6.3: Broadcast discussion

Streaming Popular Content

- Consider a popular media file
  - Playback rate: 1 Mbps
  - Duration: 90 minutes
  - Request rate: once every minute
- How can a video server handle such high loads?
  - Approach 1: Start a new "stream" for each request
  - Allocate server and disk I/O bandwidth for each request
  - Bandwidth required at server= 1 Mbps x 90

Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes
- Assume
  - All client requests that arrive before the start of a broadcast/multicast can be served by the broadcast/multicast

Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes

- Assume
  - All client requests that arrive before the start of a broadcast/multicast can be served by the broadcast/multicast

- An optimal batching protocol (and analysis)???
  - Guarantee: Maximum startup delay D
  - Can we define the protocol?

Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes

- Assume
  - All client requests that arrive before the start of a broadcast/multicast can be served by the broadcast/multicast

- An optimal batching protocol (and analysis)???
  - Guarantee: Maximum startup delay D
  - Can we define the protocol? –YES (Send as late as possible …)

Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes

- Assume
  - All client requests that arrive before the start of a broadcast/multicast can be served by the broadcast/multicast

- An optimal batching protocol (and analysis)???
  - Guarantee: Maximum startup delay D
  - Can we define the protocol? –YES (Send as late as possible …)
  - Can we calculate the average bandwidth (B), average waiting time (A) before playback, and average number in system??

Discussed in class

- Optimal batching protocol
  - Max delay = D
- Poisson process
  - Inter-arrival times (i) exponentially distributed and (ii) independent
  - Memory less arrival process

Discussed in class

- Optimal batching protocol
  - Max delay = D
- Poisson process
  - Inter-arrival times (i) exponentially distributed and (ii) independent
  - Memory less arrival process
- Renewal process
  - Identify and analyze "renewal periods" (statistically the same)
  - B = ??

Discussed in class

- Optimal batching protocol
  - Max delay = D
- Poisson process
  - Inter-arrival times (i) exponentially distributed and (ii) independent
  - Memory less arrival process
- Renewal process
  - Identify and analyze "renewal periods" (statistically the same)
  - $B = L / (D+1/\lambda)$

Discussed in class

- Optimal batching protocol
  - Max delay = D
- Poisson process
  - Inter-arrival times (i) exponentially distributed and (ii) independent
  - Memory less arrival process
- Renewal process
  - Identify and analyze "renewal periods" (statistically the same)
  - $B = L / (D+1/\lambda)$
- Poisson Arrivals See Time Average (PASTA) property
  - A = ??

Discussed in class

- Optimal batching protocol
  - Max delay = D
- Poisson process
  - Inter-arrival times (i) exponentially distributed and (ii) independent
  - Memory less arrival process
- Renewal process
  - Identify and analyze "renewal periods" (statistically the same)
  - $B = L / (D+1/\lambda)$
- Poisson Arrivals See Time Average (PASTA) property
  - $A = [D(1+\lambda D/2)]/[1+\lambda D]$

Discussed in class

- Little's law
  - # in system = (arrival rate into system) x (average time in system)
- Systems considered where
  - System = "waiting queue"
    - E[# in system] = ??
  - System = "queue or being served"
    - E[#in system] = ??

Discussed in class

- Little's law
  - # in system = (arrival rate into system) x (average time in system)
- Systems considered where
  - System = "waiting queue"
    - Average time in system = A
    - Arrival rate = λ
    - E[# in system] = $\lambda [D(1+\lambda D/2)]/[1+\lambda D]$
  - System = "queue or being served"
    - Average time in system A+L/r
    - Arrival rate = λ
    - E[#in system] = $\lambda [D(1+\lambda D/2)]/[1+\lambda D] + \lambda L/r$

Generalizations and additional details of these optimized protocols in:
"Multicast Protocols for Scalable On-demand Download, Performance Evaluation, Oct. 2006.
"Server Selection in Large-scale Video-on-Demand Systems", ACM TOMCAPP, Feb. 2010.

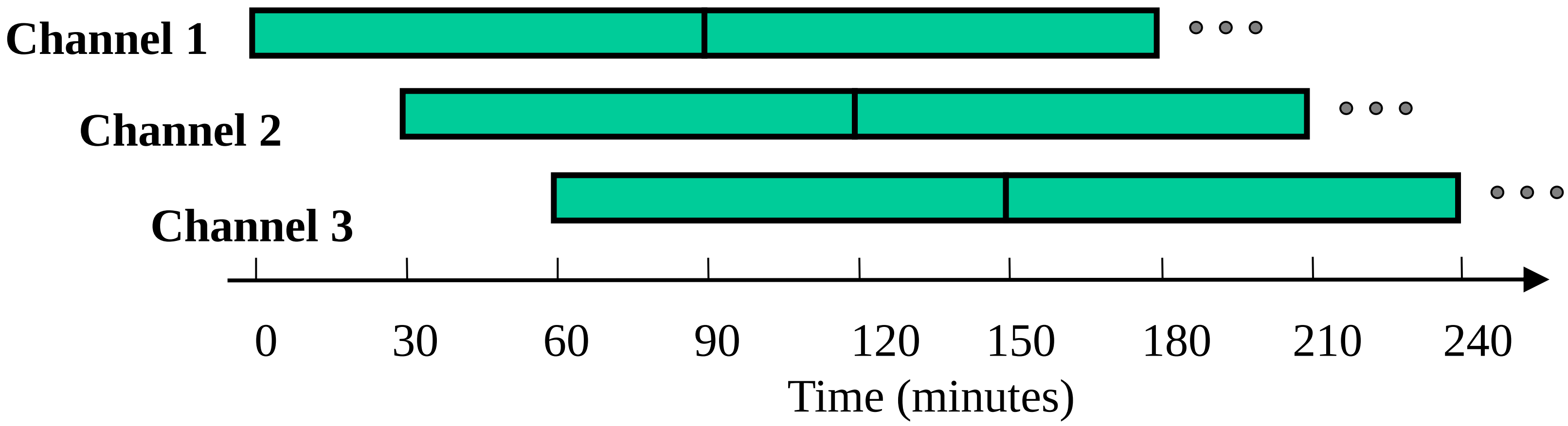
Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery (e.g., multicast/broadcast) capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes

- Consider the case without explicit requests, and scheduling of broadcast/multicast channels must be done apriori …

Streaming Popular Content using Batching

- Consider case of high request rate and D=30min…
  - Max. start-up delay = 30 minutes
  - Group requests in non-overlapping intervals of 30 min
  - Bandwidth required = 3 channels = 3 Mbps



Batching Issues

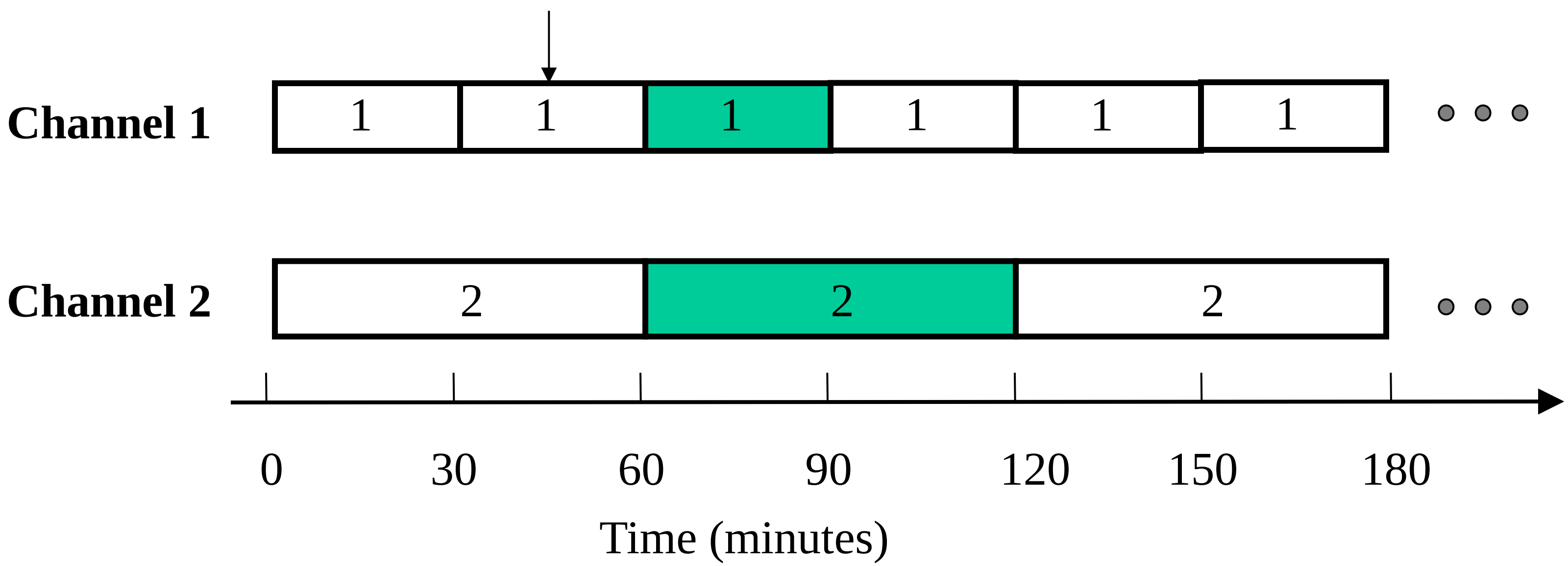
- Bandwidth increases linearly with decrease in start-up delays

- Can we reduce or eliminate "start-up" delays?
  - Periodic Broadcast Protocols

  - Stream Merging Protocols

Periodic Broadcast Example

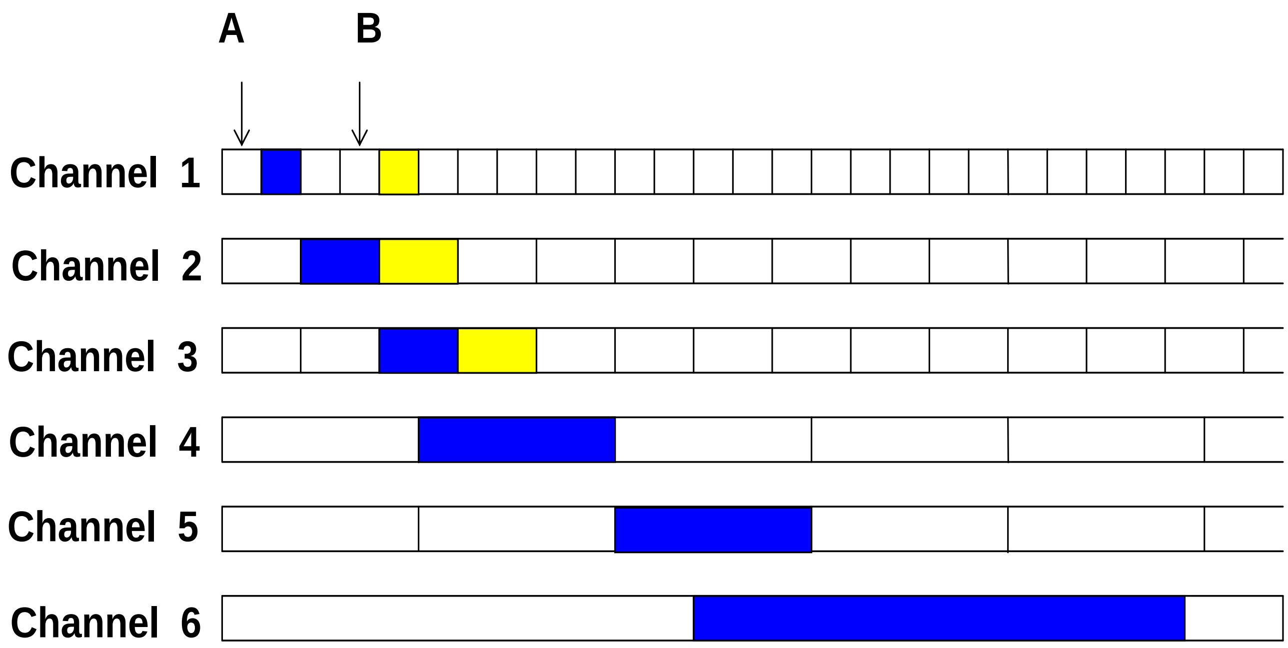
- Partition the media file into 2 segments with relative sizes {1, 2}. For a 90 min. movie:
  - Segment 1 = 30 minutes, Segment 2 = 60 minutes
- Advantage:
  - Max. start-up delay = 30 minutes
  - Bandwidth required = 2 channels = 2 Mbps
- Disadvantage: Requires increased client capabilities



Skyscraper Broadcasts (SB)

[Hua & Sheu 1997]

- Divide the file into *K* segments of increasing size
  - Segment size progression: 1, 2, 2, 5, 5, 12, 12, 25, …
- Multicast each segment on a separate channel at the playback rate
- Aggregate rate to clients: *2 x playback rate*



Comparing Batching and SB

Server Bandwidth	Start-up Delay	
	Batching	SB
1 Mbps	90 minutes	90 minutes
2 Mbps	45 minutes	30 minutes
6 Mbps	15 minutes	3 minutes
10 Mbps	9 minutes	30 seconds

- Playback rate = 1 Mbps, duration = 90 minutes
- Limitations of Skyscraper:
  - Ad hoc segment size progress
  - Does not work for low client data rates

Reliable Periodic Broadcasts (RPB)

[Mahanti *et al*. 2001, 2003]

- Optimized PB protocols (no packet loss recovery)
  - client fully downloads each segment before playing
  - required server bandwidth near minimal
  - Segment size progression is *not* ad hoc
  - Works for client data rates < 2 x playback rate
- extend for packet loss recovery
- extend for “bursty” packet loss
- extend for client heterogeneity

Generalizations and additional details in:
"Scalable On-Demand Media Streaming with Packet Loss Recovery", IEEE/ACM Trans. on Netw., 2003.
"Optimized Periodic Broadcast of Nonlinear Media", IEEE Trans. on Multimedia, 2008.
"Scalable On-demand Media Streaming for Heterogeneous Clients", ACM TOMCAPP, 2008.
"Content Delivery using Replicated Digital Fountains", Proc. IEEE/ACM MASCOTS, 2010.

Reliable Periodic Broadcasts (RPB)

[Mahanti *et al*. 2001, 2003]

- Optimized PB protocols (no packet loss recovery)
  - client fully downloads each segment before playing
  - required server bandwidth near minimal
  - Segment size progression is *not* ad hoc
  - Works for client data rates < 2 x playback rate
- extend for packet loss recovery
- extend for "bursty" packet loss
- extend for client heterogeneity

Generalizations and additional details in:

"Scalable On-Demand Media Streaming with Packet Loss Recovery", IEEE/ACM Trans. on Netw., 2003.
"Optimized Periodic Broadcast of Nonlinear Media", IEEE Trans. on Multimedia, 2008.
"Scalable On-demand Media Streaming for Heterogeneous Clients", ACM TOMCAPP, 2008.
"Content Delivery using Replicated Digital Fountains", Proc. IEEE/ACM MASCOTS, 2010.

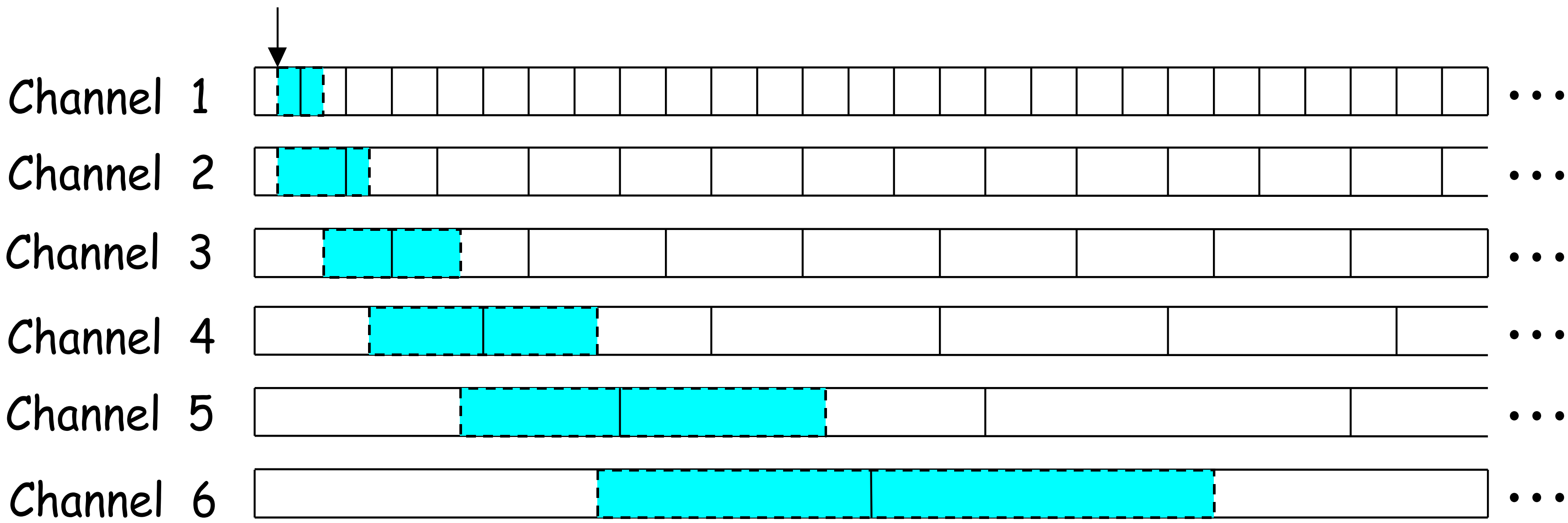
Optimized Periodic Broadcasts

- Playback rate assumed equal to 1
- r = segment streaming rate
- s = maximum # streams client listens to concurrently
- b = client data rate = s x r

Optimized Periodic Broadcasts

- Playback rate assumed equal to 1
- r = segment streaming rate
- s = maximum # streams client listens to concurrently
- b = client data rate = s x r

Optimized Periodic Broadcasts



- Playback rate assumed equal to 1
- r = segment streaming rate = 1
- s = maximum # streams client listens to concurrently = 2
- b = client data rate = s x r = 2

- length of first s segments: $\frac{1}{r} l_k = \frac{1}{r} l_1 + \sum_{j=1}^{k-1} l_j$

- length of segment k > s: $\frac{1}{r} l_k = \sum_{j=k-s}^{k-1} l_j$