**Networking Assignment – Group 13**

Group Members

* Rachael Espie
* Magdalena Veleva
* Silvi Koycheva
* Aleksandra Kliaugaite

**Part 1**

1. Formulas Used:

Raw Data Rate = (Video Resolution)x(Bits per Pixel)x(Frame Rate)x(3)x(Seconds)

Compressed Video Data = Raw Data Rate / Compression Rate

Total Overhead per Packet = 7 x Header Data (bits)

Total Packets Needed for Payload = (Compressed Video Data x 10⁹ (converting to bits))/Packet Length

Total Bits Needed to Carry Overhead Information = Total Overhead per Packet x Total Packets Needed for Payload

Total Number of Bits = Compressed Video Data + Total Bits Needed to Carry Overhead Information

Physical Layer Bit Rate Required for Live Streaming = Total Number of Bits/(Video Length in Seconds)

Calculations:

Raw Data Rate = (3,840 x 2,160) x 12 x 100 x 3 x (100 x 60) = 179,159,040,000,000 = 179,159.04 Gbits

Compressed Video Data = 179,159.04 / 250 = 716.64 Gbits

Header Data in Bits = 30 \* 8 = 240 bits

Total Overhead per Packet = 7 \* 240 = 1,680 bits

Total Packets Needed for Payload = (716.64\*109)/10,000 = 71,664,000 bits

Total Bits Needed to Carry Overhead Information = 1,680 \* 71,664,000 = 120,395,520,000 bits = 120.4 Gbits

Total Number of Bits = 716.64 + 120.4 = 837.04 Gbits

Physical Layer Bit Rate Required for Live Streaming = 837.04/(100\*60) = 0.1395 Gbits/s

= 139.5 Mbits/s

Data Rate After Channel Coding = 3/2\*139.5 **= 209.25 Mbits/s**

1. Formulas Used:

QAM Symbol Rate = 1/4 x Channel Coding x Total Number of Bits for Live Streaming

Calculations:

16 QAM = 2⁴ QAM

1/4\*3/2\*139.5 **= 52.31 Msymbols/s**

1. No, we cannot stream as the required bandwidth - 52.31 is above the available bandwidth - 40 Msymbols/s
2. Formulas Used:

File Size in Msymbols = Symbol Rate \* Transmission Time in Seconds

Download Time = File Size / Channel Bandwidth

Total Download Time = Download Time + Latency

Calculations:

Let’s assume we have a UDP connection. Then the download time is 100 min - the length of the video.

File Size in Msymbols = 52.3143 \* 100 \* 60 = 313,885.8 Msymbols

But we don’t have 52.3143 Msymbols / s available. We have channel bandwidth of 40  Msymbols / s available and we can’t stream the video in real-time.

That’s why for TCP:

Download time = = 313,885.8 / 40 = 7,847.145 s = 130.79 mins

We should also add the latency to get the total download time.

Total Download Time = 7,847.145 s + 0.13 s = 7,847.275 s **= 130.79 mins**

1. Formulas Used:

TCP Throughput = (TCP Window Size Bits/ Round Trip Latency)/Sqrt(PLR)

Round Trip Latency = (Total Distance between London/Tokyo\*2 IN METRES (\*1000)) /Speed of Cable

Bandwidth in Bit/s = Bandwidth in Msymbol/s \* Bits per Symbol

TCP Window Size in Bits = Bandwidth in Bit/s \* Round Trip Latency in Seconds

Calculations:

PLR is the % packet loss rate = 5%

Round Trip Latency = 26000x10³ / 2x10⁸ = 0.13 seconds = 130 ms (milliseconds)

Bits per Symbol = 4 because 2⁴ = 16 which is the QAM we are using

Bandwidth in Bits = = 40\*4 = 160 Mbits/s = 160 000 000 bits/s

TCP Window Size in Bits = 160 \* 10⁶ \*0.13  = 20.8 \* 10⁶ bits  = 20.8 Mbits

TCP Throughput = (20.8/0.13)/Sqrt(5) = **71.55 Mbits/s**

1. The packet size is 10,000 bits. This is quite high especially for using in an error-prone environment such as a mobile application. If a single packet is lost then we also lose 10,000 bits which will significantly impact on the streaming of the video. This could lead to frames getting lost, which can cause the video to stutter instead of flowing smoothly which is desired. If the lost frames are I Frames then this may not be as noticeable as there are no dependencies between I Frames, so once the next frame appears this error will disappear. However if this error occurs in a P Frame then this error may propagate between frames and will spread around the frames due to motion vectors. This will cause a very bad visual quality which will deteriorate greatly until a new refresh frame arrives. In error-prone environments it is better to use smaller packet sizes so that if a packet is lost then minimal data is lost as well.

In communications point of view larger packet size will have a major impact on receiver quality. Difficult to conceal the lost information when the packet size is larger under high PLR.

1. As the proposed network sends the majority of its data through fibre optic, it would dramatically improve the throughput if the QAM was increased to 128 QAM or 256 QAM, this would increase the bits per symbol from 4 to 7-8 bits per symbol.

Channel coding is used for detecting errors during transmission, which is needed in high interference environments as errors are more likely. If we increase the channel coding rate we will reduce the amount of redundant data we are sending, and increase the likely hood of detecting errors. We should increase the channel coding from 2/3 to 3/4 to make it more efficient.

Increasing the video compression rate will mean that the raw data size is reduced. Ideally we want to optimise lossless compression as much as possible as this will reduce the data without affecting the video quality. Lossy compression can also be used if the video quality is less important.

If we change to a wireless communication channel then we should make the following modifications:

* 16 QAM when conditions are good, 8 QAM when conditions are poor
* Increase channel coding rate
* Increase video compression rate

As the QAM Modulation Order increases the bit rate/throughput increases but it also reduces resistance to errors. Many over-the-air systems (satellite, cellular) dynamically adapt modulation order based on channel conditions, so that when conditions are good and interference is low then we use a higher order, and fall back to lower order when conditions are poor. 16 QAM is commonly used for satellite channel communication but in order to prevent errors 8 QAM could be used when conditions are poor.

As a wireless environment experiences a large amount of noise and interference we should use a higher channel coding rate. This will make it easier to spot and recover from errors.

Increasing the video compression rate will reduce the raw data size and as such increase the chances of real time streaming being supported.

**Part 2**

**Stage 1**

1. Your packet format and an explanation of all the fields.

Our packet is formatted using the following fields: seqnum and acknum which are integers, checksum which is of type byte and data which is an array of bytes.

Seqnum – this stores the sequence number of the packet which is either 0 or 1 and is used by receiver to check if the received packet is the one we are currently waiting for.

Acknum – this stores the acknowledgment number of the packet which is either 0 or 1. The receiver sends this acknum back to sender to indicate whether packet was corrupted or not,. This acknum is the same as the number the receiver is waiting for.

Checksum – this stores the checksum value for the given data, and is used by both the sender and the receiver to check if the data has been corrupted in any way. It is calculated by summing all the bytes in data and then multiplying this by 0xFFFFFFFF

Data – this stores the data that is being contained and sent with the packet. We receive this data from the Network Simulator and add this into a packet when rdt\_send in the sender is called.

1. Explain all situations that can occur when sending packets and how you handle them.

* Packet data is corrupted
* Packet acknum is corrupted
* Packet seqnum is corrupted
* Packet is lost
* Timeout
* Wrong packet received
* Packet data not in order

1. Post screenshots of each situation being handled and include the code fragments that handle them.
2. How you tested your code? (Note: I’m not looking for extensive JUnit tests but tests. to show your protocol works for all situations)

**Stage 2**

1. Explain how you handled pipelining.
2. 2. Explain steps 1-4 from above for pipelining. If there is no change to a specific part just state it is the same