Solutions:

1. Only unicast to each server from server:

To guarantee each client receiver their video in L seconds, assume each second clients request a video from the server, you need channels.

optimization: Dram, batching, piggybacking

1. Batching:

For clients requested the same video in short duration, multicast to each client using multicast. Independent of client arriving rate

Disadvantages:

* Near VOD due to waits of new batch
* Bandwidth of Sharing same stream among different batches not saved

1. Piggybacking:

Stream started different time slowed down/up, sync speed -> merge stream.

Disadvantage: amount of speeding < 5% -> limit efficiency

1. Periodic Broadcast:

* Fixed # of channels for each video
* Video into N segments
* Fixed start point (start playback until first segment) -> length of 1st segment/2 sec to wait
* Fixed delay: wait for d sec for playback
* Needed # of channels: N-th partial sum of harmonic series

Disadvantages:

* Wait for playback -> near VOD
* Insensitive to system load
* Difficult to achieve smooth transition (change used # of channels while not interrupt ongoing transmission)

1. Patching:

Server maintains a queue of (client id, video id) sent by clients

1. Algo:

* When a channel is free, dispatch it
* Select next video to stream on such channel based on chosen policy (e.g. FCFS, MQL, MFQ)
* Init a (Patch ID = null, Regular ID = null) token
* If MC of such video in process:

1. (PID, RID) = (null, Free Channel)
2. Append each (CID, VID = selected video) to client list of Free Channel
3. Send (PID, RID) to each CID with VID of selected video, and remove it from queue
4. Activate MC on the free channel
5. Client receive and play whole video from RID = FreeChannel

If no MC of such video in process:

1. Set PID to Free Channel
2. Set RID to latest free channel set to RID playing such video
3. Pass (PID, RID) to chosen patching algorithm (Greedy or Grace) to decide what portion of video to be played on PID = Free Channel
4. Grace patching may change PID to null or RID to Free Channel, in which case the whole video will be served on RID of free channel (execute whole steps as current MC of such video in process), otherwise:
5. Append each (CID, VID = selected video) to client lists of both channels assigned to PID and RID
6. Send (PID, RID) to each CID with VID of selected video, and remove it from queue
7. Activate MC on the free channel
8. Client receive portion of video from PID and rest from RID, and play portion from PID then RID
9. Greedy and grace determine what to broadcast on free channel when client cache size (B) is not enough to hold the missing amount of time frames (t) from regular channel:

Greedy: patch channel to stream first max(t, length of video - B) seconds of the video and regular channel stream the rest to client cache

Grace: open a new regular channel to play whole video to the client

1. Bandwidth (# of channels):
2. Hierarchical:

Every similar to leveling cache system in computer:

1. Two-level: patch the patch
2. Double Patching: clients share trans from both P and R
3. Fragmented Patching: video cut to segments, regular transmission of each segment always exists with restart rate of , arrival rates change, restart rate changes. Also, it never reaches theoretical limit of bandwidth
4. Stream Merging:

Merge transmissions in different channels based on merging channels of late arrival client to earlier. Use diagram from ref to show this.

1. Distributed:

* Increase number of servers (cluster)
* Chaining clients for p2p video streaming

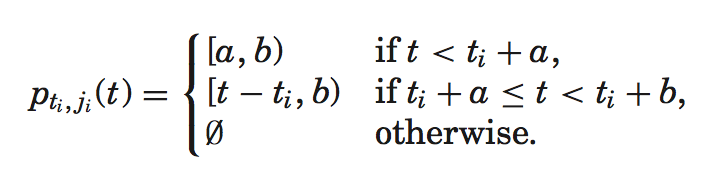
Hierarchical Patching

1. Assumption:

* Clients arrived according to poisson proc
* Limit observation to groups of clients requesting same video
* Server side: maximum 22 channels
* Client side: sufficient cache and channels to receive video

1. Legend:

* Client (c\_i) comes at time (t\_i)
* Video length: L
* Part of video played between time points a and b: p =
* :# of new patches for c\_i at t\_i
* : ji-th patch created for c\_i at t\_i
* is function time t (jointing at t, what available):



* Explanation of formula above:

1. After issuing pt\_i,j\_i, video frames is needed for VOD after is rec and played from other patch channels
2. Recourse vanished as playing through channels (for length of b-a)

* Two conditions guarantee, patches cover whole video but not wasted channel recourse