Video-on-demand broadcasting protocols

Jukka Leveelahti

17.4.2002

Tik-111.590

Multimedia Communications

Motivation

- Watch any movie at home when ever you like
- MPEG-2 at least 4 MB per second
- Too expensive ☹
- Two ways to reduce costs
  - Proactive: broadcasting
  - Reactive: many approaches

Terms

- segment
  - chunk of video, *n* of these in right order make entire video
- consumption rate
  - Processing rate of video in STB
  - *b,* unit of measure for VOD server bandwidth
- slot
  - Time for STB to consume a constant-sized segment
- channel
  - Each stream in VOD server
  - Does not need to be of bandwidth *b*
  - Each video can be distributed over several channels

Client requirements

- when channel bandwidth > *b*
- or STB listens to multiple channels
- => We need local storage

- Size of storage and number of channels are the two things to minimize with clever broadcasting protocols

Staggered broadcasting protocols

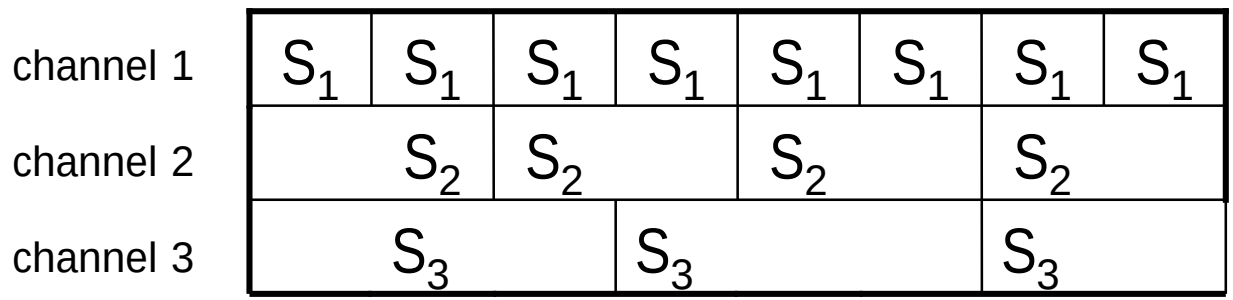
- Starting times for video are staggered evenly across certain n of channels
- video starts at every *D/n* (*D*=duration) mins = phase offset
- Not efficient for server, to cut phase offset double means doubling bandwidth
- Minimal requirements for client
- Can handle interactive VOD
- Example Canal Digital KIOSK

Pyramid broadcasting protocols

- Viswanathan, Imilienski (1995)
- each video is *n* segments, $S_1,..,S_n$
- available bandwidth divided evenly to *n* channels $C_1,..,C_n$
- *i* th segment of each video broadcasted on channel C_i
- Size of segments grow geometrically using parameter ¡Ã 1

Pyramid broadcasting protocols

- client waits for S_1 on channel C_1 and starts consuming
- to receive all the time, receiving S_i must start before S_{i-1} finishes
- client will never experience a break when
- $\alpha = b'/m$, where b' is bandwidth of each channel
- typical α is 2,5



Pyramid broadcasting - performance

- more efficient than Staggered Broadcasting
- client waiting time decreases exponentially with bandwidth
- 2h video with 10*b* bandwidth per video => 12 mins vs. 2 mins
- client requirements are high
- clients have to listen > 1 channels at once
- bandwidth per channel is very high
- requires large storage size
- optimized versions followed..

Permutation-based Pyramid Protocol

- each channel divided into *p* ¡Ã 1 subchannels for each video
- starts of segments evenly staggered on subchannels
- client listens only one subchannel at a time
- need of storage down to third comparing to basic Pyramid Broadcasting
- cost: more bandwidth for same waiting times

Skyscraper Broadcasting Protocol

- replaces geometric series for determining amount of data on each channel
- each video divided into *n* equally sized segments
- number of consecutive segments to place on each channel determined by series

  { 1,2,2,5,5,12,12,25,25,52,52,...}
- Equals of about 1,5
- Each channel requires only bandwidth *b*, can use much more channels
- Width of channel is constrained, no need of storage to store the last (large) block in last channel

Skycraper Broadcasting Protocol

- 1998 improvements: dynamical scheduling of channels and more efficient segment-to-channel series
- in total, low transfer rates and storage needs while reducing also waiting times (found in Pyramid Broadcasting)
- low transfer rate at client causes waste of bandwidth in server

Fast Broadcasting Protocol (1997)

- opposite approach to Skyscraper Broadcasting
- Series is

  {1,2,4,6,8,16,32,64,..}
- Very low waiting times
- Clients receive all data from all channels at once, leads to high transfer rate and high need of storage (up to half of video length)

Pagoda Broadcasting Protocol (1999)

- goal to broadcast segments infrequently while maintaining even transfer rate to client
- uses series like predecessors

  {1,3,5,15,25,75,125,..}
- Big difference: segments don't need to be consecutive on channels
- Uses pairs of channels when assigning segments

Slot	0	1	2	3	4	5
Channel 1	S_1	S_1	S_1	S_1	S_1	S_1
Channel 2	S_2	S_4	S_2	S_5	S_2	S_4
Channel 3	S_3	S_6	S_8	S_3	S_7	S_9

Pagoda Broadcasting Protocol

- client waits for instance of S_1 on channel C_1
- while consuming S_1 starts receiving from every other channel dedicated to that video
- each segment S_i is broadcasted at least once every i slots of time
- client will have the segment ahead in buffer or receive directly from server when needed
- Still requires storage for about half of video

- Notice: pyramid protocols don't work with interactive VOD

Harmonic broadcasting protocols

- first Juhn & Tseng (1997)
- Each video divided into *n* equally sized segments $S_1,..,S_i$
- These are continuously broadcasted in their own channels
- S_i is broadcasted in channel C_i with bandwidth *b/i*
- Sum of channel bandwidths is

$$\sum_{i=1}^{n} \frac{b}{i} = b \sum_{i=1}^{n} \frac{b}{i} = bH(n)$$

- *H(n)* is the harmonic number of *n*, hence the name

Harmonic broadcasting protocols

- series grows very slowly
- can use hundreds of segments without not much bandwidth
- example: with 5*b* 1,5 mins for 2h video
- local storage is needed about 37% of the video
- contains a bug
- fixed with Delayed Harmonic Broadcasting Protocol
- with twice the waiting time..

Harmonic broadcasting protocols

- 1998 three variations
- Cautious Harmonic Broadcasting Protocol
- C_1 not changed, C_2 alternates S_2 and S_3
- C_i from 3 to n, broadcasts S_{i+1} at bandwidth b/i
- $b/2$ more bandwidth than Delayed Harmonic Protocol but waiting time only 1 slot
- Quasi-harmonic Broadcasting Protocol
- segments are divided into fragments which are not broadcast in order
- waiting time still 1 slot, bandwidth converges to $bH(n)$ as n of subsegments increases

Harmonic broadcasting protocols

- Polyharmonic Broadcasting
- forces client to wait *m* slots before consuming
- clients can receive while waiting, segments can be brodcasted with lower bandwidth (compared to Harmonic Broadcasting)
- can use *m* times as many segments, waiting time does not increase
- uses less bandwidth for a given waiting time than Quasi-harmonic Broadcasting
- no interactive VOD with Harmonic Broadcasting

Summary - VOD server

- different protocols share the same strategy: if some videos are more popular than others and clients have local storage, then later parts of video can be broadcasted not as often as the earlier parts
- protocols can save bandwidth on VOD server
- and allow more videos
- or allow server to be cheaper

Summary - client requirements

Broadcasting Protocol	Storage requirement (% of video)	Bandwidth requirement (multiples of *b*)
Staggered	0	1
Pyramid	75	$4\text{-}5^{\text{alpha}}$
Permutation-based	20	2-3
Skyscraper	10	2
Fast	50	6-8
Pagoda	45	5-7
Harmonic	40	5-6

Summary

- Comparing server and client requirements ther is no clear winner
- For example: Polyharmonic has lowest bandwidth requirements on server, but too many data channels per video
- Pagoda is easy on server also, but client bandwidth too high
- Staggered Broadcasting still only one for interactive VOD and no extra load on client

Open questions and research

- interactive VOD
- protocols assume fixed bit rate, not the case with MPEG
- changes in video popularity are difficult to handle
- Staggered model still the easiest

Questions?

- Thanks!