Transport Protocols for Networked Games

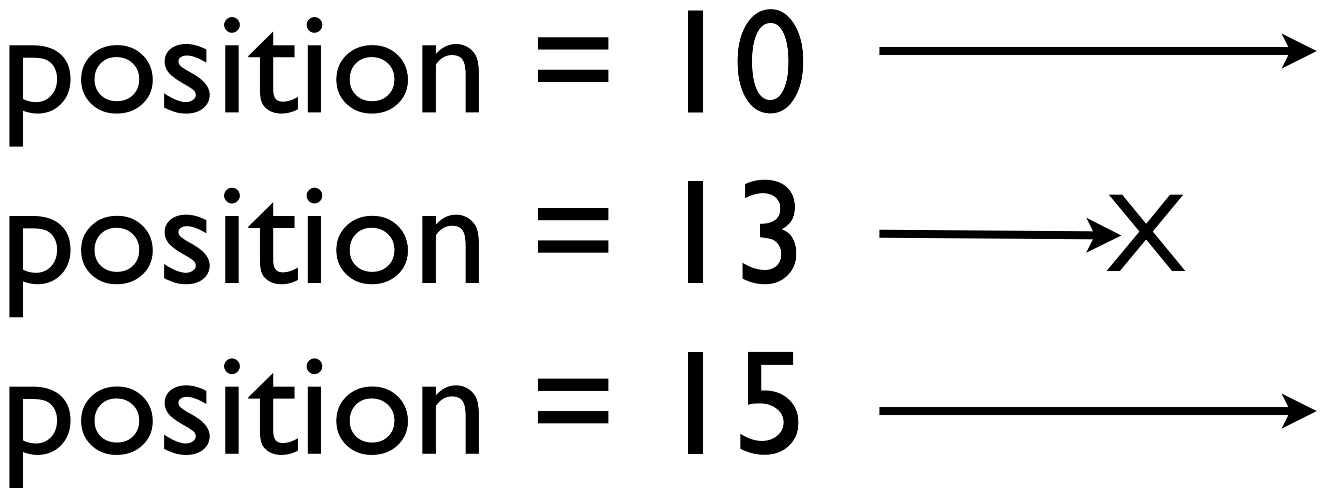
TCP or UDP ?

Why use TCP?

- TCP provides reliable, in-order delivery
- TCP goes through most firewalls, UDP does not
- TCP manages connection for us

Why not to use TCP?

- TCP incur higher latency
- Don’t always need reliability and in-order delivery
- High header overhead

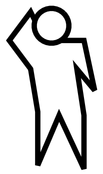


position = 10

position = 13

position = 15

Updated position not delivered to application until (outdated) lost packet is received



Gesture from someone far away need not be received reliably.

A study on ShenZhou Online shows that 46% of the bandwidth is occupied by TCP header

enet.cubik.org

A library that provides reliability, sequencing, connection managements over of

Delivery can be stream-oriented (like TCP) or message-oriented (like UDP)

Supports partial reliability

enet_packet_create (“abc”,
4, ENET_PACKET_FLAG_RELIABLE)

Retransmission triggered by Timeout based on RTT

Data in queue are bundled into one packet if there is space

enet.cubik.org

Portable, easy to use, but still, most firewalls block UDP traffic

- MMORPG that uses **TCP**: WoW, Lineage I/II, Guild Wars, Ragnarok Online, Anarchy Online, Mabinogi
- MMORPG that uses **UDP**: EverQuest, SW Galaxies, City of Heroes, Ultima Online, Asherons Call, FFXI

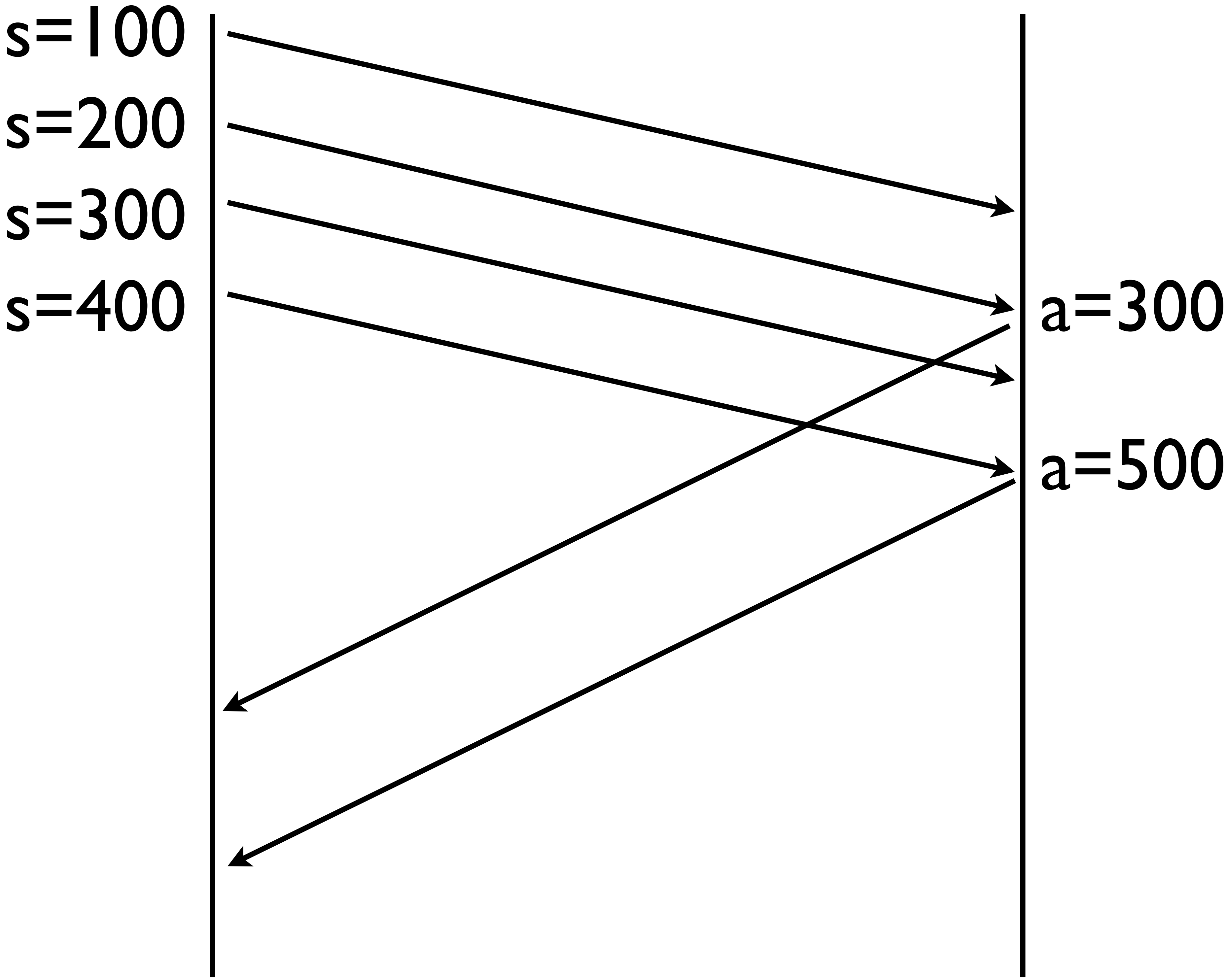
Need to study the use of TCP for networked games

How to provide reliability over UDP?

How slow is TCP, really?

Which part of TCP is the root of slowness?

A Quick Review of TCP



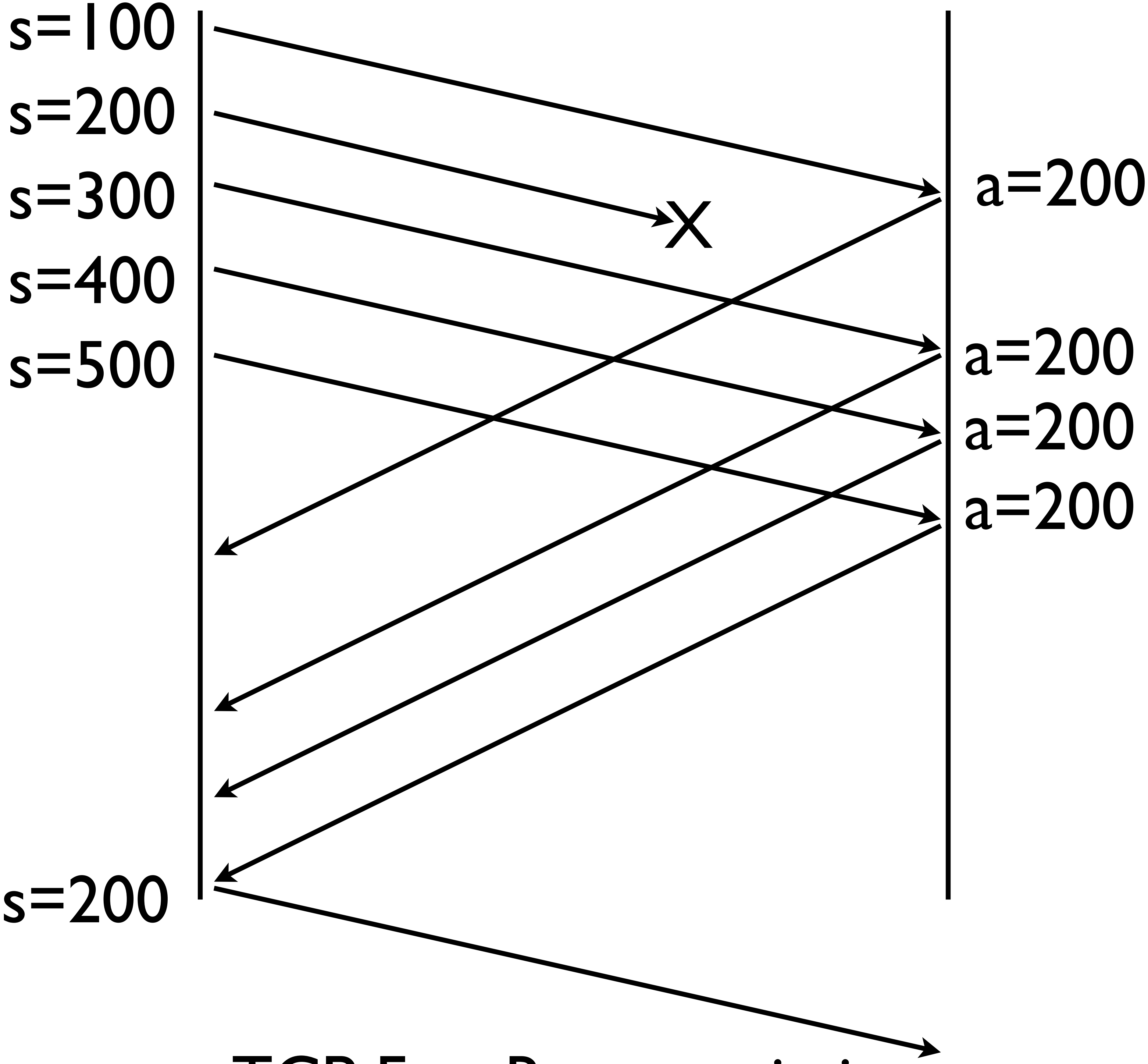
TCP Delayed ACK

TCP Spec: max **500ms** delay

Most implementation: **200ms**

Why delay ACK?

- reduce num of ACKs
- in case receiver wants to send data within 200ms (in which case it can piggyback the ACK with data)
- give sender time to buffer more data for sending (avoid silly window syndrome)



s=100

s=200

s=300

s=400

s=500

x

a=200

a=200

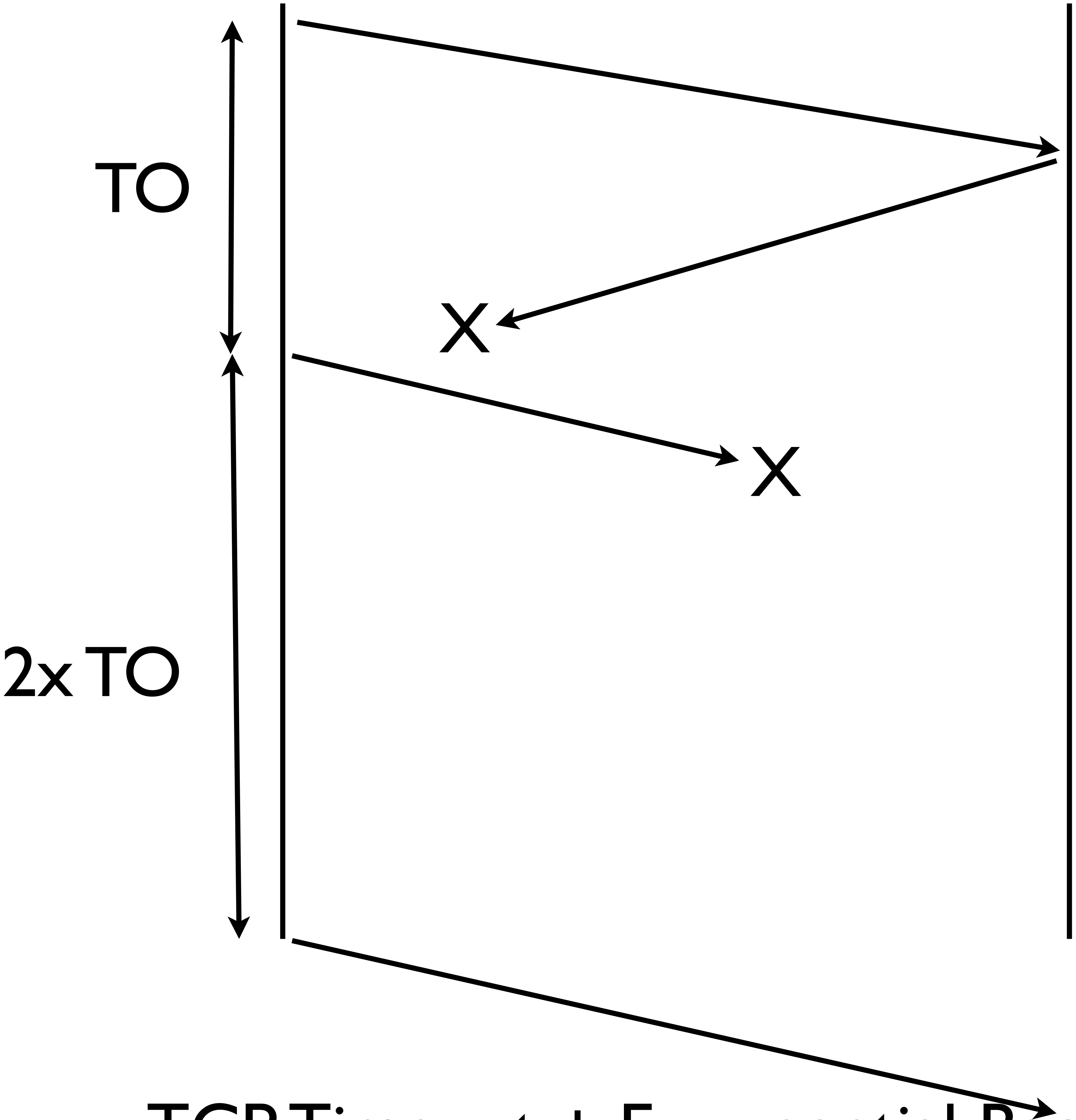
a=200

a=200

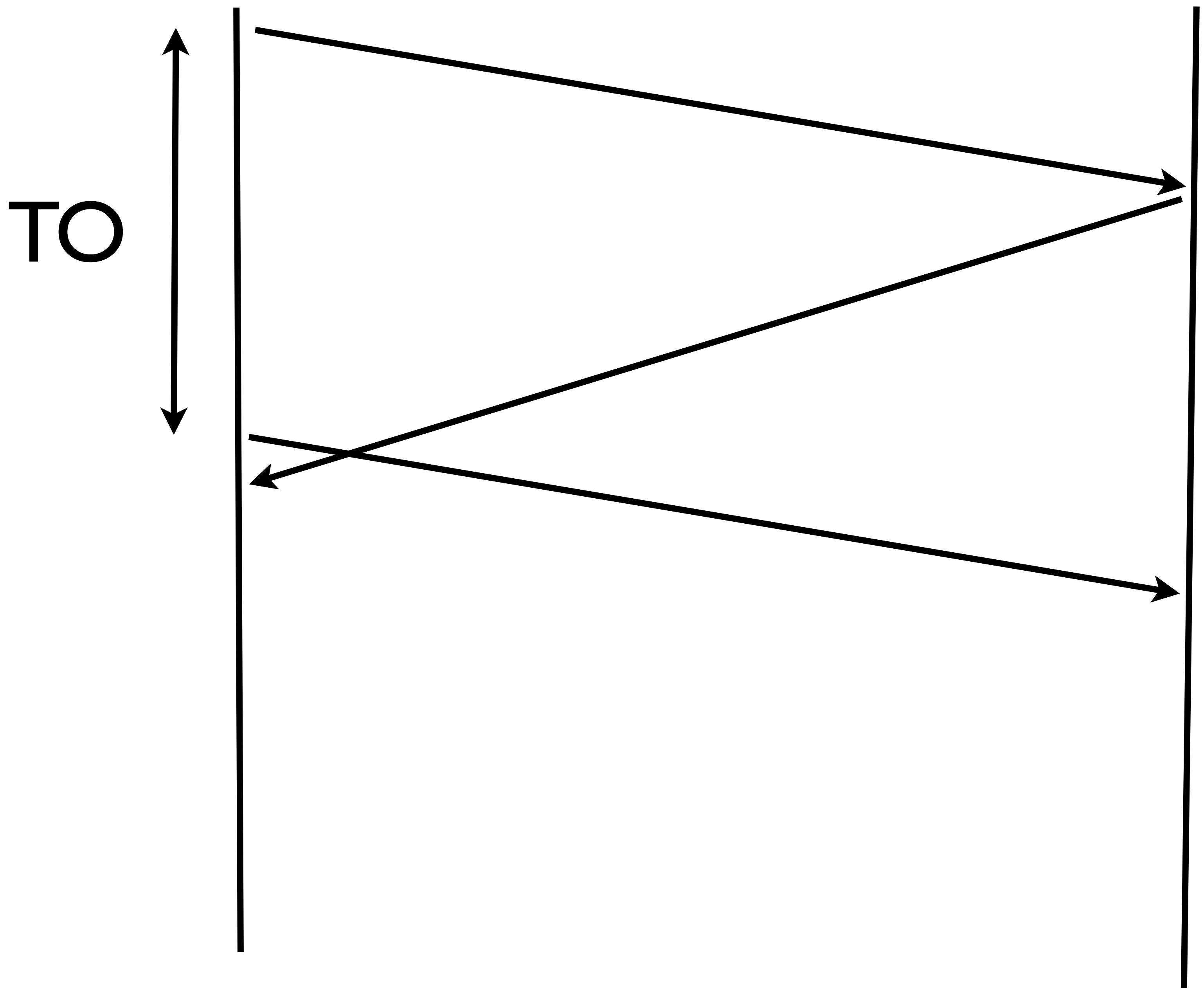
s=200

TCP Fast Retransmission

Definition of Dup ACKs in 4.4BSD and Al Stevens: “pure ACK, cannot piggyback with data”



TCP Timeout + Exponential Back-off



Spurious Retransmission

RTO estimation

$$E_i = 7E_{i-1}/8 + RTT/8$$

$$V_i = 3V_{i-1}/4 + |RTT-E_{i-1}|/4$$

$$RTO = \max(E_i + 4V_{i,}\ 1s)$$

Linux's RTO estimation

$E_i = 7E_{i-1}/8 + RTT/8$

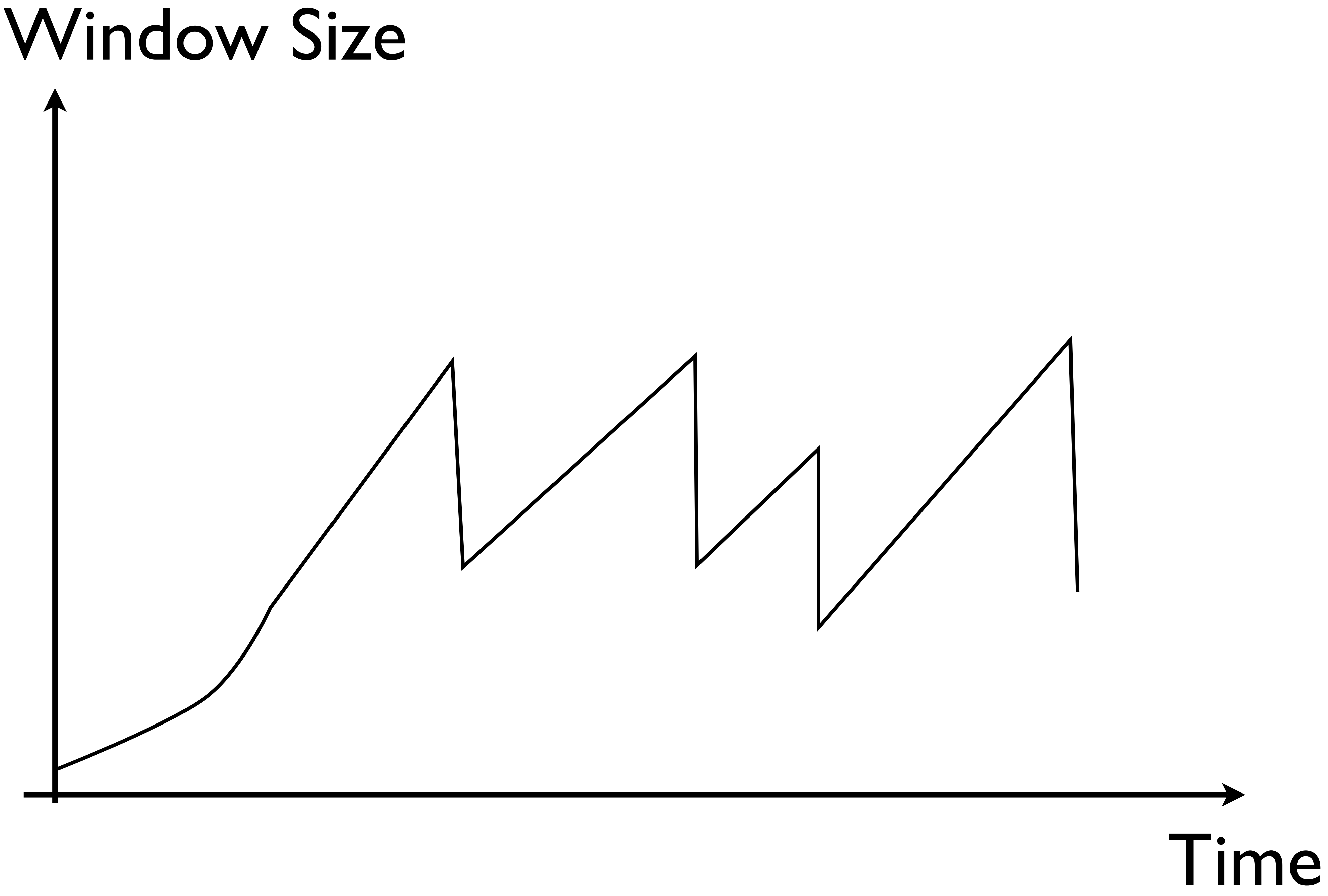
$V_i = 3V_{i-1}/4 + |RTT-E_{i-1}|/4$

$RTO = \max(200ms,$

$E_i + \min(V_i, 50ms)$

Note:
Delayed ACK =>
increase RTT =>
increase RTO

Congestion Control

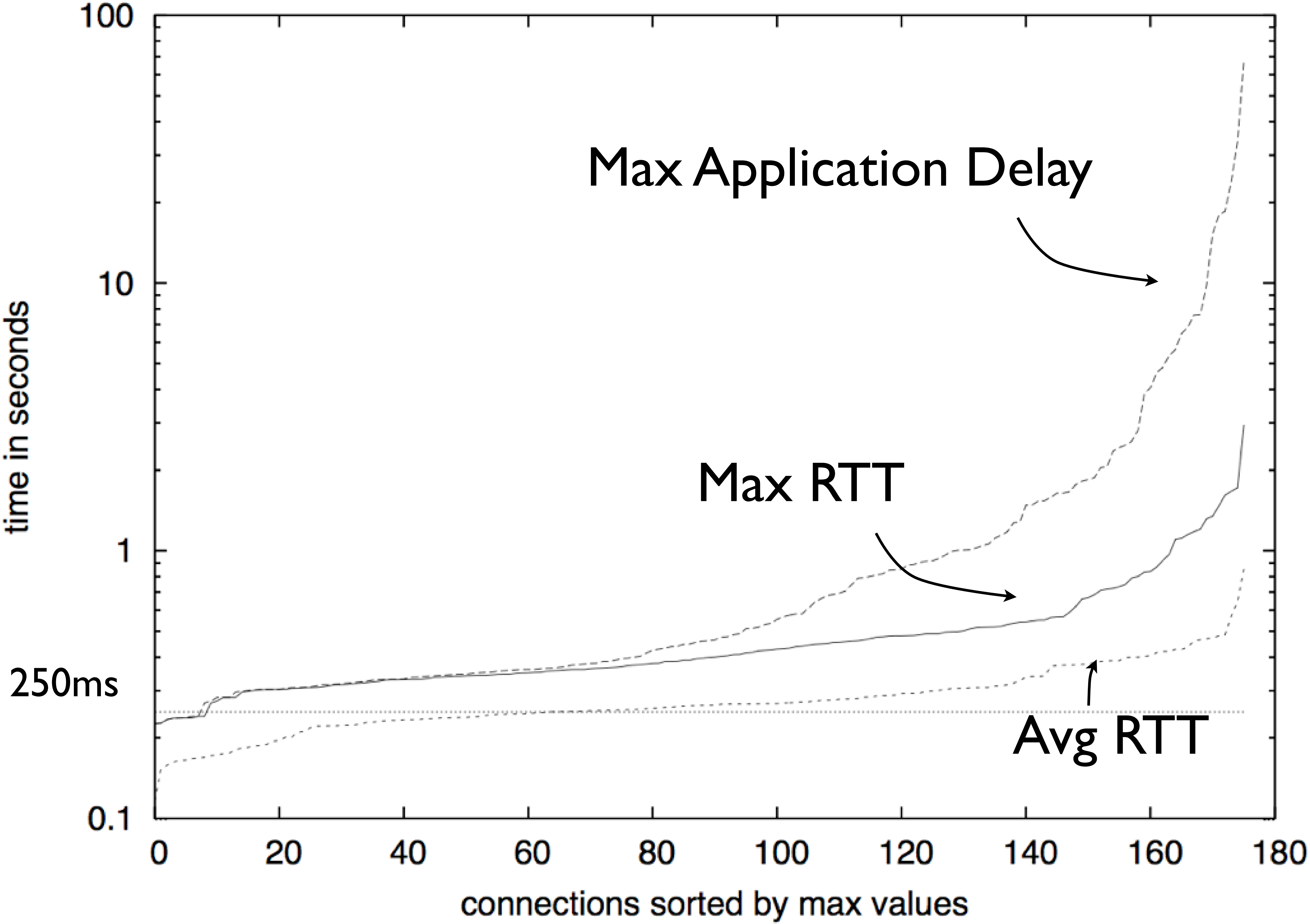


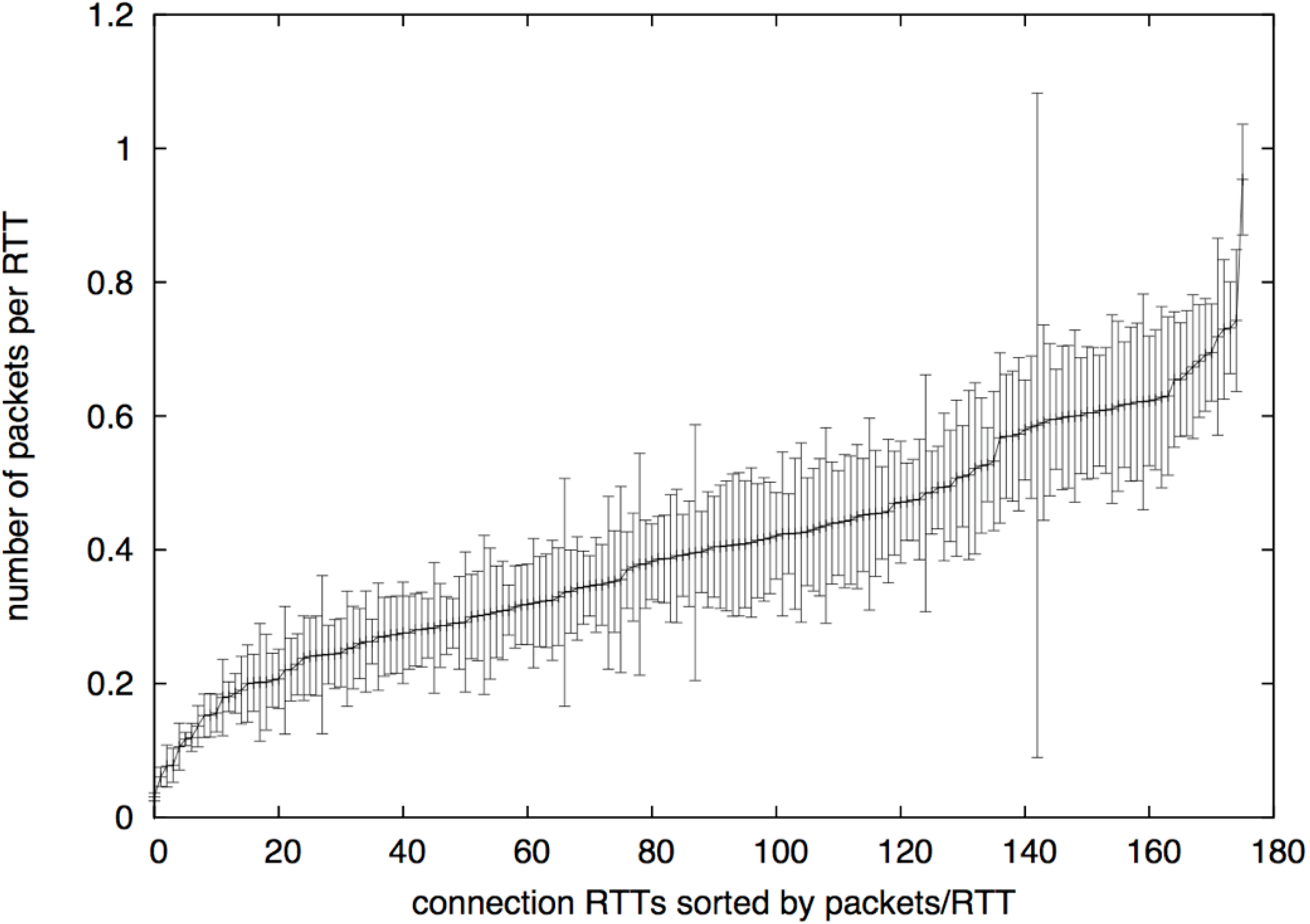
TCP Congestion Control

Congestion Window reset to 2 after an idle period (> RTO)

What does real game traffic look like?

Anarchy online


100
10
1
0.1
time in seconds
0
20
40
60
80
100
120
140
160
180
connections sorted by max values
Max Application Delay
Max RTT
Avg RTT
250ms


number of packets per RTT
0
0.2
0.4
0.6
0.8
1
1.2
0
20
40
60
80
100
120
140
160
180
connection RTTs sorted by packets/RTT

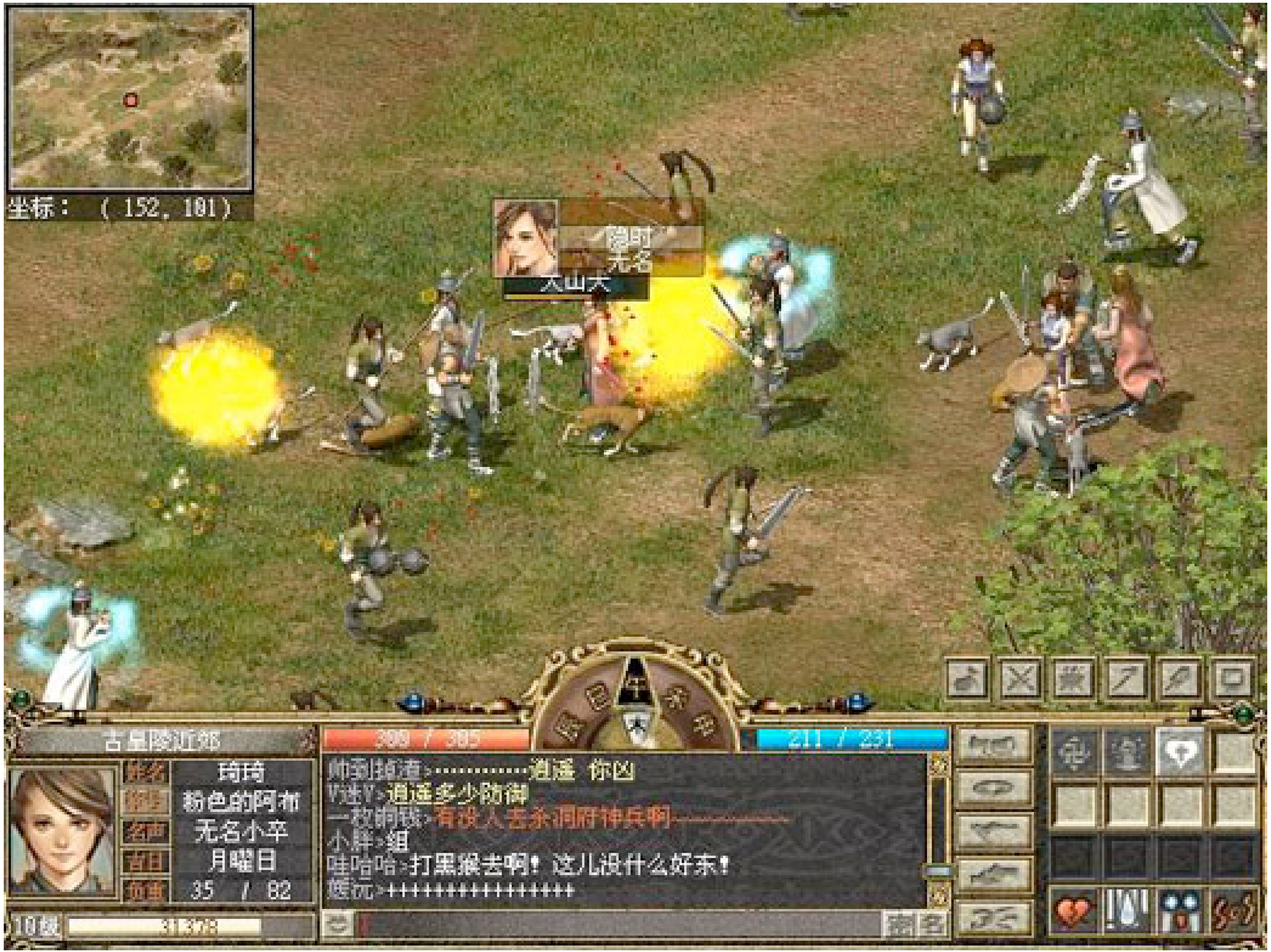
About **4** packets / sec

Average Payload:
100 Bytes

Loss Rate 1%

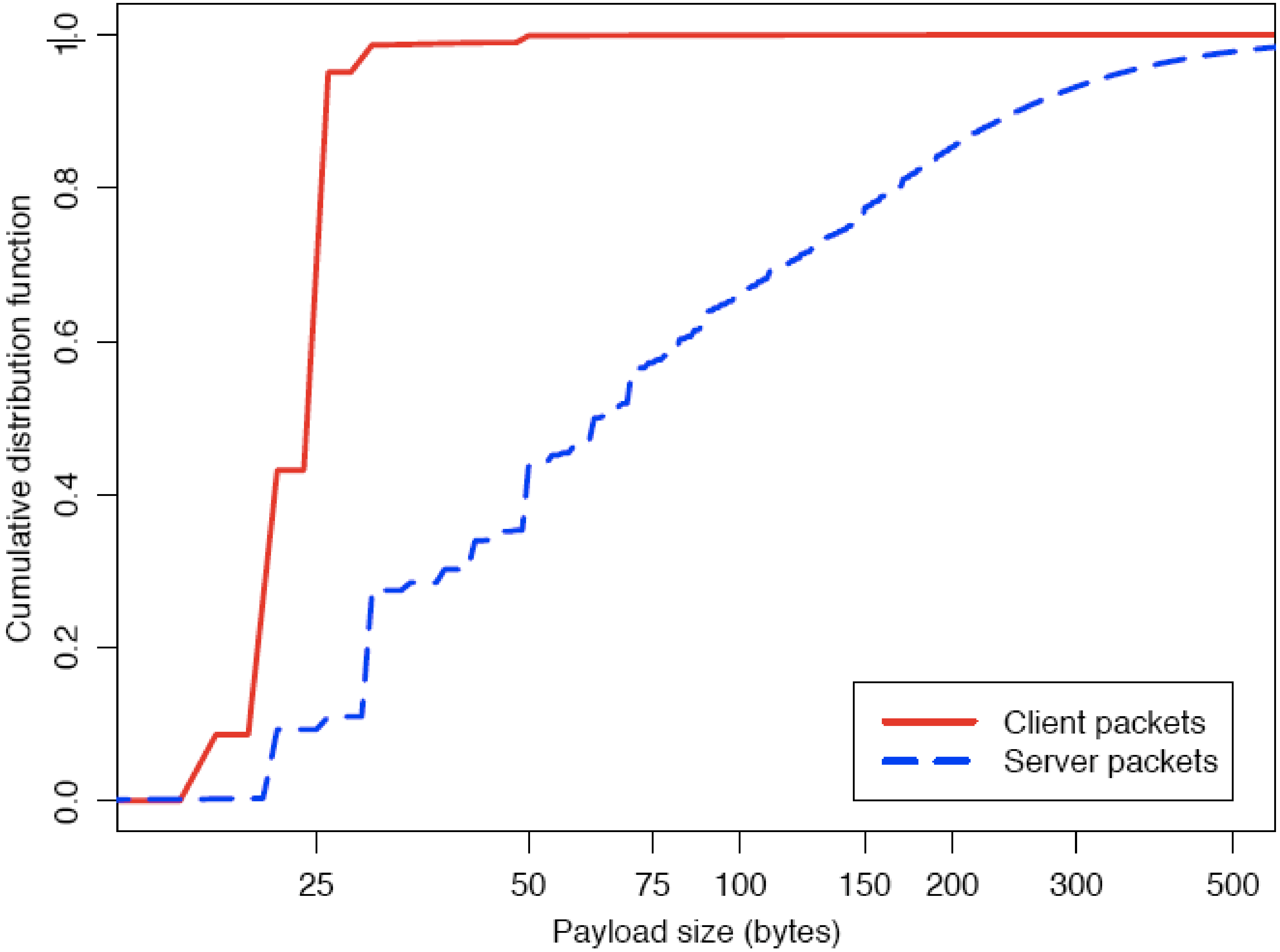
But some experience **6** retransmissions

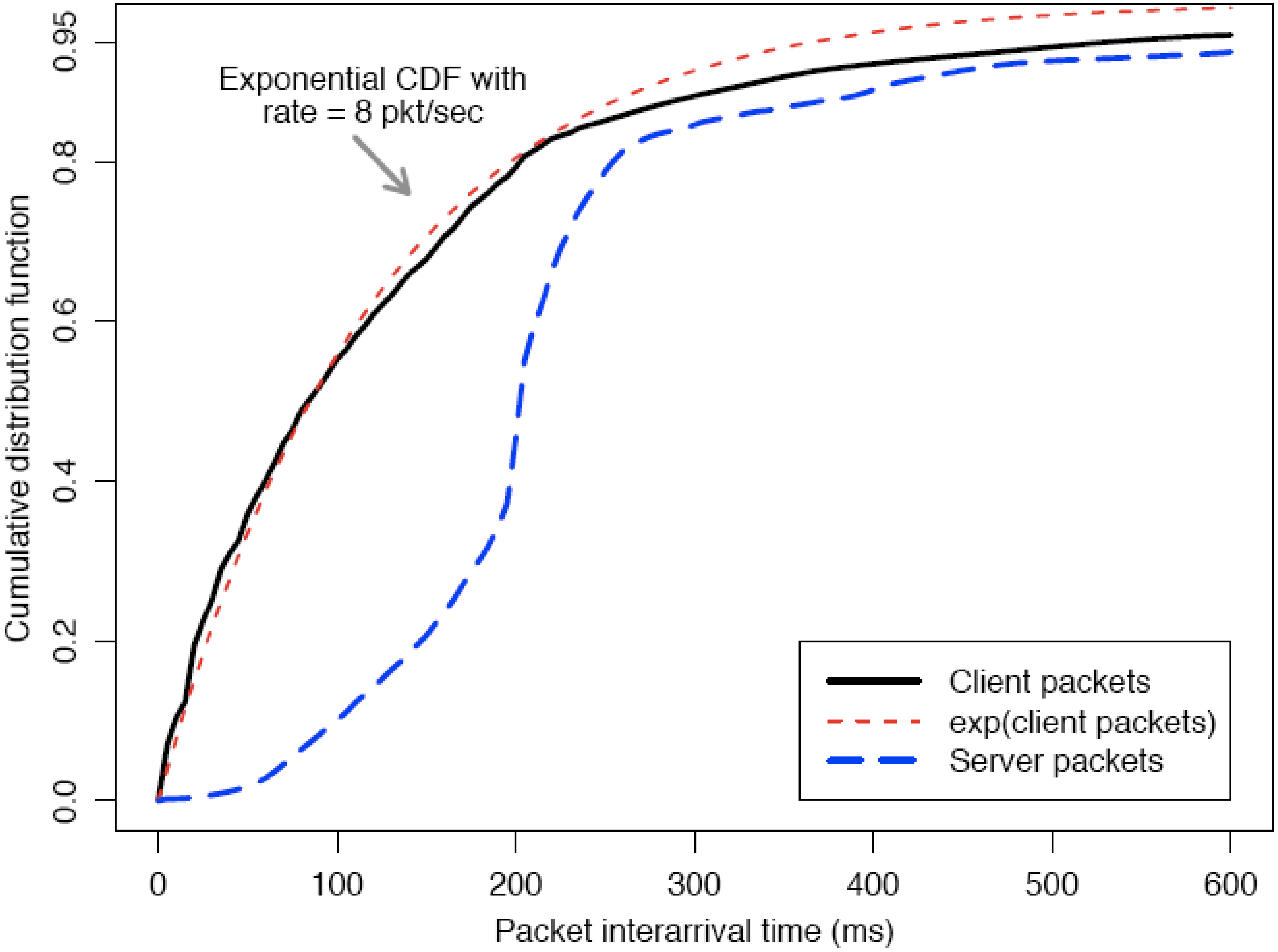
ShenZhou Online

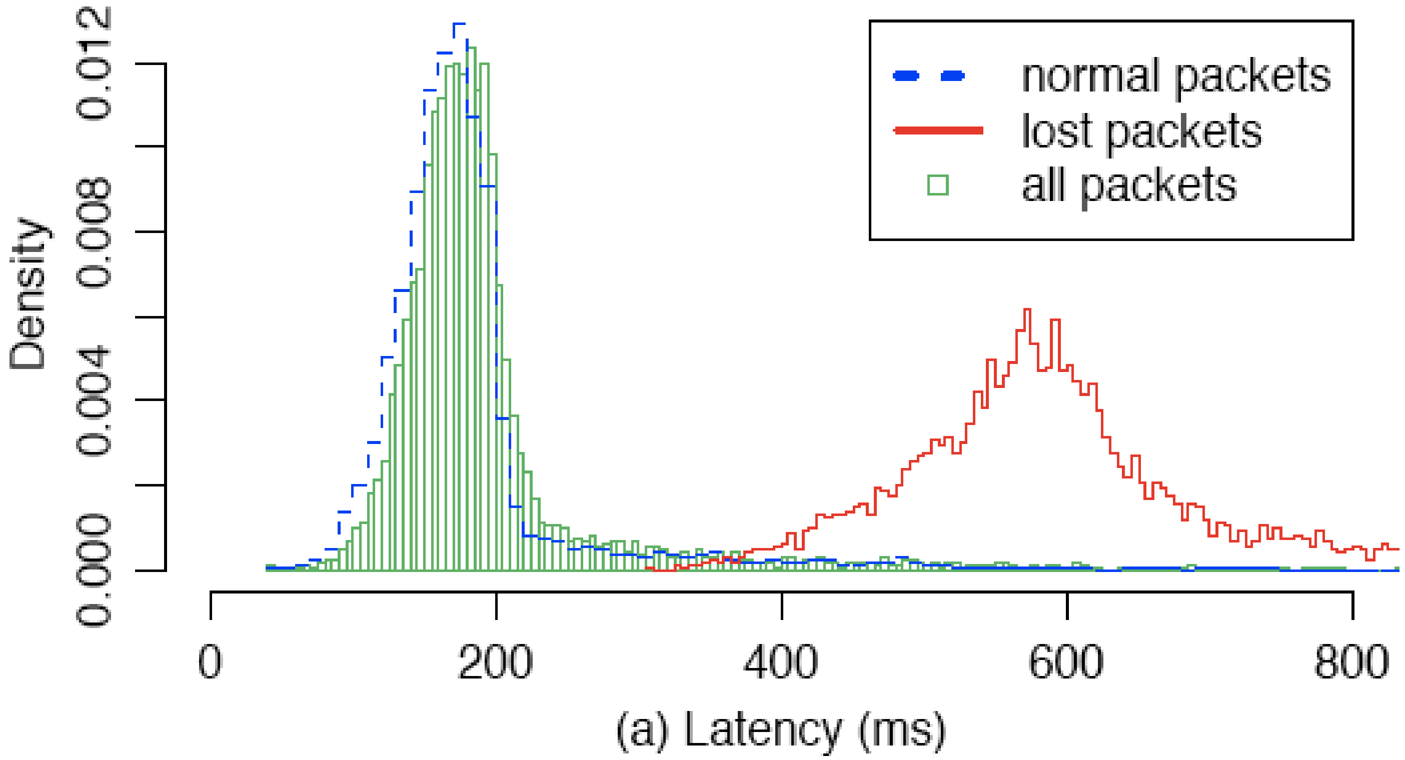


姓名	琦琦
称号	粉色的阿布
名声	无名小卒
吉日	月曜日
负重	35 / 82

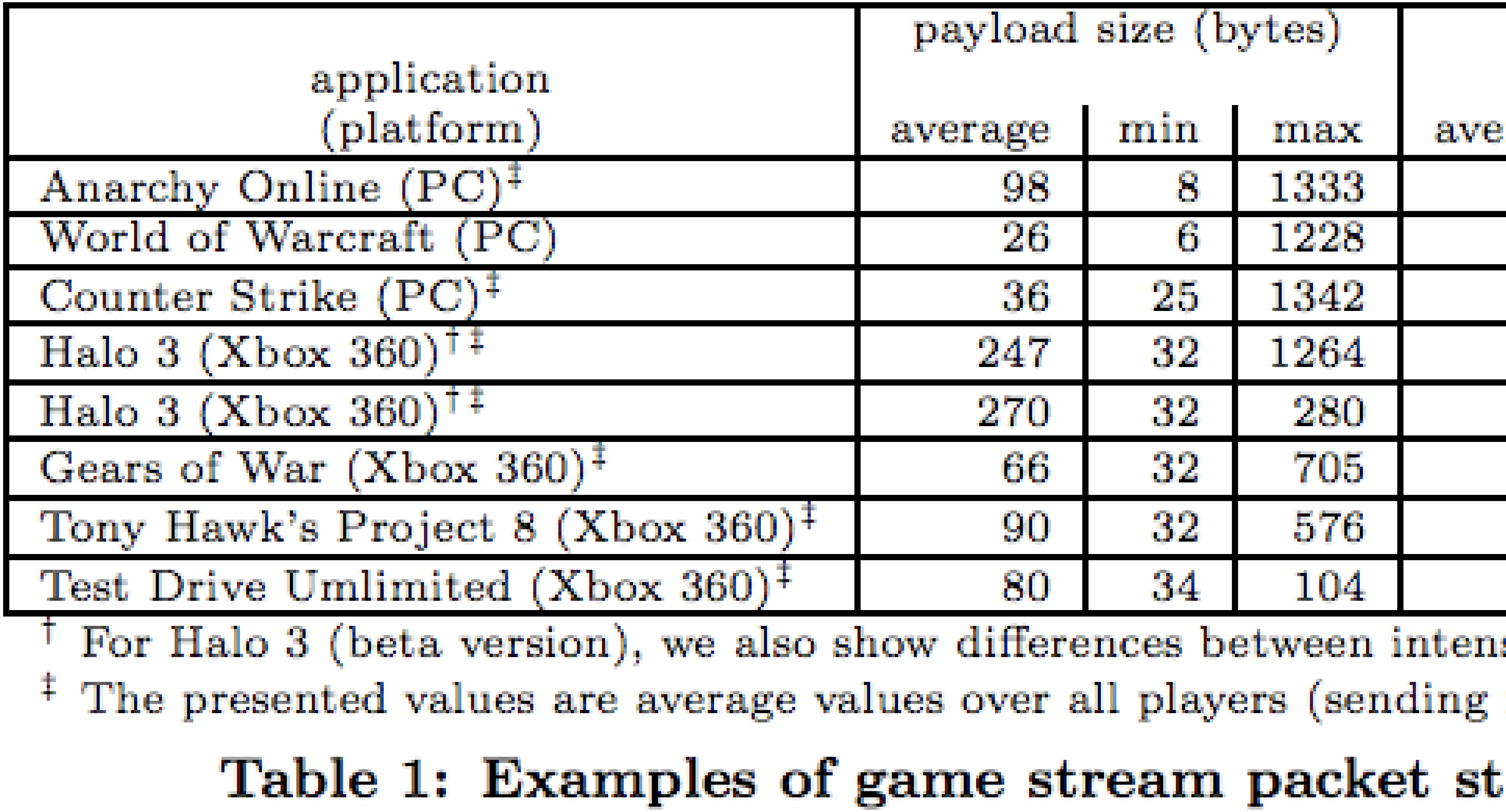
帅到掉渣>…………逍遥 你凶
V迷V>逍遥多少防御
一枚铜钱>有没人去杀洞府神兵啊————
小胖>组
哇哈哈>打黑猴去啊！这儿没什么好东！
娃玩>+++++++++++++++


Cumulative distribution function
0.0
0.2
0.4
0.6
0.8
1.0
25
50
75
100
150
200
300
500
Payload size (bytes)
Client packets
Server packets


Exponential CDF with
rate = 8 pkt/sec
Cumulative distribution function
0.0
0.2
0.4
0.6
0.8
0.95
0
100
200
300
400
500
600
Packet interarrival time (ms)
Client packets
exp(client packets)
Server packets


Density
0.000
0.004
0.008
0.012
0
200
400
600
800
(a) Latency (ms)
normal packets
lost packets
all packets

Similar stats for other games

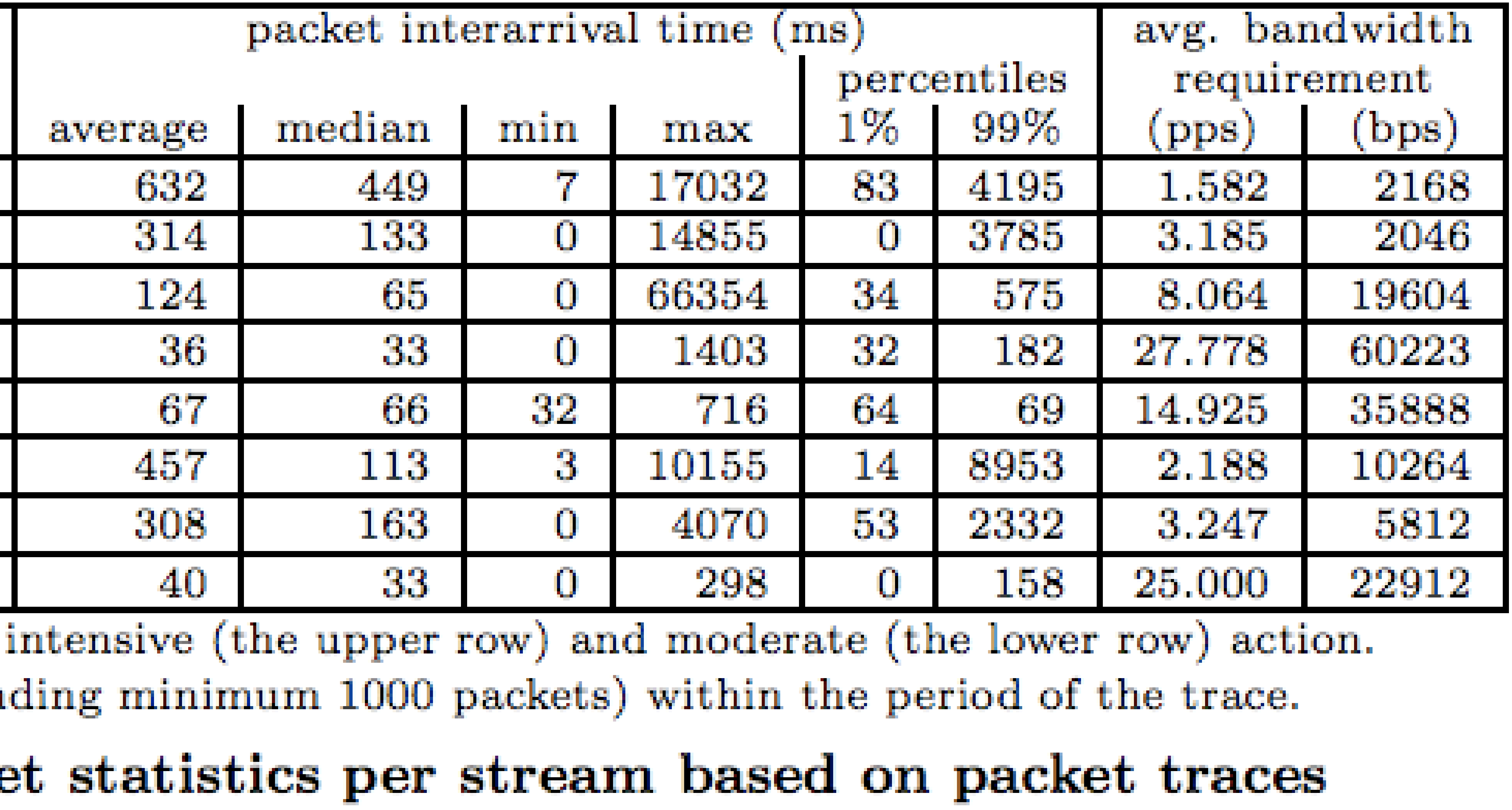


application (platform)	payload size (bytes)			
	average	min	max	ave
Anarchy Online (PC)‡	98	8	1333	
World of Warcraft (PC)	26	6	1228	
Counter Strike (PC)‡	36	25	1342	
Halo 3 (Xbox 360)†‡	247	32	1264	
Halo 3 (Xbox 360)†‡	270	32	280	
Gears of War (Xbox 360)‡	66	32	705	
Tony Hawk's Project 8 (Xbox 360)‡	90	32	576	
Test Drive Umlimited (Xbox 360)‡	80	34	104	

† For Halo 3 (beta version), we also show differences between intens

‡ The presented values are average values over all players (sending

Table 1: Examples of game stream packet st



packet interarrival time (ms)						avg. bandwidth	
				percentiles		requirement	
average	median	min	max	1%	99%	(pps)	(bps)
632	449	7	17032	83	4195	1.582	2168
314	133	0	14855	0	3785	3.185	2046
124	65	0	66354	34	575	8.064	19604
36	33	0	1403	32	182	27.778	60223
67	66	32	716	64	69	14.925	35888
457	113	3	10155	14	8953	2.188	10264
308	163	0	4070	53	2332	3.247	5812
40	33	0	298	0	158	25.000	22912

intensive (the upper row) and moderate (the lower row) action.
ding minimum 1000 packets) within the period of the trace.

et statistics per stream based on packet traces

“Thin Streams”

Findings 1:

Fast retransmission rarely triggered

In ShenZhou Online traces, fail
to trigger fast retransmission
because
insufficient dup ACK (50%)
interrupted by data (50%)

Findings 2:
Delay due mostly to timeout

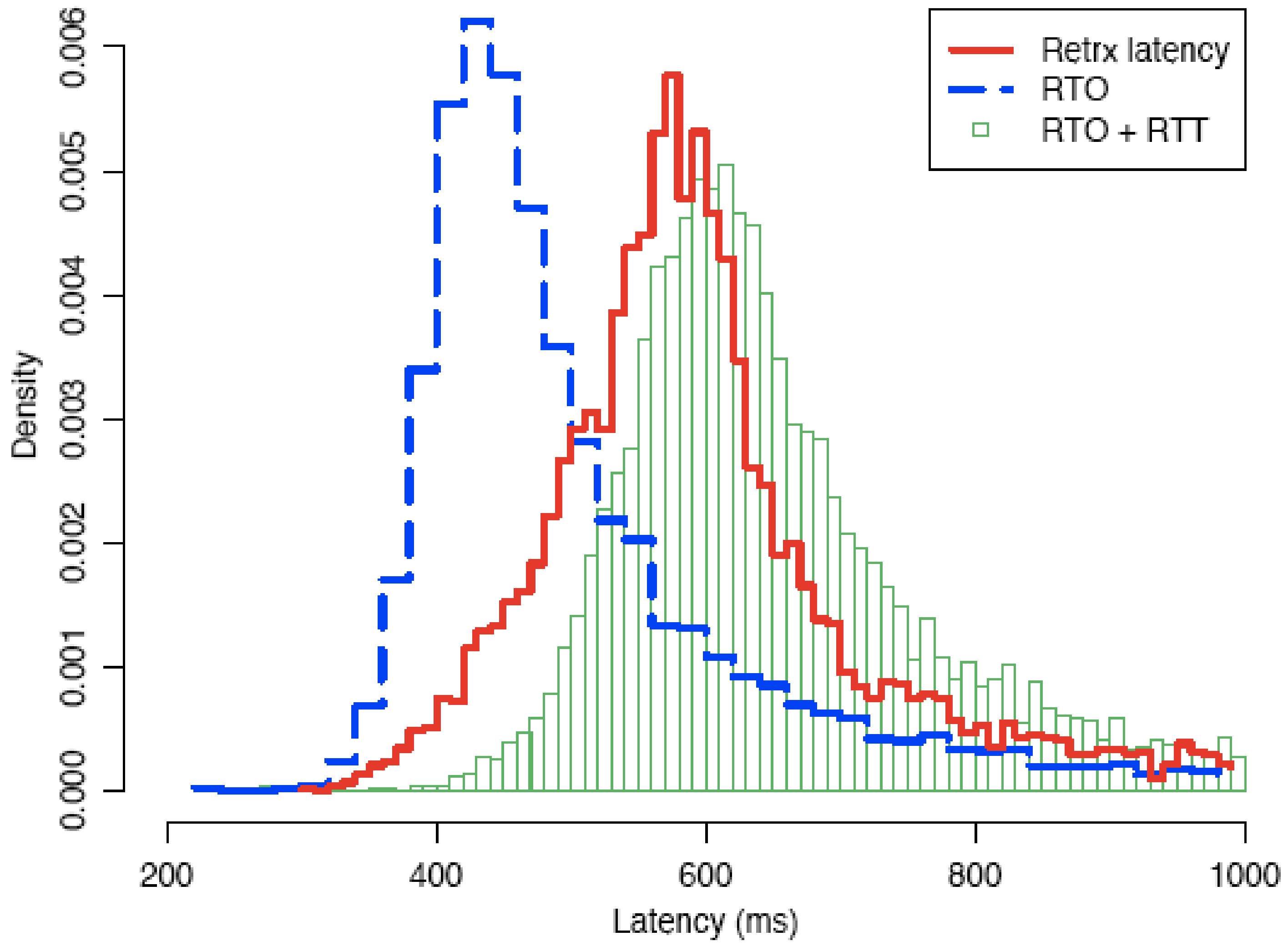


Figure 9: Average latency of dropped packets

Findings 3: Congestion Window reset is frequent

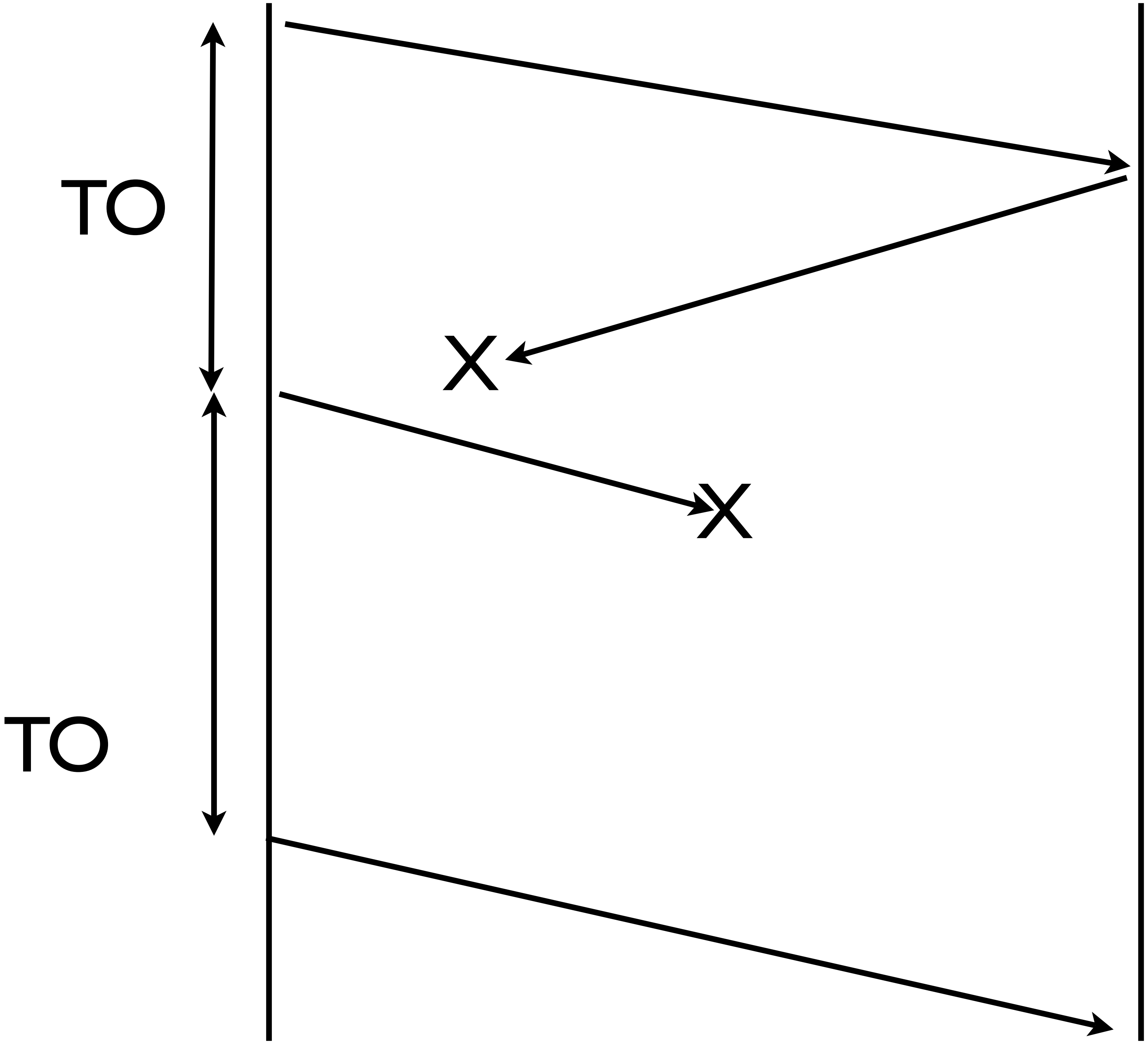
12% - 18% of packets faces window reset

think..
think..
think..
click (tank attack here) →
click (missle launch there) →
click (charge soldiers) →

The last command is delayed as congestion window = 2

How to make TCP (or, transport protocol) go faster in these games?

1. Remove exponential backoff



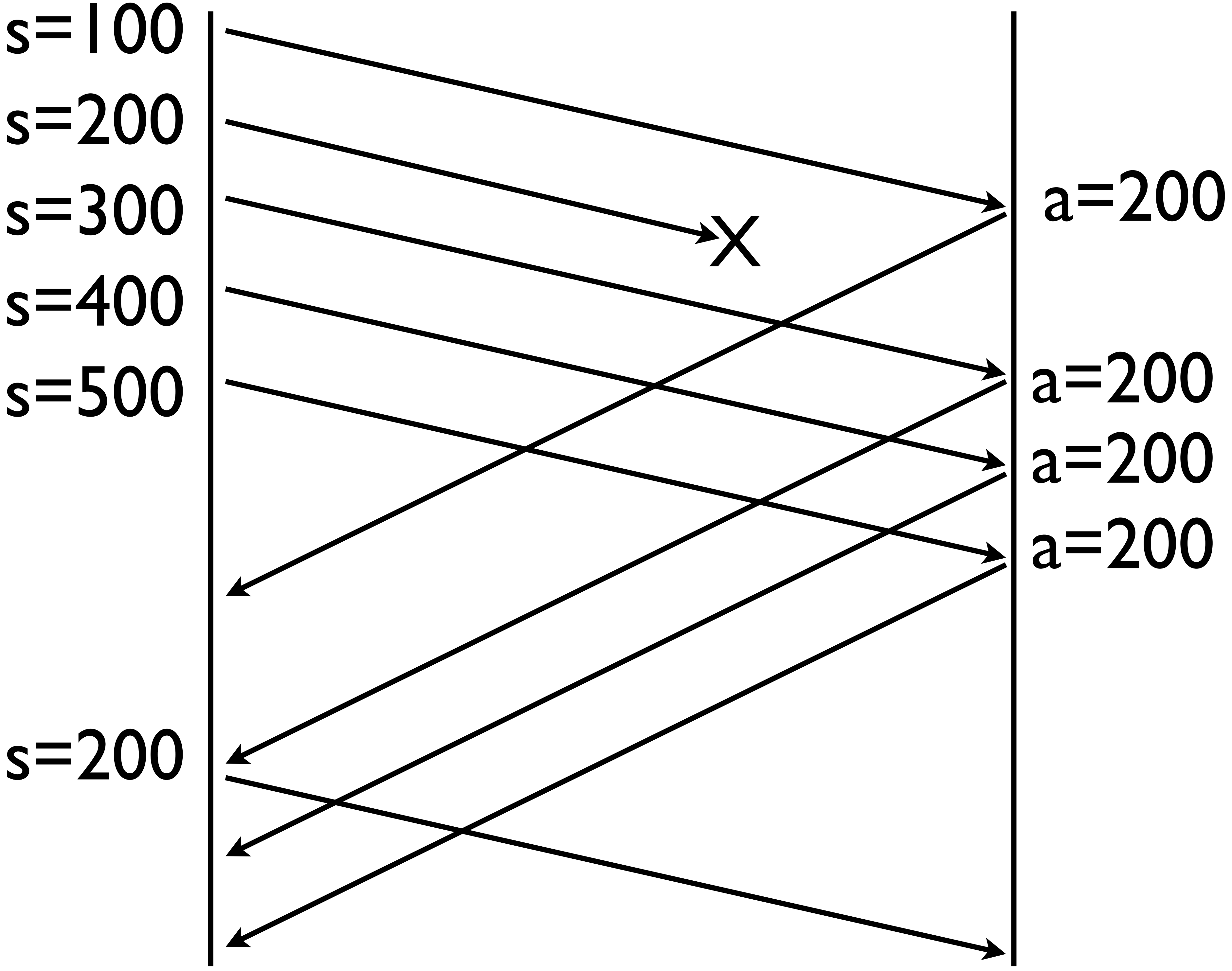
TCP Timeout

2. Make RTO Smaller

make sure minimum RTO is not 1s

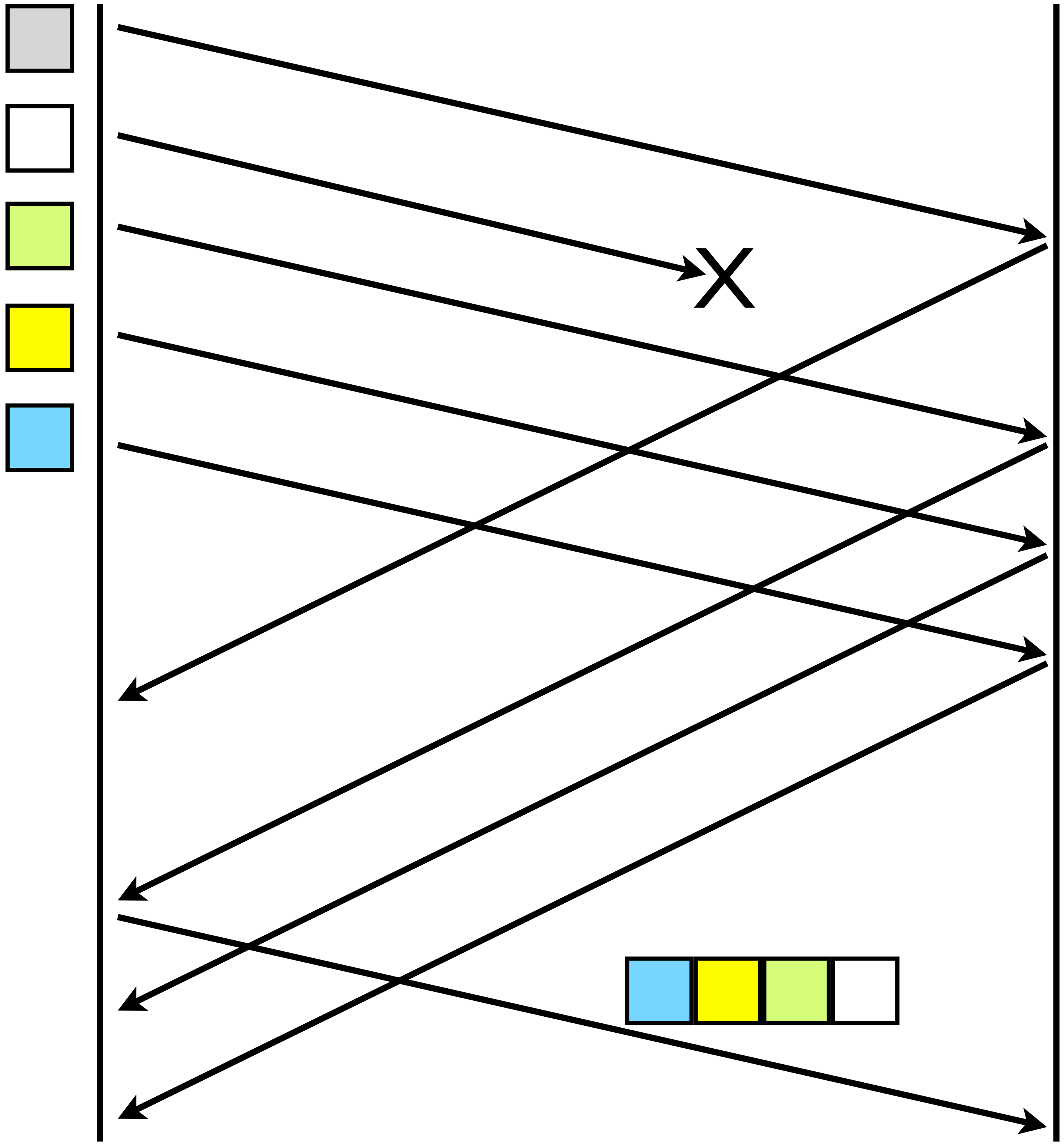
spurious retransmission
is not disastrous

3. Make Fast Retransmit Faster



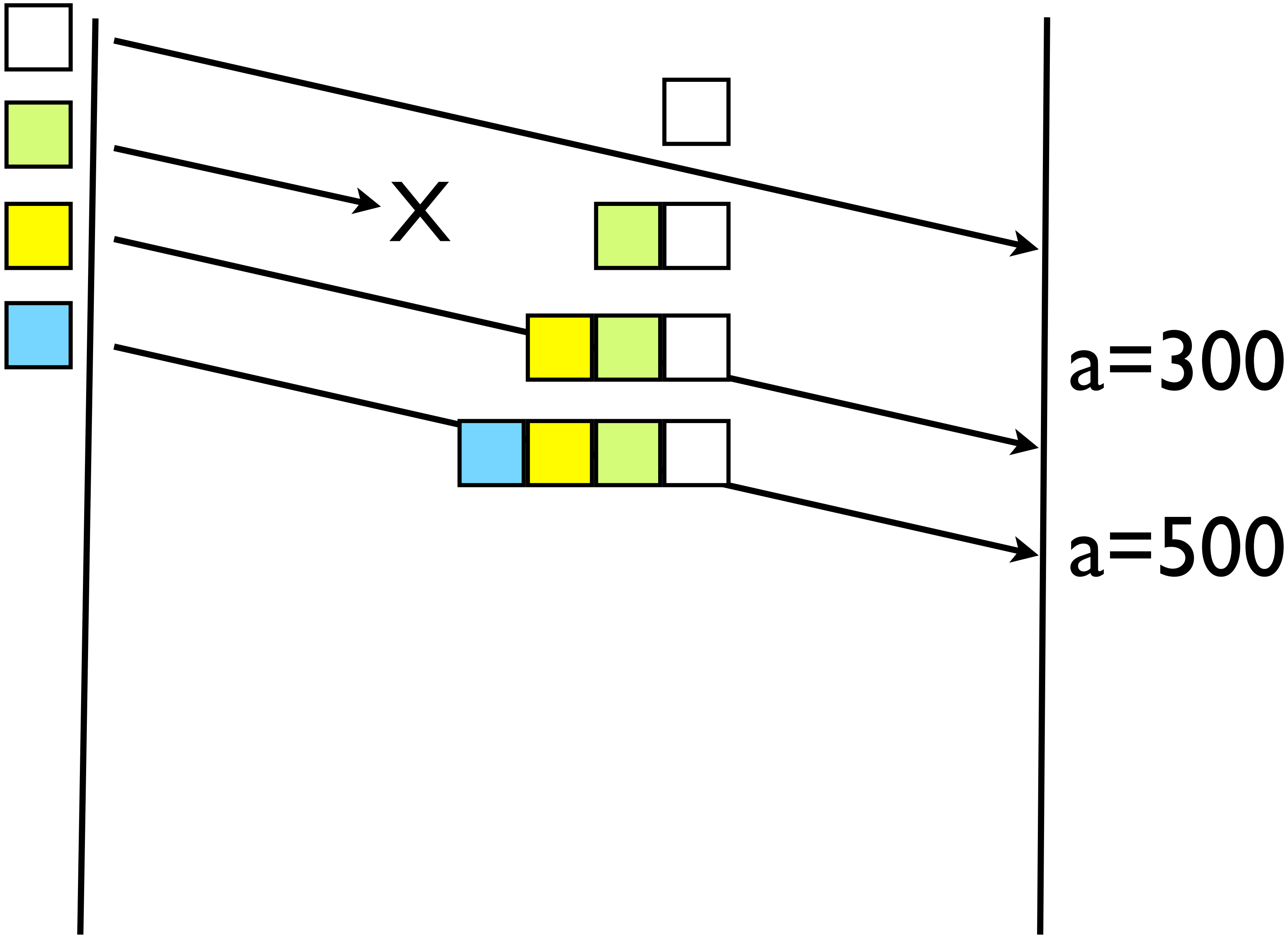
Retransmit after one duplicate ACK

4. Retransmission Bundling



Retransmit all unacknowledge data in queue

5. Redundant Data Bundling



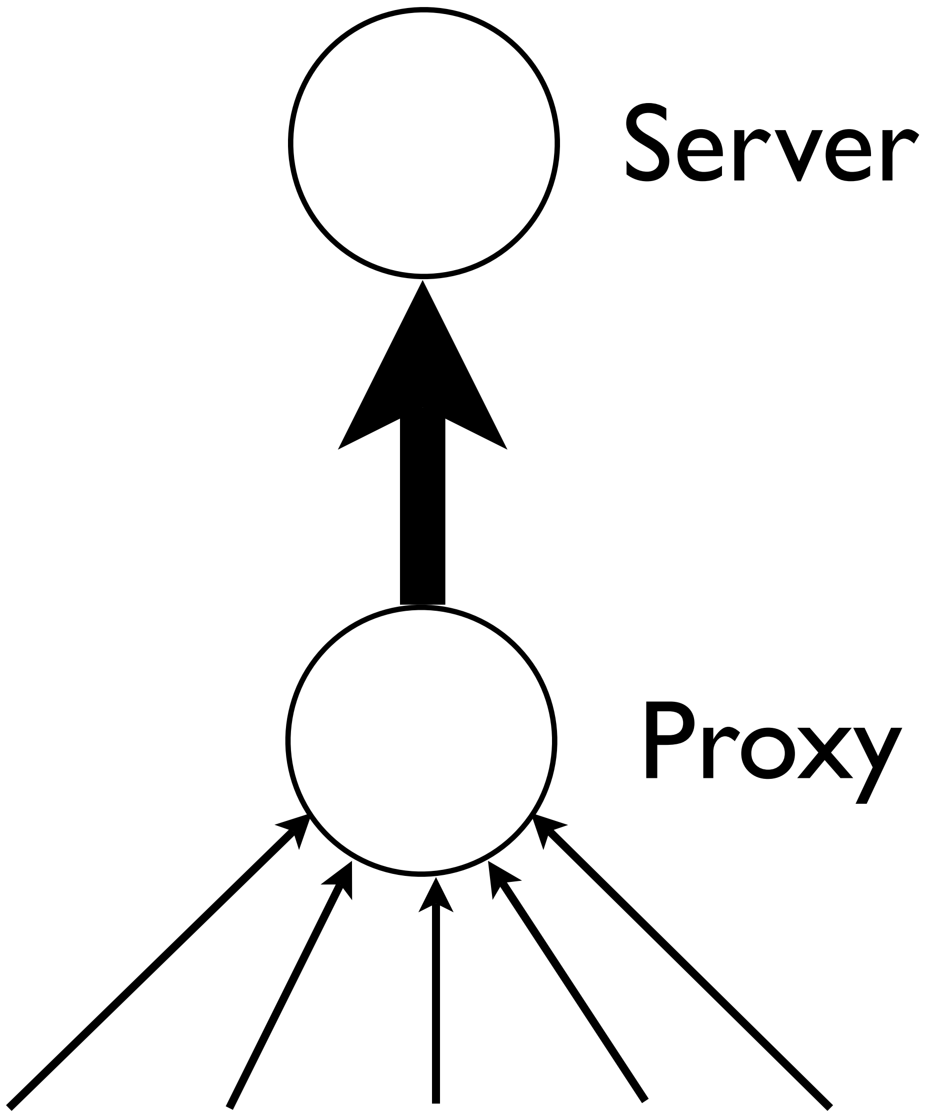
Send any unacknowledged segment in queue as long as there is space. Lost data gets recovered in the next transmission before retransmission.

6. Turn off or reduce Delayed ACKs

Packet interarrival time on average > 200ms

(can’t combine two ACKs into one)

7. Combine Thin Streams into Thicker Stream


Server
Proxy

Transport for Games

Transport for Games

- remove exponential backoff

Transport for Games

- remove exponential backoff
- reduce RTO

Transport for Games

- remove exponential backoff
- reduce RTO
- make fast retransmit faster

Transport for Games

- remove exponential backoff
- reduce RTO
- make fast retransmit faster
- retransmit agressively

Transport for Games

- remove exponential backoff
- reduce RTO
- make fast retransmit faster
- retransmit agressively
- don’t delay ACK

Transport for Games

- remove exponential backoff
- reduce RTO
- make fast retransmit faster
- retransmit agressively
- don’t delay ACK
- combine into thick streams