Improving VoD System Efficiency with Multicast and Caching

Jack Yiu-bun Lee
Department of Information Engineering
The Chinese University of Hong Kong

Contents

1. Introduction

Jack Y.B. Lee

- VoD technologies have been available for many years, why VoD services are still not popular?
  - It's expensive and not economically viable.
- How can cost be reduced?
  - By evolution of faster computer hardware, higher bandwidth network for the same price.
  - By taking advantage of economy of scales, i.e. using commodity hardware platforms like the PC.
    - E.g. parallel servers.
  - By intelligent ways of reducing the system requirement.
    - E.g. batching, caching, and piggybacking.

1. Introduction

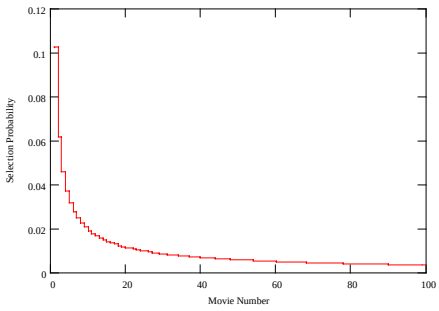
Jack Y.B. Lee

- Observation
  - In real-world applications, a large proportion of VoD users watch only a small number of popular movies.
  - Studies from traditional video rental services show that the movie popularity is Zipf distributed:

$$q_i = \frac{f_i}{\sum_{j=1}^{n} f_j}, \text{ for } i = 1 \ldots n$$

where $f_i = 1/i^{1-\theta}$

with $\theta = 0.271$.



1. Introduction

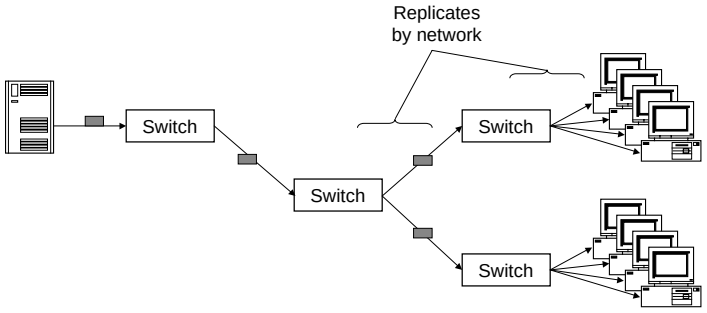
Jack Y.B. Lee

- Motivation
  - The movie popularity is highly skewed.
  - Many users are likely to watch the same movies.
  - Why not let the users share *them*?
- Share What?
  - Server
    - Share retrieved video data at the server by caching. [Freedman *et al.* 1995]
  - Network
    - Share transmitted video data by multicasting. [Dan *et al.* 1994, Li *et al.* 1996, Shachnai *et al.* 1997, etc.]
  - Client
    - Share received video data by buffering. [Sheu *et al.* 1997, Ma *et al.* 1997]

2. Previous Works

Jack Y.B. Lee

- Multicasting Video
  - Transmitting a video by multicast enables the system to serve more than one viewers using only one-channel's worth of resources at the server and part of the network.



2. Previous Works

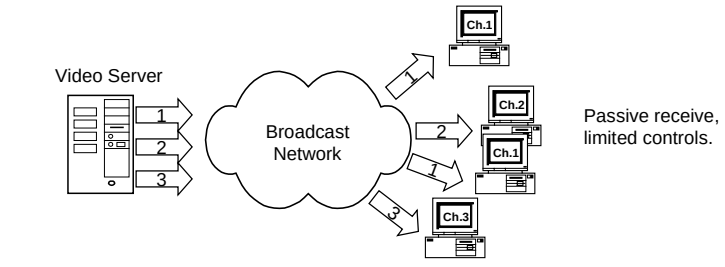
Jack Y.B. Lee

- Problems
  - Video playback of different clients are unlikely to be synchronized.
  - Hence simply sharing a multicast video session among clients arriving at roughly the same time isn't going to be very effective.
  - Interactive VCR operations cannot be supported.
- Possible Solutions
  - Tradeoff Delay (e.g. NVoD, Batching)
  - Tradeoff Buffer (e.g. Split and Merge)
  - Tradeoff Quality (e.g. Piggybacking)
  - Any combinations of the above.

2. Previous Works

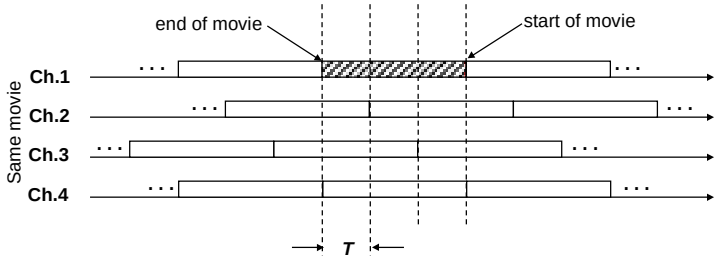
Jack Y.B. Lee

- Near-Video-on-Demand:



2. Previous Works

- Near-Video-on-Demand:



If movie length is ***L*** then number of channels needed per movie is: ***N = L / T***

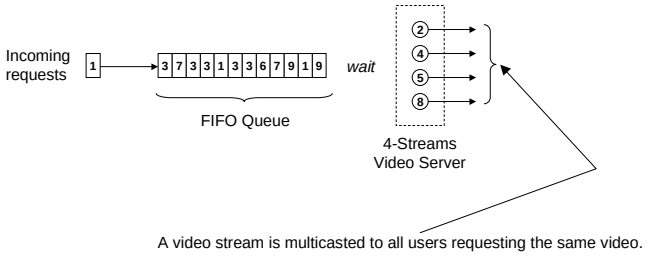
For example, if L = 120 minutes, T = 10 minutes,
then number of video channels needed N = 120 / 10 = 12 channels.

This also means that in the worst case, the user has to wait 10 minutes before viewing a movie.

System response time inversely proportional to number of required channels.

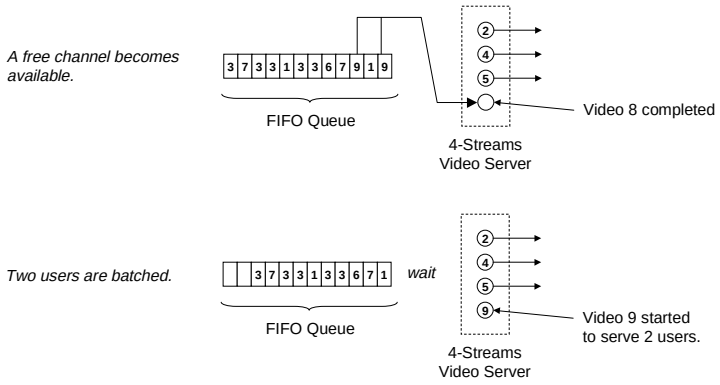
2. Previous Works

- Batching [Dan *et al*. 1994]
  - Serve *waiting users* requesting the same movie using a single multicasted video stream.
  - First-Come-First-Serve (FCFS) Scheduling
    - Incoming requests are queued:



2. Previous Works

- Batching [Dan *et al*. 1994]
  - First-Come-First-Serve (FCFS) Scheduling
    - The HOL request and all requests for the same video title are then served when a channel becomes available.



2. Previous Works

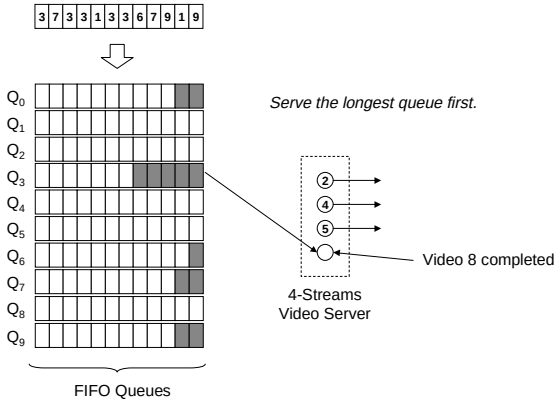
- Batching [Dan *et al*. 1994]
  - First-Come-First-Serve (FCFS) Scheduling
    - Advantage
      - Fairness (unpopular videos will not be denied service)
    - Disadvantage
      - Does not consider batching efficiency.
      - Example
        - FCFS assigns the available channel to video 9 with 2 waiting users while there are 5 users waiting for video 3.

          3 7 3 3 1 3 3 6 7 9 1 9

        - In terms of batching efficiency, the system should serve video 3 instead of video 9.

2. Previous Works

- Batching [Dan *et al*. 1994]
  - Maximal Queue Length (MQL) Scheduling
    - One FIFO queue per video title.



2. Previous Works

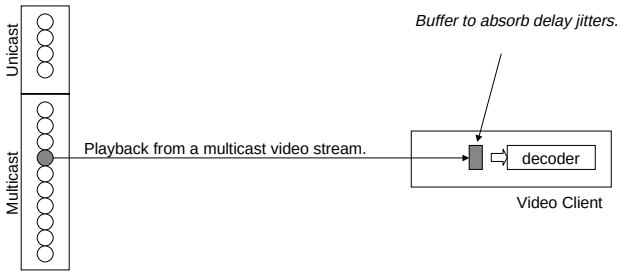
- Batching [Dan *et al*. 1994]
  - Maximal Queue Length (MQL) Scheduling
    - Advantage
      - Improved batching efficiency.
    - Disadvantage
      - No consideration for waiting time and fairness;
      - Users may leave the queue (turned away) if the waiting time is too long.
  - Results
    - Bandwidth reduction of ~70% with average response time of 2~3 minutes.
    - VCR operation is not supported.
    - Batching is efficient only at high loads, where there are sufficient number of queuing users for effective batching.

2. Previous Works

- Batching with Bridging [Li *et al*. 1997, etc.]
  - Motivation
    - To support interactive operations under batching.
  - Principles
    - Absorb the playback time differences by buffering at the client; or at an intermediate node.
    - Some capacity of the video server is for multicast, while the rest is for unicast.
    - The unicast channels are used to fill the gap between the time difference of the multicasted stream and the requested video stream.

2. Previous Works

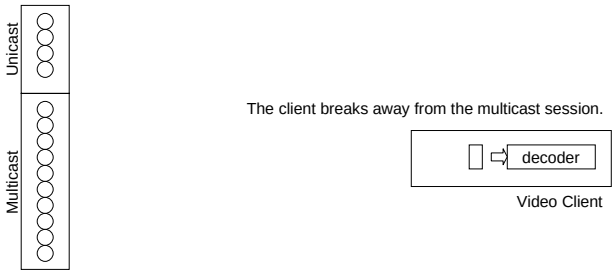
- Batching with Bridging [Li *et al*. 1997, etc.]
  - Normal Playback:



2. Previous Works

Jack Y.B. Lee

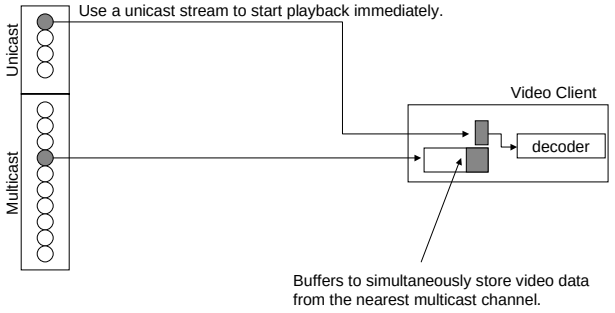
- Batching with Bridging [Li *et al.* 1997, etc.]
  - Initiates interactive control (e.g. pause):



2. Previous Works

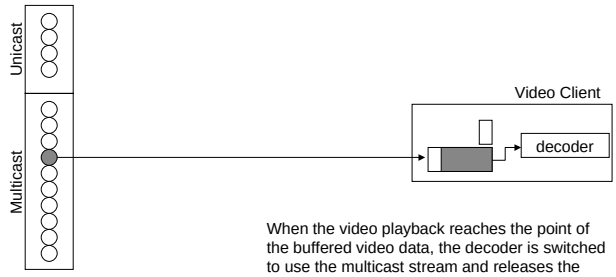
Jack Y.B. Lee

- Batching with Bridging [Li *et al.* 1997, etc.]
  - Resumes normal playback using a unicast channel:



2. Previous Works

- Batching with Bridging [Li *et al.* 1997, etc.]
  - Continues normal playback from a multicast channel:



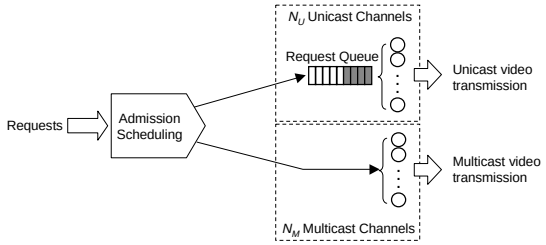
When the video playback reaches the point of the buffered video data, the decoder is switched to use the multicast stream and releases the unicast stream.

3. UVoD

- Motivation
  - TVoD performs well only at light loads while NVoD performs well only at heavy loads.
  - Batching incur substantial *startup delay* (in minutes) in order to achieve performance gains.
  - Performance of batching over a short time scale depends heavily on the request arrival pattern.
- Unified VoD Architecture
  - Achieves very significant performance gain (e.g. 500%) even with very low latency (e.g. 2 secs).
  - TVoD and NVoD can be considered special cases of the UVoD architecture.
  - Given the same number of channels, UVoD always achieves lower latency than both TVoD and NVoD.

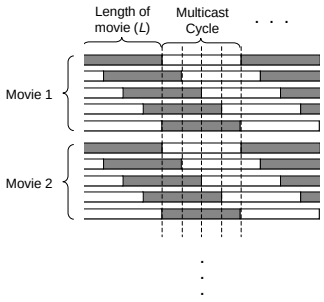
3. UVoD

- Architecture
  - Divides server and network channels into two types: unicast and multicast channels.



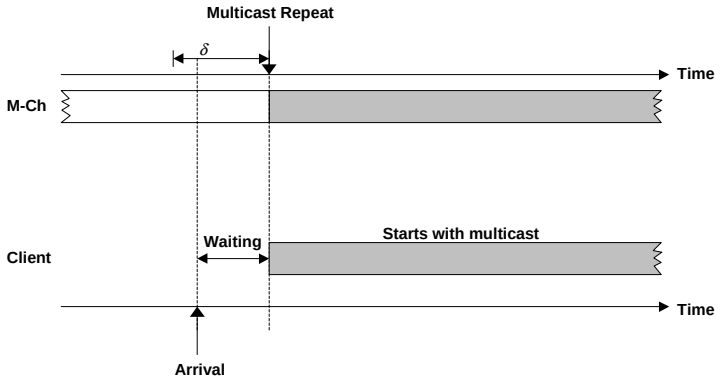
3. UVoD

- Multicast Scheduling
  - Available M-channels are equally divided among movies.



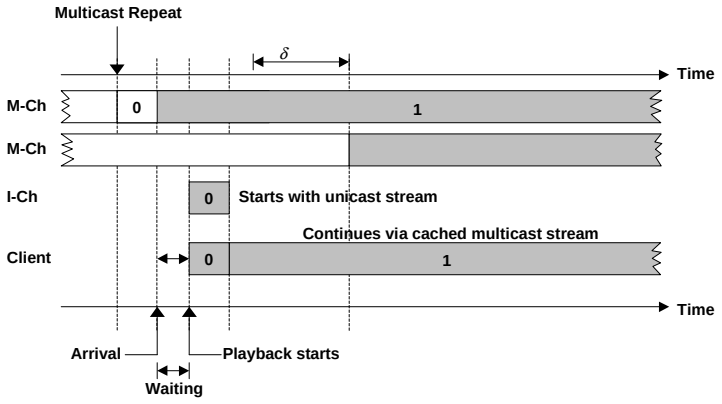
3. UVoD

- Admission Scheduling
  - A new user is admitted to the nearest multicast channel if the delay will be smaller than a given *admission threshold* δ.



3. UVoD

- Admission Scheduling
  - Otherwise the user will start with an I-Channel while simultaneously caching data from a M-Channel.



3. UVoD

- Startup Latency
  - Admission via multicast channel:

$$w_M(\delta) = \frac{\delta}{2}$$

I.e. half the admission threshold (assumes equally probable to arrive at any time within the threshold).

  - Admission via unicast channel:
    - Equals to the waiting time at the unicast channels.
    - Derivations:
      - Probability of admit-via-multicast: $P_m = \frac{\delta}{T_R}$
      - Arrival rate at unicast channels: $\lambda_u = (1 - P_m)\lambda$

3. UVoD

- Startup Latency
  - Admission via unicast channel:
    - Derivations:
      - Range of service time: $0 < s < T_R - \delta$
      - Modeled as G/G/m queue, Allen-Cunneen approximation for average wait:

$$w_U(\delta) \approx \frac{E_C(N_U, u)}{N_U(1-\rho)}\left(\frac{C_A^2 + C_S^2}{2}\right)T_S$$

Which is load-dependent.

3. UVoD

- Startup Latency
  - Observations
    - Latency in general not the same for the two cases.

      $w_M(\delta) \neq w_U(\delta)$

    - Increasing admission threshold diverges more users to the multicast channels, thereby reducing $w_U(\delta)$.
  - Adaptive Admission Scheduling
    - Adjust the admission threshold to maintain a uniform latency.

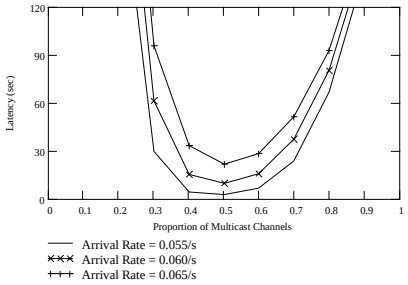
      $\delta = \min\{x \mid (w_M(x) - w_U(x)) \leq \varepsilon, T_R \geq x \geq 0\}$

3. UVoD

- Channel Partitioning
  - Problem
    - How many channels should one reserves for unicast?
  - Intuitions
    - At light loads, more channels should be allocated for unicast to reduce startup latency (i.e. approaches TVoD).
    - At heavy loads, more channels should be allocated for multicast to increase capacity (i.e. approaches NVoD).
  - Special Cases
    - Allocates all channels for unicast, equivalent to TVoD.
    - Allocates all channels for multicast, equivalent to NVoD.
  - Somewhere in between?

3. UVoD

- Channel Partitioning
  - Consider $W_U(\delta)$ versus channel partition policy:



  - Allocating too few I-Channels leads to large latency.
  - Surprisingly, allocating too many I-Channels can be counter-productive as service time will become very long.

3. UVoD

- Channel Partitioning
  - To minimize load at the unicast channels, it can be shown that a near-optimal allocation is given by:

$$N_M^{opt} = \left\langle \frac{LN}{2LM - \delta N} \right\rangle \frac{M}{N}$$

Where N_M^{opt} is the proportion of M-Channels

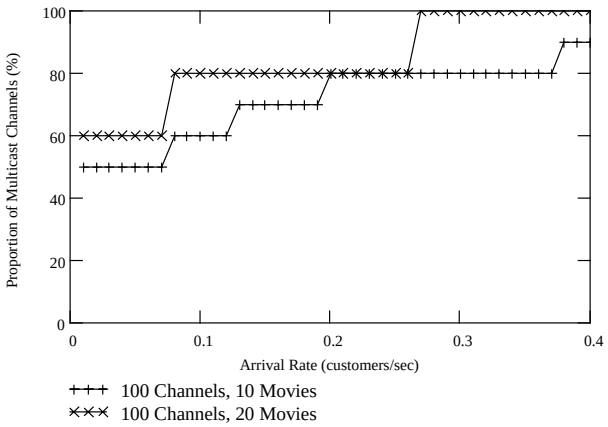
and $\langle\ \rangle$ is the rounding operator.

4. Results

- Assumptions
  - Poisson Arrivals
  - Movie length is 120 minutes.
  - Scenario 1:
    - 100 channels, 10 movies.
  - Scenario 2:
    - 500 channels, 50 movies.
  - No interactive control.

4. Results

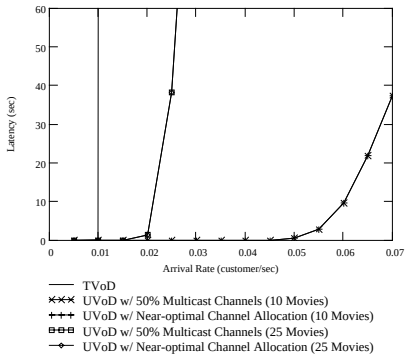
- Channel Partition Policy



4. Results

Jack Y.B. Lee

- Comparison with TVoD and NVoD
  - Light load ranges up to 0.07 (100 channels)

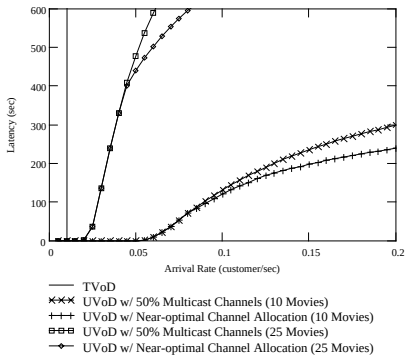


Latency for NVoD is fixed at 360 seconds (10 movies case) regardless of arrival rates.

4. Results

Jack Y.B. Lee

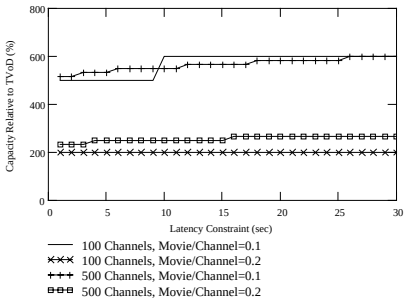
- Comparison with TVoD and NVoD
  - Heavy load ranges up to 0.2 (100 channels)



Latency for NVoD is fixed at 360 seconds (10 movies case) regardless of arrival rates.

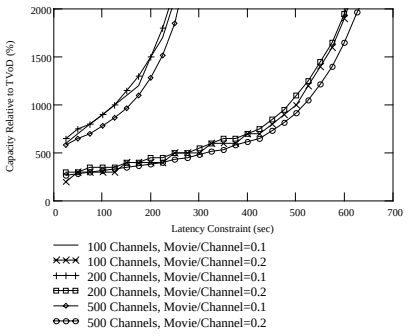
4. Results

- Performance Gain Over TVoD v.s. Latency Constraint
  - Small latency ranges (0~30s):



4. Results

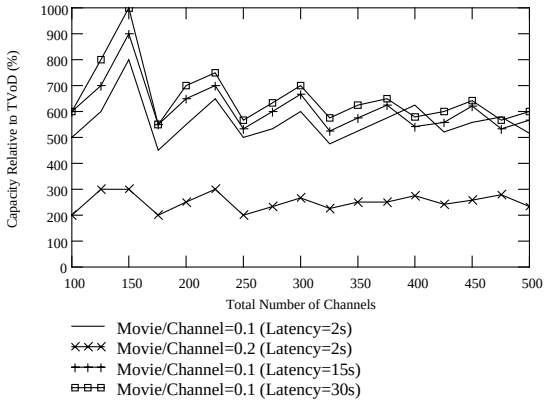
- Performance Gain Over TVoD v.s. Latency Constraint
  - Large latency ranges (0~700s):



Latency for NVoD is fixed at 360 seconds (movie-channel ratio = 0.1) and fixed at 720 seconds (movie-channel ratio = 0.2).

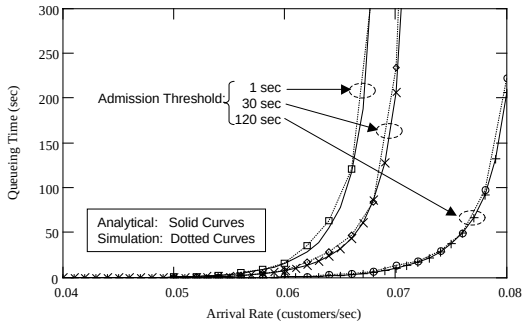
4. Results

- Performance Gain Over TVoD v.s. System Scale



4. Results

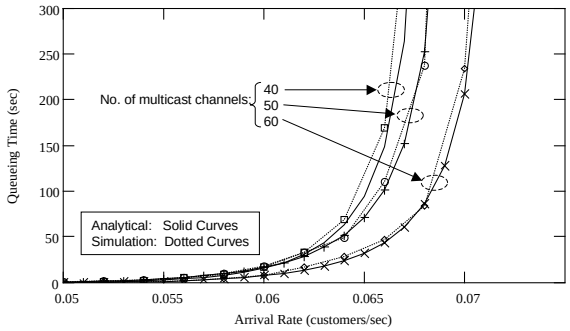
- Model Validation via Simulation
  - Fixed channel partition policy;
  - Three admission threshold settings.



4. Results

Jack Y.B. Lee

- Model Validation via Simulation
  - Fixed admission threshold setting;
  - Three channel partition policies.



5. Interactive Controls

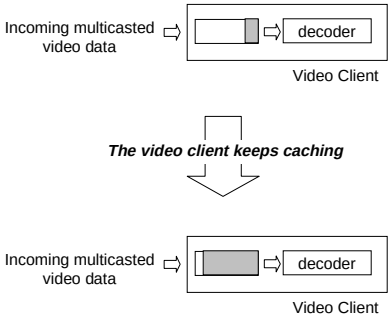
Jack Y.B. Lee

- Supporting Interactive Controls with Unicast
  - Treats users broken away from a multicast channel due to an interactive control as new users.
  - Increasing load at the unicast channels.
  - Performance gain will decrease with increase in rates of interactivity.
- Supporting Pause-Resume with Channel Hopping
  - UVoD employs static multicast scheduling such that repeating intervals are known and fixed.
  - This enables broken-away users to resume a paused video session simply by joining a nearby multicast session.
  - No overhead is incurred at the server and the network.
  - Suitable for movie-on-demand applications.

5. Interactive Controls

Jack Y.B. Lee

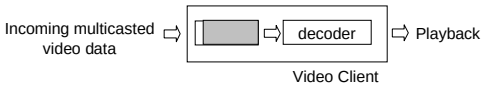
- Channel Hopping
  - User initiates pause:



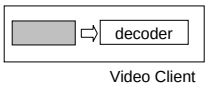
5. Interactive Controls

Jack Y.B. Lee

- Channel Hopping
  - User resumes playback before cache overflow:
    - No-op, just resumes playback via cache.



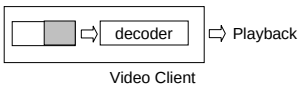
  - When cache is full, then stops caching:



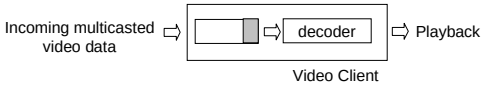
    - Note that the cache contains T_R seconds worth of video.

5. Interactive Controls

- Channel Hopping
  - User resumes playback:
    - Starts playback immediately and wait for the upcoming multicast.



    - Since every point in a movie is repeated every T_R seconds, we guarantee that the next multicast will come before the client runs out of cached video data.



6. Conclusions

- UVoD operates efficiently over a wide range of loads.
- The existing TVoD and NVoD architectures can be considered special cases of UVoD.
- Performs at least as good as and mostly better than TVoD and NVoD.
- Significant performance gain over TVoD (500%) can be achieved even at low latency (1 second).
- Zero overhead for pause-resume interactive control.

References

[1] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling Policies for an On-Demand Video Server with Batching," *Proc. 2nd ACM International Conference on Multimedia*, 1994, pp.15-23.

[2] H. Shachnai and P.S. Yu, "Exploring Waiting Tolerance in Effective Batching for Video-on-Demand Scheduling," *Proc. 8th Israeli Conference on Computer Systems and Software Engineering*, Jun 1997, pp.67-76.

[3] V.O.K. Li, W. Liao, X. Qiu, and E.W.M. Wong, "Performance Model of Interactive Video-on-Demand Systems," *IEEE Journal of Selected Areas in Communications*, vol.14(6), Aug 1996, pp.1099-1109.

[4] W. Liao and V.O.K. Li, "The Split and Merge protocol for interactive video-on-demand," *IEEE Multimedia*, vol.4(4), 1997, pp.51-62.

[5] L. Golubchik, J.C.S. Lui, and R.R. Muntz, "Adaptive piggybacking: a novel technique for data sharing in video-on-demand storage servers," *ACM Multimedia Systems*, vol.4(30), 1996, pp.14-55.

[6] A.O. Allen, *Probability, Statistics, and Queueing Theory with Computer Science Applications*, 2nd Ed. Academic Press, New York, 1990.