Nível de Enlace

Profa. Débora Christina Muchaluat Saade

Laboratório MídiaCom - UFF

debora@midiacom.uff.br

http://www.midiacom.uff.br/debora

Nível de Enlace

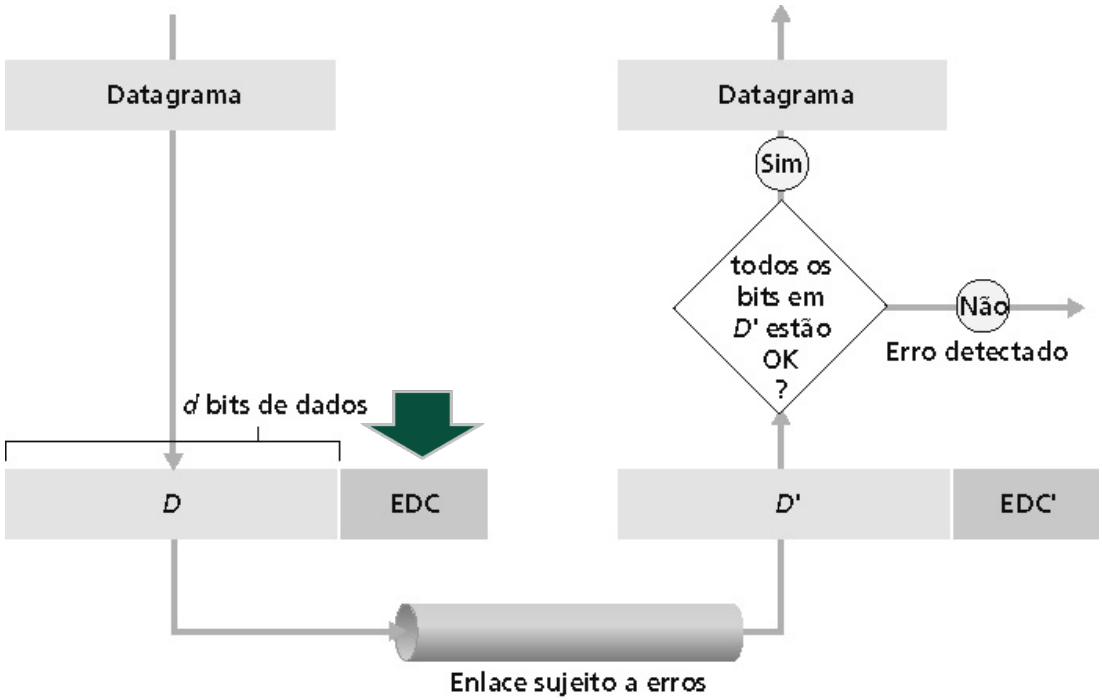
- **Funcionalidades principais:**
  - ***Oferecer serviços de transmissão de quadros ao nível de rede***
  - ***Delimitação de quadros***
  - ***Controle de erros:***
    - Detecção de erros - obrigatório
    - Correção de erros - opcional
  - ***Controle de fluxo – opcional***

Detecção de Erros

- **Detecção de erro - obrigatória**
  - ***Verificação de paridade***
  - ***checksum***
  - ***CRC – Cyclic Redundancy Check***

Detecção de Erros

Ponto-chave: enviar **informações redundantes** para detectar e corrigir erros



EDC= bits de Detecção e Correção de Erros (redundância)

D = dados protegidos por verificação de erros, podem incluir alguns campos do cabeçalho

- Quanto maior o campo EDC melhor será a capacidade de detecção e correção de erros

Detecção e Correção de Erros

- ✓ **Códigos de detecção de erros**
  - ***Menos redundância para deduzir um erro***
  - ***Bons para enlaces confiáveis***
    - Ex.: enlaces de fibra
    - Retransmissão “mais confiável”

Detecção e Correção de Erros

- ✓ **Códigos de correção de erros**
  - ***Mais redundância para deduzir quais dados foram transmitidos***
  - ***Bons para enlaces pouco confiáveis***
    - Ex: enlaces sem-fio
    - Retransmissão pode conter erros
  - ***Em geral, chamada de Correção Antecipada de Erros (Forward Error Correction – FEC)***
    - Vantagem: antecipar a decisão

Verificação de Paridade

- ✓ **Código simples de detecção de erros**
- ✓ **Bit de paridade acrescentado aos dados**
  - *Escolhido de forma que o número de bits 1 da palavra de código seja par (paridade par) ou ímpar (paridade ímpar)*
- ✓ **Receptor conta quantos bits 1 a palavra possui**
  - *Se é usada a paridade par e contou um número ímpar de 1s → Ocorreu um número ímpar de erros*
  - *Número par de erros → não são detectados*

Verificação de Paridade

- ✓ **Exemplo**
  - *1011010 enviado com paridade par → 10110100*
  - *1011010 enviado com paridade ímpar → 10110101*

- ✓ **Problema: erros ocorrem geralmente em rajada**
  - *Paridade com um bit não é suficiente*
  - *Solução → aumenta-se o número de bits de paridade*

Verificação de Paridade

- ✓ **Paridade bidimensional**
  - ***Paridade de linha***
  - ***Paridade de coluna***
  - ***Paridade dos bits de paridade***
- ✓ **Pode detectar e corrigir erros isolados**
- ✓ **Pode detectar erros duplos**

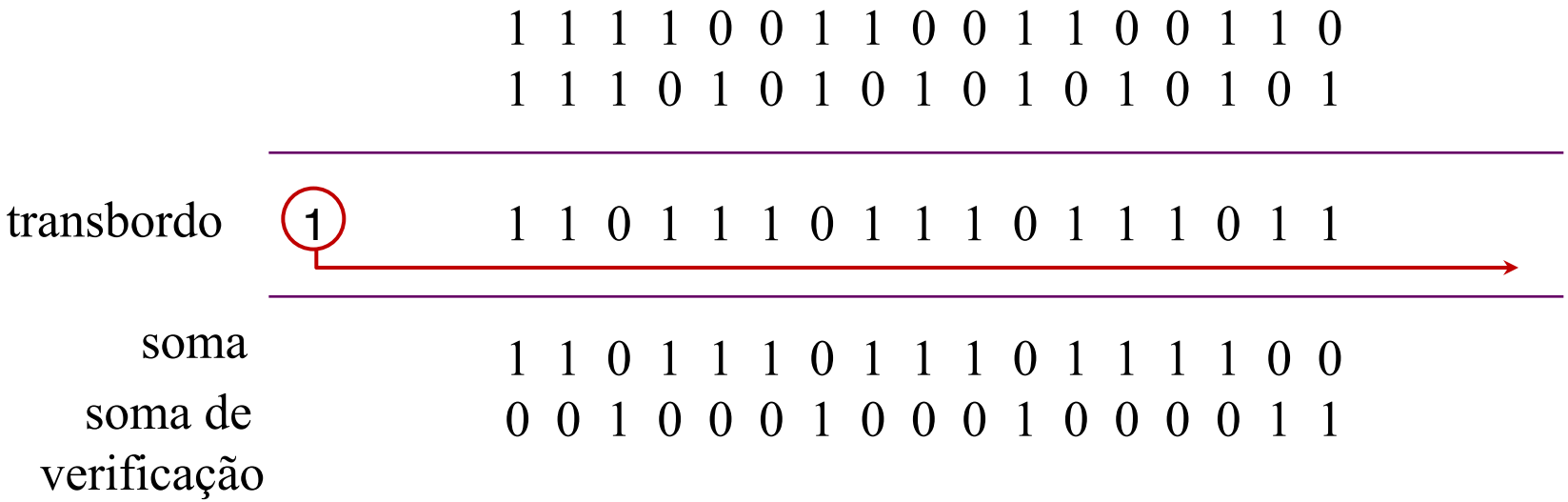
Exemplo de paridade bidimensional (fonte: Kurose)

Detecção e Correção de Erros: Soma de Verificação (*checksum*)

- ✓ **Método simples**
  - *Normalmente implementado em software*
- ✓ **Bits de dados tratados como uma sequência de números inteiros de *k* bits**
- ✓ **Soma-se esses números inteiros (em complemento a 1) e usa-se o total como bits de detecção de erros**

Detecção e Correção de Erros: Soma de Verificação (*checksum*)

- ✓ **Observação**
  - ***Ao adicionar números, o transbordo (vai um) do bit mais significativo deve ser adicionado ao resultado***
- ✓ **Exemplo: adição de dois inteiros de 16-bits**



Detecção e Correção de Erros: Soma de Verificação (*checksum*)

- ✓ **Método simples**
  - *Normalmente implementado em software*
- ✓ **Bits de dados tratados como uma sequência de números inteiros de *k* bits**
- ✓ **Soma-se esses números inteiros (em complemento a 1) e usa-se o total como bits de detecção de erros**
- ✓ **Receptor recalcula o *checksum* dos dados recebidos**
  - *Se o resultado contém algum bit 0 → erro*
- ✓ **Usado no TCP, no UDP e no IP**

Detecção e Correção de Erros: CRC

- ✓ **Código de redundância cíclica (*Cyclic Redundancy Check*) ou código polinomial**
- ✓ **Mais complexo**
  - ***Geralmente implementado em hardware***
- ✓ **Trata uma sequência de bits como representações de polinômios com coeficientes 0 e 1**
- ✓ **Quadro de *k* bits → *k* termos, de x^{k-1} até x^0**
  - ***Polinômio de grau k-1***
  - ***Ex.: 11001 → $1x^4 + 1x^3 + 0x^2 + 0x^1 + 1x^0 = x^4 + x^3 + 1$***

Detecção e correção de erros: CRC

- ✓ **Idéia com dividendo A = 7 e divisor B = 3**
- ✓ **Transmissor**

1. *Dividir A por B*

	7	3
resto	1	2

2. *Subtrair o resto de A → 7 – 1 =6*
3. *O resultado é a mensagem C a ser transmitida → 6*

Detecção e correção de erros: CRC

- ✓ **Idéia com dividendo A = 7 e divisor B = 3**
- ✓ **Receptor**
  1. ***Recebe C' (canal pode ter erros)***
  2. ***Divide C' por B***
  3. ***Caso resto=0 → não há erros***

	6	3
resto	0	2

Detecção e Correção de Erros: CRC

- ✓ **Aritmética polinomial feita em módulo 2, sem transportes para adição nem empréstimos para subtração**
  - ***Adição e subtração são idênticas à operação ou-exclusivo***

```
  10011011      11110000
+ 11001010    - 10100110
----------    ----------
  01010001      01010110
```

- ✓ **Divisão similar à binária**
  - ***Um divisor "cabe em" um dividendo se o dividendo tem a mesma quantidade de bits do divisor***

Detecção e correção de erros: CRC

- ✓ **Transmissor e receptor devem concordar em relação ao uso de um polinômio gerador G(x)**
- ✓ **Quadro de *d* bits corresponde a D(x)**
- ✓ **Grau D(x) > Grau G(x)**

Detecção e correção de erros: CRC

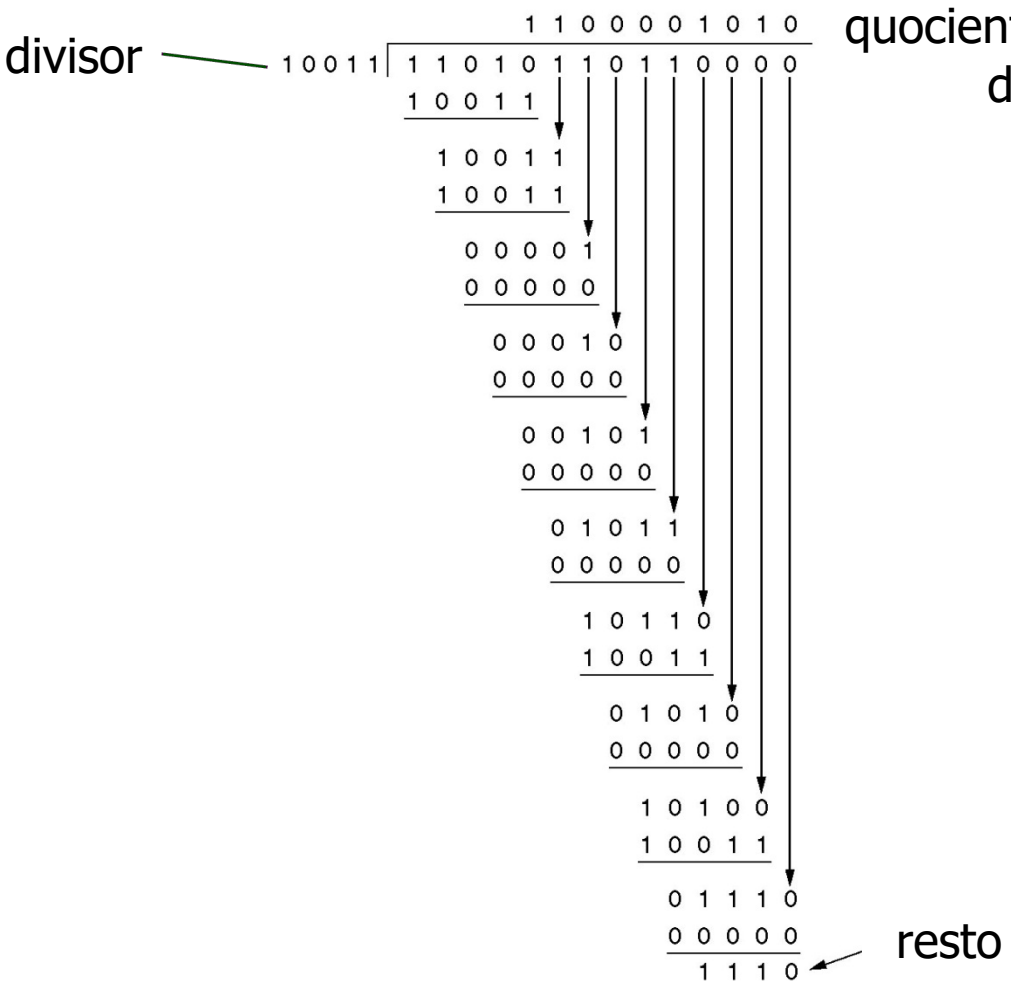
- ✓ **CRC acrescentado ao final do quadro de forma que o quadro verificado seja divisível por G(x)**
  - ***Sequência de verificação de quadro (Frame Check Sequence – FCS)***
- ✓ **Ao receber o quadro verificado, o receptor tentará dividi-lo por G(x)**
  - ***Se o resto é diferente de zero → erro***

Detecção e correção de erros: CRC

- ✓ **Algoritmo**
  - ***Seja r o grau de G(x)***
  - ***Acrescentar r bits à extremidade de mais baixa ordem de D(x)***
    - Polinômio $x^rD(x)$
  - ***Dividir a sequência de bits correspondente a $x^rD(x)$ pela sequência correspondente por G(x)***
  - ***Subtrair o resto da sequência correspondente a $x^rD(x)$***
  - ***Resultado é o quadro verificado que deverá ser transmitido***
    - Polinômio T(x)

Exemplo de cálculo de CRC (fonte: Tanenbaum)

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 0 0



$d = 10$
$r = 4$

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

Detecção e correção de erros: CRC

- ✓ **Usado em diversos padrões de redes locais e metropolitanas**
- ✓ **Exemplo de G(x) do IEEE 802**
  - $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$

Correção de Erros

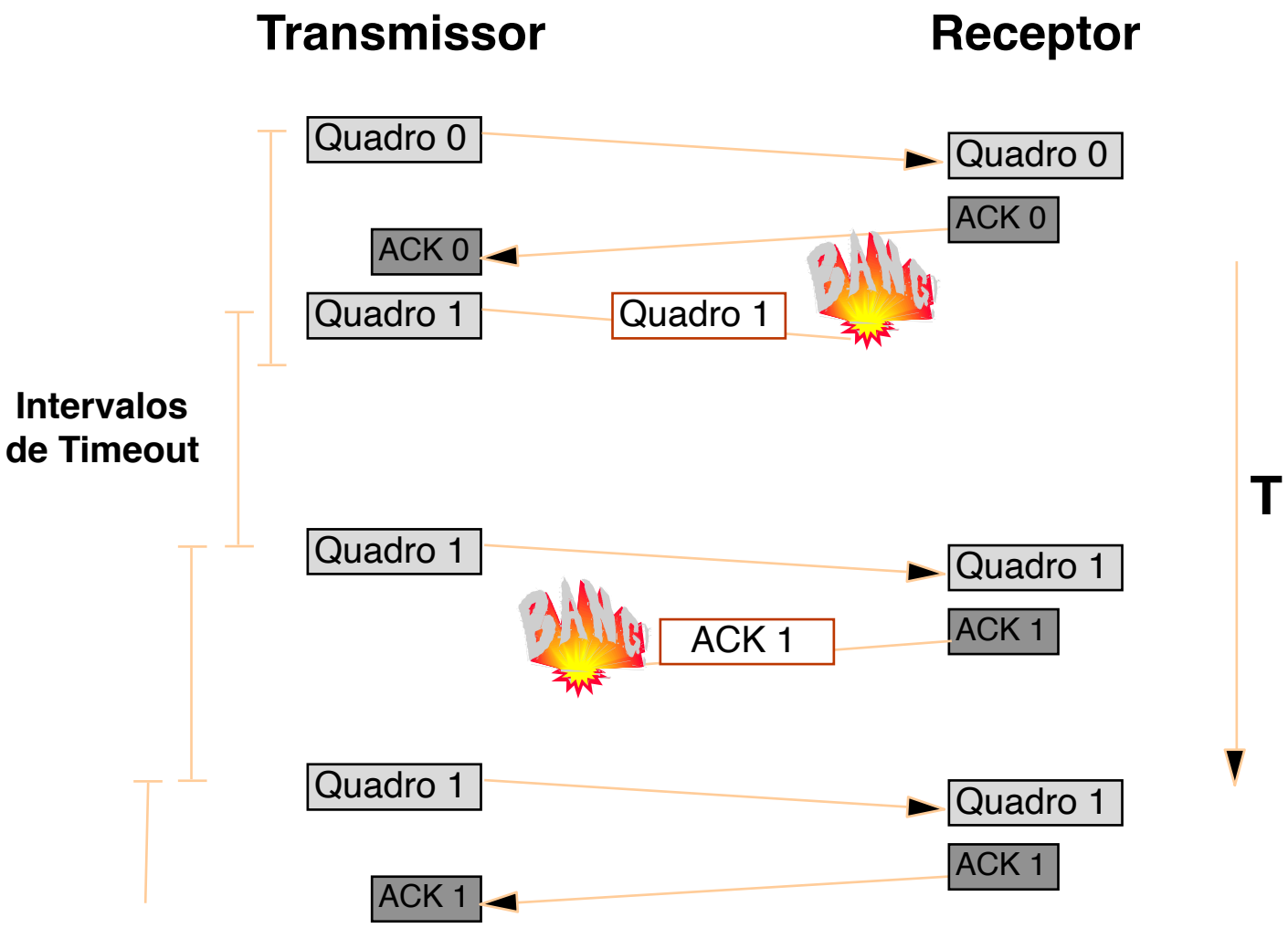
- **Correção de erros - opcional**
  - ***Código corretor de erro***
    - Ex.: Hamming Code
  - ***Protocolo de Controle de Erros***
    - Retransmissão do quadro com erro

Protocolos de Controle de Erros

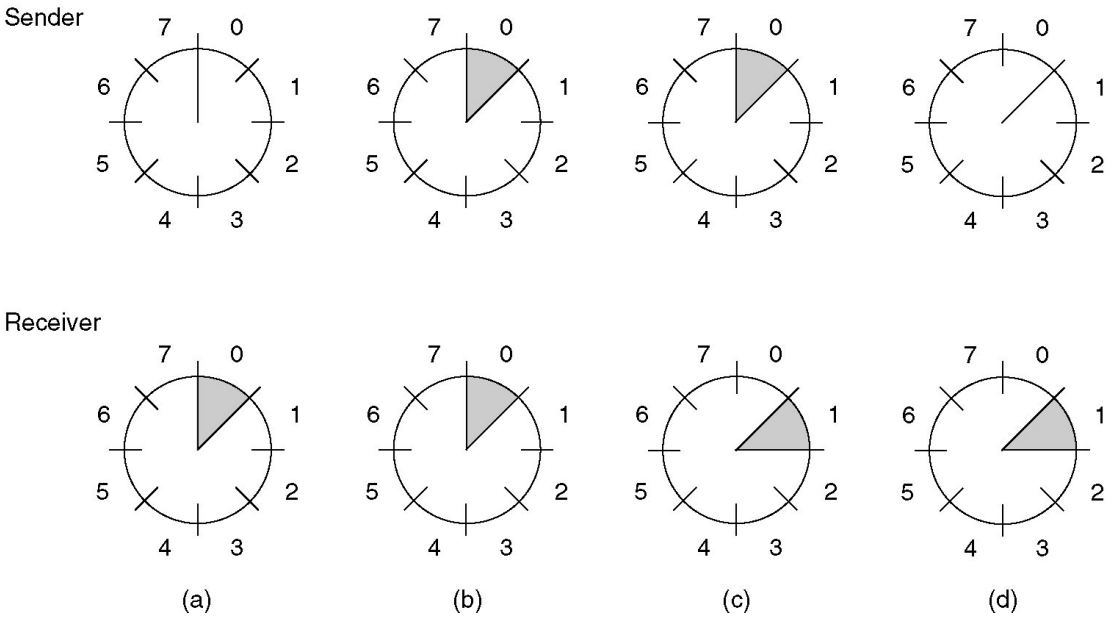
- ✓ **Todo quadro recebido deve ser reconhecido (ACK – acknowledgement)**
  - ***Se ACK não chegar, retransmite depois do timeout***
- ✓ **Janela de transmissão e janela de recepção**

- ✓ **Protocolos baseados em Janela Deslizante (*Sliding Windows*)**
  - ***Stop-and-Wait (One-Bit Sliding Window Protocol)***
  - ***Go Back N Sliding Window***
  - ***Selective Repeat Sliding Window***

Protocolo *Stop-and-Wait*



Protocolo Sliding Window



Sliding window de tamanho 1, com número de sequência de 3 bits:

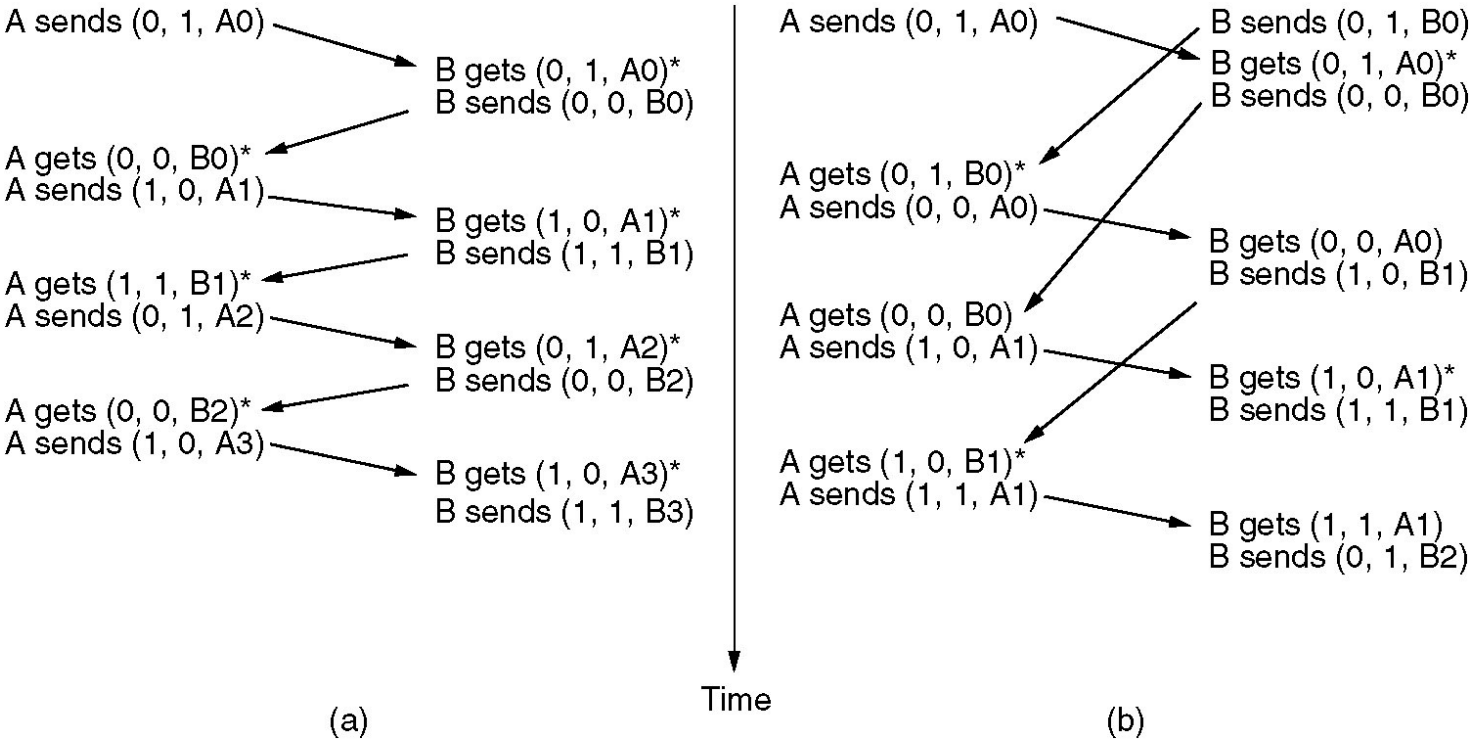
(a) Situação inicial

(b) Depois de transmitir o primeiro quadro

(c) Depois de receber o primeiro quadro

(d) Depois de receber o primeiro ACK

One-Bit Sliding Window Protocol



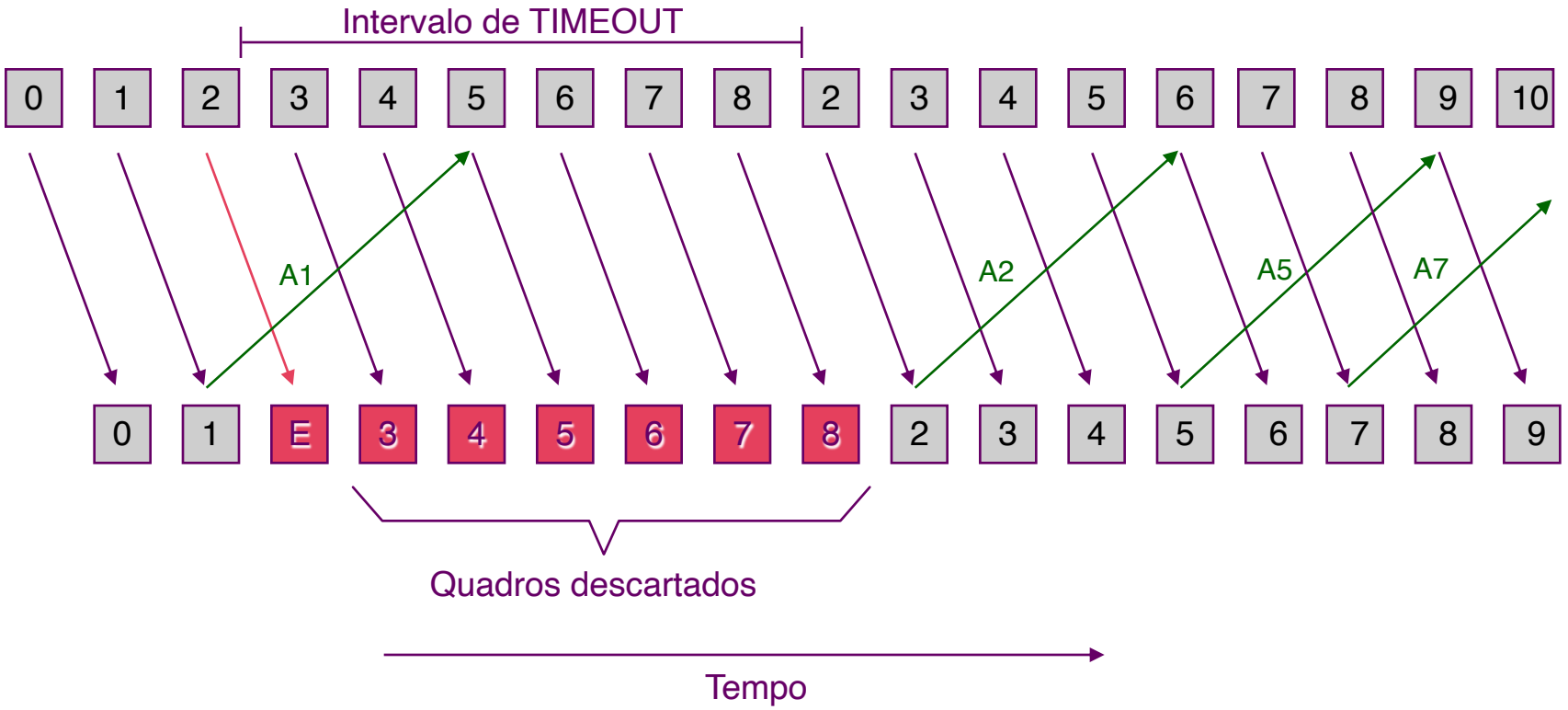
Dois cenários (a) caso normal (b) caso anormal

Notação (seq, ack, packet number).

Asterisco indica que nível de rede recebeu o pacote

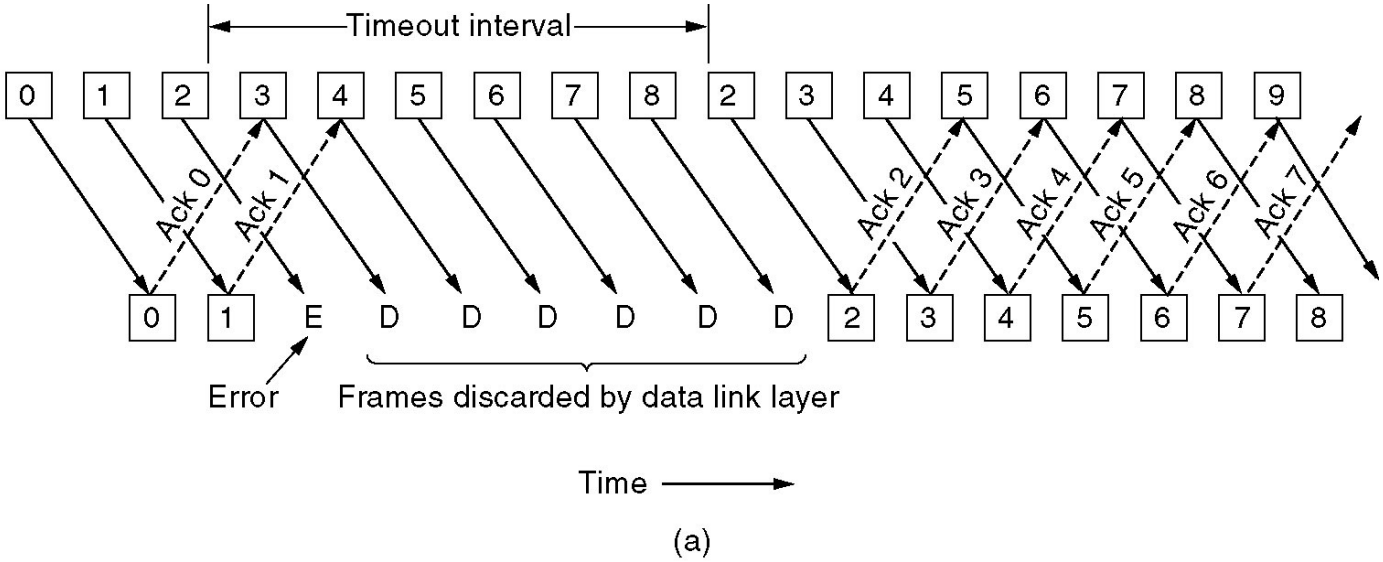
Controle de Erro: Protocolo *Go Back N*

(a) Receptor transmite ACK cumulativo



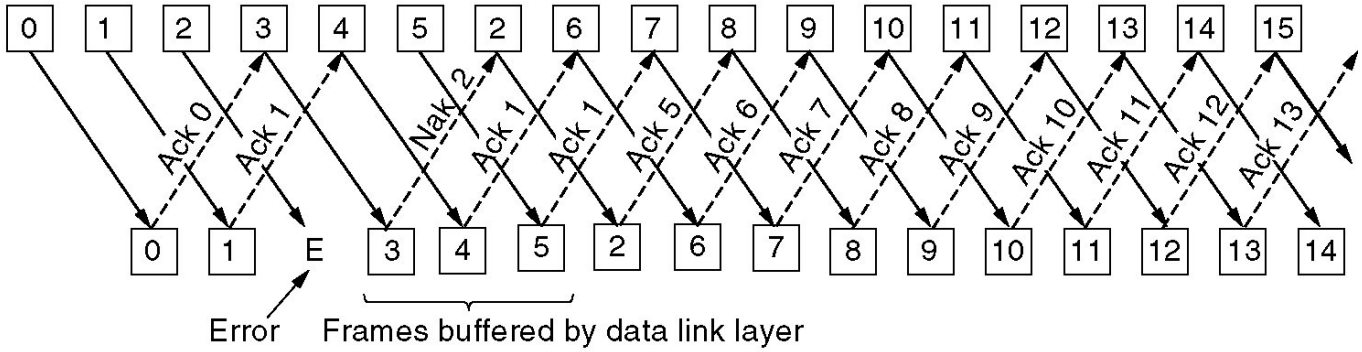
Protocolo Go Back N

(a) **Tamanho da janela de recepção é 1**



(a)

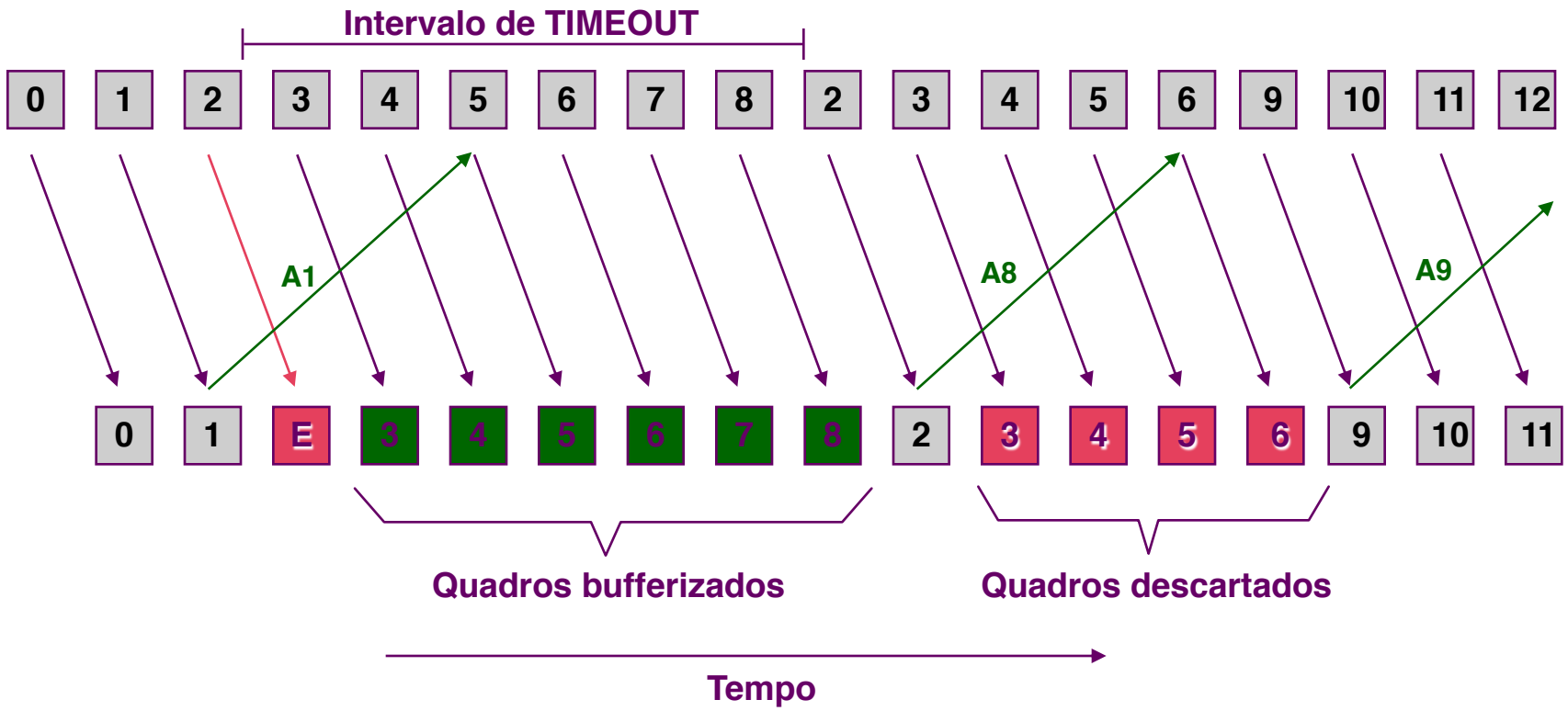
(b) **Tamanho da janela de recepção é grande e usa NACK**



(b)

Protocolo *Selective Repeat*

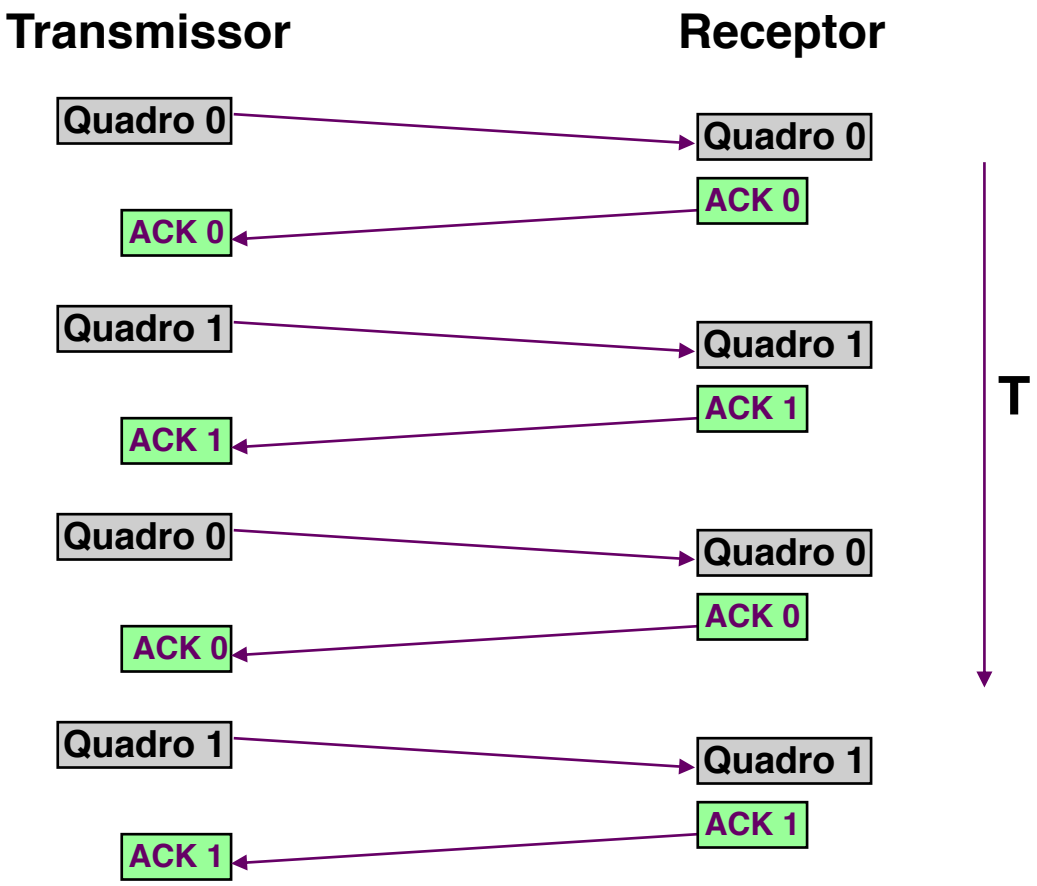
(a) Receptor transmite ACK cumulativo



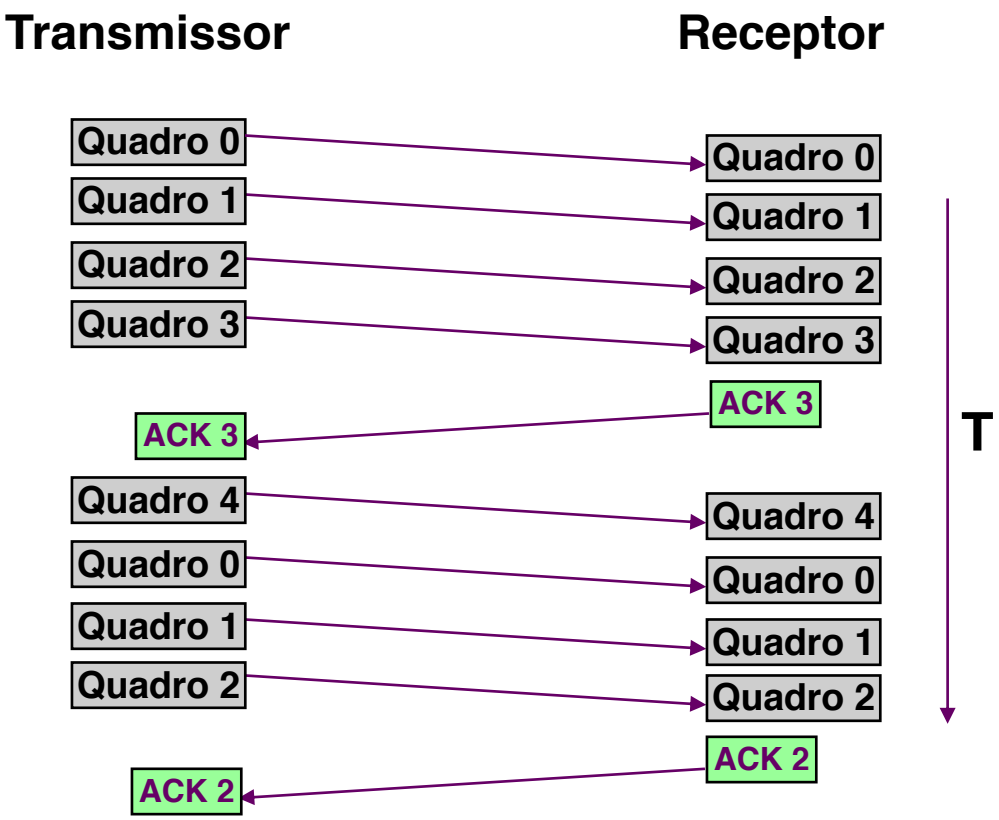
Controle de Fluxo

- ✓ **Regula o fluxo de quadros entre transmissor e receptor**
- ✓ **Resolve o problema de diferença entre velocidade de transmissão e recepção**
- ✓ **Não permite que uma estação transmissora mais rápida sobrecarregue uma estação receptora**
- ✓ **Técnicas:**
  - ***Stop-and-Wait***
  - ***Sliding Window***

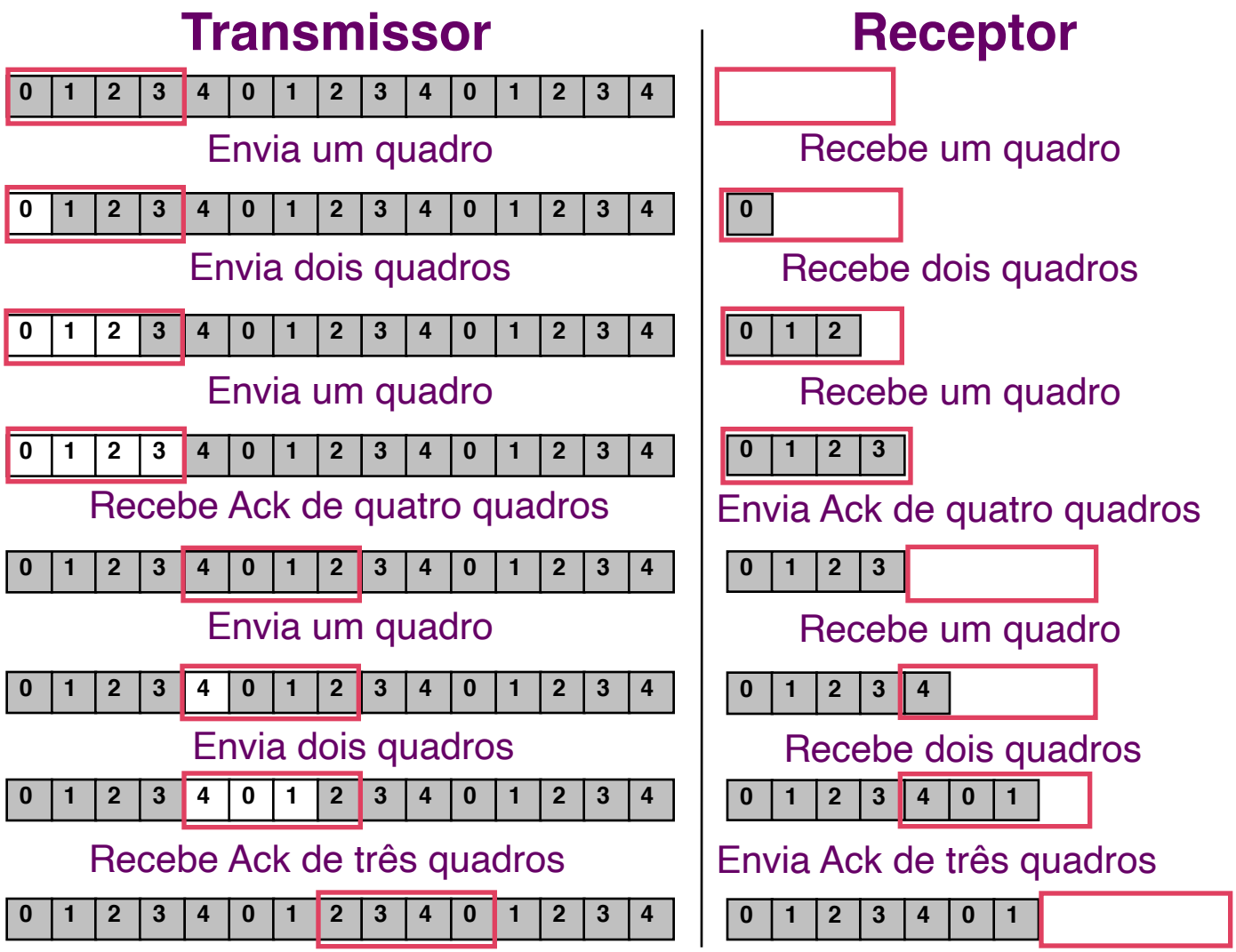
Protocolo *Stop-and-Wait*



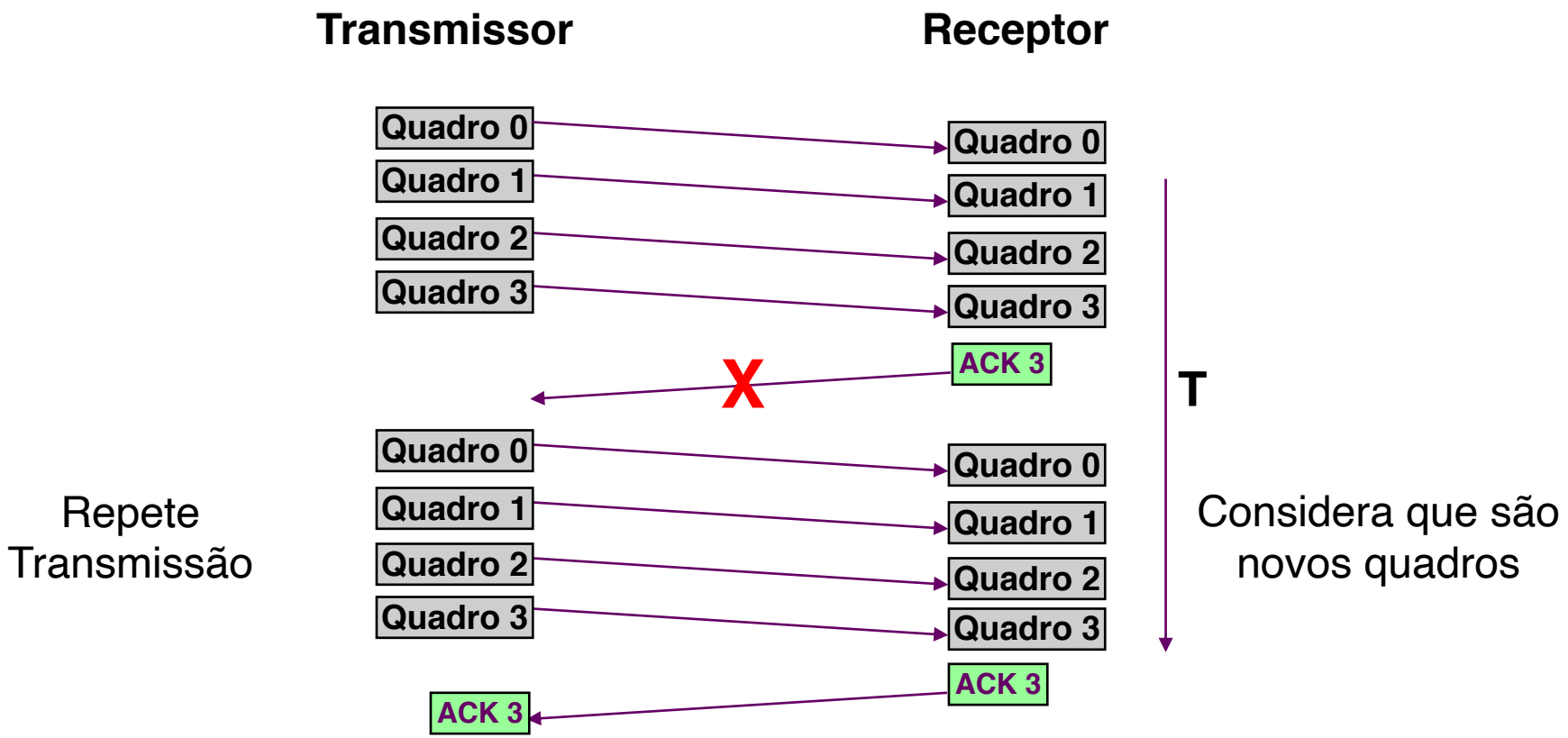
Protocolo *Sliding Window*



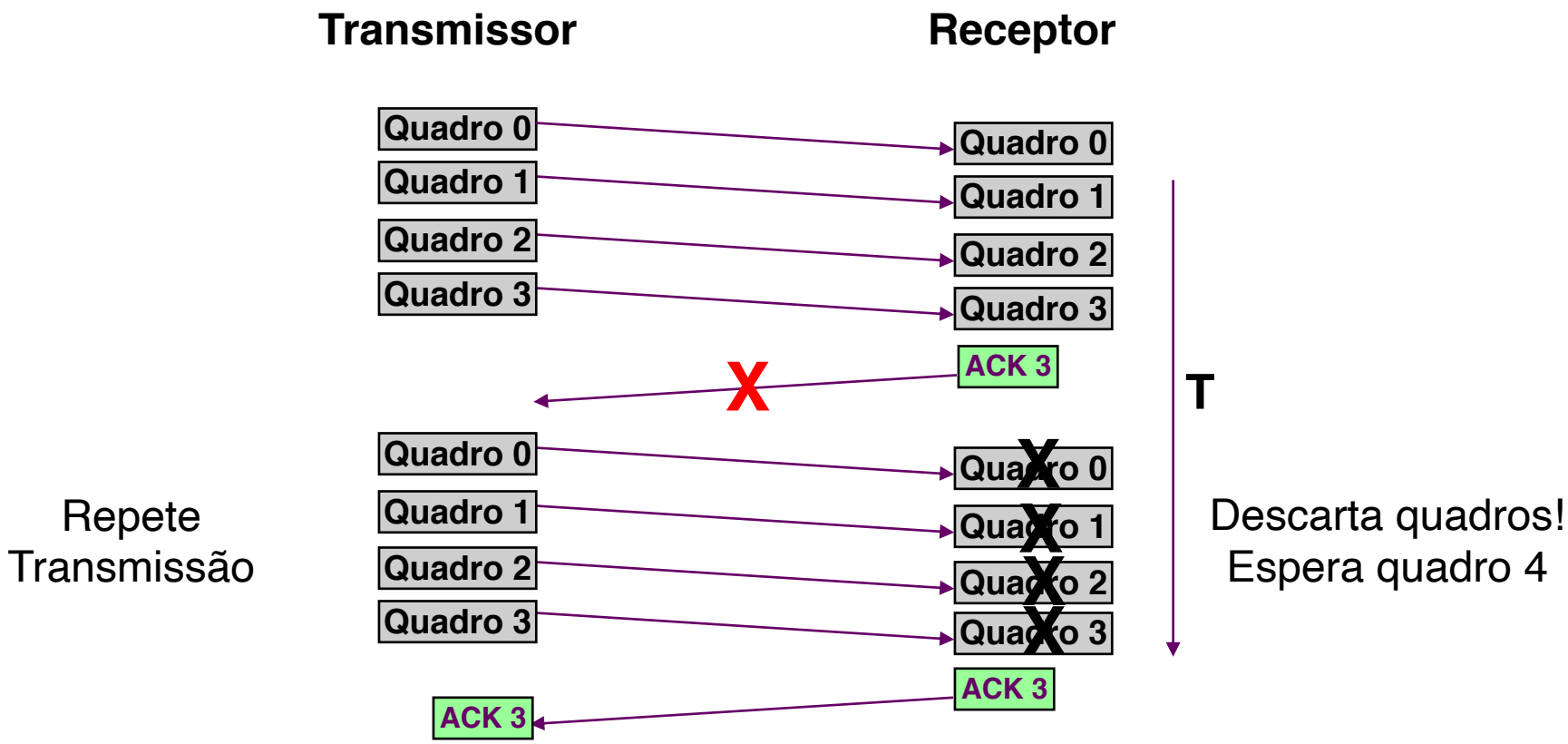
Protocolo *Sliding Window*



Protocolo *Sliding Window – Go Back N*

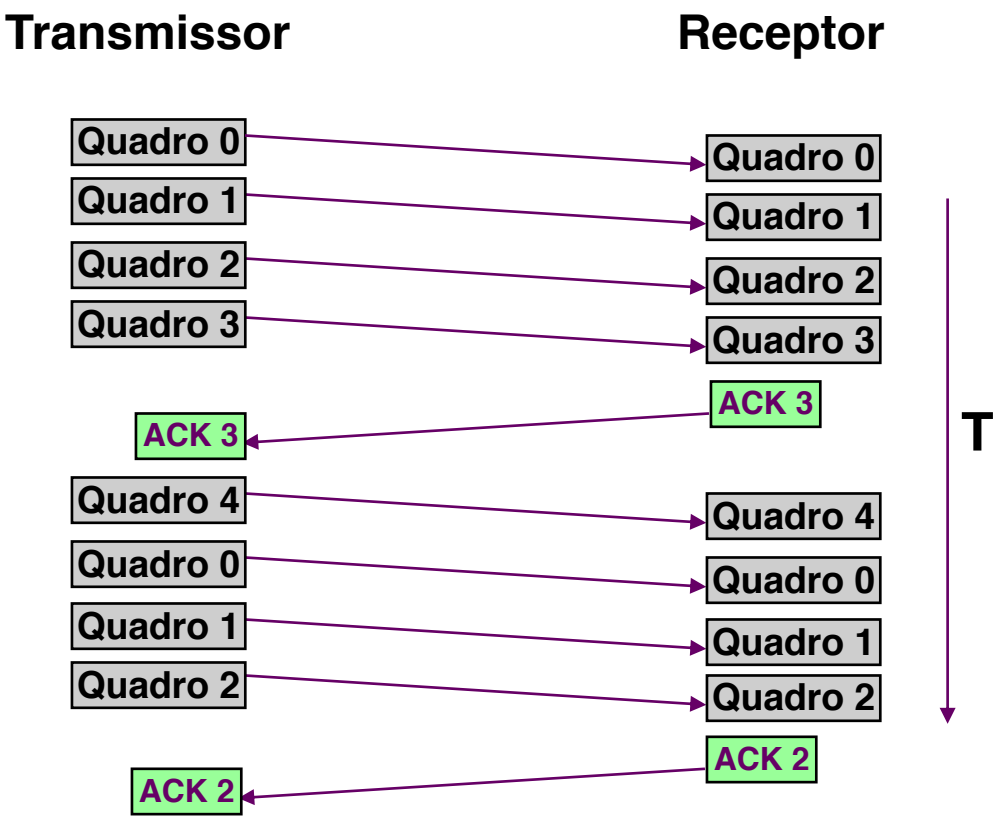


Protocolo *Sliding Window – Go Back N*



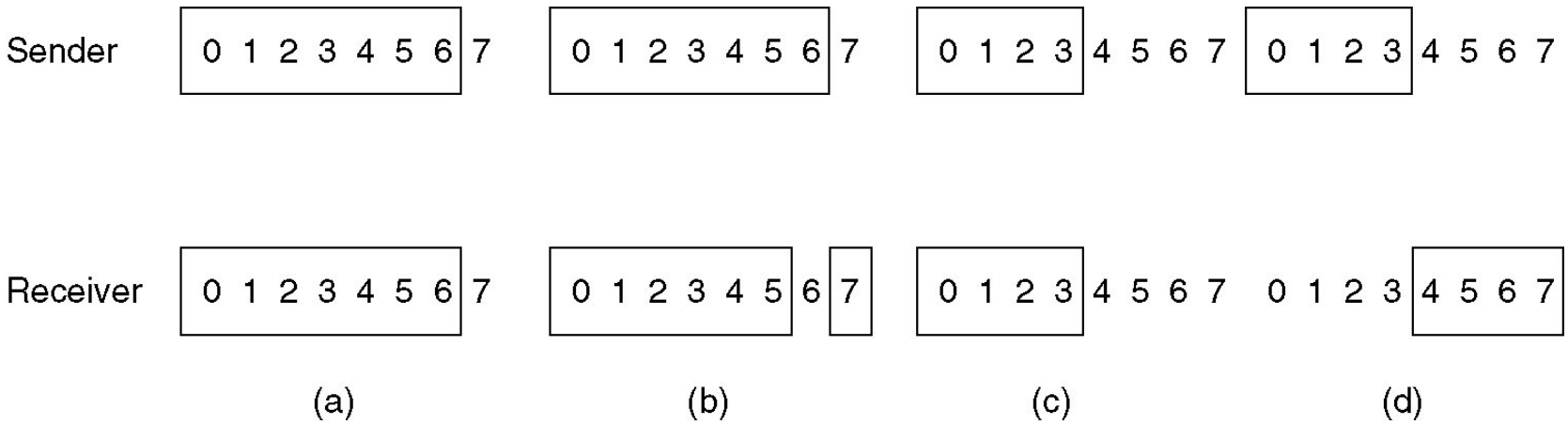
Go Back N: Se janela tem tamanho n, precisa de (n+1) números de sequência

Protocolo *Sliding Window – Go Back N*



Se janela = 4, precisa de 5 números de sequência (0 a 4)

A Sliding Window Protocol Using Selective Repeat



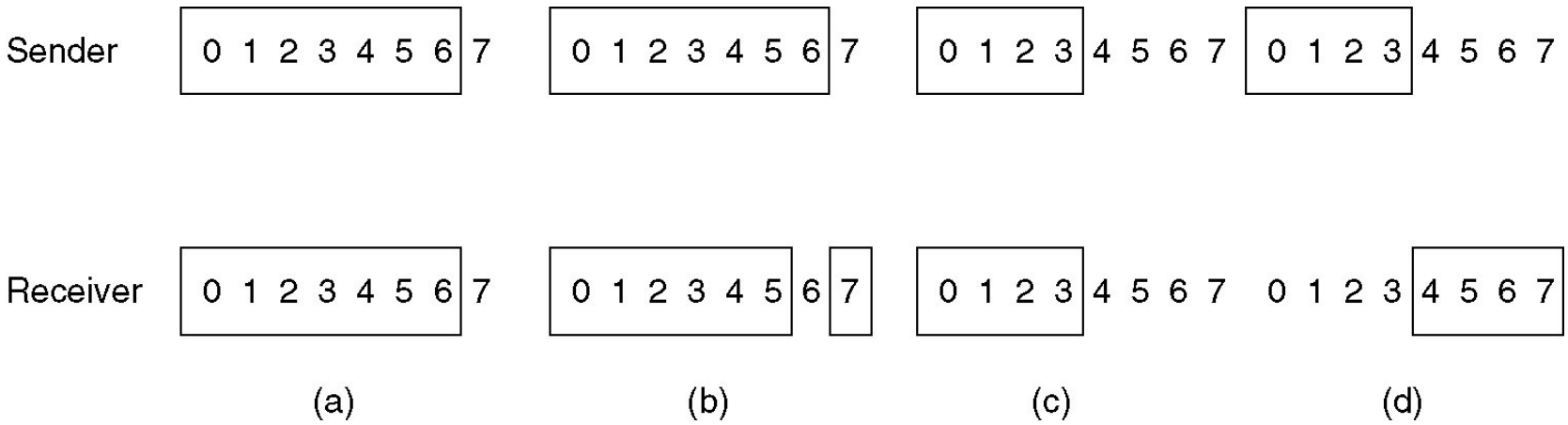
(a) Situação inicial com tamanho de janela = 7

(b) Depois de transmitir e receber 7 quadros cujos ACKs não chegaram

(c) Situação inicial com tamanho de janela = 4

(d) Depois de transmitir e receber 4 quadros cujos ACKs não chegaram

A Sliding Window Protocol Using Selective Repeat



(c) Situação inicial com tamanho de janela = 4

(d) Depois de transmitir e receber 4 quadros cujos ACKs não chegaram – 8 números de sequência (0 a 7)

Selective Repeat: Para janela de tamanho n, precisa de (2n) números de sequência