

PTC 3360

2.2 Camada de transporte: princípios da transferência confiável de dados – Parte II

(Kurose, Seção 3.4)

Agosto 2022

Capítulo 2 - Conteúdo

2.1 A camada de aplicação

2.2 A camada de transporte: princípios da transferência confiável de dados

2.3 A camada de rede

rdt3.0: Canais com erros e perdas de pacotes

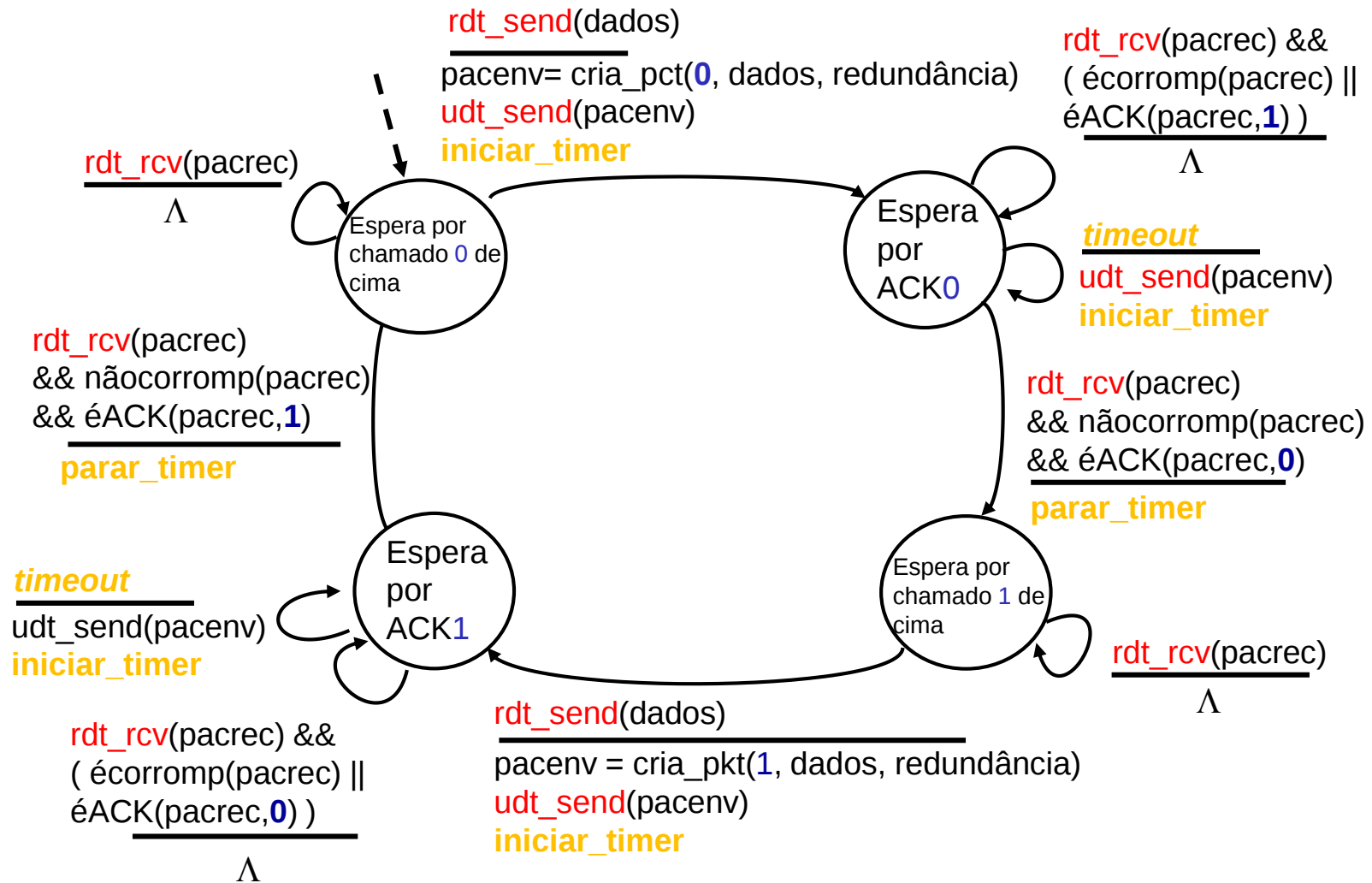
Nova hipótese: canal subjacente também pode perder pacotes (dados, ACKs)

- *Redundância*, $\#seq$, ACKs, retransmissões ajudarão... mas não são suficientes.

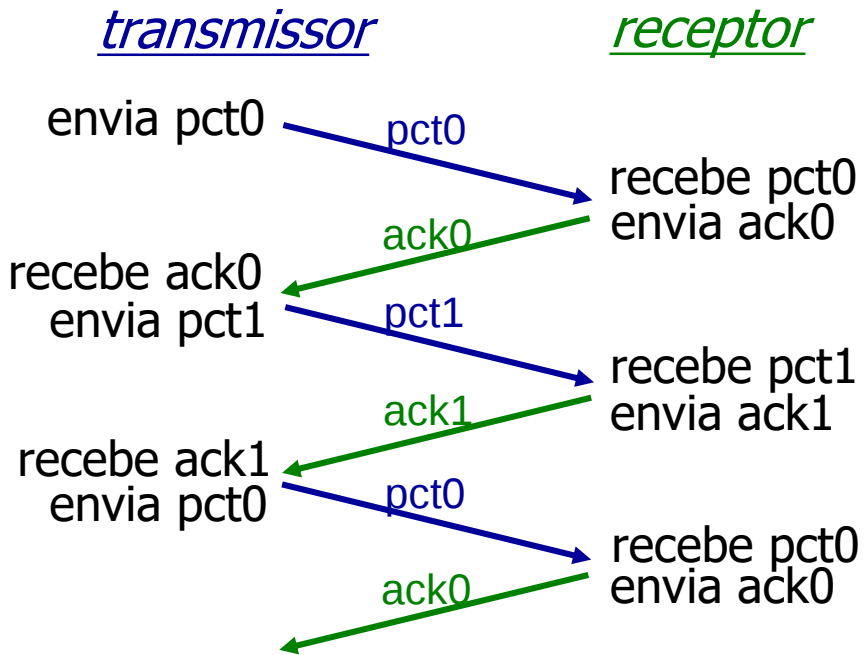
Abordagem: transmissor espera tempo “razoável” por ACK

- ❖ Retransmite se ACK não é recebido nesse tempo
- ❖ Se pacote (ou ACK) apenas atrasado (não perdido):
 - retransmissão será duplicada, mas $\#seq$ já lida com isso
 - destinatário precisa especificar $\#seq$ de pacote no ACK
- ❖ Requer *temporizador* de contagem regressiva

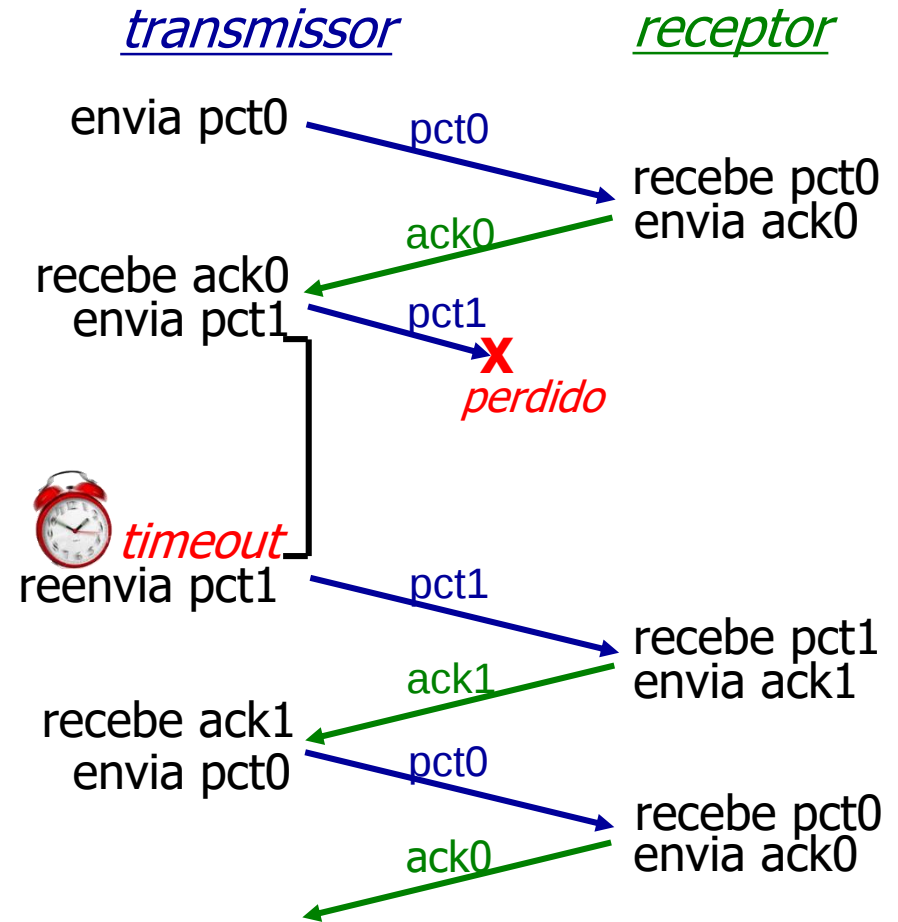
rdt3.0: Máquina do Transmissor (Receptor é Exercício)



rdt3.0 em ação (*protocolo bit alternante*)

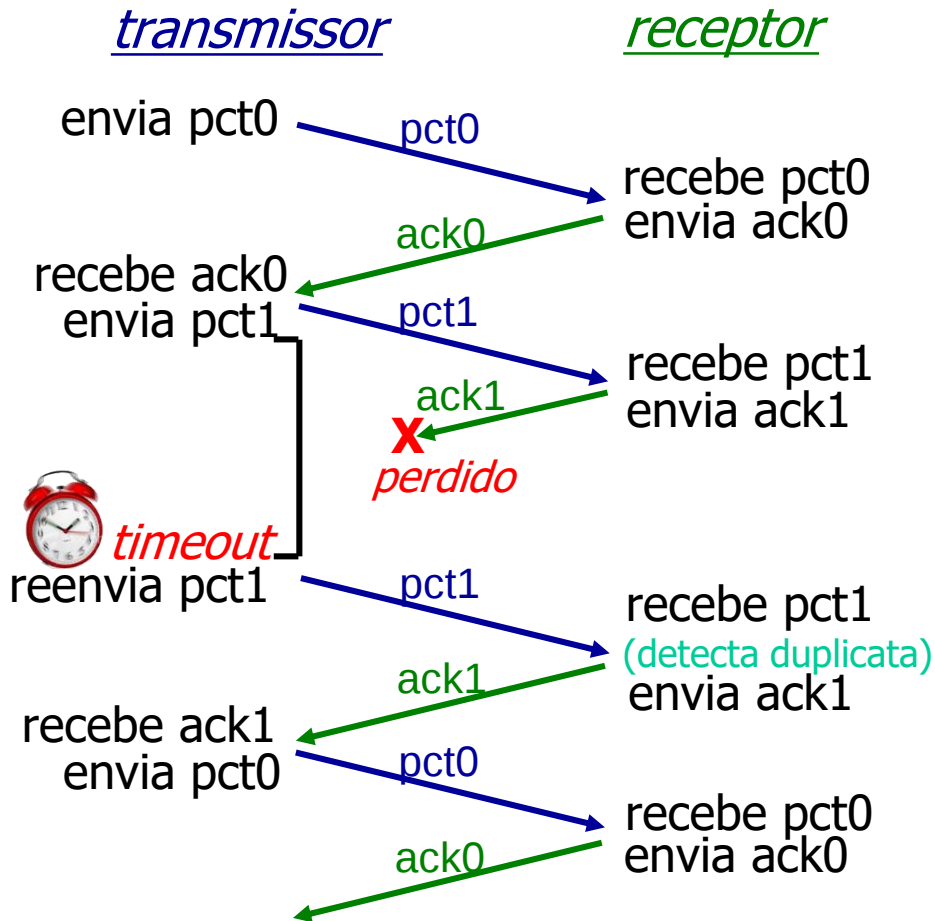


(a) sem perda

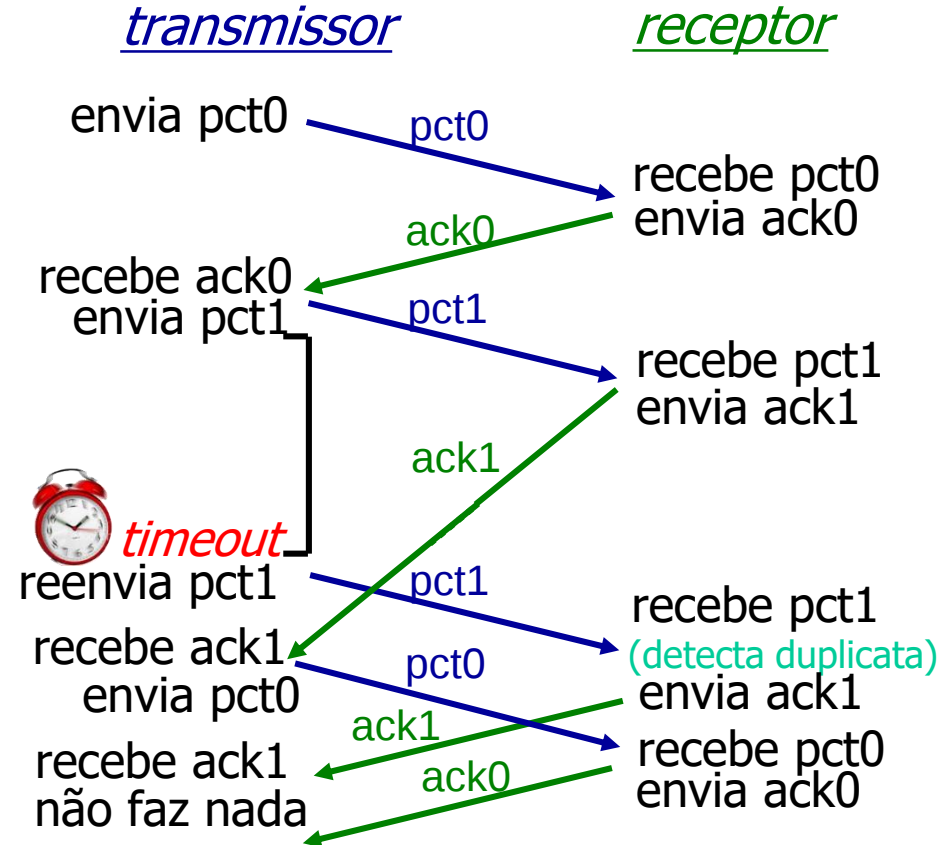


(b) perda de pacote

rdt3.0 em ação



(c) perda de ACK



(d) *timeout* prematuro/ ACK atrasado

Desempenho do rdt 3.0

- ❖ rdt 3.0 está correto, mas desempenho sofrível!
- ❖ Exemplo: Enlace de 1 Gbps, RTT 30 ms, pacote de 8 000 bits
- ❖ Atraso de Transmissão:

$$d_{\text{trans}} = \frac{L}{R} = \frac{8000 \text{ bits}}{10^9 \text{ bits/s}} = 8\mu\text{s}$$

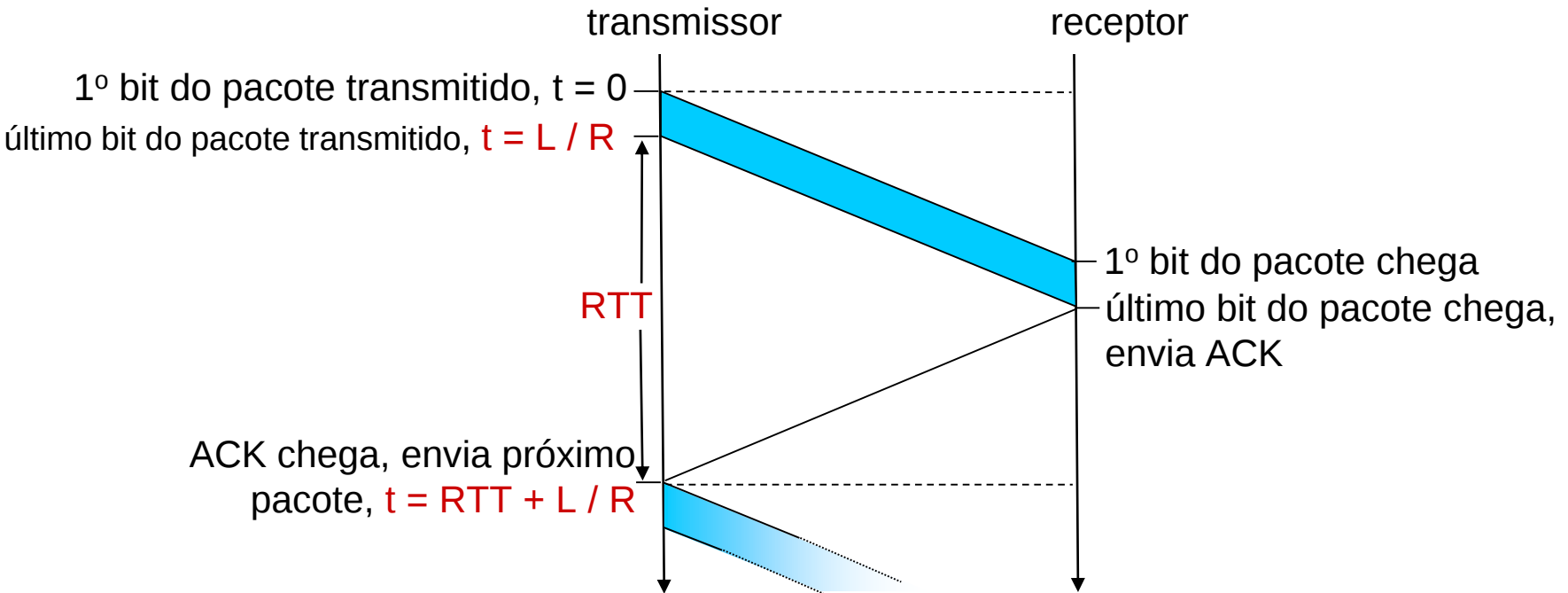
- U_{trans} : utilização – fração do tempo em que transmissor está ocupado enviando

$$U_{\text{trans}} = \frac{\frac{L}{R}}{RTT + \frac{L}{R}} = \frac{0,008}{30,008} = 0,00027$$

■ Vazão

- Um pacote enviado a cada aproximadamente 30 ms: vazão de 267 kbps em um enlace de 1 Gbps (!!!)
- ❖ Protocolo de rede limita uso de recurso físico!
- ❖ Preço pago para ter RDT (muito caro?)

rdt3.0: operação stop-and-wait



$$U_{\text{transmissor}} = \frac{\frac{L}{R}}{RTT + \frac{L}{R}} = \frac{0,008}{30,008} = 0,00027$$

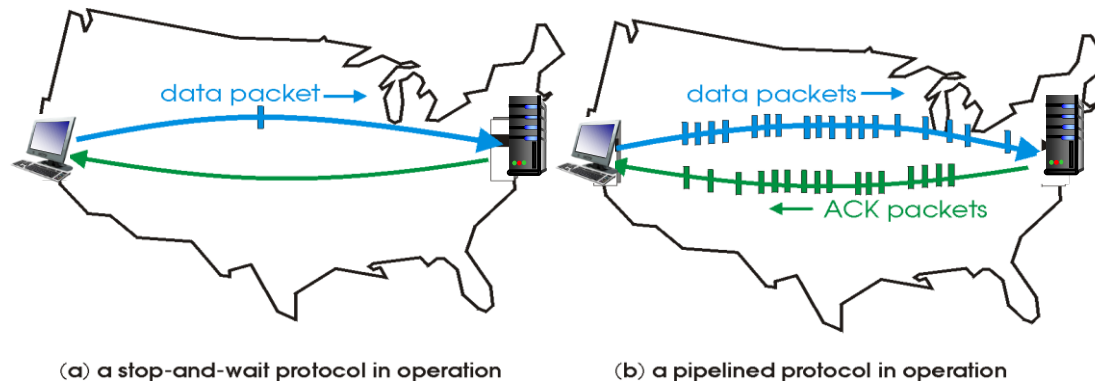
Exercícios interativos

- ❖ Exercício sobre o RDT 3.0 do livro do Kurose

Protocolos com paralelismo (*pipelining*)

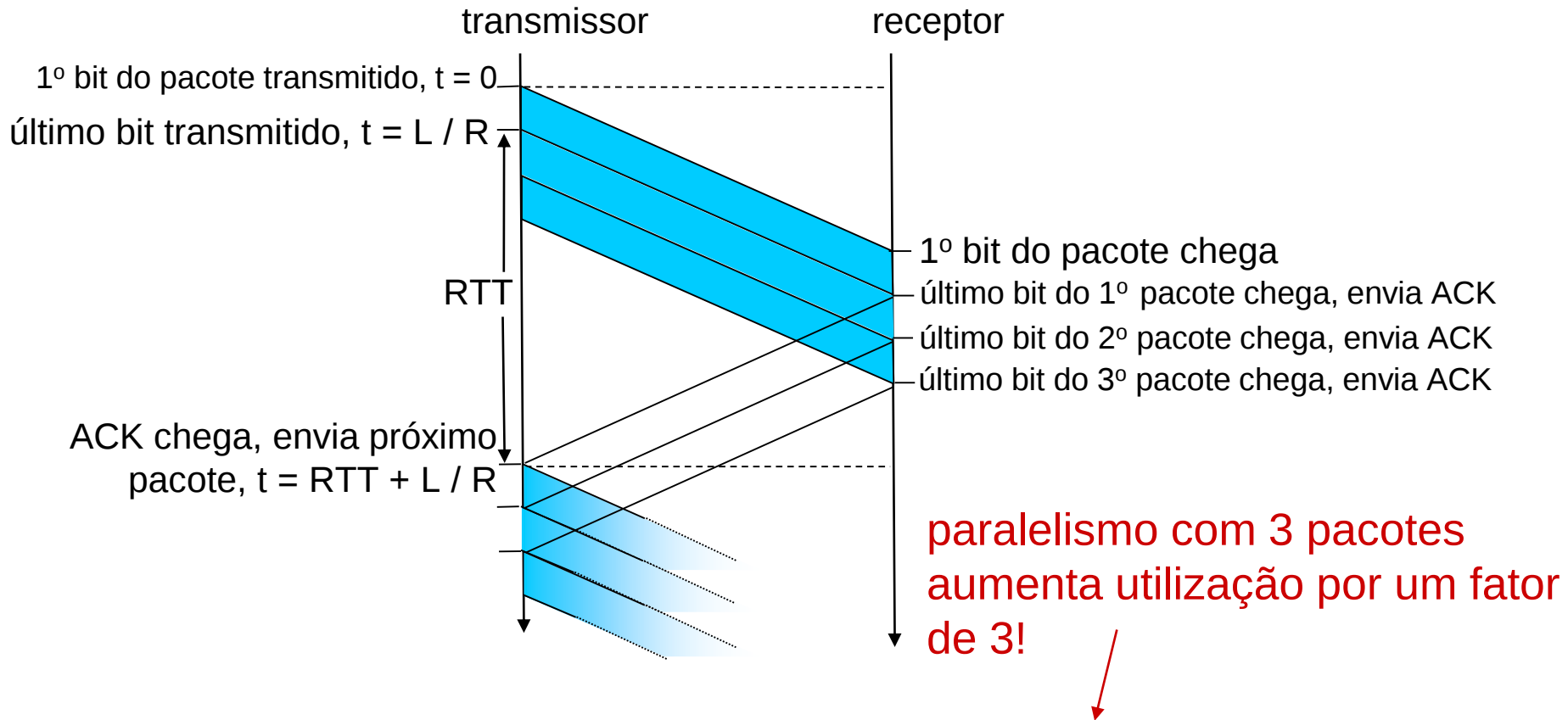
Pipelining: transmissor permite múltiplos, pacotes “*in-flight*”, ainda não reconhecidos (*acknowledged*)

- intervalo dos números sequenciais precisa ser aumentada
- *buffers* no transmissor e/ou receptor



- ❖ 2 formas genéricas de protocolos com paralelismo:
go-Back-N, repetição seletiva

Paralelismo: utilização aumentada



Qual a utilização do remetente agora?

$$U_{\text{transmissor}} = \frac{3 \frac{L}{R}}{\text{RTT} + \frac{L}{R}} = \frac{0,024}{30,008} = 0,00081$$

Técnicas básicas de paralelismo

Go-back-N (GBN):

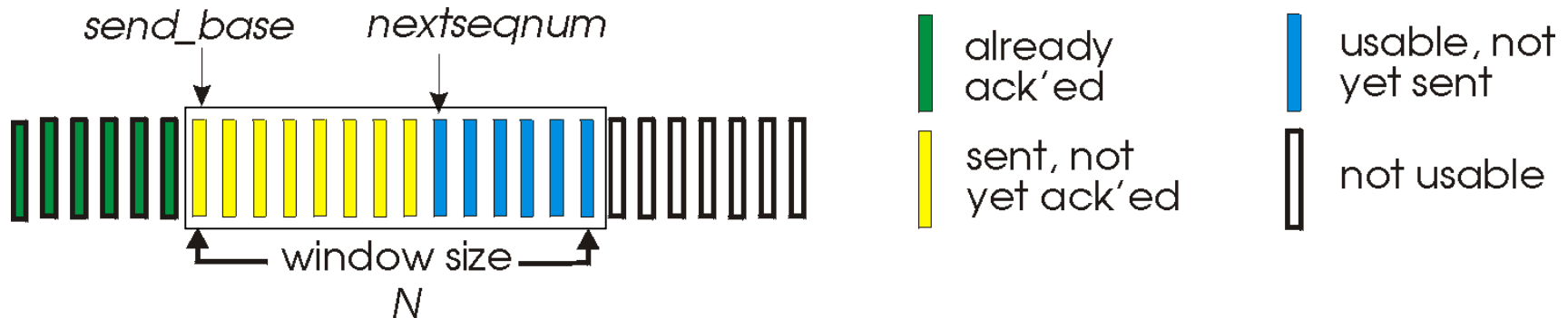
- ❖ Transmissor pode ter até N pacotes que ainda não retornaram ACKs
- ❖ Receptor envia **ACK acumulativo**
 - ACK x implica que todos os pacotes até x foram recebidos
- ❖ Transmissor tem apenas um temporizador associado ao pacote não reconhecido mais antigo
 - quando tempo expira, retransmite *todos* pacotes não reconhecidos

Repetição Seletiva (RS):

- ❖ Transmissor pode ter até N pacotes que ainda não retornaram ACKs
- ❖ Receptor envia **ACK individual** para cada pacote
- ❖ Transmissor mantém temporizador para cada pacote ainda não reconhecido
 - quando tempo expira, retransmite apenas aquele pacote associado com o temporizador expirado

Go-Back-N: Transmissor

- ❖ número sequencial de k -bits no cabeçalho do pacote
- ❖ “janela” de até N pacotes consecutivos ainda não reconhecidos permitida



- ❖ ACK(n): quando recebido, reconhece recepção de todos os pacotes com número sequencial menor ou igual a n – “ACK acumulativo”
 - pode receber ACKs duplicados
 - temporizador para pacote não reconhecido mais antigo
- ❖ **timeout**: retransmite todos os pacotes ainda não reconhecidos
- ❖ É um *protocolo de janela deslizando*

GBN em ação

janela transmissor (N=4)

0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8

transmissor

envia pacote 0
envia pacote 1
envia pacote 2
envia pacote 3
(espera)

rcb ack0, envia pct4
rcb ack1, envia pct5

ignora ACK duplicado



pct 2 timeout

envia pct2
envia pct3
envia pct4
envia pct5

receptor

rcb pct0, entrega, envia ack0
rcb pct1, entrega, envia ack1

recebe pacote 3, descarta,
(re)envia ack1

recebe pacote 4, descarta,
(re)envia ack1

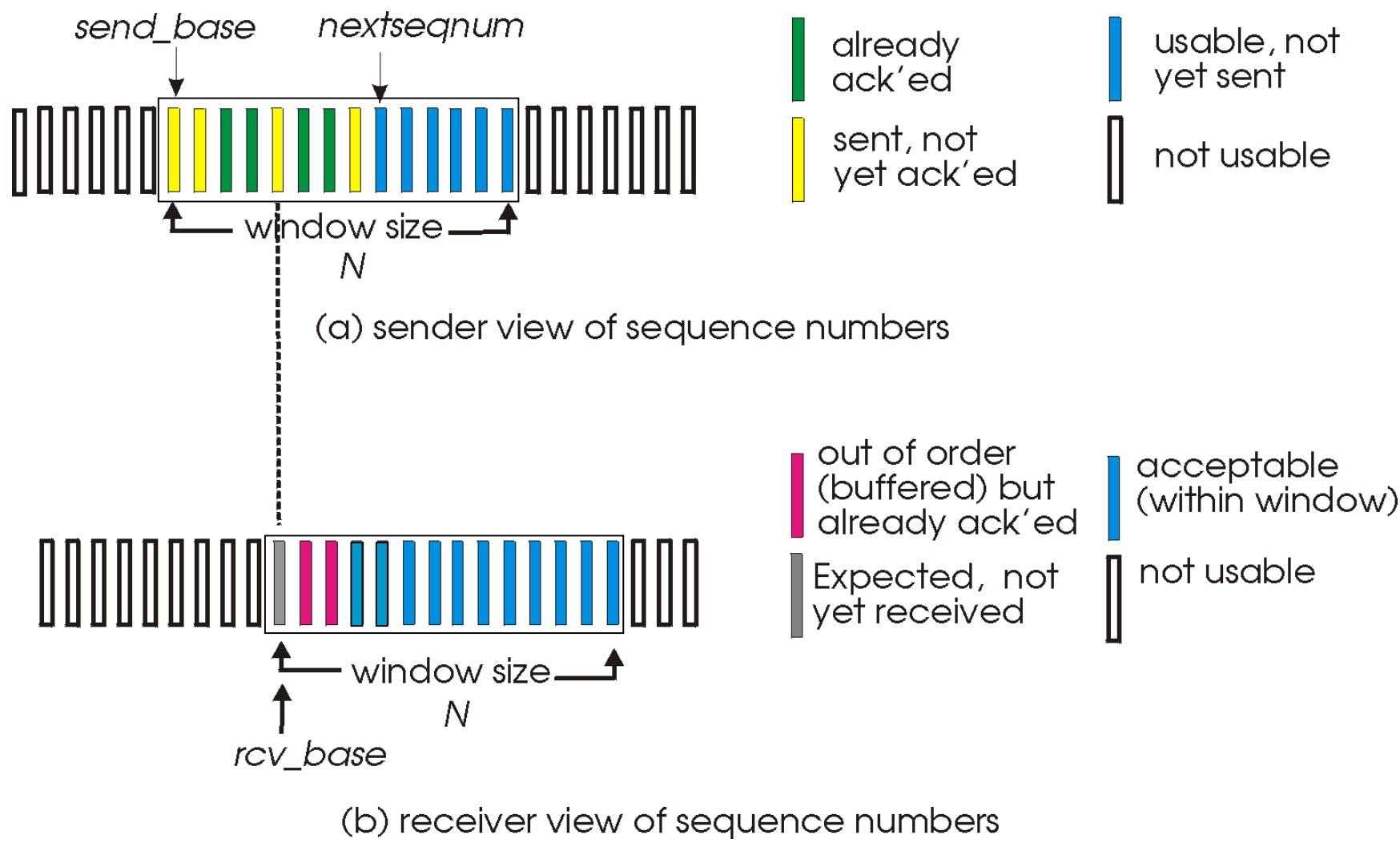
recebe pacote 5, descarta,
(re)envia ack1

rcb pct2, entrega, envia ack2
rcb pct3, entrega, envia ack3
rcb pct4, entrega, envia ack4
rcb pct5, entrega, envia ack5

Repetição Seletiva

- ❖ **Receptor** reconhece *individualmente* cada pacote recebido corretamente
 - **pacotes colocados em *buffer***, conforme necessário, para eventual entrega em ordem para camada superior
- ❖ **Transmissor** apenas reenvia pacotes para os quais ACK não foi recebido
 - **temporizador para cada pacote não ACK**
- ❖ **Janela deslizante**
 - janelas podem estar deslocadas entre remetente e destinatário
 - **mais números sequenciais necessários**

Repetição seletiva : janelas transmissor, receptor



Repetição seletiva em ação

janela remetente (N=4)

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

0 1 2 3 4 5 6 7 8

remetente

envia pacote 0

envia pacote 1

envia pacote 2

envia pacote 3

(espera)

rcb ack0, envia pct4

rcb ack1, envia pct5

guarda chegada do ack3



pct 2 timeout

envia pacote 2

guarda chegada do ack4

guarda chegada do ack5

destinatário

recebe pct 0, entrega e envia ack0

recebe pct 1, entrega e envia ack1

recebe pacote 3, *buffer*,
envia ack3

recebe pacote 4, *buffer*,
envia ack4

recebe pacote 5, *buffer*,
envia ack5

recebe pct2; entrega pct2,
pct3, pct4, pct5; envia ack2

Q: o que acontece quando ack2 chega?

Resposta: janela deslocada para n = 6

Repetição seletiva: dilema

Exemplo:

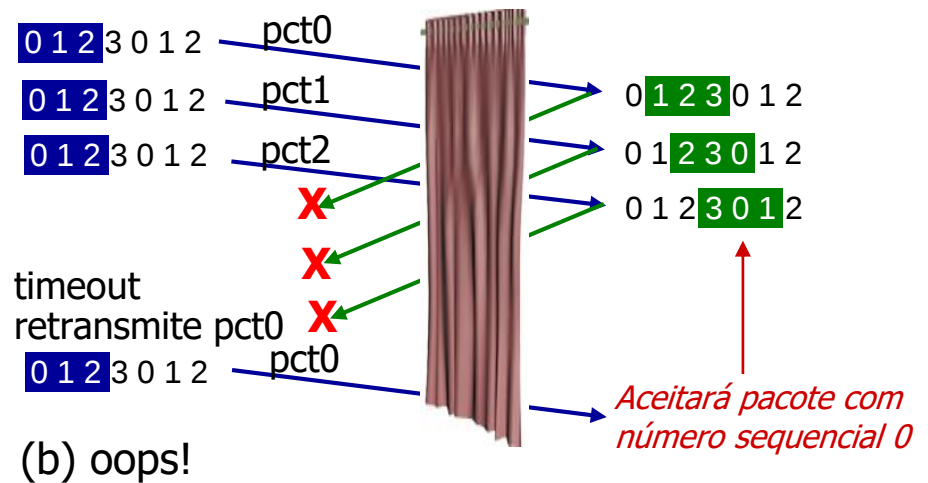
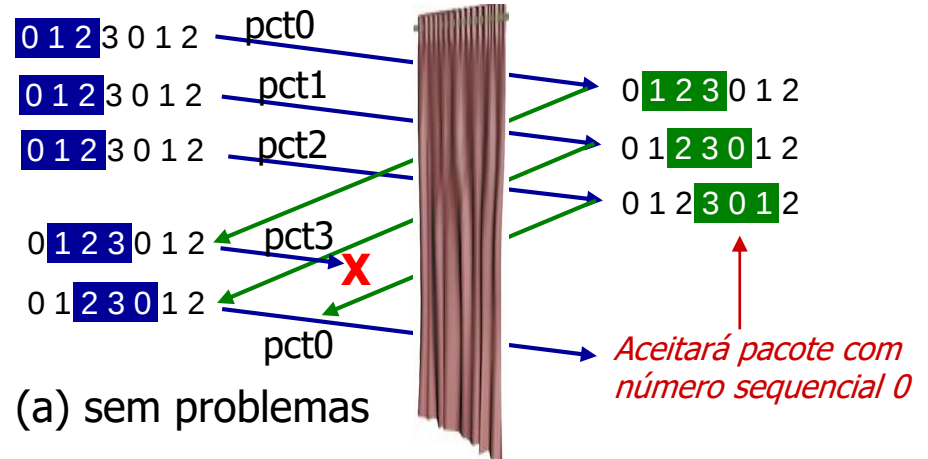
- ❖ $N=3$ e $\#seq : 0, 1, 2, 3$
- ❖ Receptor não vê diferença entre dois cenários!
- ❖ Dados duplicados aceitos como novos em (b)

Q: Qual a relação entre números sequenciais e N para evitar problema em (b)?

Resposta: Escolher intervalo de comprimento $2N$ para $\#seq$

Janela transmissor
(depois de receber)

Janela receptor
(depois de receber)



Comparação: Vantagens do GBN e da Repetição Seletiva

❖ GBN

- Menor complexidade – apenas um temporizador; não há necessidade de janelas no receptor 😊
- Não são necessário *buffers* 😊
- Muitas retransmissões desnecessárias 😞

❖ Repetição seletiva

- Maior complexidade – um temporizador por pacote 😞
- *Buffers* necessários para armazenar pacotes fora de ordem no receptor e ACKs no transmissor 😞
- Menos retransmissões – apenas pacotes de fato necessários são retransmitidos (pacote ou ACK perdidos) 😊

❖ Qual é usado nos protocolos práticos em geral?

Resposta: São casos extremos de simplicidade e complexidade, respectivamente. Veremos que o TCP usa uma combinação deles...