中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-05-014

Video Delivery on Content Networks

Ray-I Chang, Jan-Ming Ho

September 2005 || Technical Report No. TR-IIS-05-014

http://www.iis.sinica.edu.tw/LIB/TechReport/tr2005/tr05.html

Video Delivery on Content Networks

Ray-I Chang
Department of Engineering Science
National Taiwan University
Taipei, Taiwan, ROC.

Jan-Ming Ho
Institute of Information Science
Academia Sinica
Taipei, Taiwan, ROC.

Abstract. Behaviors of Internet services are changing from computation sharing to content sharing. Due to these changes, the weakness of conventional network architecture and its inadequate in service support are apparent. To resolve these problems, content networks are introduced. A content network tries to reorganize the Internet by a content-centric way to improve the system's scalability and to protect the system from distributed denial-of-service attacks. It was taken into one of the most important platform of network applications (such as e-commerce) in the future. In this paper, we investigate some key technology issues of content delivery of streaming multimedia, which has both the biggest challenge and opportunity. Different from small-sized hypertext and image, the growing-popular multimedia content has huge size. It is impractical to be cached entirely and requires to be partitioned into small portions for delivery from multi-servers. As a multimedia content is VBR (variable-bit-rate) and requires guaranteed QoS (quality-of-service), each serving-peer needs a good delivery schedule for real-time streaming. In this paper, we focus only on the multi-servers caching/streaming problem of multimedia delivery. Without addressing content indexing and routing, the proposed method can be applied for different content networks.

1 Introduction

1.1 Content Provider and Content Network Provider

Nowadays, the behaviors of Internet services are changing from computation sharing to content sharing. The weakness of conventional network architecture and its inadequate in service support are apparent. To improve the content provider's scalability and to protect it from DDoS (distributed denial-of-service) attacks, content network architecture is introduced to re-organize the Internet by a content-centric way. Traditional content providers have hosted their own content and managed their own Internet connections. To guarantee QoS (Quality-of-Service), content providers need fixed connections and usually pay based on their peak bandwidth. However, users usually request content at peak hours while the network remains unused during other parts of the day. To efficiently serve local content around the world, content providers need to run multiple data centers as shown in Fig. 1(a). This burst bandwidth utilization pattern wastes network capacity and increases service costs. It makes the build-it-yourself alternative less cost-effective.

As the demands of content distribution increase, *content providers* turn to *content network providers* (see Fig. 1(b)) to increase the performance and reliability of services and lower their total cost of ownership. Content network providers host replicas of content in cache servers located within network edge that is just one hop away from the user. A user request to a content provider is redirected to a cache server of the content network provider that is close (geographically or shortest travel time) to the user and is the least busy. To determine the cache server that is most available to a user, content network providers make use of load-balancing technology to direct traffic to the least-loaded server. By leveraging strategies of distributed caching, load balancing and request redirection systems, content is ensured to be served up in the most efficient manner based on user proximity and server load. The end users (and by association, the content providers), as well as the ISP (Internet Service Provider), are benefited.

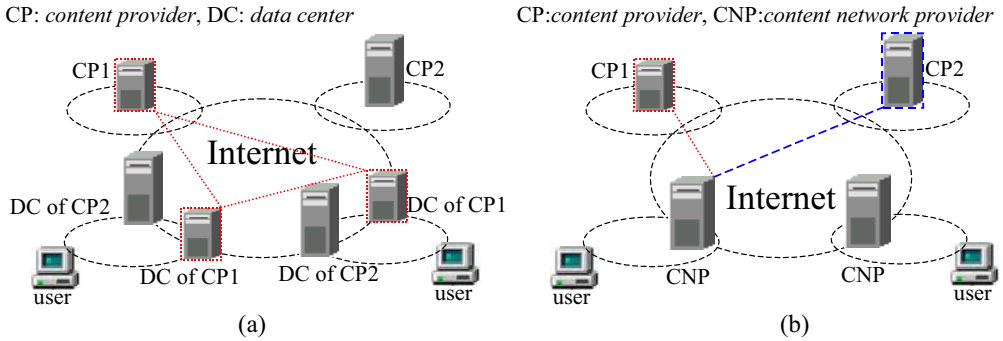

Fig. 1. (a) Traditional content providers have hosted their own content and managed their own Internet connections to run multiple data centers. (b) As the demands of content distribution increase, content providers turn to content network providers to increase the performance and reliability of services and lower their total cost of ownership.

1.2 Architecture of Content Networks

Content providers are looking to the content network as a high-performance and reliable vehicle for delivering bandwidth and delay intensive content. A content network (also called content distribution network or content delivery network) is a system, often an overlay network to the Internet, built for the high-performance and mission-critical service of contents. It requires high service availability, fast response time, and cost-effective scalability to reach the broadest possible set of intolerant users. At a minimum, it requires the following functional components that work together to accomplish these goals (see Fig. 2):

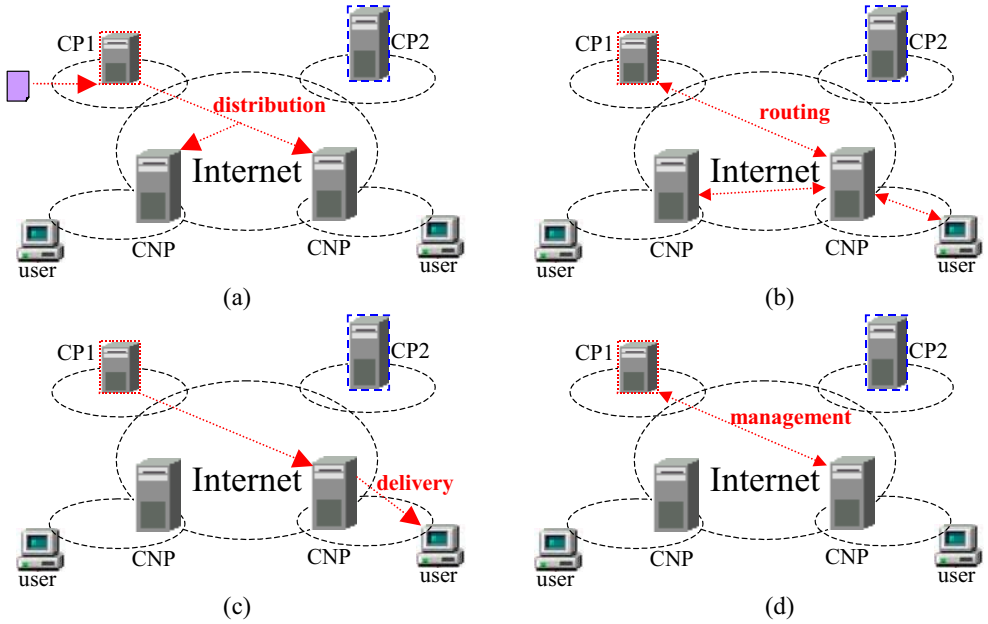

Fig. 2. (a) Content distribution service. (b) Content routing service. (c) Content delivery service. (d) Content management service. (CP = content provider, CNP = content network provider)

- *Content distribution service* to comprise a set of cache servers that cache content and work on behalf of a content provider's origin server
- *Content routing service* to bypass congested areas of the Internet by directing a user's request to the closest and most available cache server
- *Content delivery service* to deliver the best possible experience to users who are notoriously intolerant of response-time delays
- *Content management service* to refresh content and track user activity

Finally, the *accounting service* is provided to measure, log and bill the content provider based on the amount of bandwidth consumed (also, to cross-bill one another for internetworking services).

In past years, different content networks were introduced. They were usually classified on the basis of their attributes in content aggregation and placement. Content aggregation is a process of mapping contents to values in some value space and grouping them on the basis of their mapped values. It can be "semantic" or "syntactic." (Content networks that use semantic to distribute/route individual contents to content groups are called *semantic content networks*.) After content aggregation, a piece of content or a content group is placed at the network node that is either a function of the content (called "content-sensitive") or independent of it (called "content-oblivious"). The placement strategy affects optimization of content routing and the size of routing tables.

1.3 Peer-to-Peer Content Networks

In conventional content networks, content is served from a special cache server at the network edge. This architecture is poor in system scalability. Recently, the popularity of Napster and Gnutella (the preliminary of Peer-to-Peer file sharing programs) has made it clear that peer-to-peer (P2P) architecture will be one of the most important platforms of network applications in the future. In a peer-to-peer content network as shown in Fig. 3, every user that receives a file would be available to serve it on to another user when new requests are received at the next time. Demands on central servers can be further reduced if users can receive a particular file efficiently from other users who have the same file (or parts of the same file). Such a serving-peer can be the central server of content provider, a cache server at ISPs, a caching proxy on near campus or the neighbor's computer in the next office. The more computers that request the content, the more computers that are available to serve the content. This means smooth downloads for end users are less hampered by traffic congestion on the core Internet. Additionally, since not every copy of a file is downloaded from central servers, content providers can reduce demands for bandwidth with lowered costs. In ISPs, performing file distribution internally across their network boundaries can further enhance efficiency. Peer-to-peer networks can

be "centralized" or "decentralized" in content routing. A decentralized system (e.g., FreeNet and eDonkey) would have better scalability, however, it needs a good design of routing table.

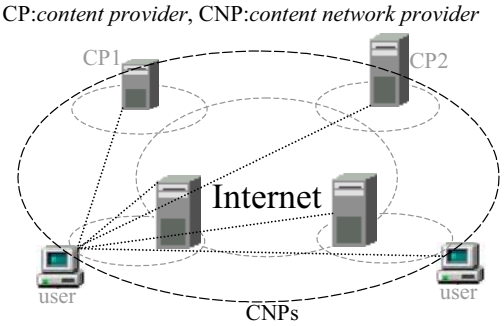

Fig. 3. In a peer-to-peer content network, a serving-peer can be the central server of content provider, a hardware-based server at ISP, a proxy server on near campus or the neighbor's computer in the next office.

In this paper, we focus only on the multi-servers caching/streaming problem of multimedia delivery over P2P content networks. This problem has both the biggest challenge and opportunity. Traditional Internet proxies consider only hypertext and image. They have small size and can be entirely cached in the proxy. However, the growing-popular multimedia contents have huge size and the serving-peers may provide only a limited storage space for caching. It is impractical to cache them entirely and requires partitioning each of them into small portions. As a multimedia content is VBR and requires guaranteed QoS, each serving-peer also needs a good delivery schedule for real-time streaming. In this paper, some new technologies to enable cost-effective content delivery of streaming media are introduced. Without addressing content indexing and routing, the proposed method can be applied on the multi-servers caching/streaming problem for different content networks.

2 Problem Definition

Notably, based on a list of predicted or reserved requests, we can automatically deliver contents at off-peak hours to fill in the troughs of bandwidth consumption. Therefore, bandwidth usage over the day is smoothed out and peak bandwidth usage is thus reduced. It lowers the cost content providers pay for every megabyte they distribute. As deliveries happen automatically in the background when users are not using their machines, users do not have to tie up their computers during regular working hours with content downloads. Actually, we can push a content through the relay cache form an on-demand cache that can deliver the content thereafter to whomever requests it, whenever they request it. If there are many users requested the same content, the best delivery paths (or trees) can be identified to deliver digital media to end users with the minimum cost. It is cost effective for content providers to deliver larger, higher-quality multimedia contents to broad audiences. In this paper, we base on this service model to consider the multi-servers caching/streaming problem.

2.1 Content Partitioning and Placement

Multimedia delivery has the most challenge due to its large size and critical real-time requirement. Usually, a multimedia content $f(.)$ can be defined as a sequence of data frames

$$f(.) = \{ f(t) \mid t = 0, 1, \ldots, n\text{-}1 \} \quad (1)$$

where $f(t)$ is the data frame played at time t. Its size is $|f(.)| = \sum_t \{ |f(t)| \}$. Assume that there are m+1 peers suggested for caching (by the applied content indexing and routing scheme) as shown in Fig. 4. For each video $f(.)$, we place a portion

$$f_k(.) = \{ f_k(t) \mid t = 0, 1, \ldots, n\text{-}1 \} \text{ (for } k = 0, 1, \ldots, m) \quad (2)$$

of video $f(.)$ in peer k. Its size is $|f_k(.)| = \sum_t \{ |f_k(t)| \}$. Notably, $f_k(t)$ is just a portion of $f(t)$ for any time t. (The value $f_k(t)$ is the same as $f(t)$ in conventional proxy caches where the entire content is stored.) In this paper, we define $f(t) = \cup \{ f_k(t) \mid \text{for } k = 0, 1, \ldots, m \}$ and $f_p(t) \cap f_q(t) = \varnothing$ for $p \neq q$ (there is no overlap between any two partitioning).

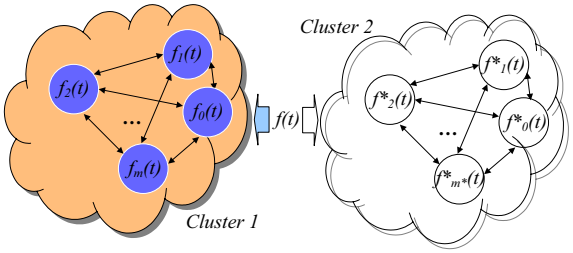

Fig. 4. In a system with lots of peers, we can place multiple copies of the multimedia content to different network clusters. For each network cluster, a video *f(.)* is partitioned for placing into selected peers.

The problem to decide $f_k(.)$ for the given constraints and cost functions (usually, the cache size and/or the network bandwidth) on each peer is called *content partitioning and placement*. As the placed content may be migrated or replicated due to its usage after this step, it is also called *initial* partitioning and placement. While considering a system with lots of peers, we can try to place multiple copies of multimedia content to different network clusters. It can improve performance and scalability of the system. In this paper, we focus only on the *initial* partitioning and placement problem with one copy of multimedia content. Partitioning models with vertical-cut and horizontal-cut are shown in Fig. 5. The same idea can be extended to place multiple copies of multimedia content and to handle their migration/replication for load balance and fault tolerance.

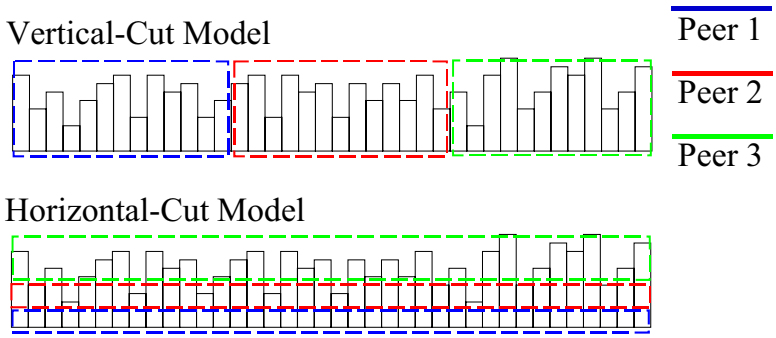

Fig. 5. Partitioning models with vertical-cut and horizontal-cut are shown.

2.2 Adaptive Smoothing with Rate-Availability

Slight network delays may go almost unnoticed for static text-based Web page download. However, for a delay-sensitive multimedia application (such as entertainment services, gaming, live videoconferences and streaming broadcasts) extra steps must be taken to ensure QoS delivery. In this paper, by monitoring the available bandwidth of each connection, an ASRA (Adaptive Smoothing with Rate-Availability) scheme is proposed. Notably, the network bandwidth is shared by different traffics. Therefore, the available rate of network is time-varying. It can be represented by a *rate availability function* (RAF)

$$z_k(.) = \{ z_k(t) \mid \text{for all } t \} \quad (3)$$

where $z_k(t)$ is the available rate of network between time t-1 and time t. For supporting QoS delivery, the delivery schedule must guarantee that its allocated rate $r_k(.)$ satisfies

$$z_k(t) \geq r_k(t) \text{ at any time } t. \quad (4)$$

If the available rate of peer k is not enough to delivery content in real-time, the system can automatically re-select servers to request more data from the peer with the best throughput. Comparing to conventional schemes that didn't consider RAF in traffic smoothing (as shown in Fig. 6(a)), our approach can support more requests with less re-selection of servers. A simple example is shown in Fig. 6(b).

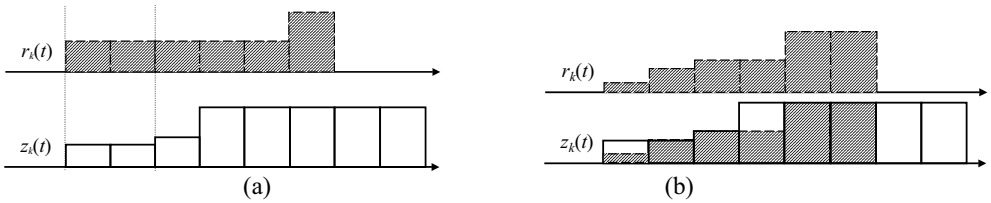

Fig. 6. (a) Conventional schemes didn't consider RAF in traffic smoothing.
(b) Traffic smoothing with RAF can support more requests with less re-selection of servers.

The proposed scheme can aggregate underutilized network capacity through time shifting to streamlined bandwidth consumption. As servers' requested data are adaptive smoothing by their available rates in real time, we can avoid points of congestion on the Internet. Furthermore, if one of the servers shuts down or stops responding, we can automatically requests the rest of the data from other servers. It can provide end users with fault tolerance. By adapting client behavior to the demand pattern of servers, the proposed system can scale limitlessly with the growth of the Internet. It is a scalable solution for network applications with high performance benefit.

2.4 Other Issues

In this paper, to assure high reliability, high performance and efficient bandwidth usage, we let each computer connect to many servers simultaneously. Therefore, a particular file can be obtained by requesting its small parts from different servers. As shown in Fig. 7, when the client peer 0 requests video $f(.)$, it can interact with other peers to delivery parts of their cached portions $g_k(.)$ directly. (Our system uses relay servers in content distribution but not in content delivery.)

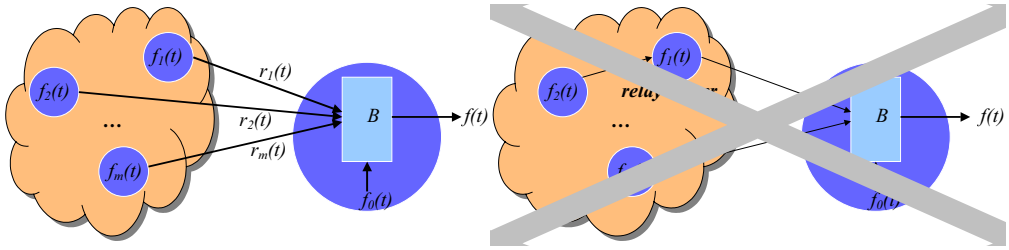

Fig. 7. When the client peer 0 requests video $f(.)$, it interacts with other peers to receive parts of their cached portions directly. There is no relay server in delivery.

Notably, if the requested content has no migration or replication after its initial placement, we can guarantee $f(t) = \cup\{ f_k(t) \mid \text{for } k = 0, 1, \ldots, m \}$ and $f_p(t) \cap f_q(t) = \varnothing$ for $p \neq q$. They imply that $g_k(.) = f_k(.)$ for all k. However, due to the changes of client behavior, the system needs to migrate and replicate the cached content to improvement its performance. There may have some overlaps in cached content between two serving-peers. Without wasting bandwidth in delivery, delivered content $g_k(t)$ can be just a portion of $f_k(t)$ and $g_p(t) \cap g_q(t) = \varnothing$ for $p \neq q$. For supporting jitter-free playback,

$$f(t) = \cup\{ g_k(t) \mid \text{for } k = 0, 1, \ldots, m \}. \quad (5)$$

Let $r_k(.)$ be a feasible transmission schedule of peer k (for $k = 0, 1, \ldots, m$) and B is the available buffer size of the client peer. To prevent buffer overflow and underflow, we have

$$\sum_{t-1}\{ g_k(t) \} + B \geq \sum_t\{ r_k(t) \} \geq \sum_t\{ g_k(t) \}. \quad (6)$$

Without loss of generality, we can assume that $r_0(t) = g_0(t) = f_0(t)$ where the local cached data $f_0(t)$ can be served by a very high rate. Fig. 8 shows a simple example of the server selection problem. The original content can be distributed by either horizontal-cut or vertical-cut partitioning.

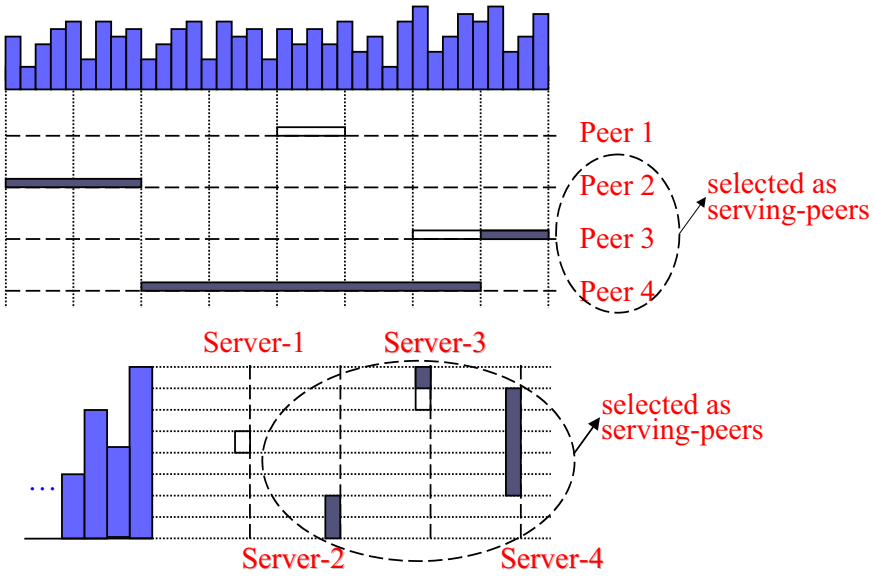

Fig. 8. A simple example of the server selection problem is shown. The original content can be distributed by either horizontal-cut or vertical-cut partitioning.

Both end users and content providers want to be certain that the network will be available at all times despite congestion or failures that may occur on individual peers. The system must contain *fault tolerance* and *load balance* capabilities to guarantee reliable QoS delivery. To prevent "peer spoofing," the system must ensure that only *authorized peers* are able to issue control commands to their contents. Moreover, only *authorized content* is delivered and that it cannot be corrupted or substituted during delivery. Peers are allowed to use *digital-right management* systems to control their distribution and playback rights. It would be better to work with existing browsers and players on the desktop without re-encoding.

3 Proposed Algorithm

3.1 Content Partitioning and Placement

Without loss of generality, we assume that there are two serving-peers said local peer and remote peer. We store a portion of video in local peer and the rest in remote peer. For potential users, the reliability of local peer is higher than that of remote peer with less error rate and high rate availability. If more data are cached at the local peer, one could allocate less bandwidth to transport video from remote peer. Therefore, not only playback jitter but also network bandwidth can be reduced. It increases the system scalability to support more users.

When a request of V is presented, video frames $f(.)$ would be sequentially delivered to the client. The time period starts from the transporting to the playing of the video is called the startup latency D (where the unit of time applied through this paper is called *frame-time* -- the time period between two contiguous frames). A simple example is given as shown in Fig. 9 where $V = \{ 3, 6, 2, 1, 6 \}$ and $D = 1$. Given the peak rate R, we denote the portion of video frame $f(i)$ cached in the local peer by $f(i)[c(i)]$. The total size of video cached is $C = \sum_{i=0}^{n-1} c(i)$. Without loss of generality, we assume that frame $f(i)$ is just displayed at time i. At any time i, the client consumes $f(i)$ for playback and receives (at most) R (bits) new data from the remote peer. As shown in Fig. 9, we have $R = 3$, $c(0) = c(2) = c(3) = c(4) = 0$ and $c(1) = 3$.

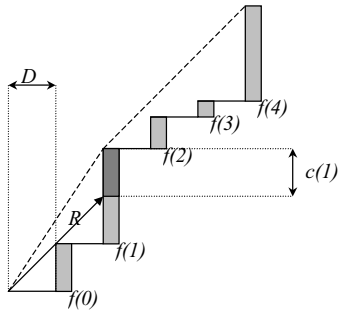

Fig. 9. The portion of the video frame $f(i)$ cached is denoted as $f(i)[c(i)]$ where $c(i)$ is the cached size.

Due to the compression technology applied, frames in an MPEG video will not be decoded unless their related I-frames are correctly received and decoded. The playback quality depends on the percentage of I-frame data received. In a video V, the total amount of I-frame data cached in the local peer is $C_I = \sum_{i=0}^{n-1}(u(i) \times c(i)) < C$. The function $u(i) = 1$ if $f(i)$ is an I-frame; otherwise, $u(i) = 0$. We can assume that the *loss rate* of the local peer $e_{p,c} = 0$ and the *loss rate* of the remote peer $e_{s,p} >> 0$. Therefore, we can roughly measure the playback quality by the ratio $(e_{s,p}*(|V|_I-C_I)+C_I):C$. $|V|_I$ is the total amount of I-frame data. Increasing the value of C_I/C (the percentage of I-frame data cached in the local peer) would decrease the probability of video decoding error and thus increase display quality.

In this paper, we propose an algorithm to resolve the proposed problem. Our algorithm starts from the traditional traffic smoothing algorithm. The intermediate result is a transmission schedule with a sequence of transmission segments. Assume that the k-th transmission segment is $S_k = (R_k, L_k)$ where R_k is the transmission rate and L_k is the time length. As the example shown in Fig. 9, we have two transmission segments $S_0 = (4.5, 2)$ and $S_1 = (3, 3)$. Given a bandwidth R, the size of video requested for caching in transmission segment $S_k = (R_k-R)*L_k$ if the given bandwidth $R < R_k$. Otherwise, $S_k = 0$. As the cache size requested in each transmission segment can be computed independently, the correction of the above equation can be proved easily. Notably, the curve shown in Fig. 9 is convex. However, the union of transmission segments may be concave while a small client buffer is introduced (see Fig. 10). It is not difficult to prove that the above equation is also correct for that case.

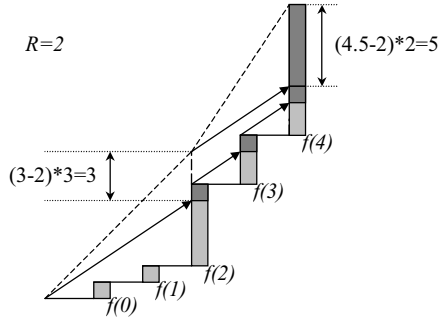

Fig. 10. A concave case of transmission segments to shown the cache size requested in the local peer.

Now, we can propose an algorithm to carefully utilize the available cache size in the local peer to reduce the network bandwidth required in the remote peer. Therefore, we can support as many users as possible. A detail description of the proposed algorithm is shown as follows.

Algorithm: MRP (Minimum-Rate Partitioning)

Use the traffic smoothing algorithm to decide the transmission segments $S_k = (R_k, L_k)$ for $k = 0$ to m. It takes O(n) in computation.

Sort these transmission segments by bandwidth requested R_k. The sequence of segments obtained $S'_k = (R'_k, L'_k)$ where $R'_k >= R'_{k+1}$.

Let $S'_{m+1} = (0, 0)$, $L = 0$, and $C' = C$

for $k = 0$ to m do {

L=L+L'$_k$

if $((R'_k - R'_{k+1})*L > C')$ then return$(R = R'_k - C'/L)$

$C' = C' - (R'_k - R'_{k+1})*L$

}

if ($C' > 0$) then return(R=0) /* download and play */
Given the network bandwidth, we can use the OC algorithm to compute the possible value of $c(i)$ for all i. The time complexity is $O(n)$.

Algorithm: OC (optimal caching)

1. $i = -D$; $b(i)=0$; /* startup latency D, buffer occupancy $b(i)$ */
2. repeat
3. i = i + 1; b(i) = **min**{ B, b(i-1) + R - f(i-1) };
4. **if** ($f(i) \leq b(i)$) **then** $c(i) = 0$; /* QoS constraints */
5. **else** { /* buffer is underflow */
6. c(i) = f(i) - b(i); b(i) = b(i) + c(i) = f(i);
7. Cache $f(i)$[c(i)] in the proxy;
8. }
9. **until** ($i > (n-1)$);

We can also apply the OSC algorithm to exchange the original cached data in the local peer $f(i)[x]$ (where x <= $c(i)$) with a previous I-frame. Therefore, the I-frame data cached in the local peer is maximized.

ALGORITHM: OSC (optimal selective caching)

OSC is constructed by rewriting step 3 and step 7 of OC:

Rewriting step 3 of OC algorithm as following:

3.A i = i + 1; b(i) = **min**{ B, b(i-1) + R - f(i-1) }; $\bigtriangledown$[i] = B-b(i);
3.B **if** ($\bigtriangledown$[i] <$\bigtriangledown$U) **then** $\bigtriangledown$U=$\bigtriangledown$[i];
3.C **if** ($f(i)$ is an I-frame) **then** {
3.D Δ(i) = f(i); U_i =$\bigtriangledown$[i]; $U_{i-(I\text{-}distance)}$ = $\bigtriangledown$U;
3.E Add the un-cached I-frame $f(i)$ to *I-list;*
3.F }

Rewriting step 7 of OC algorithm as following:

7.A **repeat** /* select the most I-frame */
7.B Get the next un-cached I-frame $f(w)$ from *I-list;*
7.C $\bigtriangledown U$=**min**{$\bigtriangledown U$, U_w}; x=**min**{$\Delta(w)$,$\bigtriangledown U$, $c(i)$};
7.D c(w) =c(w)+ x; Δ(w) = Δ(w) – x; c(i) = c(i) – x; $\bigtriangledown$U = $\bigtriangledown$U – x;
7.E Cache $f(w)$[x] in the proxy;
7.F **until** ($c(i)$=0 or $\bigtriangledown U$=0 or *I-list* = NULL);
7.G Cache $f(i)$[c(i)] in the proxy;
7.H **if** ($\Delta(w) > 0$) **then** {
7.I U_w =$\bigtriangledown$U;
7.J Add the un-cached I-frame $f(w)$ to *I-list;*
7.K }

3.2 Adaptive Smoothing with Rate-Availability

There are multi-servers in delivery. Without loss of generality, we can select any one of them to show our algorithm. In the selected connection, we define the transmission schedule $G(.)$ as a function that cumulates the amount of data received at the client. As the playback of video is assumed to be started at $t = 0$, d is the playback delay if the start time of the transmission schedule is $-d$. Assume that video data are transmitted by rate $r(t)$ between time t-1 and time t. The transmission schedule can be represented by a integration function of transmission rate $r(.)$ as follows.

$$G(t) = \sum_{i=-d}^{t} r(i)$$

The peak bandwidth of the network channel allocated for transmission is r = **max**{ $r(t) \mid \forall\ t$ }. According to the above formulation, $G(t)$ represents the amount of data sent by the server up to time t. Assume that there is no transmission error and the network delay is zero. $G(t)$ can also represent the amount of data received by the client up to time t. At a client, $G(t)$ and $F(t)$ represent the cumulated data received and consumed up to time t respectively. $b(t) = G(t) - F(t)$, called the buffer occupancy, would be the amount of transmitted data temporarily stored in the client buffer at time t. To avoid jitter in playback, a transmission schedule must be ahead of its playback schedule (such that $b(t) \geq |f_t|$ for any time t and the client buffer would not be underflow for playback). The minimal client buffer size required for supporting QoS delivery and playback is b = **max**{ $b(t) \mid \forall\ t$ }. Because $b(t) \geq |f_t|$

for any time t and $b \geq \mathbf{max}\{ |f_i| \}$, the required buffer size b is no smaller than the maximum frame size, and is not necessary to be larger than the size of video $|V|$.

Note that, given a limited buffer size, it results in loss of data if the transmission schedule sends too many data to the client buffer at the same time. Such an overflow condition should be avoided also in designing a *feasible* transmission schedule. In this paper, a transmission schedule is said to be feasible if it has no buffer overflow or underflow. Its upper bound $H(.)$ can be computed by $H(t) = \mathbf{min}\{ |V|, F(t\text{-}1) + b \}$. For any time t, the value of $G(t)$ must be not smaller than its playback schedule $F(t)$. Moreover, $G(t)$ must be not larger than its upper bound $H(t)$. Given the playback schedule $F(.)$ of a stored video V, different approaches were proposed to construct its feasible transmission schedule $G(.)$ in past years. In previous works, the performance of a transmission schedule is generally measured by its playback delay time, client buffer size, and bandwidth requirement. However, the bandwidth requirement they measured is the maximum network bandwidth required during transmission (called *peak rate*). A user request is admitted if its peak rate is smaller than the available bandwidth of the current network.

Notably, the network bandwidth is shared by different traffics. Therefore, the available rate of network is time-varying. It can be represented by a *rate availability function* (RAF) $z(.)$ where $z(t)$ is the available rate of network between time t-1 and time t. For supporting QoS delivery, the transmission schedule $G(.)$ must guarantee that its allocated rate $r(t)$ is not over the available rate $z(t)$ at any time t. RAF has been applied in admission control while a transmission schedule $G(.)$ with allocated rates $r(.)$ was given. Although the peak rate $r = \mathbf{max}\{ r(t) \mid \forall\ t \}$ may have been minimized, they did not consider available rates $z(t)$ for all time t in constructing transmission schedules. Therefore, the allocated rate $r(t)$ may be larger than its available rate $z(t)$ at time t. Previous methods those base on peak rate, instead of RAF, to construct transmission schedules would waste system resources in allocation. Therefore, the obtained $G(t)$ is not a feasible transmission schedule while RAF is considered. The system may require extending its delay time to provide $G(t)$ a guaranteed service.

In this paper, we consider RAF in constructing transmission schedule directly to for fully utilizing available resources to serve as many users as possible. Given a stored video $V = \{ f_0, f_1, ..., f_{n-1} \}$ and RAF of network $z(.)$, the amounts of minimal resource required for guaranteeing jitter-free playback can be decided by the following algorithm.

ALGORITHM: Lazy-RAF

// **INPUT**: the playback schedule $F(.)$ and RAF of network $z(.)$

// **OUTPUT**: the transmission schedule $L(.)$ with transmission rates $r(.)$,

// the minimal client buffer size b and the minimum playback delay d

$L(n\text{-}1) = |V| = F(n\text{-}1)$; $t = n\text{-}1$;

while ($L(t) > 0$) **do** {

$t = t - 1$; $L(t) = \mathbf{max}\{ F(t), L(t+1) - z(t+1) \}$;

}

Initialize $d = -t$, $b = 0$ and $r(i) = 0$ for all i;

for $t = -d+1$ **to** n-1 **do** {

$r(t) = L(t)\text{-}L(t\text{-}1)$; $b = \mathbf{max}\{ b, L(t)\text{-}F(t\text{-}1) \}$;

}

As the maximal available rate $z(t)$ is applied in each time t, the video data are transmitted and stored into the client buffer as late as possible. Therefore, the minimal buffer occupancy $L(t)$-$F(t$-1) can be determined at any time t under guaranteed QoS. It is not difficult to prove that the minimal client buffer size b and the minimum playback delay d can be achieved. Besides, given any transmission schedule $P(.)$ with RAF $z(.)$, we have $L(t) \leq P(t)$ for any time t. $L(.)$ is called the minimal $z(.)$-bounded transmission schedule.

Lemma-1: $L(.)$ is the minimal $z(.)$-bounded transmission schedule. It has the minimal buffer size and initial delay for all $z(.)$-bounded transmission schedules.

Proof:

(1) Suppose the contrary and let $P(.)$ be a $z(.)$-bounded transmission schedule, for which, there exists a time index x such that $L(x) > P(x)$. Let y be the smallest time index that satisfies $x < y$ and $L(y) = F(y) = L(x) + \sum\{ z(i)$; for $i = x+1$ to y $\}$. (The value y is existed. At least, we have the initial value $L(n\text{-}1) = F(n\text{-}1)$.) Follow the procedure steps of algorithm, $L(y) = F(y)$ implies $L(y+1) - z(y+1) \leq F(y)$. As $P(.)$ is $z(.)$-bounded, the relation $P(y) \leq P(x) + \sum\{ z(i)$; for $i = x+1$ to y $\}$ is true. We have $P(x) + \sum\{ z(i)$; for $i = x+1$ to y $\} < L(x) + \sum\{ z(i)$; for $i = x+1$ to y $\}$. That implies $P(y) < F(y)$. The underflow condition of the client buffer is occurred and $P(.)$ is not a feasible transmission schedule. It is a contradiction and $L(.)$ is the minimal $z(.)$-bounded transmission schedule.

(2) Since $L(.)$ is the minimal $z(.)$-bounded transmission schedule, it sends the minimal amount of data to the client buffer for guaranteeing jitter-free playback. At any time t, we have $L(t) \leq Q(t)$ where $Q(.)$ is any other $z(.)$-bounded transmission schedule. As buffer occupancies have the relation $L(t)\text{-}F(t\text{-}1) \leq Q(t)\text{-}F(t\text{-}1)$. It implies that $L(.)$ has the minimal buffer size (the required buffer size $\mathbf{max}\{ L(t)\text{-}F(t\text{-}1) \mid \forall\, t \} \leq \mathbf{max}\{ Q(t)\text{-}F(t\text{-}1) \mid \forall\, t \}$).

(3) At time 0, we have $L(0) \leq Q(0)$. Given the available rate $z(0)$, we have $L(-1) = \mathbf{max}\{ 0, L(0) - z(0) \} \leq \mathbf{max}\{ 0, Q(0) - z(0) \} \leq Q(-1)$. Repeat the above step, $0 < L(-d+1) \leq Q(-d+1)$ and $0 = L(-d) \leq Q(-d)$. As any other $z(.)$-bounded transmission schedule $Q(.)$ has $0 < Q(-d+1)$ and $0 \leq Q(-d)$, the transmission schedule $L(.)$ has the minimal initial delay for all $z(.)$-bounded transmission schedules.

The lemma is proved. **Q.E.D.**

In Lazy-RAF, the minimum requirements in buffer size and playback delay are decided for the given video $V = \{ f_0, f_1, ..., f_{n-1} \}$ and RAF $z(.)$. While a user request is presented, we can compare the available buffer size B (the available playback delay D) and the minimum buffer size b (the minimum playback delay d) to make the admission. If the client buffer size $B \geq b$ and the playback delay $D \geq d$ are given, we can construct a simple transmission schedule by the following algorithm.

ALGORITHM: Traffic-smoothing with RAF [stored video]
// $F(.)$ is the cumulative playback function.
// $H(t) = F(t) + b$ where b is the available buffer.
// $L_s(.)$ in $L(.)$ is the minimal RAF-bounded transmission schedule started from s.
// $A_s(.)$ is the maximal RAF-bounded transmission schedule started from s.
// G(.) is the cumulative transmission function.
Initial the start point $(s, G(s)) = (-d, 0)$.
Initialize the test end-point at time $t = -d$ and $t_A = t_L = -d+1$.
Initialize the peak bandwidth $r\_max = r_s = 0$.
$ss = s$; // the start point of the current window
$A_s(s) = G(s)$;
Repeat {
    $t = t + 1$;
    $R_L(t) = (L_s(t) - G(s)) / (t - s)$; // the lowest test rate at the time t.
    $A_s(t) = \min\{ H(t)^-, A_s(t-1) + z(t) \}$;
    $R_A(t) = (A_s(t) - G(s)) / (t - s)$; // the highest test rate at the time t.

if ($R_A(t_A) < R_L(t)$) { // => The segment is up-bounded by $A_s(t_A)$
  $r_s = R_A(t_A)$; $G(t_A) = A_s(t_A)$;
  output the transmission segment: < $G(s)$, $G(t_A)$, r_s >; // $r[s{:}t_A] = r_s$
  $s = t = t_A$; $t_A = t_L = s + 1$; r_max = max{ r_max, r_s };
  $A_s(t) = G(t)$; // -- this line can be deleted
} **else if** ($R_L(t_L) > R_A(t)$) { // => The segment is low-bounded by $L_s(t_L)$
  $r_s = R_L(t_L)$; $G(t_L) = L_s(t_L)$;
  output segment: < $G(s)$, $G(t_L)$, r_s >; // $r[s{:}t_L] = r_s$
  $s = t = t_L$; $t_A = t_L = s + 1$; r_max = max{ r_max, r_s };
  $A_s(t) = G(t)$;
} **else** { // -- can try the next frame directly.
  if ($R_A(t_A) >= R_A(t)$) { $t_A = t$; }
  if ($R_L(t_L) <= R_L(t)$) { $t_L = t$; }
}
} until (t is n-1); // the last frame
r_s = max{ min{ r_max, $R_A(t_A)$ }, $R_L(t_L)$ } // keep min peak rate & max utilization
$e = s + \lceil (|V|-G(s)) / r_s \rceil$; $G(e) = |V|$;
output segment: < $G(s)$, $G(e)$, r_s > // ==> the end of this schedule

4 Experiments

In this section, we test the proposed algorithm by different video streams: "Star War" and "Jurassic Park". Experiment results are evaluated based on the following performance indices.

Bandwidth (the peak transmission required) = R

Bandwidth utilization = $(\sum_{i=0}^{n-1} f(i) - \sum_{i=0}^{n-1} c(i)) / (R * (n+D-1))$

Percentage of data cached in the local peer = $\sum_{i=0}^{n-1} c(i) \ / \ \sum_{i=0}^{n-1} f(i)$

Table 1 summaries the values of parameters applied in our experiments. Results show that our algorithm is effective in not only the network bandwidth required but also the utilization of network bandwidth.

Table. 1. Values of parameters used in our experiments.

Parameters	Values
Startup Latency	1 frame-time
Buffer Size (Client)	200KB

To reduce the cost and improve the system scalability, the network bandwidth allocated for serving each video request must be precisely controlled. Under the same startup latency and client buffer, a good caching algorithm should carefully utilize the available cache size to reduce as many network bandwidth required as possible. In Fig. 11, the network bandwidth required for different percentages of data cached in the local peer is presented. Experiments show that the network bandwidth required is increasing when the percentage of data cached in the local peer is decreasing.

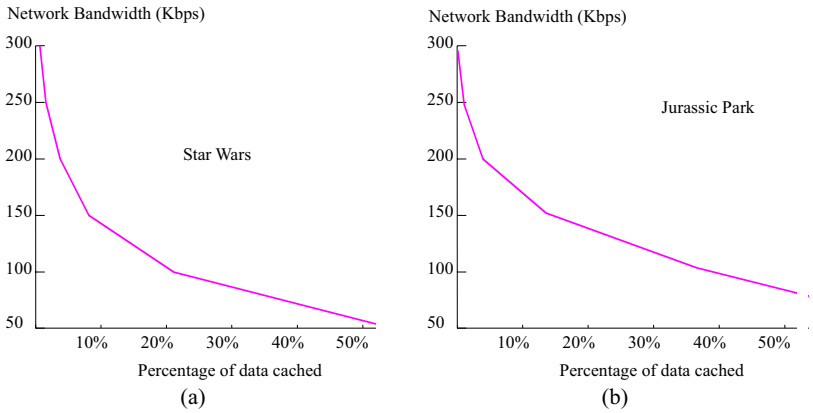

Fig. 10. The network bandwidth required for different percentages of data cached. (a) Star Wars. (b) Jurassic Park.

As the available network bandwidth is limited, we must utilize it sufficiently and avoid wasting it at any time. In a distributed multimedia system, high bandwidth utilization implies that lots of video requests can be served at the same time. In Fig. 11, we show the utilization of network bandwidth achieved. As our proposed algorithms utilize the network bandwidth allocated by an aggressive pre-fetching algorithm, the bandwidth utilization obtained is efficient. Given the same cache size in the local peer, the percentage of I-frames cached in the local peer can be applied to measure the smoothness of video playback.

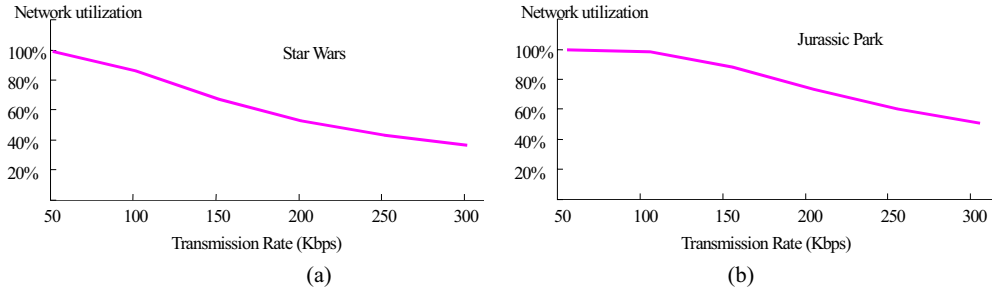

Fig. 11. Utilization of allocated bandwidth. (a)Star War. (b) Jurassic Park.

5 Conclusion

In this paper, we investigate some key technology issues of content delivery of streaming multimedia, which has both the biggest challenge and opportunity. Different from small-sized hypertext and image, the growing-popular multimedia content has huge size. It is impractical to be cached entirely and requires to be partitioned into small portions for delivery from multi-servers. As a multimedia content is VBR (variable-bit-rate) and requires guaranteed QoS (quality-of-service), each serving-peer needs a good delivery schedule for real-time streaming. We focus only on the multi-servers caching/streaming problem of multimedia delivery. Without addressing content indexing and routing, the proposed method can be applied for different content networks.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web", Scientific American, Issue: 0501, 2001.
[2] Squid Web Proxy Cache Project, http://www.squid-cache.org/.
[3] Napster Homepage, http://www.napster.com/.
[4] "Napster Messages," http://opennap.sourceforge.net/napster.txt.
[5] Sun Jxta Project, http://www.jxta.org/.

[6] http://www.microsoft.com/net/HailStorm.asp. Microsoft .NET HailStorm Project
[7] Korpela, E., Werthimer, D., Anderson, D., Cobb, J., and Lebofsky, M., "SETI@home - Massively Distributed Computing for SETI," Computing in Science and Engineering, 3(1):78-83, January/February 2001.
[8] Clarke, I., Sandberg, O., Wiley, B., and Hong, T. W., "Freenet: A Distributed Anonymous Information Storage and Retrieval System," ICSI Workshop on Design Issues in Anonymity and Unobservability, International Computer Science Institute, Berkeley, CA, 2000.
[9] http://freenetproject.org/index.php?page=protocol. "Freenet Protocol 1.0 Specification"
[10] Gnutella Homepage, http://gnutella.wego.com/.
[11] http://dss.clip2.com/GnutellaProtocol04.pdf, Clip2, "The Gnutella Protocol Specification v0.4."
[12] http://www.darkridge.com/~jpr5/. Ritter, J., "Why Gnutella Can't Scale. No, Really"
[13] Sripanidkulchai, K., "The Popularity of Gnutella Queries and Its Implications on Scalability," http://www.cs.cmu.edu/~kunwadee/research/p2p/paper.html.
[14] Jovanovic, M., "Scalability Issues in Large Peer-to-Peer Networks - A Case Study of Gnutella," University of Cininnati Technical Report 2001. http://www.ececs.uc.edu/~mjovanov/Research/paper.html.
[15] Adar, E., and Huberman, B. A., "Free Riding on Gnutella," Technical report, Xerox PARC, August 10, 2000.
[16] Akamai Homepage, http://www.akamai.com/.
[17] Ratnasamy, S., Paul Francis, Mark Handley, Richard Karp, and Scott Shenker, "A Scalable Content-Addressable Network," ACM SIGCOMM'01, August 27-31, 2001.
[18] Stoica, I., Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," ACM SIGCOMM'01, August 27-31, 2001.
[19] Andersen, D., Hari Balakrishnan, Frans Kaashoek, and Robert Morris, "Resilient Overlay Networks," 18th ACM Symposium on Operating Systems Principles, 2001.
[20] Kubiatowicz, John, et al., "OceanStore: An Architecture for Global-Scale Persistent Storage," ASPLOS 2000, November 2000.
[21] Zhao, Ben Y., John D. Kubiatowicz, and Anthony D. Joseph, "Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing," U.C. Berkeley Technical Report UCB/CSD-01-1141, April, 2001.
[22] Rowstron, A., and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," Middleware, 2001.
[23] Druschel, P., and A. Rowstron, "PAST: A large-scale, persistent peer-to-peer storage utility", HotOS VIII, Schoss Elmau, Germany, May 2001.
[24] Rowstron, A., and P. Druschel, "Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility", 18th ACM Symposium on Operating Systems Principles, 2001.
[25] Ray-I Chang, Meng-Chang Chen, Ming-Tat Ko and Jan-Ming Ho, "VBR Traffic Shaping for Streaming of Multimedia Transmission," Multimedia Networking: Technology, Management and Applications, Mahbubur Rahman Syed (Eds.), IGP, 2002.
[26] Ray-I Chang, Meng-Chang Chen, Ming-Tat Ko and Jan-Ming Ho, "Online Traffic Smoothing for Delivery of VBR Media Streams," Circuits, Systems, Signal Processing, Vol. 20, No. 1, 2001.
[27] Ray-I Chang, Meng-Chang Chen, Ming-Tat Ko and Jan-Ming Ho, "Bandwidth-Buffer Tradeoff for Delivery of Pre-Recorded Videos over Multiple Smoothing-Servers," Journal of Applied Systems Studies, Vol. 2, No. 3, 2001.
[28] Subhabrata Sen, Jennifer L. Rexford, Jayanta K. Dey, James F. Kurose, and Donald F. Towseley," Online Smoothing of Variable-Bit-Rate Streaming Video," IEEE Transaction on Multimedia, March 2000.
[29] Zhi-Li Zhang, Yuewei Wang, David H. C. Du, and Dongli Su, "Video Staging: A Proxy-Server-Based Approach to End-to-End Video Delivery over Wide-Area-Networks", IEEE/ACM Transaction on Multimedia, August 2000.
[30] Subhabrata Sen, Jennifer Rexford, and Don Towsley, "Proxy Prefix Caching for Multimedia Streams, " in proc. of IEEE INFOCOM 1999.
[31] Po-Chin Hu, Zhi-Li Zhang, and Mostafa Kaveh, "A Framework for Proxy-Based Receiver Adaption for Layered Video Transmission in Multicast Networks", in proc. of International Conference on Image Processing 1999 (ICIP'99).
[32] Jozsef Vass, Shelley Zhuang, Jia Yao, and Xinhua Zhuang, "Mobile Video Communication in Wireless Environments", in proc. of IEEE 3rd Multimedia Signal Processing 1999.

[33] J. D. Salehi, Z. L. Zhang, J. F. Kurose, and D. Towsley, "Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing," ACM SIGMETRICS, 1996.
[34] Ray-I Chang, Meng-Chang Chen, Ming-Tat Ko and Jan-Ming Ho, "Schedulable Region for VBR Media Transmission with Optimal Resource Allocation and Utilization," Information Sciences, Vo. 141, Issue 1-2, pp. 61-79, 2002.
[35] http://www.marconi.com
[36] Ray-I Chang, Meng-Chang Chen, Ming-Tat Ko and Jan-Ming Ho, "Bandwidth-Buffer Tradeoff for Delivery of Pre-Recorded Videos over Multiple Smoothing-Servers," Journal of Applied Systems Studies, Vol. 2, No. 3, 2001.
[37] Ray-I Chang, Meng-Chang Chen, Ming-Tat Ko and Jan-Ming Ho, "Online Traffic Smoothing for Delivery of VBR Media Streams," Circuits, Systems, Signal Processing, Vol. 20, No. 3, pp. 341-359, 2001.
[38] Ray-I Chang, Wei-Kuan Shih and Ruei-Chuan Chang, "Real-time disk scheduling for multimedia applications with a deadline-modification-scan scheme," Real-Time Systems, vol.19, no.2, 149-168, 2000.
[39] Ray-I Chang, Yu-Lin Chiu, "A Minimum-Rate Video-Staging Scheme for QoS-Guaranteed Communication," IEEE ISPACS, 2002.
[40] Shin-Hung Chang, Ray-I Chang, Jan-Ming Ho, Yen-Jen Oyang, "OC: An Optimal Cache Algorithm for Video Staging," IEEE Int. Conf. on Networks, pp. 711-722, 2002. (Best Paper)
[41] Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology. Andy Oram (Eds.): O'Reilly & Associates, Inc., 2001.
[42] Kung, H. T., and Wu, C. H. (2002). Content Networks: Taxonomy and New Approaches, To appear in The Internet as a Large-Scale Complex System, Kihong Park and Walter Willinger (Editors), published by Oxford University Press as part of Santa Fe Institute series, 2002
[43] Ray-I Chang, Yu-Lin Chiu, "A Minimum-Rate Video-Staging Scheme for QoS-Guaranteed Communication," *IEEE ISPACS*, 2002.
[44] Ray-I Chang, Yu-Lin Chiu, Jan-Ming Ho, "Resource Allocation for Stored Video Delivery with Rate Availability Function," *ICS*, 2002.