TCP Congestion Control in High-speed Large Delay Networks

Sudheer Poojary

IISc, Bangalore

July 30, 2010
1. Applications of the large BDP TCP variants

2. Performance Metrics for comparison of the TCP variants
   - Performance Metrics
   - TCP variants considered

3. A comparison of the various TCP variants

4. CUBIC: Experimental Results

5. H-TCP

6. Inter-Protocol Fairness
In the following cases, the performance is insensitive to choice of congestion avoidance algorithm [Leith-Shorten 08]

- The http traffic on the web is dominated by slow-start.
- Video streaming traffic (youtube) is typically rate limited at the server side.
- FTP with large file sizes exercises the congestion control algorithm.
- Data backup applications like rsync.
- Basically applications which involve transfer of large size.
Performance Metrics

- Efficiency (link utilization)
- Convergence rate
- TCP-friendliness
- RTT fairness
- Intra and Inter-protocol fairness

It has been argued that a good metric of user QoS is completion time [Leith-Shorten 08].
TCP variants considered

- In the 2.6.22 linux kernel, 13 TCP congestion control algorithms were available [Hemminger 07].
- Windows Vista uses a different congestion control algorithm called Compound TCP.
- We consider only the popular ones viz., HS-TCP, Scalable, H-TCP, BIC, CUBIC and FAST.
- There have been algorithms such as XCP and RCP which require information from the routers.
- These have not been considered.
TCP throughput is inversely proportional to $RTT^\alpha$

For AIMD and BIC, $\alpha = 2$ [BIC 04]

HSTCP has RTT unfairness $\frac{RTT_2}{RTT_1}$, Scalable has RTT unfairness $\frac{RTT_2}{RTT_1}\infty$

Therefore, Scalable and HSTCP are severely RTT-unfair.

### Table 1: The throughput ratio of two high speed flows over various RTT ratios in 2.5Gbps networks.

<table>
<thead>
<tr>
<th>Inverse RTT Ratio</th>
<th>1</th>
<th>3</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIMD</td>
<td>1.11</td>
<td>6.68</td>
<td>22.03</td>
</tr>
<tr>
<td>HSTTCP</td>
<td>1.01</td>
<td>29.19</td>
<td>107.90</td>
</tr>
<tr>
<td>STCP</td>
<td>1.01</td>
<td>127.23</td>
<td>389.13</td>
</tr>
</tbody>
</table>

**Figure**: Experimental results demonstrating unfair behaviour of HS-TCP and Scalable

- H-TCP, CUBIC and BIC are RTT-fair.
Some Experimental Results

- The experiments were done using rsync application [Hemminger 07].

![Experimental Setup Diagram]

**Figure**: Experimental Setup
Comparison of Convergence Rates

(a) CUBIC

(b) HTCP
Link Utilization

Utilization vs Queue

**Figure:** Experimental results demonstrating link utilization
Applications of the large BDP TCP variants
Performance Metrics for comparison of the TCP variants
A comparison of the various TCP variants
CUBIC: Experimental Results
H-TCP
Inter-Protocol Fairness

Fairness

Figure: Experimental results demonstrating fairness
We compare intra-protocol fairness between two flows with same RTT [CUBIC-Ha].

Figure: Comparison of Intra-protocol fairness
We compare Link utilization of these protocols.

Figure: Comparison of Link Utilization
CUBIC: RTT fairness

- We compare RTT fairness between two flows of the same protocol.
- Out of the two flows, one flow has a fixed RTT of 162 ms.

Figure: Comparison of RTT fairness
H-TCP is another recent congestion control algorithm developed by Leith, Shorten et al.

CUBIC shows slow convergence and is unfair to flows starting at a later time [Leith 08].

This is because CUBIC window growth function favours flows with larger CW.

Instead of throughputs and link utilization, they argue that flow completion time is closely related to user QoS.
Results for H-TCP

- Mean FCT when flows are started at different times.

Figure: Comparison based on Flow completion times
Results for H-TCP

(b) Domestic DSL link located in Dublin, 3Mbps downlink, 256Kb uplink

**Figure:** Comparison based on Flow completion times
The protocols could be competing against each other.

Hence it is essential to compare them against each other.

We consider experimental results [Weigle 08] with two flows from different protocols starting at different times.

The metric used to compare fairness is Asymmetry metric (A)

\[ A := \frac{(x_1 - x_2)}{(x_1 + x_2)} \]  

The system is perfectly fair when \( A = 0 \) and unfair towards user 2 when \( A = 1 \)
Inter-Protocol Fairness: Results

Figure: Comparison based on Flow completion times
Region 1 consists of CUBIC, H-TCP, HS-TCP and H-TCP-BIC

Most CUBIC flows are in region 4 and hence it needs to be more aggressive in stealing BW.

Scalable uses MIMD and is found to be too aggressive (even S-TCP-S-TCP) flows are asymmetric.

It has been shown in [Chiu Jain 89] that MIMD flows do not converge to fairness.

Flows starting at the same time converged to a fair value except for the aggressive protocols.
Conclusions

- The high speed protocols operate in two modes, hence they are TCP-friendly.
- HS-TCP and Scalable TCP are severely RTT unfair.
- Scalable TCP is also too aggressive.
- CUBIC and H-TCP are current popular TCP high-speed variants.
- CUBIC is found to be efficient and fair.
- CUBIC is however less aggressive when competing against other high speed variants.
- CUBIC could be unfair to flows with smaller CW when they are competing with flows with larger CW.
Questions?

Thank You!