Camada de Enlace de Dados

Sumário

- Introdução;
- Serviços Oferecidos à Camada de Rede;
- Enquadramento;
- Detecção e Correção de Erros;
  - Correção de Erros
- Protocolos Elementares de Enlace de Dados
  - Simplex sem restrições
  - Simplex stop-and-wait
  - Simplex para um canal com ruído

Sumário

- Protocolos de Janela Deslizante:
  - Janela deslizante de um bit;
  - Go back n
  - Retransmissão seletiva;
- Point-to-Point_Protocol (PPP)
- Bibliografia

Introdução

- Na camada de Enlace de Dados trataremos de algoritmos que permitem uma comunicação eficiente e confiável entre dois computadores adjacentes;
  - Por adjacentes, queremos dizer que as duas máquinas estão fisicamente conectadas por meio de um canal de comunicação;

- Em princípio, podemos imaginar que é trivial bits saírem de uma máquina A e serem entregues na ordem exata e que são enviados à máquina B;

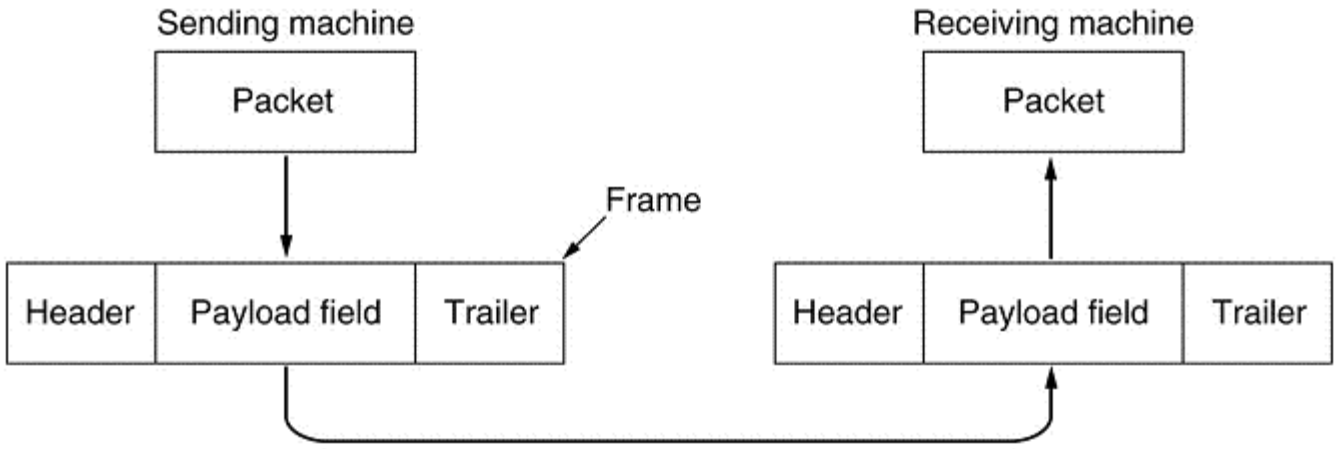
- Infelizmente, os circuitos de comunicação produzem erros ocasionais;

Introdução

- Observemos as diversas funções específicas que a camada de enlace executa para prover uma comunicação de qualidade:
  - Tratamento de erros na transmissão:
  - Controlar o fluxo de dados: evitar que receptores lentos não sejam atropelados por transmissões rápidas
  - Fornecer uma interface de serviço bem definida à camada de rede

Introdução

- Para alcançar esses objetivos, a camada de enlace de dados recebe os pacotes da camada de rede e os encapsula em quadros de transmissão:



  - Cada quadro contém um cabeçalho (header), um campo de carga util, que conterá o pacote, e um final (trailer) de quadro;

Serviços Oferecidos à Camada de Rede

- A camada de enlace de dados pode ser projetada de modo a oferecer diversos serviços, que podem variar de sistema para sistema.

- Três possibilidades razoáveis oferecidas com freqüência são:
  - Serviço sem conexão e sem confirmação:
  - Serviço sem conexão e com confirmação:
  - Serviço orientado a conexões com confirmação

Serviços Oferecidos à Camada de Rede

- Serviço sem conexão e sem confirmação:
  - Consiste em fazer a máquina de origem enviar quadros independentes à máquina de destino, sem que a máquina de destino confirme o recebimento desses quadros.
  - Nenhuma conexão é estabelecida antes ou liberada depois do processo;
  - Se o quadro for perdido devido à ruídos na linha, não haverá nenhuma tentativa de detectar a perda ou de recuperá-lo na camada de enlace de dados;

Serviços Oferecidos à Camada de Rede

- Serviço sem conexão e sem confirmação:
  - Essa classe de serviço é apropriada quando a taxa de erros é muito baixa e a recuperação fica a cargo de camadas mais altas;
  - Ela também é apropriada para o tráfego em tempo real, no qual, os dados atrasados causam mais problemas que dados recebidos com falhas;
  - A maior parte das LANs utiliza serviço sem conexão e sem confirmação na camada de enlace de dados;

Serviços Oferecidos à Camada de Rede

- Serviço sem conexão e com confirmação:
  - Quando esse serviço é oferecido, ainda não há conexões lógicas sendo usadas, mas cada quadro enviado é individualmente confirmado;
  - Caso um quadro não tenha chegado dentro de um intervalo de tempo específico, ele poderá ser enviado outra vez.
  - Esse serviço é útil em canais não-confiáveis, como os sistemas sem fio;

Serviços Oferecidos à Camada de Rede

- Serviço sem conexão e com confirmação:
  - Mas a camada de rede pode confirmar pacotes individualmente?
  - O que seria melhor, confirmar quadros individualmente ou pacotes?
    - Os quadros têm um comprimento máximo restrito imposto pelo hardware, o que não ocorre com os pacotes da camada de rede;
    - Por exemplo, se o pacote médio for subdividido em, 10 quadros, e 20% de todos os quadros forem perdido, o tempos necessário para efetivar a transmissão do pacote com sucesso poderá ser muito longo;

Serviços Oferecidos à Camada de Rede

- Serviço orientado à conexão com confirmação:
  - As máquina estabelecem uma conexão antes dos dados serem transferidos;
  - Cada quadro enviado pela conexão é numerado e a camada de enlace de dados garante que cada quadro será de fato recebido;
  - Após a entrega dos quadros, a conexão é desfeita e todos os recursos usados para mantê-la são desfeitos (buffers, variáveis, etc)

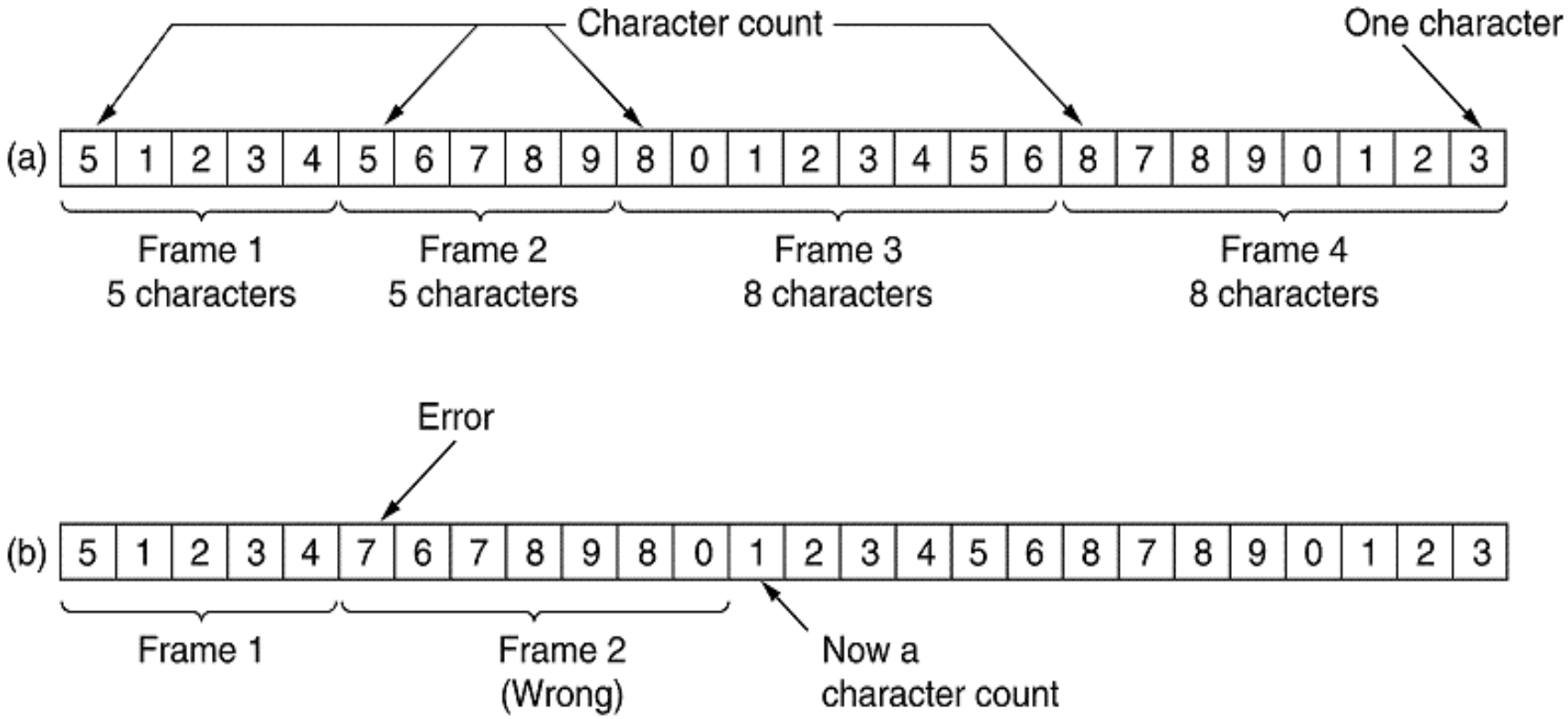
Enquadramento

- A divisão do fluxo de bits em quadros é mais difícil do que parece à primeira vista;

- Como é muito arriscado contar com a temporização para marcar o início e o fim de cada quadro, outros métodos foram criados:
  - Contagem de caracteres
  - Bytes de flags, com inserção de bytes
  - Flags iniciais e finais, com inserção de bits
  - Violação de codificação da camada física

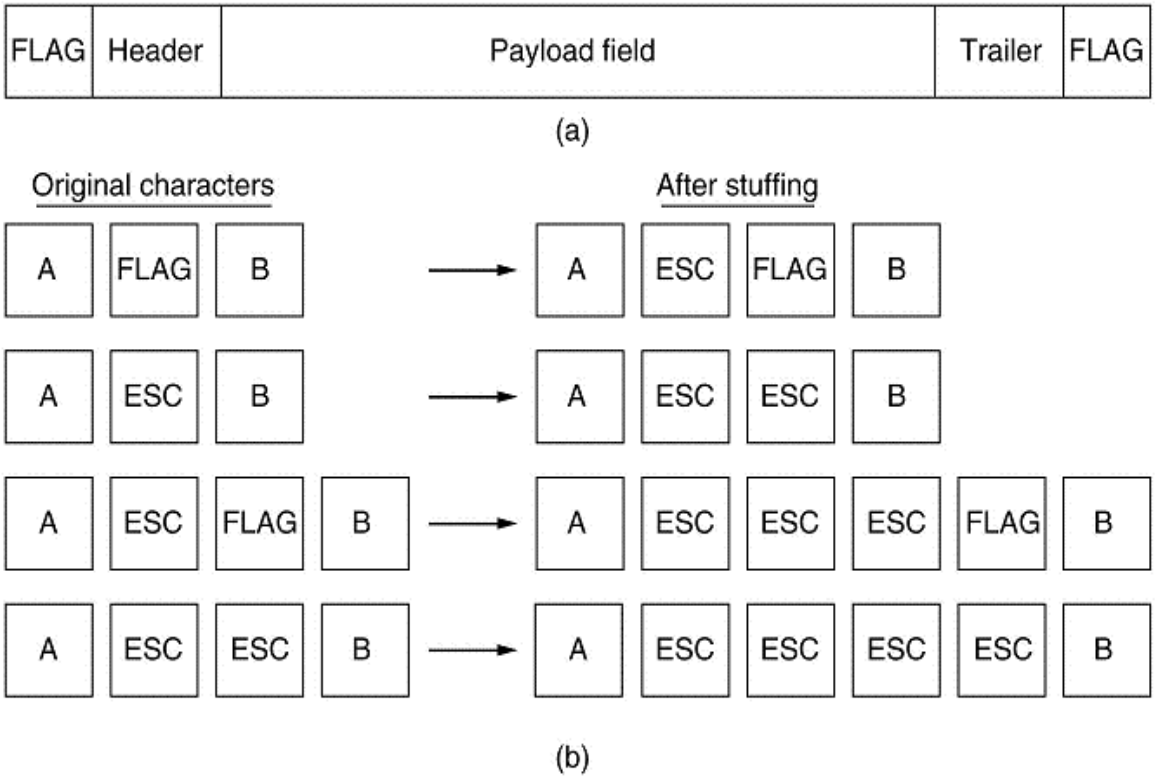
Enquadramento

- Contagem de caracteres:



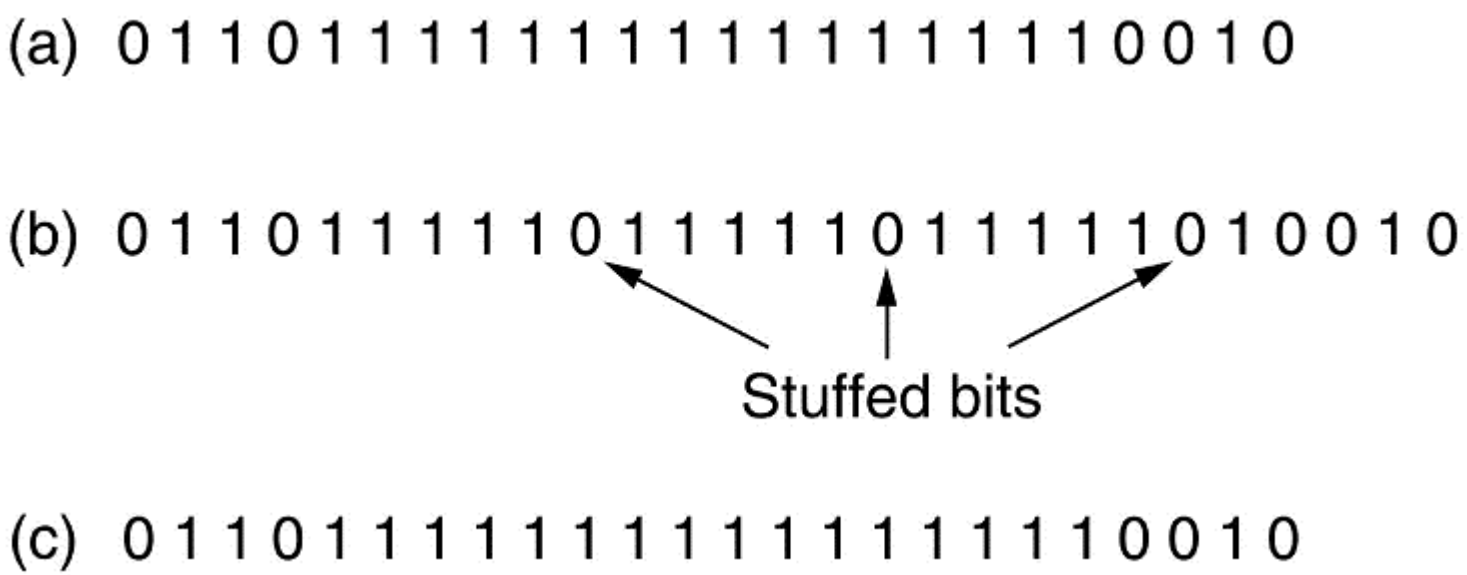
Enquadramento

- Byte de flag:



Enquadramento

- Inserção de caracteres:



Detecção e Correção de Erros

- Os projetistas de redes desenvolveram duas estratégias básicas para tratar erros:
  - Correção de erros: Incluir informações redundantes suficientes em cada bloco de dados enviados para corrigir o erro;
  - Detecção de erros: Incluir redundância suficiente apenas para permitir que o receptor deduza que houve um erro, mas sem identificar qual, e solicite uma retransmissão;

- Em canais altamente confiáveis, como os de fibra, é mais econômico utilizar um código de detecção;
  - Em canais que geram muitos erros, como enlaces sem fio, é melhor descobrir qual era o bloco original;

Detecção e Correção de Erros

- Como um exemplo simples de código de detecção de erros, imagine um código no qual um único bit de paridade é acrescentado aos dados;

- O bit de paridade é escolhido de forma que o número de bits 1 da palavra de código seja par (ou ímpar);
  - Por exemplo, quando 1011010 é enviado com paridade par, é acrescentado um bit ao final para formar 10110100
  - Com paridade ímpar, 1011010 passa a ser 10110101;
  - Detecção muito simples de erros;

Correção de Erros

- Outro método mais eficiente foi criado por Hamming (1950). É gerada uma palavra síndrome, onde o ideal é que:
  - Se todos os bits da palavras síndrome têm valor 0s, não ocorreu nenhum erro;
  - Se a palavra síndrome contém apenas um bit com valor 1, ocorreu erro em um dos bits de teste;
  - Se a palavra síndrome contém mais de um bit com valor 1, o valor numérico da palavra síndrome indica a posição do bit em que ocorreu erro;

Correção de Erros

- Para satisfazer essas características, os bits de dados e de teste são organizados em uma palavra de 12 bits;
  - As posições de bits cujo número é uma potência de 2 são reservadas como bits de teste, vejamos:

Posição	12	11	10	9	8	7	6	5	4	3	2	1
Número	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Bit de dados	D8	D7	D6	D5		D4	D3	D2		D1		
Bit de Check					C8				C4		C2	C1

Correção de Erros

- Dessa forma, o bit de correção é calculado efetuando a operação ⊕(ou-exclusivo) sobre os bits de dados que contém o bit de correção, vejamos

  C1 = D1⊕D2⊕ D4⊕D5⊕ D7

  C2 = D1⊕D3⊕ D4⊕D6⊕ D7

  C4 = D2⊕ D3⊕D4 ⊕D8

  C8= D5⊕D6 ⊕D7⊕D8

- Analisemos a situação em que uma palavra de 8 bits dada como entrada é 00111001, com bit de dados M1 na posição mais à direita;

Correção de Erros

- Vejamos os cálculos:

  $C1 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$

  $C2 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$

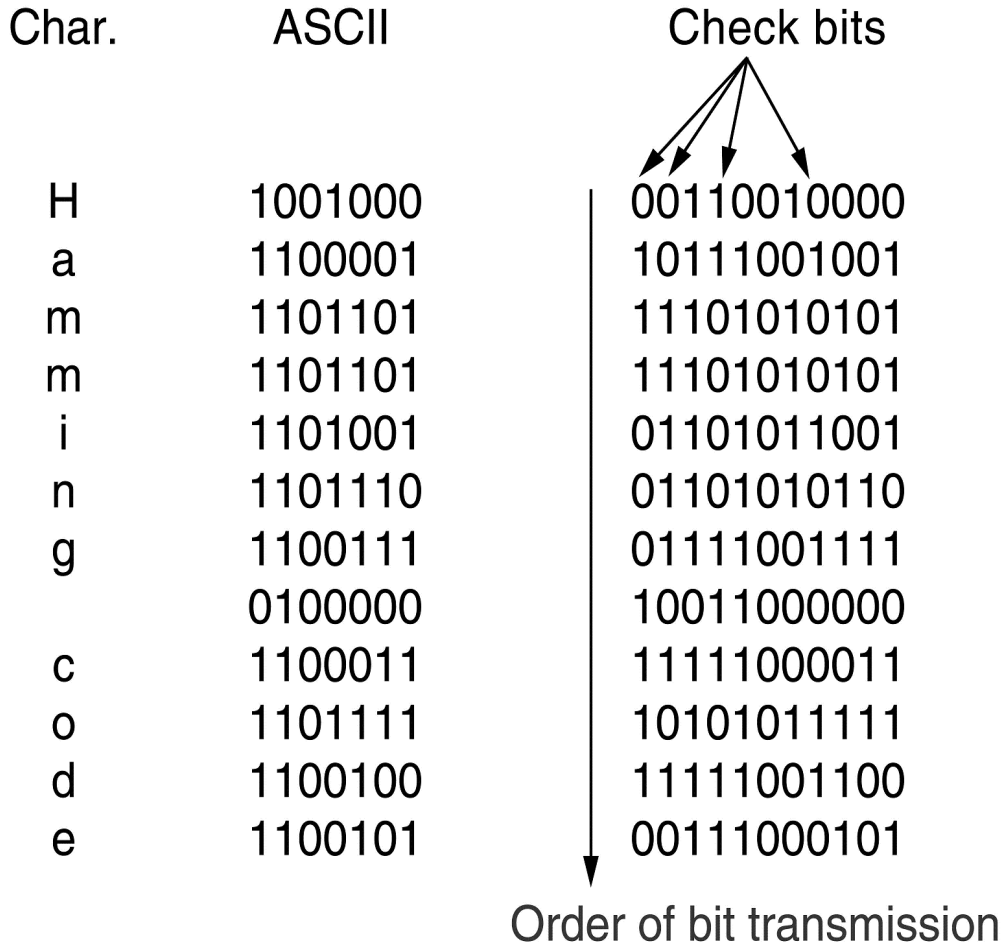
  $C3 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$

  $C4 = 1 \oplus 1 \oplus 0 \oplus 0 = 0$

- Supondo que o terceiro bit de dados (M3) foi alterado de 0 para 1. Os bits serão recalculados na leitura e a palavra síndrome indicará onde ocorreu o erro;

Detecção e Correção de Erros

- A eficiência do código de Hamming pode ser elevada enviando colunas da matriz de caracteres que deseja transmitir, veja:



Detecção e Correção de Erros

- Entre os métodos de detecção de erros, o mais difundido é o código polinomial também conhecido como código de redundância cíclica (Cyclic Redundancy Check – CRC);

- Quando o método do código polinomial é empregado, o transmissor e o receptor devem concordar em relação a um polinômio gerador, G(x), antecipadamente;

- Para calcular o total de verificação (checksum) de um quadro com m bits, que corresponde ao polinômio M(x), o quadro deve ter mais bits do que o polinômio gerador;

Detecção e Correção de Erros

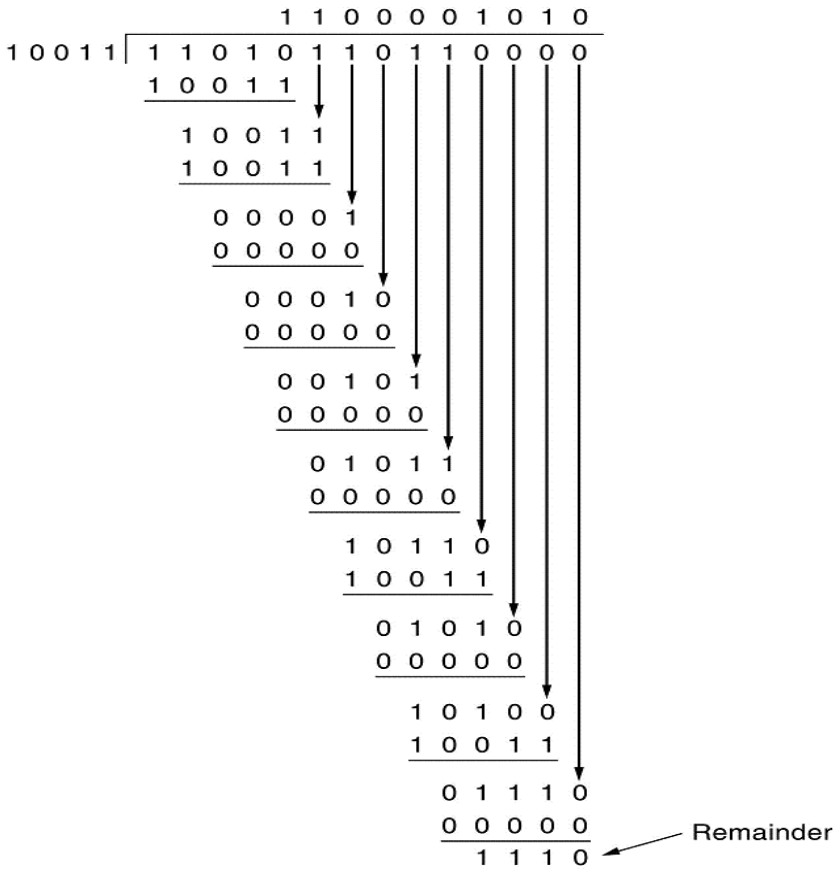
- O algoritmo para calcular o total de verificação é:
  - Seja r o grau de G(x). Acrescente r bits zero à extremidade de baixa ordem do quadro, de modo que ele passe a conter m + r bits e corresponda ao polinômio $x^r M(x)$.
  - Divida a string de bits correspondente a $x^r M(x)$ pelo string de bits correspondente a G(x) utilizando a divisão de módulo 2;
  - Subtraia o resto (que tem sempre r ou menos bits) do string de bits correspondente a $x^r M(x)$ utilizando a subtração de módulo 2. O resultado é o quadro verificado pela soma que deverá ser transmitido. Chame o polinômio de T(x).
  - A próxima figura ilustra o cálculo referente a um quadro de 1101011011, usando o gerador $G(x) = x^4 + x + 1$
  - OBS: Inserir nos slides o exemplo de módulo de 2

Detecção e Correção de Erros

Frame : 1 1 0 1 0 1 1 0 1 1

Generator: 1 0 0 1 1

Message after 4 zero bits are appended: 1 1 0 1 0 1 1 0 1 1 0 (0



Transmitted frame: 1 1 0 1 0 1 1 0 1 1 1 1 1 0

Protocolos Elementares de Enlace de Dados

- Como uma introdução ao estudo dos protocolos, vamos começar examinando três protocolos com grau de complexidade crescente:
  - Protocolo simplex sem restrições;
  - Protocolo simples stop-and-wait;
  - Protocolo simplex para um canal com ruído

- Antes, porém, é útil tornar explícitas algumas das suposições nas quais se baseia o modelo de comunicação;

Protocolos Elementares de Enlace de Dados

- Algumas definições utilizadas nos protocolos a seguir:

```
#define MAX_PKT 1024                          /* determines packet size in bytes */

typedef enum {false, true} boolean;           /* boolean type */
typedef unsigned int seq_nr;                  /* sequence or ack numbers */
typedef struct {unsigned char data[MAX_PKT];} packet;/*   packet definition */
typedef enum {data, ack, nak} frame_kind;     /* frame_kind definition */

typedef struct {                              /* frames are transported in this layer */
  frame_kind kind;                            /* what kind of a frame is it? */
  seq_nr seq;                                 /* sequence number */
  seq_nr ack;                                 /* acknowledgement number */
  packet info;                                /* the network layer packet */
} frame;
```

Protocolos Elementares de Enlace de Dados

- Considerações:
  - *seq_nr* é um inteiro pequeno usado para numerar os quadros, o que facilita sua destinação;
  - Os números de seqüência variam de 0 até MAX_SEQ (inclusive)
  - Um *packet* é a unidade de informação trocada entre a camada de rede e a camada de enlace de dados da mesma máquina
  - Um *frame* é composto por quatro campos: *kind, seq, ack e info*; os três primeiros contêm informações de controle, e o último os dados reais a serem transferidos;
  - *kind* indica se há dados no quadro;
  - *seq e ack* são usados para números de seqüência e confirmações

Protocolos Elementares de Enlace de Dados

- Procedimentos dependentes da implementação:

```
/* Wait for an event to happen; return its type in event. */
void wait_for_event(event_type *event);

/* Fetch a packet from the network layer for transmission on the channel. */
void from_network_layer(packet *p);

/* Deliver information from an inbound frame to the network layer. */
void to_network_layer(packet *p);

/* Go get an inbound frame from the physical layer and copy it to r. */
void from_physical_layer(frame *r);

/* Pass the frame to the physical layer for transmission. */
void to_physical_layer(frame *s);

/* Start the clock running and enable the timeout event. */
void start_timer(seq_nr k);

/* Stop the clock and disable the timeout event. */
void stop_timer(seq_nr k);
```

Protocolos Elementares de Enlace de Dados

- Considerações:
  - Inicialmente, o receptor nada tem a fazer. Ele fica à espera de que algo aconteça *wait_for_event(*event)*;
  - Os procedimentos *to e from_network_layer* são usados pela camada de enlace de dados para enviar e aceitar pacotes à camada de rede respectivamente;
  - Os procedimentos *from e to_physical_layer* repassam quadros entre a camada de enlace de dados e a camada física;
  - O procedimento *wait_for event* retorna *event = timeout*. Os procedimentos *start_timer e stop_timer* ativam e desativam o *timer*, respectivamente;

Protocolos Elementares de Enlace de Dados

- Procedimentos dependentes da implementação:

```
/* Start an auxiliary timer and enable the ack_timeout event. */
void start_ack_timer(void);

/* Stop the auxiliary timer and disable the ack_timeout event. */
void stop_ack_timer(void);

/* Allow the network layer to cause a network_layer_ready event. */
void enable_network_layer(void);

/* Forbid the network layer from causing a network_layer_ready event. */
void disable_network_layer(void);

/* Macro inc is expanded in-line: Increment k circularly. */
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```

Protocolos Elementares de Enlace de Dados

- Considerações:
  - Os procedimentos *start e stop_ack_timer* controlam um timer auxiliar cuja função é gerar confirmações sob determinadas condições;
  - Os procedimentos enable e disable_network_layer são usados nos protocolos mais sofisticados para impedir que a camada de rede acabe ficando sobrecarregada com pacotes para os quais não dispõe de espaço no buffer;
  - *MAX_SEQ* tem um valor diferente para os diversos protocolos, portanto, com freqüência é necessário aumentar um número de seqüência em uma unidade

Protocolo Simplex sem Restrições

- O protocolo (utopia) oferece transmissão de dados em um único sentido, do transmissor para o receptor.
- Pressupõe-se que o canal de comunicação é livre de erros e que o receptor é capaz de processar toda a entrada de uma forma infinitamente rápida;
- Conseqüentemente, o transmissor permanece em um loop enviando os dados com maior rapidez possível;

Protocolo Simplex sem Restrições

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender1(void)
{
  frame s;                              /* buffer for an outbound frame */
  packet buffer;                        /* buffer for an outbound packet */

  while (true) {
      from_network_layer(&buffer); /* go get something to send */
      s.info = buffer;                  /* copy it into s for transmission */
      to_physical_layer(&s);            /* send it on its way */
  }                                     /* Tomorrow, and tomorrow, and tomorrow,
                                           Creeps in this petty pace from day to day
                                           To the last syllable of recorded time.
                                               - Macbeth, V, v */
}
```

Protocolo Simplex sem Restrições

- O receptor espera que algum evento aconteça e a única possibilidade é a chegada de um quadro não-danificado;

```
void receiver1(void)
{
  frame r;
  event_type event;                /* filled in by wait, but not used here */

  while (true) {
       wait_for_event(&event);     /* only possibility is frame_arrival */
       from_physical_layer(&r);    /* go get the inbound frame */
       to_network_layer(&r.info);  /* pass the data to the network layer */
```

Protocolo Simplex Stop-and-Wait

- Também implementa um fluxo de dados unidirecional entre o transmissor e o receptor (simplex);
- Presume-se mais uma vez que o canal de comunicação seja totalmente livre de erros;
- No entanto, dessa vez, o receptor tem buffer finito e uma velocidade de processamento finita, vejamos o algoritmo:

Protocolo Simplex Stop-and-Wait

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void)
{
  frame s;                          /* buffer for an outbound frame */
  packet buffer;                    /* buffer for an outbound packet */
  event_type event;                 /* frame_arrival is the only possibility */

  while (true) {
      from_network_layer(&buffer); /* go get something to send */
      s.info = buffer;              /* copy it into s for transmission */
      to_physical_layer(&s);        /* bye-bye little frame */
      wait_for_event(&event);       /* do not proceed until given the go ahead */
  }
}
```

Protocolo Simplex Stop-and-Wait

- Após entregar um pacote à camada de rede, o receptor envia um quadro (sem informação) de confirmação de volta ao transmissor, antes de entrar mais uma vez no loop de espera:

```
void receiver2(void)
{
  frame r, s;                         /* buffers for frames */
  event_type event;                   /* frame_arrival is the only possibility */
  while (true) {
      wait_for_event(&event);         /* only possibility is frame_arrival */
      from_physical_layer(&r);        /* go get the inbound frame */
      to_network_layer(&r.info);      /* pass the data to the network layer */
      to_physical_layer(&s);          /* send a dummy frame to awaken sender */
  }
}
```

Protocolo Simplex para um canal com Ruído

- A primeira vista, podemos acrescentar um timer ao protocolo anterior, certo?
- Problema: Erro no pacote de confirmação e estouro do timer;
- Solução: Inserir número de seqüência em cada quadro recebido
- Vejamos o algoritmo:

Protocolo Simplex para um canal com Ruído

```
void sender3(void)
{
  seq_nr next_frame_to_send;          /* seq number of next outgoing frame */
  frame s;                            /* scratch variable */
  packet buffer;                      /* buffer for an outbound packet */
  event_type event;

  next_frame_to_send = 0;             /* initialize outbound sequence numbers */
  from_network_layer(&buffer);        /* fetch first packet */
  while (true) {
      s.info = buffer;                /* construct a frame for transmission */
      s.seq = next_frame_to_send;     /* insert sequence number in frame */
      to_physical_layer(&s);          /* send it on its way */
      start_timer(s.seq);             /* if answer takes too long, time out */
      wait_for_event(&event);         /* frame_arrival, cksum_err, timeout */
      if (event == frame_arrival) {
          from_physical_layer(&s);    /* get the acknowledgement */
          if (s.ack == next_frame_to_send) {
              stop_timer(s.ack);      /* turn the timer off */
              from_network_layer(&buffer);    /* get the next one to send */
              inc(next_frame_to_send);    /* invert next_frame_to_send */
          }
      }
  }
}
```

Protocolo Simplex para um canal com Ruído

- Depois de transmitir um quadro e ativar o timer, o transmissor espera que algo interessante aconteça. Existem apenas três possibilidades:
  - O quadro de confirmação chegar sem danos: busca o próximo pacote em sua camada de rede e coloca no buffer
  - O quadro de confirmação chegar com erro: o buffer e o número de seqüência permanecem inalterados e uma cópia do quadro poderá ser enviada
  - O timer ser desativado: mesma conseqüência do quadro de confirmação com erro;

Protocolo Simplex para um canal com Ruído

- Quando um quadro válido chega ao receptor, seu número de seqüência é conferido, para verificar se ele é uma cópia;

```
void receiver3(void)
{
  seq_nr frame_expected;
  frame r, s;
  event_type event;

  frame_expected = 0;
  while (true) {
       wait_for_event(&event);          /* possibilities: frame_arrival, cksum_err */
       if (event == frame_arrival) {    /* a valid frame has arrived. */
            from_physical_layer(&r);    /* go get the newly arrived frame */
            if (r.seq == frame_expected) {/* this is what we have been waiting for.
                 to_network_layer(&r.info);    /* pass the data to the network layer
                 inc(frame_expected); /* next time expect the other sequence nr *
            }
            s.ack = 1 – frame_expected;   /* tell which frame is being acked */
            to_physical_layer(&s);       /* send acknowledgement */
       }
  }
}
```

Protocolos de Janela Deslizante

- Nos protocolos apresentados anteriormente, os quadros de dados eram transmitidos apenas em um sentido;
- Em situação mais práticas, há necessidade de transmitir dados em ambos os sentidos (full-duplex);
- Nesse caso, quando um quadro de dados chega a seu destino, em vez de enviar imediatamente um quadro de controle separado, o receptor se contém e espera até a camada de rede enviar o próximo quadro;

Protocolos de Janela Deslizante

- Na verdade, a confirmação pega carona no próximo quadro de dados que estiver sendo enviado;

- A técnica de retardar temporariamente as confirmações e enviá-las junto com o próximo quadro de dados é conhecida como piggybacking (superposição)

Protocolos de Janela Deslizante

- Problema: Quanto tempo a camada de enlace de dados deve esperar por um pacote ao qual deverá acrescentar a confirmação?

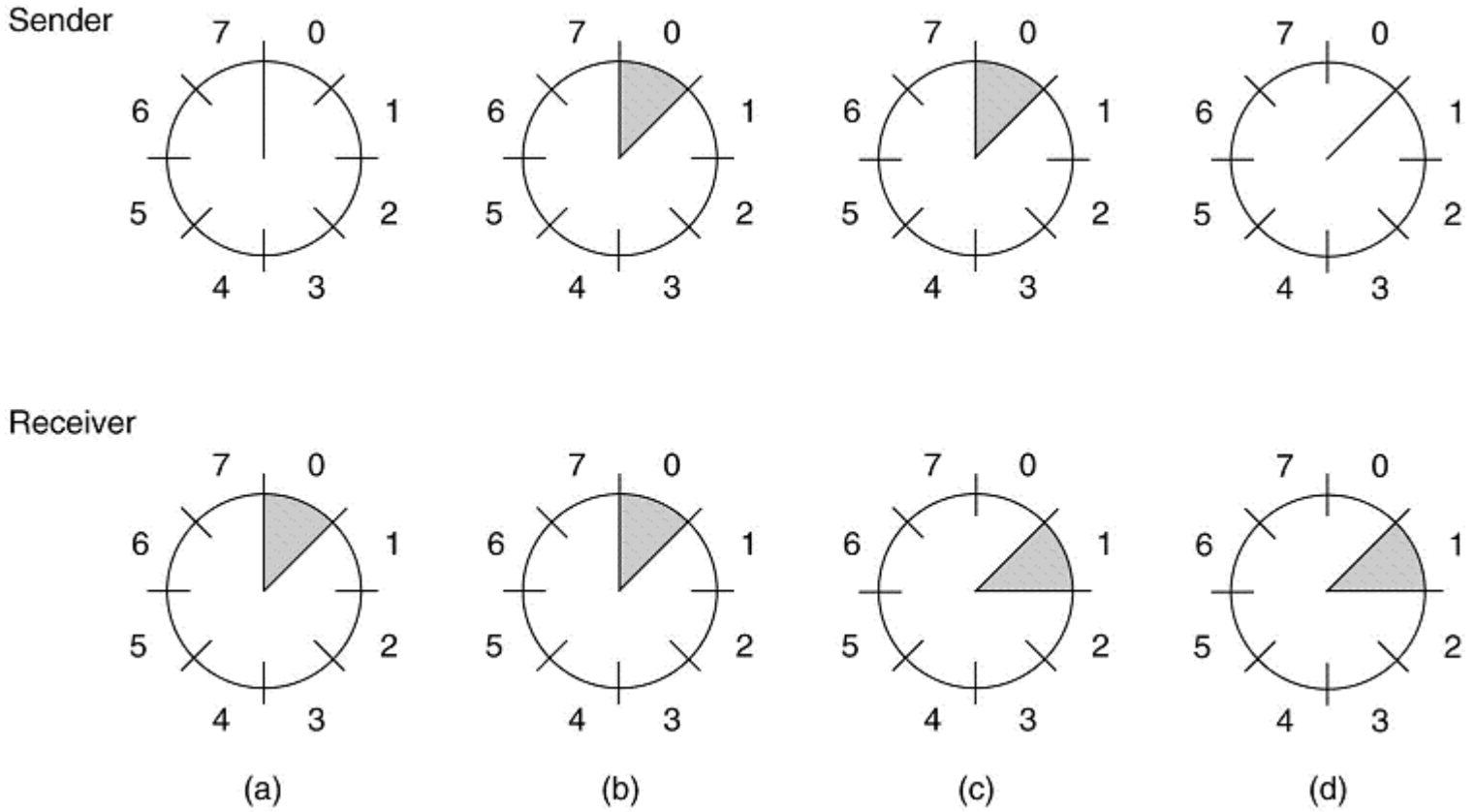
- Como a camada de enlace não pode prever quando o próximo pacote da camada de rede estiver chegando, a solução *é ad hoc*
  - Espera durante um número fixo de milisegundos;
  - Se um novo pacote chegar logo, a confirmação será acrescentada a ela; caso contrário, se nenhum pacote estiver chegado até o final desse intervalo de tempo, a camada de enlace enviará um quadro de confirmação separado;

Protocolos de Janela Deslizante

- Vejamos três protocolos bidirecionais que pertencem a uma classe de protocolos identificados como protocolos de janela deslizante;
- Os três apresentam diferenças em termos de eficiência, complexidade e requisitos de buffer;
- Em todos os protocolos, cada quadro enviado contém um número de seqüência, variando desde 0 ate algum valor máximo;

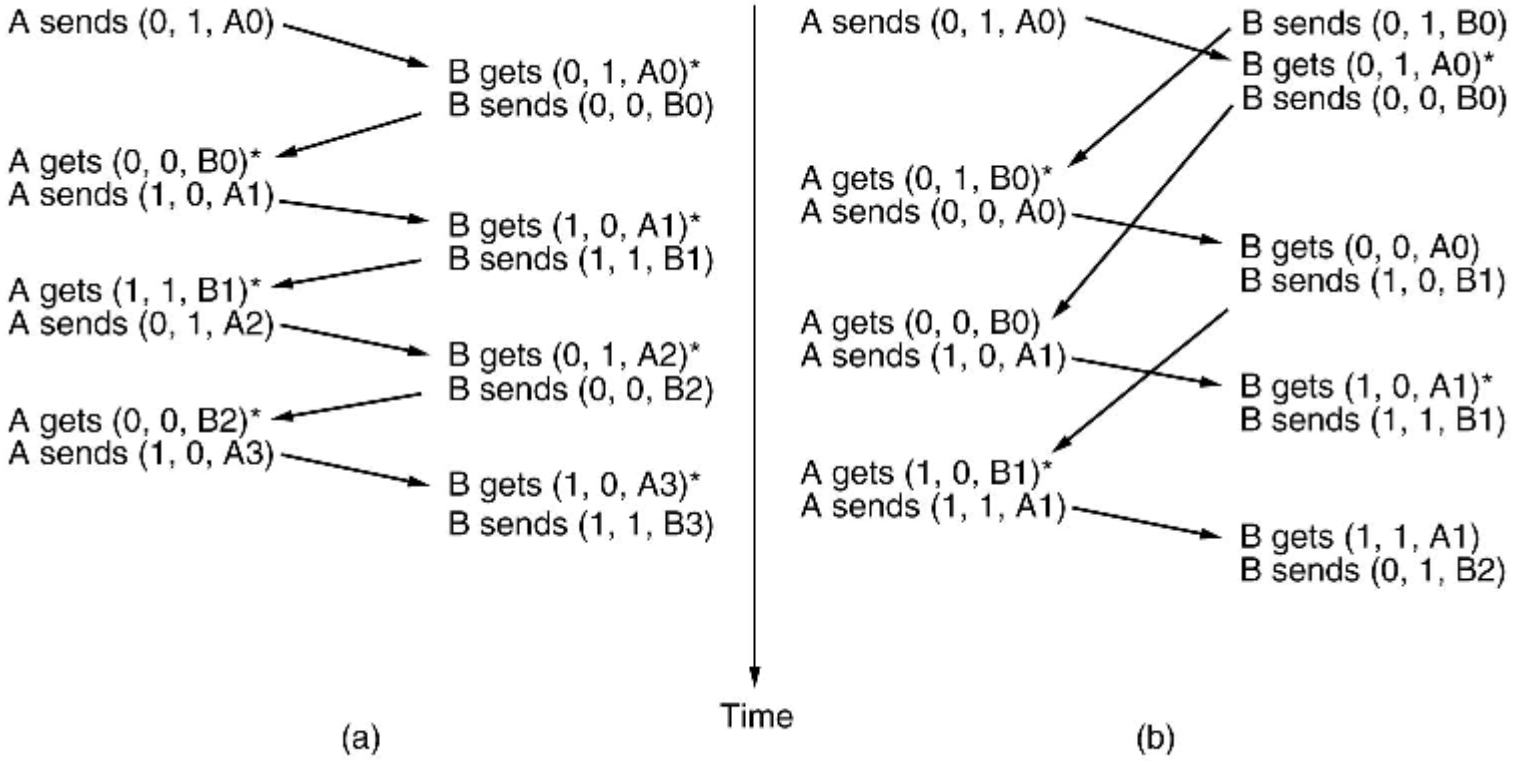
Janela Deslizante de um bit

- Janela deslizante de tamanho 1, com número de seqüência de 3 bits;



Janela Deslizante de um bit

- Vejamos um situação patológica:
  - Quadro formado por seq e ack:



Janela Deslizante que utiliza go back n

- Até agora estamos supondo implicitamente que o tempo de transmissão necessário para a chegada de um quadro até o receptor somado ao tempo de transmissão para o retorno da confirmação é insignificante;

- Além disso, definimos que o transmissor deve esperar por uma confirmação antes de enviar outro quadro;
  - Se essa restrição não for rigorosa, poderemos obter uma eficiência muito melhor.

Janela Deslizante que utiliza go back n

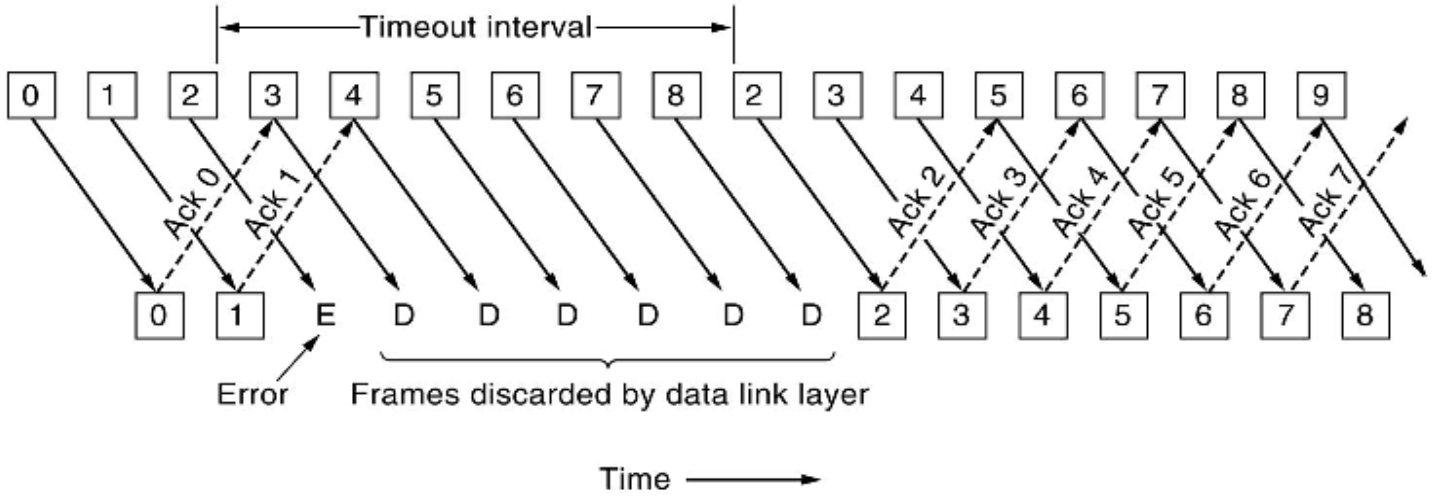
- A solução está em permitir que o transmissor envie até w quadros antes do bloqueio;

- Mas qual seria o tamanho da janela de recepção?
  - Grande quando o produto da largura de banda pelo retardo de ida e volta é grande;
  - Grande quando a largura de banda for alta e o retardo moderado;

- Essa técnica de maximizar a eficiência do canal é conhecida como pipelining de quadros.

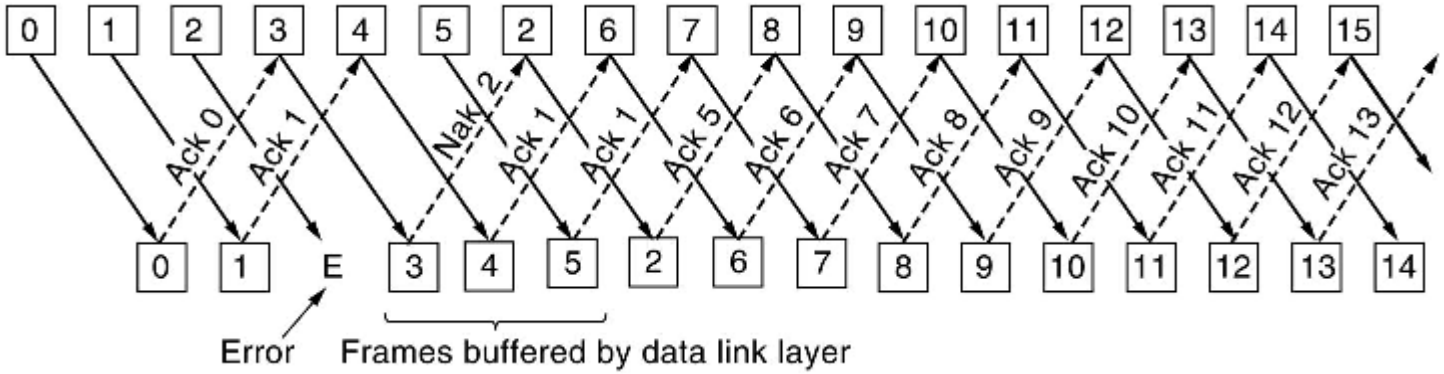
Janela Deslizante que utiliza go back n

- Há duas estratégias básicas para lidar com erros na transmissão do pipelining. A primeira é denominada go back n:
  - O receptor simplesmente descarta todos os quadros subseqüentes e não envia qualquer confirmação desses quadros descartados.



Janela Deslizante que utiliza go back n

- Outra estratégia geral para transmissão de erros quando é feito o pipelining de quadros denomina-se retransmissão seletiva:
  - Um quadro incorreto recebido é descartado, mas os quadros sem defeitos recebidos depois dele são inseridos no buffer.



  - Os NAKs estimulam a retransmissão antes de expirar o timer correspondente e, desse modo, melhoram o desempenho.

Point-to-Point Protocol – PPP

- PPP é um dos protocolos da camada de enlace de dados mais usados atualmente. Simplesmente pelo fato de possibilitar a conexão de computadores domésticos à Internet;

- Como o nome indica, o PPP é um protocolo de camada de enlace que opera sobre um enlace ponto-a-ponto:
  - Ele está definido na Request for Comments – RFC 1661 e mais elaborado em várias outras RFCs (por exemplo, as RFCs 1662, 1663 e 2153)

Point-to-Point Protocol – PPP

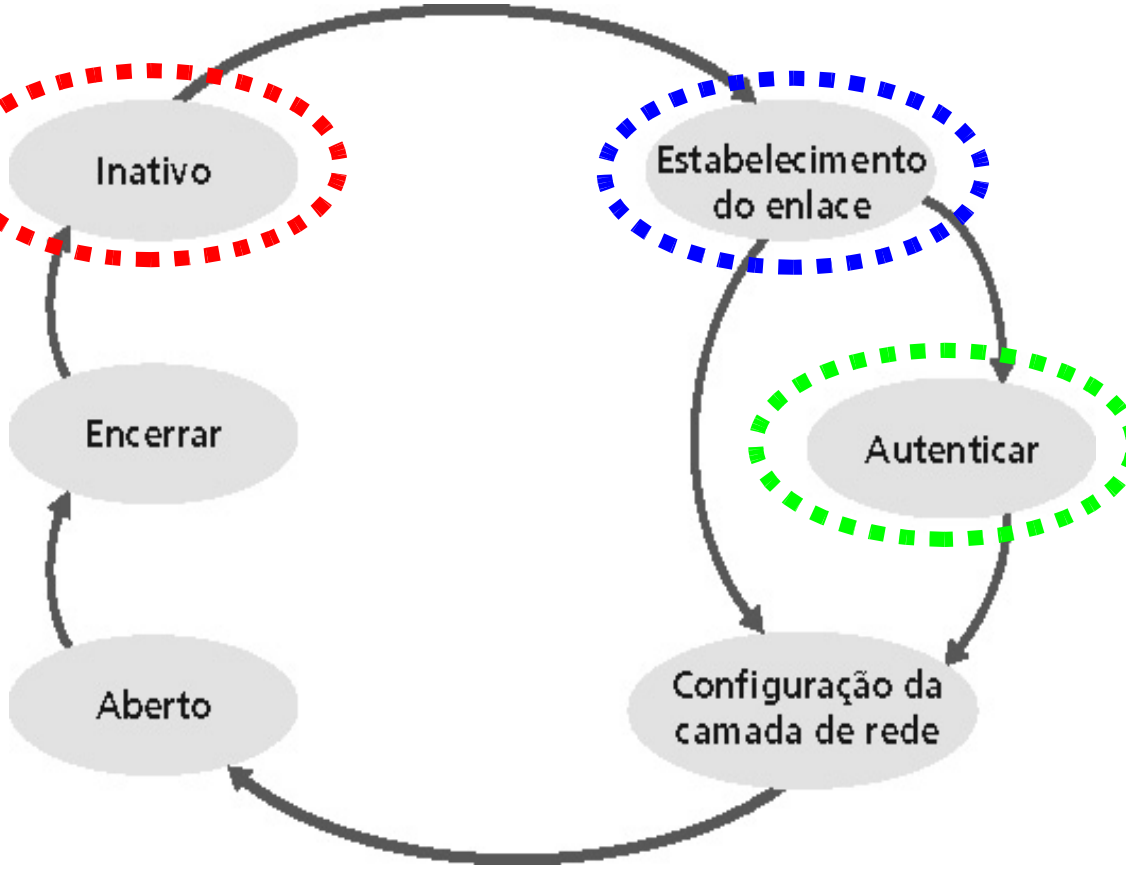
- O PPP dispõe de três recursos:
  - Um método de enquadramento que delineia de forma não-ambígua o fim de um quadro e o início do quadro seguinte. O formato do quadro também lida com a detecção de erros.
  - Um protocolo de controle de enlace (Link Control Protocol – LCP)usado para ativar linhas, testá-las, negociar opções e desativá-las novamente quando não foram mais necessárias.
  - Uma maneira de negociar as opções da camada de rede de modo independente do protocolo da camada de rede a ser utilizado. O método escolhido deve ter um NCP (Network Control Protocol – protocolo de controle de rede) diferente de cada camada de rede aceita;

Point-to-Point Protocol – PPP

- Antes que quaisquer dados sejam trocados sobre um enlace PPP, os dois pares devem primeiramente rodar uma quantidade considerável de trabalho para configurar o enlace, veja:

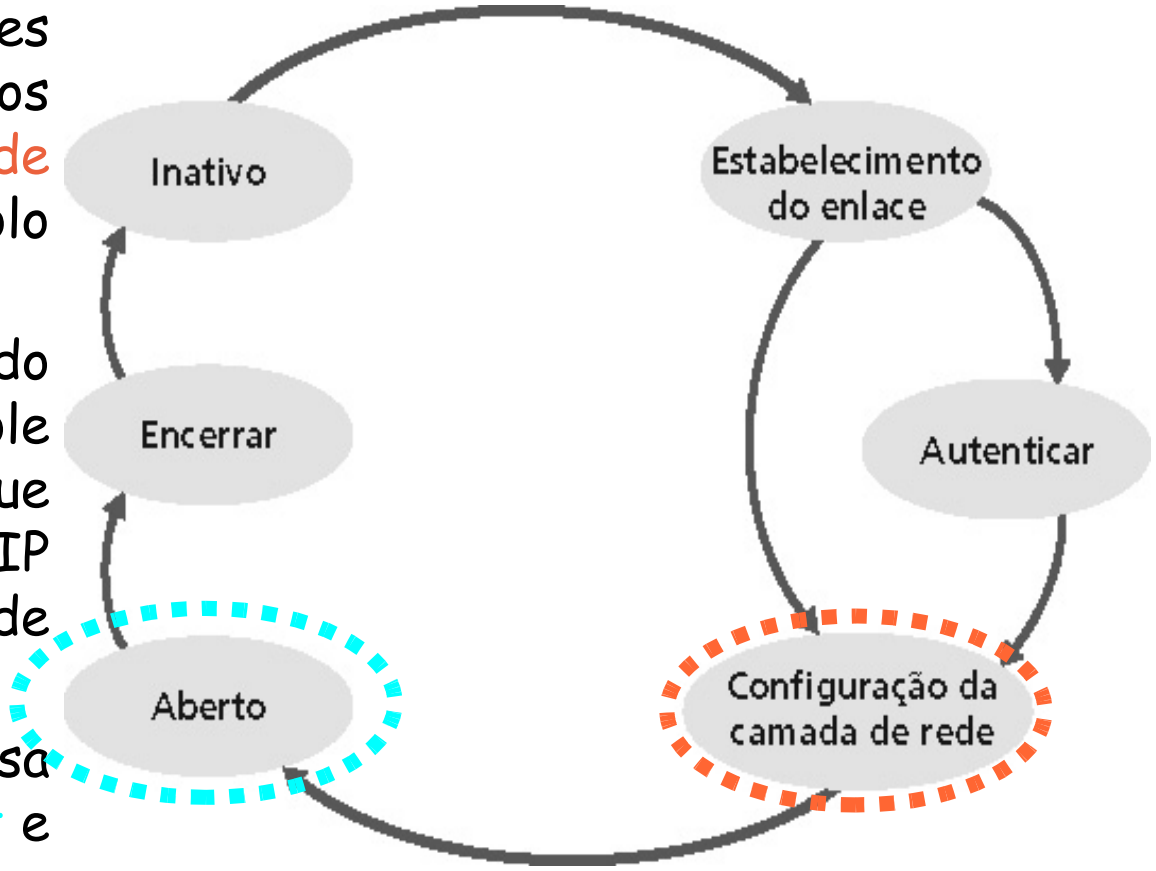
Point-to-Point Protocol – PPP

- O enlace sempre começa e termina em estado inativo;
- Quando ocorre um evento (detecção de uma portadora) o PPP entra no estado de "estabelecimento do enlace".
- Nesse caso, uma extremidade envia suas opções (tamanho do quadro, autenticação se houve, controle de fluxo) de configuração do enlace (dados LCP dentro do PPP)



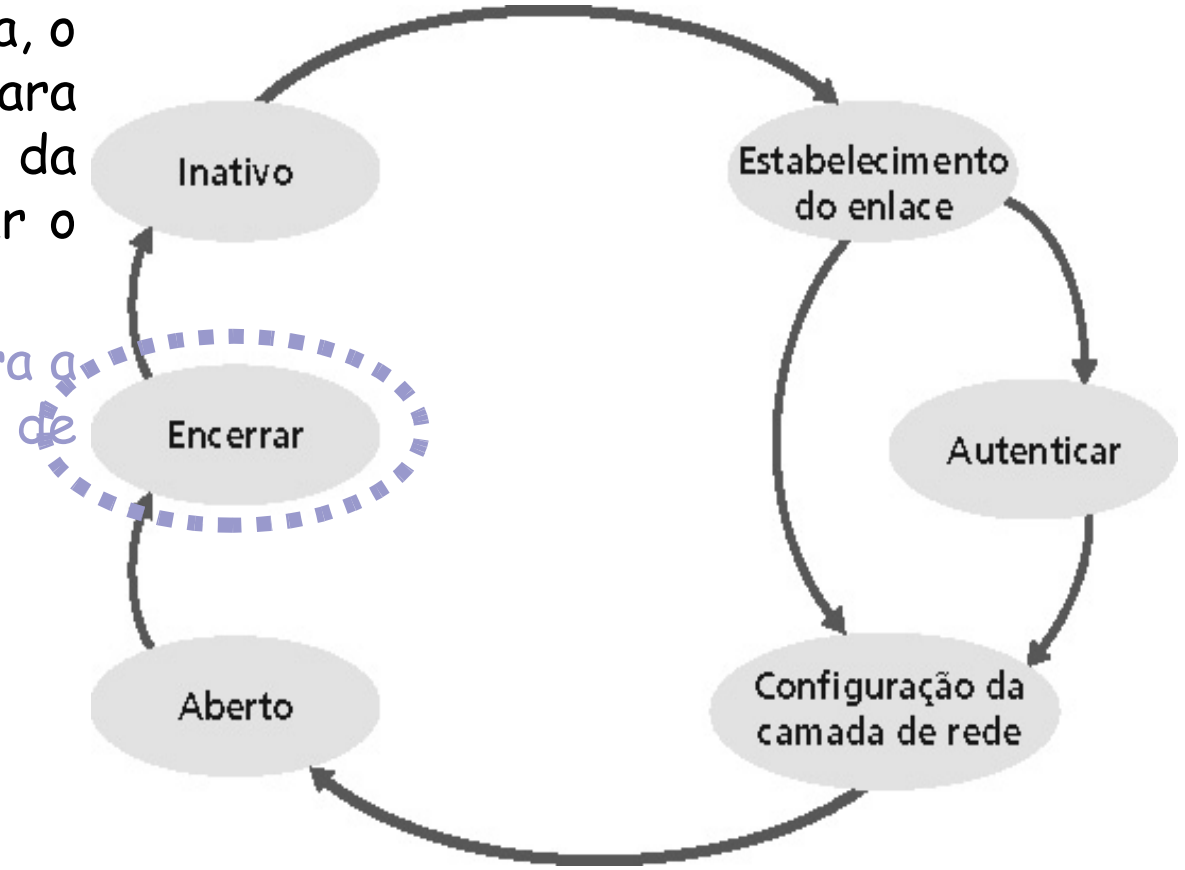
Point-to-Point Protocol - PPP

- Em seguida, os enlaces trocam pacotes específicos de controle de camada de rede para cada protocolo de rede;
- No caso do IP será usado um protocolo de controle (IPCP RFC 1332) que configure os módulos do IP em cada extremidade (dados IP dentro do PPP);
- Nesse momento, o PC passa a ser um host de Internet e pode enviar e receber pacotes IP;



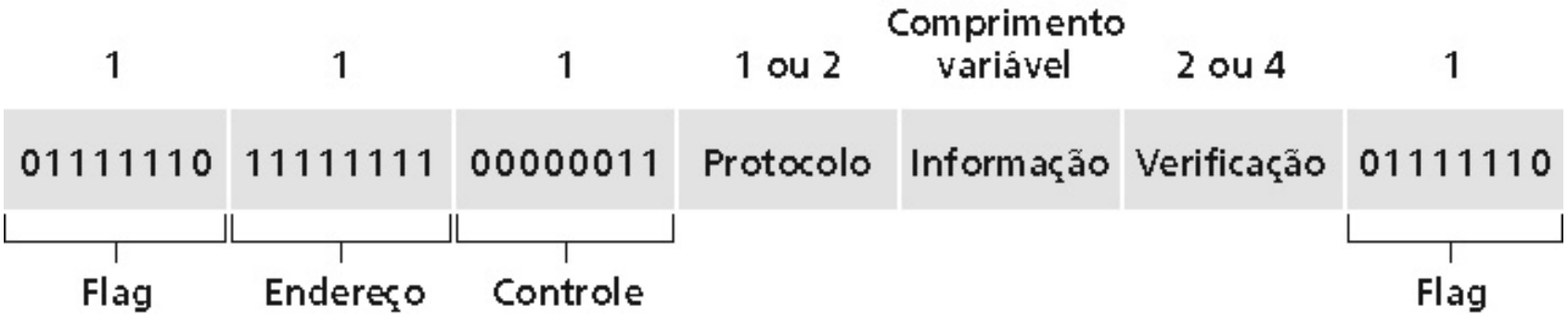
Point-to-Point Protocol – PPP

- Quando o usuário termina, o NCP é utilizado para desativar a conexão da camada de rede e liberar o endereço IP;
- Em seguida, o LCP encerra a conexão da camada de enlace de dados;



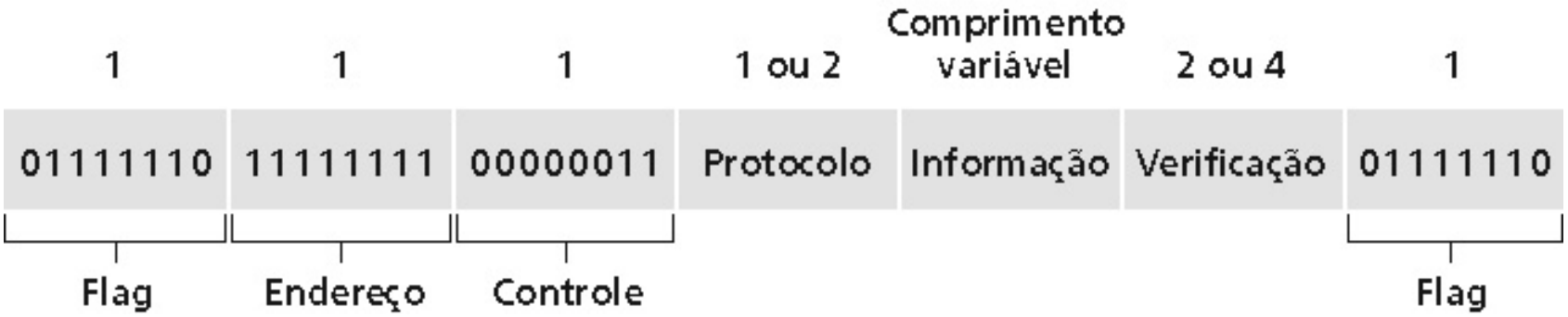
Point-to-Point Protocol - PPP

- Enquadramento:
  - info: dados da camada superior sendo transportados
  - CRC: verificação de redundância cíclica para detecção de erros



Point-to-Point Protocol - PPP

- Enquadramento:
  - info: dados da camada superior sendo transportados
  - CRC: verificação de redundância cíclica para detecção de erros



Point-to-Point Protocol - PPP

- Resumindo:
  - Enquadramento de pacote: encapsulamento do datagrama da camada de rede no quadro da camada de enlace;
  - Transporta dados da camada de rede de qualquer protocolo de rede (não apenas o IP) *ao mesmo tempo*
  - Capacidade de separar os protocolos na recepção
  - Transparência de bits: deve transportar qualquer padrão de bit no campo de dados
  - Detecção de erros (mas não correção)

Point-to-Point Protocol – PPP

- Gerenciamento da conexão: detecta e informa falhas do enlace para a camada de rede;
- Negociação de endereço da camada de rede: os pontos terminais do enlace podem aprender e configurar o endereço de rede dos outros;
- Não há correção nem recuperação de erros;
- Não há controle de fluxo;
- Aceita entregas fora de ordem
- Não há necessidade de suportar enlaces multiponto (ex., polling)

Bibliografia

- TANENBAUM, A.S.: *Redes de Computadores,* Elsevier, Rio de Janeiro: 2003.

- KUROSE, J.F e ROSS, K.W.: *Computer Networking hird edition a top-down approach featuring the Internet,* 3 ed, São Paulo: Pearson Addison Wesley, 2006.