Streaming Stored Layered Video in the Internet

Keith Ross

(joint work with Philippe De Cuetos, Despina Saparilla, Jussi Kangasharju, Martin Reisslein)

Institut EURECOM

Sophia Antipolis, France

EURECOM

Streaming Stored Video

- Streaming stored video is becoming increasingly widespread in the Internet
- High-speed access technologies will permit users to stream video at higher rates
- User access rates are highly heterogeneous.
- The Internet is a best-effort network without QoS guarantees.

Issues to be Examined

- Prefetching
- Layered video
- Multiple layers or multiple versions?
- How to cache layered video
- Interactive audio streaming (Wimba startup)

Assume: CBR video; abundant client storage

References

- D. Saparilla and K.W. Ross, Optimal Streaming of Layered Encoded Video, *Infocom 2000*, Tel Aviv.
- D. Saparilla and K.W. Rooss, Streaming Stored Continuous Media over Fair-Share Bandwidth, *NOSSDAV 2000*, Chapel Hill, 2000.
- P. Decuetos, D. Saparilla and K.W. Ross, Adaptive Streaming of Stored Video in a TCP-Friendly Context : Multiple Versions or Multiple Layers, International Packet Video Workshop, Korea, May 2001
- J.Kangasharju, F. Hartanto, M. Reisslein and K.W. Ross, Distributing Layered Encoded Video through Caches, *IEEE Infocom 2001*, Anchorage.
- Wimba: http://www.wimba.com

Outline of the Talk

I Streaming Stored Layered Video over Fair-Share Bandwidth

II Layers vs. Switching Versions over Fair-Share Bandwidth (Philippe)

III Layered Video through Caches

IV Interactive Audio Streaming (Wimba startup)

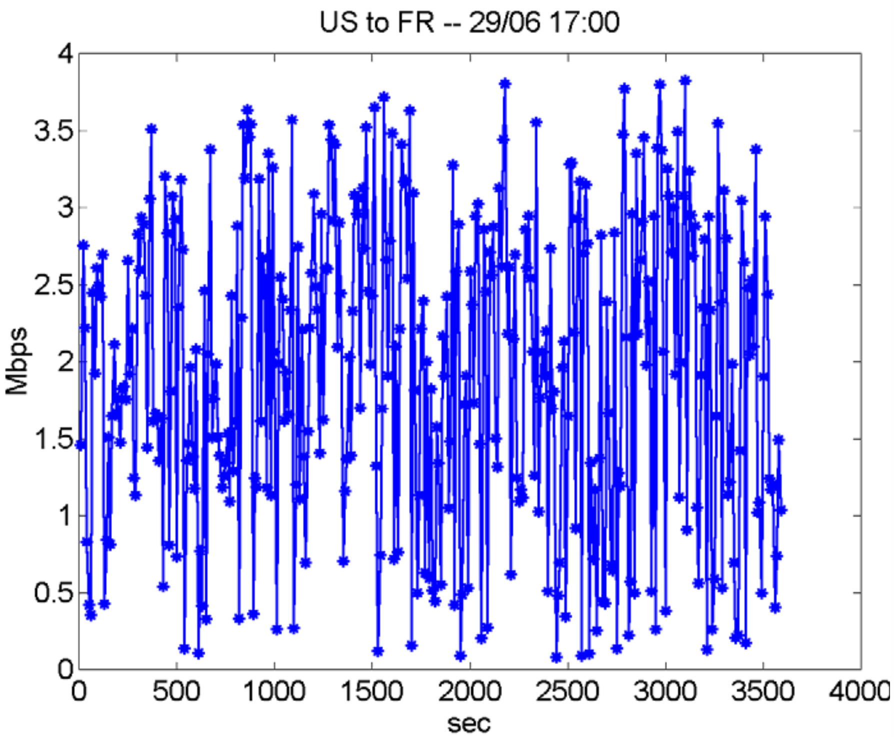
Design Principle: TCP Bandwidth

- A TCP connection roughly gets fair-share bandwidth

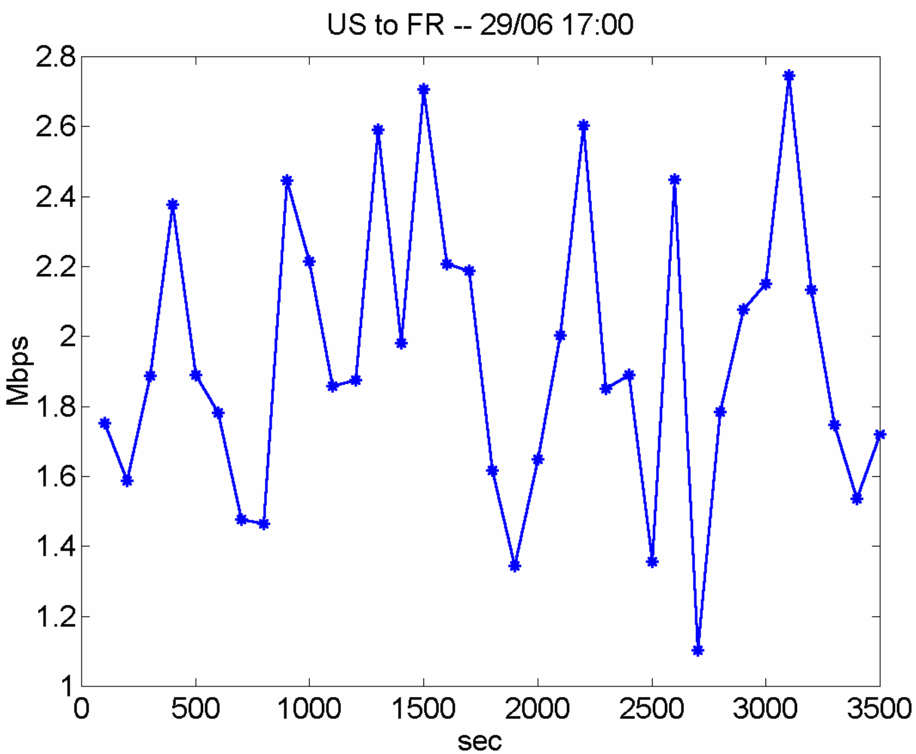
- This bandwidth varies throughout connection.
  - Changes in number of streams.
  - Changes in traffic generated by individual streams
  - Route changes.

- Assumption: Our video streaming application gets TCP bandwidth:
  - sent over HTTP/TCP
  - TCP-friendly connection

TCP Bandwidth



Averaged over 10 second intervals

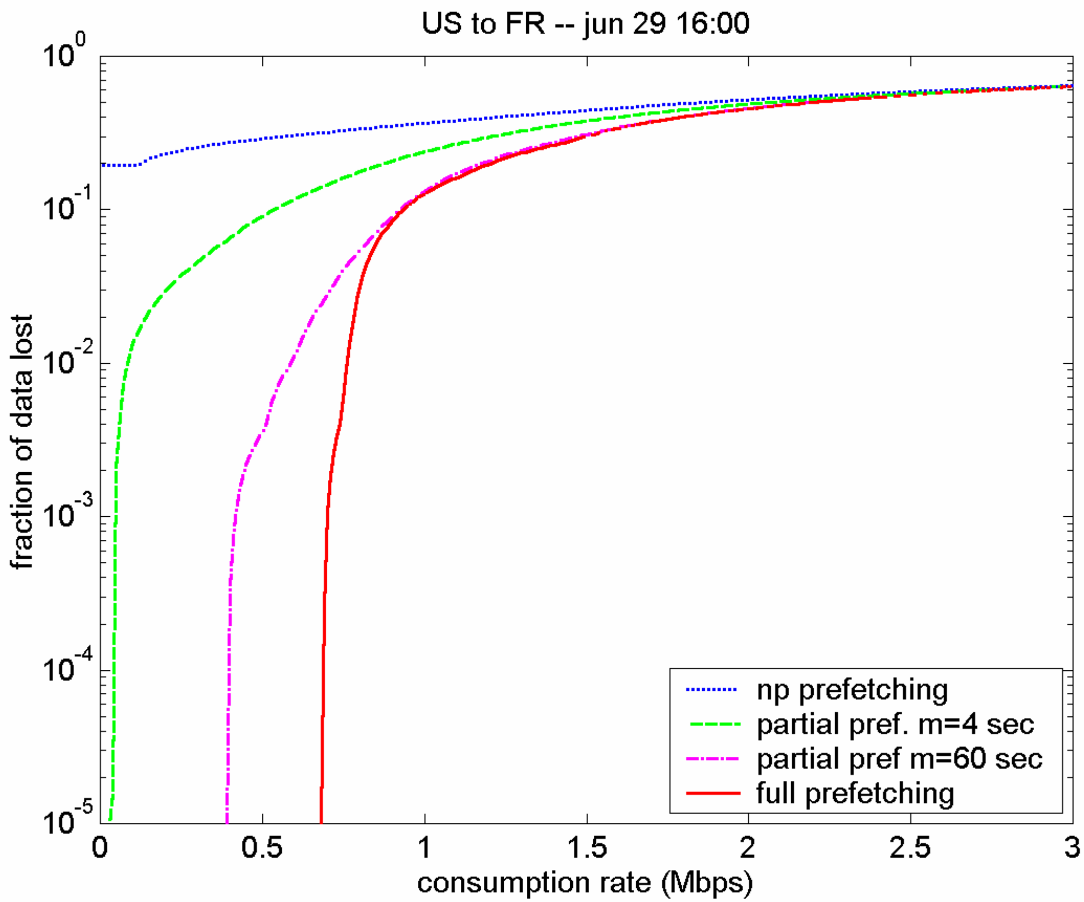


Averaged over 100 second intervals

Streaming Non-Layered CBR Video (1)

- Consider three transmission schemes:
  - Full prefetching: transmit at full available rate
  - No prefetching: transmit at *min{consumption, available rate}*
  - Partial prefetching: prefetch over short intervals
- Playback delay of four seconds; video 60 minutes long.

Streaming Non-Layered CBR Video (2)



Avg available bandwidth = 1.1 Mbps

- Maximum rate for no loss with full prefetching: 70% of average available rate
- Maximum buffer for no loss with full prefetching: 15 minutes

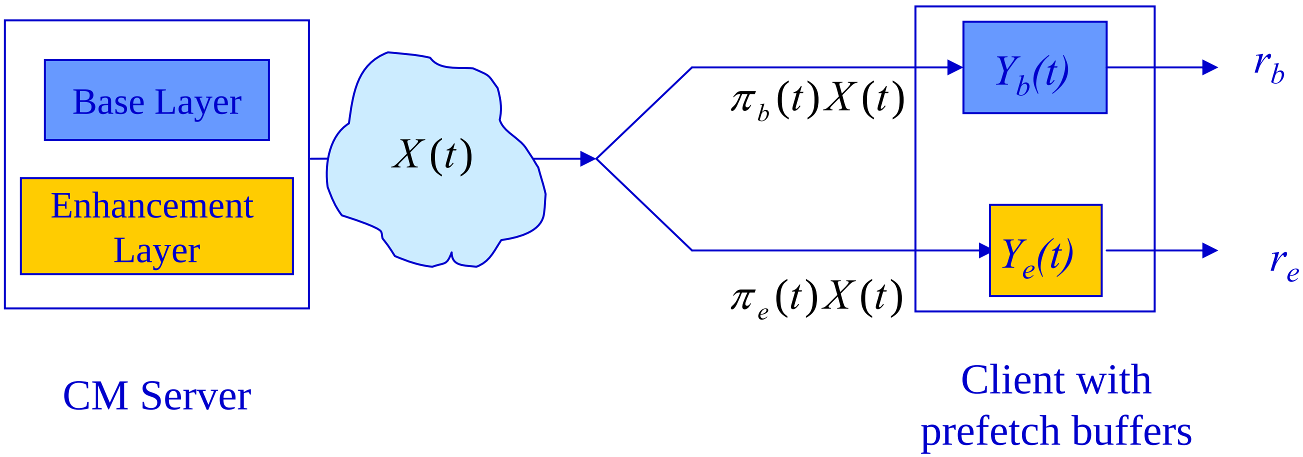
What we learned so far:

- Prefetching is essential with TCP bandwidth.
- Need to prefetch minutes into the future.
- Rate adaptation should be considered:
  - multiple versions
  - multiple layers
  - on-the-fly compression

Optimal Streaming for Layered Video (Infocom 2000)

- base and enhancement layer
- decoding constraint: enhancement layer information can only be used when the corresponding base layer information is available
- Goal: Design streaming policies that maximize the playback quality in a variable bandwidth environment.

Layered Streaming Model



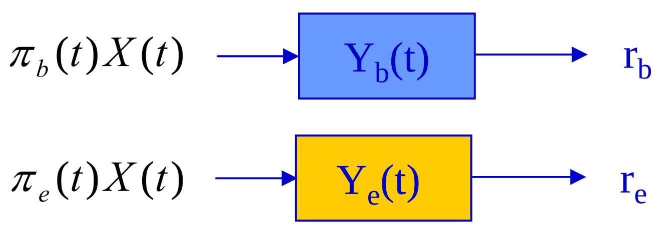
- *X(t)* = fair-share bandwidth available to the stream at time *t*
  - stochastic process whose statistical characteristics are unknown *a priori*
- r_b, r_e = encoded rates of base layer and enhancement layer
- $Y_b(t)$, $Y_e(t)$ = prefetch buffer contents at time t

Streaming Policies

- $\pi_b(t)$ can depend on the current and past history of the system, including on $X(t)$, $Y_b(t)$, and $Y_e(t)$
- Low-risk policy: $\pi_b(t)=1$
- High-risk optimistic policy:

$$\pi_b(t) = \hat{\alpha} = \frac{r_b}{r_b + r_e}$$

- Two fluid queues:



Analysis

- Initial playback delay Δ seconds.
- Delay between server and client is zero.
- Examine two cases:

  - Infinite-Length Video, $X(t)$ stationary, $\lambda = E[X(t)]$

  - Finite-Length Video of duration T seconds.

Infinite Video Duration

- Fraction of Traffic Lost
  - Base-layer loss:

$$P_b^{\pi} = \lim_{T \to \infty} \frac{\int_{t=0}^{T} [r_b t - \pi_b(t) X(t)]^{+} \, 1(Y_b(t) = 0) \, dt}{r_b T}$$

  - Enhancement-layer loss:

$$P_e^{\pi} = \lim_{T \to \infty} \frac{\int_{t=0}^{T} [r_e t - H(t)]^{+} \, dt}{r_e T} \quad ,$$

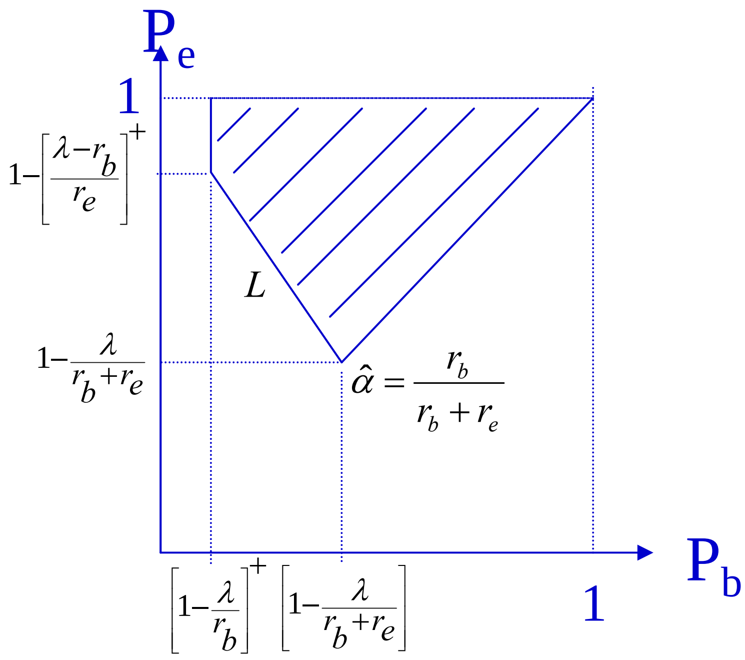
where *H(t)* is the consumption rate of enhancement-layer data at time *t*.

Partial-loss Model

- Fraction of enhancement-layer traffic consumed can be as much as the fraction of base-layer traffic consumed:

$$H(t) = \begin{cases} r_e & \text{when } Y_b(t) > 0, Y_e(t) > 0 \\ \pi_e(t)X(t) & \text{when } Y_b(t) > 0, Y_e(t) = 0 \\ r_e \dfrac{\pi_b(t)X(t)}{r_b} & \text{when } Y_b(t) = 0, Y_e(t) > 0 \\ \min\left\{ \pi_e(t)X(t), r_e \dfrac{\pi_b(t)X(t)}{r_b} \right\} & \text{when } Y_b(t) = 0, Y_e(t) = 0 \end{cases}$$

Set of Feasible Loss Probabilities: Infinite Length



- **Theorem:** Each point on *L* can be achieved by some **static policy**

Finite Video Duration

- Optimization problem:

$$\max_{\pi} \quad J_\pi = \mathrm{E}\left[d_b\left(1 - P_b^{\pi}\right) + d_e\left(1 - P_e^{\pi}\right)\right]$$

- *Let T_b^{π}, T_e^{π}* be the times at which streaming of each layer is complete
  - Lemma 1: $T_b^{\hat{\alpha}} = T_e^{\hat{\alpha}} = T_c$
  - Lemma 2: $\max\{T_b^{\pi}, T_e^{\pi}\} \le T_c$ for any policy π.
- **Theorem:**

$$\text{The policy } \hat{\alpha} = \frac{r_b}{r_b + r_e} \text{ is optimal for } J_\pi \text{ when } \frac{d_e}{d_b} \ge \frac{r_e}{r_b}.$$

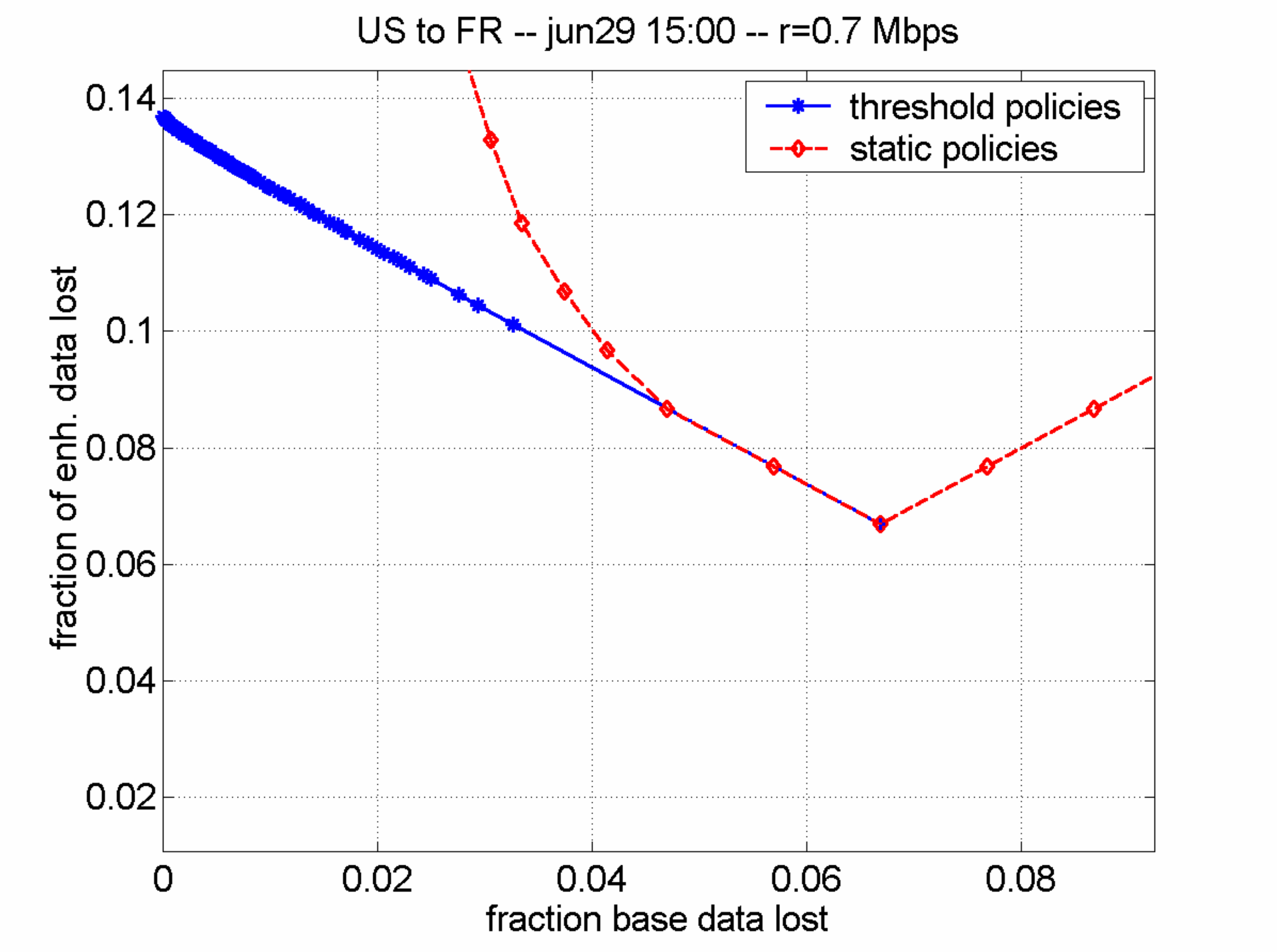
Heuristics for Finite-Length Video

- When $\frac{d_e}{d_b} < \frac{r_e}{r_b}$ static policies can perform poorly.
- Static threshold policy:

$$\pi_b(t) = \begin{cases} 1 & \text{when} \quad Y_b(t) < q_{thres} \\ \hat{\alpha} & \text{when} \quad Y_b(t) \geq q_{thres} \\ 0 & \text{when} \quad Y_b(t) > r_b(T-t) \end{cases}$$

- Must determine q_{thres}.

Threshold Policy Results



- trace results: consumption rate = avg. bandwidth

What we have learned:

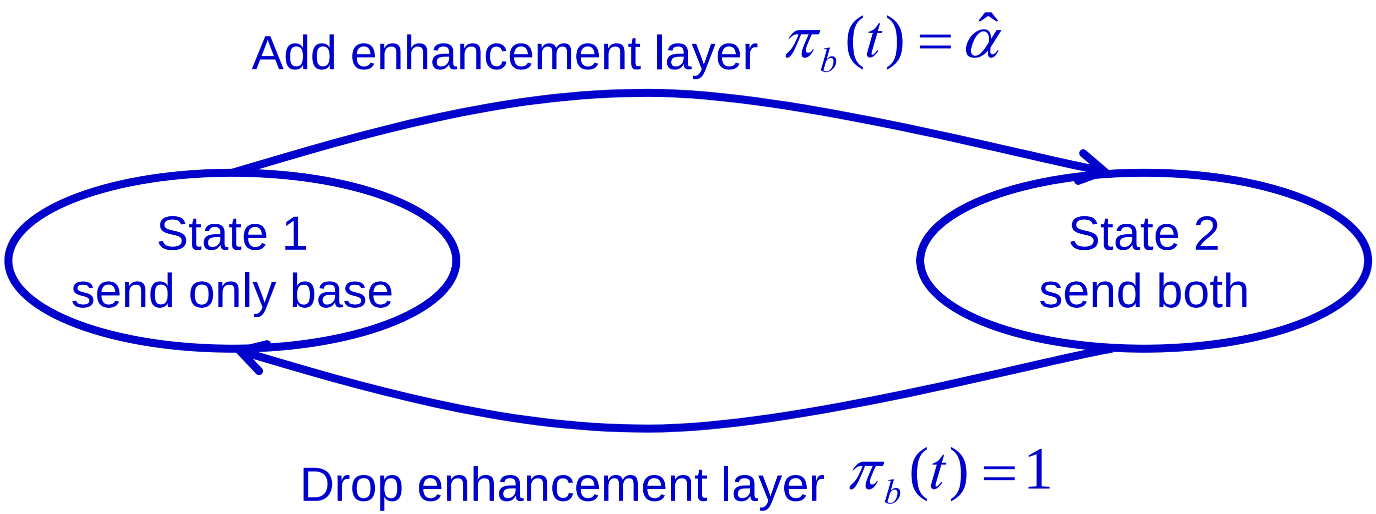
- Infinite-length video:
  - Fixed fraction of bandwidth to base layer is optimal
- Finite-length video:
  - static policy performs poorly
  - need to dynamically change policy based on prefetch buffer contents
- Note: so far have only considered time-independent thresholds

Dynamic Threshold Policies [NOSSDAV 2000]

- Now going to develop a heuristic for setting the thresholds as a function of time.

- Heuristic Philosophy:
  - try to always render the base layer
  - render the enhancement layer for as much as possible
  - avoid frequent fluctuations in quality

Dynamic Threshold Heuristics (1)



- Monitor client buffer content in each layer
- Estimate future average bandwidth (using past observations)
- Add enhancement layer if base layer can still be reliably delivered and if enhancement layer can be kept for a certain period of time

Dynamic Threshold Heuristic (2)

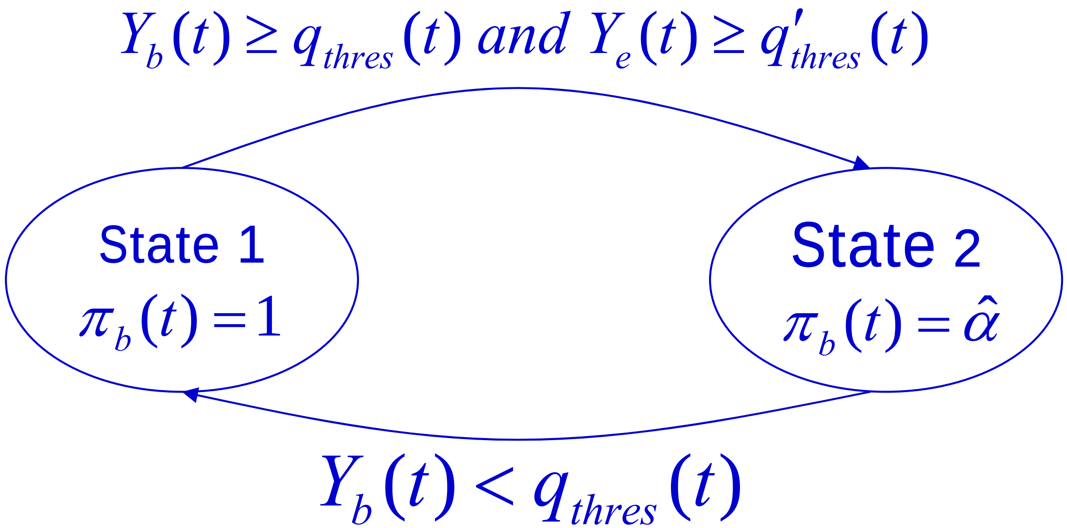
- Threshold at time *s* for adding enhancement layer is such that base-layer buffer will not starve for the next *C* seconds:

$$q_{thres}(s) = C \cdot \left(r_b - \hat{\alpha} \cdot X_{avg}(s)\right)$$

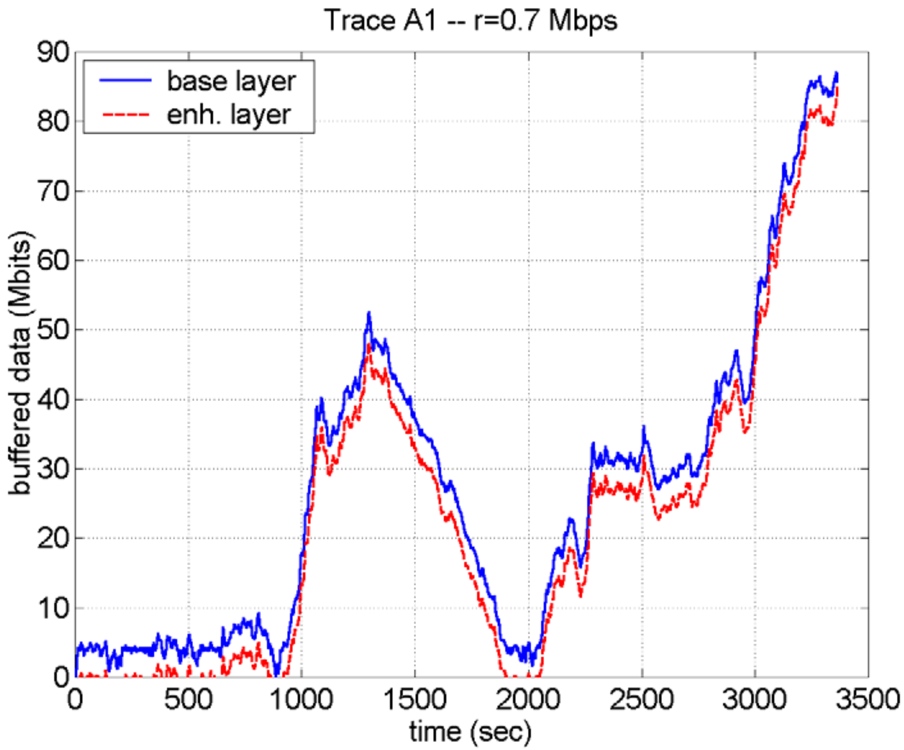
- To avoid rapid quality fluctuations, introduce a threshold for buffered enhancement layer data:

$$q'_{thres}(s) = C' \cdot \left(r_e - (1-\hat{\alpha}) \cdot X_{avg}(s)\right)$$

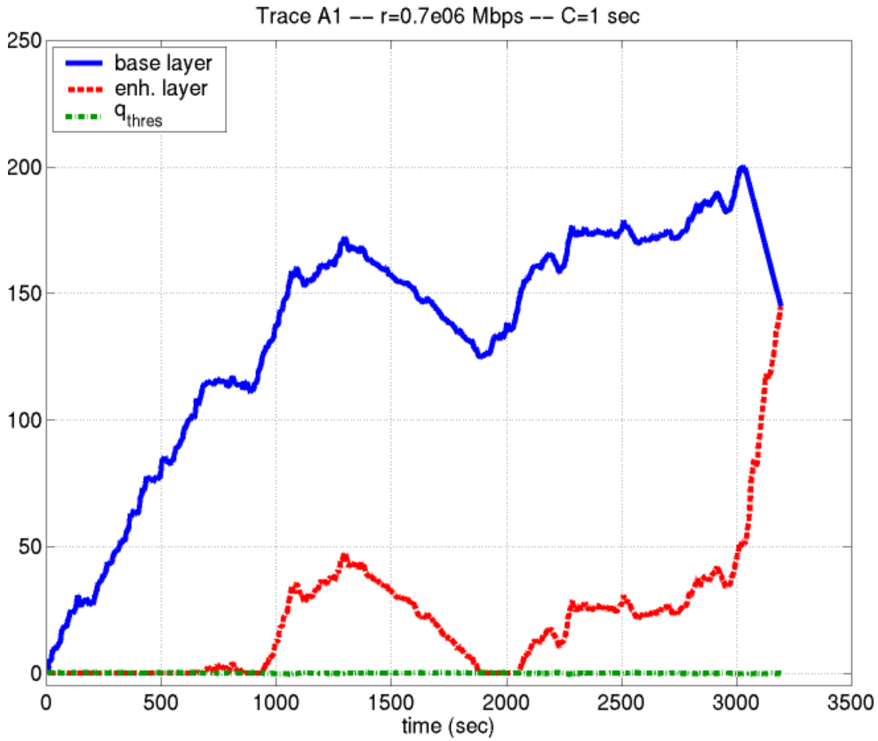
Dynamic Threshold Heuristic (3)



Static Threshold vs. Dynamic Threshold Heuristic



static threshold policies



dynamic threshold policies

Dynamic threshold heuristic results in fewer quality fluctuations with the same high-quality viewing time.

What we just learned:

- Measurement-based heuristic performs well:
  - minimal or no base layer loss
  - few fluctuations in quality
- Open issues:
  - length of prediction interval
  - suitable values for key heuristic parameters

Now that we have a handle on layered video, let's now consider multiple versions !

Outline of the Talk

I Streaming Stored Layered Video over Fair-Share Bandwidth

II Layers vs. Switching Versions over Fair-Share Bandwidth (Philippe)

III Layered Video through Caches

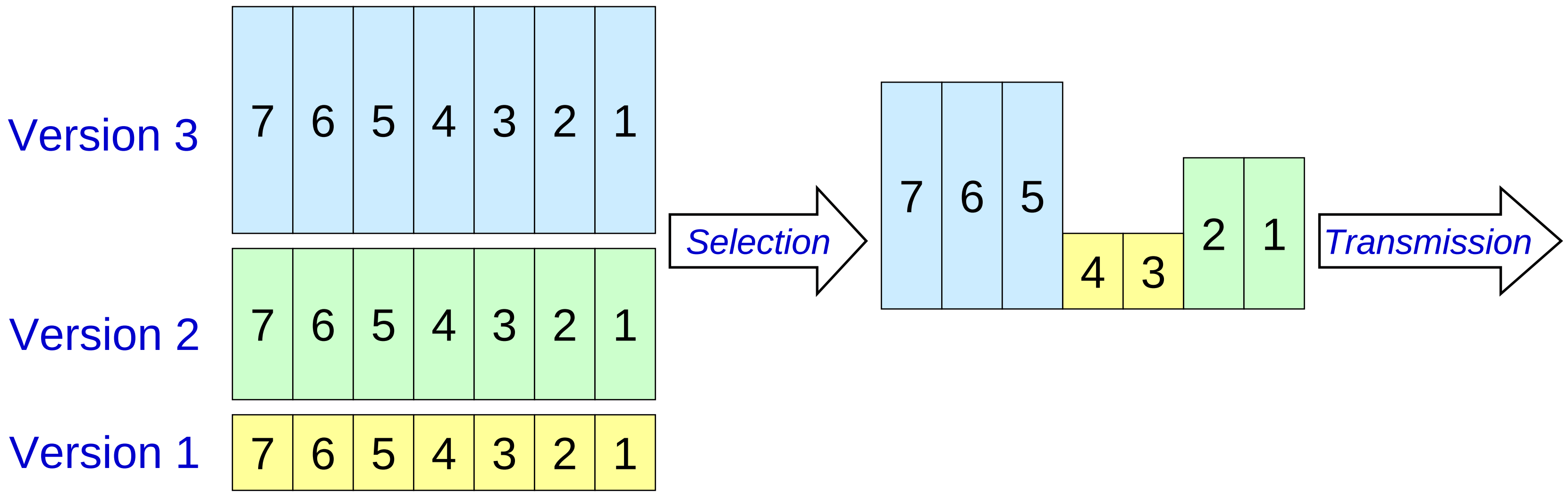
IV Asynchronous Interactive Audio Streaming

Switching Versions
[Packet Video Conf, 2001]

- As an alternative two layered video, server can store multiple versions of the same video.
- Multiple versions requires more disk space, but does it provide better viewing quality given the same X(t)?
- Comparison approach:
  - Design analogous control policies for layers and for switching among versions.

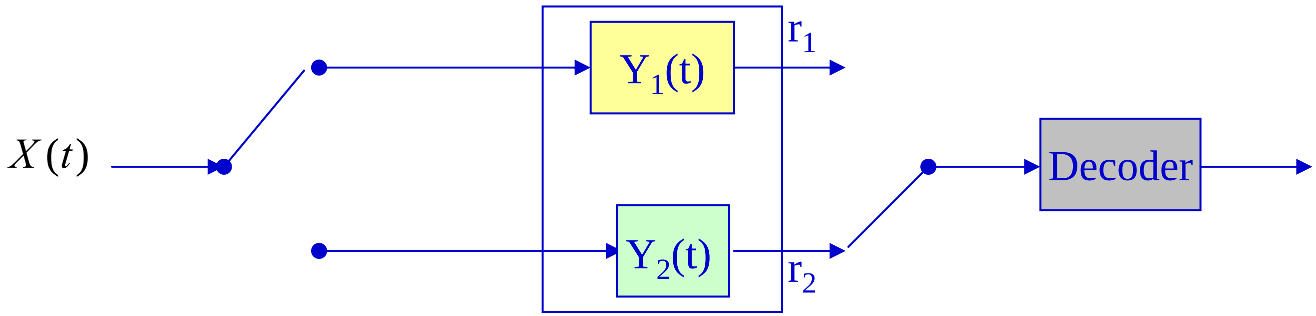
Switching Versions

- Different versions of the same video encoded at different bit-rates.
- Switch among the versions to adapt the transmission rate to the available bandwidth.



Switching Versions Control Policy

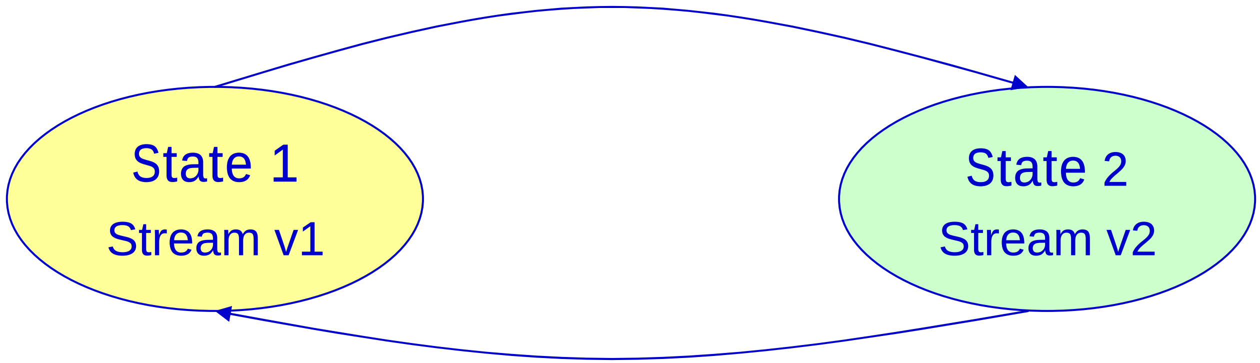
- At any instant, server has to determine which version of the video to send (version 1 or version 2).



- Server will estimate $Y_1(t)$ and $Y_2(t)$ using receiver reports (e.g. RTCP) and make the same computation of $X_{avg}(t)$.

State transition diagram for switching versions

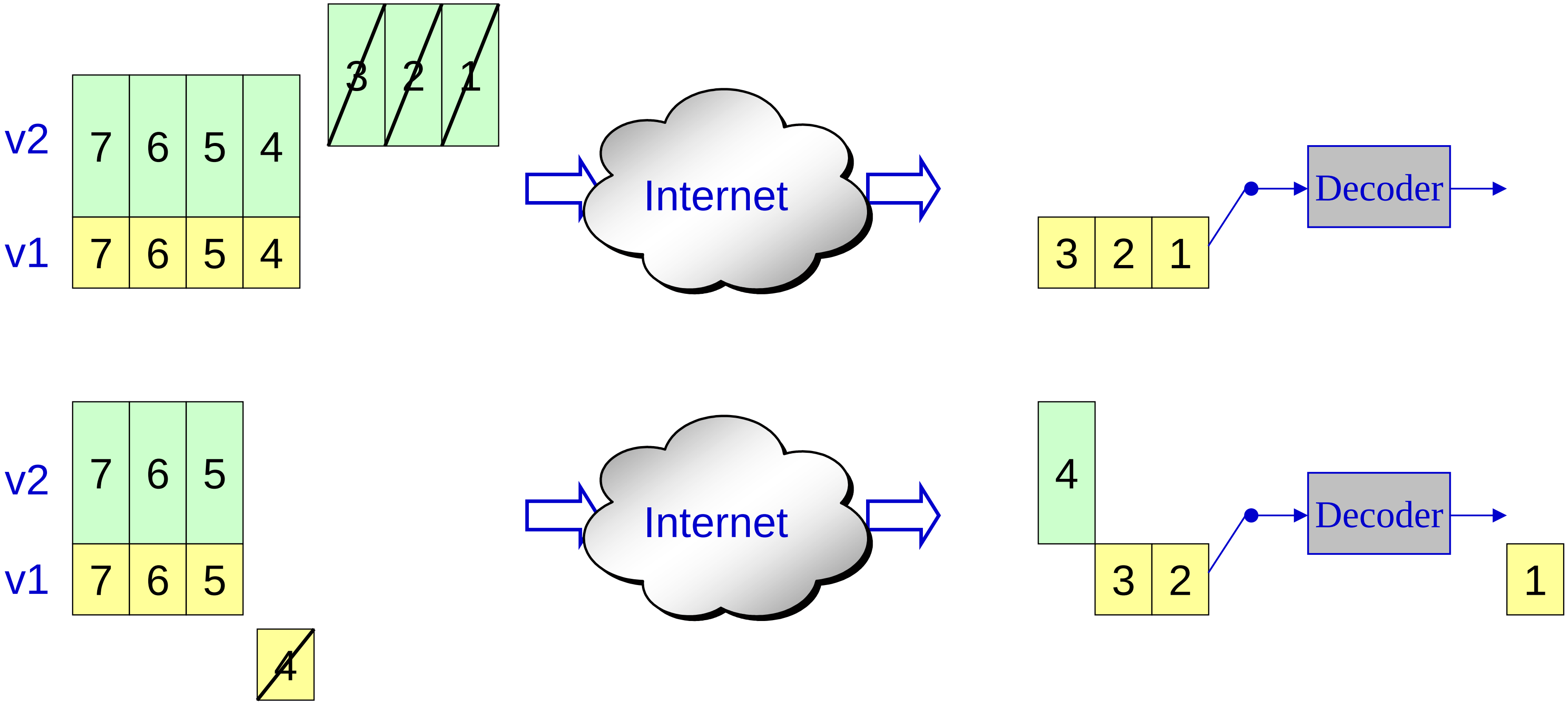
(1) $$\frac{Y_1(t)}{r_1} + \frac{Y_2(t)}{r_2} \geq C\left(1 - \frac{X_{avg}(t)}{r_2}\right) \text{ and } X_{avg}(t) \geq r_2$$



(2) $$\frac{Y_1(t)}{r_1} + \frac{Y_2(t)}{r_2} < C\left(1 - \frac{X_{avg}(t)}{r_2}\right) \text{ or } \frac{Y_1(t)}{r_1} + \frac{Y_2(t)}{r_2} < \Delta$$

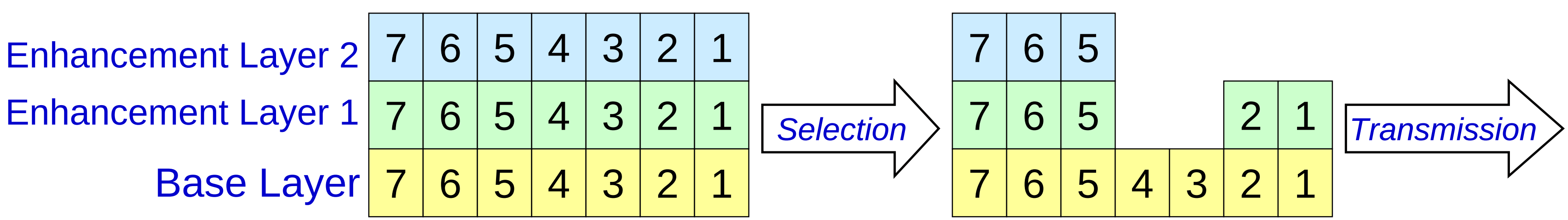
First implementation of switching versions

- The server always transmits data from v_2 beginning with the first video frame that has not yet been buffered in v_1.



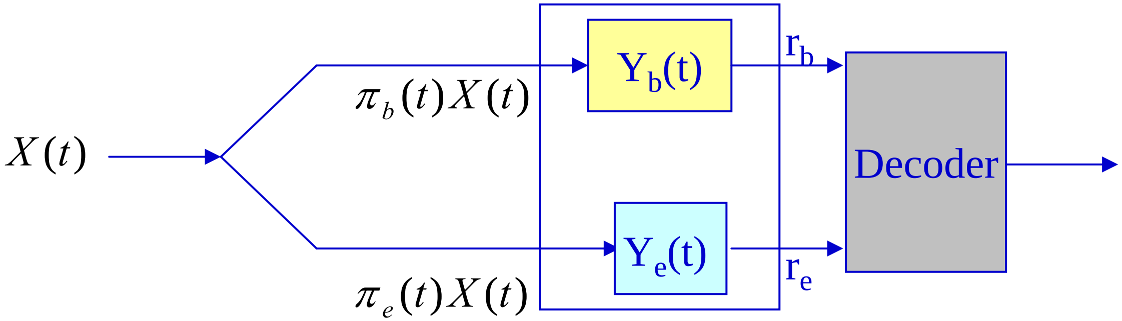
Adding/Dropping Layers : Review

- Base Layer (BL) and Enhancement Layers (EL).
- To decode higher quality layers, all lower quality layers must be available to the decoder.
- Can add/drop layers to adjust to the available bandwidth.



Adding/Dropping Layers Control Policy

Recall:



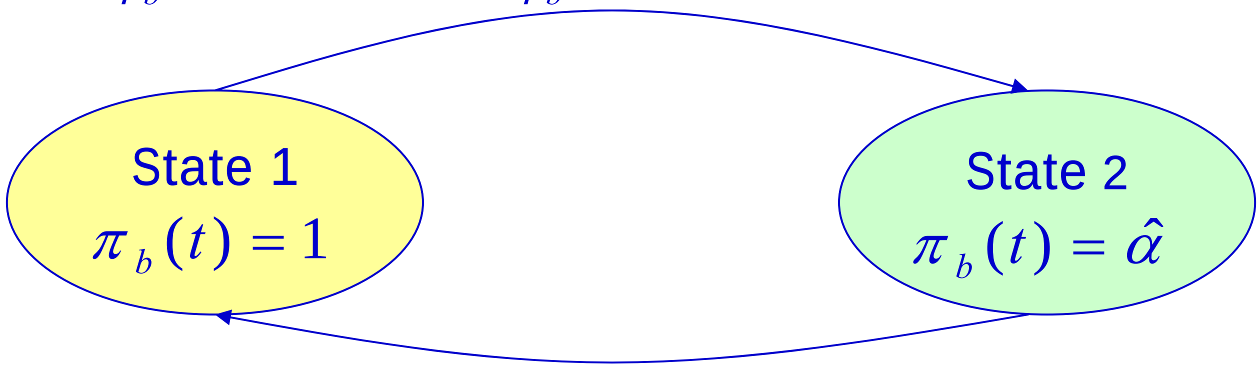
$$\begin{cases} \pi_b(t) = 1 \\ \pi_b(t) = \hat{\alpha} = \dfrac{rb}{rb + re} \end{cases}$$

=> Stream BL only

=> Stream BL & EL in proportion to their consumption rate

State transition diagram for adding/dropping layers

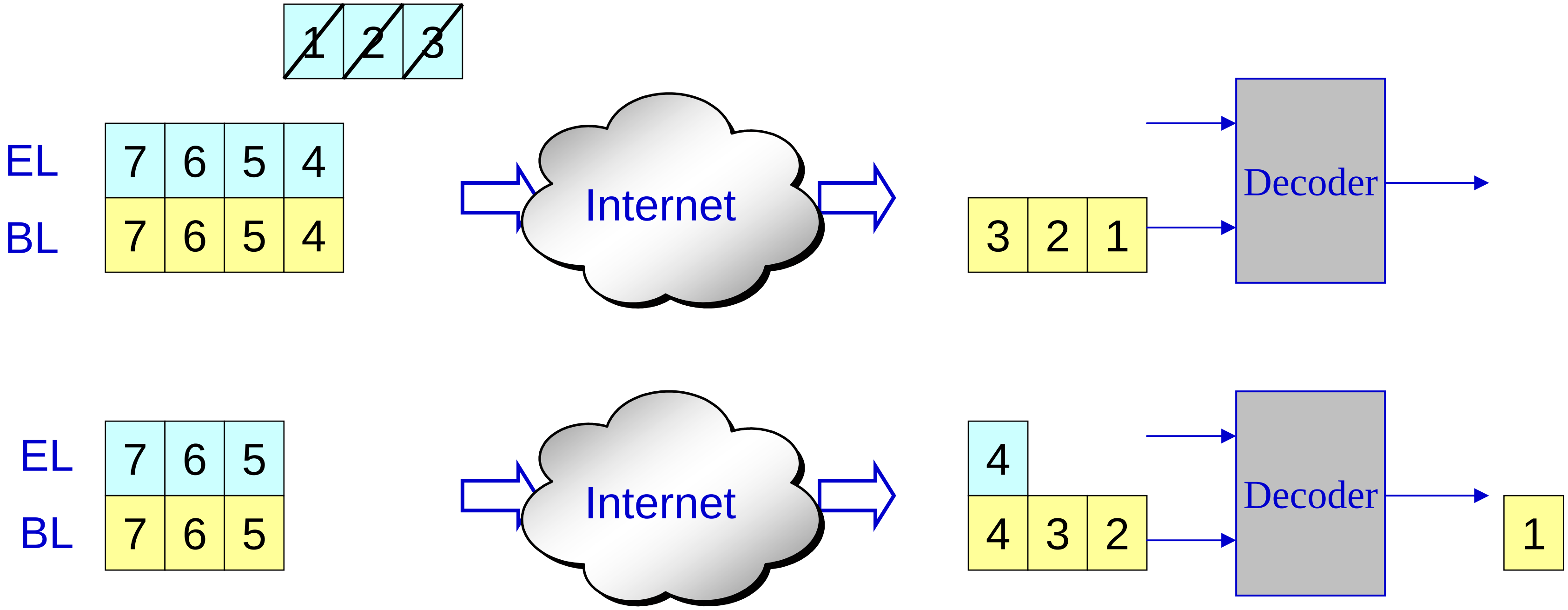
(1) $$\frac{Y_b(t)}{r_b} \geq C(1-\hat{\alpha}.\frac{X_{avg}(t)}{r_b}) \text{ and } (1-\hat{\alpha}).X_{avg}(t) \geq r_e$$



(2) $$\frac{Y_b(t)}{r_b} < C(1-\hat{\alpha}.\frac{X_{avg}(t)}{r_b}) \text{ or } \frac{Y_b(t)}{r_b} < \Delta$$

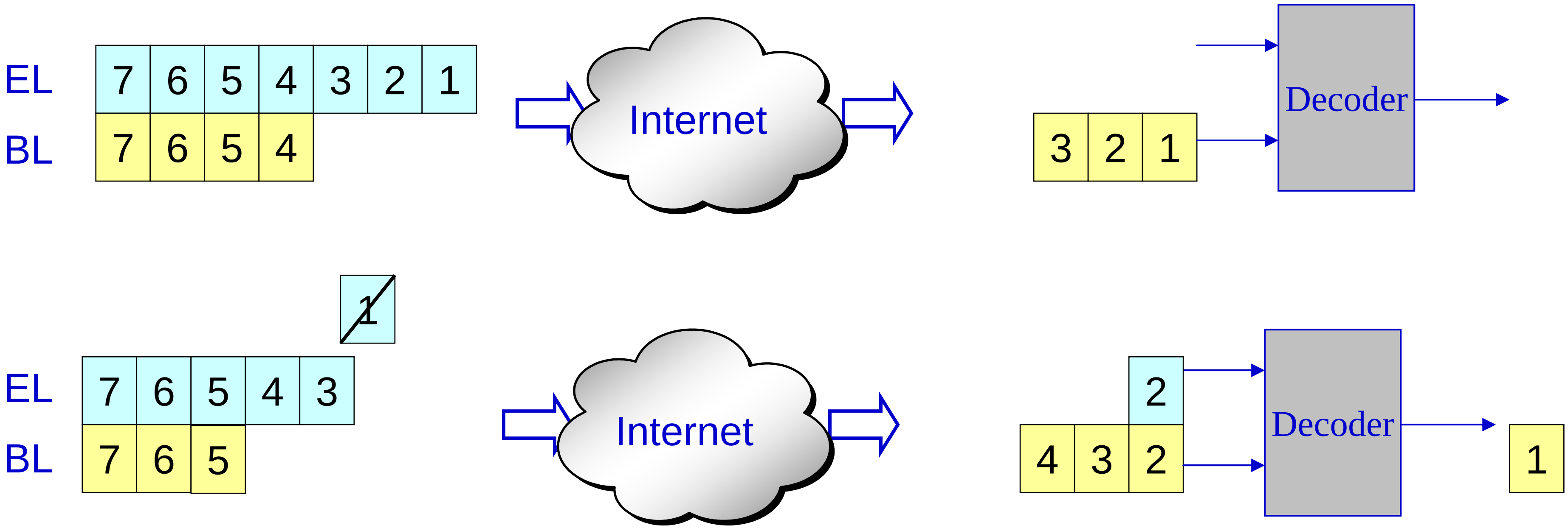
First implementation for adding/dropping layers

- Transmit the enhancement layer data with the same playback deadline as the BL currently being transmitted.



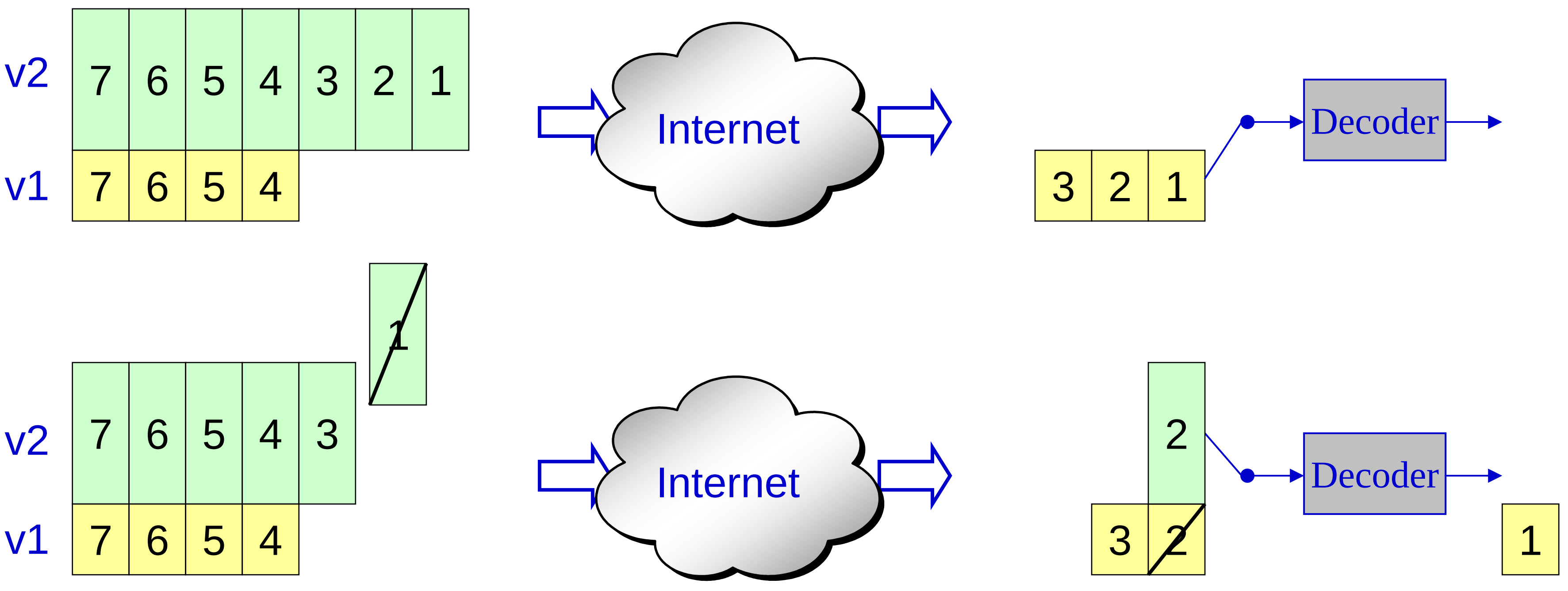
Immediate Enhancement for adding/dropping layers

- Transmit the enhancement layer data with the earliest playback deadline.
  => synchronization is more complex



Immediate Enhancement for switching versions

- Transmit data from v2 beginning with the frame which has the earliest playback deadline



Comparison of Rates

- In order to make a fair comparison :
  - BL and v1 have the same perceptual quality.
  - BL+EL and v2 have the same perceptual quality.
- Layering has a coding penalty H percent (between 1% and 10%).
- Assume that all the coding overhead is associated with the EL:

$$\begin{cases} r_b + r_e = (1 + H).r_2 \\ r_b = r_1 \end{cases}$$

Simulations

- 1-hour throughput traces from TCP connections on the Internet, averaged over 1 sec
  => TCP-friendly bandwidth conditions

- 3 different performance metrics to compare performance:
  - Fraction of high-quality viewing time (t_h).
  - Fraction of time the decoder can not display the video (t_d).
  - Quality fluctuations (S).

- Study behavior of our heuristics under different bandwidth conditions.

Results

- First implementation:
  - Performance of adding/dropping layers deteriorates as layering overhead increases.
- Immediate enhancement:
  - Inefficient for switching versions (waste of bandwidth).
  - Under certain bandwidth conditions, adding/dropping layers attains higher th than switching versions for H up to 5%

	r_2/X=0.7			r_2/X=1.0			r_2/X=1.3		
	t_h	t_d	S	t_h	t_d	S	t_h	t_d	S
Versions	**94**%	0%	5	**58**%	0%	5	**44**%	0.4%	5
Layers-immediate H = 1 %	**95**%	0%	11	62%	0%	21	**44**%	0.4%	21
Layers-immediate H = 5 %	92%	0%	17	**60**%	0%	25	41%	0.4%	25

What we just learned:

- Simple implementation :
  - Switching versions always performs better than adding/dropping layers because of layering overhead
- Immediate enhancement implementation:
  - layering's enhanced flexibility can compensate the loss in high quality viewing time due to layering overhead
- Neither scheme seems to dominate:
  - adding/dropping layers is probably better when streams pass through caches

Outline of the Talk

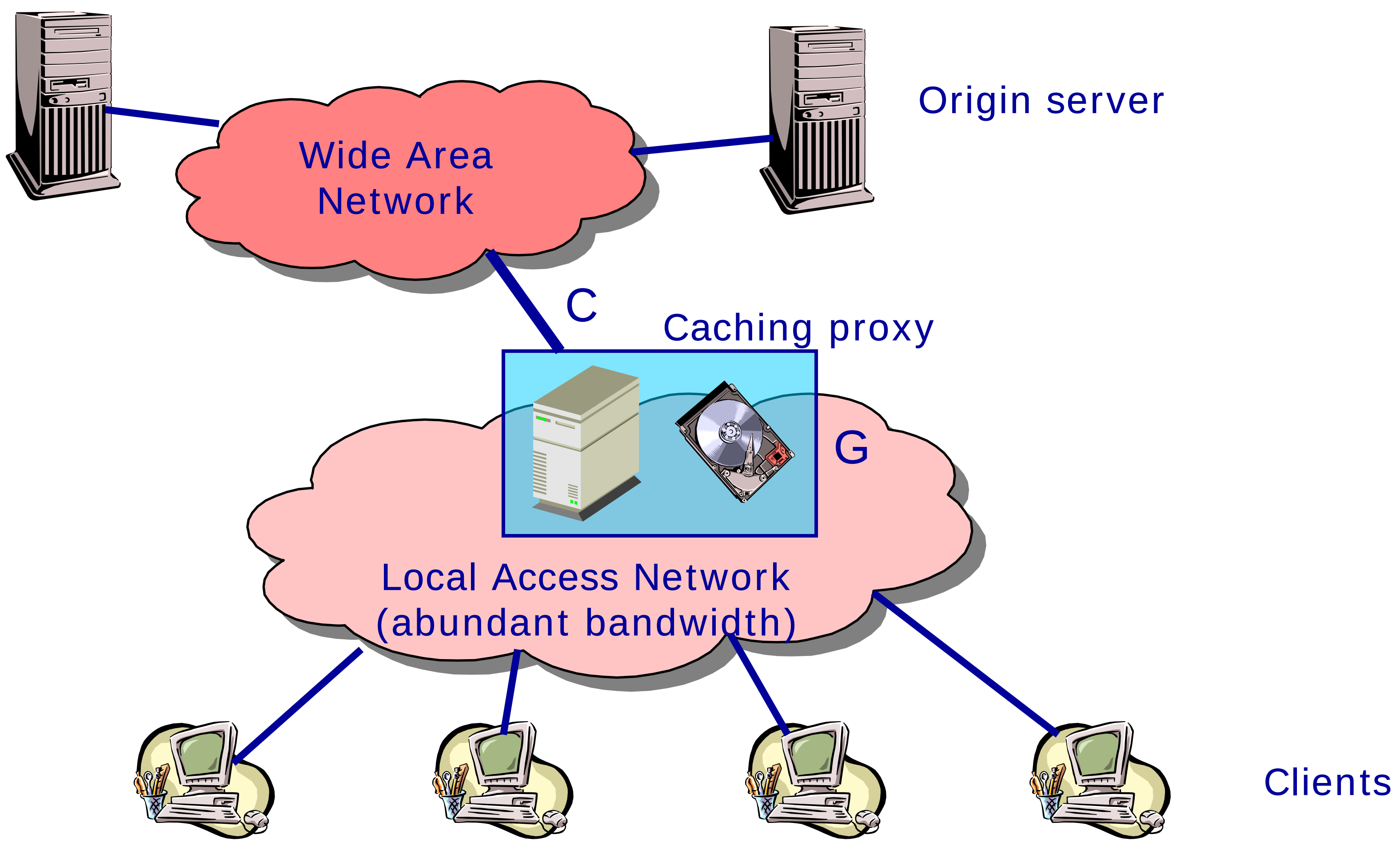
I Streaming Stored Layered Video over Fair-Share Bandwidth

II Layers vs. Switching Versions over Fair-Share Bandwidth (Philippe)

III Layered Video through Caches

IV Asynchronous Interactive Audio Streaming

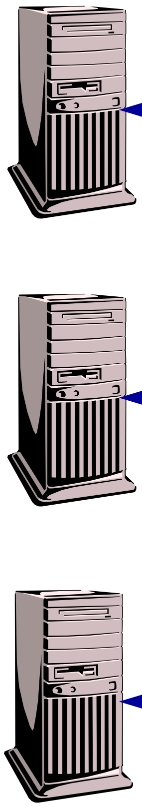
Layered Video Through Caches (Infocom 2001)



Model for Layered Caching

- M videos, each has L layers
- T(m) = duration of video m
- Each layer has rate $b_l(m)$
- Revenue $R(j, m)$ = j layers of m
- Requests for "j layers of m"
- c_m = number of layers cached for video m
- Caching Policy: $\mathbf{c} = (c_1,\ldots,c_M)$

Stream Delivery



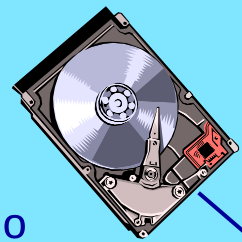
Link (Stochastic Knapsack)

- Capacity *C*
- *ML* classes, one per request type
- Request holds $r_j(m)$ of bandwidth for $T(m)$

$$r_j(m) = \sum_{l=c_m+1}^{j} b_l(m)$$

- Admit if enough capacity, otherwise reject request
- Calculate expected blocking probability

Caching proxy

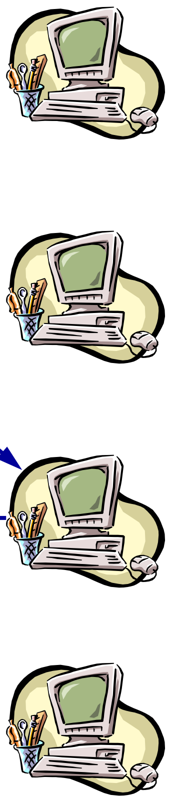


No

Yes

Is *j* of *m* cached?

Request *j* of *m*



- Calculate long run total rate of revenue

Optimization Problem

Maximize revenue

$$\max_{\mathbf{c}} R(\mathbf{c}) = \lambda \sum_{m=1}^{M} \sum_{j=1}^{L} R(j,m)\, p(j,m)(1 - B_{\mathbf{c}}(j,m))$$

subject to

$$\sum_{m=1}^{M} \sum_{l=1}^{c_m} b_l(m) T(m) \leq G$$

Optimal Caching

- Analytically intractable

- Exhaustive searches not feasible
  - M = 50, L = 2, G = 20 streams
  - $2.9 * 10^{16}$ possibilities

- Need heuristics

Heuristics

- Assign utility to each layer
- Cache layers in decreasing utility
- Three utility definitions:
  - *Popularity*: Popularity of layer + higher layers
  - *Revenue*: Popularity * revenue
  - *Revenue density*: Revenue / size
- Layer is cached only if all lower layers are cached

Definitions of Heuristics

- Popularity heuristic

$$u_{l,m} = \sum_{j=l}^{L} p(j,m)$$

- Revenue heuristic

$$u_{l,m} = \sum_{j=l}^{L} R(j,m)p(j,m)$$

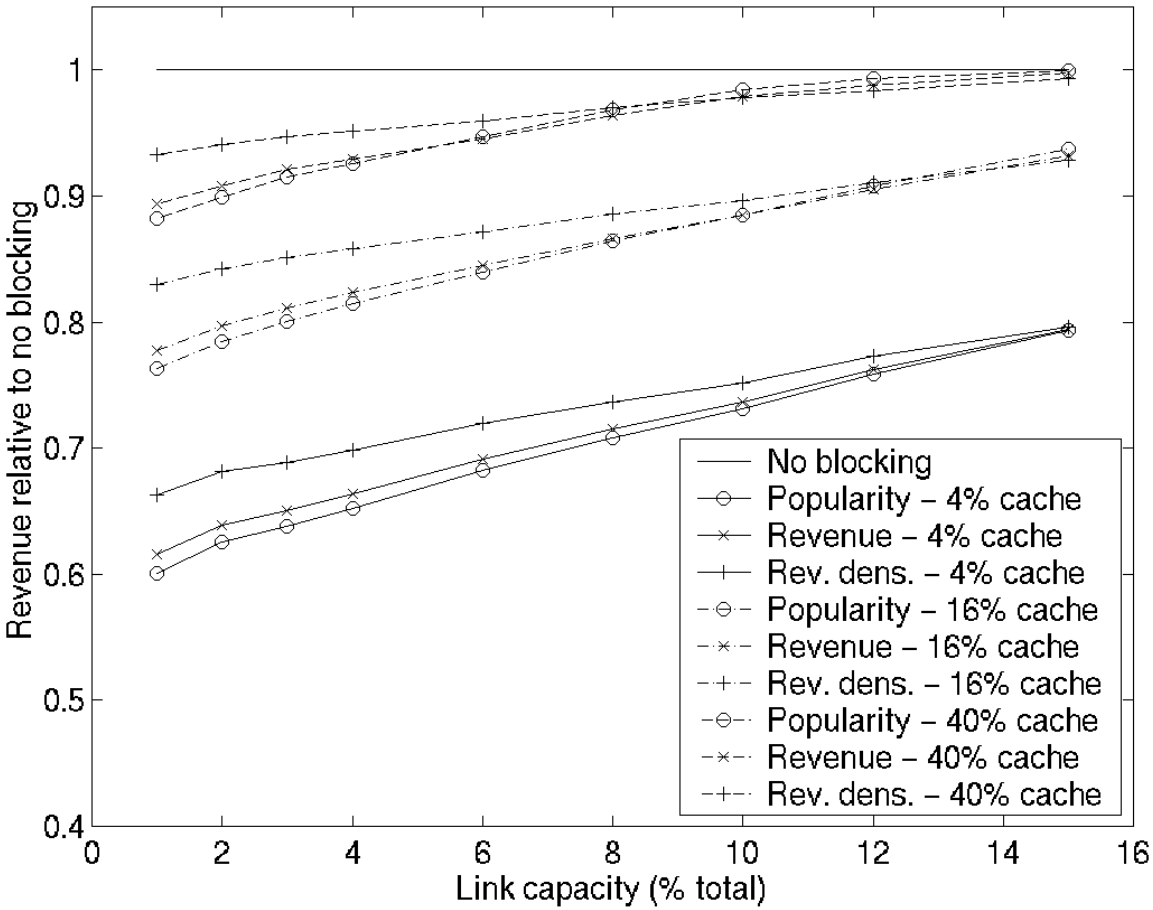
- Revenue density heuristic

$$u_{l,m} = \sum_{j=l}^{L} \frac{R(j,m)p(j,m)}{b_j(m)T(m)}$$

Evaluation Methodology

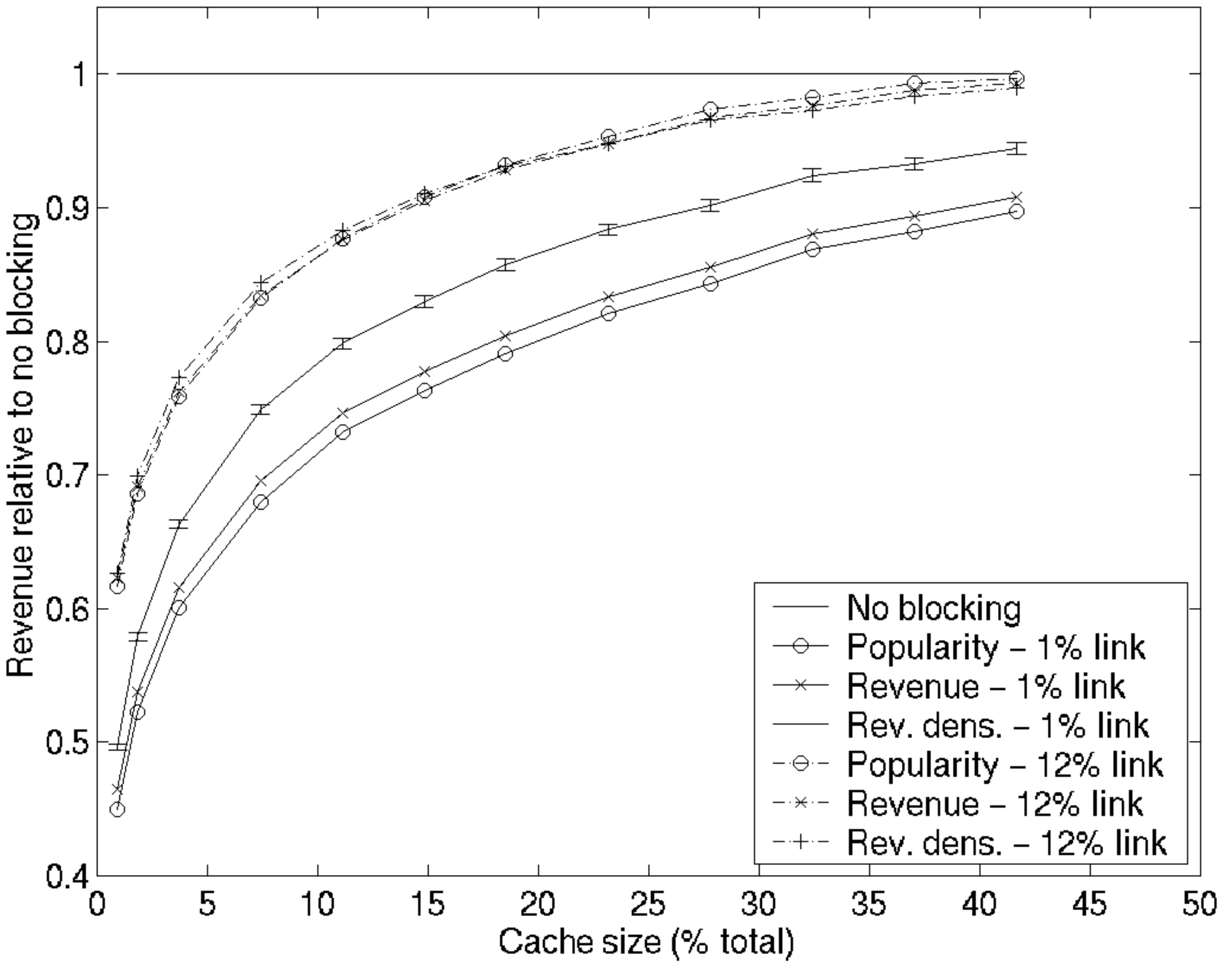
- 1000 videos, each 2 layers
- Rates uniformly distributed: 0.1 to 3 Mbps
- Revenue uniformly distributed: 1 to 10
- Length exponential, mean 1 hour
- Zipf-popularity (parameter 1.0)
- G = 12-560 GB (0.9-42 % of video bytes)
- C = 10-150 Mbit/s

Evaluation - Link Capacity



EURECOM

Evaluation - Cache Size



What we just learned:

- Interesting stochastic knapsack model for 2-resource layered video caching problem
- 3 heuristics: best is revenue density
- Work in progress:
  - Compare versions and layering through caches

Outline of the Talk

I Streaming Stored Layered Video over Fair-Share Bandwidth

II Layers vs. Switching Versions over Fair-Share Bandwidth (Philippe)

III Layered Video through Caches

IV Asynchronous Interactive Audio Streaming (Wimba startup)

Wimba

- http://www.wimba.com
- Asynchronous Interactive Voice
- eLearning applications
- bringing together the telephone and Internet worlds