Scalable and Fault-Tolerant Video-on-Demand Systems

Jack Yiu-bun Lee and Po-choi Wong
Department of Information Engineering
The Chinese University of Hong Kong

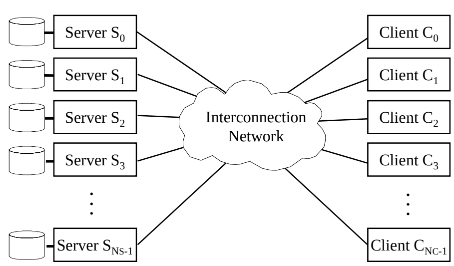
0. Contents

1. Background and Motivation

- Conventional Video-on-Demand (VoD) Systems
  - Based on single-server architecture.
    [e.g. Tobagi 1992, Rangan 1992, etc.]
  - Replication is needed to increase capacity.
    [e.g. Stoller 1995, Schaffa 1995, Barnett 1996].
  - Cannot survive server failures.
- Motivation
  - Disk array [Salem & Garcia-Molina 1986]
  - Tape array [Drapeau & Katz 1993]
  - Network striping [Brendan *et al*. 1995]
  - Server array? [Lee & Wong 1995]

2. Server Array - Architecture

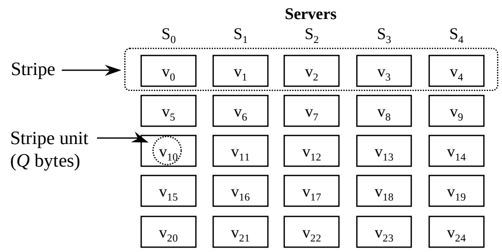
- Network Architecture
  - Many-to-many topology:



  - Autonomous servers and clients;
  - A client linked with all servers.

2. Server Array - Architecture

- Storage Architecture
  - Server Striping:

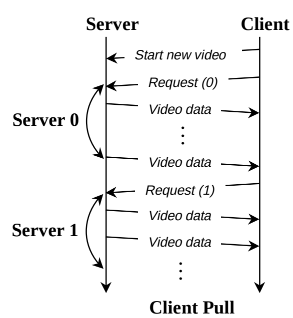


  - Fine-grain load sharing;
  - Independent of video retrieval skewness;
  - No data replication.

2. Server Array - Architecture

Jack Y.B. Lee

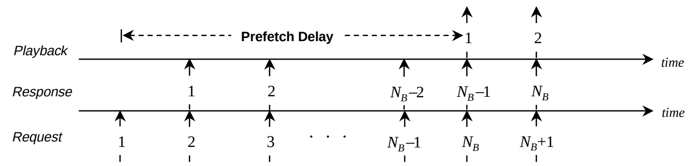
- Service Model
  - Client-Pull with Credit-Based Flow Control:



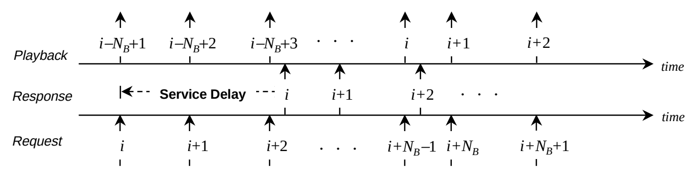
- No need for server synchronization;
- Truly autonomous servers;
- Simpler server design.

2. Server Array - Architecture

- Service Model
  - Client-Pull with Credit-Based Flow Control:
    - Prefetch Phase:

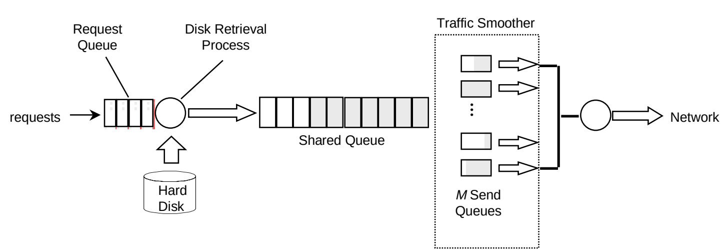


    - Playback Phase:

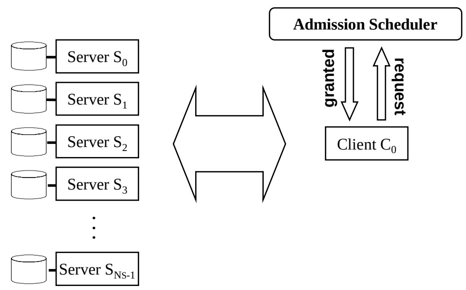


2. Server Array - Architecture

- Server Architecture
  - Request Processing Model:

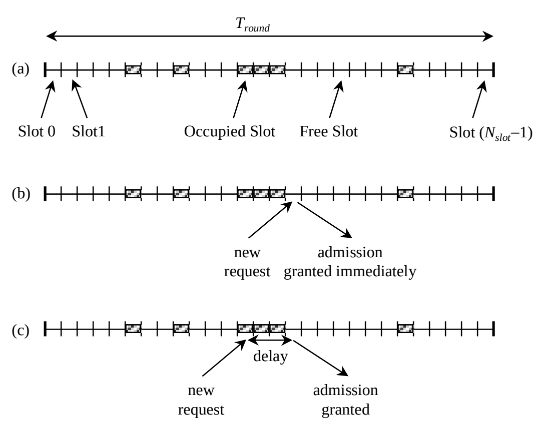


- Server Architecture
  - Admission Scheduling
    - To prevent instantaneous load imbalance.



2. Server Array - Architecture
Jack Y.B. Lee

- Server Architecture
  - Admission Scheduling
    - Start-Time Staggering:



2. Server Array - Performance Modeling

- Admission Scheduling Delay
  - Given: n - Number of current video sessions;
    N_{slot} - Max. number of video sessions;
  - It can be shown that:

$$\text{Prob \{wait k slot times\}} = V_k = \Pr\{(k+1)^{th} \text{ slot free} \mid P_k\} P_k$$
$$= \left(\frac{N_{slot}-n}{N_{slot}-k}\right)\prod_{i=0}^{k-1}\left(\frac{n-i}{N_{slot}-i}\right) \qquad 1 \le k \le n$$

Hence the average scheduling delay under a given system utilization (n/N_{slot}) can be found accordingly.

2. Server Array - Performance Modeling

- Maximum Service Delay
  - The maximum time from the server received a request to the time a complete response (a video data block) is transmitted.

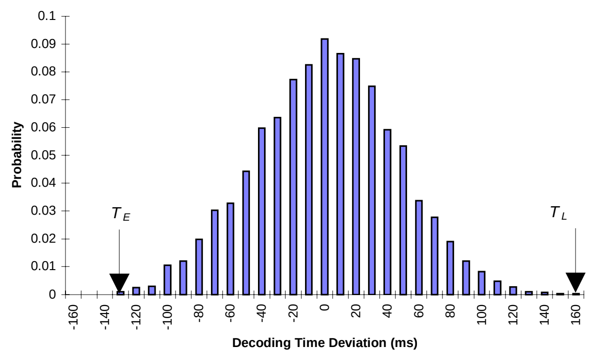
Model the request generation process	Theorem 1, 2, 3
⇓	
Derive the max. delay $D_{disk}^{\max}$ in disk retrieval process	Theorem 4
⇓	
Derive the max. delay $D_{net}^{\max}$ in transmission process	Theorem 5, 6, 7
⇓	
Max. service delay $D_{\max} = D_{proc}^{\max} + D_{disk}^{\max} + D_{net}^{\max}$	For simplicity, we assume zero processing delay in this study.
⇓	

Study the system performance when scaling up to more servers.

2. Server Array - Performance Modeling

Jack Y.B. Lee

- The Request Generation Process
  - Having an average rate with bounded jitter,
    e.g. Sigma Designs RealMagic MPEG-1 Decoder:



2. Server Array - Performance Modeling

Jack Y.B. Lee

- The Request Generation Process
  - Without admission scheduling:

Theorem 1 *The decoding time t between any two requests, i and j (j>i) is bounded by*

$$\max\{((j-i)T_{avg} - T_{DV}),0\} \le t \le ((j-i)T_{avg} + T_{DV})$$

Theorem 2 *Assume n clients generating requests independently and each client sends requests to the N_S servers in the system in a round-robin manner, then the minimum time for a server to receive k video data requests is given by*

$$T_{Request}^{\min}(k,n) = \max\left\{ \left(\left\lceil \frac{k}{n} \right\rceil - 1 \right) N_S T_{avg} - T_{DV}, 0 \right\}$$

Note that: $T_{Request}^{\min}(k,n) = 0 \quad \forall k < n$

This is the worst case where all *n* clients sends requests to the same server simultaneously when there is no admission scheduling.

- The Request Generation Process
  - With admission scheduling using start-time staggering:

Theorem 3 *If the admission scheduler is used with parameters $T_{round}=N_S T_{avg}$ and there are n clients, then the minimum time for a server to receive k video data requests is given by*

$$T_{Request}^{\min}(k,n) = \begin{cases} \max\left\{\lfloor k/n \rfloor N_S T_{avg} - T_{DV}, 0\right\}, & \mathrm{mod}(k,n)=1 \\ \max\left\{(w(N_{slot}-n)+k-2)\dfrac{N_S T_{avg}}{N_{slot}} - T_{DV}, 0\right\}, & \text{otherwise} \end{cases}$$

where $w = \left\lceil \dfrac{k}{n} \right\rceil - 1$.

With admission scheduling, requests from different clients are spreaded out to avoid the undesirable requests-synchrony problem.

- Delay in the Disk Retrieval Process
  - Disk Model:

Assumption 1 *We assume the minimum time to read a block of Q bytes from the disk, denoted by $T_{read}^{\min}$, is known.*

Assumption 2 *We assume the maximum time to read k blocks of Q bytes from the disk, denoted by $T_{read}^{\max}(k)$, is known and the function is non-decreasing with respect to k.*

Definition 2 *Define $N_{Disk}(t)$ as the minimum number of requests serviced in a time interval t during a busy period. Then we can calculate it from $T_{read}^{\max}(k)$:*

$$N_{Disk}(t) = \max\{k \mid T_{read}^{\max}(k) < t, \forall k \ge 0\}$$

Definition 3 *Define $N_{Request}(t,n)$ as the maximum number of requests generated in a time interval t by n video clients. We can derive it from (14):*

$$N_{Request}(t,n) = \max\{k \mid T_{Request}^{\min}(k,n) < t, \forall k \ge 0\}$$

- Delay in the Disk Retrieval Process
  - Maximum Delay:

Theorem 4 *The maximum number of requests that can coexist in the disk subsystem is given by*

$$L_D = \max\{N_{Request}(t,n) - N_{Disk}(t) \mid \forall t\}$$

Substituting L_D into $T_{read}^{\max}(k)$ we can obtain the maximum delay for any request to complete service in the disk subsystem:

$$D_{disk}^{\max} = T_{read}^{\max}(L_D)$$

- Delay in the Transmission Process

Theorem 5 *For a request arrives at the shared queue to find k−1, (k≥1) requests already in the system, the delay for this newly arrived request to complete service at the traffic smoother is bounded from above by*

$$T_{tx}^{\max}(k) = \begin{cases} \dfrac{MQ}{C_S} & \text{if } k \le M \\ \dfrac{(k+M)Q - MY}{C_S} & \text{otherwise} \end{cases} \qquad (19)$$

Definition 4 *Define $N_{tx}(t)$ as the minimum number of requests serviced in a time interval of t during a busy period. We can calculate it using Equation (19):*

$$N_{tx}(t) = \max\{k \mid T_{tx}^{\max}(k) < t, \forall k \ge 0\}$$

- Delay in the Transmission Process

Theorem 6 *Assume there are n video clients generating video requests simultaneously and the maximum number of requests in the disk subsystem is L_D, then the maximum number of requests departing from the disk subsystem in a time interval t is*

$$N_{out}(t,n) = \min\left\{\left\lceil \frac{t}{T_{read}^{\min}} \right\rceil, L_D + N_{Request}(t,n)\right\}$$

Theorem 7 *The maximum number of requests that can coexists in the network subsystem is given by*

$$L_N = \max\{N_{out}(t,n) - N_{tx}(t) \mid \forall t\}$$

Hence $D_{\max}^{net} = T_{tx}^{\max}(L_D)$

- Client Buffering and Video Playback Continuity
  - A client has N_B buffers, of which (N_B–1) are prefetched before playback starts.
  - Let T_i be decoding time and F_i be the time for video block i to be decoded, then continuity implies:

$$F_i \le T_i \qquad \forall i$$

  - Using Theorem 1 and expressing F_i in terms of T_{avg}, etc, we have:

$$(i - N_B + 1)T_{avg} + T_L + D_{\max} \le iT_{avg} + T_E$$

or

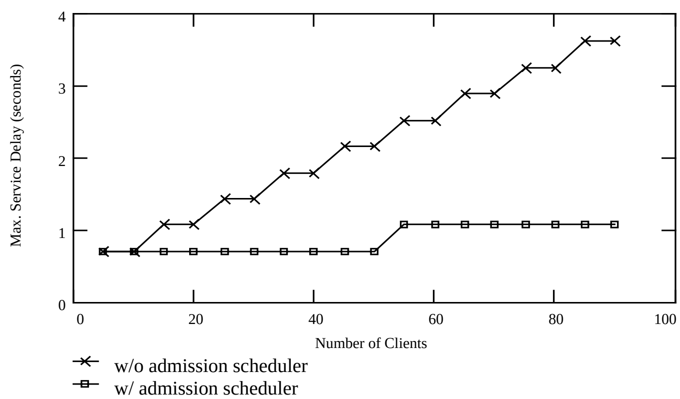
$$N_B \ge \frac{D_{\max} + T_{DV}}{T_{avg}} + 1$$

  - Prefetch delay: $D_{Prefetch} = (N_B - 1)T_{avg} + D_{\max}$

- The system parameters below are used to compute numerical results for the performance model:

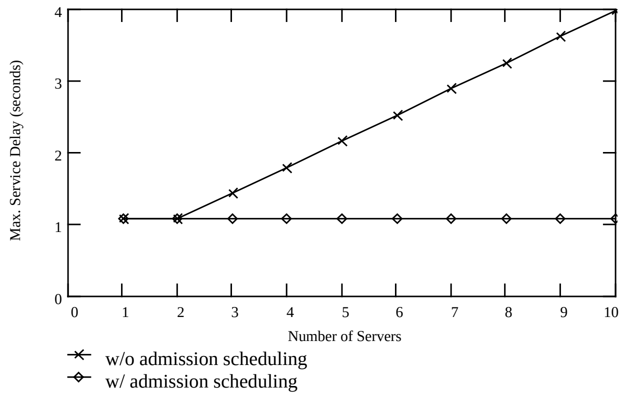
System Parameters	Symbol	Value
Spindle speed	n/a	5411 rpm
Max latency	$T_{latency}$	11ms
Number of tracks	N_{track}	2621
Raw transfer rate	R_{disk}	3.35MB/s
Single-track seek	n/a	1ms
Average seek	n/a	10ms
Max full-stroke seek	n/a	19ms
Video packet size	Y	8192 Bytes
Video block size	Q	65536 Bytes
Video data rate	R_V	150KB/s
Number of send queues in traffic shaper	M	10
Raw disk transfer rate (Seagate ST12400N)	R_{disk}	3.35MB/s
Average video block decoding time	T_{avq}	437ms
Maximum early in decoding time	T_E	-130ms
Maximum late in decoding time	T_L	160ms
Effective network throughput	C	1.875MB/s
Maximum processing delay	$D_{\max}^{proc}$	0ms

- Service Delay v.s. No. of Concurrent Clients
  - 8 Servers:



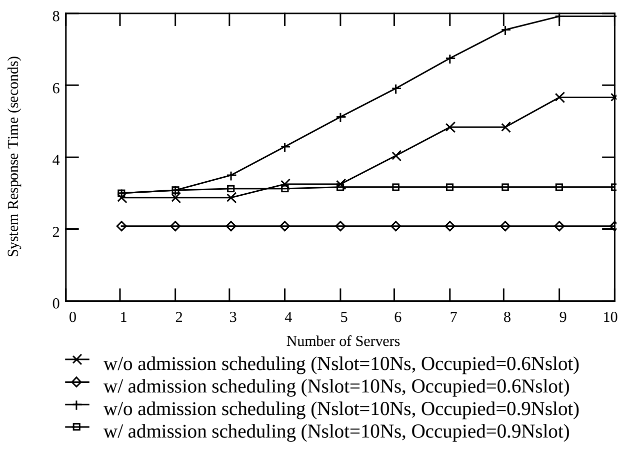
2. Server Array - Performance Evaluation

- Service Delay v.s. No. of Servers
  - Client-server ratio = 10:



2. Server Array - Performance Evaluation

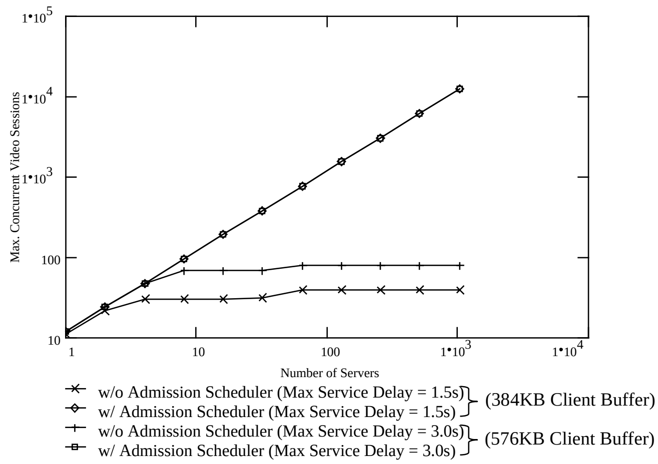
- System Response Time v.s. No. of Servers



2. Server Array - Performance Evaluation

Jack Y.B. Lee

- Scalability Under Delay Constraint



3. Redundant Array of Inexpensive Servers

Jack Y.B. Lee

- Server-Level Fault-Tolerant
  - To sustain video service and to guarantee video playback continuity despite *server failures*.
- Existing Approaches
  - Server replication
  - Data partitioning
- Redundant Array of Inexpensive Servers [Lee & Wong 1997]
  - Storage policy for data redundancy
  - Transmission policy for data redundancy

3. Redundant Array of Inexpensive Servers

Jack Y.B. Lee

- Storage Policy
  - Server striping with *redundancy*
  - N_S - number of servers, K - level of redundancy, $N_D = N_S - K$
  - E.g. N_S=5, K=1:

	S_0	S_1	S_2	S_3	S_4	Parity Calculation
Stripe	v_0	v_1	v_2	v_3	p_0	$p_0 = v_0 \oplus v_1 \oplus v_2 \oplus v_3$
	v_4	v_5	v_6	p_1	v_7	$p_1 = v_4 \oplus v_5 \oplus v_6 \oplus v_7$
Stripe unit	v_8	v_9	p_2	v_{10}	v_{11}	$p_2 = v_8 \oplus v_9 \oplus v_{10} \oplus v_{11}$
	v_{12}	p_3	v_{13}	v_{14}	v_{15}	$p_3 = v_{12} \oplus v_{13} \oplus v_{14} \oplus v_{15}$
Parity unit	p_4	v_{16}	v_{17}	v_{18}	v_{19}	$p_4 = v_{16} \oplus v_{17} \oplus v_{18} \oplus v_{19}$

3. Redundant Array of Inexpensive Servers

Jack Y.B. Lee

- Storage Policy
  - Lost-data recovery by erasure-correction (*at the client*).
  - E.g. N_S=5, K=1:

	S_0	S_1	S_2	S_3	S_4	Recovering Lost Units
Stripe	v_0	v_1	v_2	v_3	p_0	$v_2 = v_0 \oplus v_1 \oplus v_3 \oplus p_0$
	v_4	v_5	v_6	p_1	v_7	$v_6 = v_4 \oplus v_5 \oplus p_1 \oplus v_7$
Stripe unit	v_8	v_9	p_2	v_{10}	v_{11}	*not needed*
	v_{12}	p_3	v_{13}	v_{14}	v_{15}	$v_{13} = v_{12} \oplus p_3 \oplus v_{14} \oplus v_{15}$
Parity unit	p_4	v_{16}	v_{17}	v_{18}	v_{19}	$v_{17} = p_4 \oplus v_{16} \oplus v_{18} \oplus v_{19}$

Server S_2 failed

3. Redundant Array of Inexpensive Servers

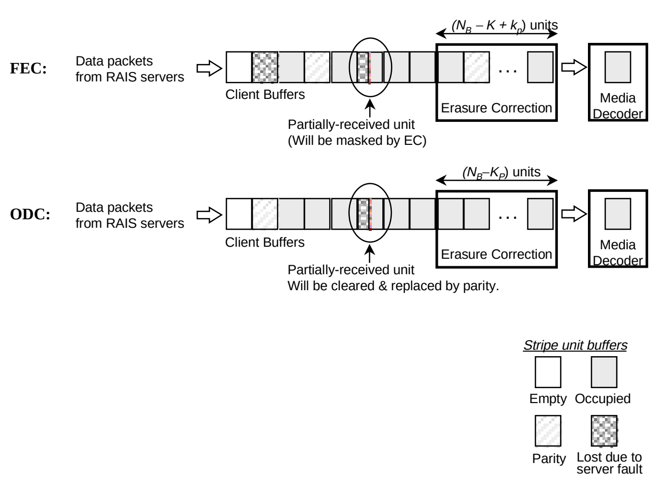
Jack Y.B. Lee

- Transmission Policy
  - Forward Erasure Correction (FEC)
    - Always transmits K_P (out of K) parity data (even when no server fails).
    - Extra bandwidth and processing needed at normal mode.
    - Overhead is $K_p/(N_S–K)$.
      E.g. N_S=4, K=K_p=1, then 33% overhead.
  - On Demand Correction (ODC)
    - Transmits parity data only when a server fails.
    - No extra bandwidth and processing overhead needed at normal mode.
    - Relies on network protocol to *detect* server failures quickly and to reconfigure the system to failure-mode operation.

3. Redundant Array of Inexpensive Servers

Jack Y.B. Lee

- Client-Side Lost-Data Recovery



3. Redundant Array of Inexpensive Servers

- Client Buffer Requirement
  - Forward Erasure Correction

$$N_B \geq n + \left\lceil \frac{n}{N_S - K} \right\rceil K_P$$

$$where \qquad n \geq \frac{(D_{\max} + T_{DV})}{T_{avg}} + N_S - K$$

  - On-Demand Correction

$$N_B \geq \frac{(2D_{\max} + T_{fault} + T_{DV})}{T_{avg}} + N_S - K$$

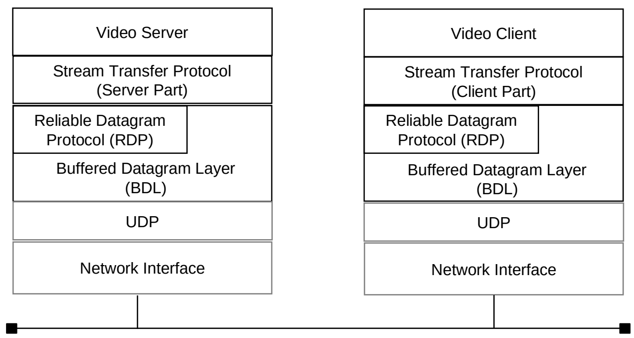
4. System Implementation Results

- Software Platform
  - Development Tool : C++
  - Operating System
    - Video Server : Windows NT
    - Video Client : Windows 3.1, Windows 95, and Windows NT
  - Disk I/O
    - Multi-threaded Asynchronous File I/O
  - Network I/O
    - UDP via Windows Sockets

4. System Implementation Results

Jack Y.B. Lee

- Software Architecture

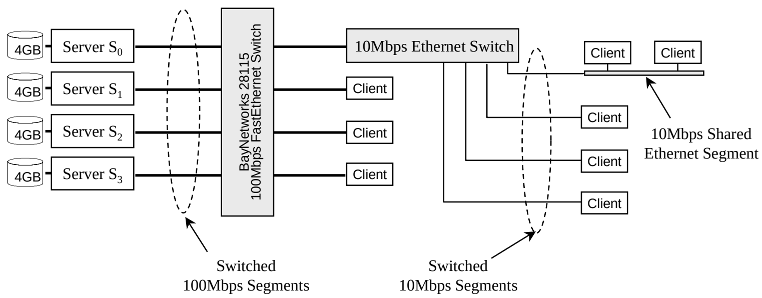


4. System Implementation Results

Jack Y.B. Lee

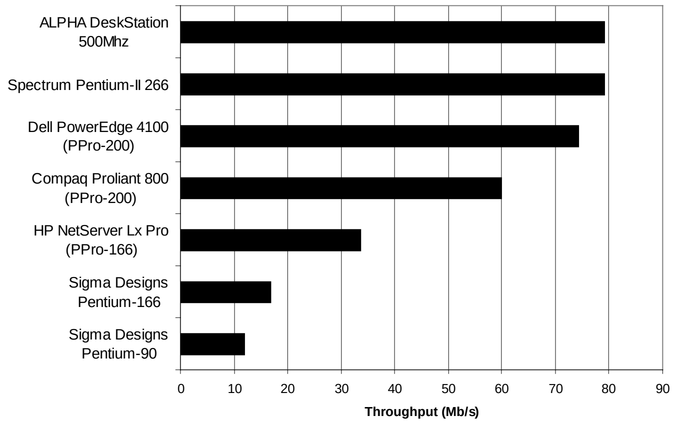
- Hardware Platform
  - Server and client : Standard PCs, DEC Alpha
  - Network : Ethernet, FastEthernet, and ATM (LANE)
  - Disk : Standard SCSI-II, UW-SCSI, etc.
  - Video Decoder
    - Sigma Designs RealMagic (MPEG-1)
    - Microsoft ActiveMovie (MPEG-1, AVI)
    - Optibase VideoPlex (MPEG-2)

- Test-bed Configuration
  - One to four servers

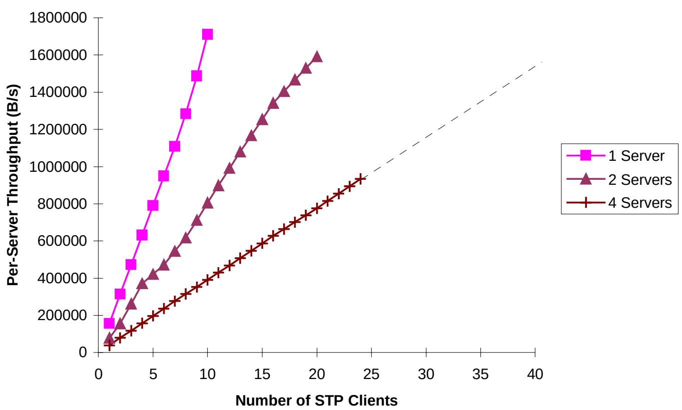


- Test-bed Configuration
  - Block size: Q = 65536 bytes
  - Measured max. service delay: D_{max} = 850 ms
  - Peak-to-peak decoding variation: T_{DV} = 290 ms
- Client Buffer Requirement
  - N_B=4 (256KB) for normal mode playback continuity.
  - N_B=8 (512KB) for FEC failure-mode playback continuity.
  - N_B=10 (640KB) for ODC failure-mode playback continuity.

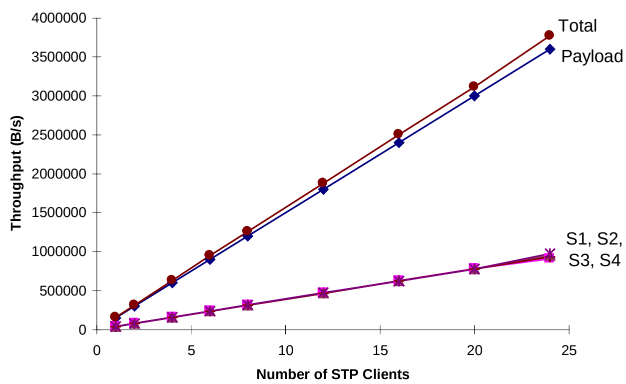
- Single-Server Capacity



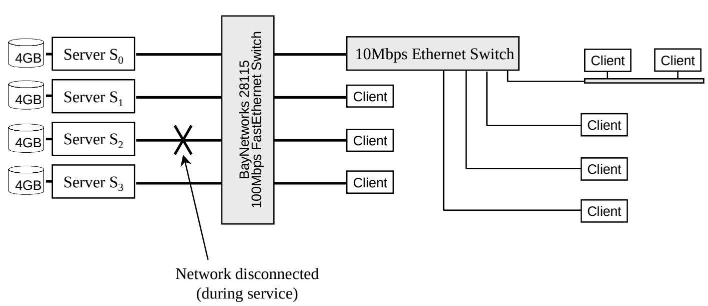
- Scalability
  - Server: Pentium-90, 32MB RAM



- Load Sharing (4 servers)

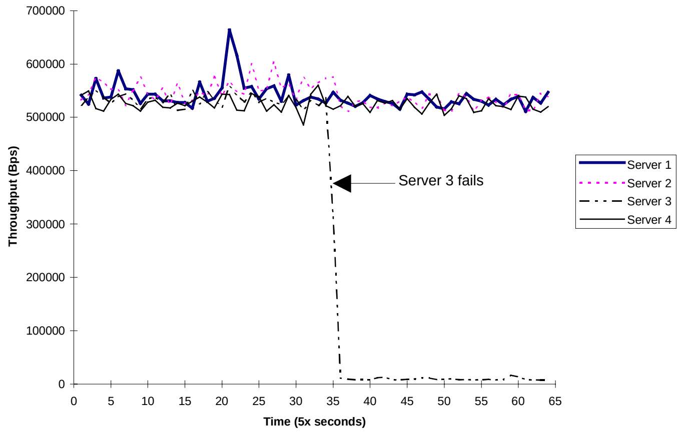


- Fault-Tolerance Experiment
  - 4-servers RAIS, with redundancy *K*=1.
  - Test load: 10 concurrent video streams (@ 1.2Mbps MPEG-1).
  - Server failure simulated by disconnecting network cable.
  - In all cases, video playback continuity is preserved at all clients.



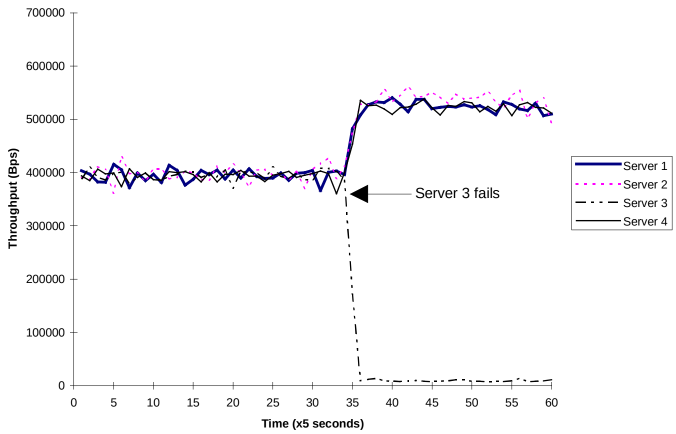
4. System Implementation Results

- Fault-Tolerance Experiment
  - Server Throughput versus Time (FEC)



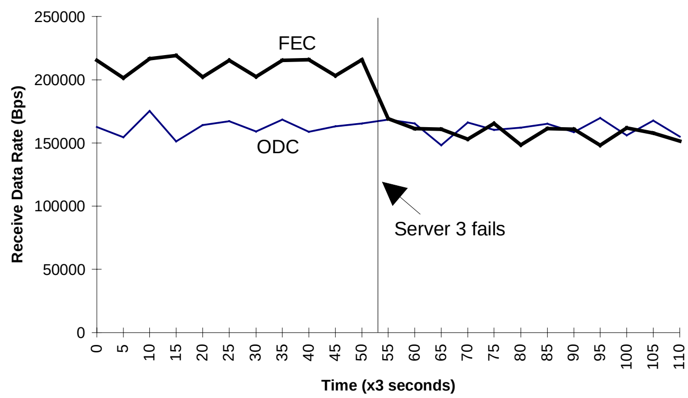
4. System Implementation Results

- Fault-Tolerance Experiment
  - Server Throughput versus Time (ODC)



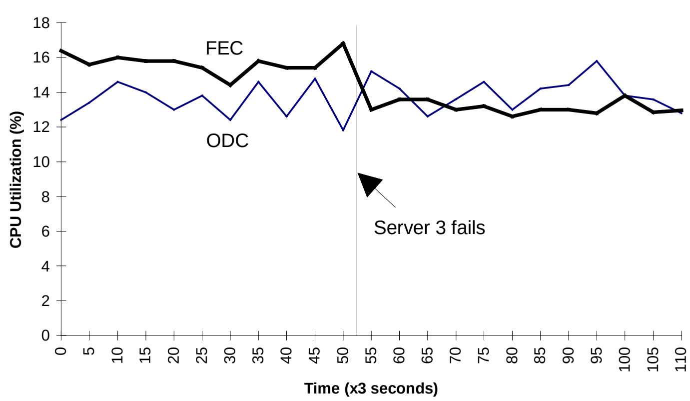
4. System Implementation Results

- Fault-Tolerance Experiment
  - Client Received Data Rate versus Time



4. System Implementation Results

- Fault-Tolerance Experiment
  - Client CPU Utilization versus Time



5. Future Research Directions

- Online Automatic System Repair
  - Rebuilds data in a failed server to a spare server.
- Push-Based Parallel Video Server
  - Server synchronization algorithms;
  - Server scheduling algorithms;
  - Failure-detection and redundancy transmission algorithms;
  - Integration with ATM QoS mechanisms.
- Scalable and Fault-Tolerant Parallel Web Server
  - Efficient striping algorithms for web objects;
  - Architecture and protocol to coexist with existing web servers and web browsers;
  - Performance analysis and scalability limits.