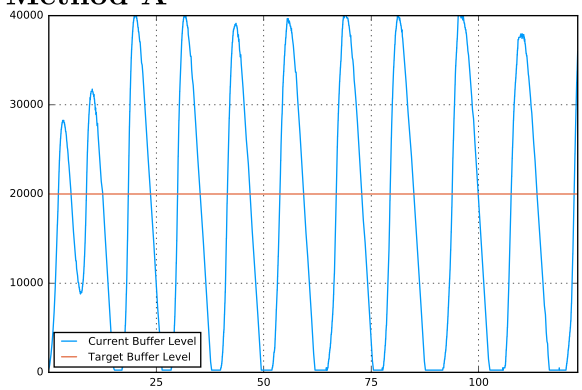
Lab 5: Congestion Control for Audio Streaming

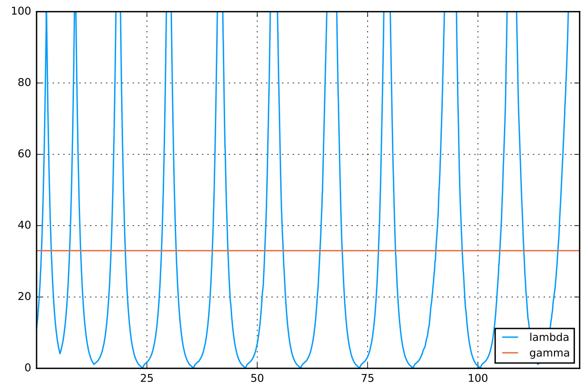
Duc Viet Le
CS536

November 17, 2016

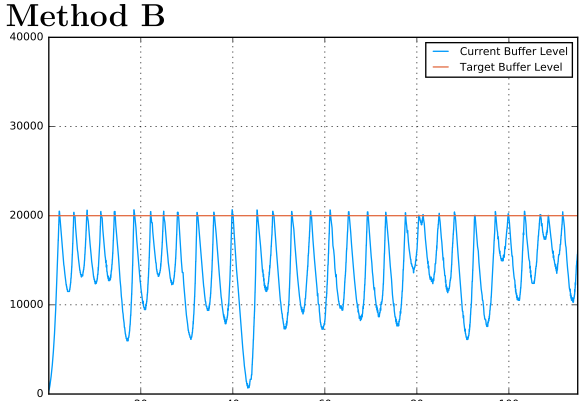
Problem 1.
Below are plots for each method:

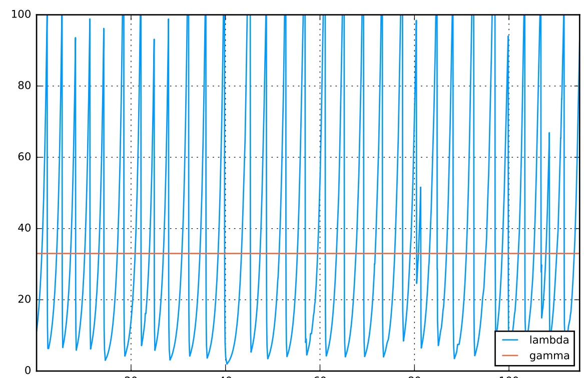
Method A



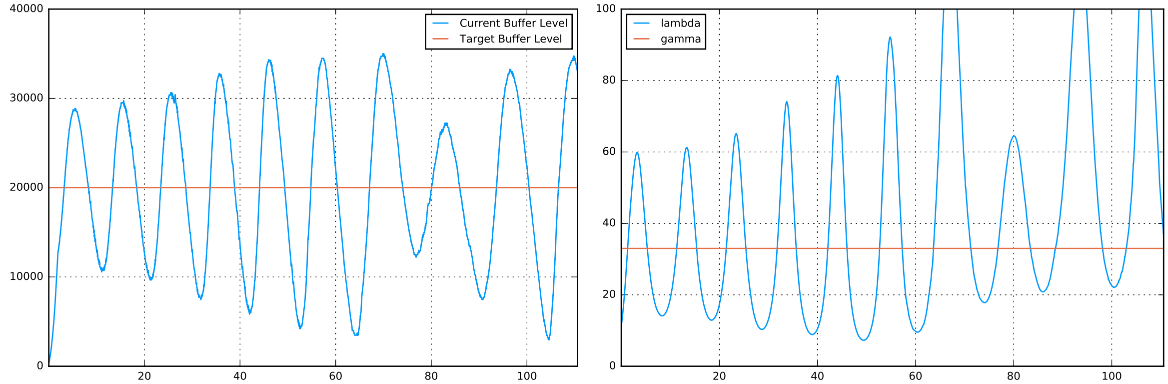


Method B

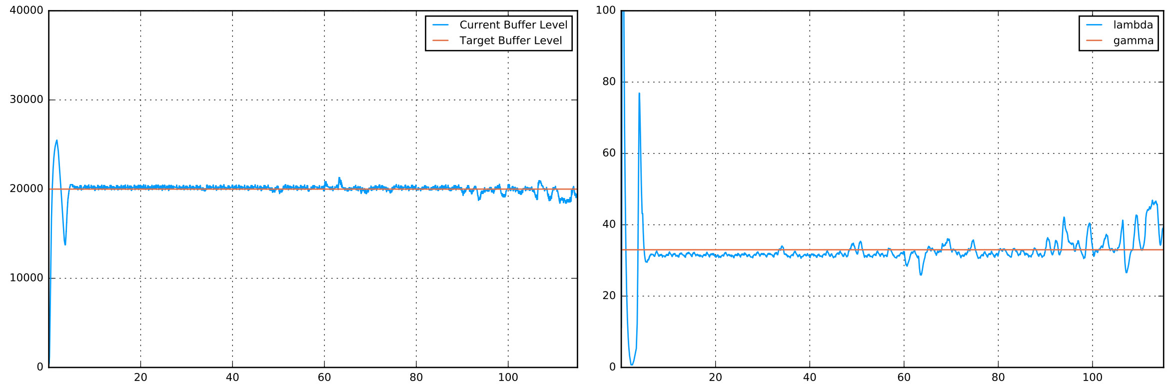




Method C



Method D



Discussion: As we discussed in class, the plot is sample of first 50 seconds of 4 methods. The parameters we used are:

1. For method A, $a = 2$
2. For method B, $a = 2, \delta = 0.5$
3. For method C, $\epsilon = 0.00005$
4. For method D, $\epsilon = 0.001, \beta = 0.1$

Method D has the best results and the streaming using method D seems to have better quality. Method B,C also has a decent streaming quality as current buffer level always above payload size. Method A does not have a good streaming quality, as the current buffer level always hits bottom. For testing streaming server with 2 clients, I don't see any significant differences when streaming server supports two clients vs one client.

Problem 2. For this problem, on client side, I keep track of the sequence number of last package (i.e `lastPacket`), and compare it with sequence number of incoming package. If the sequence number of new package is greater than the last sequence plus 1, I send a negative `ACK`, and update last sequence accordingly. On server side, I store an array that store `audioBuf` packets and `packageCounter` that store how many packets sent. If the server receives a negative `ACK`, it checks if the sequence number is greater than `packageCount - audioBuf`. If yes, the packet is still in the buffer, the server will retransmit; otherwise, ignore that negative `ACK`.

Below is graph, using method D: