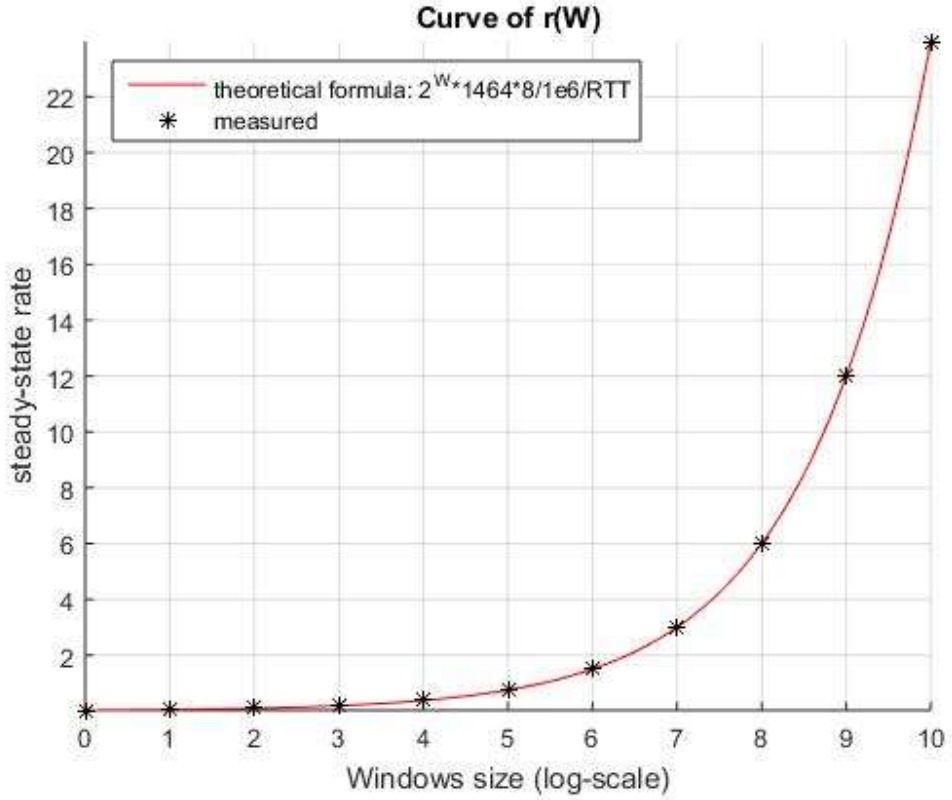
Report Homework 3
Mian Qin (UIN:725006574)

1. Discuss model of r(W) with different window size

The model of r(W) or the theoretical formula of r(W) is 2^W*1464*8/1e6/RTT (here r(W) unit is Mbps, RTT unit is second). This is because we transmit all the window successfully after a round trip time (RTT), so the goodput should be that formula.

Here I ran different W under the same RTT 0.5 second and collect the stable rates. And also drew the theoretical formula. The two perfectly match.

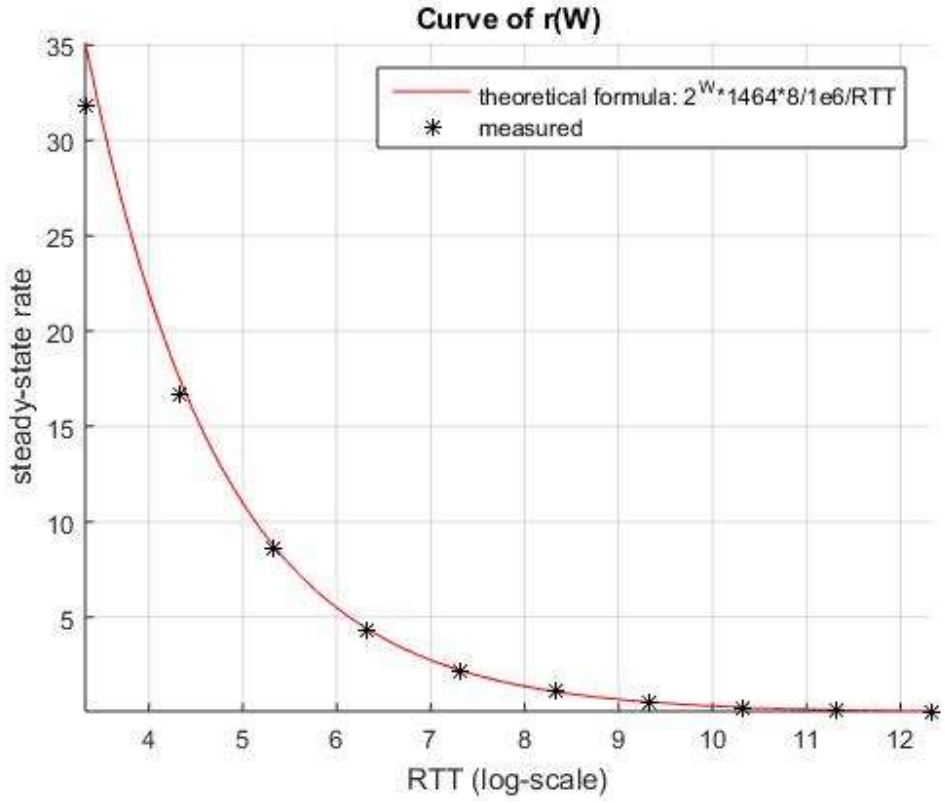


2. Discuss model of r(W) with different RTT.

Similar as the above discussion, the theoretical formula of r(W) is 2^W*1464*8/1e6/RTT (here r(W) unit is Mbps, RTT unit is second).

I ran different RTT under the same window size and collect the rates, here the rates is average rates (calculated at the end of program). I ran enough long time (around 5 minutes) because with the RTT increase to near the print out interval (2 second) the 2 second print out cannot show stable rates. Below is the plot of sthe theoretical formula. The two perfectly match.

From the figure, we can see some error with the measured value and theoretical value, that is because of the queuing/transmission delays emulated by the server and various OS kernel overhead.



3. Run with the dummy receiver.

CPU configuration:
Processor: AMD opteron(tm) processor 6172 2.10Ghz (2 core)
RAM: 32GB
Operating System: Windows 64bit

I ran three packet size configurations. For 1500 bytes packet size (default), I can achieve mostly 424Mbps throughput. After I increase the packet size 5 times (7500 bytes), I can achieve around 2Gbps. At last, I increase the packet size to 15000 bytes (10 times as default), I can get at most 3Gbps. If I increase the packet size further, there won't be too much throughput enhancement.

Interesting thing is, when I increase the packet size to 15000 (10 times as default), there will be some of WSAEWOULDBLOCK error happen for sendto function, but the program will exit normally, only cost some of the select wait time for sendto. After I changed the winsock kernel buffer from 20MB to 200MB. This blocking disappeared and the throughput got slightly improved.

Here are all the traces:

Packet size 1500 bytes:
Main: sender W = 8192, RTT 0.100 sec, loss 0 / 0, link 1000 Mbps
Main: initializing DWORD array with 2^29 elements... done in 1936 ms
Main: connected to 127.0.0.1 in 0.002 sec, pkt size 1472 bytes

[2] B 47895 (67.6 MB) N 47899 T 0 F 0 W 8192 S 280.497 Mbps RTT 0.000
[4] B 101990 (144.0 MB) N 102001 T 0 F 0 W 8192 S 316.587 Mbps RTT 0.000
[6] B 147989 (208.9 MB) N 147992 T 0 F 0 W 8192 S 268.960 Mbps RTT 0.000
[8] B 202060 (285.2 MB) N 202063 T 0 F 0 W 8192 S 316.435 Mbps RTT 0.000
[10] B 263773 (372.3 MB) N 263778 T 0 F 0 W 8192 S 361.210 Mbps RTT 0.000
[12] B 305854 (431.7 MB) N 305855 T 0 F 0 W 8192 S 246.204 Mbps RTT 0.000
[14] B 349315 (493.0 MB) N 349316 T 0 F 0 W 8192 S 254.309 Mbps RTT 0.000
[16] B 413828 (584.1 MB) N 414712 T 0 F 0 W 8192 S 382.736 Mbps RTT 0.012
[18] B 487360 (687.9 MB) N 487361 T 0 F 0 W 8192 S 424.952 Mbps RTT 0.001
[20] B 532133 (751.1 MB) N 532135 T 0 F 0 W 8192 S 261.939 Mbps RTT 0.001
[22] B 592226 (835.9 MB) N 592236 T 0 F 0 W 8192 S 351.735 Mbps RTT 0.001
[24] B 662758 (935.4 MB) N 662769 T 0 F 0 W 8192 S 412.737 Mbps RTT 0.001
[26] B 722031 (1019.1 MB) N 722035 T 0 F 0 W 8192 S 346.722 Mbps RTT 0.000
[28] B 767237 (1082.9 MB) N 767239 T 0 F 0 W 8192 S 264.445 Mbps RTT 0.000
[30] B 814684 (1149.9 MB) N 814688 T 0 F 0 W 8192 S 277.615 Mbps RTT 0.000
[32] B 864743 (1220.5 MB) N 864745 T 0 F 0 W 8192 S 292.753 Mbps RTT 0.000
[34] B 913022 (1288.7 MB) N 913026 T 0 F 0 W 8192 S 282.687 Mbps RTT 0.000
[36] B 964776 (1361.7 MB) N 964780 T 0 F 0 W 8192 S 303.001 Mbps RTT 0.000
[38] B 1022123 (1442.7 MB) N 1022127 T 0 F 0 W 8192 S 335.783 Mbps RTT 0.000
[40] B 1069238 (1509.2 MB) N 1069239 T 0 F 0 W 8192 S 275.847 Mbps RTT 0.000
[42] B 1110958 (1568.0 MB) N 1110959 T 0 F 0 W 8192 S 244.277 Mbps RTT 0.001
[44] B 1164838 (1644.1 MB) N 1164842 T 0 F 0 W 8192 S 315.498 Mbps RTT 0.000
[46] B 1214344 (1714.0 MB) N 1214348 T 0 F 0 W 8192 S 289.849 Mbps RTT 0.000
[48] B 1269329 (1791.6 MB) N 1269330 T 0 F 0 W 8192 S 321.939 Mbps RTT 0.000
[50] B 1311699 (1851.4 MB) N 1311700 T 0 F 0 W 8192 S 248.078 Mbps RTT 0.000
[52] B 1371679 (1936.0 MB) N 1371797 T 0 F 0 W 8192 S 351.893 Mbps RTT 0.001
[54] B 1453938 (2052.1 MB) N 1453945 T 0 F 0 W 8192 S 480.965 Mbps RTT 0.000
[54.302] <-- FIN-ACK 1466861 window 0
Main: transfer finished in 54.285 sec, 316475.47 Kbps, checksum: ef5f9797
Main: estRTT 0.000, ideal rate inf Kbps

Packet size 7500 bytes:
Main: sender W = 8192, RTT 0.100 sec, loss 0 / 0, link 1000 Mbps
Main: initializing DWORD array with 2^30 elements... done in 4774 ms
Main: connected to 127.0.0.1 in 0.001 sec, pkt size 7472 bytes
[2] B 53958 (384.9 MB) N 61140 T 4 F 0 W 8192 S 1825.396 Mbps RTT 0.253
[4] B 125153 (892.8 MB) N 127602 T 7 F 0 W 8192 S 1983.961 Mbps RTT 0.065
[6] B 182527 (1302.1 MB) N 189456 T 12 F 0 W 8192 S 1846.146 Mbps RTT 0.223
[8] B 242295 (1728.4 MB) N 249883 T 18 F 0 W 8192 S 1803.720 Mbps RTT 0.223
[10] B 296377 (2114.2 MB) N 304569 T 23 F 0 W 8192 S 1632.287 Mbps RTT 0.285
[12] B 366055 (2611.2 MB) N 369563 T 28 F 0 W 8192 S 1940.461 Mbps RTT 0.094
[14] B 426538 (3042.7 MB) N 434730 T 30 F 0 W 8192 S 1942.909 Mbps RTT 0.254
[16] B 488912 (3487.6 MB) N 496331 T 36 F 0 W 8192 S 1839.159 Mbps RTT 0.254
[18] B 549268 (3918.2 MB) N 557460 T 38 F 0 W 8192 S 1823.545 Mbps RTT 0.244
[18.801] <-- FIN-ACK 575425 window 0
Main: transfer finished in 18.786 sec, 1829007.88 Kbps, checksum: 39f2a0e6

Main: estRTT 0.138, ideal rate 3544643.000 Kbps

Packet size 15000 bytes:
Main: sender W = 8192, RTT 0.100 sec, loss 0 / 0, link 1000 Mbps
Main: initializing DWORD array with 2^30 elements... done in 3410 ms
Main: connected to 127.0.0.1 in 0.002 sec, pkt size 14972 bytes
[2] B 46323 (661.8 MB) N 50828 T 0 F 0 W 8192 S 3042.361 Mbps RTT 0.142
[4] B 90641 (1294.9 MB) N 98513 T 4 F 0 W 8192 S 2851.360 Mbps RTT 0.385
[6] B 126706 (1810.1 MB) N 134898 T 16 F 14 W 8192 S 2174.329 Mbps RTT 0.227
[8] B 167459 (2392.3 MB) N 174234 T 747 F 82 W 8192 S 2354.496 Mbps RTT 0.265
[10] B 191881 (2741.2 MB) N 200072 T 3490 F 248 W 8192 S 1542.549 Mbps RTT 0.353
[12] B 233797 (3340.0 MB) N 237046 T 5155 F 361 W 8192 S 2212.637 Mbps RTT 0.105
[14] B 273855 (3912.3 MB) N 281992 T 6926 F 545 W 8192 S 2687.954 Mbps RTT 0.162
[14.611] <-- FIN-ACK 287021 window 0
Main: transfer finished in 14.598 sec, 2353729.25 Kbps, checksum: 39f2a0e6
Main: estRTT 0.000, ideal rate inf Kbps

4. Compare with loss scenario.

The two scenarios have the same rate (or throughput), both around 10Mbps and that's the link capacity 10Mbps in the input arguments. The reason is the reverse path loss only loss some ACK packets, however, in our implement protocol, we don't need to get all the ACK packet. As long as the ACK packet with the sequence number greater than the send base. We assume all the packets sent from send base to the ACK sequence is received correctly. So losing some ACK packet won't lead to sender retransmission (as the below trace shows), so the performance should be the same as no loss.

Here the link capacity parameter is the bottle neck, it will lead to some queueing delay, that's why the estimate RTT is greater than the RTT parameter we set (as 0.2 second).

Here are all the traces:

0.1 reverse path loss:
Main: sender W = 300, RTT 0.200 sec, loss 0 / 0.1, link 10 Mbps
Main: initializing DWORD array with 2^23 elements... done in 48 ms
Main: connected to s8.irl.cs.tamu.edu in 0.202 sec, pkt size 1472 bytes
[2] B 224 (0.3 MB) N 448 T 1 F 0 W 224 S 2.623 Mbps RTT 0.259
[4] B 1905 (2.7 MB) N 2203 T 1 F 0 W 300 S 10.277 Mbps RTT 0.347
[6] B 3603 (5.1 MB) N 3903 T 1 F 0 W 300 S 9.943 Mbps RTT 0.347
[8] B 5301 (7.5 MB) N 5601 T 1 F 0 W 300 S 9.943 Mbps RTT 0.347
[10] B 6999 (9.9 MB) N 7299 T 1 F 0 W 300 S 9.943 Mbps RTT 0.347
[12] B 8697 (12.3 MB) N 8997 T 1 F 0 W 300 S 9.943 Mbps RTT 0.347
[14] B 10396 (14.7 MB) N 10695 T 1 F 0 W 300 S 9.943 Mbps RTT 0.347
[16] B 12094 (17.1 MB) N 12394 T 1 F 0 W 300 S 9.943 Mbps RTT 0.347
[18] B 13793 (19.5 MB) N 14093 T 1 F 0 W 300 S 9.949 Mbps RTT 0.347
[20] B 15493 (21.9 MB) N 15793 T 1 F 0 W 300 S 9.955 Mbps RTT 0.347

[22] B 17193 (24.3 MB) N 17492 T 1 F 0 W 300 S 9.949 Mbps RTT 0.347
[24] B 18892 (26.7 MB) N 19192 T 1 F 0 W 300 S 9.949 Mbps RTT 0.347
[26] B 20592 (29.1 MB) N 20892 T 1 F 0 W 300 S 9.955 Mbps RTT 0.347
[28] B 22291 (31.5 MB) N 22591 T 1 F 0 W 300 S 9.949 Mbps RTT 0.347
[29.180] <-- FIN-ACK 22920 window d70096ab
Main: transfer finished in 28.738 sec, 9340.78 Kbps, checksum: d70096ab
Main: estRTT 0.347, ideal rate 10125.648 Kbps

no loss:
Main: sender W = 300, RTT 0.200 sec, loss 0 / 0, link 10 Mbps
Main: initializing DWORD array with 2^23 elements... done in 47 ms
Main: connected to s8.irl.cs.tamu.edu in 0.202 sec, pkt size 1472 bytes
[2] B 256 (0.4 MB) N 512 T 0 F 0 W 256 S 2.998 Mbps RTT 0.270
[4] B 1939 (2.7 MB) N 2239 T 0 F 0 W 300 S 10.113 Mbps RTT 0.347
[6] B 3638 (5.1 MB) N 3938 T 0 F 0 W 300 S 9.938 Mbps RTT 0.347
[8] B 5337 (7.5 MB) N 5637 T 0 F 0 W 300 S 9.938 Mbps RTT 0.347
[10] B 7036 (9.9 MB) N 7336 T 0 F 0 W 300 S 9.943 Mbps RTT 0.347
[12] B 8735 (12.3 MB) N 9035 T 0 F 0 W 300 S 9.938 Mbps RTT 0.347
[14] B 10434 (14.7 MB) N 10734 T 0 F 0 W 300 S 9.938 Mbps RTT 0.347
[16] B 12134 (17.1 MB) N 12434 T 0 F 0 W 300 S 9.943 Mbps RTT 0.347
[18] B 13834 (19.5 MB) N 14134 T 0 F 0 W 300 S 9.949 Mbps RTT 0.347
[20] B 15534 (21.9 MB) N 15834 T 0 F 0 W 300 S 9.949 Mbps RTT 0.347
[22] B 17235 (24.3 MB) N 17535 T 0 F 0 W 300 S 9.949 Mbps RTT 0.347
[24] B 18935 (26.7 MB) N 19235 T 0 F 0 W 300 S 9.949 Mbps RTT 0.347
[26] B 20635 (29.1 MB) N 20935 T 0 F 0 W 300 S 9.943 Mbps RTT 0.347
[28] B 22336 (31.5 MB) N 22636 T 0 F 0 W 300 S 9.949 Mbps RTT 0.347
[29.139] <-- FIN-ACK 22920 window d70096ab
Main: transfer finished in 28.700 sec, 9353.15 Kbps, checksum: d70096ab
Main: estRTT 0.347, ideal rate 10125.648 Kbps

5 Receiver window change algorithm.

In my program, I have a macro of DEBUG_ON that can print out every packet transaction. After I turned it on, I can see how the receiver changes its advertised window size, that is, plus one on every ACK packet.
It is flow control technique in TCP stack.

Here are my debug traces:
Main: sender W = 10, RTT 0.100 sec, loss 0 / 0, link 1000 Mbps
Main: initializing DWORD array with 2^14 elements... done in 0 ms
[0.034] --> SYN 0 (attempt 1 of 3, RTO 1.000) to 128.194.135.82
[0.138] <-- SYN-ACK 0 window 1; setting initial RTO to 0.142
Main: connected to s8.irl.cs.tamu.edu in 0.102 sec, pkt size 1472 bytes
[0.002] --> pkt 0 ts 150 (attempt 1 of 50, RTO 0.142)
[0.103] <-- ACK 1 window 1, RTT 0.101, RTO 0.141)
[0.105] --> pkt 1 ts 253 (attempt 1 of 50, RTO 0.141)

[0.205] <-- ACK 2 window 2, RTT 0.100, RTO 0.140)
[0.207] --> pkt 2 ts 355 (attempt 1 of 50, RTO 0.140)
[0.209] --> pkt 3 ts 357 (attempt 1 of 50, RTO 0.140)
[0.307] <-- ACK 3 window 3, RTT 0.100, RTO 0.140)
[0.309] --> pkt 4 ts 457 (attempt 1 of 50, RTO 0.140)
[0.310] <-- ACK 4 window 4, RTT 0.100, RTO 0.140)
[0.312] --> pkt 5 ts 460 (attempt 1 of 50, RTO 0.140)
[0.313] --> pkt 6 ts 461 (attempt 1 of 50, RTO 0.140)
[0.314] --> pkt 7 ts 462 (attempt 1 of 50, RTO 0.140)
[0.410] <-- ACK 5 window 5, RTT 0.100, RTO 0.140)
[0.412] --> pkt 8 ts 560 (attempt 1 of 50, RTO 0.140)
[0.413] <-- ACK 6 window 6, RTT 0.100, RTO 0.140)
[0.414] <-- ACK 7 window 7, RTT 0.100, RTO 0.140)
[0.415] <-- ACK 8 window 8, RTT 0.100, RTO 0.140)
[0.416] --> pkt 9 ts 564 (attempt 1 of 50, RTO 0.140)
[0.418] --> pkt 10 ts 566 (attempt 1 of 50, RTO 0.140)
[0.419] --> pkt 11 ts 567 (attempt 1 of 50, RTO 0.140)
[0.420] --> pkt 12 ts 568 (attempt 1 of 50, RTO 0.140)
[0.421] --> pkt 13 ts 569 (attempt 1 of 50, RTO 0.140)
[0.422] --> pkt 14 ts 570 (attempt 1 of 50, RTO 0.140)
[0.424] --> pkt 15 ts 572 (attempt 1 of 50, RTO 0.140)
[0.512] <-- ACK 9 window 9, RTT 0.100, RTO 0.140)
[0.514] --> pkt 16 ts 662 (attempt 1 of 50, RTO 0.140)
[0.515] --> pkt 17 ts 663 (attempt 1 of 50, RTO 0.140)
[0.517] <-- ACK 10 window 10, RTT 0.100, RTO 0.140)
[0.518] <-- ACK 11 window 11, RTT 0.100, RTO 0.140)
[0.520] <-- ACK 12 window 12, RTT 0.100, RTO 0.140)
[0.521] <-- ACK 13 window 13, RTT 0.100, RTO 0.140)
[0.522] <-- ACK 14 window 14, RTT 0.100, RTO 0.140)
[0.524] <-- ACK 15 window 15, RTT 0.100, RTO 0.140)
[0.525] <-- ACK 16 window 16, RTT 0.100, RTO 0.140)
[0.526] --> pkt 18 ts 674 (attempt 1 of 50, RTO 0.140)
[0.527] --> pkt 19 ts 675 (attempt 1 of 50, RTO 0.140)
[0.529] --> pkt 20 ts 677 (attempt 1 of 50, RTO 0.140)
[0.531] --> pkt 21 ts 679 (attempt 1 of 50, RTO 0.140)
[0.533] --> pkt 22 ts 681 (attempt 1 of 50, RTO 0.140)
[0.536] --> pkt 23 ts 684 (attempt 1 of 50, RTO 0.140)
[0.537] --> pkt 24 ts 685 (attempt 1 of 50, RTO 0.140)
[0.538] --> pkt 25 ts 686 (attempt 1 of 50, RTO 0.140)
[0.614] <-- ACK 17 window 17, RTT 0.100, RTO 0.140)
[0.616] <-- ACK 18 window 18, RTT 0.100, RTO 0.140)
[0.617] --> pkt 26 ts 765 (attempt 1 of 50, RTO 0.140)
[0.618] --> pkt 27 ts 766 (attempt 1 of 50, RTO 0.140)
[0.626] <-- ACK 19 window 19, RTT 0.100, RTO 0.140)
[0.627] --> pkt 28 ts 775 (attempt 1 of 50, RTO 0.140)
[0.628] <-- ACK 20 window 20, RTT 0.100, RTO 0.140)

[0.629] --> pkt 29 ts 777 (attempt 1 of 50, RTO 0.140)
[0.630] <-- ACK 21 window 21, RTT 0.100, RTO 0.140)
[0.631] --> pkt 30 ts 779 (attempt 1 of 50, RTO 0.140)
[0.632] <-- ACK 22 window 22, RTT 0.100, RTO 0.140)
[0.633] --> pkt 31 ts 781 (attempt 1 of 50, RTO 0.140)
[0.634] <-- ACK 23 window 23, RTT 0.100, RTO 0.140)
[0.635] --> pkt 32 ts 783 (attempt 1 of 50, RTO 0.140)
[0.636] <-- ACK 24 window 24, RTT 0.100, RTO 0.140)
[0.637] --> pkt 33 ts 785 (attempt 1 of 50, RTO 0.140)
[0.638] <-- ACK 25 window 25, RTT 0.100, RTO 0.140)
[0.639] --> pkt 34 ts 787 (attempt 1 of 50, RTO 0.140)
[0.640] <-- ACK 26 window 26, RTT 0.100, RTO 0.140)
[0.641] --> pkt 35 ts 789 (attempt 1 of 50, RTO 0.140)
[0.717] <-- ACK 27 window 27, RTT 0.100, RTO 0.140)
[0.718] <-- ACK 28 window 28, RTT 0.100, RTO 0.140)
[0.719] --> pkt 36 ts 867 (attempt 1 of 50, RTO 0.140)
[0.720] --> pkt 37 ts 868 (attempt 1 of 50, RTO 0.140)
[0.728] <-- ACK 29 window 29, RTT 0.100, RTO 0.140)
[0.729] --> pkt 38 ts 877 (attempt 1 of 50, RTO 0.140)
[0.730] <-- ACK 30 window 30, RTT 0.100, RTO 0.140)
[0.731] --> pkt 39 ts 879 (attempt 1 of 50, RTO 0.140)
[0.732] <-- ACK 31 window 31, RTT 0.100, RTO 0.140)
[0.733] --> pkt 40 ts 881 (attempt 1 of 50, RTO 0.140)
[0.733] <-- ACK 32 window 32, RTT 0.100, RTO 0.140)
[0.734] --> pkt 41 ts 882 (attempt 1 of 50, RTO 0.140)
[0.736] <-- ACK 33 window 33, RTT 0.100, RTO 0.140)
[0.737] --> pkt 42 ts 885 (attempt 1 of 50, RTO 0.140)
[0.738] <-- ACK 34 window 34, RTT 0.100, RTO 0.140)
[0.740] <-- ACK 35 window 35, RTT 0.100, RTO 0.140)
[0.741] <-- ACK 36 window 36, RTT 0.100, RTO 0.140)
[0.743] --> pkt 43 ts 891 (attempt 1 of 50, RTO 0.140)
[0.744] --> pkt 44 ts 892 (attempt 1 of 50, RTO 0.140)
[0.819] <-- ACK 37 window 37, RTT 0.100, RTO 0.140)
[0.820] <-- ACK 38 window 38, RTT 0.100, RTO 0.140)
[0.830] <-- ACK 39 window 39, RTT 0.100, RTO 0.140)
[0.831] <-- ACK 40 window 40, RTT 0.100, RTO 0.140)
[0.833] <-- ACK 41 window 41, RTT 0.100, RTO 0.140)
[0.835] <-- ACK 42 window 42, RTT 0.100, RTO 0.140)
[0.838] <-- ACK 43 window 43, RTT 0.100, RTO 0.140)
[0.843] <-- ACK 44 window 44, RTT 0.100, RTO 0.140)
[0.845] <-- ACK 45 window 45, RTT 0.100, RTO 0.140)
[0.987] --> FIN 45 (attempt 1 of 50, RTO 0.140)
[1.089] <-- FIN-ACK 45 window fc19a074
Main: transfer finished in 0.845 sec, 620.46 Kbps, checksum: fc19a074
Main: estRTT 0.100, ideal rate 1171.200 Kbps