CPSC 441 Assignment 5

Name: Haohu Shen

UCID: 30063099

Tutorial: 04

1. When a timeout occurs, which segments are retransmitted by your program given that sending and receiving threads might be adding/moving segments to/from the transmission queue at any time? Explain your design choice.

When a timeout occurs, the run() method of the timer task will be executed at the start of line 188, thus if the last valid ACK has been received from the server, the timer task will abandon this timeout and no segments will be retransmitted.

Otherwise, the code between line 196 and line 205 will be executed under the condition that the transmission queue is locked. Thus all segments that exist in the queue at this moment will be retransmitted to the server and any modification on the queue, including removing or adding segments by other threads at the same moment will fail because of the lock and they have to wait for the retransmission.

2. What happens in your implementation if a new segment is transmitted by the sending thread while the timer task is executing?

If a new segment is transmitted, the code in line 247 is executed, after that, since the code from line 246 to line 256 are surrounded by the synchronized block on the transmission queue, any outer access to the queue at this moment from the timer task if the timeout happens have to wait for the execution of the code inside the block. On the other hand, if the segment is the first element in the queue, the timer task will be reset.

3. What happens in your code if the receiving thread stops the timer task, but while it is still processing the received ACK, the sending thread sends a new segment? Does this result in multiple timer tasks being started in your program? Explain your answer.

After the receiving thread receives an ACK, it will enter a synchronized block on the transmission queue at line 272, thus if the sending thread wants to send a new segment or check if its size is 1, it has to wait until the receiver completes its update on the transmission queue, thus and ‘if’ branch in line 254 will fail to execute, and the timer task will not start in the sender thread at this moment. Thus the timer can be safely cancelled in the receiver thread without other interferences. Also the timer can safely start in the line 292 after the update on the transmission queue since it is still inside the synchronized block. Therefore, the situation that multiple timer tasks being started will not happen in the program.