An Overview of Internet Multimedia Networking (Part II)

Prof. Reza Rejaie

Computer & Information Science

University of Oregon

http://www.cs.uoregon.edu/~reza

Winter 2003

Window-based CC

- Server maintains a window of packets in flight, called congestion window (cwn)
- cwn regulates transmission rate
  - Tx rate = (cwn * PktSize)/RTT
- Size of cwn is adjusted by Inc/Dec algorithm, for AIMD
  - No loss => cwn = cwn + 1
  - Loss occurs => cwn = cwn/2
- window-based CC are often "ack-clocked"
  - Incoming ack triggers tx of a new pkt
  - Window-based CC can become burst

Rate-based CC

- Server maintains inter-pkt-gap (ipg)
- ipg regulates transmission rate
  - Tx rate = PktSize/ipg
- ipg is adjusted by Inc/Dec algorithm, for AIMD
  - No loss => ipg = ipg * RTT / ipg + K*RTT
  - Loss occurs => ipg = 2*ipg
- Rate-based CC are not ack-clocked
  - Outgoing pkts are decoupled from incoming acks
  - No of pkts in transit is not directly controlled
  - May significantly overshoot available bw !!

CC for Streaming

- Goals: to achieve a fair share of network bw among co-existing flows
  - TCP-friendly CC is desired because more than 90% of Internet traffic consist of TCP flows
- Conflicting requirements:
  - Smooth adjustments in bw are desired
  - Responsive to rapid changes in network load
- Equation-based CC

Equation-based CC

- TCP response function
  - $T = (k * PktSize)/(RTT * p^{1/2})$
  - p is loss event rate, k= 1.3
- This equation estimates an upper bound on TCP's sending rate in steady state
- To design a "simple" equation-based CC:
  - Periodically measure RTT & p to estimate TCP BW
  - Change the tx rate to estimated TCP rate
- This simple approach may not result in a smooth variations in bw nor be tcp-friendly
- TCP-friendly Rate Ctrl (TFRC)

2) Error Control

- Loss Detection vs Loss Repair
- Loss Detection: Timeout or Gap in seqno
- Loss Repair:
  - Sender-based repair techniques
  - Receiver-based repair techniques
    - Error Concealment, do Not require assistance from sender
    - Good when sender is unable or unwilling to help with recover
  - These are complementary,
- Applicable to both unicast and multicast streaming. Scaling and heterogeneity issues are harder for multicast

Sender-based repair (taxonomy)

- Active:
  1. Retransmission
- Passive:
  1. Interleaving
  2. Forward Error Correction (FEC)
     1. Media independent
     2. Media specific

Retransmission

- Bandwidth efficiency (+)
  - Only lost packets are retx
- Allows selective recovery (+)
  - Can selectively repair only important pkts
- Long recovery delay (-)
  - Requires at least 3 one-way-delay
- May not be appropriate for:
  - Multicast streaming,
  - Interactive applications, long delay
    - Forward Error Correction

Media-Independent FEC

- *k* data pkts are coded to *n* pkts for tx
  - *n-k* additional check pkts
- Parity Coding
  - XOR *n-1* data pkts => *n*th parity pkt
  - Loss of *1* out of *n* pkts can be recovered
  - In general case k lost pkts can be recovered!
  - Different combination of pkts can be XORed

Media-Independent FEC (cont)

- Advantages:
  - Media independent
  - Repair pkt is exactly similar to the lost pkt
  - Relatively low computation overhead
- Disadvantages:
  - Additional delay for processing
  - Extra bandwidth for sending FEC pkts
  - Difficulty for decoder implementation

Media-Specific FEC

- Add a low quality copy of a pkt, to one of the following pkts
  - Low quality copy is used when the main pkt is lost
  - Mostly used for audio
- Could achieve very low latency
- Can improve robustness against burst loss
  - Place a low quality copy N pkts later
  - The larger N, the more robust to burst loss, the higher the recovery delay

Interleaving

- interleaving is useful when
  - Data unit is small
  - Long end-to-end delay can be tolerated
- Resequence pkts before transmissio
- Interleaving disperse the effect of pkt loss

[1 5 9 13] [2 6 10 14] [3 7 11 15] [4 8 12 16]

1 X 3 4 5 X 7 8 9 X 11 12 13 X

Receiver-based repair (taxonomy)

- Insertion-based schemes
  - Inserting a fill-in pkt instead of lost pkt
  - e.g. silence of noise for audio
- Interpolation-based schemes
  - Use pattern matching to derive a replacement pkt which is expected to be similar to lost pkt
- Regeneration-based scheme
  - Derive the decoder state from pkt surrounding the loss and generate a replacement

A summary

- Primary approaches:
  - Retransmission (Retx)
  - Forward Error Correction (FEC)
- Retx has higher BW efficiency but results in longer recovery delay => More buf.
  - Appropriate for non-interactive app
  - Allows selective recovery
- FEC has shorter recovery delay but lower BW efficiency => Less buf.
  - How should we determine proper level of FEC?
- ➢ *Effect of pkt loss on perceived quality is encoding-specific*

3) Quality Adaptation

- How should the quality of delivered stream be adjusted as available bw changes?
- Basic idea:
  - Long-term increase in bw => increase of quality (i.e. increase stream bw)
  - Long-term decrease in bw => decrease in quality (i.e. decreasing stream bw)
  - Short-term variations in bw can be absorbed by buffering and should not trigger change of quality

Different Encoding Schemes

- Non-layer encoding
  - available bw is determined by desired quality, and content (e.g. MPEG-2)
  - Encoded stream is often VBR
- Layer (Hierarchical) encoding
  - Stream is encoded into dependent layers
  - BW of layers is often exponentially increase
- Multiple Description (MD) encoding
  - Stream is encoded into independent layers
- As flexibility of encoding increases, encoding efficiency decreases!

Different Approaches to QA

- Adaptive encoding
  - Encoder adjusts its quality on the fly
  - BW range of an efficient encoders is limited!!
  - Encoding is CPU-intensive => not good for big servers
- Trans-coding
  - Trans-coder can also change resolution of a pre-encoded stream
  - May not result in graceful degradation in quality

Diff. Approaches to QA II

- Selective Frame Dropping
  - Drop less important frames to match stream bw with available network bw
  - Does not result in graceful degradation in quality
  - Its performance depends on availability of low importance frames, i.e. depends on content
- Switching between multiple encoding
  - Server switches between streams with different quality as bw changes
  - Its performance depends on bw of encoded streams

Diff. Approaches to QA III

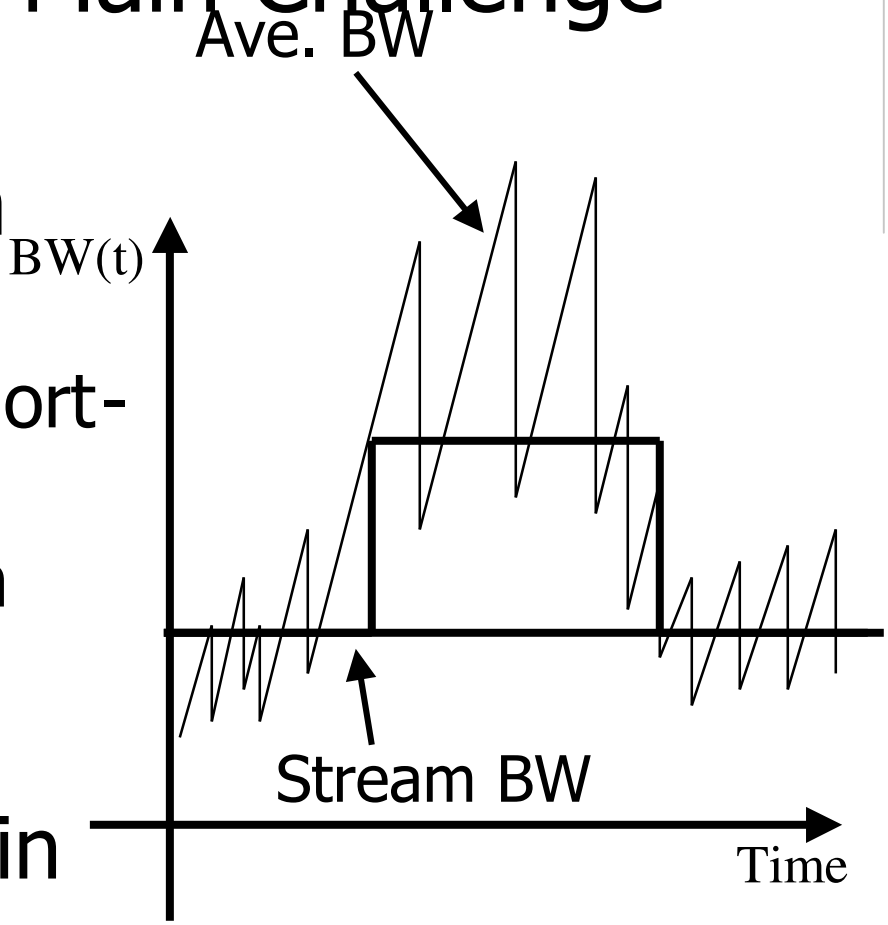
- Using Layered encoding
  - Add/drop layer as available bw changes
  - Pkts of the base layer are more important!
  - It might not be effective with exponential bw distribution among layers
- Using Multiple Description encoding
  - Similar to layered encoding case, but pkts of all layers have the same value

➢ *Effect of quality adaptation on perceived quality is encoding-specific*

➢ *How/When should adaptation be performed?*

Quality Adaptation: Main Challenge

- Mismatch between ave. BW and stream BW
  - Buffering absorbs short-term mismatch
  - Long-term mismatch triggers quality adaptation
- Frequent variations in quality is disturbing

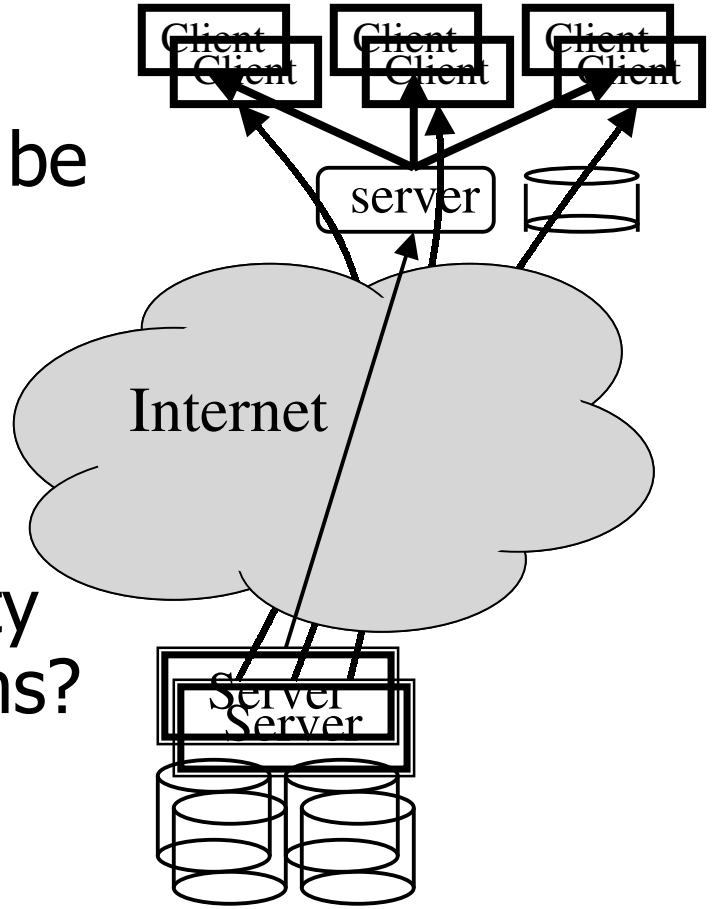


Limitations of Client-Server Arch.

- **Limited scalability**
- Limited & unstable quality
- Asynchronous access could be inefficient
- RTT could be high
  - High delay VCR-functions
  - Large startup delay

❖ Increasing network capacity doesn't solve these problems?

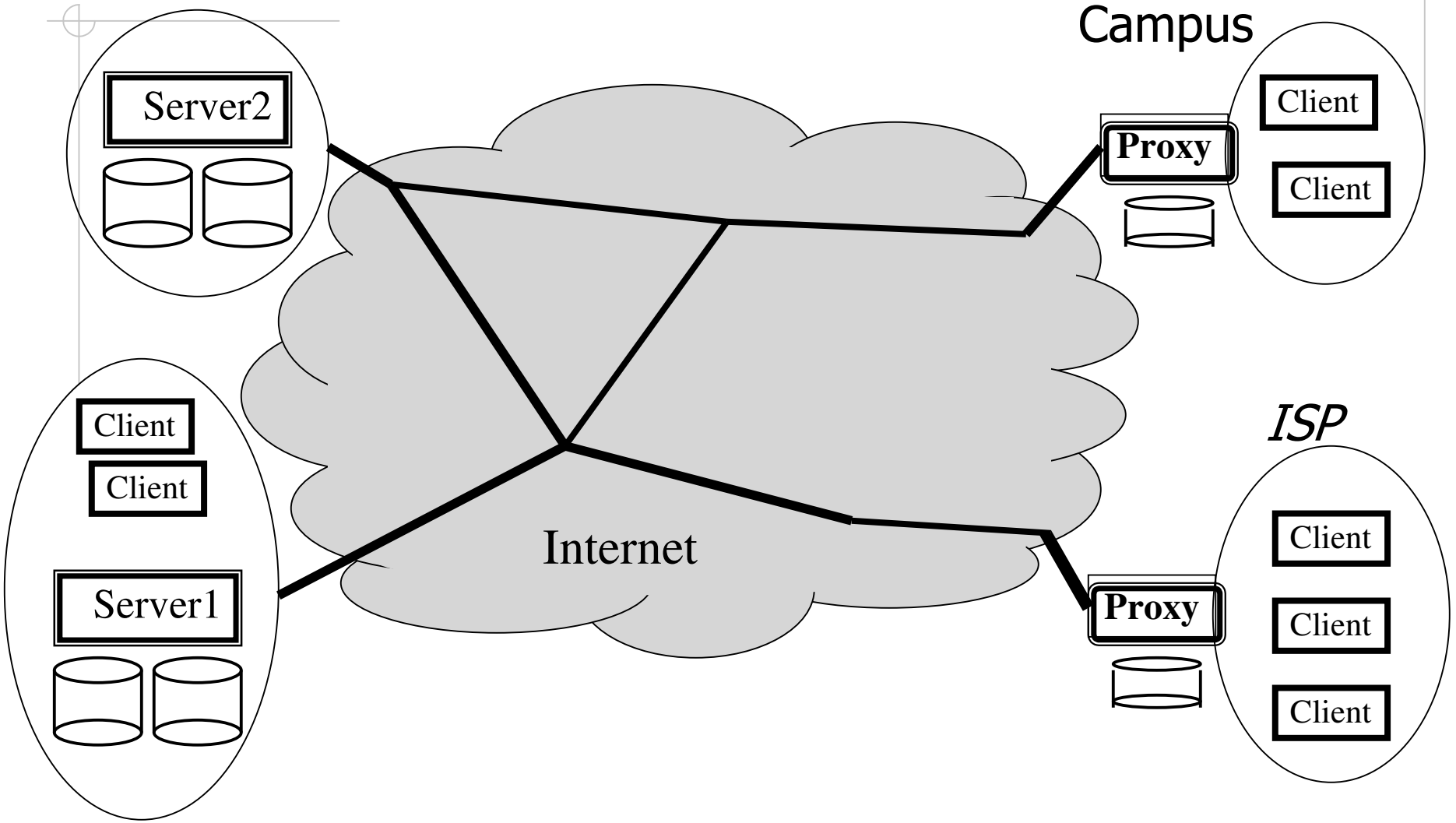
➢ *Content should be close to interested clients*



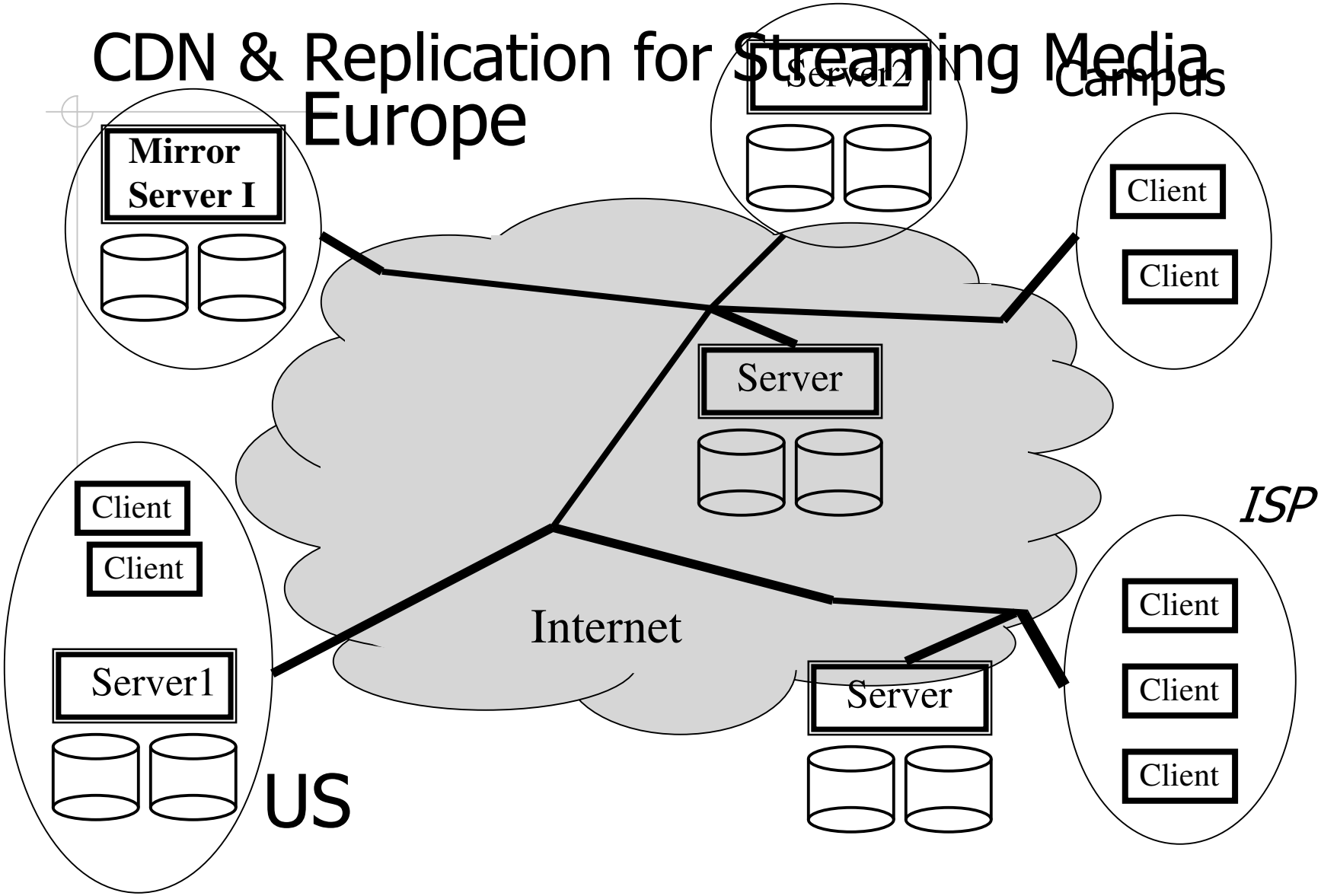
Multimedia Distribution

- Multimedia proxy caching, CDN & Mirror servers
- Multimedia Proxy services
  - Prefix caching
  - Loss repair
  - Smoothing, video staging
- Multicast-based distribution techniques
  - RLM
  - Skyscraper
- Multimedia traffic measurement
- Peer to peer streaming

Proxy Caching for Streaming Media



CDN & Replication for Streaming Media



Proxy Caching, CDN, Mirror (I)

- Location:
  - Proxy is often in the same campus
  - CDN server is often in the same region
  - Mirror server is often in the same country/continent
- Content management
  - A proxy adaptively caches popular streams from different servers based on clients' interest
  - A CDN server statically replicates content of a single server

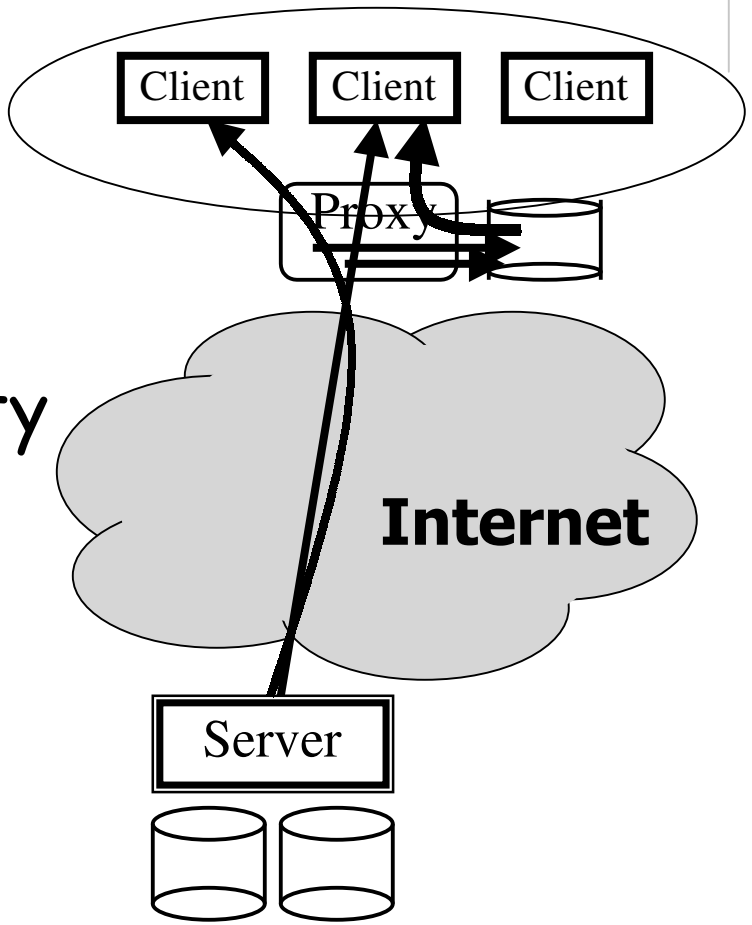
Proxy Caching, CDN, Mirror II

- Administration
  - Clients administer proxy caches (CC at UO)
  - CDN providers maintain CDN servers
    - Content-providers have to pay for this service
    - e.g. Akamai distributes CNN content
  - Content-providers often maintain Mirror servers
- Other Issues
  - Mirror servers should be kept consistent

➢ These approaches *are complementary*

Multimedia Caching: The idea

- Exploit layered organization
  - Relay on a ***cache miss***
  - Pre-fetch on a ***cache hit***
    - If higher quality is required
- Stream quality gradually increases with its popularity
- Proxy is able to properly glue intranet & Internet transport
- Desired quality is **not** guaranteed



Design Issues

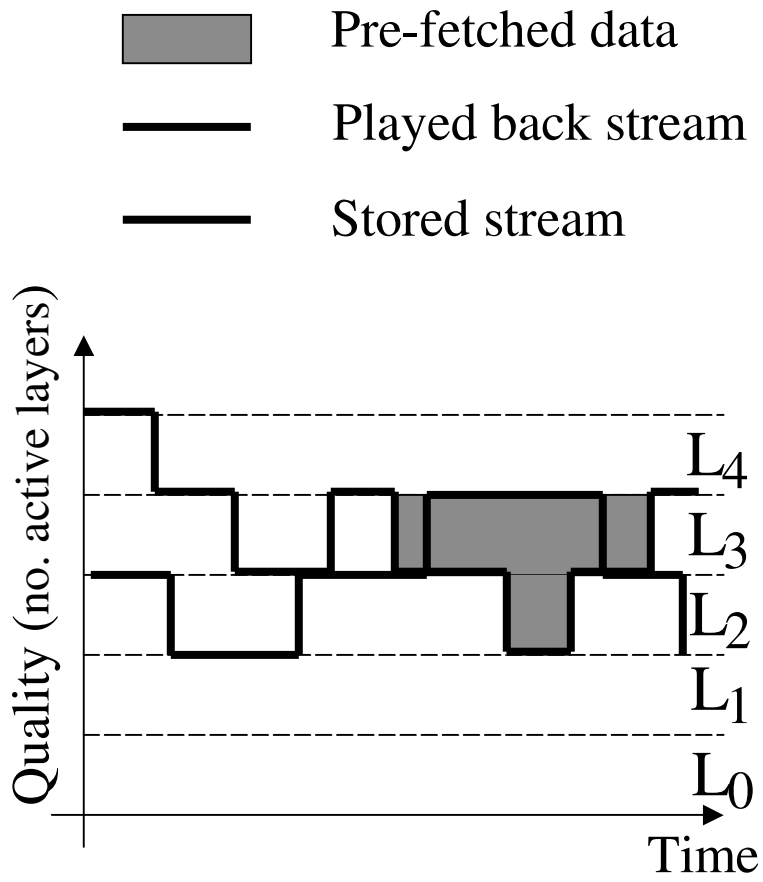
- Assuming locality of reference exists
- Goal: Cache popular streams with appropriate quality
  - Streaming from the cache => reducing network load => improving Scalability
  - Caching higher layers => Improving delivered quality
- Appropriate quality is determined by
  - Client bandwidth
  - Popularity of streams

Main Components

- Fine-grain prefetching
  - To prefetch only the pieces that are required
- Fine-grain cache replacement
  - Streaming objects are larger than Web object
  - Proxy often replaces pieces (e.g. layers, segments) of a cached stream instead of the entire object
    - Higher caching efficiency (+)
    - Which piece to replace (-), Need new popularity functions.
    - Partially cached layers (-)
- Ideally, the cache only contains popular streams with right quality

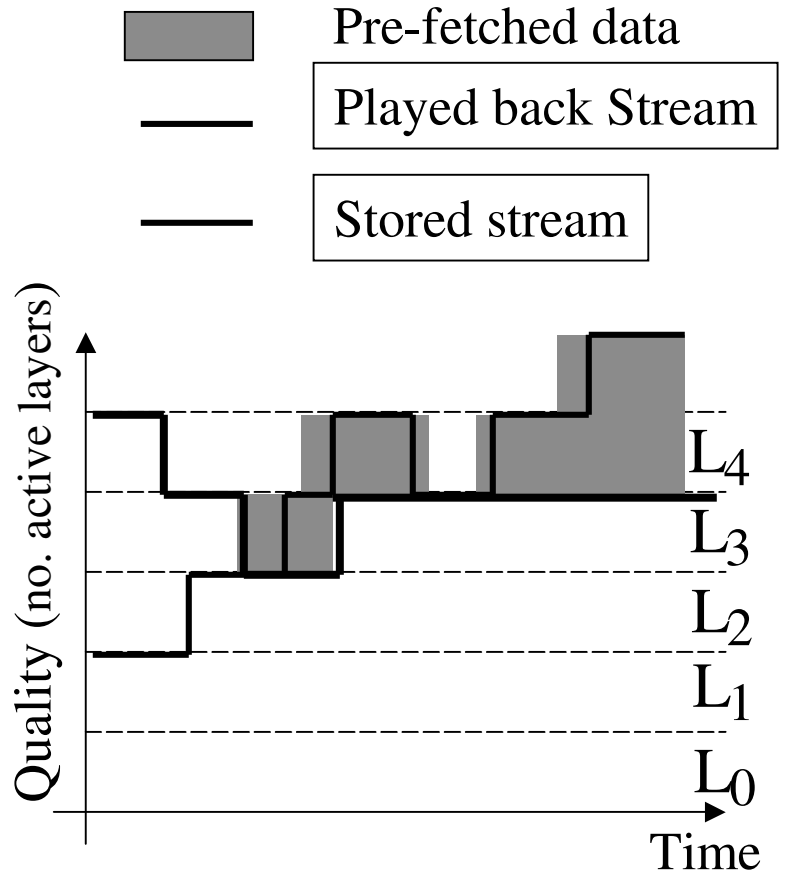
Playing Lower Quality

- Missing pieces of the active layers are pre-fetched on-demand
- Required pieces are identified by QA
- Prefetching smoothes out quality of cached streams



Playing Higher Quality

- Pre-fetch higher layers on-demand
- Pre-fetched data is always cached
- Must pre-fetch a missing piece before its playback time
- Tradeoff



Extending Existing Web Caches?

- Notion of "quality" for cached objects does not exist in Web caching
  - Atomic delivery
  - Atomic replacement
- Existing multimedia caches
  - A web cache + a media player
- Two possible extensions:
  - LQC: Cache and replay low-quality streams
  - HQC: Pre-fetch max. quality of poplar streams
- ✓ *Existing Web caches can not support Mcaching efficiently*