User-Perceived Latency to Measure CCAs

Mingrui Zhang @ UNL, Vidhi Goel @ Apple, Lisong Xu @ UNL

Congestion control algorithms (CCAs) used by transport protocols like TCP are usually designed and evaluated using the following performance metrics: throughput, round-trip time (RTT), fairness, and link utilization. While these metrics are important, especially for network operators, they are not sufficient to describe the latency experienced by the users. In this paper, we propose an additional metric, called User-Perceived Latency (UPL), to more accurately measure the latency perceived by the users than RTT.

1 Current Latency Metric: RTT

Let's consider a case where a user sends a request to a server using a TCP connection. One example is when a user sends an HTTP request via a web browser to a web server. Fig. 1(a) illustrates the RTT [1] of the request. Specifically, RTT = $L_F + L_{AB}$.

- L_F = the total latency for the request to travel from the user to the server (i.e., forward direction), which includes all related propagation delays, queuing delays, and transmission delays of the request.
- L_{AB} = the total latency for the ACK to travel from the server to the user (i.e., backward direction), which includes all related propagation delays, queuing delays, and transmission delays of the ACK.

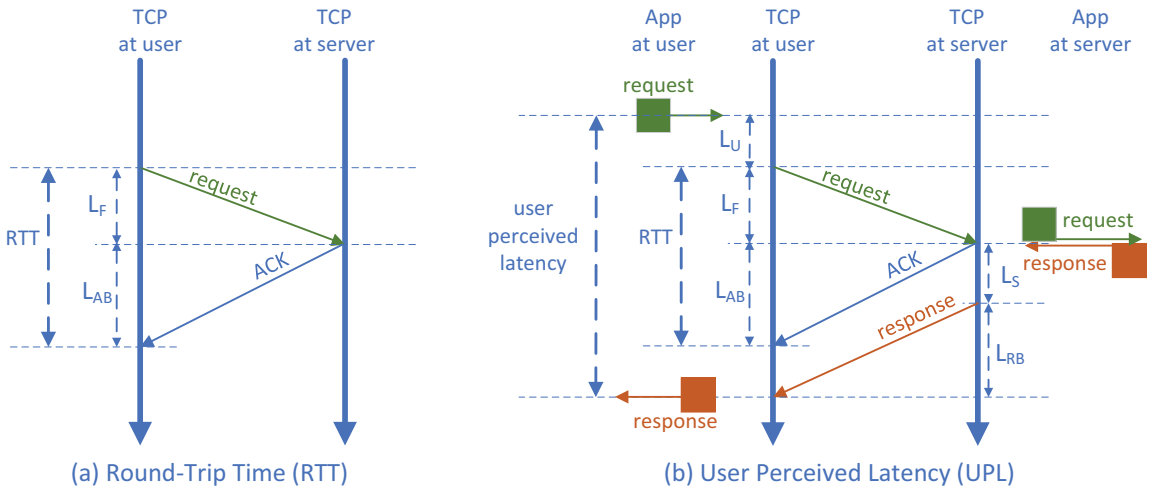

Figure 1: RTT and UPL

[1] It does not show the possibly additional delay of the delayed acknowledgment technique at the TCP server.

2 Proposed Additional Latency Metric: UPL

When a user sends a request to a server using a TCP connection, the user typically waits for the response from the server. For example, a user sends an HTTP request via a web browser to a web server and then waits for the requested web page. Fig. 1(b) illustrates the UPL [2], which is the total latency for the user to get the response after sending the request. Specifically, UPL = $L_U + L_F + L_S + L_{RB}$.

- L_U = the latency that the request waits in the TCP send buffer at the user until the TCP congestion window allows.
- L_S = the latency that the response waits in the TCP send buffer at the server until the TCP congestion window allows.
- L_{RB} = the total latency for the response to travel from the server to the user (i.e., backward direction), which includes all related propagation delays, queuing delays, and transmission delays of the response.

In this paper, we propose UPL as an additional performance metric for designing and evaluating transport protocols and specifically the CCAs that they use. We believe that UPL more accurately describes the user perceived latency than RTT due to the following differences.

1) Intuitively, UPL describes the overall latency perceived by a user both inside the network and at the host (i.e., user and server), whereas RTT only captures the latency inside the network. For example, if the ACK and response experience the same total latency from the server to the user (i.e., $L_{AB} = L_{RB}$), we have UPL = $L_U + L_F + L_S + L_{RB}$ = $L_U + L_F + L_S + L_{AB}$ = $(L_F + L_{AB}) + (L_U + L_S)$ = RTT + $(L_U + L_S)$, where $L_U + L_S$ is the total latency at the host.

2) While both RTT and UPL depend on the network traffic in both the forward and backward directions, only UPL depends on the TCP congestion windows in both forward and backward directions. Specifically, L_U depends on the congestion window of the forward direction and L_S depends on the congestion window of the backward direction.

3) While RTT typically involves only a pair of packets (e.g., a data packet and the corresponding ACK), UPL may involves a sequence of packets if the request or response is very long. For example, if the requested web page is very long and requires a sequence of packets, then UPL depends on when the user receives the last packet of the request.

3 Discussion

Delay-based CCAs, such as BBR, attempt to reduce buffer bloat on the network by periodically measuring the minimum RTT and adjusting their sending rates to avoid filling up the bottleneck buffer. While this is great in keeping the buffer bloat on the network to a minimum, but it could still inflate the UPL if this leads to buffering at the hosts. While one can argue that buffering on the network is worse than buffering on the hosts (e.g., user and server), but both these could lead to an increase in UPL which is what matters to the user. So, if we make UPL as one of the performance metrics for CCAs, this would promote better design of CCAs to measure what the actual user experience would be like. Note that we propose UPL as an additional performance metric instead of replacing current metric RTT.

[2] It assumes that the time for the server to analyze the request and generate the response is zero.