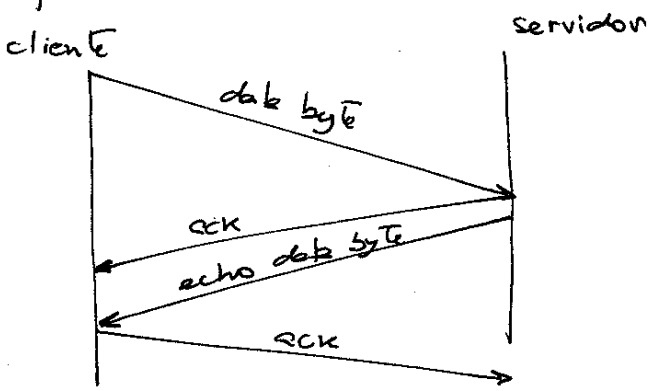
PROTOCOLOS FUNDAMENTAIS DA INTERN. ET
AULA 8
TCP TRANSMISSÃO DE DADOS INTERACTIVA
" " " EM MASSA (BULK)

A maior parte das sessões interactivas geram um byte (caracter) — o segmento TCP contem apenas 1 byte de dados.
O servidor responde com outro caracter — o "eco"



LAB

telnet carneiro echo
d
a
t
e
\n

rlogin carneiro
d
a
t
e
\n

tcp dump host carneiro

Delayed Acknowledgment (Confirmação atrazada)

Para economizar datagramas TCP, o protocolo espera até 200 ms (na esperança de haver dados para enviar) o envio da confirmação — é o piggyback do ACK

Algoritmo de Nagle - não enviar segmentos pequenos se estiver à espera de um ACK
- não tem efeito em LAN onde o tempo de resposta (round trip time) é muito curto (cc 16 ms)

- tem efeito em WAN (tempo de resposta longo)

(os caracteres vão-se acumulando no buffer de saída) e são enviados todos de uma só vez poupando datagramas

TAMANHO DA JANELA (WINDOW SIZE)

- indica o número de bytes que podem ser enviados sem saturar (overflow) o buffer de entrada

- não é relevante em comunicações interactivas

LAB (verificar window size durante uma sessão telnet, rlogin echo)

TRANSMISSÃO DE DADOS EM QUANTIDADE

TCP utiliza o sliding window protocol para transmitir dados em modo contínuo

- transmite vários segmentos — não ultrapassando o window size — sem esperar por ACK
(comparar com um stop-and-wait protocol)

fundamental para optimizar a cadência de dados em WAN com tempos de resposta muito longos

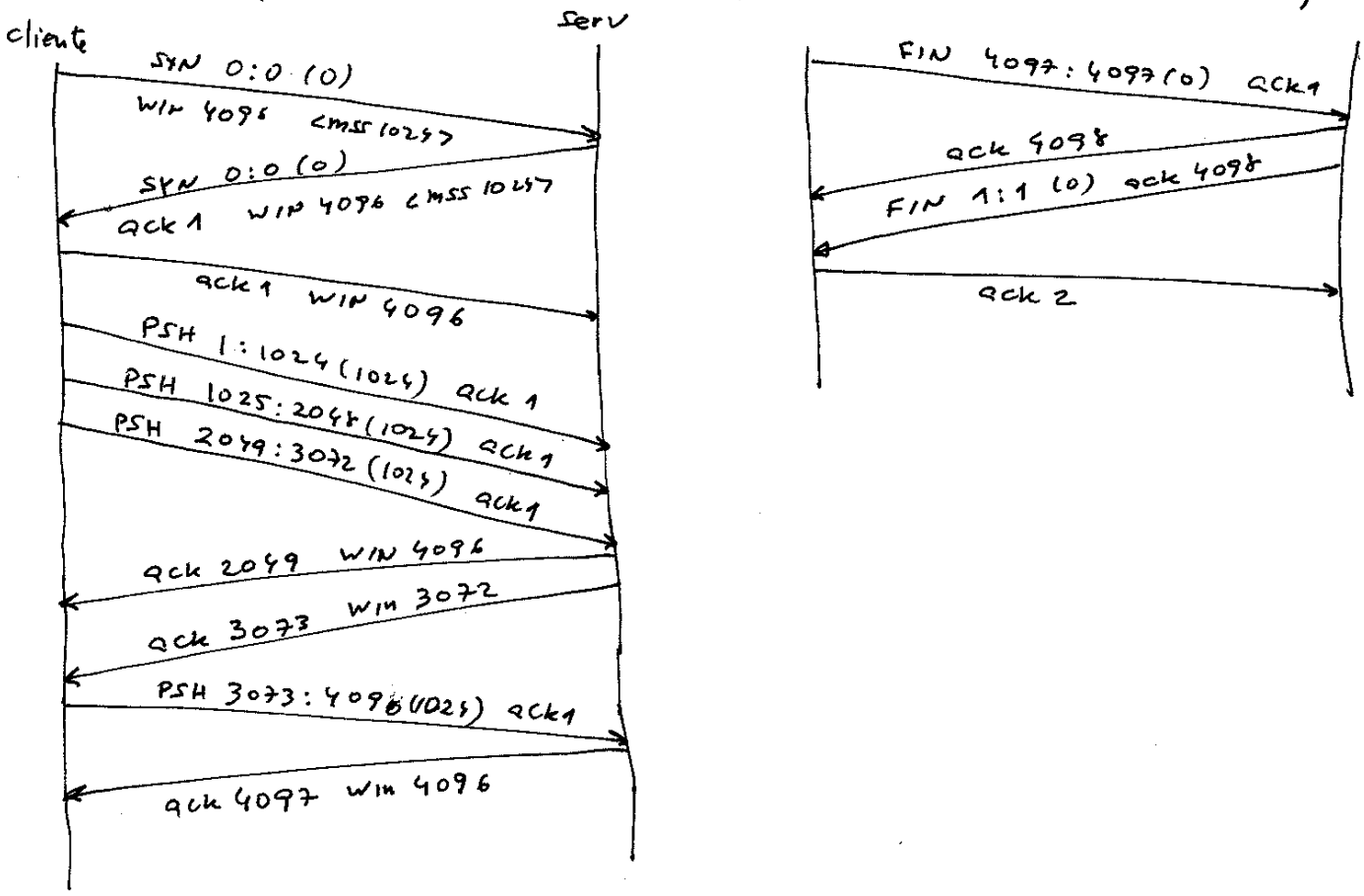
- o datagrama de ACK confirma a recepção de vários segmentos ~~datagramas~~ simultâneamente

LAB stevens 276

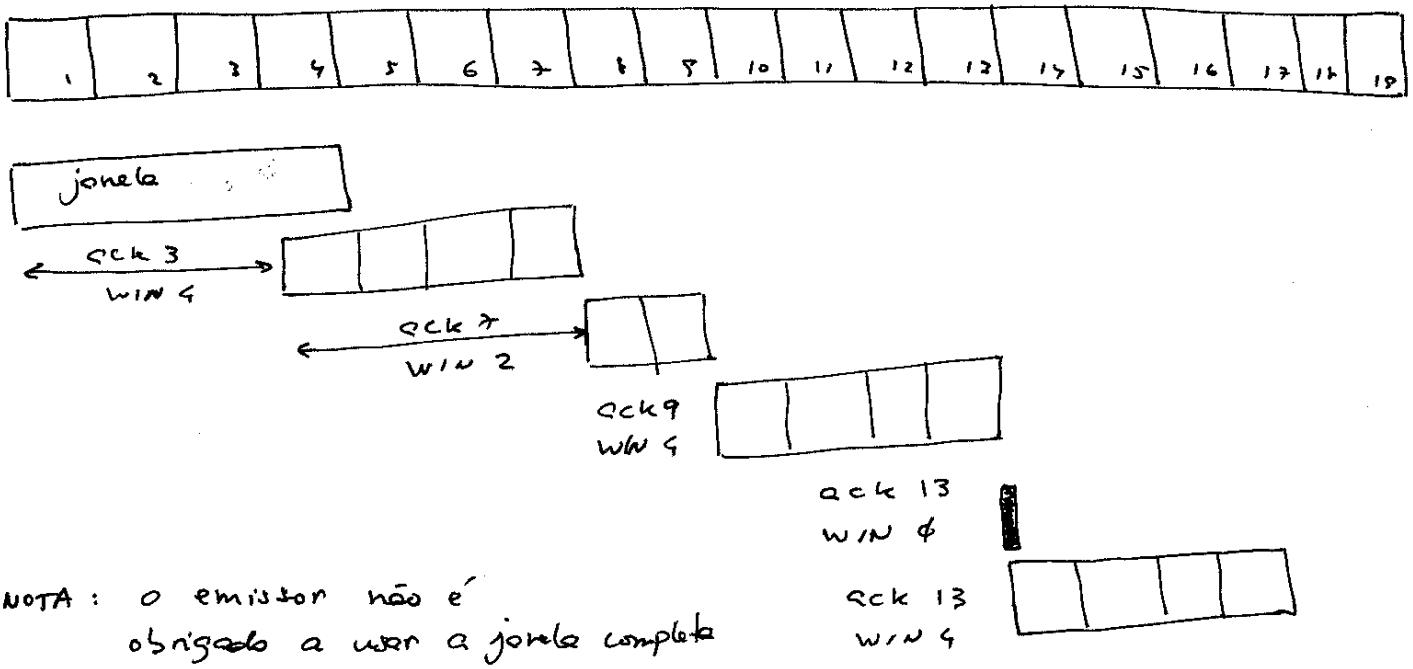
sock -i -s 7777 (servidor)

sock -i -n8 servidor 7777

tcpdump (src servidor dst cliente) or (src cliente dst servidor)



PROTOCOLO DA JANELA DESLIZANTE (SLIDING WINDOW) stevens pag 284



NOTA: o emissor não é obrigado a usar a janela completa

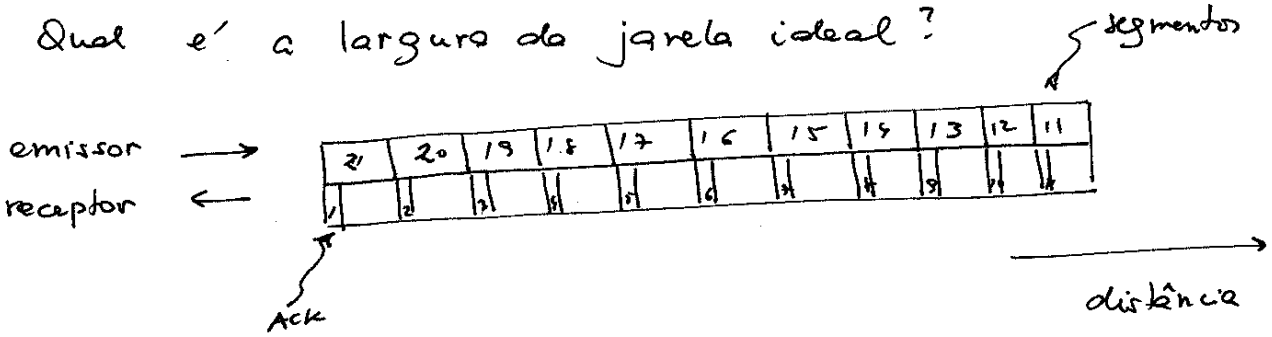
SLOW START ALGORITMO

- o emissor começa por enviar um segmento. Quando recebe o primeiro ACK envia dois novos segmentos. Quando recebe o ACK para estes dois segmentos envia quatro novos segmentos

Há um crescimento exponencial limitado pelo window size.

PRODUTO ATRASO - LARGURA DE BANDA

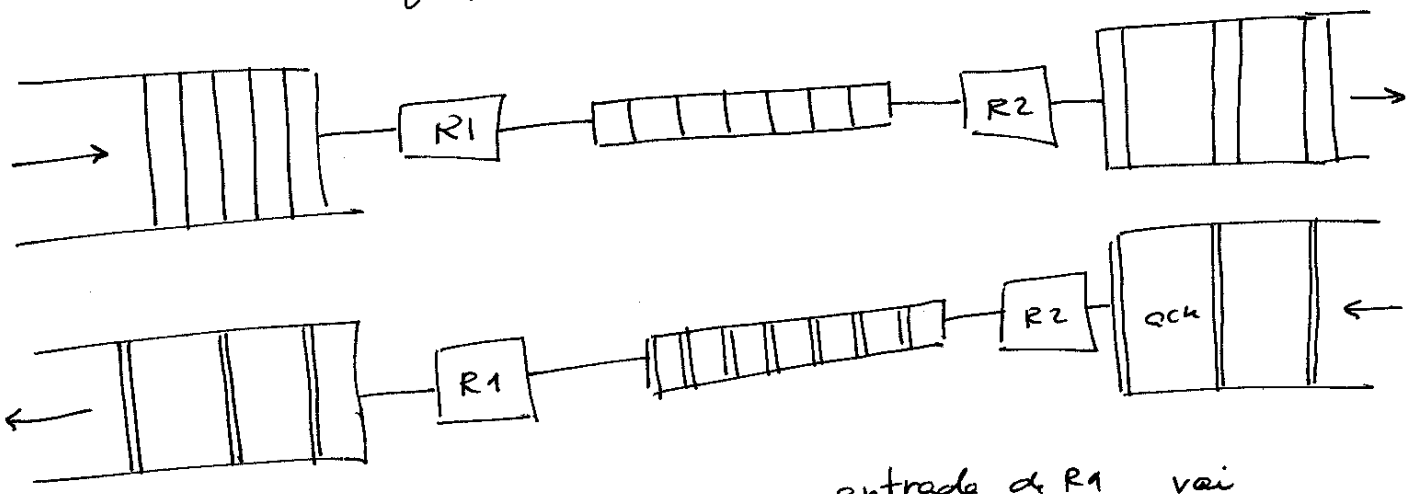
Qual é a largura da janela ideal?



LAB

capacidade do canal (bits) = BW (bits/s) x round trip time (s)

CONGESTÃO - ligação de grande capacidade ligada a ligação de fraca capacidade



Há congestão porque o buffer de entrada de R1 vai acumulando pacotes que não consegue enviar. A congestão acontece quando o buffer de entrada ficar cheio e começar a recusar (drop) pacotes

CONGESTÃO ≡ PERDA DE PACOTES

Como é detectada a perda de pacotes?

- time out (não estão a chegar ACK) — caso + grave
- estão a chegar ACKs em duplicado — caso - grave

ALGORITMO DE CONGESTÃO

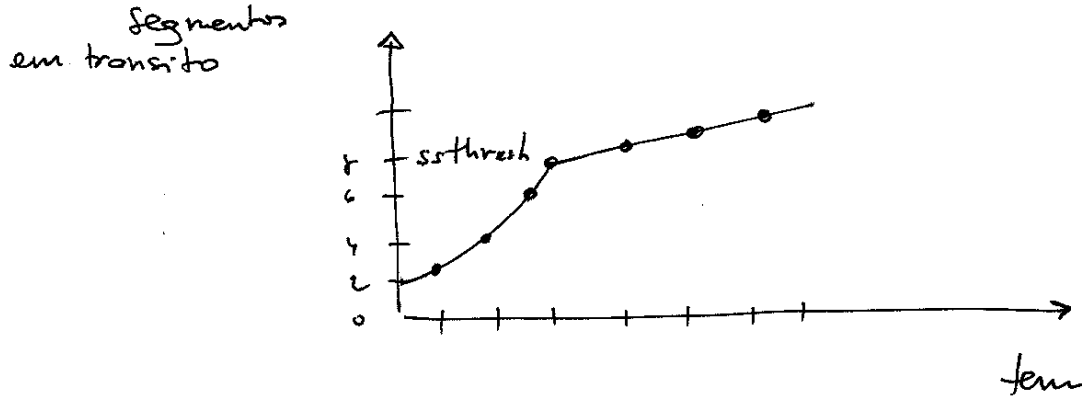
(pacotes perdidos detectados por timeout)

- slow start threshold (ssthresh) colocado a metade da janela actual (= valor dos segmentos em transito)
- retransmitir pacote; esperar ACK deste pacote
- valor (em bytes) dos segmentos enviados inferior a ssthresh?

Inferior: enviar 1 segmento, esperar Ack, enviar 2 segmentos, esperar ACK, enviar 4 segmentos até atingir ssthresh — crescimento exponencial

Superior: definir janela de congestão = ssthresh

Sempre que chegar um ack (não repetido) redifinir janela congestão ← janela congestão + 1 seg até atingir window size — crescimento linear



(detecção de pacotes perdidos por Ack repetidos)

Stevens 308

ALGORITHOS DE RETRANSMISSÃO RÁPIDA e RECUPERAÇÃO RÁPIDA

Retransmissão rápida - 3 (ou mais) Ack do mesmo segmento indica repetir esse segmento imediatamente

Recuperação rápida -

- colocar ssthresh a metade da janela actual
- colocar a janela actual a ssthresh + 3 segmentos
- sempre que chegar 2 Ack repetidos aumentar a janela actual de mais um segmento
- quando chegar um Ack que confirma a recepção do segmento retransmitido bem como de todos os enviados no tempo intermédio até terem chegado os 3 Ack repetidos

colocar janela actual = ssthresh — significa diminuir os segmentos em transito para metade daqueles que provocaram congestão

- aumentar de um segmento a janela actual por cada Ack até window size

RTT (Round trip time) Estimator

porquê? Valor do timeout depende do conhecimento correcto do RTT

$$Err = M - A$$

média ← $A \leftarrow A + g \times Err$ $\quad g = 0.125 \quad A_{inicial} = 0$

desvio da média ← $D \leftarrow D + h \times (|Err| - D)$ $\quad h = 0.25 \quad D_{inicial} = 3\,s$

$$RTO = A + 4D$$

Retransmission time Out

LAB window probes Stevens 32.3

— Testa se uma janela de ϕ é actualizada

Sock -i -s -P 100 000 5555 (server)

Sock -i -n 20 server 5555 (cliente)

LAB silly window syndrome Stevens 326

Sock -i -s -P4 -p2 -r 256 7777

Sock -i -n 6 server 7777

P4 pausa 4 seg antes do 1º read

p2 " 2 seg entre cada read