A Review On Content Delivery Network

Frederic Thouin and Mark J. Coates
Department of Electrical and Computer Engineering
McGill University
Email: {fthoui,coates}@ece.mcgill.ca

for AAPN & Nortel
Version 5.2 - Last modified on 2005-04-27

I. Introduction

With the advent of services and applications such as media streaming over the Internet, speed and reliability have become critical issues for users. Content delivery networks (CDNs) are (overlay) networks that are designed to distribute files to a set of clients. Through approaches such as replication of content at multiple servers (proxies) and redirection of user requests to closer servers, content delivery networks (CDNs) attempt to minimize latency at the user-end while reducing bandwidth consumption and load at the origin server. By sharing the load among various locations closer to the user-end, CDNs can deliver content to users in a timely manner. Content replication enhances robustness so that CDNs can maintain reliable service in case of failures.

As streaming media is already a multi-billion dollar industry and will probably be a dominant application in the future, the amount and nature of traffic it generates should be considered during the design of content-delivery networks. The delivery of streaming media causes new problems that did not apply to the distribution of HTTP objects: streaming objects are much larger than web objects and hence create much more traffic [1]. Furthermore, it is no longer possible to assume infinite storage size at the replica locations which makes calculations less trivial [2]. If implemented over existing network architectures, CDNs are a form of overlay network; their design consists of resource allocation decisions and the development of a combination of intelligent routing, performance monitoring and careful caching strategies. It is questionable whether even the best such design is scalable to scenarios where millions of geographically-distributed users are requesting on-demand video. When it is possible to augment or replace the existing underlying network, new challenges present themselves; ideally, the CDN traffic should be explicitly considered during topology design.

We plan to address the two challenges of (i) designing a CDN that operates over an existing network with known architectural properties, and (ii) performing the dual design of a metro-area agile all-photonic network (AAPN) [3] and CDN. As a first step, it is valuable to review the solutions proposed in the literature regarding the delivery of multimedia objects. This technical report serves that purpose and is organized as follows. Section II presents the parameters, constraints and various algorithms to solve the replica location and content allocation problem. Section III surveys various traffic models and file popularity distributions. Section IV covers the techniques used for request routing and content delivery. Finally, section V discusses the implementation of content delivery in an AAPN.

II. Topology Design

The purpose of a CDN is to transmit to users the content they requested in the most efficient manner, that is, meeting the quality of service (QoS) requirements at the lowest cost possible. To improve the QoS, CDNs use proxy servers between clients and the origin server that are called replicas or surrogates. The clients' requests are then re-routed to these closer servers. Placing copies of objects closer to the user minimizes the delay at the user-end while reducing the bandwidth requirements at the origin server. Since cost is also considered, placing replicas very close to the clients, in order to achieve very small delay, is not a viable solution because of the storage costs it incurs. On the other hand, placing the replicas too close to the origin requires far too much bandwidth to handle all the traffic. The *replica placement problem* consists of determining the location of objects such that the performance is maximized given an infrastructure or that the infrastructure cost is minimized for a given user perceived quality (e.g. delay, video quality, etc.). To solve the replica placement problem, it is crucial to first determine a good cost function which is minimized whilst respecting appropriate constraints. This cost function can be quite complex and the optimal solution (lowest cost) is often impossible to find within a reasonable time frame. For this reason, a wide variety of heuristics (algorithms) exists to approximate the problem and to find a near-optimal solution.

A. Problem Definition

CDNs are usually modeled as read-only (or read-mostly) workloads using classic network problems like the k-median problem or the facility location problem [4]. In the k-median problem, the objective is to select k locations for replicas among m potential sites for a fixed k. The choice for this value of k is not obvious; if the value is too small, clients are forced to take a longer route (long response time and high load on the network) while if the value is too large, the hit ratio becomes smaller and hence cost of delivery is shared by fewer requests. Also, a high number of replicas results in a considerable traffic load to distribute the objects to the replicas. Therefore, contrary to intuition, deploying as many replicas as possible is not always good. A solution, to avoid this tedious task of determining a value for k, is to find the subset of the m locations that minimizes the cost over all possible values of k, which is known as the facility location problem [2].

1) Parameters and Constraints: Important factors to consider when determining the cost function are the internodal distance between clients, replicas and origin servers. Many metrics are used to represent distance such as network latency, number of hops, or link cost (also called bandwidth cost). Another way to express distance is transmission cost; the cost to transmit a bit on a specific path [5], [6]. In [7], requests are served by the closest replica, so the authors use distance as a means to measure the users' perceived quality by summing the user-replica distance over all requests. In the streaming case, finding the multicast tree that minimizes the bandwidth cost is a trade-off between minimizing distance and maximizing the number of clients sharing a path segment (streaming and

multicast are discussed in section IV-B). Therefore, the authors of [8] argue that closest server and shortest path routing does not necessarily lead to lowest cost. Instead, to calculate the delivery cost, they use the total network bandwidth, which is expressed as the sum (possibly weighted) of the bandwidth required for each hop on the delivery path.

Another key parameter is the storage server cost [6], [7], [9], [10], or replication cost, of keeping a copy of an object at a given location. Moreover, authors in [6], [9] add a fixed start-up cost or a server installation cost in addition to storage cost. Also, in [7], the authors propose an algorithm where the location of the replica changes dynamically. So the start-up cost is expressed as the addition or removal of a replica site. However, in addition to storing the object, the server must also be able to serve all the incoming requests for this specific file. So, the server cost includes the cost of the required bandwidth, which is proportional to the popularity of the file it stores. The popularity of an object is the number of times it is requested (number of read accesses) in a given time interval (file popularity is discussed in section III-A). A server that hosts very large files (high storage cost) which are not popular (like archives) has low bandwidth requirements.

In multimedia applications, because of the size of the objects, it is not always possible to have complete replicas of the origin server, due to unacceptably large storage costs. Therefore, a selection of the objects is stored at proxy servers; the choice is based on popularity and hit ratio. The hit ratio represents the probability that a user's request is served on a given path [2]. The decision of whether to place a file at a replica is based on its size and its popularity: is the object popular enough (able to maintain a given hit ratio) to deserve the storage space it requires? As the popularity of an object can change through time, it might be necessary to replace objects or update them. In HTTP applications, objects are small and the transmission cost from the origin server to the replica is negligible. However, video objects are much larger and the distribution of a document is only compensated by a finite number of requests from the client. Therefore, the number of updates or replacements required can be another factor considered.

On top of these parameters, constraints can be included in optimization of the cost function. Depending on the given infrastructure, the storage capacity of the servers can be upper-bounded [6], [11]. While not enforcing a fixed restriction on the total capacity, the authors of [9] require it to be at least equal to the total demand. Others put a constraint on the minimum availability of any object in the system. In [6], [9], all requests must be handled and all objects must be available. The authors of [9] also add constraints on the load capacity of the server (number of requests it can cope with) and a quality of service (QoS) threshold (maximum delay) for each request.

2) Cost functions: The cost functions based on these parameters and constraints can be divided into categories according to whether they consider a single or multiple objects and whether they take storage into account [4]. In a single object cost function, only the aggregate user demand is considered; the specific objects requested are not important. In the case of streaming media applications, a common choice for the delivery cost model is one that considers the bandwidth required by the servers and network as the only factor [8], [12]. An alternative choice is a cost function based simply on distance and hit ratio [2] (Table I). In [7], the storage, or hosting cost, is part of the so-called maintenance cost, which also includes the cost of updating the copies at the different locations (Table I).

A more complicated case is one where there are many different objects in the system, each with different popularity (user demand). A proposed solution to this case is to minimize the transmission cost (similar to what is done in the examples above with delivery cost) by finding the position for each object that results in the largest savings in transmission cost [5] (Table I). However, it is often impossible to minimize the cost while maximizing the performance because these are two conflicting objectives. The authors of [11] map the quality of the service into the cost domain by determining the amount the customers are willing to pay for maximal performance. Finally, in the case where both multiple objects and storage are considered [6], [9], [10], the cost function can be the sum of the start-up cost, storage cost and transmission cost, as shown in Table I.

B. Replica Placement Heuristics

The cost functions presented in section II-A.2 are often complex and obtaining the optimal solution is impractical. The heuristics (algorithms with no guarantee of finding a solution) presented in this section offer near-optimal performance. In simpler scenarios, it is sometimes possible to calculate the optimal solution and use it as a reference to evaluate the performance of heuristics.

A popular heuristic, considered by many authors [2], [9], [11], [13], is greedy selection. It first chooses the replica that minimizes the total cost and then selects a second replica among the remaining sites such that the total is minimized when combined with the first choice. Replica sites are added either until a predetermined number of sites is reached (k-median problem) or when adding more replicas increases the total cost (facility location problem) [2]. Genetic algorithms [6] are another approach to select the sites based on a cost function.

Although these methods are known to perform very closely to the optimal solution (within a factor of 1.1-1.5), they require knowledge about the client locations in the network and internodal distances [14]. Among the alternatives to greedy algorithms are hot-spot [2] and max fan-out [2], [14]. In the hot-spot algorithm, the traffic generated near each site is used as the metric for selection and is expressed as the total number of requests from clients within a given range. At each step, the algorithm selects the hottest (maximum number of requests) site available. This is different from the greedy scheme as the latest choice does not depend on the combined cost with previous selections. The max fan-out algorithm behaves similarly with the difference that the metric used is the number of input/output terminals at each site. In both cases, sites are added until a local minimum is reached. As routers with high fan-out are usually busy, the solution is to build a cluster of

TABLE I
CATEGORIES OF COST FUNCTIONS

	Storage	Example	Other References
Single Object	No	$\sum_{\forall j} d_j \cdot hitratio \cdot c_{ij} + d_j \cdot (1 - hitratio) \cdot (c_i + c_{ij})$ [2]	[8], [12]
	Yes	$[A(x) + M(x)] \cdot \tau(x,d) + \sum_{j=1}^{\lvert V_R \rvert} \sum_{k=1}^{C} (d_j^{k+} C^+ + d_j^{k-} C^-)$ [7]	
Multiple Object	No	$\sum_{\forall i} saving(m_i) = \sum_{\forall i} C_i(0) - C_i(m_i u / b_i)$ [5]	[11]
	Yes	$\sum_{i=1}^{n} C_F \cdot y(i) + \sum_{i=1}^{n} \sum_{k=1}^{K} C_s \cdot [\sum_{j=1}^{n} x_k(i,j)/M]^+ + \sum_{i=1}^{n} \sum_{k}^{K} \sum_{\substack{j=1 \\ j \neq i}}^{n} C_{ij} \cdot x_k(i,j)$ [6]	[9], [10]

replicas as close as possible to high fan-out routers [14]. By using the sum of distances between each client and its replica as performance metric, these strategies usually work very well (within 1.1-1.2 of greedy placement). However, performance decreases when the number of clients is small.

The system state might change through time and the quality of an originally near-optimal configuration can deteriorate substantially. In order to adapt to system variations, we can periodically execute any of the aforementioned static algorithms to reposition replicas such that cost is minimized. However, if the period between two executions is not chosen carefully, the replica placement determined by the last algorithm execution can become poorly matched to the current network state. The authors of [7] propose a dynamic algorithm that analyzes the current configuration and removes unnecessary replica(s) (if possible) if it can support an increase in user demand. If the current configuration cannot, the algorithm adds one or more replica(s) while taking the cost of these changes into account. When considering the average number of replicas, user-replica average distance and number of requests that cannot be served as performance measures, the heuristic performs within 2-4% of the optimal strategy as computed by solving the Markov decision model.

d_j	demand from client j
$hitratio$	hit ratio of replica i
c_{ij}	cost (distance) from client j to replica i
c_i	cost (distance) from the origin to replica i
$A(x)$	user-perceived quality in network configuration x
$M(x)$	maintenance cost per unit of time of a network configuration x
$\tau(x,d)$	dwell time of a network configuration x
v_i	prefix size
$C_i(v_i)$	transmission cost for video i if a prefix v_i is stored
u	smallest unit of cache allocation
m_i	size of video i
b_i	mean bandwidth of video i
$y(i)$	1 if a server is installed at location i
$x_k(i,j)$	transmission cost of program k from location i to j
C_F	installation cost of a server
C_S	storage cost
M	number of multiple accesses
C_{ij}	transmission cost per program from location i to j

C. Content Allocation

Deciding upon the location of the proxy servers is not the only task, because the determination of the content stored at each one of these locations is non-trivial. The choice of content has an impact on the total cost (amount of storage required) and on the user perceived quality. If the selection is poorly made, the hit ratio is low and users are forced to retrieve the data from the origin server.

One of the strategies is server replication, which consists of placing copies of the origin server at strategic places in the network. Server replication partitions the network resulting in lower bandwidth requirements at the expense of server cost. When using such a strategy, the placement of the servers that minimizes total cost is above the head end switches (which connect the users to the network); between 70% and 90% of the binary tree depth [10]. However, according to [15], CDN using edge delivery (files are transmitted to users via servers placed on the edge of the Internet) is not technically and economically scalable to delivering high-quality broadband video with adequate QoS. The authors propose to have so-called leaf servers in local-area networks (LANs), which support a relatively small number of clients, as second-tier surrogates. The motivation for this approach is that heavy traffic does not go beyond the last foot of the edge servers and LANs have abundant and stable bandwidth, are less dependent on a sophisticated direction system and have a higher degree of personalization.

As mentioned in section II-A.1, it is not always possible to have complete replicas of the origin server because the large size of multimedia objects leads to a high storage cost. An alternative is to store only specific objects from the origin at

the surrogate servers; upstream bandwidth is reduced at the cost of increasing the storage for caching the most popular programs. Using the cost model they developed, the authors of [10] found that overall minimum cost is achieved when 15% of the programs are cached at 80% of the tree depth. It is obvious that the popularity of an object changes through time and a hot (popular) file might become cold (the number of requests falls below a given threshold) at any instant. Knowing that 10% of objects cover 80% of requests for Web objects, to maintain request coverage stable for long periods, it is important to replace objects that become cold with hot objects, a procedure called incremental clustering in [13].

Program caching can also be performed at more than one level in the network hierarchy [10]. The idea is to use a main cache to reduce overall system cost and a secondary cache at a higher level for fine tuning the performance. When the main cache is close to the root, the cost of the system is mainly driven by the bandwidth component which makes the secondary cache almost useless. As the main cache is placed closer to the user, storage starts being the dominant factor and splitting the cache becomes advantageous. If the client request rate is high and/or proxy storage is limited, storing file prefixes rather than full files significantly reduces delivery cost [5]. It prevents clients from experiencing delays and jitter and reduces traffic on the origin-proxy path [16]. Still, the authors of [12] argue that storage at proxies is only effective if the origin is not multicast-enabled, the file request is low or the cost of a proxy is a small fraction of the origin server.

An efficient way to improve the performance is by sharing the content of the different surrogate servers by grouping them into clusters. Clustering avoids the duplication of content at servers that are close to each other. In a hierarchical content routing scheme [17], the request can be served by the local server (local hit), by another server in the same cluster (intra-cluster), by a server outside the cluster (inter-cluster), or by the original content server. Another approach is to cluster data using correlation distance (spatial, temporal, session clustering or popularity-based) [13].

Finally, it is worth mentioning that there are two different approaches to allocate content [13]. First, in the client-initiated approach, or pull-caching, the replica retrieves the copy of an object in the case of a cache miss. On the other hand, in a server-initiated approach, or push-caching, content is distributed to replicas before any requests for this data have been made. If we anticipate that a specific object will be very popular (e.g. blockbuster movie release), it can be advantageous to distribute the object prior to any requests in order to avoid cache misses and longer delays.

III. Object Statistics

When allocating content with a program caching scheme, only the most popular files are stored, with the aim of minimizing the storage and bandwidth needs. By using an appropriate popularity distribution, we can predict the hit ratio at a replica site given the set of files it is hosting. If we know the number of requests served by the replica and the origin, we can estimate the amount of bandwidth required on both links by modeling the requirements of a movie file.

A. File popularity

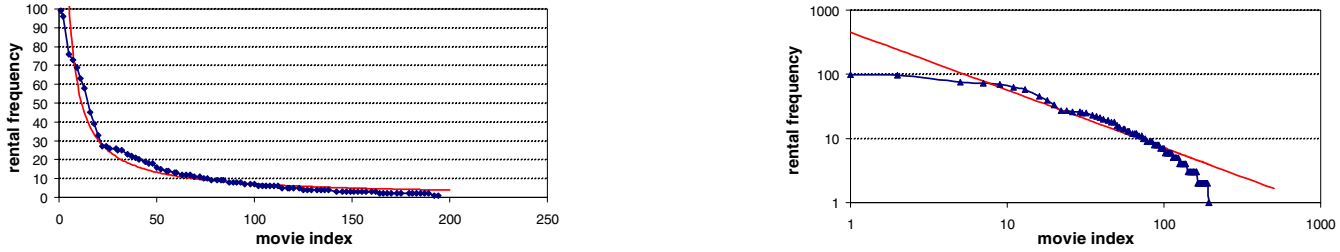
Previous studies in the distribution of multimedia files in CDNs have used Zipf's Law to characterize the popularity of the different files [5], [10], [12], [17]. In Zipf-like distributions [19], access frequency for file of rank i is equal to C/i^{α}, where C is a normalization constant and $\alpha > 0$ is the distribution parameter. Such distributions generate a linear curve in a log-log plot of access frequency versus rank. In the case of video-on-demand (VoD) applications, researchers also adopt the Zipf approach [20]–[22] to model popularity using data like rental statistics for a week from a video store [23]. On a linear scale, this data seems to fit a Zipf curve (Fig. 1(a)). However, looking at the log-log graph (Fig. 1(b)), we can see that the part of the curve for the most popular files is flattened and does not fit the Zipf linear curve.

The authors of [18] explain this behaviour by analyzing the characteristics of video object access. VoD system users rarely access the same file twice because the files are not modified (fetch-at-most-once). However, new files are often added to the system. In contrast, Web objects are accessed more than once because they are updated regularly (fetch-repeatedly). Since the popularity of a movie diminishes in time, when new titles are added to the system, they become the most popular titles. Hence, popularity distributions need to be adjusted over time. To model the flattened part of the curve, the authors of [24] used a mixture of two Zipf distributions, after noticing the the log-log graph is divided in two linear curves. Although the mixture model fits the data reasonably well, there is no explanation of why the mixture is a realistic model. The authors of [18] propose a model that is driven by Zipf's Law, but takes into account the "fetch-at-most-once" and "new arrivals" factors. When a client makes a second request, the previously fetched files are removed from the distribution and access probabilities are recalculated to have a total probability of 1. Also, when an object is added to the system, its popularity rank is determined from a Zipf distribution, the rank of existing files which are less popular is decreased and probabilities are recalculated to have a total probability of 1.

B. Traffic Models

Various models exist to determine the amount of bandwidth required by an object or by a specific type of traffic. In our case, we are interested in modeling traffic generated by high quality video for applications like VoD. Without using any compression schemes, it would be difficult to transmit DVD-like quality videos over the Internet because of their large bandwidth requirements. For that reason, methods like MPEG, which can achieve high compression ratio while maintaining good quality, are used and expected to generate a large part of the Internet traffic in the future.

MPEG videos are encoded using a variable bit rate (VBR) making traffic modeling a non-trivial task. The VBR is caused by the fact that compression is done by encoding each frame using one of three different schemes: intra(I), predicted(P) and bidirectional(B). I-frames are encoded with a low compression ratio, but are independent and act as reference point. P-frames provide a higher compression by using motion-compensated



(a) linear scale

(b) log-log scale

Fig. 1. (a) The popularity distribution from a 1992 video rental data set used to justify Zipf s law in many video-on-demand papers, along with a Zipf curve fit with $\alpha = 0.9$, and (b) the same data set and curve fit plotted on a log-log scale. Contrary to the assumption of many papers, video rental data does not appear to follow Zipf s law. [18]

prediction based on the previous I or P frame. Finally, B-frames achieve the highest level of compression by using both the previous and next frame in the sequence for its prediction. These different levels of compression produce frames with different sizes and hence a variable bit rate. MPEG movies use a group-of-picture (GOP) structure based on a (N,M) cyclic format; each sequence contains N frames (6, 8, 10, etc.) with the first one being an I-frame and every Mth one a P-frame [25]. A full-length movie is usually encoded with one GOP structure even though the MPEG standard allows the use of many different structures.

The variable bit rate (VBR) and high burstiness of these movies makes it difficult to predict the required resources. By reserving resources based on average rates, long delay can be experienced in case of bursts or when the source is transmitting at peak rates. On the other hand, prediction using the peak rates results in under-utilization of the network. This problem can be solved by using a stochastic process to model the dynamics of VBR video traffic. Models of this nature take advantage of the statistical properties of the source to achieve higher utilization of the bandwidth. However, the authors of [26] argue that using these models has several drawbacks, mainly arising from difficulty of implementation and complexity. Therefore, as an alternative, they suggest deterministic models, which provide an absolute upper bound (worst case) on the source's arrival traffic. Although some authors argue that worst-case approaches lead to low-utilization of the network, empirical evidence indicates that peak-rate allocation is not required for deterministic models [27]–[30]. In practice, these models are parameterized so that a bound can be established on the arrival rate from the source. As an example, the token-bucket uses a parameter pair (average rate, bucket depth) to establish this upper bound. As this solution is not suitable for VBR, the authors of [31] present an improved version of the leaky-bucket scheme by updating the parameter pair every GOP. Also, they take advantage of the fact that I-frames and P-frames can tolerate one extra frame delay compared to B-frames to reduce the bandwidth requirements [32]. Their simulations show better accuracy and higher utilization than previous leaky-bucket models or peak rate models.

The deterministic models presented above are called data-rate models (DRMs) because they only consider the rate at which data is arriving. While these models are good for predicting average packet-loss probability, they fail to identify such details as percentage of frames lost or incomplete [25]. Alternatively, there are frame-size models (FSMs) which generate the size of individual MPEG frames that can afterwards be used to deduce the data-rate. The authors of [25] show, through model simulation, that even a small loss rate can have a significant effect on a large number of I-frames. Because loss of an I-frame (or part of it) affects an entire GOP, the video quality decreases substantially. They propose two FSMs that generate frame sizes for full-length VBR videos preserving both GOP periodicity and size-based video-segment transitions, which previously proposed FSMs failed to do. Like other authors [33], [34], the authors of [25] have chosen to use a Markov renewal process to model the transitions between the video-segment of various sizes. The authors of [35] argue that it is important to consider the entire auto-correlation structure (many models deal with I, B and P frames sub-sequences separately).

Another consideration with video on demand (VoD) when predicting the required bandwidth is that several videos can be transmitted simultaneously on the same link. In that case, effective bandwidth per video (measure of the amount of bandwidth that a given source will use over a given time period) is in fact much lower because the average frame-size of a VBR video is usually different in different segments; this is known as multiplexing gain. The authors of [34] have developed a Markov-modulated gamma (MMG)-based model to predict the value of this multiplexing gain.

TABLE II
CATEGORIES OF TRAFFIC MODELS

DRMs (Deterministic)	: [26]–[32]
FSMs (Stochastic)	: [25], [33]–[35]

IV. CDN FUNCTIONS

There is more to a CDN than deciding upon the location of the replicas and the content stored at each of these sites. The CDN must distribute content to the replicas, route clients' requests to the appropriate site and must deliver the content from the replica (or origin server) to the client.

A. Request Routing

Request routing is a function performed by a CDN which consists of directing the client requests to the best surrogate server. The objective of a request routing algorithm is to exclude the surrogate servers that provide low performance while avoiding overloading the others.

For a replicated server system, one of the simplest approaches is Round-Robin (RR) [36]. This algorithm selects which surrogate serves a specific request in a cyclic mode without considering the state of the network. It means that a RR scheme can assign a surrogate that is overloaded or out of service to handle a specific request. On the other hand, there are many schemes which use various metrics to make a better decision than the RR algorithm. For example, the Response Time (RT) [37] algorithm selects the surrogate based on the response time the user previously experienced with a particular server. Although this scheme distributes requests among the different surrogates more efficiently than the RR scheme and provides users with low delay, it does not necessarily prevent overloading. On the other hand, the Load scheme [38] assigns a probability to each surrogate in inverse proportion to the client-replica path's current utilization. So, the Load algorithm prevents overloading by reducing the chance of a request being served by a busy server. Worst Surrogate Exclusion (WSE) [36] is an algorithm that takes full advantage of the CDN architecture by using latency, cluster request rate and link load and capacity. The algorithm is based on three concepts: the exclusion of surrogates with latency higher than the estimated average system response time, the equalization of the average response time and the prevention of overloading the surrogate servers. Based on simulation results, the authors show how WSE performs better than the other schemes which either consider only one metric (Load and RT) or do not consider the network at all (RR).

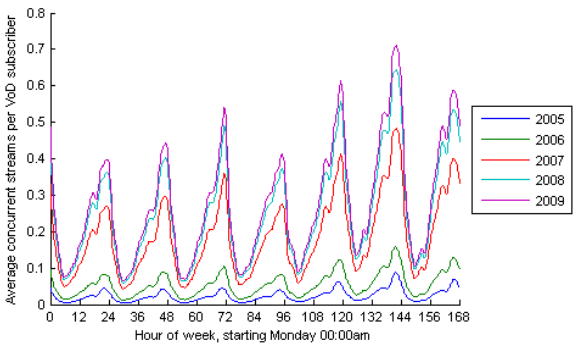
When program caching is preferred to server replication, the request routing algorithms are different than those just described. Because surrogate servers are hosting sets of different objects, requests cannot be simply routed according to some metric. A simple method is the query-based scheme [17], in which a proxy broadcasts a query to other nodes in its cluster if it does not have the requested content locally. If a node in its cluster responds positively, the request is routed to that server. The downside of this approach is that the queries and replies generate a significant amount of traffic. An alternative is a digest-based scheme [39] where each proxy maintains a list of the information stored on all others. Although there is no "query traffic", these lists need to be kept up-to-date date which, again, can produce significant traffic. One way to reduce this "update-traffic" is to centralize the list of files hosted by each proxy on a directory server [40]. Even if this approach helps to reduce undesired traffic, it has the disadvantage of having a single point of failure. The authors of [17] propose a solution called the semi-hashing based approach which has small routing overhead and high efficiency. Their scheme is a modified version of the hashing method [41], [42] which uses the content's URL, the address of the proxies and a hashing function to redirect the request to a designated proxy. Their enhancement consists of reserving a portion of storage at each proxy for local popular content. They show that even if the amount of storage dedicated is very small (smaller than 20%), there is a significant improvement in performance (higher hit-ratio). The only constraint is that cooperating proxies must be close to one another because requests are often redirected.

B. Concurrency and Content Delivery

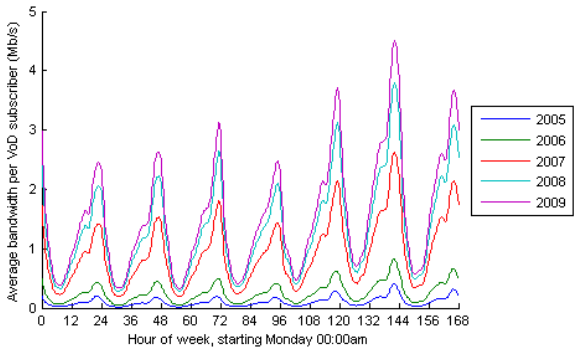
Content delivery is a function of CDN which consists of transmitting objects from the surrogate servers (or origin servers) to the clients. A popular technique to transmit large multimedia files over the Internet is called streaming. It allows clients to start displaying the data before the entire file has been transmitted which is useful if the user does not have fast access or the file to send is very large. In order to provide high quality streaming, VoD systems need to have an effective transmission scheme that reduces the substantial bandwidth requirements. Fig. 2 depicts projections of the usage (in terms of concurrent streams and bandwidth per subscriber) of VoD during the next five years. Although these expectations are not based on actual data, the presence of peak hours, during which bandwidth requirements are substantially greater than at other times, is highly likely. If the system is designed to support the peak rates, it will be under-utilized outside the high-usage periods [43]. On the other hand, if it is not, the customers will experience poor service during the busy hours. To eliminate the peaks and maintain a constant rate, one solution is to request content ahead of time. However, this procedure requires users to have a device that can store the content at home.

VoD is unicast in nature (there is a dedicated stream to each user), which imposes significant bandwidth pressure on the network. In order to handle this demand there are a variety of techniques for transmitting video data to clients that aim to reduce the bandwidth requirements. An example of such transmission schemes is batching, which collects requests that arrive within a given time interval and then multicasts the stream to the clients [44]. Multicasting is a one-to-many connection where multiple clients receive the same stream from a server by monitoring (listening) a specific multicast IP address. In patching (stream tapping) [45]–[47], the video is streamed to the first client who requests it, but future clients listen to the broadcast of the video while they receive the missing prefix (part of the movie that was streamed before they started listening) from a proxy server. In broadcast connections, as opposed to reactive connections, the client is passive and has no control on the stream (when it starts or stops).

The problem with these techniques is that they all require the path between the server and the client to be multicast-enabled (all the routers on the path must be able to interpret Class D IP addresses), but multicast capability is far from being fully deployed on the Internet [5]. One solution when the end-to-end network provides only unicast service is to use proxy-assisted transmission schemes (one-to-one connection between the server and the client). By using patching in the unicast context (which is possible because proxies can forward



(a) Concurrent streams per subscriber



(b) Bandwidth per subscriber

Fig. 2. VoD usage projections by time of day and day of week [43]

one copy of the data to multiple clients), the authors of [5] derived a transmission scheme that takes advantage of prefix caching at proxy servers. The proxy transmits the prefix to the clients (if present locally) and schedules the transmission of the suffix from the origin server. If a request arrives within a given interval after the transmission of the suffix starts, the proxy can schedule a patch from the origin for the missing part of the suffix.

Another alternative to IP multicast is called Application-layer Multicast [48]. In this method, end hosts need to maintain a data forwarding path for nearby hosts instead of using a multicast delivery tree. The authors of [49] propose a mechanism called Active Video Delivery (AVD) that takes advantage of application-layer multicast. Although AVD does not require all the routers on the transmission path to be multicast-enabled, it achieves the same efficiency as IP Multicast.

V. Directions for Agile All-Photonic Networks (AAPNs)

A. AAPN Architecture

An AAPN is a network in which the transmission and the switching through the core are done purely in the optical domain (all-photonic). When data enters the network, it is converted into an optical signal and it is converted back to an electronic signal when it reaches an exit-point. This means that the switching inside the network is performed using all-optical switches which have greater capacity than electronic switches and therefore remove a bottleneck of current high speed networks. Another advantage is that data format and bit rate are completely transparent to the switches.

An AAPN is built using an overlaid star topology which connects all the edge nodes together using central core nodes (Fig. 3). An edge node is the interface between the AAPN and the opto-electronic networks outside of the AAPN. These nodes can support a different number of wavelengths meaning that they do not all have necessarily the same traffic capacity. However, each node must be able to support a certain amount of traffic with every other edge node. All these edge nodes are connected to each other through more than one core nodes (for robustness). The core nodes are basically optical switches with an opto-electronic interface for control. The clients are connected to a single edge node (or second one for backup) directly or through a switch, which is the case in Fig. 3.

The switching can be performed in one of the three following ways. First, an entire wavelength can be dedicated to a connection between two edge nodes for persistent traffic. For connections with slowly-varying demand, there is a second modality based on time-slot reservation where each connection between two edge nodes is allocated a number of slots. Finally, to handle unanticipated bursts in traffic demand, some wavelengths are used for optical burst switching [3].

B. Motivation and problem statement

The distribution of objects from origin server to replicas is not negligible in the multimedia case due to the size of the files. Therefore, having most of the transmission path going through the AAPN is a clear advantage. Based on the AAPN topology and the CDN mechanisms, it seems that the replica servers should be collocated with the edge nodes of the AAPN while the replica-client path is outside of the AAPN. Even if replicas should handle most of the user requests, some will be served from other replicas or the origin server directly. Routing this part of the delivery traffic, along with the distribution traffic from origin servers to replicas, through the AAPN should result in significant ameliorations in performance and cost for the CDN.

There are two different cases to consider for the design of a network to deliver content through an AAPN. First, we can assume an existing AAPN where the edge and core nodes locations have already been decided. In that case, the CDN traffic is allocated a fraction of the overall AAPN traffic, thereby putting a constraint on the load from the origin to the replicas. Hence, going back to the problem definition in section II-A, the potential sites to host replicas are the locations of the AAPN edge nodes and the objective is to select how many and which ones should be used to host replica servers to achieve the lowest cost possible.

The other case is where the AAPN and CDN topology are jointly designed. As it is anticipated that the CDN will account for a substantial portion of the AAPN traffic, it could influence the location of the AAPN edge nodes. The origin servers will definitely generate a large amount of traffic for the distribution

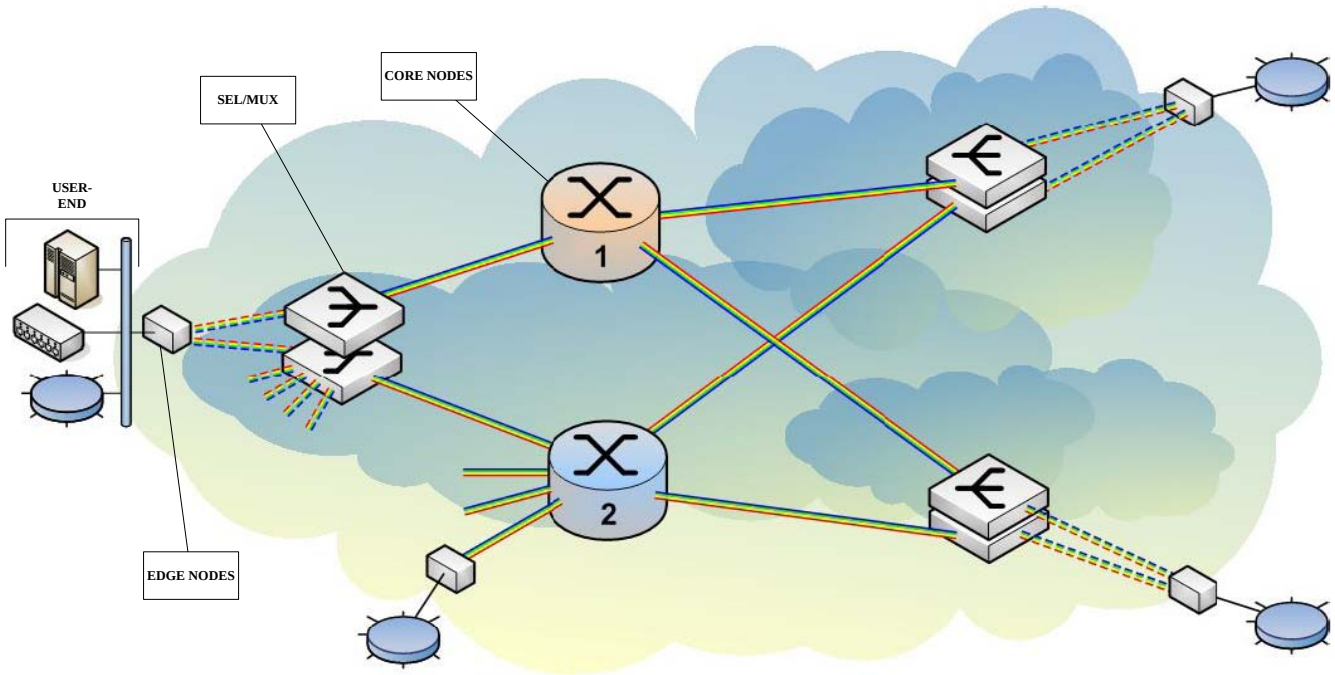


Fig. 3. The three-layer design includes edge nodes (switches that perform the O-E-O conversion), selector/multiplexor (Sel/Mux) devices, and all-photonic switches as the core nodes. The edge nodes are formed into sets and each set is connected to one or more Sel/Mux devices. Each Sel/Mux device is connected via DWDM equipment to one core node. [50]

of objects to replica servers or for the delivery to users. Thus, it makes sense to collocate AAPN edge nodes with origin servers. Also, the users for a VoD system are mainly located in residential areas, which is typically not the main source for other types of network traffic, so the presence of a CDN changes the traffic pattern in the network. We propose to adopt an iterative process for the joint design. First we decide upon the location of the AAPN edge nodes based on a prior model for the traffic pattern in the network. We then determine the placement of the replicas of the CDN for a specific demand. This placement changes the traffic pattern, so we repeat the AAPN topology design step (placement of edge nodes) for the new model of traffic demand. This process is repeated to adjust the locations according to the performance of the prior setup until a local minimum is reached.

C. Discussion

The aim of this section is to propose a direction for solving the location-allocation problem for a CDN in an AAPN, based on the various solutions reviewed in this paper.

1) Bandwidth and storage cost: The overall goal is to place replicas at the best locations possible in the network. We approach this task by constructing an appropriate cost function and identifying constraints on the minimization. We expect the cost function to have the following form:

$$\text{Cost} = a \cdot \text{storage cost} + b \cdot \text{bandwidth cost}$$

We expect the storage cost to include a start-up cost and a second term based on the amount of storage. Due to the substantial cost of software, we anticipate the start-up cost to be the dominant term in the equation. Depending on the design of the servers, the second term can have many forms. If we assume that the required bandwidth capacity is proportional to the amount of data stored then the second term can be expressed as a function of the bandwidth capacity of the server. Another possible design is one where the storage and capacity are independent. In that case, there is the possibility of adding a third term to the equation.

Another parameter to consider in the cost function is the bandwidth cost. This is expressed as the link cost in an AAPN for local distances ($< 80km$) in (1a) and for broader distances ($> 80km$) in (1b) [51].

$$COST_{direct} = 2 \cdot n \cdot C_{IF} + n \cdot C_f \cdot d_{AB} \tag{1a}$$

$$\begin{aligned} COST_{DWDM} = {} & 2 \cdot n \cdot (C_{IF} + C_{DWDM}) \\ & + (n/dw_{max}) \cdot \left(C_f + \frac{C_{LA}}{max_{amp}} \right. \\ & \left. + \frac{C_{D_{Reg}}}{max_{reg}} \right) \cdot d_{AB} \end{aligned} \tag{1b}$$

n	Number of interfaces from node switch where $n =$ ceil$(T_{AB}/10)$ if one fiber can support 10 Gb.
T_{AB}	Traffic between node A and node B. (Gb)
C_{IF}	Node switch interface cost. ($)
C_f	Cost of fiber. ($/km)
d_{AB}	Distance between node A and node B. (km)
C_{DWDM}	Cost of CWDM equipment ($)
dw_{max}	Max. number of fibers supported
C_{LA}	Cost of line amplifier. ($)
max_{amp}	Max. distance between two amplifiers. (km)

$C_{D_{Reg}}$ Cost of regenerator. ($)
max_{reg} Max. distance between two regenerators. (km)

Both (1a) and (1b) require the list of the internodal distances in the topology. Previous work in AAPN topological design [50] has been performed using known demand (population) and infrastructure (cables, buildings, etc.) and we will also assume that this information is available. This assumption eliminates the main drawback of greedy approaches [2], [9], [11], [13] to the placement problem. The hot-spot or max fan-out approaches can be misleading in the AAPN star topology because edge nodes are placed geographically to support the amount of traffic in that area.

2) Popularity distribution and traffic model: The traffic between each node, T_{AB}, is a function of what programs are hosted at each replicas, the popularity distribution used and the bandwidth required by each file.

As was discussed in section III-A, the popularity distribution for the files in systems like VoD is not well-modelled by a Zipf-like distribution. To model the "fetch-at-most-once" and new arrivals characteristics, the solution proposed by the authors of [18] is appropriate. The people using a VoD system are not the same in the afternoon and at night. One way to address this is to have different popularity ranking based on the time of the day, e.g., movies/programs for children are more popular in the afternoon. Furthermore, because of the popularity of blockbuster releases it can be worthwhile combining push-caching and pull-caching; latest releases (or their prefix) can be systematically distributed to replicas.

The bandwidth requirements of each object are determined using a traffic model. The FSMs, like the one presented in [25], are stochastic models which are very efficient for effectively balancing the tradeoff between loss and bandwidth consumption. We intend to investigate whether an FSM or a simpler deterministic model should be used. A model similar to the one proposed in [31] could be used to establish an upper bound on the requirements for each object.

3) Content delivery and request routing: The content delivery scheme depends on whether the network is multicast-capable, which has not been addressed in the AAPN architecture. A multicast-enabled path enables more efficient distribution (between the origin servers and the replicas) and delivery (between the replica (or origin) and the client). Also, note that the replica-client path will not be part of the AAPN and that the presence of multicast-enabled routers is not a design choice in that case. Still, there are techniques like AVD [49] which offer an alternative to IP multicast, outside of the AAPN.

Request routing and content sharing are other important things to consider during the design. Because clients are connected to a single edge node in the AAPN architecture, request routing between replicas and replica clustering are not possible. Even if content cannot be shared among the various sites, a replica site could host a cluster of replica servers that shares content and traffic. In the case of a miss at the replica, requests are usually served by the origin server, but to reduce the load on the origin these requests could be routed to another replica. To do so, lists must be maintained to know what each replica is currently hosting. This list can be stored at a directory server collocated with each core node, which eliminates the single point of failure. The need to collocate a server with a core node is solved by placing an edge node at the core node location.

REFERENCES

[1] L. Kontothanassis, R. Sitaraman, J. Wein, D. Hong, R. Kleinberg, B. Mancuso, D. Shaw, and D. Stodolsky, "A transport layer for live streaming in a content delivery network," in *Proc. of the IEEE*, vol. 92, Sept. 2004.

[2] M. Yang and Z. Fei, "A model for replica placement in content distribution networks for multimedia applications," in *Proc. of IEEE ICC*, Anchorage, AK, May 2003.

[3] G. Bochmann, M. Coates, T. Hall, L. Mason, R. Vickers, and O. Yang, "The agile all-photonic network: An architectural outline," in *Proc. Biennial Symp. on Comm.*, Kingston, Canada, May 2004.

[4] M. Karlsson, C. Karamanolis, and M. Mahalingam, "A unified framework for evaluating replica placement algorithms," Hewlett-Packard Laboratories," Technical report, 2002.

[5] B. Wang, S. Sen, M. Adler, and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," in *Proc. of IEEE Infocom*, New York, NY, June 2002.

[6] S.-J. Kim and M. Choi, "A genetic algorithm for server location and storage allocation in multimedia-on-demand network," in *Proc. of Symp. on Trends in Comm.*, Bratislava, Slovakia, Oct. 2003.

[7] N. Bartolini, F. Presti, and C. Petrioli, "Optimal dynamic replica placement in content delivery networks," in *Proc. of IEEE ICON*, Sydney, Australia, Sept. 2003.

[8] M. K. V. J. M. Almeida, D. L. Eager and S. Wright, "Minimizing delivery cost in scalable streaming content distribution systems," *IEEE Trans. on Multimedia*, vol. 6, pp. 356–365, April 2004.

[9] T. Nguyen, C. Chou, and P. Boustead, "Resource optimization for content distribution networks in shared infrastructure environment," in *Proc. of ATNAC*, Melbourne, Australia, Dec. 2003.

[10] F. Schaffa and J.-P. Nussbaumer, "On bandwidth and storage tradeoffs in multimedia distribution networks," in *Proc. of IEEE Infocom*, Apr. 1995.

[11] S. Buchholz and T. Buchholz, "Replica placement in adaptive content distribution networks," in *Proc. of the Symp. on Applied Computing*, Nicosia, Cyprus, Mar. 2004.

[12] J. Almeida, D. Eager, M. Ferris, and M. Vernon, "Provisioning content distribution networks for streaming media," in *Proc. of IEEE Infocom*, New York, NY, June 2002.

[13] Y. Chen, L. Qiu, W. Chen, L. Nguyen, and R. Katz, "Efficient and adaptive web replication using content clustering," *IEEE J-SAC*, vol. 21, Aug. 2003.

[14] P. Radoslavov, R. Govindan, and D. Estrin, "Topology-informed internet replica placement," in *Proc. of IEEE WCW*, Boston, MA, June 2001.

[15] J. Lu, "An architecture for delivering broadband video over the internet," in *Proc. of ITCC*, Las Vegas, NV, Apr. 2002.

[16] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *Proc. of IEEE Infocom*, New York, NY, Mar. 1999.

[17] N. Jian, D. Tsang, I. Yeung, and H. Xiaojun, "Hierarchical content routing in large-scale multimedia content delivery network," in *Proc. of IEEE ICC*, Anchorage, AK, May 2003.

[18] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, modeling, and analysis of a peer-to-peer file-sharing workload," in *Proc. of ACM SOSP*, Oct. 2003.

[19] G. K. Zipf, *Human Behavior and the Principal of Least-Effort*. Addison-Wesley, 1949.

[20] D. S. A. Dan and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in *Proc. of ACM Multimedia*, San Francisco, CA, Oct. 1994.

[21] K. A. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems," in *Proc. of ACM SIGCOMM*, Cannes, France, Sept. 1997.

[22] J. Segarra and V. Cholvi, "Distribution of video-on-demand in residential networks," in *Proc. of IDMS*, Lancaster, UK, Sept. 2001.

[23] (2000, March) Video store magazine. Published by Avanstar Comm. [Online]. Available: http://www.videostoremag.com

[24] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, "Analysis of educational media server workloads," in *Proc. of NOSSDAV*, Port Jefferson, NY, June 2001.

[25] U. Sarkar, S. Ramakrishnan, and D. Sarkar, "Modeling full-length video using markov-modulated gamma-based framework," *IEEE/ACM Trans. on Networking*, vol. 11, pp. 638–649, Aug. 2003.

[26] D. E. Wrege, E. W. Knightly, H. Zhang, and J. Liebeherr, "Deterministic delay bounds for VBR video in packet-switching networks: fundamental limits and practical trade-offs," *IEEE/ACM Trans. on Networking*, vol. 4, pp. 352–362, June 1996.

[27] R. Cruz, "A calculus for network delay, part I: Network elements in isolation," *IEEE Trans. on Information Theory*, vol. 37, pp. 114–131, January 1991.

[28] E. Knightly and H. Zhang, "Traffic characterization and switch utilization using deterministic bounding interval dependent traffic models," in *Proc. of IEEE Infocom*, Boston, MA, April 1995.

[29] J. Liebeherr, D. E. Wrege, and D. Ferrari, "Exact admission control for networks with a bounded delay service," *IEEE/ACM Trans. on Networking*, vol. 4, pp. 885–901, Dec. 1996.

[30] H. Zhang and D. Ferrari, "Improving utilization for deterministic service in multimedia communications," in *Proc. of ICMCS*, Boston, MA, May 1994.

[31] C. Lee, C. Lin, and P. Chang, "An improved traffic modeling scheme for MPEG video over content delivery networks," in *Proc. of IEEE ICCS*, Singapore, Nov. 2002.

[32] W. Tan and A. Zakhor, "Packet classification schemes for streaming MPEG video over delay and loss differentiated networks," in *Proc. of Int'l Packet Video Workshop*, May 2001.

[33] J.-A. Zhao, B. Li, and I. Ahmad, "Traffic modeling for layered video," in *Proc. of IEEE ICME*, Baltimore, MD, July 2003.

[34] W. Zhou, S. Ramakrishnan, D. Sarkar, and U. Sarkar, "Bandwidth estimation for multiplexed videos using MMG-based single video traffic model," in *Proc. of Globecom*, San Francisco, CA, Dec. 2003.

[35] Q. Zhang, C. Lin, H. Yin, and Q.-H. Dai, "An accurate scene-based traffic model for mpeg video stream," in *Proc. of IEEE ICECS*, Sharjah, United Arab Emirates, Dec. 2003.

[36] M. Masa and E. Parravicini, "Impact of request routing algorithms on the delivery performance of content delivery networks," in *Proc. of IPCCC*, Phoenix, AZ, Apr. 2003.

[37] R. L. Carter and M. E. Crovella, "Server selection using dynamic path characterization in wide-area networks," in *Proc. of IEEE Infocom*, Kobe, Japan, Apr. 1997.

[38] J. S. Chase, "Server switching: Yesterday and tomorrow," in *Proc. of IEEE WIAPP*, San Jose, CA, July 2001.

[39] A. Rousskov and D. Wessels, "Cache digests," *Computer Networks and ISDN Systems*, vol. 30, no. 22-23, pp. 2155–2168, 1998.

[40] S. Gadde, M. Rabinovich, and J. S. Chase, "Reduce, reuse, recycle: An approach to building large internet caches," in *Proc. of HotOS*, Cape Cod, MA, May 1997.

[41] V. Valloppillil and K. W. Ross, "Cache array routing protocol v1.0," Internet draft, Feb. 1998.

[42] D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina, K. Iwamoto, B. Kim, L. Matkins, and Y. Yerushalmi, "Web caching with consistent hashing," *Computer Networks*, vol. 31, no. 11-16, pp. 1203–1213, 1999.

[43] K. Couch. (2005, January) Raising the bar for triple play with VoD. [Online]. Available: http://www.convergedigest.com/blueprints/ttp03/2005nortel1.asp?ID=189&ctgy=Headend

[44] C. Aggarwal, J.Wolf, and P.Yu, "On optimal batching policies for video-on-demand storage servers," in *Proc. of ICMCS*, Hiroshima, Japan, June 1996.

[45] S. Carter and D. Long, "Improving video-on-demand server efficiency through stream tapping," in *Proc. of IEEE ICCCN*, Las Vegas, NV, Sept. 1997.

[46] K. A. Hua, Y. Cai, and S. Sheu, "Patching: a multicast technique for true video-on-demand services," in *Proc. of ACM Multimedia*, Bristol, England, Sept. 1998.

[47] L. Gao and D. F. Towsley, "Supplying instantaneous video-on-demand services using controlled multicast," in *Proc. of ICMCS*, Florence, Italy, June 1999.

[48] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *Proc. of ACM Sigmetrics*, Santa Clare, CA, June 2000.

[49] C. C. Hsu, T. Aoki, and H. Yasuda, "Distributing video content using a router-assisted multicast mechanism," in *Proc. of IEEE PACRIM*, Victoria, Canada, Aug. 2003.

[50] L. Mason, A. Vinokurov, N. Zhao, and D. Plant, "Topological design and dimensioning of agile all photonic networks," 2005, to appear in Computer Networks Journal, Special issue on Optical Networking.

[51] L. Mason and N. Zhao, "Cost model for different circuit designs," private communication, McGill University, Montreal, Quebec, Sept. 2004.