Abstract:

We extend our earlier study on the buffer requirement problem for TCP over ABR. Here, a worst case scenario is generated such that TCP sources send a burst of data under the conditions when the sources have high congestion window the ACR (allowed cell rate) rate for ABR is high. We find that ABR using ERICA+ switch algorithm can maintain the queue lengths (hence the buffer requirements) bounded even for the worst case for a reasonable number of sources. We present analytical arguments for the expected queue length and simulation results for varying number of sources and parameter values.
1. Introduction:

We extend our earlier studies [1,2,7,8] of buffer requirements of TCP over ABR. In those studies, we had shown that to achieve zero loss ABR service requires switch buffering which is only a small multiple of round trip times of the feedback delay. The buffering depends heavily upon the switch scheme used.

One of the issues related to buffer sizing is that it is possible for a source to get a high ACR and keep it. As long as, it keeps sending a packet before 500 ms, the use-it or lose-it policy will not be triggered. The source can then use the high ACR to dump a large amount of data suddenly. The effect can be amplified with many sources trying to do the same. This contribution studies this particular case.

In this study we generate a worst case in which the TCP sources have a high congestion window and all ACR rates are high. Under such a condition the TCP sources are made to dump a huge burst of data into the network. We show that even under these circumstances ABR using a good switch algorithm like ERICA+ [3] performs well and controls the queues. In our previous simulations we had used only upto 15 TCP sources. In this study we present results of simulation which have upto 200 TCP sources.

2. Generating Worst Case TCP traffic

TCP has its own congestion control mechanism. It uses the "slow start" mechanism which is explained in detail in [4,5]. The amount of traffic which TCP can send into the network is dependent on TCP congestion window size and the ACR value. The maximum burst possible which TCP can send is equal to the maximum congestion window size.
The TCP takes \( \log(\text{congestion window}) \) number of round trips to reach the maximum congestion window. In normal TCP activity when this maximum congestion window is achieved, the ACRs may not be high. In order to generate worst case conditions in which TCP has maximum congestion window size and the ACR is high, we designed the following scenario.

The configuration used is the \( N \) TCP Sources connected to \( N \) TCP Destinations via two switches and a bottleneck link (LINK1). The following figure shows the configuration used.

\[
\begin{array}{c}
\text{Src 1} \\
/ & \text{Dest 1} \\
\text{-------} \text{------} \text{LINK1} \text{-------} \text{-------} \\
\text{Sw1} & \text{Sw2} \\
\text{Src N} \\
\text{-------} \text{------} \\
\end{array}
\]

Initially to build up the congestion window each source sends one segment of data every \( 't' \) milliseconds. One thousand such segments are sent so that congestion window reaches a maximum for all the sources. The sources send segments in a staggered manner, so that not all sources send the segments simultaneously. This is done so that the TCP data can be sent without overloading the network. Since the network is not overloaded, the ACRs are high when the congestion window reaches the maximum values. At this point all the sources synchronize and send a burst of data (burst size equal to maximum congestion window). Since every \( 't' \) milliseconds, one segment is sent and one thousand segments are sent by each source, the synchronization time is just after 1000\( 't' \) milliseconds. This is the worst case burst size which arises due to the TCP sources.

3. TCP Options and ERICA+ parameters

We use a TCP maximum segment size (MSS) of 512 and 1024 bytes.
The MTU size used by IP is generally 9180 bytes, so there is no segmentation caused by IP. Since our simulations are performed under no loss conditions, the results hold even for TCP with fast retransmit and recovery or TCP with SACK (Selective Acknowledgement).

The TCP data is encapsulated over ATM as follows. First, a set of headers and trailers are added to every TCP segment. We have 20 bytes of TCP header, 20 bytes of IP header, 8 bytes for the RFC1577 LLC/SNAP encapsulation, and 8 bytes of AAL5 information, a total of 56 bytes. Hence, every MSS of 512 bytes becomes 568 bytes of payload for transmission over ATM. This payload with padding requires 12 ATM cells of 48 data bytes each.

The ERICA+ algorithm is an extension of ERICA which uses the queueing delay as an additional metric to calculate the feedback. ERICA+ uses four parameters: a target queueing delay ($T_0 = 500$ microseconds), two curve parameters ($a = 1.15$ and $b = 1.05$), and a factor which limits the amount of ABR capacity allocated to drain the queues ($QDLF = 0.5$). In our previous study [6] we had enhanced our ERICA+ algorithm to handle variance in traffic. In our current simulations, this exponential averaging option is included to handle the variances in the traffic. The ERICA+ algorithm is designed to handle short spikes in ACR, achieve fairness and correctly count bursty sources.

4. Analytical prediction

In this section we derive the analytical value for the queue lengths. The queue length is given in two formulae for underloaded and overloaded networks.

\[
Q_{len} = N \times \text{floor}(\frac{cwnd_{max}}{48}) \quad \text{for } N \leq \text{floor}(\frac{1000 \times t}{g}) \quad (\text{formula 1})
\]

\[
= N \times 367 \times t \quad \text{cells} \quad \text{for } N > \text{floor}(\frac{1000 \times t}{g}) \quad (\text{formula 2})
\]

where

- $N$ - number of sources
- $cwnd_{max}$ - maximum congestion window size
- $t$ - time between consecutive segments
- $g$ - time between sources sending segments

Initially, for a small number of sources, the network is underloaded, so the ACRs are high when the burst occurs. When all the sources send the burst simultaneously, the whole burst can be sent into the network and this results in large switch queues.
Let 'cells_per_mss' be the number of cells required to send a segment and 'mss' be the maximum segment size.

The burst size due to one source = floor(cmall_mss*(cwnd_max/mss)) cells

So if there are N sources, the expected queue length is

\[ Qlen = N \times \text{floor}(\text{cells_in_mss} \times (\text{cwnd_max/mss})) \text{ cells} \]

Substituting cells_in_mss = ceiling(mss/48) in the above we get

\[ Qlen = N \times \text{floor}(\text{cwnd_max}/48) \text{ cells} \]

Under no loss condition, the TCP congestion window increases exponentially in the "slow start" mechanism until it reaches the maximum value. As the number of sources increase the load of the network increases. In the simulation every source sends a segment of size 512 bytes every 't' milliseconds. The time between two sources sending the segment is 'g' microsecond. At time 0, source 1 sends a segment, then at time 'g' microsecond, source 2 sends, at time '2g' microsecond source 3 sends and so on. Also source-1 sends its second segment at time 't' millisecond and so on (See figure 2).

Figure 2: Timing diagram: sources sending segments

---

\[ \text{Legend} \]
S-i - Source number 'i'.

The network becomes overloaded when the input rate is greater than the output rate. In the simulation, the network will get overloaded if number of sources
Once the network is overloaded the ACRs are low and each of the N sources gets approximately \((1/N)\)th of the bandwidth. Since the ACRs are low the burst which occurs after 1 second should not give rise to large queues. There are still switch queues which occur initially when the network has still not realized that it is overloaded.

The TCP congestion window grows exponentially whenever it receives an acknowledgement. Let \(d\) be the length of the links in kms. The acknowledgement arrives after round trip time of \(30*(d/1000)\) ms. But by this time the source would have generated more segments to send. After each round trip, each source will send twice the number of segments. A larger round trip can give rise to a larger burst. The networks get overloaded when the number of segments sent just fills up the pipe.

Let \(k\) be the number of round trips it takes to overload the network. Time taken for transmitting one cell is 2.72 microsecond. Hence, in \(367*t\). Each segment gives rise to \('cells_in_mss'\) cells. Therefore, the network gets overloaded when

\[
2^k * \text{cells\_in\_mss} = 367*t
\]

\[
\Rightarrow \quad k = \log \left( \frac{367*t}{\text{cells\_in\_mss}} \right)
\]

\[
\Rightarrow \quad k \approx 5
\]

For \(t = 1\) millisecond. \(mss = 512\)

\[
2^k * \text{cells\_in\_mss} = 367*t
\]

\[
\Rightarrow \quad k = 4.934
\]

\[
\Rightarrow \quad k \approx 5
\]

(\(\approx\) means approximately equal to)

Hence, in this case after 6 (= \(k+1\)) round trips the network detects the overload if \(N > 20\). The maximum queues should occur after 6 round trips for \(N > 20\).

The value of the maximum queue in an overloaded network can be derived as follows.
The burst due to one source = \(2^k \times \text{cells\_in\_mss} = 367\times t\) cells

Hence the burst due to \(N\) sources = \(N\times367\times t\) cells

For \(t = 1\) millisecond, \(g = 50\) microsecond, \(mss = 512\) and \(cwnd\_\text{max} = \) 65536, we get

\[
Q_{\text{len}} = \begin{cases} 
N\times1365 & \text{for } N \leq 20 \\
N\times367 & \text{for } N > 20 
\end{cases}
\]

5. Simulation results

We present the maximum queues at the bottleneck switch for the simulations where the number of sources \((N)\) varies from 2 to 200. The results are shown in Table 1. The same information in graphical form will be presented at the ATM Forum and will be available at our web site after the Forum meeting.

The length of all the links are 1000 km, giving rise to 15 ms propagation delay from source to destination. The round trip time (RTT) is 30 millisecond. The feedback delay (the delay between the source and the bottleneck link) is 10 ms. A round trip of 30 ms corresponds to 11029 cells. The other parameter values are \('t' = 1\) millisecond, \('g' = 50\) microsecond, \('mss' = 512\) bytes, \('cwnd\_\text{max}' = 65536\).

The maximum queue length for sources \(N \leq 20\) was at time around 1 second. For \(N > 20\), the maximum queue was at time around 300 millisecond (which agrees with the analytical prediction). The queue lengths increase with the number of sources as expected. The queue lengths also correspond to the burst size sent by the sources. But, this is true only when the number of sources are less than or equal to 20. For \(N = 30\), there is sharp decrease the maximum queue length and again it starts (for \(N > 40\)) increasing but now it increases at a slower rate.

From the Table 1, it can be seen that for \(N \leq 20\), the simulation agrees well with the analytical values. For \(N > 30\) initially the queue lengths in the simulation are higher than the analytical value, but later it becomes lower than the expected queue length. This can be explained as follows, initially the traffic generated in each cycle can be sent without pending queue in the next cycle. But as the number of sources increase \((N > 80)\) in each
cycle there are some pending cells which get sent in the next cycles. Because of this the network detects the overload at lesser queue lengths than the analytical value. A more rigorous analytical derivation is necessary to take this into account.

Table 1: Effect of number of sources
Actual and Expected queue lengths

<table>
<thead>
<tr>
<th># TCP Sources</th>
<th>Max Queue size</th>
<th>Analytical Queue Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(cells)</td>
<td>(cells)</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>2</td>
<td>1575</td>
<td>2730</td>
</tr>
<tr>
<td>3</td>
<td>3149</td>
<td>4095</td>
</tr>
<tr>
<td>5</td>
<td>6297</td>
<td>6825</td>
</tr>
<tr>
<td>10</td>
<td>14131</td>
<td>13650</td>
</tr>
<tr>
<td>20</td>
<td>29751</td>
<td>27300</td>
</tr>
<tr>
<td>30</td>
<td>20068</td>
<td>11010</td>
</tr>
<tr>
<td>40</td>
<td>19619</td>
<td>14680</td>
</tr>
<tr>
<td>50</td>
<td>24162</td>
<td>18350</td>
</tr>
<tr>
<td>60</td>
<td>28006</td>
<td>22020</td>
</tr>
<tr>
<td>70</td>
<td>30109</td>
<td>25690</td>
</tr>
<tr>
<td>80</td>
<td>31439</td>
<td>29360</td>
</tr>
<tr>
<td>90</td>
<td>34530</td>
<td>33030</td>
</tr>
<tr>
<td>100</td>
<td>38088</td>
<td>36700</td>
</tr>
<tr>
<td>110</td>
<td>43672</td>
<td>40370</td>
</tr>
<tr>
<td>120</td>
<td>44939</td>
<td>44040</td>
</tr>
<tr>
<td>130</td>
<td>44708</td>
<td>47710</td>
</tr>
<tr>
<td>140</td>
<td>44744</td>
<td>51380</td>
</tr>
<tr>
<td>150</td>
<td>46058</td>
<td>55050</td>
</tr>
<tr>
<td>160</td>
<td>48880</td>
<td>58720</td>
</tr>
<tr>
<td>170</td>
<td>50784</td>
<td>62390</td>
</tr>
<tr>
<td>180</td>
<td>49961</td>
<td>66060</td>
</tr>
<tr>
<td>190</td>
<td>53366</td>
<td>69730</td>
</tr>
<tr>
<td>200</td>
<td>55618</td>
<td>73400</td>
</tr>
</tbody>
</table>

To study the effect of different parameters a full factorial experiment was carried out for different parameter values. For a given number of sources the following parameters were varied: maximum segment size (mss = 512,1024 bytes), time between two sources sending segments (g = 50, 100 microseconds), time between successive segments of a source (t = 1, 10 milliseconds) and distance of the links (d = 1000,2000 kms).
Table 2: Full factorial experiment

<table>
<thead>
<tr>
<th>#</th>
<th>mss/g/t/d</th>
<th>N=3</th>
<th>N=10</th>
<th>N=30</th>
<th>N=40</th>
<th>N=50</th>
<th>N=100</th>
</tr>
</thead>
</table>

(NA - data not available)

Table 2 shows for number of source $N=3,10,30,40,50,100$ the maximum queue sizes for 16 experiments. From Table 2, the following observations can be made. These observations support the analytical results.

- In line numbers 10 and 14 for $N = 3,10$ the queue size is less compared to other lines, since in these cases congestion is detected early, so the ACRs are low when the burst is sent. Queue length is given by the formula 2.

- For $N = 30,40,50,100$, when $t = 10$ millisecond, the queue lengths agrees with one given by the formula 1.

- If the network is overloaded then a larger round trip time gives larger queue lengths. (e.g., see line 1 and 2 for $N = 30,40,50$).

- Lines 3 and 4 indicate that in underloaded conditions the queue length is given by the formula 1.

- As expected the segment size of 512 and 1024 do not affect the queue sizes.
6. Summary

We generated artificially a worst case scenario and studied the buffer requirement problem for TCP over ABR. The buffer size required depends heavily on the switch scheme used. Our simulations are based on our ERICA+ switch scheme. For worst case scenario generated both analytical prediction and simulation results for queue lengths (and hence buffer requirements) are given. The simulation agrees well with the analytical prediction. The queue lengths is affected by maximum congestion window size the round trip time, network congestion (overloaded or underloaded) and number of sources. It is not affected by maximum segment size.

7. References:


Communications Magazine, January 1997 issue.


All our papers and ATM Forum contributions are available through http://www.cis.ohio-state.edu/~jain/
Worst TCP: Qlen Vs Number of sources

Queue length (cells)

Number of source

Simulation
Analytical