Reasons not to Parallelize TCP Connections for Fast Long-Distance Networks

Zongsheng Zhang
Go Hasegawa
Masayuki Murata

Osaka University

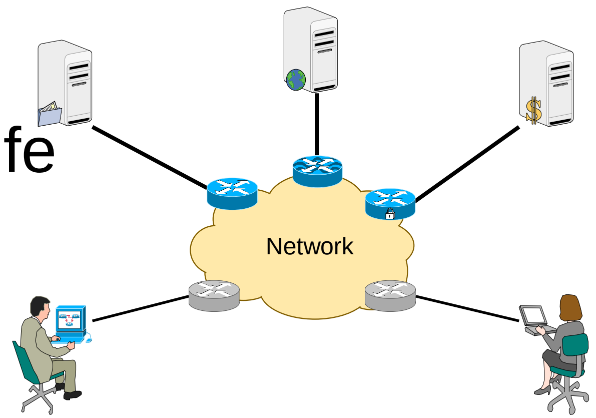
Contents

OSAKA UNIVERSITY

- Introduction
- Analysis of parallel TCP mechanism
- Numerical results
- Conclusion

Introduction

- Status of today's network
  - Networks have been a part in daily life
  - TCP/IP is the keystone of networks
  - TCP Reno is the most widely used transport-layer protocol



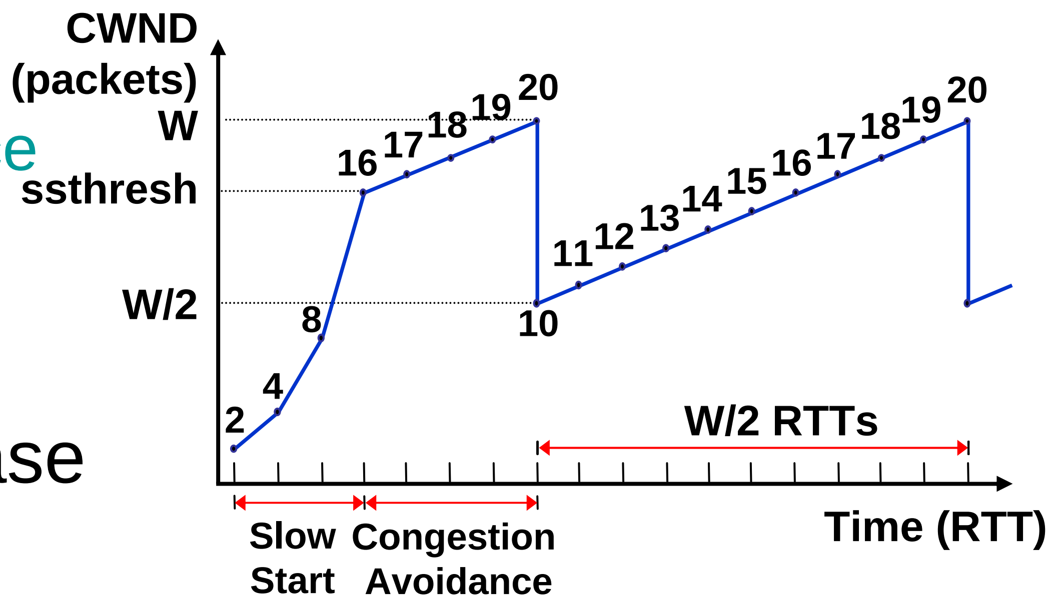
However,

- Continuous and explosive growth of the Internet
  - Especially fast long-distance networks (LFNs).
- Appearance of data intensive applications, e.g., Data Grid, Storage Area Network.
  - Hosts have gigabit-level network interface.
  - Perform backups, synchronize databases.

Inability of TCP Reno

Congestion control in today’s Internet

- Transmission Control Protocol (TCP)
  - Instrumental in preventing congestion collapse
  - Limit transmission rate at the source
  - Window-based rate control -- Congestion window (CWND)
- Four algorithms:
  - Slow-start
  - Congestion-avoidance
  - Fast-retransmit
  - Fast-recovery
- Additive Increase Multiplicative Decrease (AIMD) algorithm

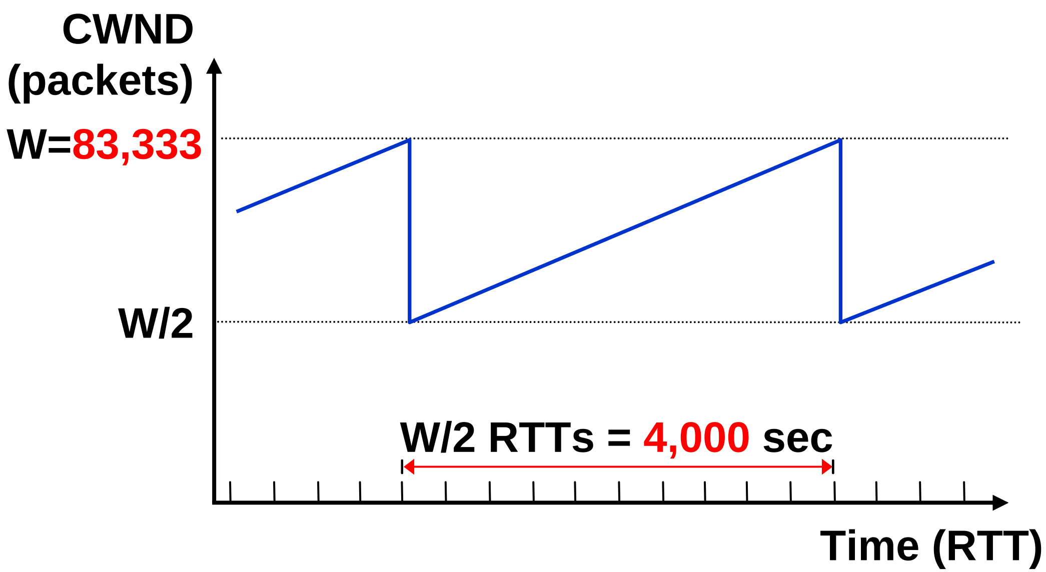


An example[★]

- For fully utilizing a link of 10 Gbps with
  - Round Trip Time (RTT): 100 ms
  - Packet size: 1,500 bytes
  - Requirement: CWND = **83,333** packets

If the AIMD algorithm is used,

4,000 seconds are needed to recover throughput, once packet loss occurs.



[★] S. Floyd, "HighSpeed TCP for large congestion windows," *RFC 3649*, December 2003.

What's wrong with TCP?

- TCP was designed when T1 (1.544 Mbps) was a fast network.
- Additive Increase Multiplicative Decrease (AIMD) algorithm of TCP Reno in congestion avoidance phase:
  - No packet loss (AI): increase congestion window by one packet/RTT ➡ too slowly
  - Packet loss (MD): decrease congestion window by half ➡ too dramatically
- It doesn't perform well in LFNs because of congestion window (CWND) algorithms.

Solutions

- Patches, e.g., SACK option, NewReno, ECN.
  - ➡ The problem of AIMD is not solved.
- Traditional method: parallel TCP mechanism
  - Parallel TCP is adopted in present applications, e.g. GridFTP. An important reason is that parallel TCP is easy to be implemented in application layer.
- High-speed protocols: New algorithms for updating CWND, e.g., HighSpeed TCP (HSTCP), Scalable TCP, FAST TCP, and XCP.
- An important yet neglected topic: Parallel TCP v.s. High-speed protocols, which should be employed in your future application.

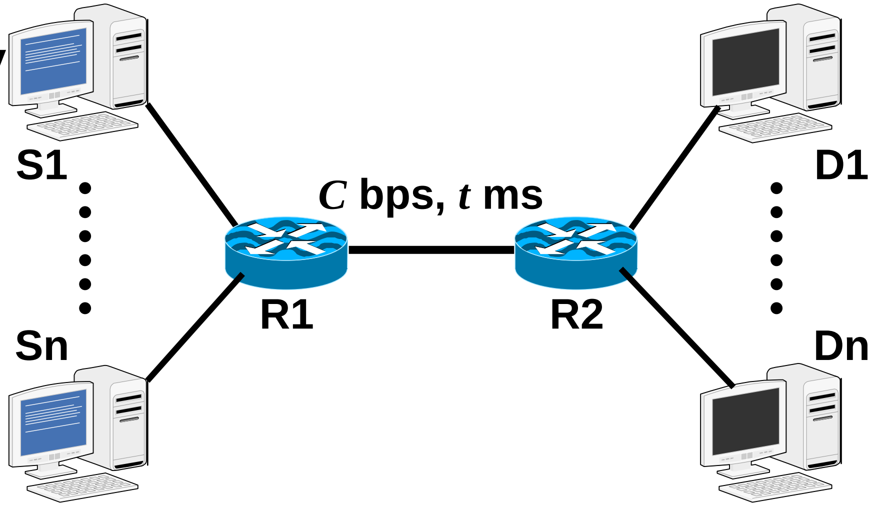
Is parallel TCP really effective?

- Characteristics of parallel TCP
  - Parallel TCP uses many concurrent TCP connections for one task
  - Mechanism of parallel TCP can be viewed from different points, e.g.,
    - It uses a larger AI parameter than normal TCP, or
    - Each TCP connection uses a “stripped” network link

  However,
  - Is it easy to determine the number of TCP connections?
  - Is parallel TCP really effective?

Performance analysis

- Model: Dumbbell topology
- DropTail mechanism
- Performance metrics:
  - *packet drop rate* (*p*)
  - *goodput*



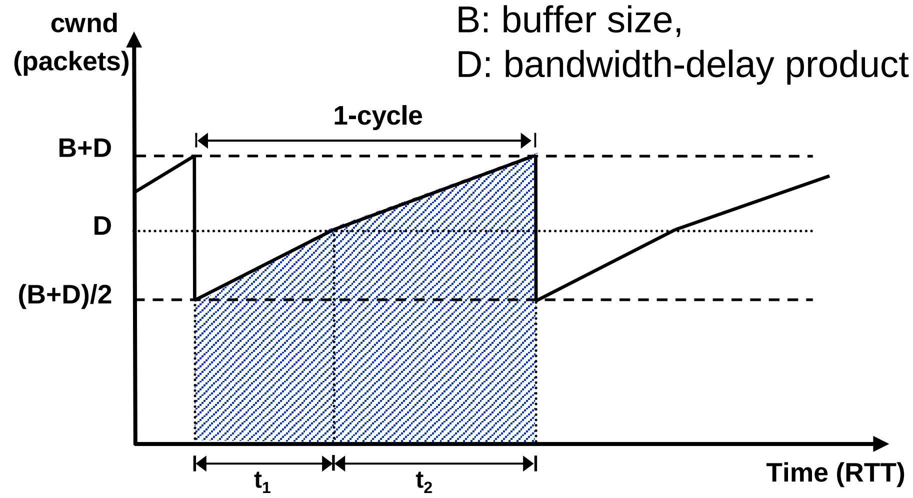
$$goodput = throughput \times (1 - p)$$

- Two extreme cases are considered for analysis.
  - Synchronization case: TCP connections are synchronized ➡ lower limit of throughput
  - Non-synchronization case: TCP connections are not synchronized at all. ➡ upper limit of throughput.

Synchronization case

- Congestion window

$$cwnd \leftarrow cwnd + \frac{a(cwnd)}{cwnd}$$
$$cwnd \leftarrow (1 - b(cwnd)) \times cwnd$$
$$a(cwnd) = N, b(cwnd) = 1/2$$



- Packet drop rate

$$p = \begin{cases} 0 & \text{if } N \times W_{max} \leq B + D \\ \dfrac{8N^2}{3(B+D)(B+D+2N)} & \text{if } N \times W_{max} > B + D \end{cases}$$

- Throughput

$$throughput = \begin{cases} N \times \dfrac{W_{max}}{RTT} & \text{if } N \times W_{max} < D \\ BW & \text{if } D \leq N \times W_{max} \leq B + D \\ \dfrac{N_{pkts} + N \cdot p_{to} \cdot E(n)}{t1 + t2 + p_{to} \cdot E(t)} & \text{if } N \times W_{max} > B + D \end{cases}$$

Non-synchronization case

- Each TCP connection (*square root p formula*):

$$b(p) \approx \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 min\left(1, 3\sqrt{\frac{3bp}{8}}\right)p(1+32p^2)}$$

RTT -- average round trip time, T_0 -- the timeout time,
b -- number of packets that are acknowledged by a received ACK,
p -- packet loss rate.

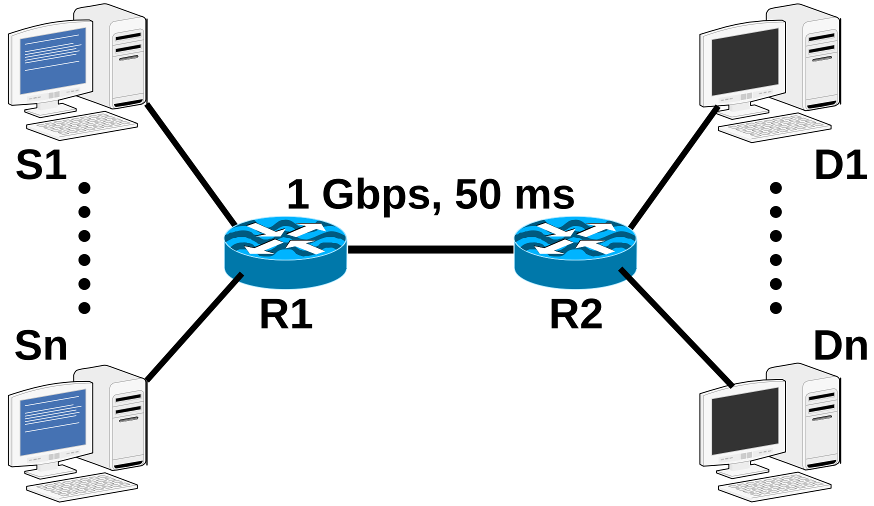
- Total behavior (aggregate CWND is a normal distribution):

$$W(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

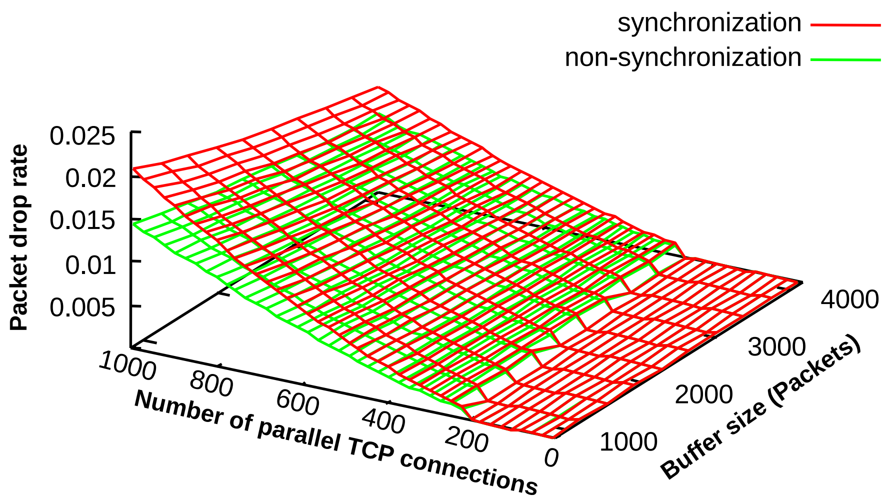
W -- aggregate of congestion window size.
μ -- mean of the aggregate congestion window size,
σ -- standard deviation.

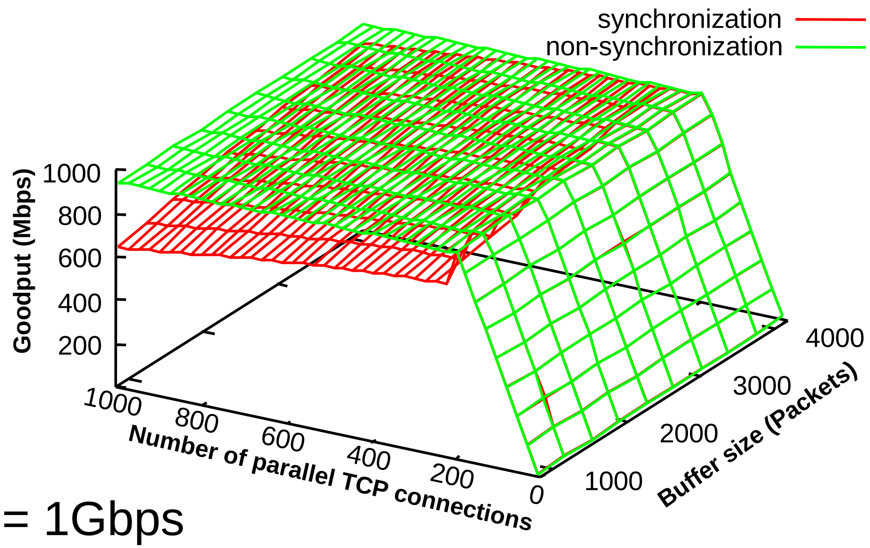
Example

- Parameters:
  - Bandwidth = 1 Gbps,
  - RTT = 100 ms,
  - Buffer size = (0.1--0.5)BDP,
  - Packet size = 1500 Bytes,
  - T0 = 5*RTT,
  - Wmax = 64 KBytes.



Results – goodput & packet drop rate

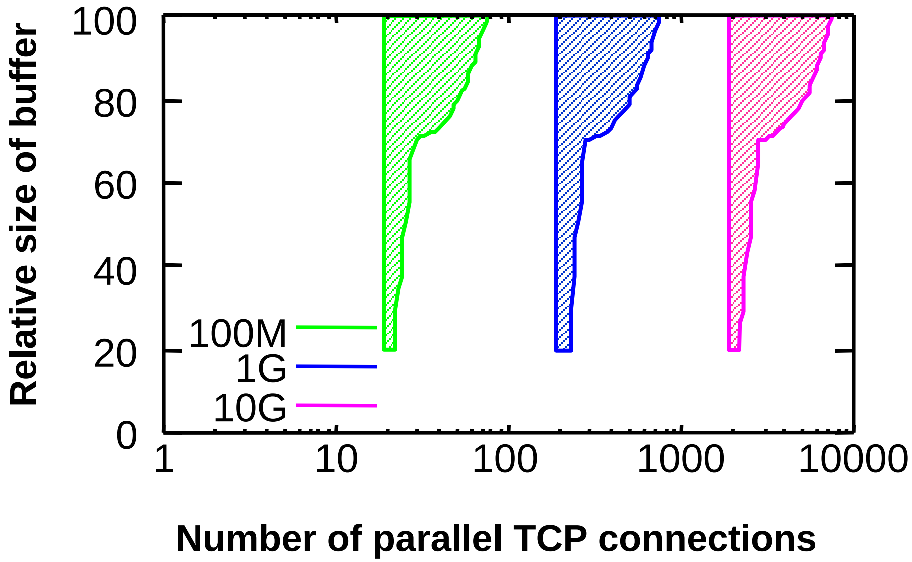


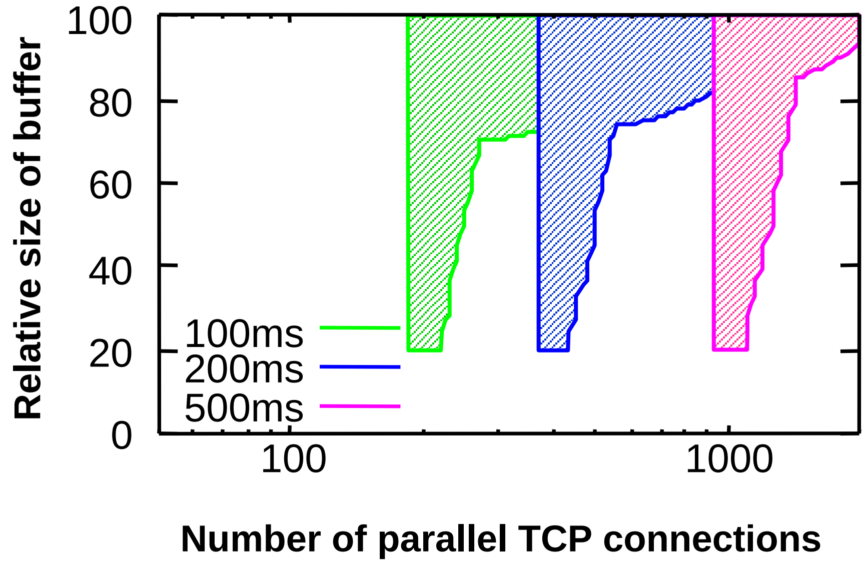


Bandwidth = 1Gbps

- When the number of TCP connections is larger than a certain value, as the number of parallel TCP connection is increased:
  - Packet drop rate becomes large.
  - Goodput is decreased.
- Goodput in non-synchronization case is better than that in synchronization case.
  - However, synchronization is common when DropTail is deployed.
- In synchronization case, if buffer size of router is small, the performance deteriorates significantly as the number of TCP connections is increased.
  - But, it is difficult to build a router with a large buffer size.

Results – contour of goodput





- Parallel TCP possesses the property that can lead to synchronization.
  - Pass through the same path, have the same RTT
- Goodput $\geq$ 95% bottleneck link bandwidth
- Parameters of the left figure: BW = 100 M/1 G/10 Gbps, RTT = 100 ms.
- Parameters of the right figure: BW = 1 Gbps, RTT = 100/200/500 ms.
- Difficulty: Select the number of TCP connections for the expected throughput.
  - Especially in the case of small buffer size of router.
  - The buffer size cannot be large enough as the link bandwidth becomes more large.

Supplemental discussion

- Dynamic network resources allocation of parallel TCP (GridFTP v2) [★]
  - Determination of the granularity of changing the number of TCP connections is required.
  - It is difficult to manage opening/closing of TCP connections and control data channels dynamically.
  - This mechanism determines the number of TCP connections based on measurement of network conditions.
  - Because the number of TCP connections is changed dynamically, setting up the chunk size is not easy.
- For dynamic networks, high-speed protocols can offer more flexibility because of its inherent characteristics.

[★] I. Mandrichenko, W. Allcock, and T. Perelmutov, "GridFTP v2 protocol description," Available as: http://www.ggf.org/documents/GFD.47.pdf, May 2005.

Conclusion

OSAKA UNIVERSITY

- Parallel TCP mechanism, one approach for LFNs, is investigated by mathematical analysis.
- The throughput of two extreme cases, synchronization case and non-synchronization case, are considered as lower and upper limits.
- The analysis results reveal the difficult using parallel TCP in practice for the sake of approving throughput, especially in case of small router buffer size and coming high-speed networks.
- In contrast, high-speed protocols are better choices for your future applications.

Thanks