X^{12} + x^{11} + x^3 + x^2 + x + 1$	Em sistemas síncronos, possuidores de um comprimento igual a 6 bits.	Detecta até 12 erros simultâneos.
CRC-UTI	$X^{16} + x^{12} + x^5 + 1$	Em sistemas síncronos, possuidores de um comprimento de caractere igual a 8 bits	Detecta até 16 erros simultâneos e 99% dos casos de erros simultâneos maiores que 12.

Figure 4. CRC

1.5. Código de Hamming

O protocolo de Hamming é uma técnica de correção de erros amplamente utilizada em comunicações de dados. Ele foi desenvolvido para detectar e corrigir erros de transmissão de dados, garantindo a integridade dos dados durante a comunicação. O protocolo de Hamming funciona adicionando bits extras conhecidos como bits de redundância à mensagem original. Esses bits são calculados de tal forma que eles permitem ao receptor detectar e corrigir erros na mensagem recebida. Quando a mensagem é recebida, o receptor verifica os bits de redundância para determinar se houve erro na transmissão. Se for detectado um erro, ele pode ser corrigido usando informações contidas nos bits de redundância. A vantagem do protocolo de Hamming é sua capacidade de detectar e corrigir erros de transmissão de dados, garantindo a integridade dos dados durante a comunicação. Além disso, ele é escalável e pode ser ajustado para atender às necessidades de diferentes aplicações e tipos de erro.

Em resumo, o protocolo de Hamming é uma solução eficaz e confiável para garantir a integridade dos dados durante a comunicação de dados. Ele é amplamente utilizado em aplicações críticas, como sistemas de comunicação militares e financeiros, devido à sua capacidade de detectar e corrigir erros com alta taxa de precisão.

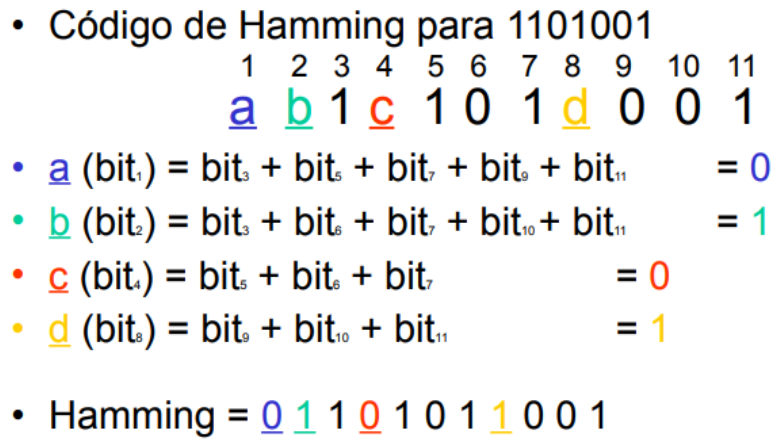

Figure 5. Hamming

2. Implementação

O código é executado pelo terminal, quando aberto, é pedido ao usuário que ele escreva uma mensagem, que representa a aplicação transmissora, depois o programa pede para que o usuário escolha um dos dois tipos de enquadramento (Contagem de caracteres ou inserção de bytes), através de uma numeração de zero a um. Logo em seguida o programa pede para que o usuário escolha um dos três tipos de detecção/correção de erro (Bit por paridade par, CRC ou código de Hamming), através de uma numeração de zero a dois. Uma vez que o tipo de codificação é escolhido, tanto a mensagem quanto a escolha da codificação é enviado para a parte do programa responsável pela camada de aplicação transmissora, que transforma a mensagem de uma string para um vetor de inteiros que corresponde a mensagem escrita em ASCii. Deve-se também escolher uma porcentagem para que ocorram 'erros' durante a transmissão.

2.1. Contagem de caracteres

O Código de Contagem de Caracteres funciona adicionando um número de contagem de caracteres à mensagem original antes de transmiti-la. Esse número representa o número de caracteres na mensagem original e é usado pelo receptor para verificar se todos os caracteres foram recebidos corretamente. Se houver algum erro na transmissão, o número de contagem de caracteres não corresponderá ao número real de caracteres na mensagem recebida, indicando um erro na transmissão. Visualização do enquadramento por contagem de caracteres:

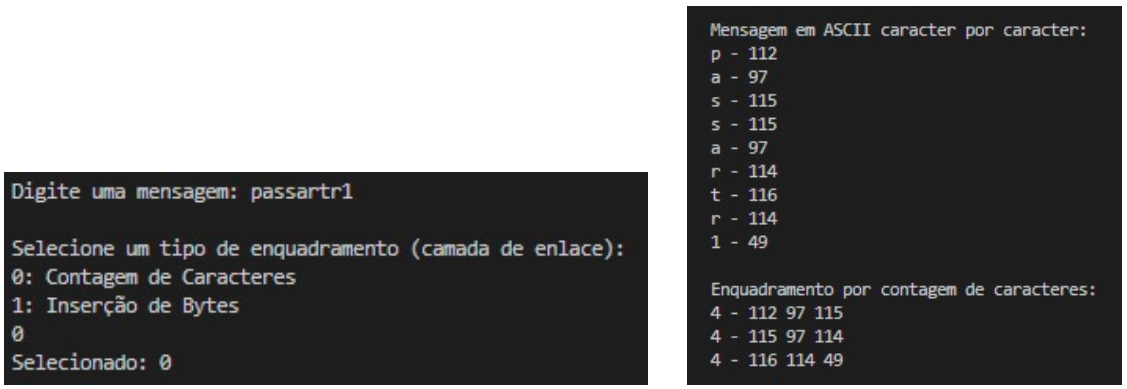

Figure 6. Contagem de caracteres

2.2. Inserção de bytes

Ele funciona adicionando "bytes de verificação" à mensagem original antes da transmissão. Esses bytes são gerados a partir dos dados da mensagem original e são usados pelo receptor para verificar se todos os dados foram recebidos corretamente. Se houver erro na transmissão, os bytes de verificação não corresponderão aos dados recebidos, indicando um erro na transmissão. Quando isso ocorre é mostrado no terminal uma mensagem de erro.

2.3. Bit de paridade par

No código de bit de paridade par, o emissor adiciona um bit adicional à mensagem original, definindo-o como 1 se o número total de bits 1 na mensagem for ímpar, ou 0

se for par. Em seguida, a mensagem é transmitida. No receptor, o bit de paridade é verificado comparando-se com o resultado esperado. Se houver discrepância, é indicativo de um erro na transmissão. Quando isso ocorre é mostrado no terminal uma mensagem de erro.

2.4. CRC

O funcionamento do código de detecção de erros Cyclic Redundancy Check (CRC) é baseado em adicionar informação redundante aos dados a serem transmitidos, permitindo que erros sejam detectados e corrigidos na recepção.Ao transmitir um dado, o remetente divide o dado pelo gerador (uma sequência de bits pré-determinada) e adiciona o resto da divisão ao final do dado transmitido. Na recepção, o destinatário divide o dado recebido pelo mesmo gerador e verifica se o resto da divisão é zero. Se for zero, significa que não houve erro na transmissão e o dado pode ser considerado válido. Caso contrário, o destinatário sabe que houve erro na transmissão e pode tomar medidas para corrigir o erro ou descartar o dado. Caso seja detectada uma inconsistência na mensagem recebida é mostrado no terminal uma mensagem de erro.

2.5. Código de Hamming

O funcionamento do código de correção de erros Hamming em C++ é baseado em adicionar informação redundante aos dados a serem transmitidos, permitindo que erros sejam detectados e corrigidos na recepção. Segue exemplo do envio da mensagem ”abobora” enviada com contagem de caracteres:

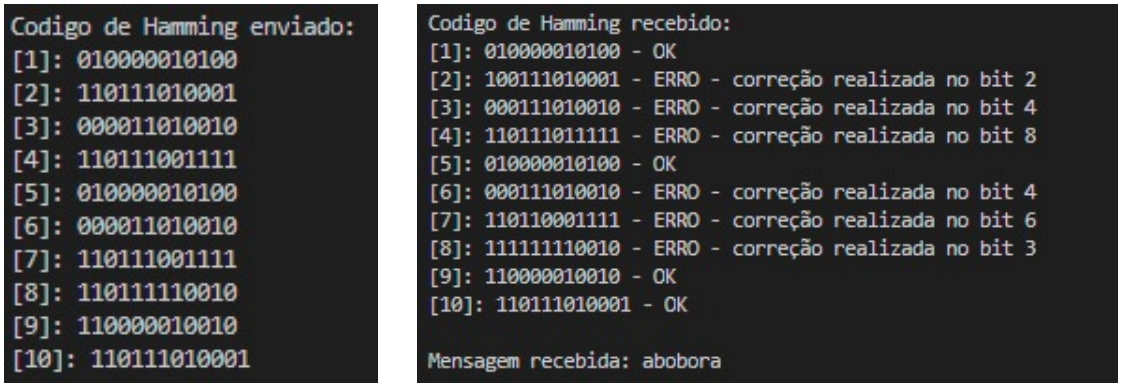

Figure 7. Correção de erro pelo código de Hamming

Porém existem casos que o código não é capaz de corrigir a mensagem por limitações do próprio código de Hamming, como por exemplo neste caso onde ocorrem mais de um erro por byte recebido:

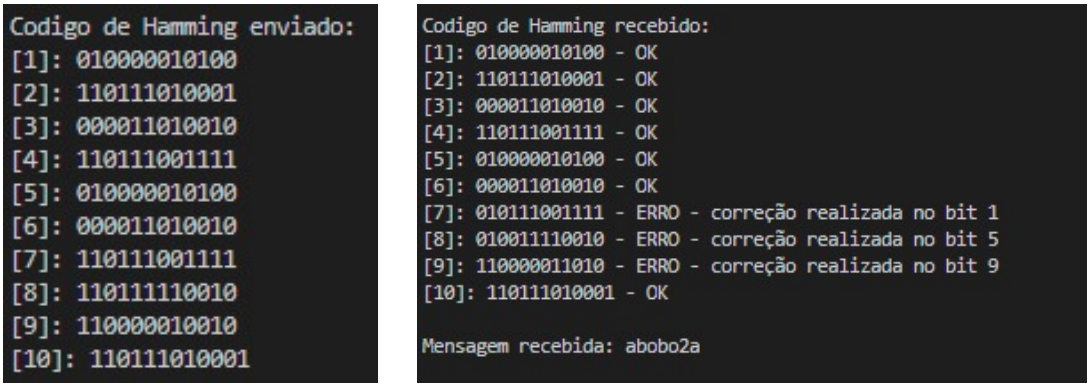

Figure 8. Falha na correção de erro pelo código de Hamming

Agora somente para exemplificar o funcionamento utilizando o método de inserção de bytes, segue o mesmo exemplo aplicado por essa opção:

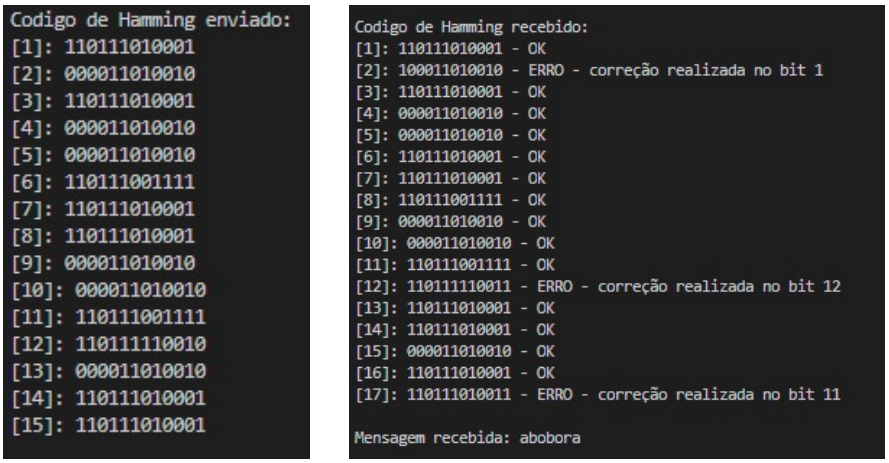

Figure 9. Correção de erro pelo código de Hamming

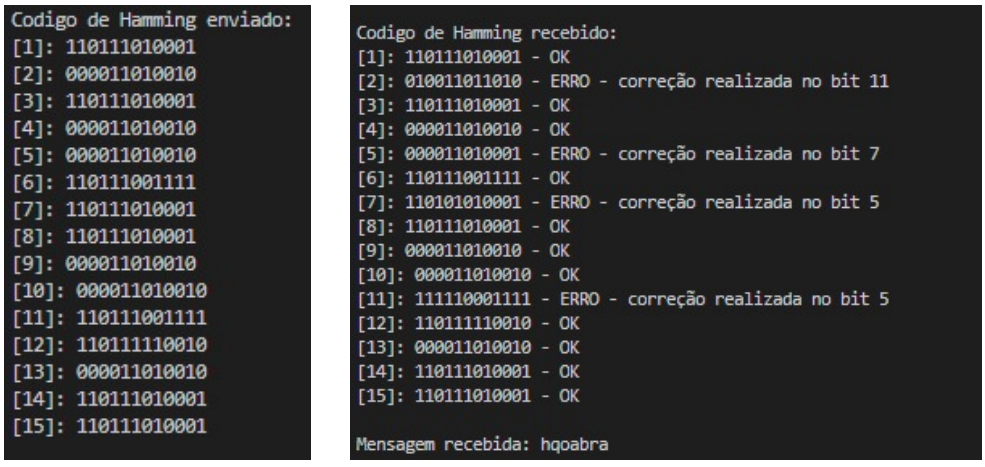

Figure 10. Falha na correção de erro pelo código de Hamming

3. Participação dos membros

Nós trabalhamos de forma coordenada na elaboração do código e do relatório, de maneira que todos foram responsáveis de forma substancial para realização das partes.

4. Conclusão

Em conclusão, diante do exposto no trabalho, a simulação da camada enlace de rede embora sem utilidade prática, serviu para compreender e avaliar a eficiência e desempenho de diferentes protocolos de codificação de sinal. Neste trabalho, foi explorado o funcionamento dos protocolos de Contagem de caracteres, Inserção de bytes ou caracteres, Bit de paridade par, CRC e Hamming, bem como uma breve exposição de suas utilidades.

É importante destacar que cada protocolo tem suas vantagens e desvantagens e é importante escolher o protocolo mais adequado às necessidades específicas de cada aplicação. Em suma, esta simulação foi um trabalho proveitoso para pormos em prática os conhecimentos adquiridos em sala de aula.

5. Referências

Video aulas e slides disponibilizados pelo professor durante o semestre letivo