NTNU
The Norwegian University of
Science and Technology
Department of Telematics

TTM4100
Communication – Services and Networks

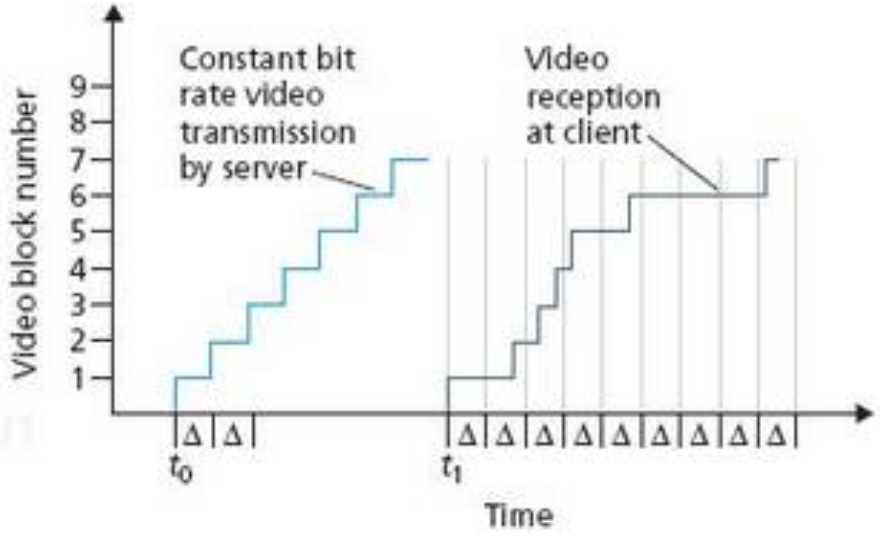
Assignment for Chapter 7: "Multimedia Networking"

Deadline of submission: **26.03.2017**

The assignment questions are selected from the Review Questions and Problems of Chapter 7 in the textbook: J. F. Kurose and K. W. Ross. *Computer Networking: A Top-Down Approach (International Edition, sixth edition)*. Please note that there are modifications to the questions in the textbook, the questions in this document are to be used if there are differences.

The following selected questions should be answered and the answers should be submitted to the Its Learning System. For these selected questions, two or more choices are provided for each of their sub-questions, and one of them is correct.

1. Consider the figure below. Similar to our discussion of Figure 7.1, suppose that video is encoded at a fixed bit rate, and thus each video block contains video frames that are to be played out over the same fixed amount of time, Δ. The server transmits the first video block at t_0, the second block at t_0+ Δ, the third block at t_0+ 2Δ, and so on. Once the client begins play out, each block should be played out Δ time units after the previous block. (Problem P1, Chapter 7, page 658)



1.a) Suppose that the client begins play out as soon as the first block arrives at t_1. In the figure above, how many blocks of video (including the first block) will have arrived at the client in time for their play out?

1.a.1 Package 1, package 4, package 5, package 6
1.a.2 All the packages
1.a.3 Package 1, package 2, package 5, package 6
1.a.4 Package 4 and package 5

1.b. Suppose that the client begins play out now at t_1+ Δ . How many blocks of video (including the first block) will have arrived at the client in time for their play out?

1.b.1 Package 1, package 4, package 5, package 6
1.b.2 All the packages
1.b.3 Package 1, package 2, package 3, package 4, package 5, package 6

1.b.4 Package 1, package 3, package 4, package 5, package 6

1c. In the same scenario at (1.b) above, what is the largest number of blocks that is ever stored in the client buffer, awaiting play out?

1.c.1 1 packages
1.c.2. 2 packages
1.c.3 3 packages
1.c.4 4 package
1.c.5 7 package

1d. What is the smallest play out delay at the client, such that every video block has arrived in time for its play out?

1.d.1 1 Δ
1.d.2. 2 Δ
1.d.3 3 Δ
1.d.4 4 Δ

2. Recall the simple model for HTTP streaming shown in figure 7.3. Recall that B denotes the size of the client's application buffer, and Q denotes the number of bits that must be buffered before the client application begins playout. Also r denotes the video consumption rate. Assume that the server sends bits at a constant rate x whenever the client buffer is not full. (Problem P2, Chapter 7, page 659)

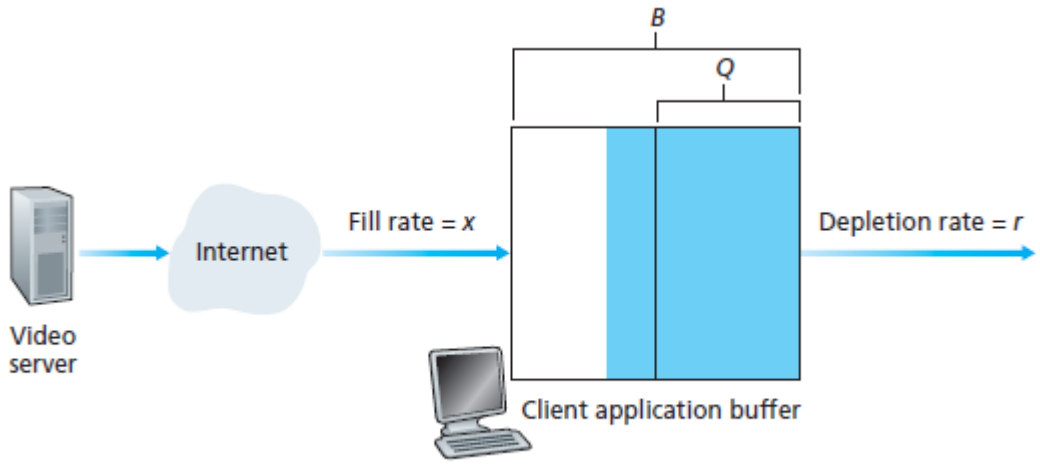


Figure 7.3 ♦ Analysis of client-side buffering for video streaming

2.a) Suppose that x < r. As discussed in the text in this case playout will alternate between periods of continuous playout and periods of freezing. What is the length of each continuous playout as a function of Q, r and x?

2.a.1 Q/(x-r)
2.a.2 Q/(r-x)
2.a.3 x/(Q-r)

2.a.4 r/(x-Q)

2.b) What is the length of each continuous freezing period as a function of Q, r and x?

2.a.1 Q/r
2.a.2 Q/x
2.a.3 Q/(r-x)
2.a.4 Q/(x-r)

2.c) Now suppose that x > r. At what time t = t_f does the client application become full?

2.a.1 Q/(x-r)
2.a.2 Q/x
2.a.3 Q/x + (B-Q)/(x-r)
2.a.4 Q/x + (B-Q)/(r-x)

3.a) Consider an audio conference call in Skype with N > 2 participants. Suppose each participant generates a constant stream of rate r bps. How many bits per second will the call initiator need to send? And how many bits per second will each of the other N-1 participants need to send? (Problem P14, Chapter 7, page 662)

*3.a.1) Initiator: (N-1)*r, other: r*
*3.a.2) Initiator: (N-1)(N-1)*r, other: r*
*3.a.3) Initiator: N*r, other: r*N*

3.b) Repeat part (a) for a Skype video conference call using a central server.

*3.b.1) Initiator: r, other: r*N*
*3.b.2) Initiator: r*N, other: r*
3.b.3) Initiator: r, other: r

3.c) Repeat part (b), but now when each peer sends a copy of its video stream to each of the N-1 other peers

*3.c.1) Initiator: (N-1)*r, other: (N-1)*r*
*3.c.2) Initiator: N*r, other: (N-1)*r*
3.c.3) All: (N-1)(N-2)*r*