Improving the WWW: Caching or Multicast?[*]

Pablo Rodriguez Keith W. Ross Ernst W. Biersack
Institut EURECOM
2229, route des Crêtes, BP 193
06904, Sophia Antipolis Cedex, FRANCE
{rodrigue, ross, erbi}@eurecom.fr
Tel: (33) 04 93 00 2673
Fax: (33) 04 93 00 2627

November 4, 1998

Abstract

We consider two schemes for the distribution of Web documents. In the first scheme the sender repeatedly transmits the Web document into a multicast address, and receivers asynchronously join the corresponding multicast tree to receive a copy. In the second scheme, the document is distributed to the receivers through a hierarchy of Web caches. We develop analytical models for both schemes, and use the models to compare the two schemes in terms of latency and bandwidth usage. We find that except for documents that change very frequently, hierarchical caching gives lower latency and uses less bandwidth than multicast. For rapidly changing documents, multicast distribution reduces latency, saves network bandwidth, and reduces the load on the origin server. Furthermore, if a document is updated randomly rather than periodically, the relative performance of CMP improves. Therefore, the best overall performance is achieved when the Internet implements both solutions, hierarchical caching and multicast.

Keywords: WWW, Multicast, Caching, Push

1 Introduction

The response time to retrieve a Web document can be very frustrating, particularly when a document is retrieved over a transoceanic link. For example, from Europe it can take minutes to retrieve a small document from North America during local work hours. Even within North America latency can be unsatisfactory when trying to retrieve a document from a congested server during busy hours.

It is therefore of great interest to implement schemes in the WWW that reduce latency. One popular scheme is to use shared caches within a network (in addition to the local Web cache at the client [15] [28]). A shared Web cache can be placed at the institutional, regional, or even national level. To illustrate the idea, consider a large institution, such as a large university, confined to a contiguous geographical area. The clients of such an institution are typically interconnected with a low-cost high-speed LAN. Suppose the institution also attaches a shared Web cache to the LAN. In this common scenario, each request is first sent to the institution's cache; if the document is present in the cache, the document is delivered to the client over the high-speed LAN; otherwise, the cache retrieves the document from the origin server, places a copy in its storage, and forwards the document to the client. If the hit rate at the institutional cache is high, then the institution enjoys significant reductions in average latency and bandwidth usage.

[*]To appear in Computer Networks and ISDN Systems, 1998

A single-shared cache at any network level (institution network, regional network, or national network) may not be satisfactory for two reasons. First, it may not have enough storage capacity to be able to cache all the popular documents (i.e., the documents requested more than once in, say, a week). Second, it may not have enough processing power or enough access bandwidth to expediently handle the requests. To overcome these two problems, multiple caches can be installed within a network level; for example, a university could have tens of workstations, each with 10 Gbytes of storage, and each serving as a shared cache. These sibling caches can be made to cooperate with each other by using Internet Caching Protocol (ICP) [28] [27] or hash routing [20] [24].

A more serious problem with caching is that documents can become stale, i.e., the version of the document in the cache becomes out-of-date. As we shall briefly discuss in the body of this paper, HTTP/1.1 (and to a more limited extent HTTP/1.0) provide cache-control mechanisms which allow a cache to determine whether its copy of the document is up-to-date or stale. But if a document – such as a news report, a weather forecast, or a stock-market quote – is updated frequently, then caching of the document may be of little benefit, and may in fact actually increase latency.

An alternative way of dealing with popular and frequently-changing documents is to distribute them with **continuous multicast push (CMP)** [19] [1]. Here, a server housing a popular and frequently-changing document continuously multicasts the latest version of the document on a multicast address. Clients tune into the multicast group for the time required to reliably receive the document and then leave the multicast group. More specifically, (1)The server sends the popular document cyclicly into the multicast tree; (2) a client that desires a CMP document obtains the document's multicast address from the document's URL; (3) the client then joins the multicast group and stays in the group until it receives the entire document reliably. The Web server should only consider using CMP for highly-popular documents. (Our analysis shall show that the less frequently changing documents should be distributed with hierarchical caching.)

A nice feature of the CMP is that clients are assured of receiving an up-to-date version simply by joining the multicast tree of the document. An obvious disadvantage is that CMP requires that a reliable multicast infrastructure be present in the Internet. Such an infrastructure may not be widespread in the Internet for several years.

Yet another means to distribute Web documents is to use **hierarchical caching**. In this scheme, shared caches are present at the institutional, regional, and national levels. Each client points its browser to its institutional cache, each institutional cache points to its regional cache, and each regional cache points to its national cache. Thus, when a client desires a document, a series of requests is sent up the caching hierarchy until an up-to-date copy of the document is found. When the document is found, either at a cache or at the origin server, it travels down the hierarchy, leaving a copy at each of the intermediate caches. (Again, multiple caches can be placed within each ISP at any tier to improve performance and increase storage capacity.) It is interesting to note that *hierarchical caching mimics reliable multicast*. For example, to distribute a popular document from a Web server to M clients, it is not necessary for the server to send M copies of the document. Instead the server sends at most one copy to each of the national caches; each national cache sends at most one copy to each of the regional caches that point to the national cache, etc. (This sending is done asynchronously, driven by the client request pattern.) Thus, the caches act as application-layer multicast nodes, and the documents are sent from cache to cache by tunneling across routers.

Of course, caching requires cache servers to be purchased – perhaps a large number for each ISP in order to have sufficient storage, processing power, and access bandwidth. But institutional, regional and national ISPs are prepared to pay the cost, as they are currently aggressively deploying caches [2]. Hierarchical caching can be deployed much more frequently than reliable multicast since it operates at the application layer rather than at the network and transport layers.

In this paper we compare the distribution of *hot and changing documents* by CMP and by hierarchical caching. We develop analytical models for both CMP and for hierarchical caching. We suppose that N hot-changing documents have been identified. To simplify the analysis and to not obscure the key points, we assume that each of N documents is updated at the same rate, e.g., once every minute. For the CMP model, we distribute these N hot-changing documents with CMP; the less frequently-changing documents are distributed within an existing caching hierarchy. For the caching-hierarchy model, we assume all cacheable documents are distributed through the cache hierarchy, including the N hot-changing documents. After developing performance models for these two distribution schemes, we attempt to shed some insight on when it is preferable to CMP the N hot-changing documents or preferable to simply distribute all the documents through a caching hierarchy.

Our analytical models and computational work show that unless the document changes very frequently (on the order of a few minutes or less), then hierarchical caching gives lower latency. The superior performance of caching is primarily due to the fact that available transmission rate for caching is the lowest rate between cache server and receiver, whereas the available transmission rate for multicast is the lowest rate between the origin server and receiver. If the document does not change too frequently, there is a high probability that an up-to-date version is in a nearby cache, and can therefore be delivered at a relatively high rate. However, if the N documents change very rapidly, then an up-to-date document will rarely be available in a nearby cache, and CMP will perform better. Furthermore, if a document is updated randomly rather than periodically, the relative performance of CMP improves.

This paper is organized as follows. In Section 2 we describe CMP and hierarchical caching in greater detail. In Section 3 we describe our specific model for comparing CMP to hierarchical caching. In Sections 4 and 5 we provide latency analyses for CMP and hierarchical caching, respectively. In Section 6 we present a numerical comparison of the two schemes. In Section 7 we provide a bandwidth utilization analysis for the two schemes. In Section 8 we summarize our findings and briefly discuss how integrating caching with multicasting can improve performance.

2 Overview of Hierarchical Caching and CMP

2.1 Hierarchical Caching

Caching takes place at the application layer and allows for incremental deployment. Hierarchical caching is already a fact of life in much of the Internet [2]. Most ISPs and institutions connected to the Internet have been installing caches to reduce the bandwidth and decrease the latency to their clients [2] [6] [5] [18] [12]. However, caching does not come for free and there are still open issues relating to it: (1) Installing a cache requires additional resources including computers, disks, software, and system administrators. (2) Caches need to cooperate together to increase the hit rate [6] [22] [20]. (3) Caches need to maintain document consistency and provide the user with the most recent update.

In fact, the effort of installing a caching hierarchy resembles the effort that was required to put in place the first experimental multicast overlay network called **MBONE** [9] [13]. A cache hierarchy mimics a reliable multicast distribution scheme but with "application hop"-by-"application hop" congestion control and reliability. The Harvest cache [6] and its public domain derivative Squid [25] are currently the most popular caching hierarchy technologies on the Internet [2].

Hierarchical caching works as follows. At the bottom level of the hierarchy there are the client caches. When a request is not satisfied by the client cache, the request is redirected to the institutional cache. At the institutional level several caches may cooperate to increase the hit rate and distribute the load. If the document is not found at the institutional level the request is then forwarded to the regional cache which in turn forwards unsatisfied requests to the national cache. If the document is not found at any cache level, the national cache contacts directly the origin server. When the document is found, either at a cache or at the origin server, it travels down the hierarchy, leaving a copy at each of the intermediate caches. Placing a copy at the different caching levels does not add any store-and-forward delay because a cache starts forwarding the document to the lower cache level as soon as it starts receiving the document without waiting for its complete reception.

One of the main drawbacks of caching is that receivers may obtain stale documents. The current HTTP 1.0 protocol provides several mechanisms to keep cache consistency. Each document has a time-to-live (TTL) which is set by the server and indicates for how long the document will remain unchanged. If the server updates its content at fixed known intervals (i.e. periodically) the cache knows exactly when the document is fresh or stale without contacting the server. However, many times the origin server can not foresee when its documents are going to change and therefore it can not provide an accurate time-to-live. In this situation when the TTL expires the cache checks the document's consistency with the server. The cache can send an "if-modified-since" request to the origin server with a timestamp. Upon reception, the server checks whether the document has been modified since the timestamp. If so, the server returns the code "200" and the new copy; otherwise, the server returns the code “304”, which stands for document unmodified.

The difficulty with the TTL approach is that it is often hard to assign the appropriate time-to-live of a document. If the value is too small, the server is burdened with many “if-modified-since” messages, even if the document is not changed. If the value is too large, the probability that the user sees a stale copy increases. The *adaptive TTL* approach tries to handle this problem by allowing the cache manager to assign a time-to-live value based on the observed lifetimes of the document. If caches are asked to deliver the most up-to-date copy to the receivers, a pooling every time mechanism must be used. The cache sends an “if-modified-since” request every time that a request for a document hits the cache [14].

Some new protocol headers concerning caching have been introduced in version 1.1 of HTTP [10], which is currently being deployed. This new headers provide significant improvement over the mechanisms used in HTTP 1.0. The new headers allow the origin servers to specify the maximum time that a document may be kept at the caches. Additionally, clients can specify the degree of staleness acceptable for a certain document, which relaxes the inflexible reload option that always polls the origin server.

When a document is requested through a caching hierarchy some additional delays are introduced: (1) *Resolution delay*, which is the time to check if the document is present in an ISP (ICP queries [26], hashing function [20], routing [**?**]). In order to keep this delay low a caching hierarchy should not have more than three levels [6]; (2) *TCP delay*, which is due to the slow start phase of the different TCP connections between every cache level [17]. The slow start is more relevant when the completion time of the document is small. The effect of this delay is reduced with persistent TCP connections [10]; (3) *Server delay*, which is due to busy servers that need to deal with many requests for document updates from several national caches. (4) *Queuing delay*, which is due to queues on busy caches. In this paper we pay a particular attention to the impact of the queuing delays experienced on the caches due to their limited access bandwidth to the Internet. The queuing delay becomes very significant when the caches are busy.

A caching hierarchy cannot satisfy all requests arriving to it. Some recent studies show that even for infinite size caches the achievable hit rate in a caching hierarchy is limited [23] [2] [21]. Requests not satisfied on the caching hierarchy are called *misses*. Misses can be classified into: (1) *First-access misses*, which occur when accessing documents the first time; (2) *Capacity misses* which occur when accessing documents previously requested but discarded from the cache to make space for other documents; (3) *Update misses*, which occur when accessing documents previously requested but already expired; (4) *Uncacheable misses*, which occur when accessing documents that need to be delivered from the final server (e.g. dynamic documents generated from cgi scripts). First-access misses are much higher than any other kind of misses [23] and may account for the 20% of all requests. We expect capacity misses to be a secondary issue for large-scale cache architectures because it is becoming very popular to have caches with huge effective storage capacities. We therefore assume that each cache has infinite storage capacity. We also ignore uncacheable misses, as they do not impact significantly the main conclusions of this paper.

2.2 CMP

A CMP distribution takes a popular and frequently changing document and continuously multicasts it on a multicast address. Clients tune into the multicast group for the time required to reliably obtain the document and then they leave the multicast tree. CMP takes place at the transport layer with reliability and congestion control ensured by the end systems (server and clients). In the context of the Internet, CMP requires that the network connecting a server with its clients is multicast capable: a single packet sent by a server will be forwarded along the multicast tree (see Figure 1).

Where the multicast tree forks off, the multicast router will replicate the packets and send a copy on every outgoing branch of the multicast tree. Multicast routers on the Internet were first introduced via an overlay network called MBONE consisting of computers that executed the multicast routing software and that were connected via tunnels. While today the multicast routing software is part of any new router that is installed, not all the existing routers have been enabled to be multicast capable. Therefore, multicast routing on the Internet is not yet everywhere available.

A CMP distribution does not suffer from problems of over-loaded servers or caches. The multicast server does not deal directly with the receivers, reducing the server complexity and thus scaling very well. Receivers always receive the last available update of the document. Additionally, a multicast distribution uses bandwidth

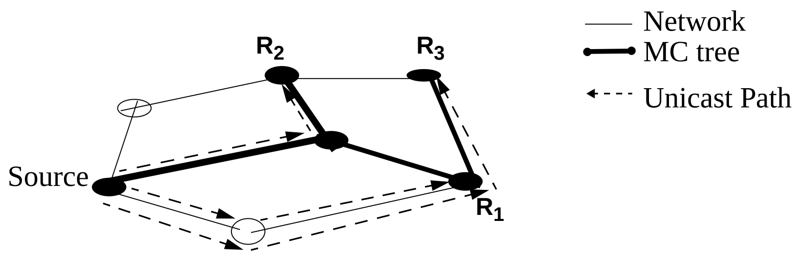


Figure 1: Network topology with multicast tree.

efficiently by sharing *all* common paths between the source and the receivers. Although a cache hierarchy mimics reliable multicast, it does not perform a perfect multicast as within an ISP the same version of the same document can be sent over a link multiple times. (However, there are a number of proposals to have caches communicate via multicast [26] [16]). Thus, CMP is an attractive scheme to deliver hot-changing documents.

However, multicast distribution of Web documents on the Internet is still in its infancy as a viable service; in fact, very few network providers offer it as a service [11]. In particular, a continuous multicast distribution requires an infrastructure with the following components: (1) Session servers or a similar mechanism are needed to map the document's name into a multicast address. (2) A Web server needs to monitor the number of document requests and their rate of change to decide which documents to multicast and when to stop multicasting them. (3) There is an overhead in the multicast capable routers to maintain state information for each active multicast group. (4) There is also an overhead due to the join and prune messages needed for the multicast tree to grow and shrink depending on the location of the receivers. (5) Multicast congestion control is still an open issue.

3 The Model

As shown in Figure 2, the Internet connecting the server and the receivers can be modeled as a hierarchy of ISPs, each ISP with its own autonomous administration. We shall make the reasonable assumption that the Internet hierarchy consists of three tiers of ISPs: institutional networks, regional networks, and national backbones. All of the clients are connected to the institutional networks; the institutional networks are connected to the regional networks; the regional networks are connected to the national networks. The national networks are also connected, sometimes by transoceanic links. We shall focus on a model with two national networks, with one of the national networks containing all of the clients and the other national network containing the origin servers.

In order to have a common basis for the comparison of caching versus multicast, as shown in Figure 3 we model the underlying network topology as a full O-ary tree. Let O be the nodal outdegree of the tree. Let H be the number of network links between the root node of a national network and the root node of a regional network. H is also the number of links between the root node of a regional network and the root node of an institutional network. Let z be the number of links between a origin server and root node (i.e., the international path). Let d be the propagation delay on one link, homogeneous for all links. Let l be the level of the tree. $0 \leq l \leq 2H + z + 1$

We assume that bandwidth is homogeneous within each ISP. Let C_I, C_R, and C_N be the **bandwidth capacity** of the links at the institutional, regional, and national networks. Let C be the bottleneck link capacity on the international path. Receivers are only on the leaves of the tree and not on the intermediate nodes.

3.1 Document Model

In order to not obscure the key points, we make a number of simplifying assumptions. We assume that all documents are of the same size, S bytes. We assume that each LAN issues requests at a rate of β_{LAN}. We assume that there are N hot-changing documents, all of which being candidates for CMP. From each

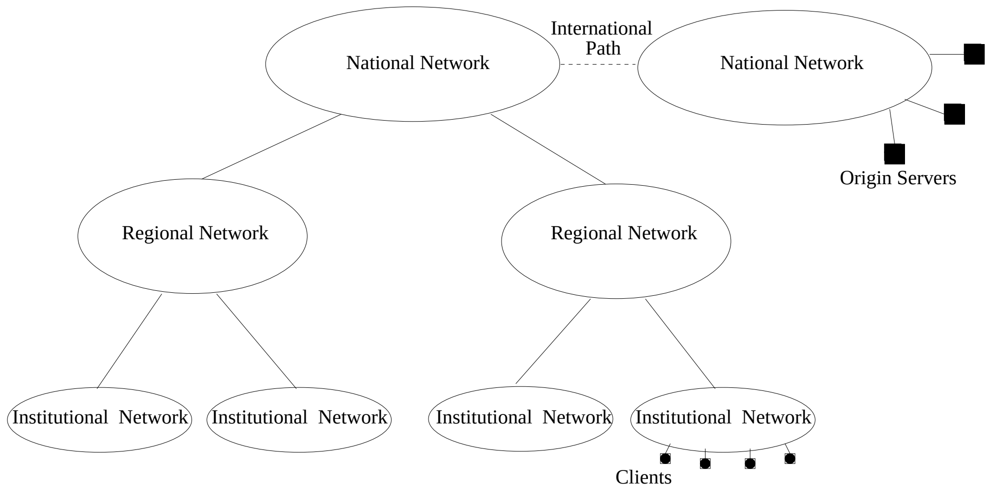


Figure 2: Network topology

LAN the request rate for any one of the hot-changing documents is the same and is denoted by λ_{LAN}. Thus the total request rate from a LAN for the hot-changing documents is $\beta_{LAN}^{HC} = N\lambda_{LAN}$. Denote $\beta_{LAN}^{B} = \beta_{LAN} - \beta_{LAN}^{HC}$ for the rate of the remaining *"background traffic"*. Finally, let Δ be the update period of a hot-changing document. Initially we assume that all hot-changing documents change periodically every Δ seconds. In this case, caches do not need to contact the origin server to check for the document's consistency, and the N hot-changing documents can be removed from the caches every Δ seconds. We shall also consider non-periodic updates. For notational convenience, denote

$$\lambda_l = \lambda_{LAN} \cdot O^{l-1},$$

for the aggregate request rate from all LANs below the multicast tree rooted at level l. Also denote $\lambda_{tot} = \lambda_{LAN} O^{2H}$ for the total request rate, aggregated over all LANs.

3.2 Hierarchical Caching Model

Caches are usually placed at the access points between two different networks to reduce the cost of traveling through a new network. As shown in Figure 4, we make this assumption for all of the network levels. In one country there is one national network with one (logical) national cache. There are O^H regional networks and every one has one (logical) regional cache. There are O^{2H} local networks and every one has one (logical) institutional cache. Caches are placed on height 1 of the tree (level 1 in the cache hierarchy), height $H+1$ of the tree (level 2 in the cache hierarchy), and height $2H+1$ of the tree (level 3 of the hierarchy). If a requested document is not found in the cache hierarchy the national cache requests the document directly from the server.

Caches are connected to their ISPs via access links. We assume that the capacity of the access link at every level is equal to the network link capacity at that level, i.e., C_N, C_R, and C_I for the respective levels.

For simplicity we assume that clients' local caches are disabled. (We could include local client caches in the model, but they would only complicate the analysis without changing the main conclusions.) The **hit rate** is the percentage of requests satisfied by the caching hierarchy. The hit rate for documents at the institutional, the regional, and the national caches is given by HIT_I, HIT_R, HIT_N. Some typical values are $HIT_I = 0.5$, $HIT_R = 0.6$, $HIT_N = 0.7$ [23] [21].

3.3 CMP Model

For CMP we assume that the same hierarchical caching infrastructure is in place, and that most of the documents are distributed with the caching infrastructure. However, the N hot-changing documents are distributed

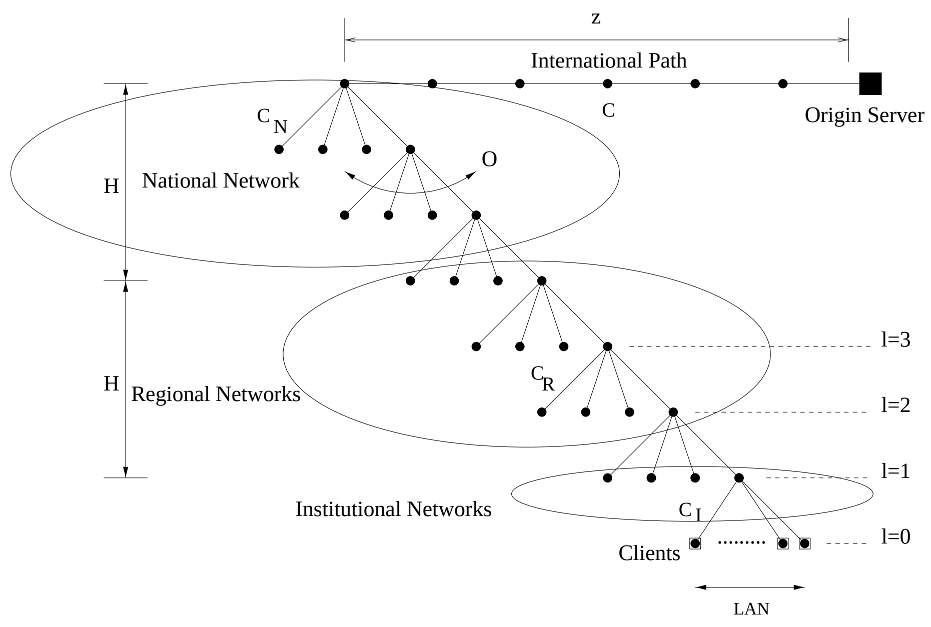


Figure 3: The tree model.

with CMP.

For the multicasting, we assume the origin server to be connected to the clients via a core based tree [4, 7]. The server sends to the core, which is the root for a shortest path tree [8], where a receiver is connected to the core via a shortest path through the network. Let μ_{cmp} be the multicast transmission rate for a single document. In Section 6.2 we shall show how μ_{cmp} can be calculated.

4 Latency Analysis for CMP

In this section we model the expected latency to distribute a hot-changing document by CMP. The average end-to-end latency, denoted by T, has two parts:

1. $E_{cmp}[T_c]$ the connection time. This is the time to join the multicast tree with the document.
2. $E_{cmp}[T_t]$ the transmission time. This is the time to transmit the document, which is equal to the document length S divided by the multicast transmission rate, $E_{cmp}[T_t] = S/\mu_{cmp}$.

Thus, the average latency for a hot-changing document when distributed by CMP is

$$E_{cmp}[T] = E_{cmp}[T_c] + S/\mu_{cmp}$$

We now proceed to calculate $E_{cmp}[T_c]$. We assume that the receiver knows the MC address associated with the Web document, and the paths are symmetric with respect to delay, loss, etc. The connection time is measured up to the point in time where the receiver gets the first bit of the document. Let L be a random variable denoting the number of links traversed to meet the multicast tree. Because we are assuming a propagation delay of d seconds in each direction, the connection time is

$$E_{cmp}[T_c] = 2d \cdot E_{cmp}[L] \tag{1}$$

The expected number of traversed links that a join needs to travel in order to meet the multicast tree is:

$$E_{cmp}[L] \quad = \quad \sum_{l=1}^{2H+z+1} l \cdot P(L = l)$$

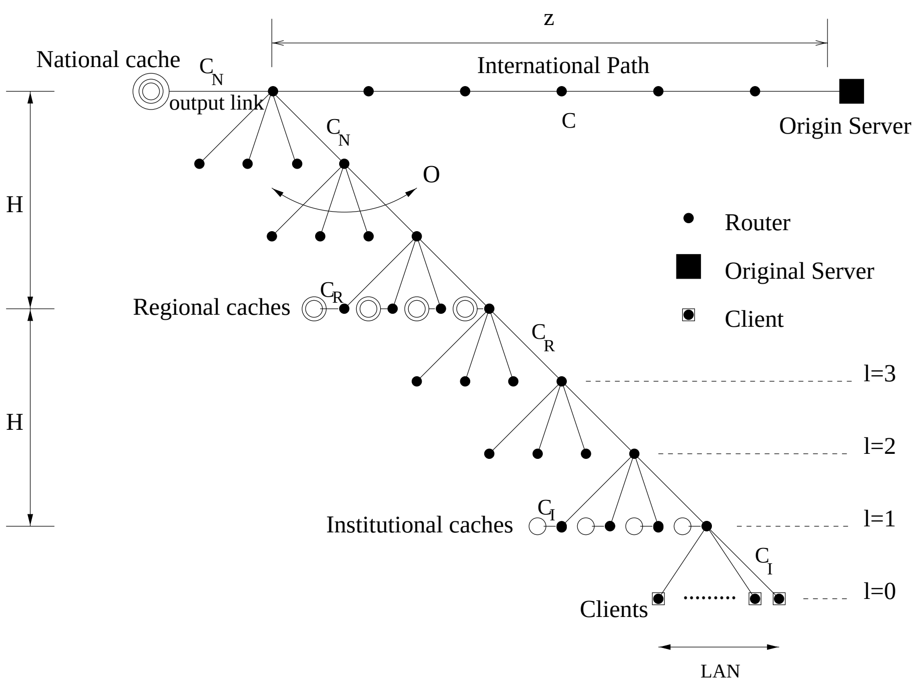


Figure 4: The tree model. Caching placement.

To obtain $P(L = l)$ we use:

$$P(L = l) = \begin{cases} P(L \geq l) - P(L \geq l+1) & 1 \leq l < 2H + z + 1 \\ P(L \geq l) & l = 2H + z + 1 \end{cases}$$

Note that $P(L \geq l)$ is the probability that a new join meets the multicast tree carrying the document at level l or higher, but not before. Clearly $P(L \geq 1) = 1$.

Now consider the sub-tree rooted at level $l - 1$. Requests for a document are generated at a rate λ_{l-1} within this sub-tree. Each request causes the multicast tree to extend on that subtree towards the requesting receiver (if the tree is not already there). Each receiver keeps the tree extended during the transmission time of the document, S/μ_{cmp} The number of requests being serviced is the number of customers in an $M/D/\infty/\infty$ queue with arrival rate λ_{l-1} and service time S/μ_{cmp}. The probability that a join has to travel more than $l - 1$ levels is the probability that there are no customers in the $M/D/\infty/\infty$ queue at level $l - 1$:

$$P(L \geq l) = \begin{cases} 1 & l = 1 \\ e^{(-\lambda_{l-1} \cdot S/\mu_{cmp})} & l \geq 2 \end{cases} \tag{2}$$

Having calculated the distribution of L, we can determine $E_{cmp}[T_c]$ and therefore the total average latency, $E_{cmp}[T]$:

$$E_{cmp}[T] \quad = \quad 2d \cdot \sum_{l=1}^{2H+z+1} l \cdot [e^{(-\lambda_{l-1} \cdot S/\mu_{cmp})} - e^{(-\lambda_l \cdot S/\mu_{cmp})}] + \frac{S}{\mu_{cmp}}$$

5 Latency Analysis for Hierarchical Caching

In this section we determine the average latency for a hot-changing document which is updated periodically every Δ seconds. The average latency for hierarchical caching has two components:

1. $E_{cache}[T_c]$, the time to connect to the document. This is the time to find the document in the nearest server (cache or origin) plus the round-trip times for establishing the TCP connections.
2. $E_{cache}[T_t]$, the transmission time for the document. This is the time to transmit the document from server/cache to client.

We have

$$E_{cache}[T] = E_{cache}[T_c] + E_{cache}[T_t].$$

5.1 Connection Time

We now determine $E_{cache}[T_c]$. Let L denote the number of links traversed to find the document. L is a random variable which takes values in $\{1, H+1, 2H+1, 2H+z+1\}$. We model

$$E_{cache}[T_c] = 4d \cdot E_{cache}[L].$$

The rationale for the $4d$ term is as follows. On a caching hierarchy a TCP connection is opened between every caching level before starting the transmission of the Web document. In a multicast distribution the delivery is open-loop and there is no previous connection set-up requirement. A TCP connection uses a three-way handshake protocol that increases the number of links traversed on a caching hierarchy $E_{cache}[L]$ by a factor 2 compared to a multicast distribution. We assume that the operating system in the cache gives priority at establishing TCP connections.

We now proceed to calculate the distribution of L for the caching hierarchy. To obtain $P(L = l)$ we use

$$\begin{aligned}
P(L = 1) &= 1 - P(L \geq H+1) \\
P(L = H+1) &= P(L \geq H+1) - P(L \geq 2H+1) \\
P(L = 2H+1) &= P(L \geq 2H+1) - P(L \geq 2H+z+1) \\
P(L = 2H+z+1) &= P(L \geq 2H+z+1)
\end{aligned}$$

Note that $P(L \geq l)$ is the probability that the document is present at level l or higher, but not before. Consider the term $P(L \geq H+1)$. This is the probability that a request from a LAN does not find the document in the institutional cache. Let t denote the time into the interval $[0, \Delta]$ at which a request occurs. The random variable t is uniformly distributed over the interval. Thus

$$P(L \geq H+1) = \frac{1}{\Delta}\int_0^{\Delta} P(L \geq H+1 | t = \tau) d\tau$$

Now $P(L \geq H+1 | t = \tau)$ is the probability that no request from the same LAN arrive in the interval $[0, \tau]$, i.e.

$$P(L \geq H+1 | t = \tau) = e^{-\lambda_{LAN} \cdot \tau}$$

Combining these two formulas and integrating gives

$$P(L \geq H+1) = \frac{1}{\lambda_{LAN} \cdot \Delta}(1 - e^{-\lambda_{LAN} \cdot \Delta})$$

Similarly

$$\begin{aligned}
P(L \geq 2H+1) &= \frac{1}{O^H \cdot \lambda_{LAN} \cdot \Delta}(1 - e^{-O^H \lambda_{LAN} \Delta}) \\
P(L \geq 2H+z+1) &= \frac{1}{O^{2h} \cdot \lambda_{LAN} \cdot \Delta}(1 - e^{-O^{2H} \lambda_{LAN} \Delta})
\end{aligned}$$

Notice that $\lambda_{LAN}\Delta$ is the expected number of requests from a LAN is an update period. As $\lambda_{LAN}\Delta \to \infty$, the above three probabilities approach zero because with high probability the document is in the institutional cache. On the other hand, as $\lambda_{LAN}\Delta \to 0$, the above probabilities approach one because with high probability the document is only available at the origin server.

5.2 Transmission Time

The transmission time is the time to send the document from server to client once all the TCP connections are in place. We make the realistic assumption that the caches operate in a cut-through mode rather than a store-and-forward mode, i.e., when a cache begins to receive a document it immediately transmits the document to the subsequent cache (or client) while the document is being received.

The transmission time depends on L, the closest level with a copy of the document:

$$E_{cache}[T_t] = \sum_{l \in \{1, H+1, 2H+1, 2H+z+1\}} E_{cache}[T_t | L = l] \cdot P(L = l)$$

Recall that the distribution of L is given in previous subsection. We now procede to calculate $E_{cache}[T_t | L = l]$. To this end, we determine the aggregate request arrival rate for each of the caches. Denote the request arrival rate for the instituional, regional, and national caches by β_I, β_R and β_N, respectively. The average request rate for all documents arriving to a cache at level l is filtered by the hits at lower caches:

$$\beta_I = \beta_{LAN} \qquad l = 1$$

$$\beta_R = O^H \beta_{LAN}(1 - HIT_I) \quad l = H + 1$$

$$\beta_N = O^{2H} \beta_{LAN}(1 - HIT_R) \quad l = 2H + 1$$

In order to estimate $E_{cache}[T_t | L = l]$ we use a simple M/D/1 queue to model the queuing delays on the ouput links of the instutional, regional and national networks; see Figure 5.

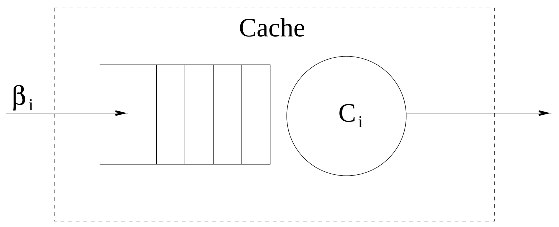


Figure 5: Queueing model for the load on the caches.

Denote D_j for the delay at a given level of the caching hierarchy. This delay accounts for the queueing time plus the service time S/C_j. The M/D/1 theory gives:

$$D_j = \frac{S}{C_j - \beta_j S} \cdot (1 - \frac{\beta_j S}{2C_j}) \quad j \in \{I, R, N\}$$

When a document is hit at the national cache, it first experiences a delay due to the queueing and service time at that national cache. Then, the document is forwarded to the regional level. If the regional cache is idle ($\beta_R S/C_R \approx 0$) the document is forwarded to the institutional caches without any additional delay. However, in the case that the regional cache is very congested ($\beta_R S/C_R \approx 1$) the document will experience an additional delay due to the queuing time plus the service time at the regional cache. The same happens when the document is sent to the institutional cache. Therefore, in the case that all caches at the national, regional, and institutional levels are very congested the document will experience a delay equal to the sum of the delays at each of these levels; even in the cut-through mode. On the other hand, if only the national cache is congested and the regional and institutional caches are idle, the delay experienced by the document is only the one at the national cache.

Now consider $E_{cache}[T_t | L = l]$. For $l = 1$, we clearly have $E[T_t | L = l] = D_I$. For $l = H + 1$, the document first experiences the delay D_R at the regional cache. Then, the document is forwarded to the institutional cache. At the institutional cache, the document experiences an additional delay D_I which accounts for the queueing time plus the service time at that cache. When the institutional cache is idle the document does not experience any additional queueing or service time. The queueing time is zero when the cache is idle. The factor $S/C_I \cdot (1 - \beta_I S/C_I)$ is intented to substract the service time from the delay D_I when the cache is idle. The cases $l = 2H + 1$ and $l = 2H + z + 1$ are analogous.

$$E_{cache}[T_t|L = l] = \begin{cases} D_I & l = 1 \\ D_R + D_I - \frac{S}{C_I} \cdot (1 - \frac{\beta_I S}{C_I}) & l = H + 1 \\ D_N + D_R + D_I - \frac{S}{C_R} \cdot (1 - \frac{\beta_R S}{C_R}) - \frac{S}{C_I} \cdot (1 - \frac{\beta_I S}{C_I}) & l = 2H + 1 \\ \frac{S}{\mu_{cache}} + D_N + D_R + D_I - \frac{S}{C_N} \cdot (1 - \frac{\beta_N S}{C_N}) - \frac{S}{C_R} \cdot (1 - \frac{\beta_R S}{C_R}) - & l = 2H + z + 1 \\ \frac{S}{C_I} \cdot (1 - \frac{\beta_I S}{C_I}) & \end{cases} \quad (3)$$

where μ_{cache} is the caching transmission rate for a single document on the International Path. In Section 6.2 we consider in more detail how to calculate μ_{cache}.

6 CMP vs Hierarchical Caching: Numerical Comparison

The following parameteres will be fixed for the remainder of the paper, except when stated differently. The network is modelled with $O = 4$ as nodal outdegree of the MC tree; $H = 3$ as the distance between cache hierarchy levels, yielding $O^H = 64$ regional caches and $O^{2H} = 4096$ institutional caches; $z = H = 3$ as the distance in the International Path.

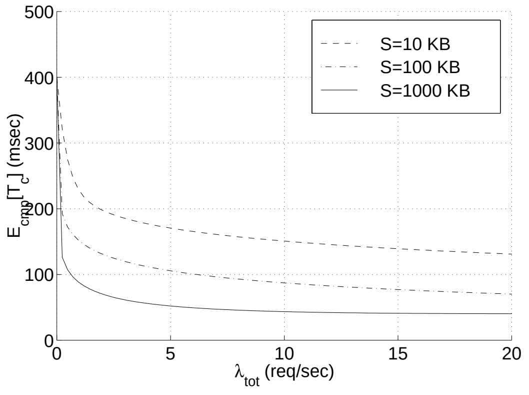
6.1 Connection Time

The connection time for a CMP and a cache distribution depends on how close the document is to the receivers at every moment. In a CMP distribution, a document is at a certain level of the multicast tree for a time equal to the transmission time S/μ_{cmp} regardless of its update period Δ. On a caching distribution a document is at a certain level of the caching hierarchy for a time equal to the update period Δ regardless of its document size S (given an infinite cache size). If the cache space is limited a document can be also removed from a cache due to space constraints.

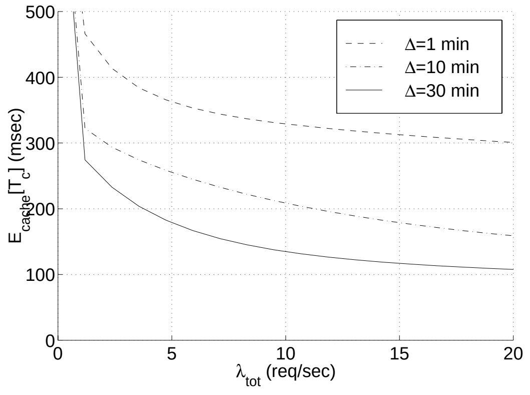
Figure 6(a) shows the Connection time for a CMP distribution $E_{cmp}[T_c]$ depending on different document sizes S and on the total request rate λ_{tot}. The values for the parameters μ_{cmp} and d are taken from recent caching studies [23] [21] [3]. We model different transmission times S/μ_{cmp} for a document by varying the document size. When the number of requests is very small, it is very likely that a join has to travel all the way to the original server in order to meet the multicast tree. An arriving request can not share any branch of the multicast tree built for past requests because it is already shrunk. When the number of requests is high a new request will meet the multicast tree at a lower level. For very popular documents regardless of its update period Δ the multicast tree is always very close to the receivers.

As we see in Figure 6(a), the connection time of a CMP distribution clearly depends on the document size S. For very small documents, the transmission time is very small. The tree shrinks very frequently, reducing the probability of meeting the tree at a low level. For larger documents the tree is kept extended for a longer time reducing the connection time of new requests.

Figure 6(b) shows the connection time for a caching distribution: $E_{cache}[T_c]$ depends on λ_{tot} and the update period Δ. We see that if the document is rarely requested, the average number of travelled links



(a) Multicast



(b) Caching

Figure 6: Connection time $E_{cmp}[T_c]$ for a multicast and connection time $E_{cache}[T_c]$for a caching distribution, depending on the total request rate for different update periods Δ and document sizes S. $\mu_{cmp} = 1$ KB/sec. $d = 20$ msec.

needed to meet a cache with an up-to-date document is high. For high request rates, a newly arriving request meets the up-to-date document at a closer caching level.

The main reasons that explain why the Connection time on a CMP distribution is shorter than that on a caching distribution are the following: 1) A multicast tree has a higher degree of granularity than a caching hierarchy. If a document is not hit at level l it can be hit at level $l - 1$. However, on a caching hierarchy if a document is not hit at level l, the closest level where the document can be hit is at level $l - H$. 2) When a document is very popular the multicast tree is always very close to the receivers independently on how fast the document changes. In a caching distribution even if a document is very popular the number of links traversed to hit the document is very much determined by the update period Δ. 3) A multicast distribution is an open-loop distribution where no connection is established. A caching distribution uses TCP to previously establish a connection, which adds a factor of 2 to the Connection time.

6.2 Transmission Time

In order to provide results for $E_{cmp}[T_t]$ and $E_{cache}[T_t]$, we need to calculate μ_{cmp} and μ_{cache}. First, we calculate the transmission rate μ_{cmp} for a hot-changing document in a CMP distribution . Then, we argue that the transmission rate μ_{cache} (equation 3) for a hot-changing document in a caching distribution when the document is hit at the original server is $\mu_{cache} = \mu_{cmp}$. The transmission rate μ_{cmp} for a hot-changing document is determined by the minimum transmission rate at any level of the network. The link with the most traffic and the lowest capacity is the international link. Therefore, the international link is the bottleneck for the end-to-end multicast transmission. In this calculation, we assume that there is at least one interested receiver for each of the N hot-changing documents at every moment. In this situation, a multicast distribution needs to continuously send the N documents from the original server to all LANs. The available end-to-end CMP transmission rate for the hot-changing documents is equal to the capacity C on the international path minus the capacity needed for the background traffic that is not satisfied by the caching hierarchy. If this capacity is equally shared by the N hot-changing documents, the CMP transmission rate for a hot-changing document is given by:

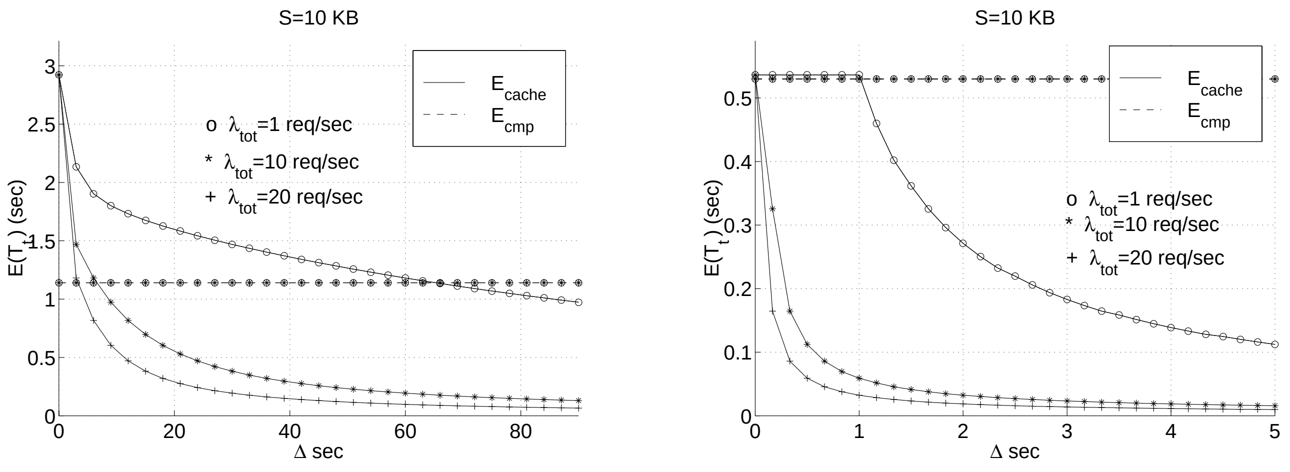
$$\mu_{cmp} = \frac{C - \beta_{LAN}^{B} O^{2H}(1 - Hit_N) \cdot S}{N}$$

Next, we argue that $\mu_{cmp} = \mu_{cache}$. A CMP distribution is continuously sending the N hot-changing documents at a rate μ_{cmp}. If the hot-changing documents change so frequently that every document request sees a document update, a caching distribution resembles to a CMP distribution in the sense that the N hot-changing documents need also to be continuosly delivered from the original server. In this situation the rate μ_{cache} at the International Path on a caching distribution is equal to the end-to-end CMP rate $\mu_{cache} = \mu_{cmp}$. When several requests see an unmodified document, only one copy of the document is sent through the international path to the caching hierarchy every period Δ. The rest of the requests inside that period Δ are satisfied from local copies at lower level caches. In this situation the available caching transmission rate μ_{cache} for one hot-changing document through the international path is higher than the CMP rate μ_{cmp}. Therefore considering $\mu_{cache} = \mu_{cmp}$ is being pessimistic for the transmission time in a caching distribution through the international path.

6.3 Numerical Analysis

We pick some typical values for the different parameters in the model so that we can show some quantitative results. We take $\beta^{B}_{LAN}/\beta^{H,C}_{LAN} = 10$, meaning that the request rate for hot-changing documents is ten times lower than the request rate for the rest of the traffic. This ratio may vary if more documents appear to be frequently changing. We have also chosen some indicative values for the link capacities at the different hierarchy levels: $C_I = 100$ Mbps, $C_R = 45$ Mbps, $C_N = 45$ Mbps, $C = 34$ Mbps.

We analyze two different scenarios: (1) The access link of the national cache is being used close to its full capacity and therefore the queuing delays on the national cache increase the caching latency; (2) The access link of the national cache is not being used close to its maximum capacity and the bottleneck for a caching distribution is the limited bandwidth of International Path. For a CMP distribution the bottleneck is always in the International Path. We model these two different scenarios by varying β_N. Choosing $\beta_N S = 0.9C_N$ we can model the first scenario. Decrasing β_N to $\beta_N S = 0.6C_N$, we can model the second scenario. Given the two scenarios we calculate the transmission and total latency for a hot-changing document that is distributed through a caching or a CMP model. Varying the update period Δ and the total request rate λ_{tot}, we determine when caching has lower latency then CMP.



(a) The bottleneck is given by the limited bandwidth on the access link of the national cache. $\beta_N S = 0.9 \cdot C_N$

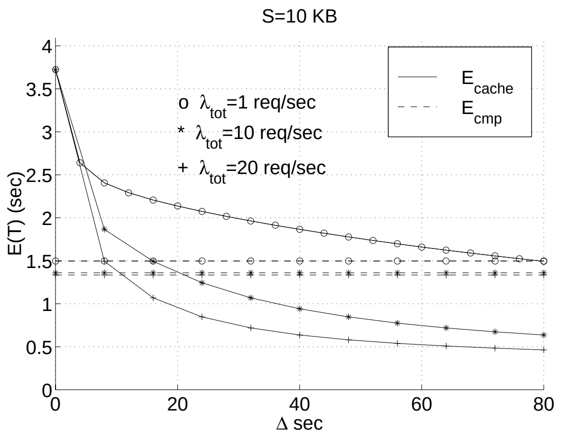
(b) The bottleneck is given by the limited bandwidth on the International Path. $\beta_N S = 0.6 \cdot C_N$

Figure 7: Cache and CMP transmission time T_t for a hot-changing document for different Δ. $C_I = 100$ Mbps, $C_R = 45$ Mbps, $C_N = 45$ Mbps, $C = 34$ Mbps. $S = 10$ KB.

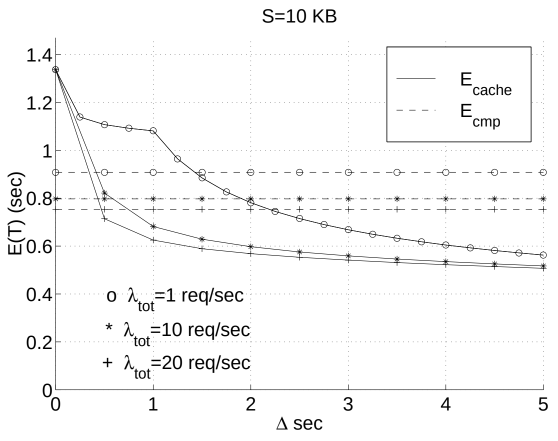
In Figure 7 we plot the transmission time for a document on a caching and a CMP distribution depending on the update period of the document for the two different scenarios. From Figure 7(a) we see that if the bottleneck for a caching distribution is given by the limited access link of the national cache, a caching distribution of frequently-changing documents has higher transmission times than a CMP distribution. However, when the bottleneck for both a CMP and a caching distribution is placed on the International Path (Figure 7(b)), a frequently-changing document will have the same transmission time in a caching distribution and a CMP distribution. When the document does not change more frequently than a few seconds 7(b) or a few tens of seconds 7(a) a caching distribution always has a lower transmission time than a CMP distribution.

6.4 Total Latency T

On Figure 8 we plot the total latency T comprising the connection time T_c and the transmission time T_t for CMP and caching. We have also considered both bottleneck scenarios described on the previous section.



(a) The bottleneck is given by the limited bandwidth on the access link of the national cache. $\beta_N S = 0.9 \cdot C_N$



(b) The bottleneck is given by the limited bandwidth on the International Path. $\beta_N S = 0.6 \cdot C_N$

Figure 8: Cache and CMP Total Latency T for a hot-changing document for different Δ. $C_I = 100$ Mbps, $C_R = 45$ Mbps, $C_N = 45$ Mbps, $C = 34$ Mbps. $S = 10$ KB.

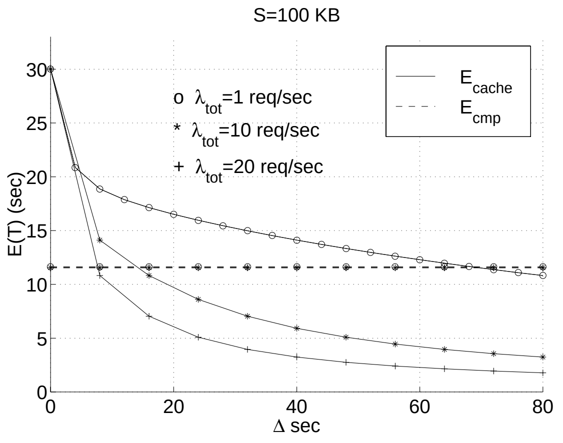
Comparing Figure 7 and Figure 8 we see that adding the connection time T_c to the transmission time T_t increases the latency differences between a caching distribution and a CMP distribution for a frequently-changing document. Additionally, we see that the curves are displaced to the right increasing the turning point where caching is better than CMP.

From Figure 8(a) we observe that if the access link of the national cache is very congested and the documents change faster than several tens of seconds a CMP distribution has a lower total latency than a caching distribution. When the access link of the national cache is less congested (Figure 8(b)), the value of Δ where caching becomes preferable than CMP gets smaller.

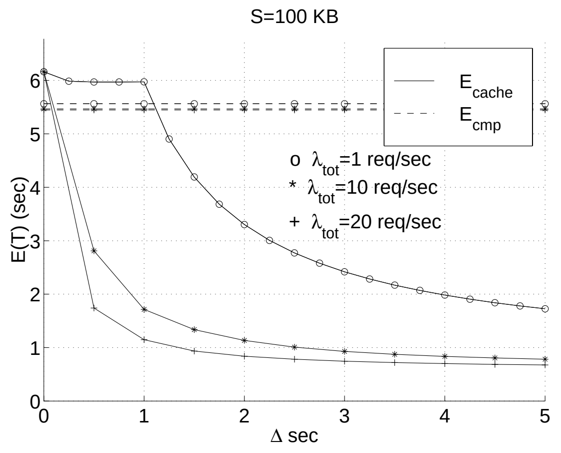
The total latency of a CMP distribution is almost insensitive to changes in the request rate λ_{tot}. Only the connection time of a CMP distribution depends on λ_{tot}; the transmission time of a CMP distribution is independent λ_{tot}. The latency in a caching distribution has a higher dependency on the popularity of the document (given by λ_{tot}) than in a CMP distribution. Both, the connection time and the transmission time of a caching distribution depend on λ_{tot}. For higher λ_{tot} the document is found at lower levels which have higher capacities. The less popular the document, the higher is the latency time to retreive a document in a caching hierarchy.

Note that we only consider documents that are very popular, which is why λ_{tot} takes very high values. We claim that non-popular documents should not be sent via CMP because the bandwidth savings are not significant while there is significant overhead in maintaining many multicast trees.

On Figure 9 we plot the transmission time for a larger document $S = 100$ KB. We see that when the document size incresases the the latency differences between a caching distribution and a CMP distribution get reduced for those values of Δ where a CMP distribution is better than a caching distribution. This is because for large document sizes, the connection time contributes less to the total latency than the transmission time. A surprising result is that varying the document sizes, the value of Δ for which a CMP distribution has lower latency than a caching distribution varies slightly. This is an interesting result because it suggests that the point at which the CMP distribution is better than the caching distribution does not greatly depend on the document size.



(a) The bottleneck is given by the limited bandwidth on the access link of the national cache. $\beta_N S = 0.9 \cdot C_N$



(b) The bottleneck is given by the limited bandwidth on the International Path. $\beta_N S = 0.6 \cdot C_N$

Figure 9: Cache and CMP Total Latency T for a hot-changing document for different Δ. $N = 100$. $C_I = 100$ Mbps, $C_R = 45$ Mbps, $C_N = 45$ Mbps, $C = 34$ Mbps. $S = 100$ KB.

In the case that the access link of the national cache is very congested one could be tempted to re-direct all requests at the regional caches to the origin server. However redirecting requests for Hot-Changing documents from the regional level to the origin server increases the traffic in the International link and the load in the origin servers. The reasoning is the following. For a Hot-Changing document, it is likely that all regional caches are interested in the last document update. The national cache only asks for one copy of a document to the origin server and then forwards it to all interested regional caches. If the regional caches would directly request a Hot-Changing document from the origin server, there would be O^H copies of the same document being sent from the origin server through the International Path instead of one.

For the Background traffic, documents are not so popular or they do not change so fast. Therefore re-directing requests to the origin servers does not place such an additional traffic overhead on the International Path or such a load on the server. On the other hand, the congestion problems on the access link of the national cache are avoided. A caching scheme that is able to avoid congested caches is the ICP caching resolution protocol [26]. When a cache does not have a document, it sends ICP queries to all its siblings, parents and to the origin server. If the national cache is very loaded and the document is not in the sibling caches the document is directly requested from the origin server avoiding the congested national cache.

Another way to alleviate the bottleneck on the access link from the national cache to the national network is by installing multiple national caches each with its own access link. Documents can be partitioned between the different national caches. A national cache can share its documents with another national cache via a hash function [20]. In this manner the load is distributed among the national caches reducing the queuing delays. Similarly, ICP can be used to distribute the load between the multiple national caches.

6.5 Random Updates

For periodic updates, the caches do not need to poll the source to check the document's consitency. Indeed, the document is removed from the caches after a period Δ; if the document is in the cache, then the document is up-to-date. For random document updates, if caches want to provide strong consistency to the receivers, the caches need to poll the server for every request. In this case, the number of links traversed in the caching connection time is equal to the number of links between the receivers and the origin server, i.e., $E_{cache}[L] = 10$. On a CMP distribution the connection time does not depend on the update period Δ of a document. Looking at Figure 10 we observe that a CMP distribution has always lower connection times than a caching distribution when strong document consistency is required.

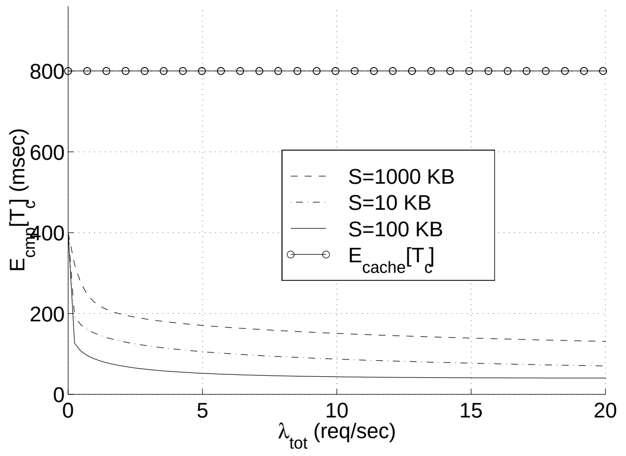


Figure 10: Connection time on a CMP and a caching distribution $E_{cmp}[T_c]$, $E_{cache}[T_c]$ when documents are randomly updated.

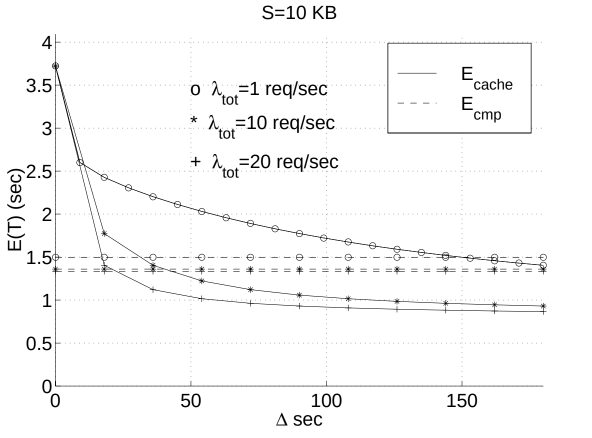
Assuming that the update period is exponentially distributed with average Δ, then the distribution of the number of requests R in an update period is:

$$P(R = r) = (1 - \gamma) \cdot \gamma^r , \tag{4}$$

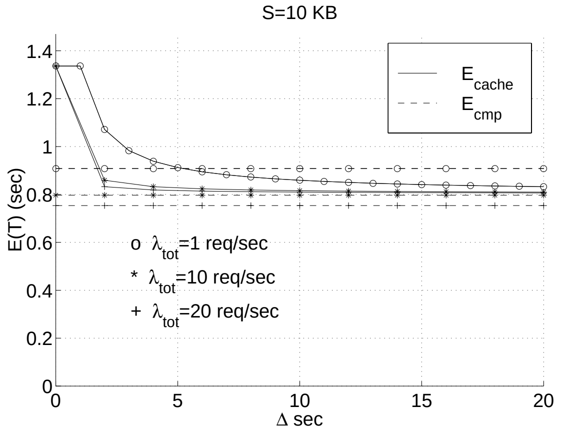
where $\gamma = (\lambda_{tot}\Delta)/(\lambda_{tot}\Delta + 1)$. From the distribution of R we can calculate the distribution of L, the transmission time (Section 5), and the total latency for a caching distribution when the documents are randomly updated. In Figure 11 we plot the total latency $E_{cmp}[T]$ and $E_{cache}[T]$ for random updates given that the access link of the national cache is a bottleneck and given that the bottleneck is on the international path. From Figure 11(a) we see that for random updates the total latency for a caching distribution increases compared to the case where the documents change periodically (Figure 8(a)). Additionally the value of Δ for which CMP beats caching increases.

For small document sizes $S = 10$ KB, the connection time becomes very relevant especially when the network is not congested. In Figure 11(b) the access link of the national cache is not congested and the connection time predominates over the transmission time. Given that for random updates the connection time is always higher on a caching distribution than on a CMP distribution (Figure 10) the total latency of a caching distribution is higher than the total latency of a CMP distribution even for higher values of Δ.

Therefore, in the case that a document is randomly updated a caching distribution needs to poll every time the server, increasing the total latency over that in a CMP distribution. One way to avoid having caches check for the document's freshness is to use a source-intitiated invalidation scheme [14]. The origin server sends invalidation messages to communicate the caches that a certain document has expired. The caches do not need to worry about the consistency of the document; if the document is in the cache, it is up-to-date; if the document is not in the cache, it is because no one has yet asked for the last version of the document.



(a) The bottleneck is given by the limited bandwidth on the access link of the national cache. $\beta_N S = 0.9 \cdot C_N$

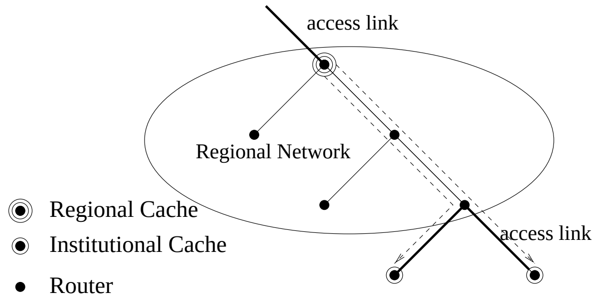


(b) The bottleneck is given by the limited bandwidth on the International Path. $\beta_N S = 0.6 \cdot C_N$

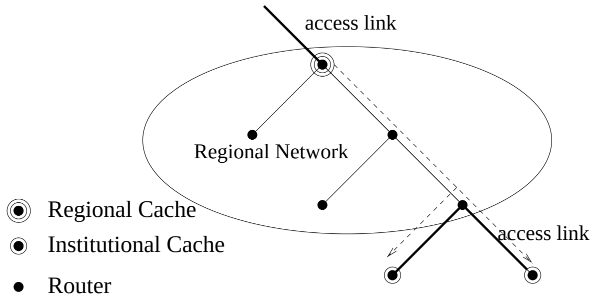
Figure 11: Total Latency T on a multicast distribution and a caching distribution when documents are randomly updated.

7 Bandwidth

Instutional networks are connected to regional ISPs via access links, and regional ISPs are connected to national ISPs via access links (see Figure 12). While end users are concerned with the retrieval latency, ISPs are mainly concerned with bandwidth usage inside their network and bandwidth usage in their access links. In this section we calculate i) the bandwidth usage in the access links and ii) the bandwidth usage inside every ISP.



(a)



(b)

Figure 12: Caching distribution (left) and Multicast distribution (right).

When a popular Web document expires frequently, a caching hierarchy resembles a CMP distribution in the sense that a new document update is continuously transmitted from the origin server to the receivers. However, within an ISP a caching distribution requires more network bandwidth than a multicast distribution because the communication between the different cache levels is done via unicast and not via multicast (Figure 12). When a popular document does not change frequently, a CMP distribution uses more bandwidth (in access link and within ISP) than a cache distribution because a CMP distribution sends the same document over and over the same links while a cache distribution only sends the document once every period Δ. We now develop a model that quantifies these observations.

7.1 Bandwidth in the Access Link

We first calculate the **bandwidth usage** in the access link for a cache distribution, we then calculate the same bandwidth usage for a CMP distribution. We do the analysis for the access link than joins the institutional network to the regional network, and for the access link that joins the regional network to the national network.

First consider a caching hierarchy. The average bandwidth usage by a hot-changing document in the access link that connects the institutional network to the regional network is given by:

$$BW_{cache} = \frac{S}{\Delta} \cdot [1 - exp(-\lambda_{LAN} \cdot \Delta)]$$

where $1 - exp(-\lambda_{LAN} \cdot \Delta)$ is the probability that the document is sent across the institutional-regional access link in a period Δ. For the regional-national access link this probability is given by $1 - exp(-\lambda_{LAN} \cdot O^H \cdot \Delta)$, which is higher than the probability for the institutional-regional access link because the number of receivers that are satisfied through the regional-national link is higher.

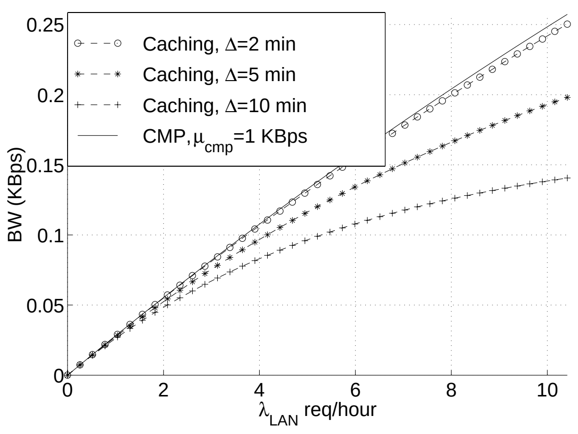
For a CMP distribution we assume that the multicast server is sending at a constant rate μ_{cmp}. The average bandwidth usage in the access link depends on the probability that the multicast tree is extended through that access link. For the institutional-regional access link the bandwidth is given by:

$$BW_{cmp} = \frac{S}{\Delta} \cdot P(L' = 1)$$

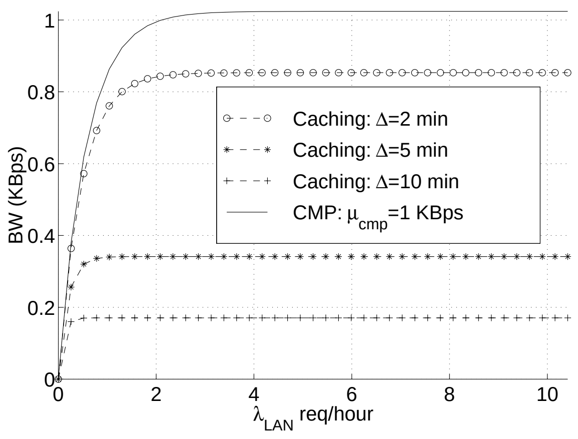
where $P(L' = 1)$ is the probability that the multicast tree is extended to level $l = 1$. From equation 2:

$$P(L' = 1) = 1 - exp(-\lambda_{LAN} \cdot S/\mu_{cmp})$$

For the regional-national access link the probability that the multicast tree is extended through level $l = H + 1$ is given by $1 - exp(-\lambda_{LAN} \cdot O^H \cdot S/\mu_{cmp})$.



(a) Bandwidth usage in the institutional-regional access link.



(b) Bandwidth usage in the regional-national access link.

Figure 13: Bandwidth used by a caching distribution and a CMP distribution in the access link of an institutional network and a regional network. $S = 100$ KB. $\mu_{cmp} = 1$ KBps.

Figure 14 shows the access link bandwidth usage by a CMP distribution and a cache distribution depending on the request rate from a LAN. We take $S = 100$ KB and $\mu_{cmp} = 1$ KBps as illustrative values. We see that when the Web document changes every 2 minutes or faster, both a CMP distribution and a cache distribution use similar bandwidth in the access link. However, when the Web document changes less often a CMP distribution sends more copies of the same document through the access link than a cache distribution, resulting in a higher bandwidth usage. The bandwidth differences betweeen a CMP distribution and a caching distribution are higher in the access link that joins the regional and the national networks than in the access link that joins the institutional and the regional networks (see Figure 13(a) and Figure 13(b)).

7.2 Bandwidth in the ISP

In this section we calculate the bandwidth used inside one ISP. We consider an ISP at the regional level and an ISP at the national level.

First consider a cache distribution. The bandwidth usage by a cache distribution inside a regional ISP is given by:

$$BW_{cache} = (H-1) \cdot O^H \cdot [1 - exp(-\lambda_{LAN} \cdot \Delta)] \cdot \frac{S}{\Delta}$$

where $H-1$ is the number of network levels inside the regional ISP (without considering the access link) and $O^H \cdot [1 - exp(-\lambda_{LAN} \cdot \Delta)]$ is the expected number of transmissions through any network level of the regional ISP. For a national ISP the expected number of transmissions through any of its network levels is given by $O^H \cdot [1 - exp(-\lambda_{LAN} \cdot O^H \cdot \Delta)]$

When a CMP distribution is used, the bandwidth in a regional ISP is given by:

$$BW_{cmp} = \mu_{cmp} \cdot \sum_{l=2}^{H} O^{H-l+1} \cdot P(L' = l)$$

where $P(L' = l)$ is the probability that the multicast tree is extended to level l

$$P(L' = l) = 1 - exp(-O^{l-1} \cdot \lambda_{LAN} \cdot S/\mu_{cmp})$$

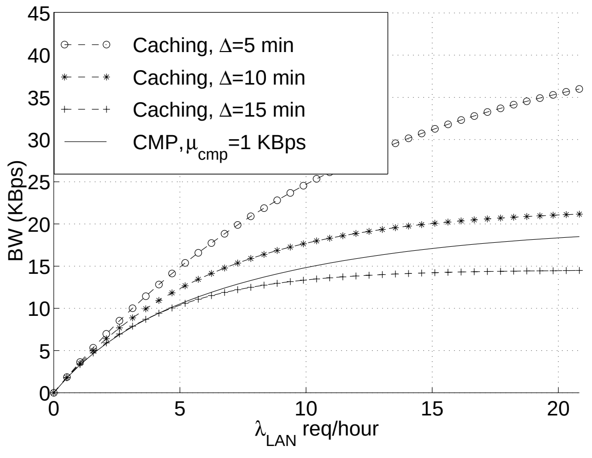
For the national ISP l takes values between $H+2$ and $2H$.

Figure 14 shows the total bandwidth usage inside a regional ISP and a national ISP when Web documents are distributed via hierarchical caching or via CMP. When a Web document changes every 10 minutes or faster, hierarchical caching uses more bandwidth than a CMP distribution. This is because hiearchical caching uses unicast transmissions between the different levels of the caching hiearchy while CMP shares all the common paths inside the network. However, if the document changes every 15 minutes or less then a CMP distribution uses more bandwidth than hierarchical caching because CMP is sending several copies of the same up-to-date document through the same links.

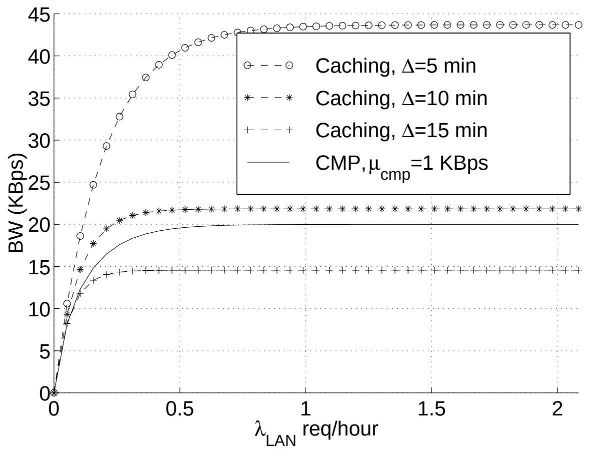
8 Caching and Multicast: Push Caching

In this section we propose a mechanism that combines caching and multicast to reduce the latency to the receivers.

There are several solutions to improve the transmission time on a caching scheme: 1) Increase the bandwidth of the links. Increasing the bandwidth the problem disappears. However, there can always be new applications with bigger documents that consume again the available bandwidth. Sometimes the problem is not the bandwidth in the links but the cache processing power. In this case more machines are needed to cooperate sharing the load to reduce the latency. 2) Reduce the request rate at every cache by distributing the documents over several caches. A hash function can be used to locate the copy of a document. This changes the topology model from a hierarchical topology to a *distributed* topology [23]. 3) Use bandwidth more efficiently. Multicasting updates for popular documents from one cache level to the other saves bandwidth on the access links.



(a) Bandwidth usage in the regional ISP.



(b) Bandwidth usage in the national ISP.

Figure 14: Bandwidth used by a caching distribution and a CMP distribution inside a regional ISP and a national ISP. $S = 100$ KB. $\mu_{cmp} = 1$ KBps.

As we have seen on the previous sections, a multicast distribution is a very efficient way to distribute documents to a high number of synchronised receivers, reducing the bandwidth consumption. On the other hand, a caching distribution resembles a multicast distribution with memory capacity on each of its nodes, allowing for fast local retransmissions. The ideal scenario would be the case where an origin server could previously know which of the caches are interested in a document. Then, the document could be multicasted from the origin server towards the interested caches. Knowing in advance which documents are of interest for the caches is not an easy problem. However, this is an easier task in the case of very popular documents or in the case of a subscription model.

Prefetching Hot-Changing documents in caches closer to receivers, the model changes from a *receiver-initiated* caching scheme to a *push-caching* scheme [12] [23]. Multicasting documents in advance to the institutional caches i) reduces the bandwidth usage, ii) reduces the connection time to the time to connect the institutional cache, and iii) reduces the transmission time because the transmission rates at the low hierarchy levels are higher. Not only Hot-Changing documents can be pushed, however more aggressive push schemes require that the available disk space in the cache is not a constraint.

A caching-multicast cooperation could work like this: 1) The origin Web server monitors the popularity of its documents and when they expire. 2) Every time that a popular document changes, the Web server can take the decision to multicast the document update towards all the national caches. 4) The national caches keep track of which documents are popular for their children-caches. Based on this information the national caches decide to forward the document update or to remove it, performing a *geographical filtering*. 5) The national caches that have regional children-caches interested in that document update, will forward the update towards all their regional caches via multicast. 6) The regional caches will do the same process with the institutional caches.

The drawback of this approach is that some caches may receive a document update for which they are not interested. One possible solution would be to announce the document update previously on a signaling multicast group. All interested caches would join the multicast group and receive the corresponding document update leaving the group later.

9 Conclusions

A caching hierarchy requires ISPs to invest in caches. But ISPs are aggressively introducing caches throughout the world in order to reduce average latency and bandwidth usage. Furthermore, no fundamental changes need to be made to TCP/IP or routers in order to introduce caches into the Internet. On the otherhand, multicast protocols operate at the network and transport layers. Because multicast requires fundamental changes in the Internet, widespread deployment of multicast in the Internet continues to be a slow process. Widespread deployment of reliable multicast is many years away.

One of the principle applications of reliable multicast is the distribution of Web documents. In this paper we show that unless a Web document changes very frequently, caching distribution gives lower latency and bandwidth usage than multicast. The reduced latency is principly because with multicast the transmission rate for a document is the bottleneck rate between server and client; for caching, the transmission rate is the bottleneck rate between the nearest cache with the document and the client. We also observe that a caching hierarchy provides a reliable multicast service with local recovery. The multicasting is performed on a cache-to-cache basis, with tunneling across routers between neigboring caches.

If a document changes very frequently, then CMP gives lower average latency than a cache hierarchy. This is because in a cache distribution it is neccesary to check for documents' consistency before documents can be delivered to the receivers. Additionally with caching top-level caches can become congested resulting in high queueing delays.

10 Acknowledgments

The support of P.Rodriguez by the European Commission in form of a TMR (Training and Mobility for Researchers in Europe) fellowship (ERBFMBICT972639) is gratefully acknowledged. We would like to thank Jörg Nonnenmacher for his help and for the fruitful discussions that we had. Eurecom's research is partially supported by its industrial partners: Ascom, Cegetel, France Telecom, Hitachi, IBM France, Motorola, Swisscom, Texas Instruments, and Thomson CSF.

11 Vitae

Pablo Rodriguez

Pablo Rodriguez (rodrigue@eurecom.fr) was born in Oviedo, Spain in 1972. He obtained his M.Sc. degree as a Telecommunication Engineer from the University of Navarra UPNA, Spain in 1995. He did his Master Thesis at the department of Electrical and Electronic Engineering at King's College, London. Then, he moved to the Swiss Federal Institute of Technology EPFL, Lausanne where he did Postgraduate Studies in Communication Systems. Recently he has joined to the Institut Eurecom in Sophia Antipolis, France where he is working towards a PhD in the networking area.

His research interests are reliable multicast transmission, caching schemes for the Web and push technologies.

Keith W. Ross

Keith W. Ross (ross@eurecom.fr) is a Professor in the Multimedia Communications Department at Institute Eurecom, in Sophia Antipolis, France. From 1985 to December 1997, Keith Ross was with the University of Pennsylvania as Assistant, Associate, and Full Professor. Keith Ross received his BS from Tufts University (1979) in Electrical Engineering, his MS from Columbia University (1981) in Electrical Engineering , and his Ph.D. from the University of Michigan (1985) in Computer, Information and Control Engineering. He has published over 40 papers in leading journals and has published a book on multiservice loss models for broadband telecommunication networks. He is currently writing a multimedia textbook on Internet protocols and data networks. He is or has been on the following editorial boards: Queuing Systems, Theory and Applications; Probability in the Engineering and Information Sciences; Operations Research; Telecommunications Systems; and IEEE Transactions on Automatic Control. He was the Program Chairman of the 1995

INFORMS Telecommunications Conference.

Ernst W. Biersack

Ernst Biersack (erbi@eurecom.fr) received his M.S. and Ph.D. degrees in Computer Science from the Technische Universität München, Munich, Germany.

From 1983 to 1989 he was a research scientist at the Technische Universität München. From March 1989 to February 1992 he was a Member of Technical Staff with the Computer Communications Research District of Bell Communications Research.

Since March 1992 he has been an Associate Professor in Telecommunications at Institut Eurecom, in Sophia Antipolis, France. His current research is on "Architectures for a Scalable High-Performance Video Servers", "Reliable Multicast Transmission", and "Web Caching". For his work on synchronization in video servers he received the Outstanding Paper Award of the IEEE Conference on Multimedia Computing & Systems 1996.

He currently serves on the editorial board of IEEE Network Magazine, ACM Multimedia Systems, and ACM/IEEE Transactions on Networking. Mr. Biersack is a member of the IEEE and ACM.

References

[1] K. V. Almeroth, M. H. Ammar, and Z. Fei, "Scalable Delivery of Web Pages Using Cyclic Best-Effort (UDP) Multicast", In *Proceedings of IEEE INFOCOM*, San Francisco, CA, USA, March 1998.

[2] M. Baentsch, L. Baum, G. Molter, S. Rothkugel, and P. Sturm, "World Wide Web Caching: The Application-Level View of the Internet", *IEEE Communications Magazine*, pp. 170–178, June 1997.

[3] M. Baentsch, G. Molter, and P. Sturm, "Introducing application-level replication and naming into today's Web", *Computer Networks and ISDN Systems*, 28:921–930, 1996.

[4] T. Ballardie, P. Francis, and J. Crowcroft, "Core Based Trees (CBT)", In *Proceedings of SIGCOMM'93*, pp. 85–95, San Francisco, CA, USA, October 1993, ACM.

[5] A. Bestavros et al., "Application-level Document Caching in the Internet", In *Proc. of SDNE'95: The second International Workshop on Services in Distributed and Network Environments*, Whistler, Canada, June 1995.

[6] A. Chankhunthod et al., "A Hierarchical Internet Object Cache", In *Proc. 1996 USENIX Technical Conference*, San Diego, CA, January 1996.

[7] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei, "The PIM Architecture for Wide–Area Multicast Routing", *IEEE/ACM Transactions on Networking*, 4(2):153–162, April 1996.

[8] M. Doar and I. Leslie, "How Bad is Naïve Multicast Routing", In *Proceedings of IEEE INFOCOM'93*, volume 1, pp. 82–89, 1993.

[9] H. Eriksson, "MBONE: The Multicast Backbone", *Communications of the ACM*, 37(8):54–60, August 1994.

[10] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, et al., "RFC 2068: Hypertext Transfer Protocol — HTTP/1.1", January 1997.

[11] A. Gove, "Stream on: RealNetwork isn't about to make multicast video a mass medium", *The Red Herring*, November 1997.

[12] J. Gwertzman and M. Seltzer, "The Case for Geographical Push Caching", In *Proceedings of the Fifth Annual Workshop on Hot Operating Systems*, pp. 51–55, Orcas Island, AW, May 1995.

[13] V. Kumar, "The MBONE FAQ", Collection of Information about MBONE, January 1997.

[14] C. Liu and P. Cao, "Maintaining Strong Cache Consistency in the World-Wide Web", In *Proceedings of ICDCS*, may 1997.

[15] A. Luotonen, *Web proxy servers*, Prentice-Hall, 1998.

[16] R. Malpani, J. Lorch, and D. Berger, “Making World Wide Web Caching Servers Cooperate”, In *Fourth International WWW Conference*, Boston, Dec 1995.

[17] V. N. Padmanabhan and J. Mogul, “Improving HTTP Latency”, In *Second World Wide Web Conference ’94: Mosaic and the Web*, pp. 995–1005, October 1994.

[18] D. Povey and J. Harrison, “A Distributed Internet Cache”, In *Proceedings of the 20th Australian Computer Science Conference*, Sydney, Australia, February 1997.

[19] P. Rodriguez and E. Biersack, “Continuous Multicast Distribution of Web Documents over the Internet”, *IEEE Network Magazine*, 12, 2:18–31, Mar-Apr 1998.

[20] K. W. Ross, “Hash-Routing for Collections of Shared Web Caches”, *IEEE Network Magazine*, 11, 7:37–44, Nov-Dec 1997.

[21] A. Rousskov, “On Performance of Caching Proxies”, , NoDak, feb 1998.

[22] N. G. Smith, “The UK national Web cache - The state of the art”, *Computer Networks and ISDN Systems*, 28:1407–1414, 1996.

[23] R. Tewari, M. Dahlin, H. M. Vin, and J. S. Kay, “Beyond Hierarchies: Design Considerations for Distributed Caching on the Internet”, , The Univertisy of Texas at Austin, feb 1998.

[24] V. Valloppillil and K. Ross, “Cache Array Routing Protocol v1.1”, February 1998, Internet Draft, http://ds1.internic.net/internet-drafts/draft-vinod-carp-v1-03.txt.

[25] D. Wessels, “Squid Internet Object Cache: http://www.nlanr.net/Squid/”, 1996.

[26] D. Wessels and K. Claffy, “Application of Internet Cache Protocol (ICP), version 2”, Internet Draft:draft-wessels-icp-v2-appl-00. Work in Progress., Internet Engineering Task Force, May 1997.

[27] D. Wessels and K. Claffy, “Internet Cache Protocol Version 2”, March 1997, Internet Draft, http://ds.internic.net/internet-drafts/draft-wessels-icp-v2-00.txt.

[28] Yeager and McGrath, *Web Server Technology*, Morgan Kaufman Publishers, San Francisco, 1996.