CS 176c
Discussion Section

May 15th

Review Questions

- Reconstruct Table 7.1 for when Victor Video is watching a 4 Mbps video, Facebook Frank is looking at a new 100 Kbyte image every 20 seconds, and Martha Music is listening to 200 kbps audio stream.

	Bit rate	Bytes transferred in 67 min
Facebook Frank	160 kbps	80 Mbytes
Martha Music	128 kbps	64 Mbytes
Victor Video	2 Mbps	1 Gbyte

Table 7.1 ♦ Comparison of bit-rate requirements of three Internet applications

	Bit rate	Bytes transferred in 67 mins
Facebook Frank	40 kbps	20 Mbytes
Martha Music	200 kbps	100 Mbytes
Victor Video	4 Mbps	2 Gbytes

- There are two types of redundancy in video. Describe them, and discuss how they can be exploited for efficient compression.

- Spatial Redundancy
  - It is the redundancy within a given image.
  - Image consists of mostly white space has a high degree of correlation or similarity.
  - We can compress image without much loss in quality
- Temporal Redundancy
  - If two frames of consecutive images are exactly the same (like in a video) we can just skip out on the re-encoding part and specify that the next image is just like the previous.
  - This saves a lot of computation and memory resources

- Suppose an analog audio signal is sampled 16,000 times per second, and each sample is quantized into one of 1024 levels. What would be the resulting bit rate of the PCM digital audio signal?

- 1024 levels -> 10 bits per sample ($2^{10} = 1024$)
- Resulting bit rate = 16,000 x 10 = 160,000 = 160 Kbps

- Multimedia applications can be classified into three categories. Name and describe each category.

- Streaming stored audio or video
  - Prerecorded media.
  - Users send requests to servers to view the media.
  - Ex. Netflix, YouTube
- Conversational VoIP
  - Real time voice/video transmission from one user to other
  - Ex. Skype, FaceTime
- Streaming live audio and video
  - Users can receive live media over the internet
  - Ex. YouTube live, Facebook live

- Streaming video systems can be classified into three categories. Name and briefly describe each of these categories.

- UDP Streaming
  - The server transmits video at a rate that matches the client’s video consumption rate by clocking out the video chunks over UDP at a steady rate.
- HTTP Streaming
  - The media is stored on an HTTP server (accessed through an URL)
  - User establishes TCP connection with the server and issues GET request
  - Server sends back the media within the response message
- Adaptive HTTP Streaming (DASH)
  - The media is encoded in multiple versions, with different bitrate & quality level
  - The type of bitrate transmitted depends on the available bandwidth of the user at that point of time
  - This is dynamic in nature and adapts to changing bandwidth

- List three disadvantages of UDP streaming.

- Due to unpredictable and varying amount of available bandwidth between server and client, constant-rate UDP streaming can fail to provide continuous play out.
- It requires a media control server, such as an RTSP server, to process client-to-server interactivity requests and to track client state for each ongoing client session.
- Many firewalls are configured to block UDP traffic, preventing users behind these firewalls from receiving UDP video.

- With HTTP streaming, are the TCP receive buffer and the client's application buffer the same thing? If not, how do they interact?

- No
- Client -> Application reads bytes from the TCP receive buffer and places the bytes in the client application buffer.

- Consider the simple model for HTTP streaming. Suppose the server sends bits at a constant rate of 2 Mbps and playback begins when 8 million bits have been received. What is the initial buffering delay t_p?

- The initial buffering delay is t_p = Data bits/bandwidth = 4 seconds

- Why is a packet that is received after its scheduled playout time considered lost?

- A packet that arrives after its scheduled play out time cannot be played out.
- Therefore, from the perspective of the application, the packet has been lost.

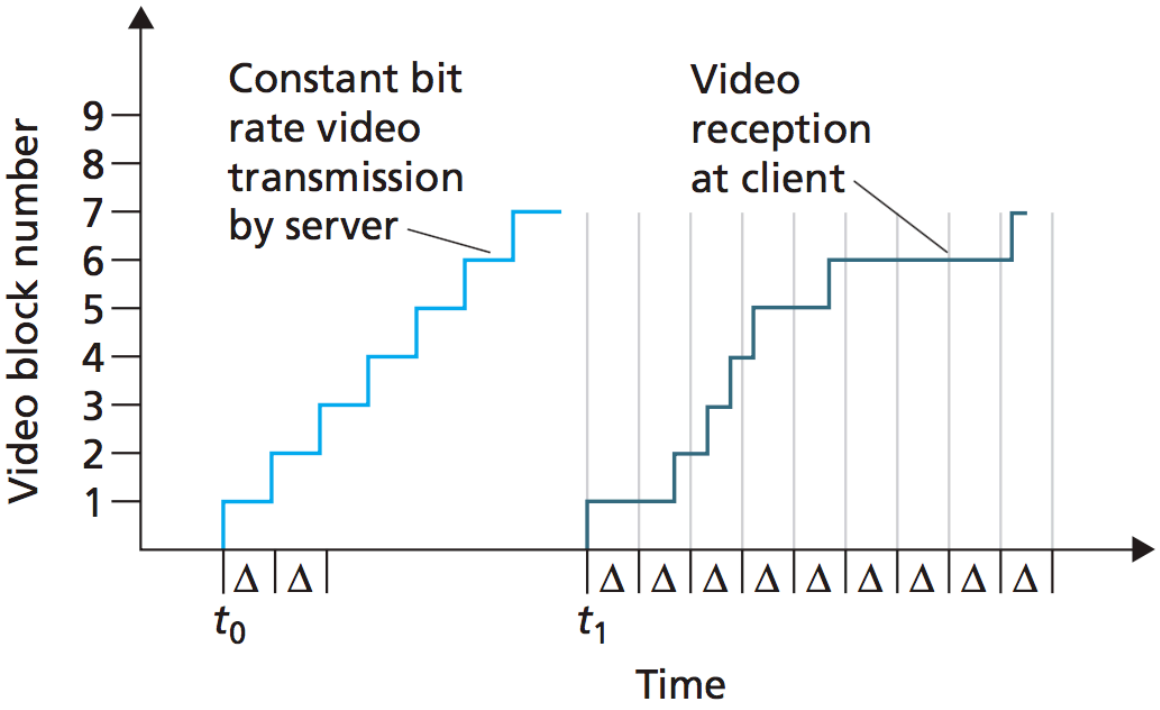
- How are different RTP streams in different sessions identified by a receiver? How are different streams from within the same session identified?

- RTP streams in different sessions
  - Different multicast addresses
- RTP streams in the same session
  - SSRC field
  - RTP packets are distinguished from RTCP packets by using distinct port numbers.

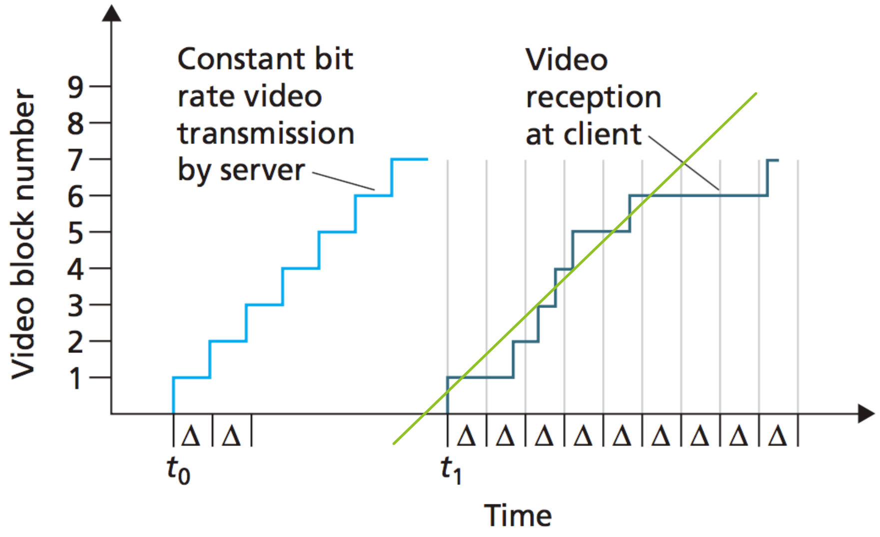
- What is the role of a SIP registrar? How is the role of an SIP registrar different from that of a home agent in Mobile IP?

- SIP Registrar keeps track of the users and their associated IP addresses
- It forwards INVITE messages to the IP addresses to users (in its domain)
- This property is similar to that of authoritative Nameserver in DNS

- Consider the figure below. 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 $t0$, the second block at $t0 + \Delta$, the third block at $t0 + 2\Delta$, and so on. Once the client begins playout, each block should be played out Δ time units after the previous block.



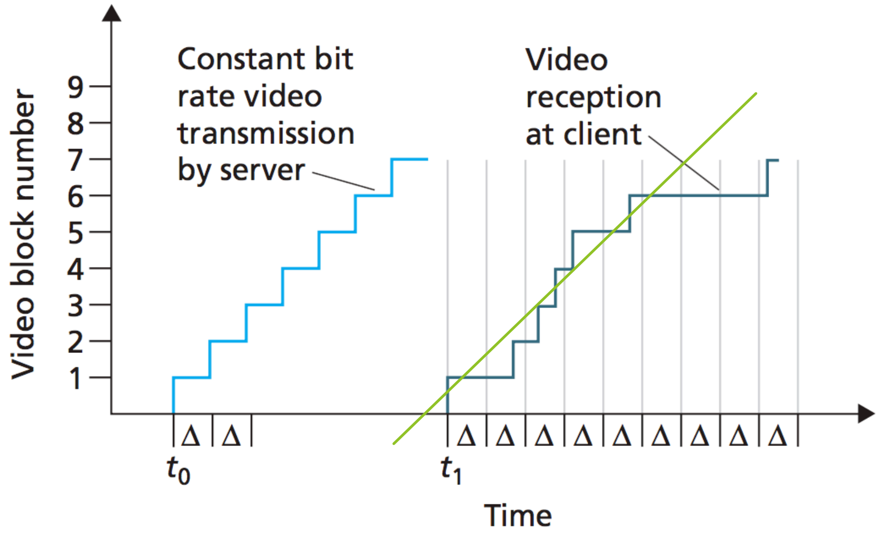
- Suppose that the client begins play out as soon as the first block arrives at $t1$. In the figure, how many blocks of video (including the first block) will have arrived at the client in time for their playout? Explain how you arrived at your answer.
- Video playback timeout: Δ
- So subsequent blocks should arrive before -> $t_1 + \Delta$, $t_1 + 2\Delta$, $t_1 + 3\Delta$ …
- Blocks 1, 4, 5, 6 arrive before their timeout



- Suppose that the client begins playout now at $t1 + \Delta$. How many blocks of video (including the first block) will have arrived at the client in time for their playout? Explain how you arrived at your answer.

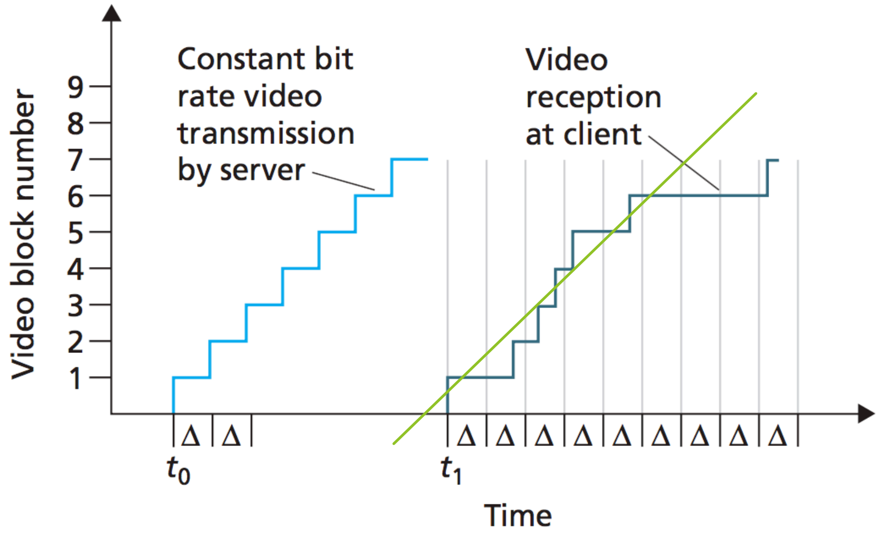
- i^{th} block should arrive before $t_1 + (i+1)\Delta$ time.

- Blocks 1 - 6 before their respective timeouts



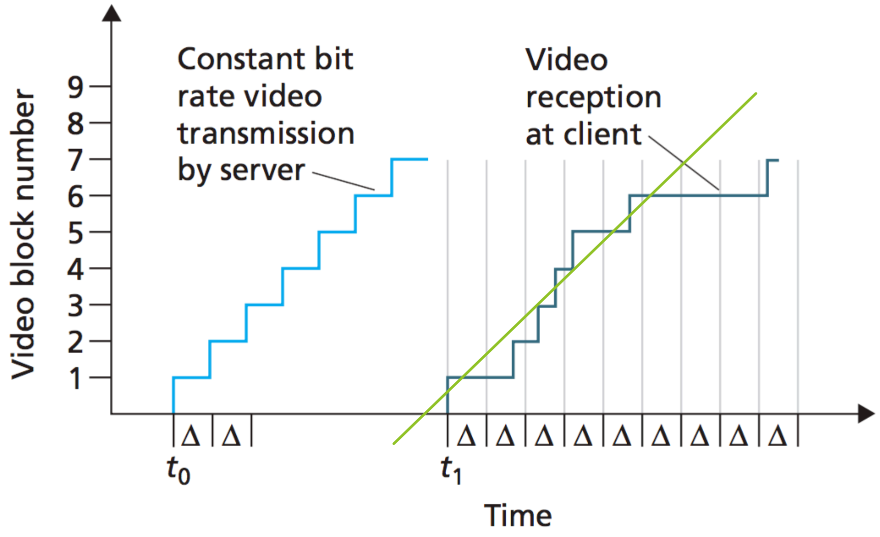
- In the same scenario at (b) above, what is the largest number of blocks that is ever stored in the client buffer, awaiting playout? Explain how you arrived at your answer.

- Max of 2 blocks are stored in the client buffer
- Blocks 3, 4 arrive before $t_1 + 3\Delta$ and after $t_1 + 2\Delta$ -> Store in the client buffer
- Block 5 arrives before $t_1 + 4\Delta$ and after $t_1 + 3\Delta$ -> Store in the buffer along with block 4



- What is the smallest playout delay at the client, such that every video block has arrived in time for its playout? Explain how you arrived at your answer.

- The smallest playout at the client should be $t_1 + 3\Delta$ to ensure that every block has arrived in time.



- Compare the procedure described in Section 9.3 for estimating average delay with the procedure in Section 3.5 for estimating round-trip time. What do the procedures have in common? How are they different?

- The two procedures are very similar.
  - They both use the same formula
  - Exponentially decreasing weights for past samples

- So what's the difference?
  - The time when the data is sent and when the acknowledgement is received is recorded on the same machine.
  - For the delay estimate, the two values are recorded on different machines.
  - Thus the sample delay can actually be negative. When?