

Trabalho Prático: Camada de Enlace de Dados, Controle de Erro

Teleinformática e Redes 1

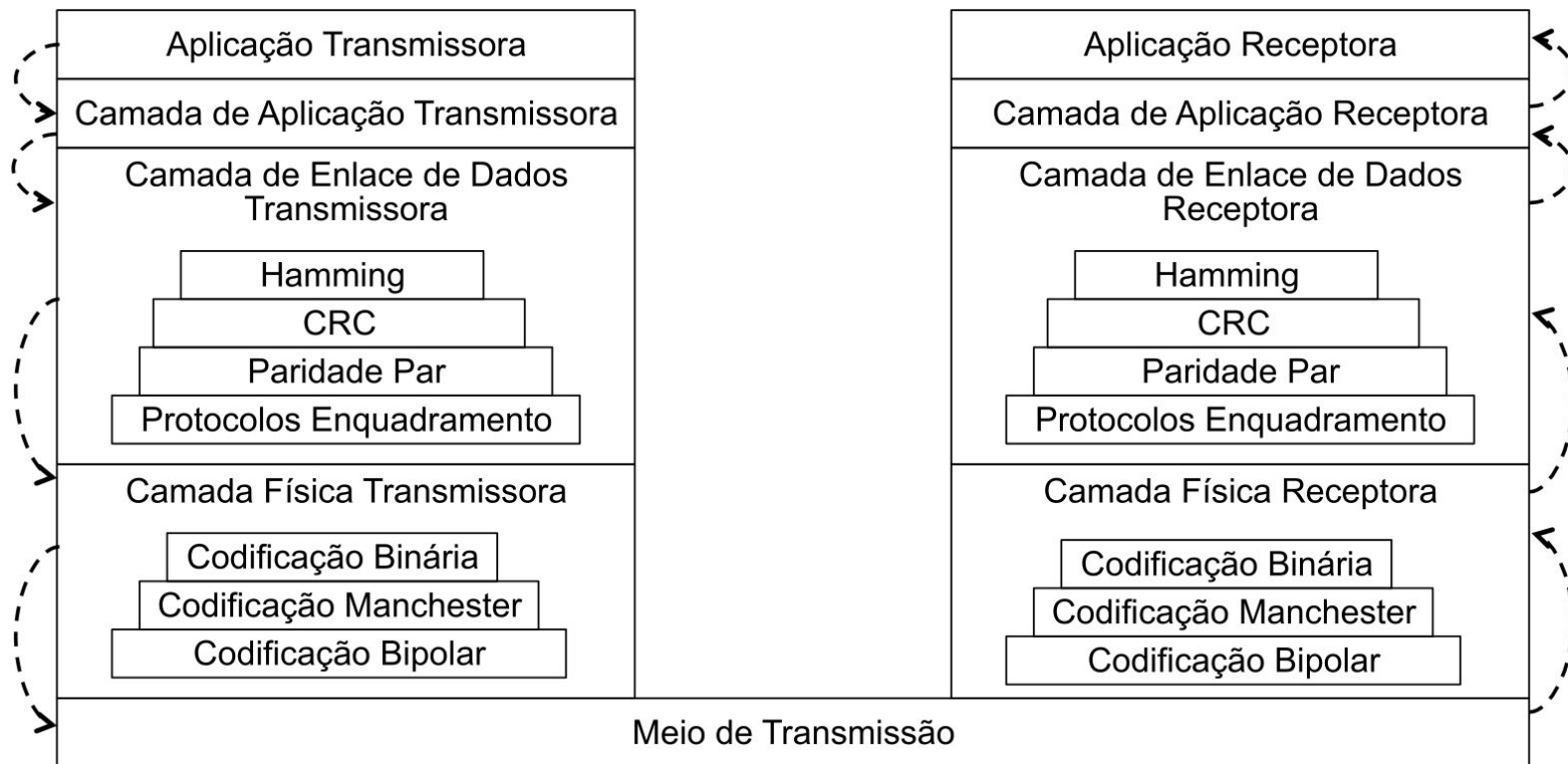


Departamento de Ciência da Computação
Universidade de Brasília

Descrição

- Acrescentar ao código “Simulador de redes” os protocolos vistos para o controle de erros da informação:
 - Hamming

Camada de Enlace de Dados - CONTROLE DE ERRO



Transmissão

Camada de Enlace de Dados, Controle de Erro - Transmissão

```
void CamadaEnlaceDadosTransmissora (int quadro []) {  
    //codigo aqui  
} //fim do metodo CamadaEnlaceDadosTransmissora
```

```
void CamadaEnlaceDadosTransmissoraEnquadramento (int quadro []) {  
    //codigo aqui  
} //fim do metodo CamadaEnlaceDadosTransmissoraEnquadramentos
```

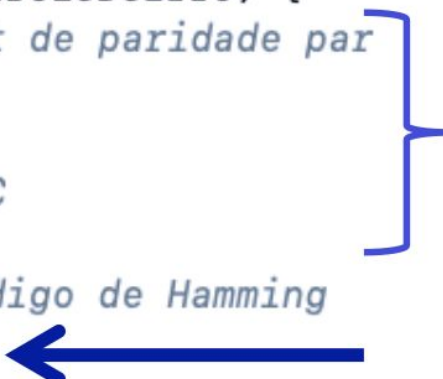
```
void CamadaEnlaceDadosTransmissoraControleDeErro (int quadro []) {  
    //codigo aqui  
} //fim do metodo CamadaEnlaceDadosTransmissoraControleDeErro
```

Camada de Enlace de Dados, Controle de Erro - Transmissão

```
void CamadaEnlaceDadosTransmissora (int quadro []) {  
    CamadaDeEnlaceTransmissoraEnquadramento(quadro);  
    CamadaDeEnlaceTransmissoraControleDeErro(quadro);  
    //chama proxima camada  
    CamadaFisicaTransmissora(quadro);  
} //fim do metodo CamadaEnlaceDadosTransmissora
```

Camada de Enlace de Dados, Controle de Erro - Transmissão

```
void CamadaEnlaceDadosTransmissoraControleDeErro (int quadro []) {  
    int tipoDeControleDeErro = 0; //alterar de acordo com o teste  
  
    switch (tipoDeControleDeErro) {  
        case 0 : //bit de paridade par  
                //codigo  
                break;  
        case 1 : //CRC  
                //codigo  
        case 2 : //codigo de Hamming  
                //codigo  
                break;  
    }//fim do switch/case  
  
}//fim do metodo CamadaEnlaceDadosTransmissoraControleDeErro
```



Feito na subetapa anterior

Camada de Enlace de Dados, Controle de Erro - Transmissão

```
void CamadaEnlaceDadosTransmissoraControleDeErroBitParidadePar (int
    quadro []) {
    //implementacao do algoritmo
} //fim do metodo
    CamadaEnlaceDadosTransmissoraControleDeErroBitParidadePar

void CamadaEnlaceDadosTransmissoraControleDeErroCRC (int quadro []) {
    //implementacao do algoritmo
    //usar polinomio CRC-32(IEEE 802)
} //fim do metodo CamadaEnlaceDadosTransmissoraControleDeErroCRC


void CamadaEnlaceDadosTransmissoraControleDeErroCodigoDeHamming (int
    quadro []) {
    //implementacao do algoritmo
} //fim do metodo
    CamadaEnlaceDadosTransmissoraControleDeErroCodigoDehamming
```


MEIO DE COMUNICAÇÃO - ALTERADO!

Alterar o meio de comunicação

PROVOCAR O ERRO

```
void MeioDeComunicacao (int fluxoBrutoDeBits []) {  
    //OBS: trabalhar com BITS e nao com BYTES!!!  
    int erro, porcentagemDeErros;  
    int fluxoBrutoDeBitsPontoA [], fluxoBrutoDeBitsPontoB [];  
  
    porcentagemDeErros = 0; //10%, 20%, 30%, 40%, ..., 100%  
    fluxoBrutoDeBitsPontoA = fluxoBrutoDeBits;  
  
    while (fluxoBrutoDeBitsPontoB.lenght != fluxoBrutoDeBitsPontoA){  
        if ((rand()%100)== ... ) //fazer a probabilidade do erro  
            fluxoBrutoBitsPontoB += fluxoBrutoBitsPontoA; //BITS!!!  
        else //ERRO! INVERTER (usa condicao ternaria)  
            fluxoBrutoBitsPontoB==0 ?  
                fluxoBrutoBitsPontoA=fluxoBrutoBitsPontoB++ :  
                fluxoBrutoBitsPontoA=fluxoBrutoBitsPontoB--;  
    }  
} //fim do while  
} //fim do metodo MeioDeTransmissao
```



Recepção

Camada de Enlace de Dados, Controle de Erro - Recepção

```
void CamadaEnlaceDadosReceptora (int quadro []) {  
    //codigo aqui  
} //fim do metodo CamadaEnlaceDadosReceptora
```

```
void CamadaEnlaceDadosReceptoraEnquadramento (int quadro []) {  
    //codigo aqui  
} //fim do metodo CamadaEnlaceDadosReceptoraEnquadramento
```


```
void CamadaEnlaceDadosReceptoraControleDeErro (int quadro []) {  
    //codigo aqui  
} //fim do metodo CamadaEnlaceDadosReceptoraControleDeErro
```

Camada de Enlace de Dados, Controle de Erro - Recepção

```
void CamadaEnlaceDadosReceptora (int quadro []) {  
  
    CamadaDeEnlaceReceptoraEnquadramento(quadro);  
  
    CamadaDeEnlaceReceptoraControleDeErro(quadro);  
  
    //chama proxima camada  
    CamadaDeAplicacaoReceptora(quadro);  
  
} //fim do metodo CamadaEnlaceDadosReceptora
```

Camada de Enlace de Dados, Controle de Erro - Recepção

```
void CamadaEnlaceDadosReceptoraControleDeErro (int quadro []) {  
    int tipoDeControleDeErro = 0; //alterar de acordo com o teste  
  
    switch (tipoDeControleDeErro) {  
        case 0 : //bit de paridade par  
                //codigo  
                break;  
        case 1 : //CRC  
                //codigo  
                break;  
        case 2 : //codigo de hamming  
                //codigo  
                break;  
    }//fim do switch/case  
  
} //fim do metodo CamadaEnlaceDadosReceptoraControleDeErro
```



Feito na subetapa anterior

Camada de Enlace de Dados, Controle de Erro - Recepção

```
void CamadaEnlaceDadosReceptoraControleDeErroBitDeParidadePar (int
    quadro []) {
    //implementacao do algoritmo para VERIFICAR SE HOUVE ERRO
} //fim do metodo
    CamadaEnlaceDadosReceptoraControleDeErroBitDeParidadePar

void CamadaEnlaceDadosReceptoraControleDeErroCRC (int quadro []) {
    //implementacao do algoritmo para VERIFICAR SE HOUVE ERRO //usar
    polinomio CRC-32(IEEE 802)
} //fim do metodo CamadaEnlaceDadosReceptoraControleDeErroCRC
```

```
void CamadaEnlaceDadosReceptoraControleDeErroCodigoDeHamming (int
    quadro []) {
    //implementacao do algoritmo para VERIFICAR SE HOUVE ERRO
} //fim do metodo CamadaEnlaceDadosReceptoraControleDeErroCodigoDeHamming
```